Université de Technologie de Compiègne – UTC
HEUDIASYC – Heuristique et Diagnostic des Systèmes Complexes

An Introduction to Cognitive Agents and its Aspects to Permit Understanding How to Construct Personal Assistant Agents with Spoken Dialog Capabilities

Author: Emerson Cabrera Paraiso
Email: emerson.paraiso@hds.utc.fr

Compiègne - 2002

Abstract

The use of agents is spread in several domains. The application vary from personalized information management, to management of complex commercial and industrial processes. Several commercial products already use agent technology, and many more applications are under development.

It is also imperative that many researchers use the meaning of agents without correctness or justification.

This document presents an overview on agents and correlates areas needed to understand how to construct assistant-agents that have vocal capabilities.

A main assumption is that the reader is familiar with Artificial Intelligence. Where appropriated, references to more detailed treatments are provided.

This document contains personal impressions and definitions about the subjects that are presented.

At the beginning, we clarify the agent research area and prepare the reader to understand how to design Conversational Agents. To do so, we discuss the meaning of an agent, the Multi-agent systems, the way that we can handle and manage knowledge, how to improve the agent interfaces (Human-Computer Interface and Dialog systems) and some agent applications, such as Personal Assistant Agents.

Keywords

Agents, Multi-agent Systems, Personal Assistant Agents, Human-Computer Interface

Contents

1. Introduction

Recent years have seen a marked increase of interest in agent-oriented technology, in several areas of computer science, including both software engineering and artificial intelligence. Agents are being used, for applications as diverse as personalized information management, electronic commerce, interface design, computer games, and management of complex commercial and industrial processes. Several deployed systems already use agent technology, and many more commercial and industrial applications are under development.

It is also imperative that many researchers use the meaning of agents without correctness or justification.

A non-complete list, shown below, extracted from the *Call for Papers* of "The Second International Joint Conference on Autonomous Agents and Multi Agent Systems – AAMAS 2003", presents some interests research topics:

* action selection and planning
* adaptation and learning
* agent architectures
* agent-based software engineering
* agent communication languages
* artificial market systems and electronic commerce
* autonomous robots
* believability
* communication, collaboration, and interaction of humans and agents
* coordinating perception, thought, and action
* designing agent systems
* expert assistants
* evolution of agents
* foundational issues
* games and agents
* human-like qualities of synthetic agents
* information agents
* instruct ability
* integration and coordination of multiple activities
* knowledge acquisition and management
* integration and coordination of multiple activities
* knowledge acquisition and management
* logics for agents
* lessons learned from deployed agents
* lifelike qualities
* meta-modeling and meta-reasoning
* middle-agents (e.g., matchmakers, brokers, routers)
* mobile agents
* modeling the behavior of other agents
* models of emotion, motivation, or personality
* multi-agent teams
* multi-agent communication, coordination, and collaboration
* multi-agent simulation, verification, and validation
* network agents

* ontologies
* organization of agent societies
* privacy and agents
* real-time performance
* standards for agents
* synthetic agents
* system support for the implementation of agents
* virtual markets
* user modeling

This document presents an overview on agents and correlates areas need to understand how to construct assistant-agents that have vocal capabilities.

A main assumption is that the reader is familiar with Artificial Intelligence (AI). Where appropriated, references to more detailed treatments are provided.

This document contains personal impressions and definitions about the subjects on it presented.

As the *agent* become so popular it is almost impossible to cover all related domains. Thus, this document contains information on the following topics:

- Multi-Agent Systems;
- Human-Machine Interface ;
- Dialog Systems;
- Ontology;
- Knowledge Management.
- Personal Assistant Agents.

The list, not so restricted, matches the research areas of interest for our work at Université de Techonologie de Compiègne (UTC), which addresses the construction of agents with vocal interface.

After reading this document a reader should be able to answer the following questions:

What is an Intelligent Agent?

How are agents structured internally?

How do agents represent their knowledge?

How does each agent model the world?

What communication protocols do agents use?

How do agents coordinate their actions?

How do agents help on knowledge Management?

How can an agent interact with humans?

Let us start with, a chapter describing what an Agent is.

2. Agent

Researches involved in agent development have offered a variety of definitions to the word "agent". In (Russel & Norvig 95), *Russel* defines a generic agent as: “An agent is anything that can be viewed as perceiving its environment through sensors and acting upon that environment through effectors.”. Another definition extracted from (Wooldridge & Jennings 95) states: "... a hardware or (more usually) software-based computer system that enjoys the following properties:

- autonomy: agents operate without the direct intervention of humans or others, and have some kind of control over their actions and internal state;
- social ability: agents interact with other agents (and possibly humans) via some kind of agent-communication language;
- reactivity: agents perceive their environment, (which may be the physical world, a user via a graphical user interface, a collection of other agents, the Internet, or perhaps all of these combined), and respond in a timely fashion to changes that occur in it;
- pro-activeness: agents do not simply act in response to their environment, they are able to exhibit goal-directed behavior by taking the initiative.”

In this document an agent is defined as follow:

“***A computational entity that has a certain capacity of intelligence, that runs independently (in parallel) and executes actions that can be influenced by interactions with other agents.*** ”

2.1. Properties

In (Stan & Graesser 96) a table is presented (see Table 1) that lists several ***desired*** properties of an agent.

Property	Other Names	Meaning
reactive	(sensing and acting)	responds in a timely fashion to changes in the environment
autonomous		exercises control over its own actions
Goal-oriented	pro-active purposeful	does not simply act in response to the environment
temporally continuous		is a continuously running process
communicative	socially able	communicates with other agents, perhaps including people
flexible		Actions are not scripted
Learning	adaptive	changes its behavior based on its previous experience
Mobile		able to transport itself from one machine to another
character		believable "personality" and emotional state.

Table 1: agent properties.

This is a set of *desired* properties. Whenever an entity is ***reactive, autonomous, and goal-driven*** it can be distinguished as an *Agent*. ***Temporally Continuous, communicative, learning and flexible*** are also very desirable features. Of course, other properties could be mandatory for some applications.

2.2. Capacities and Behavior

Concerning agents capacities and behavior, we have (Shmeil 99):

(i) cognition/reaction:

a. cognitive agent: are agents that possess knowledge of their environment; use sophisticated knowledge representations, have expertise, goals and plans. In addition to their traditional mechanisms they must interact and engage in cooperation with other agents (Shen et al. 01);

b. reactive agents: are agents that react promptly to external calls. This type of agent, in general, is associated with a lack or a small capacity of reasoning.

(ii) autonomy: means that an agent has its control implemented internally, i.e., it is self-controlled.

(iii) sociability: is the capacity that allows an agent to know about others and to use this knowledge in its inferences.

2.3. Belief, Desire and Intention

Most intelligent agent technologies have an architecture based on the Belief, Desire and Intention (BDI) approach. Intelligent agents using the BDI model have a set of:

- beliefs representing what the agent "knows";
- desires/goals representing what the agent is trying to achieve;
- intentions to achieve the agent's current goals;
- plans that are combinations of actions that achieve certain outcomes or respond to events and are used by the agent to further its intentions.

When an event occurs, the agent:

- looks for relevant plans (plans that answer this type of event);
- then, for each relevant plan, the agent examines its appropriateness to the situation the agent finds itself in;
- then, the agent selects and starts executing the most appropriate plan it founds.

Additionally, the agent performs ongoing reasoning functions to decide:

- what goal to pursue or alternatively what event to react to;
- how to pursue the desired goal;
- when to suspend/abandon the goal, or change to another goal.

The agent may also vary its balance between reactive and deliberative behavior by changing the amount of time allowed for deciding what to do next. This enables the agent to be more or less sensitive to changes in the environment, that is, be more or less "committed" to its current plan.

2.4. Objects X Agents

Agents and objects share many characteristics, but they are not the same. Agent-oriented programming (AOP) could be considered a specialization of the object-oriented programming (OOP) paradigm (here the key word is programming, I also agree that an object is not an agent). OOP views systems as consisting of objects communicating with one another to perform internal computations, whereas AOP specializes this view to have agents (instead of objects), whose internal computations are based on beliefs, capabilities, and choices, that communicate with each other using messages (Flores-Mendez 99).

The differences between AOP e OOP can be summarized in three great distinctions:

- the degree to which agents and objects are autonomous. We thus do not think of agents as invoking methods upon one-another, but rather as requesting actions to be performed. In the object-oriented case, the decision lies with the object that invokes the method. In the agent case, the decision lies with the agent that receives the request;
- the notion of flexible (reactive, pro-active, social) autonomous behavior. In this sense, an agent is more flexible;
- agents are each considered to have their own thread of control, in the standard object model, there is a single thread of control in the system.

In (Shen et al. 01), a comprehensive distinction between agents and objects is given.

2.5. Communication and Interaction

To allow agents to interoperate, communication languages have been designed. *Knowledge Query and Manipulation Language* (KQML) is the most frequently used (Finin et al. 94). KQML provides a set of performatives based on speech acts. For helping agents to express a message content more efficiently, Ontology has been used successfully. Find following a specific chapter on Ontology.

As long as many platforms, languages and systems have been developed, standards are necessary. FIPA is an organization that is developing standards for software agents to allow heterogeneous agent systems to interact. The promotion of technologies and interoperability specifications that facilitate the end-to-end work of intelligent agent systems in modern commercial and industrial settings, is the mission of FIPA.

2.6. Further Information

Following a list of good places to visit:

http://www.agentcities.org/
http://agents.umbc.edu/
http://www.csc.liv.ac.uk/~mjw/links/
http://www.cse.sc.edu/%7Ehuhns/#pub
http://www.daml.org/
http://www.aaai.org/AITopics/newstopics/agents.html
http://www.cs.washington.edu/research/projects/WebWare1/www/softbots/softbots.html
http://www.agentlink.org/
http://www.agentbase.com/survey.html
http://www.msci.memphis.edu/~franklin/AgentProg.html#def
http://citeseer.nj.nec.com/491166.html
http://www.intelligent-agents.com/
http://www.intelligent-agents.com/8How_to_develop_an_agent/Development_tools/
http://www-2.cs.cmu.edu/~sycara/

After having defined an agent we return to our selected list of interest:
- Multi-Agent Systems;
- Dialog Systems;
- Ontologies;
- Personal Assistant Agents;
- Human-Machine Interface;
- Knowledge Management.

Now is a good time to draw figures that can link all these subjects together. Thus, the following figures establish a connection among them. First, we need a representation for an agent: lets use the proposed generic structure found in (Ramos 00), and shown on Figure 1.

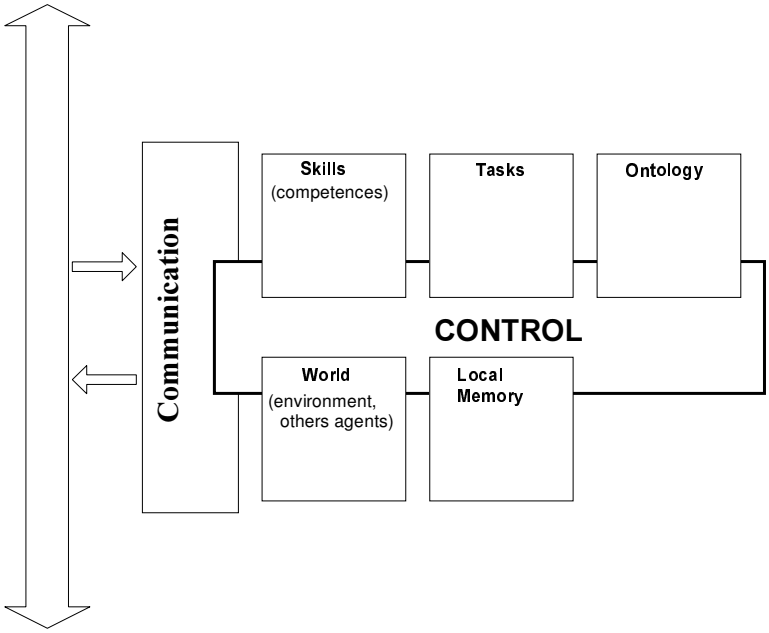

Figure 1: a generic agent representation.

Let us describe each component briefly:

- Communication: responsible for sending and receiving messages to or from the environment;
- World: contains an internal representation of other agents and of the environment;
- Skills: contains the set of services performed by an agent. A particular skill can be expressed as procedures or as rules. If the skill is complex, then a plan for executing it can be created, and the derived tasks are spawned and broadcast on the network;
- Tasks: is a representation of what the agent is currently executing for internal control purposes;
- Ontology: specification of a vocabulary; or a taxonomy (domain), used to interpret messages;
- Local Memory: stores the memory, goals and skill descriptions;
- Control: it controls the different modules. It may be an inference machine.

Since this is a "minimum" structure, we can attach modules to produce actions for achieving a specific goal. For instance, we might attach a module responsible for "tracking" the user, during her work, to produce a Personal Assistant Agent. This can be seen on Figure 2, proposed by (Ramos 00).

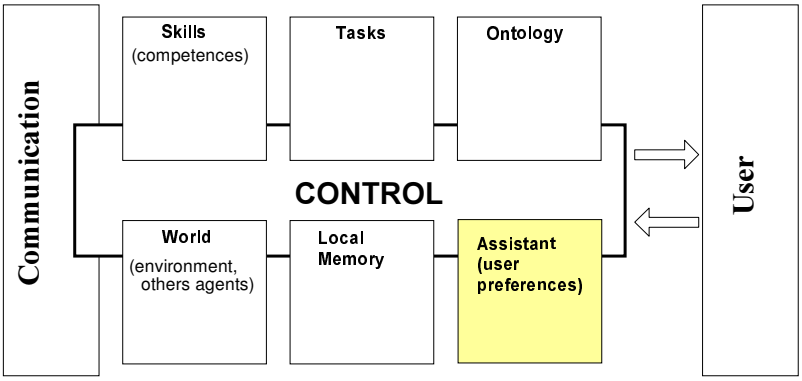

Figure 2: a generic agent + assistant module.

Following this simple principle we can produce others agents and maybe, have a Multi-agent system as shown on Figure 3. Thus, that list of interested points (Multi-Agent Systems, Dialog Systems, Ontologies, Assistant Agents, Human-Machine Interface and Knowledge Management) that we selected before may be found in a single system.

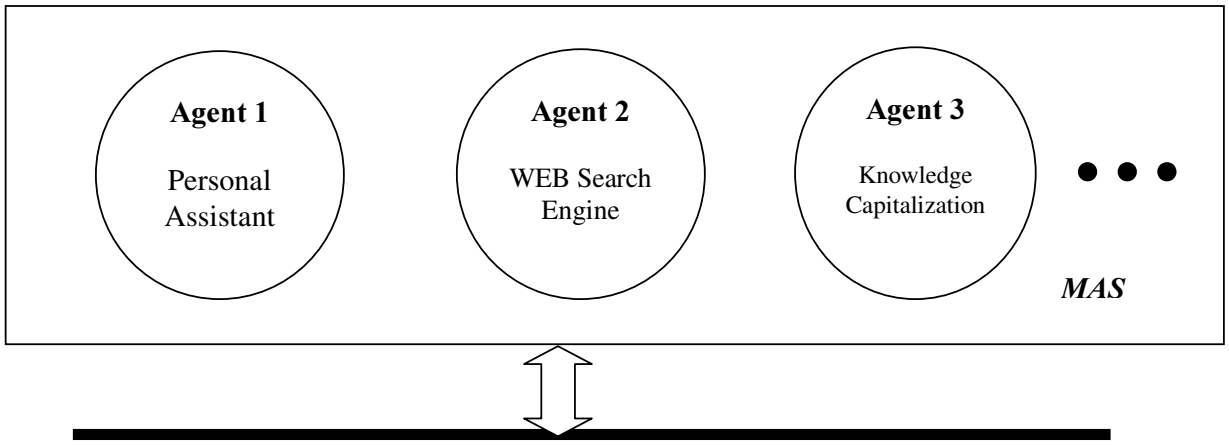

Figure 3: a simple multi-agent system representation.

Finally, the following pages, are structured compilation of information concerning the list of selected points. Each part contains:

- a definition;
- possible applications;
- a list of WEB sites, next events (congress, seminars, etc) and related books.

3. Multi-Agent Systems

A vast area of research in computing is the development of toolkits that allow the computational systems to interact for improving of their results. It is understood that two or more systems working together can cooperate and obtain better results than working separately.

Agent-Based software engineering proposes to facilitate the software creation enabling them to interoperate. AI joins to propose solutions. Classically, AI provides solutions to complex problems that "conventional" systems cannot do themselves. If several parts of a complex system become intelligent, then they will perform new tasks that a centralized one could not perform. The great challenge in such systems is to manage a large system with several processes in action, and to allow them to exchange their results and experiences to improve the quality (Paraiso 96).

As cited by (Sycara 98), the most powerful tools for handling complexity are modularity and abstraction. A Multi-Agent Systems (MAS) is a modular organization formed by agents that are specialized at solving a particular problem. This decomposition allows each agent to use the most appropriate paradigm for solving a particular problem.

3.1. Definitions

Multi-agent systems are computational systems in which several agents cooperate to achieve some task, which might not, or not as easily, otherwise be achieved.

Sycara also listed four characteristics for a MAS:

1) each agent has incomplete information or capabilities for solving the problem has a whole;
2) there is no global control:
3) data are decentralized;
4) computation is asynchronous.

In an MAS, agents are independent, they share an environment, may compete for the same resources, such as time, space, tools, machines; and cooperate. Even if they do not have the same language, they have the capability of formatting and transmitting their needs using a common and well known structure of representation.

Due to efficiency, an agent, in a MAS, has two main goals:
b) to do its tasks and;
c) to support other agents, whenever possible.

As a consequence, agents in a MAS need to coordinate their activities and to cooperate, in order to avoid duplication of effort (d'Inverno & Luck 01). Also, an agent will need a communication protocol and a knowledge information language.

3.2. Cooperation in a MAS

An agent-based system needs more than a protocol or a communication language: the agents need motivation for cooperation. The level of abstraction or considerations to be taken is closely connected to the system's purpose. To exemplify, (Genesereth 94) listed three desired behaviors of

an agent in a community of agents: veracity, autonomy and commitment. These three principles can define a cooperation policy. In addition, the agents need to be organized to enhance collaboration.

Two different approaches have been explored: *Task Sharing* and *Result Sharing*. In *Task Sharing*, the conversation begins when an agent has a task that it is incapable to solve by itself. *Result Sharing* is used by agents that share their results with the community. See (Paraiso 96) for further details.

3.3. Challenges on MAS Development

The design and implementation of a MAS is not an easy task. Many researchers have contributed to help MAS engineers. Let us list some challenges and problems, posed by (Bond & Gasser 1998):

1) how to formulate, describe, decompose, and allocate problems and synthesize results among a group of intelligent systems?
2) How to enable agents to communicate and interact? What communication languages and protocols to use? How can heterogeneous agents interoperate? What and when can they communicate?
3) How to ensure that agents act coherently in making decisions or taking actions? How to avoid unstable system behavior?
4) How to enable individual agents to represent and reason about actions, plans, and knowledge of other agents to coordinate with them?

Solutions to such problems are proposed by various researchers. It is not the purpose of this document to describe them.

3.4. Applications

The range of MAS applications vary from manufacturing to process control, air-traffic control, or information management.

In the field of process control, **ARCHON** was a successfully MAS prototype (Wittig 92). ARCHON stands for *Architecture for Cooperative Multi-Agent Systems*. ARCHON was applied to several process-control applications, including electricity transportation management.

OASIS is an agent-based air-traffic control system (Ljunberg & Lucas 92). In this system, agents are used to represent both air-craft and the various air-traffic control systems in operation.

Multi-Agent Based Financial Investment Adviser is a MAS for helping investors. To this end, this adviser consists of four principal components (Zhang 00):

- information gathering agents that are responsible for gathering relevant information on the Internet;
- data mining agents that are in charge of discovering knowledge from retrieved information;
- group decision making agents that can effectively use available knowledge and appropriate information to make reasonable decisions, and;
- a graphical user interface.

GIR is an Intelligent Management of Computer Networks system, which has the goal of monitoring and managing computer networks, relying on the use of Artificial Intelligence techniques (Britto et al. 98). The main techniques employed by GIR involve the application of Multi-Agent Systems, Machine Learning, Data-Mining, and Expert Systems.

The work is based on the following features:
(a) the use of distributed agents for the intelligent search of information in the network, supplying it in a more abstract way, adapted to the decision-making task;
(b) the use of machine learning and data-mining techniques that, starting from the log files that register previous problems and their solution, allow the use of experience thus obtained in the solution new problems; and
(c) the use of heuristics and conduction rules supplied by experts, through a decision support system, as an advisor element to network operators.

3.5. OMAS Platform

There are platforms specially design to implement agents and MAS. OMAS (Open Multi-Agents System) is a platform developed at UTC - Université de Technologie de Compiègne (Barthès 00). An OMAS agent has a rather complex structure, implemented as sub-objects. In OMAS, agents have a private Agent Communication Language (ACL) similar to KQML.

In OMAS, agents are totally independent but belong to a cluster called a "coterie." Communications occur thanks to several protocols (basic and a modified version of Contract-Net). The protocol is specified at the message level and protocols can be mixed. Messages are sent in three modes: point-to-point, multicast, or broadcast. Agents are cloned from a generic agent and given skills. Agents can dynamically enter or leave the group.

3.6. Further Information

Following a list of good places to visit:

http://www.aaai.org/AITopics/html/multi.html
http://www.multiagent.com/
http://www.acm.org/crossroads/xrds5-4/multiagent.html
http://www.limsi.fr/Individu/guichard/biblioSMA.html

The University of Westminster is offering a graduate curse on MAS: *M.Sc. in Intelligent and Multi-Agent Systems*. Information available at: http://www2.wmin.ac.uk/oreillyp/imas.html.

At http://www.multiagent.com/ is available a list of commercial companies that use AI approaches to deploy commercial products.

4. Human-Machine Interface

4.1. Definitions

Human-computer interface (HCI) is a discipline concerned with the design, evaluation and implementation of interactive computing systems for human use. The field of HCI covers all aspects of how people communicate and interact with computer systems. It encompasses many disciplines including engineering, computer science, social science, human factors and psychology.

Human-computer interaction is concerned with the joint performance of tasks by humans and machines; the structure of communication between human and machine; human capabilities to use machines; algorithms and programming of the interface itself; the process of specification, design, and implementation of interfaces. Human-computer interaction thus has science, engineering, and design aspects (Hewett et al. 97).

4.2. Usability

The aim of HCI is to improve the software usability. It is common to find software that satisfies all technical specifications but is not user-friendly.

Usability is the central point. Usability brings to:

- Improved productivity: perform a task in a faster and efficient way;
- Fault minimization: correct treatment of errors;
- Ease of learning: require a few formal training;
- Ease of memorizing: facilitate the memorization of commands;
- Improved user satisfaction: confidence and security.

4.3. HCI X AI

AI joins HCI to build powerful systems. Many contributions from AI researchers increases HCI design and implementation phases. Studies on Automatic Voice Treatment (analysis, synthesis, etc), Natural Language Processing (analysis, comprehension and formulation), Image Processing, Computer Vision, etc, are some examples.

Another popular area of research is *Dialog*. Dialog Systems are systems in which human users speak or type to a computer in natural language. In this area AI (and specially Agents) is providing successful contributions.

The developing of Web based tools for intelligent learning and automated learning have been the focus of many groups.

4.4. Further Information

Following a list of good places to visit:

http://www.acm.org/sigchi/
http://sigchi.org/cdg/
http://www.ic.unicamp.br/~hans/xchart/main/

http://sigchi.org/cdg/cdg2.html
http://www.ime.usp.br/~hitoshi/mac446/
http://www.limsi.fr/Individu/guichard/

5. Dialog Systems

Dialog systems are becoming an area of increasing interest, both in research and in practical applications. Dialog systems are among the few concrete examples of goal-oriented interaction between humans and computers, and are therefore an exciting test bed for artificial intelligence research.

5.1. Definitions

Dialog systems are systems in which human users speak or type to a computer using natural language, in order to achieve their goals.

A dialog consist of a *question-réponse-évaluation* interaction. In the example below, a fragment of a dialog illustrates some of the power of this sort of systems. A human (USR) interacts with a Dialog Systems (SYS). Specifically, the user must collaborate with the system to manage responses to emergency calls.

Example

(1) USR: We need to get an injured woman in Senlis to the hospital.
(2) SYS: OK
(3) USR: What vehicles are available?
(4) SYS: There are ambulances in Beauvais and Compiègne.
(5) USR: OK. Use one from Beauvais.
(6) SYS: Do you know that A1 is blocked due to construction?
(7) USR: Oh
(8) USR: Let's use the A80 instead.
(9) SYS: OK. I'll dispatch the crew.

Nowadays, the systems face different technical difficulties concerning the recognition and comprehension of terms; creating reasonable and natural answers and the management of a flexible dialog (Hembecker 02). Most dialog systems are composed of a grammar and a dialog manager, that analyze the inputs to identify the user intentions. The dialog model may be restricted to *questions-answers* (Clark 99), may have *Conversational Interfaces* (Kölzer 99) or may be applied to automated natural spoken systems (Gorin 02). The higher the system complexity, the more complicated is the dialog model.

5.2. Complexity

We can summarize the challenges in building dialog systems to handle tasks in complex domains as follows (Allen et al. 01):

- handling the level of complexity of the language associated with the task;
- integrating a dialog system with a complex "back-end" reasoning system (e.g., a factory scheduler, a map server, an expert system for kitchen design);
- intention recognition: determining what the user is trying to do by saying it;

- enabling mixed-initiative interaction, in which either the system or the user may control the dialog at different times in order to make the interaction most effective.

Task Complexity	Technique Used	Dialog Phenomena Handled	Example of Task
least complex	Finite-state Script	User answers questions	Long-distance dialing
	Frame-based	User asks questions, simple clarifications by system	Getting train arrival and departure information
	Sets of Contexts	Shifts between predetermined Topics	Travel booking agent
	Plan-based Models	Dynamically generated topic structures, collaborative negotiation subdialogs	Kitchen design consultant
most complex	Agent-based Models	Different modalities (e.g., planned world and actual world)	Disaster relief management

Table 2: Dialog and Task Complexity (Allen et al. 01).

The simplest systems are the finite-state systems that follow a script of prompts for the user. This technique works only for the simplest tasks. The frame-based approach includes most of the spoken dialog systems constructed to date. In this approach, the system interprets the speech to acquire enough information in order to perform a specific action. The context is fixed in such systems since they do only one thing. The next level up in complexity involves representing the task by a series of contexts, each represented using the frame-based approach.

The next two levels of complexity have more complicated tasks. In fact, the tasks require the system to maintain an explicit model of the tasks and/or the world and to reason about such models. The language and the dialogs become significantly more complicated. In the plan-based approach, the dialog involves constructing a plan with the user, interactively(e.g., a design for a kitchen, a plan to evacuate personnel of an island).

The highest level of complexity involves agent-based models. These dialogs may still involve planning, but also may involve executing and monitor operations in a dynamically changing world (e.g., emergency rescue coordination).

5.3. A Personal Assistant Agent: Example of Dialog Handling

Personal Assistant Agents are systems that support the human in complex environments. The assistant agent may helps to unload ordinary tasks from the human (see more details in Chapter 7). They can work independently of the human, but might interact with him. In order to achieve interaction, a dialog module can be added to them. In (Ramos 00), an assistant agent model is proposed with dialog skill (see figure 4).

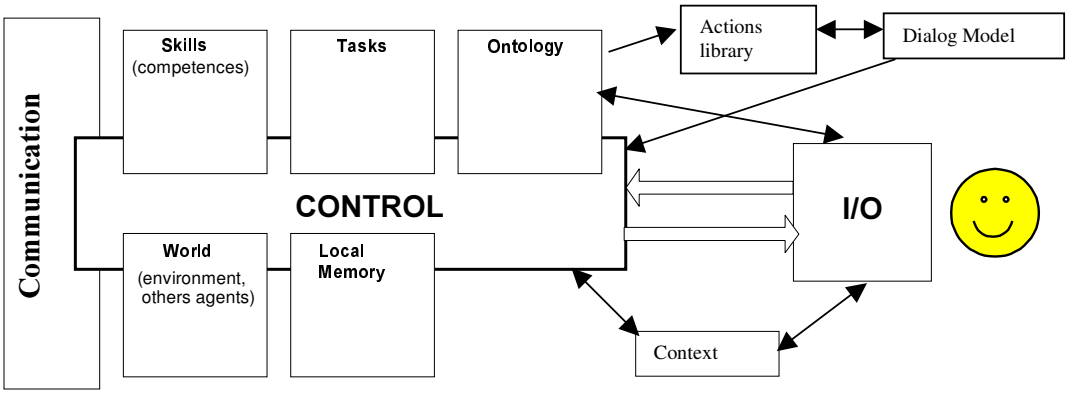

Figure 4: an Assistant Agent that can handle dialogs.

In this structure, a user request is analyzed using an Ontology and a sequence of actions (in this sense a dialog) is generated.

As the dialog is a multiple step process, a *Context* Memory was adapted in order to hold pieces of the conversation. This memory stores the discussion theme and the user intentions.

5.4. Further Information

Following a list of good places to visit:

http://www.cs.pitt.edu/~litman/courses/cs3710/cs3710.html
http://www.dfki.de/etai/SpecialIssues/Dia99/
http://www.inria.fr/rapportsactivite/RA96/dialogue/dialogue.html
http://www.cs.rochester.edu/research/cisd/
http://www.sanpo.t.u-tokyo.ac.jp/~nigel/agents.html
http://www.shlrc.mq.edu.au/masters/811/topics.html
http://issco-www.unige.ch/projects/im2/mdm/
http://fife.speech.cs.cmu.edu/Courses/11716/
http://www.research.att.com/~algor/hmihy/papers.html

6. Ontologies

The word "ontology" seems to generate a lot of controversy in discussions about AI. It has a long history in philosophy, in which it refers to the subject of existence. It is also often confused with epistemology, which is about knowledge and knowing.

6.1. Definitions

An ontology is a specification of a conceptualization (Gruber 93).

In (Gruber 93), we can also find the following statement:

"An ontology is a set of definitions of content-specific knowledge representation primitives: classes, relations, functions, and object constants."

The main purpose of an ontology is enabling knowledge sharing and reuse. The key components that make up an ontology are a *vocabulary* of basic terms and a precise *specification* of what those terms mean.

An Ontology is a powerful way to:

- share common understanding of the structure of information among people or software agents;
- enable reuse of domain knowledge;
- make domain assumptions explicit;
- separate domain knowledge from the operational knowledge;

For the purposes of this document, an **ontology** is a formal explicit description of concepts in a domain of discourse (**classes -** sometimes called **concepts**), properties of each concept describing various features and attributes of the concept (**slots -** sometimes called **roles** or **properties**), and restrictions on slots (**facets -** sometimes called **role restrictions**). An ontology together with a set of individual **instances** of classes constitutes a **knowledge base** (Noy & McGuinness 01).

A class can have **subclasses** that represent concepts that are more specific than the superclass.

6.2. Developing an Ontology

In practical terms, developing an ontology includes:

- defining classes in the ontology;
- arranging the classes in a taxonomic (subclass–superclass) hierarchy;
- defining slots and describing allowed values for these slots;
- filling in the values for slots for instances.

It is necessary to choose a language to represent the ontology. Currently two approaches can be founded: one based on logics and another based on frames.

Resource Description Framework (RDF) is a language to represent an ontology based on frames. RDF provides a data model for representing formal semantics of information (meta-information) using XML as an interchange syntax. The RDF specifications provide a system to support the exchange of knowledge on the Web.

There are several frameworks available for editing and using an ontology. **Protégé-2000** is an ontology editor and a knowledge-base editor. **Protégé-2000** is also an open-source, Java tool that provides an extensible architecture for the creation of customized knowledge-based tools. It is available at: http://smi.stanford.edu/projects/protege.

The simple example of ontology shown below, was implemented using **Protégé-2000**.

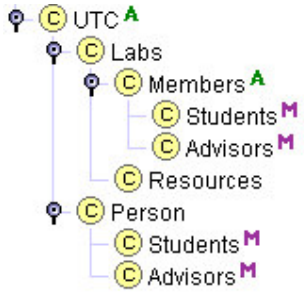

Figure 5: graphical representation of ontology UTC.

The ontology **UTC** has a superclass named UTC. Labs and Person are classes directly linked to UTC. Labs has two sub-classes Members and Resources. Members has two sub-classes: Students and Advisors.

Person is a class that has two sub-classes: Students and Advisors. Thus, Students and Advisors are multiple inherited (Members and Person).

Each class can be defined as Abstract, which means that it does not have any direct slot, and Concrete, which defines classes with direct slots. In this sense, the UTC ontology is composed by:

Class	Type
UTC	Abstract
Labs	Concrete
Members	Abstract
Resources	Concrete
Person	Concrete
Students	Concrete
Advisors	Concrete

Table 3: Classes Type

After defining the class structure, all slots can be defined (see Table 4).

An advantage of using **Protégé-2000** is that a JAVA application can inherit **Protégé-2000** java classes and use them on it. **Protégé-2000** generates the equivalent knowledge base in two different formats: RDF and Lisp classes.

Slot Name	Class where it appears (Direct – D or Inherited – I)	Type	Cardinality
Computers	Resources – D	Integer	0 .. 1
Lab_name	Labs – D Members – I Students – I Advisors – I	String	0 .. 1
Name	Person – D Students – I Advisors – I	String	0 .. 1
Phd_students	Advisors – D	Instance of Students	0 .. 4
Research	Advisors – D	String	0 .. n
Thesis	Students – D	String	0 .. 1

Table 4: ontology UTC: Classes X Slots.

Building an ontology appears to be similar to building an object-oriented program, but there are profound differences:

» Classes and objects in a program represent data structures;
» Classes and objects in ontologies represent the world.

There is often a correspondence between data structures in programs and definitions in an ontology. Classes in an ontology must reflect the structure of the world, not the structure of the data.

6.3. Further Information

Following a list of good places to visit:

http://www.ontoweb.org/
http://www.ontology.org/index.html
http://www.cse.sc.edu/research/cit/
http://www.ime.usp.br/~eudenia/iia/onto/index.html#mainref
http://www-ksl.stanford.edu/kst/what-is-an-ontology.html
http://www.daml.org/ontologies/
http://www.ksl.stanford.edu/software/ontolingua/

7. Personal Assistant Agents

7.1. Definition

Personal Assistant Agents (PAA) are systems that support humans in their work based on computers. The agent helps to unload more mundane tasks from the human. In this context, the agent, or agents, show autonomy, while they work with the user cooperatively.

Agents can assist users in a range of different ways (Maes 97):

- they hide the complexity of difficult tasks;
- they perform tasks on the user's behalf;
- they can train or teach the user;
- they help different users collaborate;
- they monitor events and procedures.

The assistant becomes gradually more effective as it learns the user's interests and preferences. The way that an agent can learn from or about the user can be one of the following:

- it observes and imitates the user's behavior;
- it adapts based on user feedback;
- it can be trained by the user on the basis of examples;
- it can ask for advice from other agents assisting others users.

7.2. User X PAA

The interaction between the user (many times referenced as a *master*) and the PAA can be done in different manners. In general, it depends on the adopted communication approach and, of course, on the application domain and architecture. The use of natural dialogs, or even spoken dialog, might facilitate this effort.

Another important aspect to be considered is the PAA model of relationship user-PAA. In (Ramos 00), three models of relationship are presented: isolated model, distributed model and integrated model.

7.3. OMASWA Personal Assistant Agent

OMASWA (Open Multi-Agent Systems with Assistants) is an MAS that integrates human agents via personal assistants (Ramos 00). The OMASWA platform is the specialization of OMAS platform that was briefly introduced in Chapter 3. In OMASWA an PAA has the structured presented on Figure 6. To a generic agent (presented in Chapter 2), Ramos added two new components to implement an PAA: *Master Model* and *User Interface*.

Master Model contains the user model known by the PAA. The *User Interface* component is the interface with the user. It can handle dialogs as well.

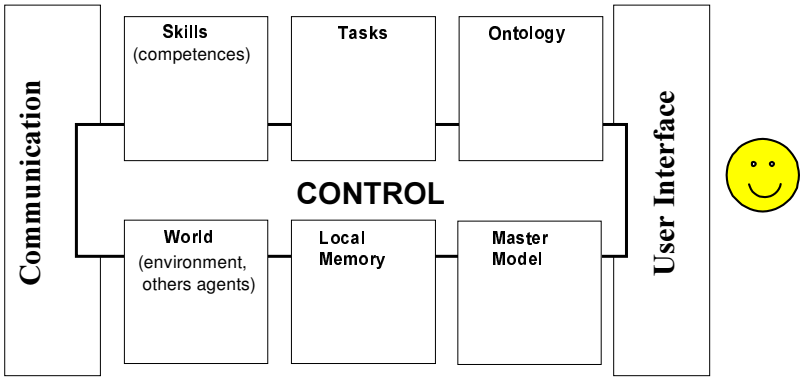

Figure 6: An OMASWA Personal Assistant.

7.4. Applications

The set of tasks or applications with which an agent can assist the user is virtually unlimited: information filtering, information retrieval, mail management, meeting scheduling, etc. Following are some developed applications.

LETIZIA is an autonomous interface agent that helps the user during WEB search activities. It learns the user interests and anticipates WEB pages of possible interest. As the user operates a conventional Web browser, the agent tracks user behavior and attempts to anticipate items of interest by doing concurrent, autonomous exploration of links from the user's current position. The agent automates a browsing strategy consisting of a best-first search augmented by heuristics inferring user interest from browsing behavior (Lieberman 95).

WebMate is a personal browsing and searching agent. It accompanies the user when he/she uses the Internet, and providing information it gathers based on the user profile, which is created as the user browses the Internet and use WebMate. The main WebMate features are:

- Searching enhancement, (which includes parallel search, searching keywords refinement and relevant feedback), allows WebMate to send search requests to search engines, get results from these requests, and reorder them according to the degree of overlapping which occurs among the different search engines.
- Browsing assistant performs the following tasks: it learns the user current interests, recommends new URLs to the user according to his profile and selected resources. It also allows the user to rename URLs with aliases, or to monitor bookmarks, to find other pages like the current page that is browsing, to send the current browsing page to others people, and to pre-fetch the hyperlinks of the page that he is browsing.
- Offline browsing allows the user to download pages for offline browsing, get the references of pages, and be able to print the page complete with the URL references intact.
- Filtering HTTP header will record the http header and all the transactions between the user browser and the WWW servers, filter the incoming cookies to protect the user privacy.
- WebMate can dynamically set up many different resources, including search engines, on-line dictionaries, and on-line translation systems. Also, WebMate is programmed in Java, which makes it portable to all operating systems, and able to multi-thread.

Another work, developed at UTC, involves the use of PAA to learn the user's model in a MAS for Collaborative Filtering application (Enembreck 02). PAAs offer an interface to the user where he can store documents. PAAs are integrated to a collaborative tool used as a document database. The user tells to the PAA what are the most interesting documents and when the assistant must update his profile based on such documents. In order to model the user document classification and information retrieval techniques are used. The idea is to learn (create) a model representing a collection of documents, to organize, retrieve and propose other documents according to their similarity of content using the model.

7.5. Further Information

Following a list of good places to visit:

http://lieber.www.media.mit.edu/people/lieber/Lieberary/Letizia/Letizia.html
http://www.ri.cmu.edu/people/sycara_katia.html

8. Knowledge Management: Agent Approach

Knowledge Management (KM) is an important field of application for AI and related techniques. The task is not so easy, since it deals with logically and physically dispersed actors and knowledge sources, different degrees of formalization of knowledge, different kinds of (web-based) services and (legacy) systems, conflicts between local (individual) and global (group or organizational) goals.

8.1. Definition

KM is the process through which organizations generate value from their intellectual and knowledge-based assets. Most often, generating value from such assets involves sharing them among employees, departments and even with other companies in an effort to devise best practices. An approach to KM can result in improved efficiency, higher productivity and increased revenues in practically any business function.

8.2. Agent Approach

Agent approaches have already been successfully employed for many partial solutions within the overall picture: Agent-based workflow, cooperative information gathering, intelligent information integration, or personal information agents are established techniques in this area.

To exemplify, lets take project under development at UTC.

8.3. MAS for KM Systems

The aim of this work is to build a multi-agent architecture for KM systems in research and development (R&D) projects (Tacla & Barthès 02). R&D teams have no time to organize project information nor to articulate the rationale behind the actions that generate the information. The main goal is to provide a system, that supports collaborative work and helps to capture and to organize experiences without overloading the team members with extra-work.

Some general requirements for a KM system that guide the project:

- it must cover as much of the R&D activity as possible;
- it must save time by helping the user in the day-to-day activities;
- it must support users in creating and sharing knowledge;
- it must be reliable, secure and persistent.

The basic idea is to consider that knowledge is contained in the information produced, retrieved and exchanged among members of the project and to try to organize the information automatically and locally for each team member. The result is a distributed group memory where each member keeps part of the knowledge of the team. Thus, the first reason to employ MAS in KM systems is that, like in a team, an MAS is composed by a group of possibly heterogeneous and autonomous agents that share a common goal and work cooperatively to achieve it.

Initially there are two types of agents: *Service agents*, that provide a particular type of service corresponding to specific skills, and, *Personal assistants* in charge of interfacing humans to

the system. Their particular skills are devoted to understanding their master and presenting the information intelligently and in a timely manner.

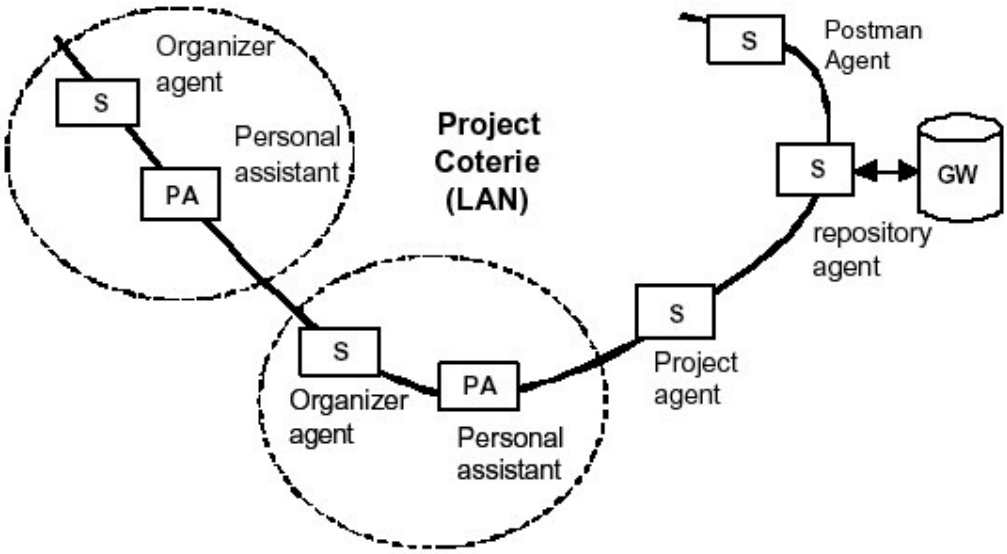

Figure 7: The multi-agent architecture for KM systems.

The Figure 7 shows the agents that compound the architecture for the KM systems.

In this architecture, the PAs play a major role in the KM system. Firstly, they are in charge of all exchanges of information among team members. Secondly, a PA 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. So the PAs must capture and represent the team members' operations helping them in the process of preserving and creating knowledge

There are several *Service agents*. A *Service agent*, called *Project agent*, holds all the values shared by the group, among them the ontologies (the user preferences are related to three ontologies: the domain ontology, containing the main concepts of the domain of the project; the documentation ontology, holding the concepts concerning the documentation domain (e.g. categories of documents); and the project tasks ontology (e.g. prototype construction)). The team members can only extend these ontologies (i.e. add a new child concept). In this way, the 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 he integrates the project is asking for the ontologies.

Each PA is supposed to help his/her master to organize his/her documentation. In order to do that, each one of them works jointly with its *Organizer agent*, a *Service agent* able to build representations of the documents allowing for late retrieval.

As the architecture should be independent of any specific software tool, it integrates a *Service agent* called *Repository agent*, it encapsulating the groupware that must be part of the architecture. The referred agent offers services of saving and retrieving documentation but, depending on the tool it encapsulates, it can offer other kinds of service like WEB search, and e-mail management.

8.4. Further Information

Following a list of good places to visit:

http://www.brint.com/km/
http://www.mitre.org/work/knowledge_mgt.shtml
http://www.cikm.org/2002/

9. Personal Assistant Agents and Spoken Dialog Handling

In this document we had presented a non extensive, but comprehensive background to understand how to design cognitive agents to be applied to many situations. At the beginning, our concerns was to clarify the agent research area and to prepare the reader to the topic that begins now: design of Conversational Agents. To do so, we briefly discussed the meaning of an agent, the Multi-agent systems, the way that we can handle and manage knowledge, how to improve the agents interface (HCI and Dialog systems) and some agent applications, such as Personal Assistant Agents.

The concept of Conversational Agents and how it is connected to our interest remained unclear. A Conversational Agent is an interface agent that is capable of developing natural conversations (it can dialog) with the user, using typed natural language statements, or even spoken statements. Thus, if we want to build an PAA with voice handling, it must be a Conversational Agent.

As we already discussed in Chapter 5, to develop a dialog systems is not an easy task. Whenever we add voice handling to the dialog, we introduce an additional complex component. The development of a spoken dialog system requires the integration of the various components of spoken language technology, such as speech recognition, natural language processing, dialog modeling, and speech synthesis.

9.1. Voice Handling

Many applications already have voice capabilities. Simple inquiries about bank balance, movie schedules, and phone call transfers can already be handled by telephone-speech recognizers.

Voice activated data entry is particularly useful in medical or darkroom applications, where hands and eyes are unavailable, or in hands-off or eyes-busy command and control applications. Speech could be used to provide more accessibility for the handicapped (wheelchairs, robotic aids, etc.) and to create high-tech amenities (intelligent houses, cars, etc.), as well. Adding voice as an input option, it introduces a ‘multi-modal’ computer environment, freeing users from dependence on the mouse, keyboard and stylus for many applications. With voice handling, users can enter and access data and commands more rapidly by speaking than, by typing or tapping, and thus can be more productive.

It is important to realize that a speech interface by itself does not solve the problem of Spoken Dialog. To replace the operations of menu selection by using speaking predetermined phrase that performs the equivalent operation, may aggravate the problem, since the user would need to remember a potentially long list of arbitrary commands.

Conversational interfaces, on the other hand, can bring the opportunity for the user to state what he wants to do in his own terms, just as he would do to another person, and the system takes care of the complexity.

9.2. PAA with Voice Capabilities

A possible PAA with voice handling structured is shown in Figure 8. The content of each component must be analyzed from many points of view to assure that this structure is sufficient. The voice handling functions is implemented at the *Conversational User Interface* component.

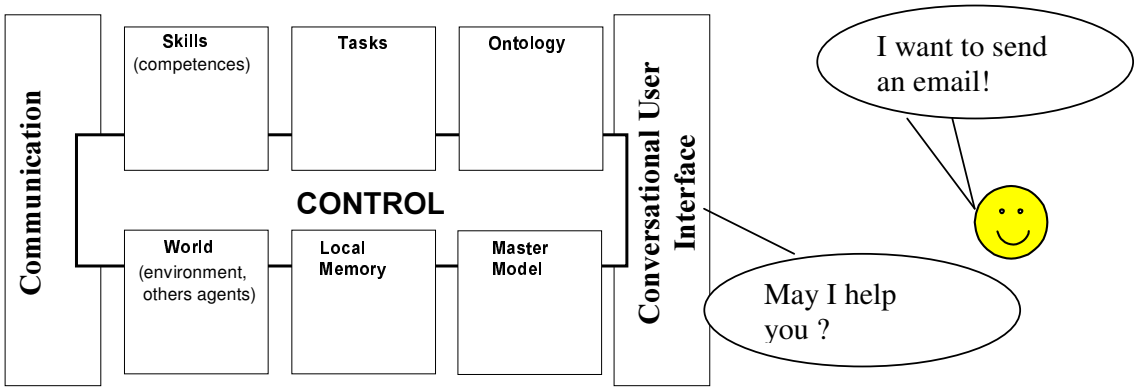

Figure 8: PAA with voice handling.

Many technical aspects need to be fully studied and tested. This is the next step of the work at UTC.

First, we will implement conventional voice oriented applications, such as, a bank account manager. This first study will allow us to face the real aspects on developing voice oriented interfaces. After listing and analyzing the problems, we will address the next steps.

By now, we can anticipate the need of a dialog system and a component to correct and/or to enrich the translation made by the Speech Engine. For that, we can study the use of Ontologies, for instance.

10. Conclusions and Next Steps

Many technical aspects need to be fully studied and tested. We will implement conventional voice oriented application, such as, a bank account manager. This first study will allow us to face the real aspects on developing voice oriented interfaces. After listing and analyzing the problems, we will address the next steps.

By now, we can anticipate the need for a dialog system and a component to correct and/or to enrich the translation made by the Speech Engine. For that, we can study the use of Ontologies, for instance.

Another important issue is to find an application to orient our work. In fact, this real application could add some technical aspects that we did not covered in this document.

It is important to say, as well, that implementation tools, programming language and speech recognizer tools will be necessary and this is the time to start choosing them. In a next document, we will discuss this point.

Finally, we'd like the reader to give us back any suggestions, possible modifications or comments related to this document.

References

(Allen et al. 01) Allen, J. F.; Byron, D. K.; Dzikovska, M.; Ferguson, G.; Galescu, L.; Stent, A. 2001. Towards Conversational Human-Computer Interaction. AI Magazine 2001.

(Barthès 00) Barthès, J-Paul. 2000. *OMAS v 1.0 Technical Reference*, Techical Report nº 144, HEUDIASYC UMR 6599, Université de Technologie de Compiègne.

(Bond & Gasser 1998) Bond, A. H.; Gasser, L. 1998. Readings in Distributed Artificial Intelligence. Morgan Kaufmann.

(Britto et al. 98) Britto, A.; Paraiso, E.; Scalabrin, E.; Kaestner, C.; Avila, B. 1998. Intelligent Management of Computer Networks: The GIR Proposal. XVIII International Conference of the Chilean Computer Science Society - Antofagasta, Chile.

(Clark 99) Clark, P.; Thopson, J.; Porter, B. 1999. A Knowledge-Base Approach to Question-Answering. AAAI 99 - Fall Symposium on Question Answering Systems.

(d'Inverno & Luck 01) d'Inverno, M.; Luck, M. 2001. Understanding Agent Systems. Spring Series on Agent Technology.

(Enembreck 02) Enembreck, F. 2002. Agents for Collaborative Filtering. Internal Rapport at HEUDIASYC – UTC.

(Finin et al. 94) Finin, T.; McKay, D.; Fritzson R.; McEntire R. 1994. The KQML Knowledge Exchange Protocol. Third International Conference on Information and Knowledge Management. National Institute of Standards and Technology, Gaithersburg, Maryland.

(Flores-Mendez 99) Flores-Mendez, R. A. 1999. Towards a Standardization of Multi-Agent System Frameworks. At http://www.acm.org/crossroads/xrds5-4/multiagent.html.

(Genesereth 94) Genesereth, M. R.; Ketchpel, S. P. 1994. Software Agents. Stanford University Logic Group.

(Gorin 02) Gorin, A. L.; Abella, A.; Alonso, T.; Riccardi, G.; Wright, J. H. 2002. Automated Natural Spoken Dialog. Computer-Innovative Technology for Computer Professionals, IEEE.

(Gruber 93) Gruber, T. 1993. A translation approach to portable ontology specification. Knowledge Acquisition 5.

(Hembecker 02) Hembecker, F. 2002. Parser Baseado em Casos na Compreensao de Dialogo: uma Proposta de Aplicacao do Direct Memory Access Parsing. Dissertation at PUCPR. (in Portuguese)

(Hewett et al. 97) Hewett, T.; Baecker, R.; Card, S.; Carey, T.; Gasen, J.; Mantei, M.;Perlman, G.;Strong, G.;Verplank, W. 1997. ACM SIGCHI Curricula for Human-Computer Interaction.

(Kölzer 99) Kölzer, A. 1999. Universal Dialogue Specification for Conversational Systems. IJCAI 99 – Workshop on Knowledge and Reasoning in Practical Dialogue Systems.

(Lieberman 95) Lieberman, H. 1995. Letizia: An Agent That Assists Web Browsing. In *Proceedings of the International Joint Conference on Artificial Intelligence,* Montreal.

(Ljunberg & Lucas 92) Ljunberg, M; Lucas, A. 1992. The OASIS Air-Traffic Management System. In Proceedings of the Second Pacific Rim International Conference on AI.

(Maes 97) Maes, P. 1997. Agents that Reduce Work and Information Overload. Software Agents. pp. 145-164.

(Noy & McGuinness 01) Noy, N. F.; McGuinness, D. L. 2001. Ontology Development 101: A Guide to Creating Your First Ontology. Stanford Knowledge Systems Laboratory Technical Report KSL-01-05 and Stanford Medical Informatics Technical Report SMI-2001-0880.

(Paraiso 96) Paraiso, E. 1996. Concepção e Implementação de um Sistema Multi-Agentes Para Monitoração e Controle de Processos Industriais. Dissertation at CEFETPR. (in Portuguese).

(Ramos 00) Ramos, M. 2000. Structuration et évolution conceptuelles d'um agent assistant personnel dans les domaines techniques. PhD thesis at UTC. (in French)

(Russel & Norvig 95) Russel, S.; Norvig, P. 1995. Artificial Intelligence : A Modern Approach. Prentice Hall.

(Shen et al. 01) Shen, W.; Norrie, D.; Barthes, J. P. 2001. Multi-agent Systems for Concurrent Intelligent Design and Manufacturing. Taylor & Francis.

(Shmeil 99) Shmeil, M. A. H. 1999. Sistemas Multiagente na Modelação da Estrutura e Relações de Contratação de Organizações. PhD thesis at Faculdade de Engenharia Universidade do Porto. (in Portuguese)

(Stan & Graesser 96) Stan, F.; Graesser, A. 1996. Is it an Agent, or just a Program ?. In Proceedings of the Third International Workshop on Agent Theories, Architectures, and Languages, Springer-Verlag.

(Sycara 98) Sycara, K. 1998. Multiagent Systems. AI Magazine.

(Tacla & Barthes 02) Tacla, C.A., Barthès, J-P. 2002. A Multi-agent Architecture for KM Systems, Second IEEE International Symposium on Advanced Distributed Computing Systems ISADS 2002 Guadalajara, Mexico.

(Wittig 92) Wittig, T. 1992. ARCHON: Architecture for Cooperative Multi-Agent Systems, Ellis Horwood.

(Wooldridge & Jennings 95) Wooldridge, M. ; Jennings, N. J. 1995. Agent Theories, Architectures, and Languages: a roach. Springer-Verlag.

(Zhang 00) Zhang, C.; Zhang, Z.; Li, Y. 2000. Multi-Agent Based Financial Investment Adviser: Design and Implementation. International ICSC Congress Intelligent Systems & Applications ISA'2000.

List of Abbreviations

AI – Artificial Intelligence

ACL - Agent Communication Language

AOP - Agent-Oriented Programming

BDI - Belief, Desire and Intention

FIPA – Foundation for Intelligent Physical Agents

HCI - Human-Computer Interface

KM - Knowledge Management

KQML - Knowledge Query and Manipulation Language

MAS - Multi-Agent Systems

OOP - Object-Oriented Programming

PAA - Personal Assistant Agents

RDF - Resource Description Framework

R&D - Research and Development

List of Figures

List of Tables