Agents as Animated Creatures: An useful metaphor

Sonia V. Rueda
Departamento de Ciencias de la Computación
UNIVERSIDAD NACIONAL DEL SUR
Bahía Blanca, ARGENTINA
e-mail: srueda@criba.edu.ar

Abstract

Agents represent a new paradigm for developing software applications. In spite of the interest of this particular way of looking at problems, there is considerable discrepancy on exactly *what is* an agent and *which* are the attributes and properties that characterize them. The agent-based view offers not only a powerful repertoire of tools and techniques but also provides a powerful *metaphor* that brings about a better conceptualization of the problems in different areas of Computer Science.

Many computational metaphors have become transparent. That fact has brought confusion to the use of some terms even at the level of fundamental ideas. Mature science is supposed to converge on a theoretical characterization of the systems it studies. Instead computational approaches seem to be diverging, with no broad consensus on central concepts. In particular, the agent concept generates a good deal of controversy.

We consider here a metaphor that has become close to invisible with the hope of clarifying it. This view knits together the different threads that make up the agent model and offers the *animated metaphor* as a conceptual tool for analysis, design and implementation of a diverse range of software applications. That metaphor makes possible new ways of creating programs and, more broadly, new ways of thinking about programs.

1 Introduction

The proliferation of interconnected computer networks has favored the development of new applications, such as electronic commerce, intelligent manufacturing and long distance education and entertainment. All of these applications require mechanisms for updating and querying large systems which have been built from distributed, heterogeneous autonomous components. These mechanisms must be sufficiently flexible and extendible as to allow dynamic adding and removal of components. The complexity and dynamism of these systems compromises the uniformity and consistency of the information.

One possible approach to the development of large, complex systems is to radically change the way we think about programs by using metaphors which imitate real world elements. *Agents* and *multi-agent systems* have been proposed as a natural solution for the development of applications in open, distributed environments with a high level of inter-operability requirements [Jen95].

In the loosest sense of the term, an agent is simply one whose functionality is distributed among active functional modules. By this criterion, even the most ordinary programs could be viewed as agents. However, agents, by their nature, emphasize action-taking rather than computation in the narrow sense. This paper explores a fuller form of *animacy* as a particular way of thinking about agents. An agent may be viewed as an *animated creature* that has *goals*, a degree of *autonomy* and the ability to dynamically *react* to its environment. The *animated metaphor* makes new ways of creating programs, and in a broader sense, new ways of thinking about programs, possible.

The agent model abandons the conventional metaphor in which a computer is presented as a mere instruction-follower, giving it an appearance which is completely anthropomorphic. An agent has a task to accomplish, it reacts to the events that affect its work, and is capable of detecting conflicts within its environment. In a multi agent system, the environment includes other agents, therefore the mapping between the anthropomorphic representation and the computational reality is extended to include social characteristics such as coordination, cooperation, and negotiation. Agent technology raises new problems such as inter-agent conflict and new tasks such as making the activity of a complex society of agents understandable to the user.

The remainder of this work is structured as follows. Section 2 gives a brief overview of contemporary metaphor theory and explores two particular types of metaphors based on some form of anthropomorphism and animism. The anthropomorphic and animate metaphors are powerful for they are capable of exploiting knowledge regarding humans and human actions. They let us think about complex systems in terms of purpose and function. The relevant characteristics of the animate real are puerpose, reactivity and autonomy.

Section 3 presents the agent paradigm as a model strongly based on the animate metaphor. Basic properties of animism, purposefulness, reactivity and autonomy, are central concepts of the agent paradigm. These properties are intimately bound together and require some form of intelligence. Animacy is evident in the *personal assistant* and the *coach* metaphors. In both cases an agent is an animated creature, specialized and personalized, which collaborates with the user within a mutual working environment.

The capacity of an agent is limited by its knowledge, its computational resources and its perspective. The characteristics of complex problems exceed the capacities of an individual agent and require the creation of agent organizations. A multi-agent system is a weakly coupled network of heterogeneous agents working together to solve problems which surpass their individual abilities. In multi agent systems agents do not function in isolation. The environment contains other agents. In section 4, the animate metaphor is extended to include social properties that model the interaction between agents. Finally, section 5 presents some conclusions.

2 Metaphors

In classical theories of languages, metaphors are associated with literary language, especially poetry. In this conception, a metaphor is a linguistic expression in which one or more words are used in a figurate sense, i.e., an expression is used to denote a concept similar to the one literally referred to. In Classic Language Theory metaphors are a central theme in the study of the semantics of a natural language, but they are not linked to thought.

Contemporary theories of metaphor, search for a rigorous generalization that governs metaphors [Lak93]. Every metaphor is viewed as establishing a mapping between domains. Under this conception a metaphor is no longer just a literary tool. It involves thought, linking one mental domain with another.

In contemporary theories the term "metaphorical expression" refers to a twist of the language, while "metaphor" is used to refer to a conceptual mapping which serves as a cognitive tool. A metaphorical expression is the concrete realization of a metaphor [Lak93].

Contemporary Metaphor Theory has discovered a system of metaphors which structures our current conceptual system. According to this theory, a metaphor is not just a basic linguistic mechanism, but a way of reasoning about a little known domain using one which is better known. Using the mapping of domains which a metaphor establishes, the knowledge of one domain, along with a strongly structured correspondence, allow us to reason about another, less familiar domain.

The concept metaphor is strongly tied to the concept of analogy, yet they encompass different meanings. They both allow the establishment of mappings between domains, yet said establishment is different in each case. In an analogy the mapping is explicitly constructed and precisely specified. In a metaphor, however, the mapping, and even the domains, are left implicit. For example, an expression such as: "My life is a like boat which is adrift", establishes an analogy. In contrast, the expression: "My life is a boat which is adrift", is a metaphor. Metaphors and analogies are powerful learning tools, denoting different points of view for the same cognitive process.

Many metaphors are so deeply rooted into our thinking habits and our language that it is difficult to see them as such. If a metaphor is so conventional that it is impossible to distinguish the mapping between domains, then such a metaphor is said to be *dead* or *transparent.* Some authors do not consider these phrases to be metaphors [Lak81].

Metaphors play a key, yet controverted role in scientific discourse. The use of metaphors has been continuous in the explanation of new concepts in terms of others which where already known, as well as in the exploration of domains and the description of their characteristics, often playing a foundational role in the formation of new domains. Even so, some science formalists oppose the use of metaphors, considering them a non-rigorous way of thinking.

Aristotle differentiated literal and metaphorical meaning, and he recognized the importance of metaphors in poetry, although he demanded that they be eliminated from scientific discourse. Yet metaphors are of common use in scientific discourse, particularly in informal and educational environments. While scientific rhetoric may aspire to be literal, the use metaphors is inevitable when presenting new theories based on old concepts.

Kuhn considered that all scientists should be trained in the use of metaphors, since their role is not marginal role. In fact, he believed them to be crucial to the reproduction of any scientific field [Kuh93]. Carl Einstein, an expressionist writer, said: "Metaphor and metaphoricity refer to more than one literary process; they characterize a general mood and attitude. In the metaphor one avoids repeating facts and weakens contact with reality. Metaphoricity is justified by the illusion of arbitrarily creating something new every moment."

In the field of Computer Science, Dijkstra considers that the use of metaphors should be avoided in those domains which suffer from continuous and vertiginous change [Dij89]. In spite of this, the use of metaphors has been widespread in the exploration of artificial worlds and in the description of their characteristics.

Metaphors in Computer Science come so naturally that their presence is hard to notice. Many terms such as "stack" or "garbage collection" have taken a technical meaning all of their own, to the point of loosing site of the original domain. This phenomenon, in which metaphorical terms become literal, exists in many scientific fields, but it is especially frequent in Computer Science.

2.1 Anthropomorphic Metaphor

A metaphorical model structures the knowledge from an abstract and unfamiliar domain by establishing a mapping with the concepts and relationships belonging to a more concrete, well known and already structured domain. An anthropomorphic metaphor is a mapping that describes the activity of a machine or other inanimate object in terms of human qualities. The reason that anthropomorphic metaphors are useful and powerful is because they allow us to explain de workings of systems that are too complex to be understood exclusively in mechanical or physical terms, based on our knowledge of human beings and their actions.

Computer systems, both hardware and software, are so particularly inclined to being anthropomorphisized due to their complexity and apparent autonomy. The use of the term "computation" to refer to the activity carried out by a complex electronic device is, in itself, a metaphor. As in any other case, this metaphor privileges certain aspects of the domain over others. In this case, the operations carried out internally. However, computers are devices which are not solely used for computing. So, the computational metaphor is insufficient.

Metaphors are a frequent tool in the formation of programmers. In most introductory programming courses a program is defined as a recipe. This metaphor allows the students to grasp the behavior of the computer in terms of a familiar domain. Another concrete example is the drawing of rectangles to represent variables and the values they contain. These variables, in turn, give names to memory cells. What is more, the memory as a storage space, is in itself a metaphor. In any case, the point is to describe something abstract in terms of something more tangible.

In every programming language there is an underlying metaphorical model which is in someway anthropomorphic. This model is evident in Object Oriented Programming, in the Actors Model, and even in Logo. However, in each case the metaphor plays a fundamentally didactic role. It is used to illustrate some initial concepts. In an already implemented system, the only persisting relationship to the anthropomorphic metaphor may be thought of as an "instruction follower".

One of the goals of Artificial Intelligence is the derivation of computational models for human thought. The nature of this process is essentially metaphoric, but the direction of the mapping is opposite to the one we have seen up to this point. In the field of Distributed Artificial Intelligence a system is divided into concurrent components communicating with each other. The anthropomorphic metaphor is present in the representation of the components and in the way they communicate.

2.2 Animated Metaphor

An *anthropomorphic metaphor* describes the activity of a machine or other lifeless object by establishing a mapping with qualities which are generally human. However, the term *anthropomorphic* includes other animals besides humans. An *animated metaphor* is a more appropriate mapping because it describes the activities of a machine or other inanimate object in terms of an *animated creature*. The animated metaphor lets us think about a system in terms of its purpose or function, making it possible to understand it even without any knowledge of its implementation [Tra96].

The concept of animation is so primitive that it is difficult to define precisely. Animation may be thought of as a basic category of the human mind, which presents itself in a variety of different forms throughout the cognitive levels. The roots of animation may be found based on perceptual processes, but its ramifications escape perception and affect the way in which we think about actions as well as social behavior [Pap80].

The dividing line between what is animate and inanimate would appear to be universal and innate. The ability to make such a division is, in a way, perceptual, since it is based on the capacity to distinguish, firstly, motion from immobility and, secondly, autonomous movement from induced movement. From this point of view, however, animate and alive would be equivalent categories.

The categories animate and alive are related but are clearly distinct. For example, plants are alive but they are not (usually) animated. On the other hand, a robot would not be included in the category of living things and yet it could be animated. Computers blur the dividing line between what is animate and what is not, as few other objects do.

Animacy can be understood as a framework or way of thinking, rather than as a category. It serves as a source domain for metaphorically understanding some mechanical systems. Travers defines the term *animated creature* as an entity which is endowed with the following properties:

- *autonomy*: an animated creature is capable of initiating an action without external cause.
- *purpose*: The actions carried out by an animated creature are oriented towards the accomplishment of some goal. In this way, actions may be evaluated in terms of their effectiveness.
- *reactivity*: Animated creatures modify their actions to adapt to changes in their environment.

These properties are bound together according to a complex structure, in which they usually oppose each other. The most controverted relationship is between autonomy and reactivity: the more a creature reacts to its environment the more autonomy it loses.

The animated metaphor has been exploited for years in order to explain the operations of the Logo language to children. Under this metaphor, the computer is a "little person" capable of carrying out specific tasks as well as ordering other little people to carry out other tasks. Each little person is asleep until it is awakened so it may carry out its task. When it is finished with its task, it goes back to sleep. All of the little people are reactive, but not autonomous. Specialization is, in some way, a form of purpose.

The animated metaphor is underlying in the design of graphic user interfaces. For years, interface design was based on the *direct manipulation model*. This is to say, the user explicitly initiates all tasks and monitors all events [Sch88]. The system acts only due to an explicit requirement from the user. Techniques from the field of AI have been used to implement a complementary style of interaction referred to as *indirect management*. A graphic interface is an anthropomorphic entity that controls the environment and acts as an intermediary between the user and system.

We may conclude, then, that the animated metaphor has been present, implicitly or explicitly, in an array of different areas. It is in the agents model, however, where it receives singular transcendency. Agent technology is an attempt to construct a new paradigm for programming that makes computational animism explicit and attempts to extend it beyond the instruction-following metaphor to include the central properties of animism. An agent combines the power of computation with the aspects of animism that previous forms of computational animism have left out. Complex behavior is achieved by combining simple agents into more complex structures.

3 Agents

The notion of an intelligent automaton that dutifully serves computer users has been cause of fascination since the appearance of the very first digital computers. Yet computers are not very good at knowing what to do. Every action that a computer performs must be explicitly anticipated and coded [Woo99]. Should a system confront a situation that hasn't been anticipated by its designer, the result will probably not be the expected one. The underlying metaphor is that of the obedient servant: literal, absolutely lacking in imagination, and with no capacity for independent decision making.

The concept of an agent brings forth images of human-like automatons that work without supervision on tasks deemed beneficial to us. The agents take over human tasks and interact with people in human-like ways. Some agents have the potential to form their own goals and intentions, to initiate actions on their own without explicit instruction, and to offer people helpful suggestions.

Agents are being used in an increasingly wide variety of applications, in which they assist users in many different ways by:

- reducing the complexity of hard-to-solve problems
- acting on behalf of the user
- training in the use of resources
- monitoring events

Therefore, the concept of "agent" should not be difficult to define. However, this term, as has happened with so many others, has been used to refer to, or describe, many related, but often different, concepts. The situation grows even more complicated if we consider the term agent in connection to other terms, e.g., intelligent agents, network agents, software agents, *et cetera*. As a result, there is currently no widely accepted universal definition of agent. In one extreme approach an agent is a cognitive entity, endowed with intelligence, perception, feelings and emotions. Under this perspective, a computational entity could not be considered an agent [Huh97]. An opposite approach considers an agent to be an automaton that behaves according to how it was programmed. This characterization is overly permissive, rendering any piece of software an agent.

The key aspect which distinguishes these two approaches is complexity. The more permissive characterization does not attribute any special behavior to differentiate an agent from any other software entity. Hence, simplicity compromises the level of abstraction. In contrast, in the more rigid characterization, the level of abstraction required is excessively high, to the point of excluding any artificial system from being considered an agent. Even though there still does not exist an agreed-upon definition for the term agent, the vast majority of the approaches converge to some intermediate posture.

3.1 Agents as personal assistants

One of the first areas of application for agents was that of human-computer interfaces design. Interface agents radically change the style of human-computer interaction. The user delegates a range of tasks to personalized agents that can act on the user´s behalf.

The metaphor used is that of a personal assistant who collaborates with the user in the same work environment [Mae94]. An agent is not merely an interface between the user and the system. It is an animated creature, specialized, personalized, and capable of initiating autonomous actions for the user. The user, in turn, can take action and perform tasks on his own, if he so wishes. To the user the agent is a proactive artifact that can perform fairly sophisticated tasks and can also exhibit learning abilities [Cho98]. The agent:

- Knows about its user, his wishes, interests, habits and preferences
- Is informed about other agents and their profiles

- Is able to perform a set of tasks on behalf of a user
- Has communication skills that enable it to interact with the user
- Is able to monitor, interpret, and modify its environment in order to be able to perform task autonomously
- Is able to collect and present information to its user
- Can structure system elements in real time to meet users' needs

The automation benefits of an agent are applicable for automating the repetitive behavior of a single user, particularly when this repetitive behavior is dissimilar across the general user population. Repetitive behavior can be either *time-based* or *event-based.* A time-based task is something that the user does at a particular time. An event-based task is something that the user does in relation to another task [Cag97].

The personal assistant metaphor may be extended to that of an *expert assistant*, which is capable of collaborating with groups of users connected via a network. An agent with learning capability can be used as an expert that offers *customized information* to a group of users with similar attributes and *repetitive task automation* to groups with similar behavior.

Agent technology can successfully work in place of a human coach, giving personalized instruction in a graphical and animated environment [Sel94]. The *advisory metaphor* builds agents with the explicit goal of training an individual user. The human-like style of the personal assistant metaphor persists but in educational situations. The agent watches the user´s actions, records its experience, builds an adaptive user model and creates personalized help.

The agent is an advisory that does not interfere with the user´s actions but comments opportunistically to help. The agent can learn and reason about the user and use this information to guide computer reaction. The adaptive scenario moves users to a learn-while-doing approach. The terms coach, tutor, and wizard are often used when referring to this type of agents. In each case an agent is an animated creature with human-like attitudes and characteristics.

3.2 Agents as animated entities

In an interpretation that supports the animated metaphor, the agent must portray the three basic properties of animation: *purpose*, *reactivity*, and *autonomy.* These three properties are intimately bound together. An autonomous entity, initiates an action on its own, but it must have some reason to do so. The simplest reason is a reaction to a change in its environment. Autonomy allows an agent to act and interact in accordance to its own goals. Reactivity attempts to keep coherency in the system as a whole. A *proactive agent* combines both properties: it may act in response to stimuli from the environment, but it is also capable of initiating actions conceived to reaching their own goals.

We may say, then, that an agent is an autonomous entity with an explicit goal. The actions of an agent are in relation to its goal and to what it perceives, and in turn, these actions can induce changes in the environment. Goals give agents a completely anthropomorphic appearance: an agent has a job to do and it detects, and reacts to, the events which affect its goals.

In the agents model the role of the animated metaphor is therefore central. The model redefines the way in which software is built and interacted with according to a mapping that establishes a strong correspondence between the computer and an animated entity. The anthropomorphic characteristics present themselves as much in the language as in the interface, and the user is encouraged to think in terms of the agents activity. An agent is not a mere "instruction follower", but an animated entity with its own individual characteristics.

The anthropomorphic qualities of agents give rise to some difficulties: as is the case with any autonomous entity of significant complexity, agent behavior cannot be fully analyzed. Agents may exhibit accidental or intentional unanticipated behaviors.

In the assistant metaphor the agent acts on behalf of the user. In interactions on public networks the agent is responsible for the user. The agent may contain personal information about the user that must be subject to privacy and integrity protection. Security exceeds the scope of agents. However, the success of agents depends on the effectiveness of security services.

3.2.1 Purpose and Behavior

The actions carried out by an agent are oriented towards the accomplishment of specific purpose or goal. These goals are strongly related to reactivity and require a certain conflict detecting capacity. In this way, an agent may react in light of events that affect its goals and its actions may be weighed in terms of effectiveness. Goals form the conceptual and computational base for the behavior of an agent. However, these goals may be explicit or they may be implied in the specification of the agent's behavior.

In *pure behavior based systems*, the specification of behavior is explicit, but the purpose is implied in the design. In a complex system, it may be difficult to discern the purpose of an agent and its relation to the environment. In contrast in *pure goal oriented systems*, the purpose of the agent is explicit. Each agent uses planning or some other decision process to direct its actions towards the accomplishment of its goals.

Alternatively, the behavior and purpose of an agent may be explicit within its computational representation. The purpose of an agent may be specified by means of declarative information that allows the agent to determine when it must execute, as well as monitor the environment and its own actions.

3.2.2 Reactivity

An agent operates within a changing world, therefore it must be capable of sensing changes and of reacting adequately. Reactivity denotes the ability of an agent to receive information from its environment through its sensors, and to take action in such a way as to somehow change said environment.

An agent has a repertoire of actions available to it. This repertoire represents the agent's *effectoric capability*, i.e., its ability to modify the environment [Woo99]. In most domains, an agent does not have complete control over its environment. Not all actions can be performed in all situations and the same action performed twice in seemingly identical circumstances might appear to have different effects.

The key aspect of reactivity is deciding *which* of the actions should be executed in order to achieve maximum goal satisfaction. The complexity of this decision process depends fundamentally on the properties that characterize the environment.

Reactivity separates an agent from the mindless executor suggested by the "instruction follower" model. Even if an agent is capable of executing a sequence of instructions, it must also be capable of responding to changes in its environment.

3.2.3 Autonomy

Autonomy is the characteristic that allows an agent to control its own actions as well as its own internal state, functioning without need of direct external stimuli. This aspect is vital from the application point of view because agents operate inside a dynamic environment, within which they must be able to initiate actions on their own. The success of solution regarding the paradigms level of application is strongly related to the an agent's capacity to make it's own decisions.

Autonomy may be implemented using a function which determines the agents course of action based on its set of beliefs. These beliefs comprise a partial vision of the agent's world and include its internal and external state [Had96] [Tra96].

The result of an agents actions may affect not only its goals and its internal state, but also the context. The majority of the decision-taking functions developed for autonomous agents concentrate on selecting a rational behavior from an individual point of view, i.e., they do not consider the impact of their behavior on the context.

3.2.4 Rationality and Intelligence

We consider an agent to be *rational* if its knowledge concerning which actions it may carry out allow it to reach some specific goal [Ste96]. A *rational agent* reacts to changes in its environment and it attempts to reach a maximum level of utility acting based on its knowledge and perceived evidence.

Most of the formalizations referring to rationality require that the agent have preferences within the possible states of the world, some knowledge of its surroundings and that it be able to recognize the actions that allow it to maximize these preferences. The basic mechanism that allows the coordination of actions to reach goals is called *planning*.

Rationality requires some abilities, such as [Huh97]:

- perception
- capacity for interpretation and categorization of perceived information
- reasoning based on intentions and beliefs
- decision taking
- plan selection, revision and construction
- capacity to carry out actions and plans

Rationality is in someway determined by the agent's level of perception, knowledge, and computational power.

Intelligence is an attribute which is difficult to define. The actions carried out by an agent must show some form of intelligence based on:

- Perception
- Knowledge
- Learning
- Reasoning
- Adaptation

In the personal assistant metaphor, perception is a vital property because it lets the agent understand the users needs and desires based on his/her perceived behavior. The agent must be able to *capture* the behavior and knowledge of the user in order to be able to act on his/her behalf. Therefore, we may measure an agents intelligence based on its capacity to acquire abilities and knowledge, as well as its capacity to reason based on said knowledge.

The more modest forms of intelligence allow the user to express his/her *preferences* and the agent to act according to them. An agent with a higher level of intelligence is endowed with some ability to *reason*. In this case the agent is able to carry out some inference process which allows it to obtain new information based on its current knowledge. On an even higher level, an agent may *learn* and modify its behavior based on acquired knowledge.

An appropriate approach to thinking about intelligence for the task at hand requires taking into account the relationship between an intelligent creature and its environment. Intelligence is located in the relationship between the mind and the world. It is therefore crucial that agents be embedded in some kind of environment in which they can act on other active or passive objects. The environment is an important component, since it determines the kinds of actions and interactions that can take place.

The construction of an intelligent agent must consider the environment in which it will operate. Interactive computer environments provide both the worlds and raw material where construction can take place.

3.2.5 Knowledge and Learning

All approaches to agents share the notion of "expertise". In real life somebody is considered an expert if he or she has a large knowledge domain in the form of facts and rules. Knowledge is the trademark of the expert [Cho98]. Knowledge enables the expert not only to recognize and formulate problems but also to choose promising problem solving strategies, be they successful or not.

An agent has some knowledge about its environment that allows it to carry out its task. Knowledge may be acquired through [Cag97]:

- A specification provided by the designer. The designer of the system obtains a model of the domain with which the knowledge base is constructed.
- A specification provided by the user. In this alternative, the user is responsible for providing the knowledge necessary for the agent to carry out its task.
- The system observes and records the users actions.

The first alternative is overly rigid because the agents knowledge is determined during design. The final alternative suggests that the agent *learns* when it acquires new knowledge, but of course, the implementation is more complex. Many systems combine these alternatives so that every agent starts out with an initial knowledge base that is later expanded through interaction the with the user.

Numerous authors have explored the potential of machine-learning techniques to automatically create and maintain customized knowledge for personal software assistants [Mit9494] [Mae94]. The agent is given a minimum of background knowledge and it learns appropriate behavior from the user. This approach assumes that each user has a repetitive, perceivable behavior which is potentially different from the behavior of other users. In other words, the users actions are repeated a significant number of times following a pattern different to the behavior observable in other users. The personal assistant metaphor has a fundamental role in this approach: an agent is an assistant that acquires competence as it interacts with the user. In parallel, the user gains confidence in the agent as he/she notices its capacity to recognize his/her habits and preferences.

3.2.6 Other anthropomorphic qualities of agents

The characterization of agents in anthropomorphic terms may be extended in several ways. It is possible to increase the computational estates in order to reflect emotional states. What is more, we may use these states to modify the facial or bodily appearance of the characters.

An agent should be able to explore different methods for reaching its goals. One approach to this problem is that each agent govern the work of a set of subordinate agents, or subagents. Subagents have the same goal as the main agent, but they try to reach said goal by different methods. The main agent activates the subagent following some strategy until the goal is reached. The subagents are, in a way, losing their autonomy, since their behavior is fundamentally reactive.

Agents should be capable of acquiring experience, remembering which methods where more suitable for attacking a given problem and under which circumstances they worked. An expert agent is capable of remembering its decisions, without having to repeat actions which turned out to be fruitless. The challenge is to find an adequate representation for each situation so as to make the detection of coincidences relatively easy. Clearly, the expertise of the agent compromises its simplicity.

The characterization of an agent may be completed with additional properties, such as mobility, security, and others. However, it is probable that these qualities will turn out to be useful in some application domains, while being inconsequential in others.

4 Multiagent Systems

The capacity of an intelligent agent is limited by its knowledge, computational resources, and perspective. Complex problem requirements exceed the capacities of an individual agent and require the creation of *agent organizations* [Huh97].

The agents model is extended to attack the problem of building large and complex systems. Under the animated metaphor, a complex system is conceived as an organization of coordinated animated creatures. An *animated system* is a virtual world formed by interacting animated creatures.

A multiagent system is a system design and implemented from an organization of interacting agents. Each agent is an individual component specialized on a specific function and responsible for attending to some particular aspect of the global problem. Agents must coordinate their actions to achieve the common goal and resolve conflicts amon their individual goals. Interaction between agents includes coordination, cooperation and negotiation: all anthropomorphic activities. In a multiagent system the agents have a lifelike behavior that is non mechanical and can be spontaneous [Cho98].

In a multiagent system, the behavior of an agent is affected by the presence of other agents in its environment. Agents:

- perceive and communicate with other agents within a social context.
- confront their goals with the behavior and action carried out by other agents
- operate on a certain level of uncertainty, due to the fact that they must rely on only partial information

Investigations in multiagent systems have used as a permanent source of inspiration works in philosophy, political and economical sciences, sociology, etcetera. The *scientific community* and the *contract-net protocol* [Dav88] are metaphors that establish a systematic mapping between systems of agents and human organizations. The animated metaphor is extended to include social characteristics.

4.1 Characteristics of Multiagent Systems

The outstanding characteristics of a multiagent system are [Woo99]:

- Each agent has a partial viewpoint of the global problem, limited by its own capacity and knowledge.
- Agents are autonomous and probably heterogeneous, there is no global control.
- Information is decentralized and no agent has access to all of the available information
- Computation is asynchronous.

The fundamental advantages of a multiagents systems over a monolithic ones consisting of single simple agents are [Cag97]:

- They support the animated metaphor which allows the modeling of societies of interacting autonomous agents.
- They allow the solving of problems which are to large to be handled by a single agent with limited capacities, therefore distributing the expertise among a set of agents.
- They favor extensibility and reuseability through the interconnection and inter operation of preexisting legacy systems. The abilities of an agent may be broadened and the number of agents may grow.
- They are more flexible due to the fact that the different abilities of agents may be organized in different ways in order to adapt to modifications in the problem

- They are more reliable because agents may have redundant capacities, and they may assume the responsibilities of the failing components.

A multiagent system improves efficiency on several levels [Cho98]:

- In the *access to distributed information*, particularly because it is possible to access results and high-level information directly, instead of transmitting unprocessed data.
- In the *computation* because concurrency allows for a better use of resources
- In the *verification* because the anomalies of a component may be resolved locally, avoiding their propagation to the entire system.
- In the *maintenance* because modularity allows the system to be more flexible and adaptable to change.

The design and implementation of a system of coordinated agents also presents some challenges, namely:

- Formulation, description and decomposition of problems
- Synthesization of results
- Interaction: what, when, and how to communicate
- Representation and reasoning about actions, both proper and foreign
- Recognition and conciliation of conflicting or uneven perspectives and goals

In turn, the problems mentioned for single agents are extended and transformed into collective phenomena. In particular, the unexpected behavior of an agent, be it accidental or intentional, my present itself when the agent acquires new abilities by observing the behavior of other agents.

4.2 Heterogeneity

In a system of multiagents, the agents present different levels of heterogeneity. Agents may be identical or they may differ in the resources they provide, in their specialty, in the methods they use to solve problems or in the design restrictions from which they have been built.

In the development of an open system, it is fundamental that as few restrictions as possible be placed on the design of the agents. It should be possible for agents to have different internal architectures and be developed in different programming languages as well as on different platforms. In any case, with a greater or lesser level of heterogeneity, agents should share a language that enables them to interact.

Bird classifies the various reasons that lead to heterogeneity among agents into *syntactic*, *semantic*, and *control* [Bir93]. Syntactic differences come about when different formalisms are used to represent knowledge. Heterogeneity turns out to be useful in this case for maintaining alternative perspectives of the same knowledge. If the formalisms are computationally equivalent then it is possible to build an agent that implements the conversion between them. Semantic heterogeneity arises when the same knowledge can be interpreted in different ways by different agents. This aspect is probably the hardest one to manage.

Control differences are a result of the use of different inference strategies to process the same knowledge. In this case, heterogeneity can lead to confusion because it may be difficult to identify which method is being used. Moulin considers that an autonomous agent has certain abilities, such as perception, reasoning, intentions, and planning, that may be present in different levels [Mou96]. From the level of each of these capacities, Moulin classifies agents into three groups: *reactive agents*, *intentional agents*, and *social agents*.

A purely *reactive agent* reacts to changes in its environment or to messages sent by other agents. It is not capable of acting according to interests of its own. An expert system composed of a set of rules, a base fact, and an inference engine constitutes a purely reactive system.

An *intentional* agent is capable of reasoning about its intentions and beliefs, create plans of action, and carry them out. A planning system may be thought of as a system of intentional agents.

A *social agent* maintains an explicit model of other agents and it modifies said model dynamically. A social agent reasons, plans, and makes decisions considering its own expectations but also its knowledge of other agents. A social agent's level of complexity is considerably greater to that of the other types of agents.

4.3 The properties of agents in a multiagent system

In a multiagent system, each agent interacts with others within a social context in order to reach certain global objectives. Each agent must be able to execute some actions inherent to their specific abilities, but it must also have capacities that allow it to measure the social impact of their behavior.

Each agent is characterized by *intrinsic* properties that are related to it and its behavior. These are extended by *extrinsic* properties that are related to the behavior of the agent within a context in which other agents participate. Let us note that even within a system with only a single simple agent, there exists an environment which the agent perceives and that it somehow affects its behavior. The dependency of an agent to its environment is strongly connected to its level of autonomy and reactivity.

In a multiagent system, the environment includes other agents. The intrinsic properties of agents are still purpose, reactivity, autonomy, but the requirements to support them are greater. In general, in a system of multiagents, not all agents are equally endowed with each of the abilities.

Reactivity refers to the capacity of agent to react to certain events that somehow modify its environment. In a multiagent system, these changes can be a result of the actions of other agents. Clearly, not all changes to the environment will cause every agent to react. Each agent may sense only those events that require its intervention, or it may also be able to sense some events, even when it doesn't react to them.

The autonomy of an agent is related to the predictability of its actions. The less predictable the behavior of an agent is, the greater its autonomy. This property my present itself on different levels depending on the characteristics of the application. Even within the same application, the level of autonomy can depend on the perspective being considered. In this way, an agent whose behavior seems to be absolutely autonomous regarding another, can be a third agent's slave.

A completely autonomous agent will act exclusively in accordance to its own needs. In an extreme, an *autistic* agent will have no awareness if its environment, i.e., its actions will be completely autonomous. Absolute autonomy, in general, is not useful within a system of multiagents because, beyond the individual purpose of each agent, there exists a global goal that requires a certain level of sociability on behalf of the agents.

A social agent is aware of its colleagues and it attempts to coordinate its action with those of the other agents. The social agent's autonomy is in a way restricted by its *social responsibility.*

4.3.1 Social Responsibility

An individual's social capacity makes him or her consider the effect his or her actions will have on the community. The decision making function will not depend exclusively on the individuals own goals. It will be designed with a certain degree of social responsibility. A decision-taking function with social responsibility should be capable of:

- Identify the set of all action combinations it can carry out.
- Define an order of preference among the different combinations of actions in the set.

Clearly, the agent can elaborate a preference order using different criteria. From an *altruistic* point of view, the best combination of actions will be that which achieves the greatest global benefit. On the other extreme, a totally *selfish* posture will select as preferable the combination which grants the greatest individual benefit. Between these extremes, it is possible to recognize three intermediate approaches [Kal97]:

- **Individualist with social responsibility**: Considers as candidates those combinations of actions which grant an individual benefit without bringing detriment to others. In other words, this approach still gives priority to the agents own interests but discards those actions that harm the agent's society.
- **Collaborator**: includes among the candidates those combinations which bring benefit to the society as a whole, even when no direct benefit is obtained by the executing agent.

- **Cooperative**: considers as candidates those combinations of actions that bring a social benefit and that, combined with the actions of other agents, also bring individual benefit. This is to say, a combination of actions may be considered even when they will cause detriment to the executing agent, if said agent is sure that its loss will be compensated by another agent.

All three alternatives follow the *Principal of Social Rational*. In a system based on this principal each agent makes decisions considering its own objectives and those of society as a whole. Eventually, the actions taken by an agent with social responsibility will reduce its potential benefit.

In each case it is (naively) assumed that each agent keeps the agreements in which it participates and that said agent can predict the impact of its actions over its own state, over the environment, and over the rest of the affected agents. In the first two approaches each agent analyzes the benefit and the detriment its actions will bring both to itself and to its society. Neither of these approaches considers the effect of the combined actions of several agents.

The agents models is particularly valuable when cooperation is exploited as much as possible. We may then say that the animated metaphor contains social characteristics. However, as is the case with human societies, it is not always possible to maximize the level of cooperation.

4.3.2 Coordination, Cooperation and Negotiation

The concepts of coordination, cooperation and negotiation are central to the development of multiagent systems and they are strongly related amongst each other. The use of each of them has given way to a wide array of often inconsistent interpretations.

In an environment with limited resources, agents must coordinate their actions with each other to further their own interests or to satisfy group goals. The actions of multiple agents need to be coordinated because there are dependencies between the agents' actions, there is a need to meet global constraints, and no single agent has sufficient competence, resources or information to achieve system goals [Huh99] [Jen96]

Coordination is a basic property of a system that is formed by agents that carry out activities within a shared environment. It is process from which agents reason about their local actions, as well as anticipate the actions of others, in an attempt to insure coherency in the behavior of the community to which they belong. The fundamental goal of coordination is to insure:

- **Cover**: every part of the global problem must be within the capacities of at least one of the agents.
- **Consistency**: The actions of the agents are guided by consistent processes.
- **Connectivity**: agents must be capable of interacting so that their partial activities can be integrated into a global solution

All of these goals must be reached considering limited resources, that is, restrictions over the systems processing power, in computation as well as communication.

Cooperation arises from the coordination of non-antagonistic agents. In order for cooperation to be successful, each agent must maintain a model of the other agents, and it must be able to develop models of future interactions. A basic strategy shared by many of the protocols for cooperation is to decompose and then distribute tasks. The best known mechanism used for task distribution is *the contract net protocol* [Woo99]. The contract net protocol is an interaction protocol for cooperation based on the contracting mechanism used by businesses to govern the exchange of goods and services. An agent wanting a task solved is called the *manager*, agents that might be able to solve the task are called *contractors*. The roles of agents are not specified in advance. Any agent can act as manager or contractor. The contract net protocol uses the following metaphor: finding an appropriate agent to work on a given task.

Negotiation is a process in which two or more parties exchange information and come to an agreement. The main ingredients of negotiation are *communication* and *decision*. Negotiation is used for [Mul96] :

- **Task and resource allocation**

- **Determination of organizational structure**
- **Recognition and resolution of conflicts**
- **Resolution of goal disparities**

Then, negotiation arises when two or more agents, human or artifical, need to work together in order to reach a shares goal or different individual goals. Negotiation is a process in which each party attempts to optimize the satisfaction of its goals, offering or requiring actions of the other parties [Cag97]. The concepts of cooperation, coordination and negotiation emerge from complex social relationships and they are strongly related to *communication*. Agents communicate in order to understand and be understood.

4.3.3 Communication

Agents communicate in order to better achieve their goals or those of the society in which they exist. Communication can enable the agents to coordinate their actions and behavior, resulting in systems that are more coherent. There are three aspects to the formal study of communication: *syntax*, *semantics* and *pragmatic*. Meaning is a combination of semantics and pragmatics.

Spoken human communication is used as the model for communication among computational agents [Woo99]. A popular basis for analyzing human communication is *speech act theory*. Speech act theory views human natural languages as *actions*, such as request, suggestions, commitments and replies. The theory uses *performative* to insure that there is no doubt about the type of message. The performative simplify the design of software agents. The metaphor for communication once again establishes an anthropomorphic mapping.

An agent's ability to communicate allows it to interact with the user and with other agents. User communication techniques range from e-mail messages to dialogs with an anthropomorphic appearance. In the case of inter-agent communication the typical solution is the use of some communication language, such as Telescript, SmalltalkAgents, KIF, *etc.*

4.4 The Cooperating Experts Metaphor

A natural metaphor for the problem of interacting agents in a distributed environment is that of a group of experts that attempt to accomplish a large and complex task [Dav88]. None of the experts controls the others, but one of them may be responsible for transmitting the final solution outside the group. This metaphor is adequate because it allows the identification of some of the actions relevant to this interaction.

- The way in which they interact
- The manner in which the load is distributed
- The way in which the obtained results are integrated by each of the parties.

The 'task-sharing' model describes a form of interaction in which each of the parties spends most of the time working on its own task, occasionally interacting with another member of the group to require assistance, provide help or exchange results. Each member of the group may require assistance because the task it must perform is too great or escapes its abilities. If it is too great then it may partition it and ask for help from those members that are able to handle the subtasks. It may also transfer to problem to other members of the group if it is not specialized in the task it must perform.

When one of the members of the group is going to transfer a task, it may or may not know of an expert that may perform said task. If it does know of an expert, then it notifies it of its desire to transfer the task. If does not know of anyone capable of carrying out the task, then it may ask for volunteers through a mass solicitation. All experts that are available may offer themselves as volunteers and the solicitor may select one of them. All interaction among members of the group has the form of a negotiation that tends to distribute the load and allows for each task to be performed by specialists.

5 Conclusions

Modern information systems are distributed, large, open and heterogeneous. The topology of these systems is dynamic and their content is changing so rapidly that it is difficult for a user to obtain correct information and for the enterprise to maintain consistent information. Computers have become tightly connected, both with each other and their users. The increasing complexity of information systems and computers is matched by an increasing complexity of their applications. Agent technology promises to provide high-level interaction for modern computing and information processing systems. Agents must operate robustly in rapidly changing, unpredictable or open environments, where there is a significant possibility that actions can fail.

Multiagent systems are the best way to characterize or design distributed computing systems. In multiagent systems, agents do not function in isolation, the environment contains other agents. It makes sense to view such systems as a social metaphor. The possibility of an agent interacting with another in unanticipated ways causes the agent's developer to think about it, and construct it, differently.

Over the past two decades, a number of significant conceptual advances have been made in both the design and implementation of individual autonomous agents as well as multiagent systems. Agents research needs the integration of many different research domains and the results serve to directly benefit everyone. As we have shown in this work the animated metaphor is a tool with the capacity of playing an important role in developing and analyzing agent models.

References

[Tra96] Travers, M., *Programming with Agents: New metaphors for thinking about computation*, Ph.D. Thesis, MIT, Boston, Massachusetts, 1996.

[Cyp94] Cypher, A., D. Smith and J. Spohrer, *KidSim: Programming Agents without Programming Language*, Communications of the ACM, **(37) 7**, pp 54-67, 1994.

[Cyp95] Cypher, A. and D. Smith, *KidSim: End User Programming of Simulations*, CHI95. Denver, Colorado, 1995.

[Pap80] Papert, S., *Mindstorms: Children, Computers and Powerful Ideas*, Basic Books, New York, 1980.

[Lak93] Lakoff, G., *The contemporary theory of metaphor* in *Metaphor and Thought*, A. Ortony Ed., Cambridge University Press, 1993.

[Lak81] Lakoff, G., Johnson, M., *Metaphors we believe by*, The University of Chicago Press, 1981.

[Min87] Minsky, M., *Society of Mind*, Simon & Schuster, New York. 1987.

[Sho93] Shoham, Y., *Agent Oriented Programming*, Artificial Intelligence, **60(1)**, 51-92, 1993.

[Cag97] Caglayan A., Harrison C., *AGENT Sourcebook* , Wiley Computer Publishing, 1997.

[Cho98] Chorafas, D., *Agent Technology Handbook*, McGraw-Hill Series on Computer Communications, 1998.

[Woo99] Wooldridge, M., *Intelligent Agents*, The MIT Press, 1999.

[Huh97] Huhns, M., Singh, M., *Agents and Multiagents Systems: Themes, Approaches and Challenges* in Readings in Agents, Morgan Kaufman Publishers, Inc. 1997.

[Huh99] Huhns, M., Stephens, L., *Multiagent Systems and Societies of Agents*, The MIT Press, 1999.

[Sch88] Schneiderman, B., *Direct Manipulation: A step beyond programming languages*, IEEE Comput., 16,8 Aug, 57-69, 1988.

[Mae94] Maes, P., *Agents that reduce work and Information Overload*, Communication of ACM, 37,7 31-40, 1994.

[Sel94] Selker, T., *COACH: A Teaching Agent that Learns*, Communication of ACM, 37,7 92-99, 1994.

[Gen94] Genesereth, M. Ketchpel S., *Software Agents*, Communication of ACM, 37,7 48-53, 1994.

[Mit9494] Mitchell, T. *Experience with a Learning Personal Assistant*, Communication of ACM, 37,7 80-91, 1994.

[Rie94] Riecken, P., *Intelligent Agents*, Communication of ACM, 37,7 18-21 1994.

[Mou96] Moulin, B. Chaib-draa, B., *An Overview of Distributed Artificial Intelligence*, Foundations of Distributed Artificial Intelligence, John Wiley and Sons, 1996.

[Mul96] Müller, J., *Negotiation Principles*, Foundations of Distributed Artificial Intelligence, John Wiley and Sons, 1996.

[Dav88] Davis, R., Smith, R., *Negotiation as a metaphor for distributed problem*, In Readings in Distributed Artificial Intelligence, Morgan Kaufman, 1988.

[Dij89] Dijkstra, E., *On the Cruelty of Really Teaching Computing Science*, CACM 32 (12) 1398-1404, 1989.

[Kuh93] Kuhn, T., *Metaphor in Science* , Metaphor and Thought, Cambridge University Press, 1993.

[Jen96] Jennings, N. *Coordination Techniques for Distributed Artificial Intelligence*, Foundations of Distributed Artificial Intelligence, John Wiley and Sons, 1996.

[Kal97] Kalenka, S., Jennings, N., *Socially Responsible Decision Making by Autonomous Agents*, Proceedings of 5th Int. Colloq. on Cognitive Science, (Eds. Korta, K., Sosa, E., Arrazola, X.)153-169, 1999.

[Ste96] Steiner, D., *IMAGINE: An Integrated Environment for Constructing Distributed Artificial Intelligence Systems*, Foundations of Distributed Artificial Intelligence, John Wiley and Sons, 1996.

[Had96] Haddadi, A., Sundermeyer, K., *Belief-Deside-Intention Agent Architectures*, Foundations of Distributed Artificial Intelligence, John Wiley and Sons, 1996.

[Jen95] Wooldridge, M. and Jennings, N., *Intelligent agents: Theory and Practice.*, The Knowledge Engineering Review 10,115-152, 1995.

[Bir93] Bird, S., *Towards a taxonomy of multi-agent systems*, Inter. J. Man/Mach. Stud. 36, 689-704, 1993.

[Ind93] Indurkhya, B., Metaphor as change of representation: an artificial intelligence perspective, J.Expt.Theor.Artif.Intell.9, 1997.

[Dor96] Doran, J.E., Franklin, N., Jennings, N., Norman, T., *On Cooperation in Multi-Agent Systems*, The Knowledge Engineering Review 12(3) 309-314, 1996.

[Jen98] Jennings, N., Sycara, K., Wooldridge, M., *A Roadmap of Agent Research and Development*, in Autonomous Agents and Multi-Agent Systems, 1, 275-306, 1998.