A SHORTED PRESENTATION OF DISTRIBUTED INTELLIGENT AGENTS

POPA Ilie
University of Pitesti,
Electronics and Computers Department,
Targul din Vale Street, No. 1, Pitesti, Arges, 110040, Romania
ci.popa@yahoo.com, ilie.popa@upit.ro

Abstract. The paper is a brief overview of distributed intelligent agents, as a software domain which uses the latest methods of artificial intelligence for learning and self-sustaining, of parallel and distributed processing and distributed systems communication techniques. Begins with a very brief history, continues with the presentation of the concept of distributed intelligent agent, some elements regarding to classes and architectures, mains agents typology described briefly. It shows also the motivations and advantages of using these structures.
Keywords: Intelligent Software Agent (*ISA*), Multi-Agent systems (*MAS*), Collaboration Learning Agents (*CLA*), Distributed Artificial Intelligence (*DAI*), Parallel Artificial Intelligence (*PAI*), Distributed Problem Solving (*DPS*), Open Agent Architecture (*OAA*).

1. Introduction

The concept of agent was first proposed in 1977 by Hewitt's Actor Model - "A self-contained, interactive and concurrently-executing object, possessing internal state and communication capability [1, 2]."

The software agent systems are a direct evolution from *MAS*. *MAS* evolved from *DAI*, *DPS* and *PAI*, thus inheriting all characteristics from *DAI*.

John Sculley's 1987 "Knowledge Navigator" video portrayed an image of a relationship between end-users and agents [1, 3]. Being an ideal first, this field experienced a series of unsuccessful top-down implementations, instead of a piece-by-piece, bottom-up approach.

In the lasts ten years the approach of *DAI* was very intensive. Nwana's Category of Software Agent [4] is presented in Figure 1.

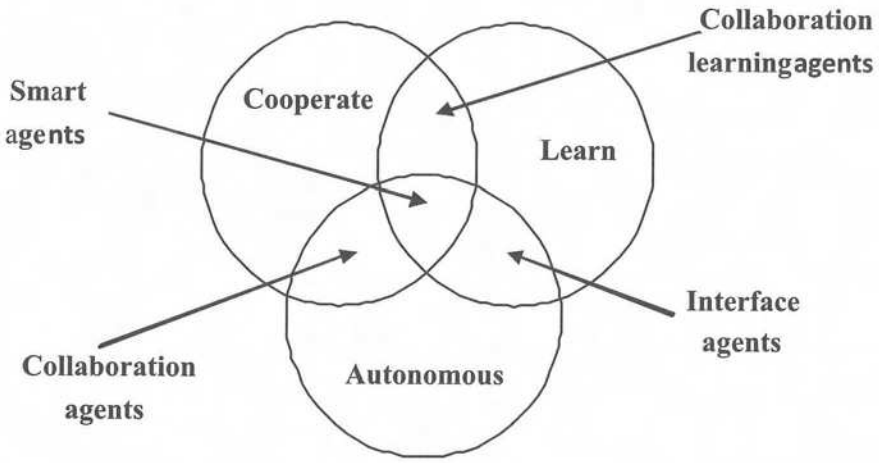

Fig.1.

The term "*agent*" describes a software abstraction, a concept, similar to Objects Oriented Program (*OOP*) terms such as methods, functions, and objects. The concept of an agent provides a convenient and powerful way to describe a complex software entity that is capable of acting with a certain degree of autonomy in order to accomplish tasks on behalf of its user. But unlike objects, which are defined in terms of *methods* and *attributes*, an agent is defined in terms of its behavior.

The definition of a software agent defines that the agent possesses the following minimal characteristics:

- *Delegation* - the agent performs a set of tasks on behalf of a user, or other agents, that are explicitly approved by the user;
- *Communication skills* - the agent needs to be able to interact with the user to receive task delegation instructions, and inform task status and completion through an agent user interface or through an agent communication language;
- *Autonomy* - the agent operates without direct intervention to the extent of the user's specified delegation;
- *Monitoring* - the agent needs to be able to monitor its environment in order to be able to perform tasks autonomously;
- *Actuation* - the agent needs to be able to affect its environment via an actuation mechanism for autonomous operation;
- *Intelligence* - the agent needs to be able to interpret the monitored events to make appropriate actuation decisions for autonomous operation.

The design of intelligent agents, or intelligent software agents is a branch of artificial intelligence research and have the following main abilities: *to adapt* and *to learn*[5].

Adaptation implies sensing the environment and reconfiguring in response. This can be achieved through the choice of alternative problem-solving-rules or algorithms, or through the discovery of problem solving strategies. Adaptation may also include other aspects of an agent's internal construction, such as recruiting processor or storage resources.

Learning may proceed through trial-and-error and then it implies a capability of introspection and analysis of behavior and success. Alternatively, learning may proceed by example and generalization and then it implies a capacity to abstract and generalize.

Various authors have proposed different definitions of agents, these commonly include concepts such as:

- *persistence* - code is not executed on demand but runs continuously and decides for itself when it should perform some activity;
- *autonomy* - agents have capabilities of task selection, prioritization, goal-directed behavior, decision-making without human intervention;
- *social ability* - agents are able to engage other components through some sort of communication and coordination, they may collaborate on a task;
- *reactivity* - agents perceive the context in which they operate and react to it appropriately.

The *Agent* concept is most useful as a tool to analyze systems, not as a prescription. The concepts mentioned above often relate well to the way we naturally think about complex tasks and thus agents can be useful to model such tasks.

Beginning from these above presentation, the usage agent motivation consists in their advantages, as follows:

- Large-scale, complex, distributed systems: understand, built, manage;
- Open and heterogeneous systems - build components independently;
- Distribution of resources;
- Distribution of expertise;
- Needs for personalization and customization;
- Interoperability of pre-existing systems / integration of legacy systems.

2. Classes and architecture of agents

The main classes of agents presented in literature are: single-agent; since-agents; *MAS*; mobile-agent.

Since agents are well suited to include their required resources in their description, they can be designed to be very loosely coupled and it becomes easy to have them executed as independent threads and on distributed processors. Thus they become *distributed agents* and the considerations of distributed computing apply. Agent code is particularly easy to implement in a distributed fashion and should scale well.

When several agents interact, they may form a *multi-agent system* (*MAS)*[6]. Characteristically such agents will not have all data or all methods available to achieve an objective and thus will have to collaborate with other agents. As with distributed agents, data is decentralized and execution is asynchronous.

Mobile agents [7] represent agent code that moves itself, including its execution state, on to another machine, to continue execution there. Agents can be used to gather system information, taking back-up of files by copying them in client-server paradigm, monitoring network throughput or to check resources availability and moderating the resource utilization of system by checking the services running on system.

A *fuzzy agent* [8] is a software agent that implements fuzzy logic. This software entity interacts with its environment through an adaptive rule-base and can therefore be considered as a type of intelligent agent.

The agent general architecture is very diverse. It can be more simple or more complex and strong depends on type and application. There are no generally accepted models because the concept of intelligent agent is a continuously evolving and, today, most used is open agent architecture. However, general structure of a simple-agent and multi-agent can be those shown in figures 2 and 3 respectively.

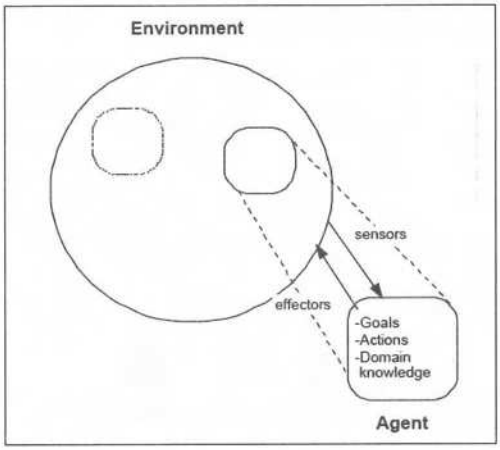

Fig. 2. A general single-agent framework

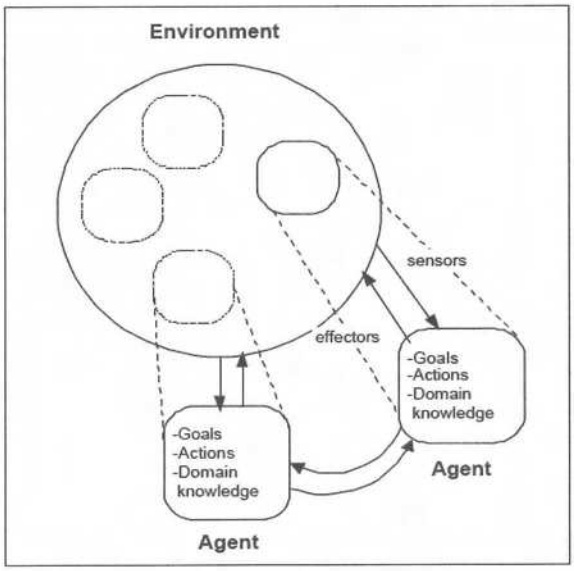

Fig. 3. A general multi-agent framework

3. Agent Typology

Today, the agent typology is very diverse. Figure 4 summarizes the types and lists the order in which they are surveyed. In particular, we would overview them in terms of *some* or *all* of following: their essential metaphors, hypotheses/goals, motivations, roles, prototypical examples, potential benefits, key challenges, and some other general issues about the particular agent type. We further briefly present some of these typologies.

Collaborative Agents [9] emphasis autonomy and cooperation with other agents in order to perform tasks for their owners. General characteristics of these agents include: autonomy, social ability, responsiveness and pro-activeness. The *motivation* for having collaborative agent systems may include following:

- To solve problems that are too large for a centralized single agent to do due to resource limitations or the sheer risk of having one centralized system;
- To allow for the interconnecting and interoperation of multiple existing legacy systems (expert systems, decision support systems, etc.);
- To provide solutions to inherently distributed problems, e.g. distributed sensor networks;
- To provide solutions which draw from distributed information sources;
- To provide solutions where the expertise is distributed, e.g. in health care provisioning;
- To enhance modularity, speed (due to parallelism), reliability (due to redundancy), flexibility and reusability at the knowledge level;
- To research into other issues, e.g. understanding interactions among human societies.

An example of Collaborative agent is CMUs Pleiades System

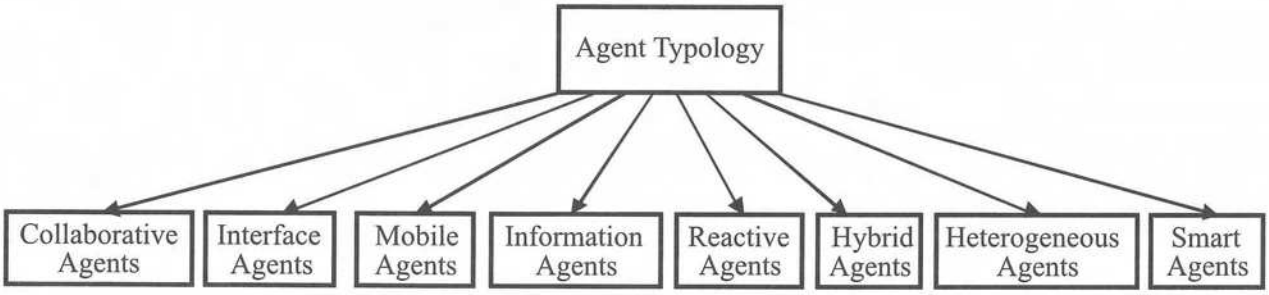

Fig. 4. The typology of agents

Interface Agents [10] emphasis autonomy and learning in order to perform tasks for their owners. Essentially, interface agents support and provide assistance, typically to an user learning to use a particular application such as a spreadsheet or an operating system. The user agent observes and monitors the actions taken by the user in the interface, learns new short-cuts, and suggests better ways of doing the task. Thus, the user agent acts as an autonomous personal assistant which cooperates with the user in accomplishing some task in the application.

Interface agents learn typically to better assist its user (learn from the user) in four ways:

- By observing and imitating the user;
- Through receiving positive and negative feedback from the user;
- By receiving explicit instructions from the user;
- By asking other agents for advice (learn from peers).

An example of *Interface agent* is RETSINA. This is a domain-independent and reusable infrastructure on which MAS, services, and components live, communicate, and interact.

Mobile Agents are computational software processes capable of roaming wide area networks (WANs) such as the *www*, interacting with foreign hosts, gathering information on behalf of its owner and coming back home having performed the duties set by its user. Agent code is referred to as mobile code. Agents can be used to gather system information, taking back-up of files by copying them in client-server paradigm, monitoring network throughput or to check resources availability and moderating the resource utilization of system by checking the services running on system.

So their motivation includes the following anticipated benefits:

- Reduced communication costs;
- Limited local resources;
- Easier coordination;
- Asynchronous computing;
- It provides a natural development environment for implementing;
- A flexible distributed computing architecture;
- Represent an opportunity for a radical and attractive rethinking of the design process.

Information/Internet Agents [10] perform the role of managing, manipulating or collating information from many distributed sources.

The *motivation* for developing information/internet agents is:

- Firstly, there is simply a yearning need/demand for tools to manage such information explosion. Everyone on the www would benefit from them in just the same way as they are benefiting from search facilitators such as Spiders, Lycos or Webcrawlers.
- Secondly, there are vast financial benefits to be gained.

Reactive Software Agents [11] is simple processing units that perceive and react to changes in their environment. Do not have a symbolic representation of the world and do not use complex symbolic reasoning. Intelligence is not a property of the active entity but it is distributed in the system, and steams as the result of the interaction between the many entities of the distributed structure and the environment.

Hybrid Agents [12] is a combination of collabora-tive, interface, mobile, internet and reactive agents types above presented. The key *hypothesis* for having hybrid agents or architectures is the belief that, for some application, the benefits accrued from having the combination of philosophies within a singular agent is greater than the gains obtained from the same agent based entirely on a singular philosophy.

Heterogeneous Agent Systems [13] unlike hybrid systems, refers to an integrated set-up of at least two or more agents which belong to two or more different agent

classes. A heterogeneous agent system may also contain one or more hybrid agents.

4. Conclusions

Distributed systems, especially those with many components or modules distributed wide geographic area are more easily monitored, tested and maintained with the help of digital society with self-healing. This solution is based, on the one hand, a decentralization of control, and on the other hand, the intelligent agent, implementing a decision making system distributed by self-healing. This leads to a significant number of advantages, among which we can mention:

- increase the flexibility and scalability of the maintenance which system allows to maintain the same quality;
- maintenance module capacity to reach deep into the lower levels of the system;
- higher speed due to parallelism and distribution of tasks;
- communication load reduction due to decentralized administration;
- easy maintenance and further development of the application, allowed the high level of modularity.

Among some future development directions of distributed intelligent agents, we can mention:

- Distributed Network Security Management Using Intelligent Agents;
- IntelligenTester – Test Sequence Optimization framework using Multi-Agents;
- Distributed Intelligent Agents for Holonic Control of Adaptive Manufacturing Systems.

5. References

[1] Popa Ilie, *Distributed intelligent agents*, ECAI 2009 – International Conference-3rd Edition, 3-5 Julay, 2009, Pitesti, Romania, Proceedings, pp.65.

[2] Carl Hewitt. *Viewing Control Structures as Patterns of Passing Messages* Journal of Artificial Intelligence. June 1977.

[3] John Sculley and John A. Byrne, *Odyssey: Pepsi to Apple*, New York : Harper & Row, ©1987.

[4] Nwana, H. S. (1996). *Software Agents: An Overview* **11** (3). Cambridge University Press, Knowledge Engineering Review. pp. 205–244. 1996. Software Agents: An Overview. Knowledge Engineering Review, Vol. 11, No. 3, 205-244, Cambridge University Press.

[5] Boy , G. A. 1997. Software Agents for Cooperative Learning. In Software Agents, ed J. M. Bradshaw . Menlo Park, Calif.: AAAI Press.

[6] Wooldridge, M., Introduction to MultiAgent Systems. Wiley, New York, 2002.

[7] David Kotz and Robert S. Gray, *Mobile Agents and the Future of the Internet*, Department of Computer Science / Thayer School of Engineering, Dartmouth College, Hanover, New Hampshire, 2010

[8] Ralph D. Badinelli, *An examination of fuzzy control systems in multi-agent networks*, Cambridge Academic Conference 2011, Virginia Tech,

[9] Yezdi Lashkari , Max Metral , Pattie Maes, *Collaborative Interface Agents*, Proceedings of the Twelfth National Conference on Artificial Intelligence, 2009.

[10] Shirley Lincicum, *Introduction to Interface Agents*, http://www.wou.edu/library/staff/lincicum, 2013

[11] Grant H. Kruger, Chao Chen, James M. Blum, Albert J. Shih, Kevin K. Tremper, *Reactive Software Agent Anesthesia Decision Support System*, Department of Mechanical Engineering, University of Michigan, Ann Arbor, 48109, USA

[12] Ferrie, Kevin; Spicknall, Mark H., *A Hybrid-Agent-Based Approach for Block Break Definition Using Fuzzy Logic*, Journal of Ship Production, Volume 21, Number 3,August 2005, pp. 146-159(14)

[13] Yuanshi Zheng, Long Wang, *Consensus of heterogeneous multi-agent systems without velocity measurements*, International Journal of Control, Volume 85, Issue 7, 2012.