Analysis of Internet Multi-Agent Based System for Zeus and SkeletonAgent Frameworks

DAVID CAMACHO, RICARDO ALER, CÉSAR CASTRO, JOSÉ M. MOLINA
Computer Science Department
Carlos III University of Madrid
Avda. Universidad 30. 28911 Leganés Madrid
SPAIN

Abstract: - Because of the evolution of the World Wide Web, the fields of intelligent agents and multi-agent systems have become a focus of interest. This intensive research has originated several frameworks, or toolkits, that are used by the researchers to develop Internet agent-based systems. Their aim is to help the designers and engineers to build complex systems based on the agent concept. This paper presents a brief description of two of those frameworks: ZEUS and SkeletonAgent. The aim of this paper is to compare them in a common domain: to search for news in several electronic newspapers. Because every one of those multi-agent toolkits has different features, the effort to design and build systems with them, as well as their behavior is expected to be different. The paper will measure the software effort to deploy the first agent and the global multi-agent system for each technology. It also studies the software reusability and the empirical performance evaluation.
Key-Words: - Internet Intelligent Systems, Web Agents, Intelligent Software Agents, Software Reusability, MultiAgent Frameworks Analysis.

1 Introduction

There are several computer research fields very active for designing and developing Internet-based systems. From Software Engineering [3] to Distributed Artificial Intelligence [6], they apply their theories and models to implement this kind of systems. Many different companies, research centers, laboratories, universities, etc… are designing, and implementing adaptive and intelligent systems that manage, with more or less success, the information stored in Internet. Currently, this increasing interest in the research and development of intelligent software agents, has originated the apparition of different toolkits, frameworks, libraries, etc… for programming agents and for defining the architecture of the system. This kind of systems, usually named Multi-Agent Systems (MAS), is a very active research field [6,10]. There are both commercial and research products. Products like AgentBuilder, Gossip (Tryllian), Intelligent Agent Factory, Jumping Beans belong to the former category whereas Jade, Jafmas, MadKit, ***ZEUS***[1] belong to the latter.

When a designer needs to build his/her own MAS for a particular task, it would be useful to have guidelines to select the most appropriate [8]. The main goal of this paper is to evaluate the software effort to build an Internet-based system when some of those previous frameworks are used in a specific common domain.

This paper is divided into 5 sections. Section 2 presents a brief description of the frameworks analyzed. Section 3 describes the MAS implemented to test the previous frameworks. Section 4 presents the experimental evaluation of the different toolkits when the MAS built are used in a specific domain. And finally, section 5 presents the conclusions of the paper.

2 Zeus and SkeletonAgent Multi-Agent Frameworks

Two different frameworks have been considering in this paper:

- ***ZEUS***. Developed by BT Laboratories in the Advanced Applications & Technology Department.(http://www.labs.bt.com/projects/agents/zeus/index.htm).

[1] It is possible to find more information about these (and other) tools in:
- www.multiagent.com/Software/Tools_for_building_MASs/index.html
- agents.umbc.edu/Companies/index.shtml

- ***SkeletonAgent***. Developed by the Systems: Complex and Adaptive Laboratory (Scalab). (http:// scalab.uc3m.es/~agente).

Next subsections will describe the main features for each framework.

2.1 The ZEUS Agent Building Toolkit

The goal of ZEUS project [1,5], is to facilitate the rapid development of new multi-agent applications by abstracting into a toolkit the common principles and components underlying some existing multi-agent systems.

ZEUS makes several assumptions about the agents to be built. Agents are deliberative, goal-directed and rational. They are always truthful when dealing with other agents. They can have many goals and can engage in a variety of tasks. Finally, they are Temporally continuous.

The main characteristics in the building process of a multi-agent system, in ZEUS, could be summarized in:

1. This toolkit has user-friendly graphical interfaces that allow programming and debugging the agents of the system.
2. ZEUS provides to engineers a complete reference about the architecture of the agents and the specifications about the multi-agents that could be developed within the framework.

The multi-agent architecture defined by ZEUS is shown in Fig.1. It uses the following kinds of agents:

- Agent Name Server (ANS). Any agent in the system can request to the ANS for the address of other agents.
- Facilitator Agent. These agents receive and reply to queries from agents about the abilities of other agents. They work by periodically querying all the agents in the society about their abilities and storing the returned information in their Acquaintance Database.
- Generic Agents. These agents perform different tasks and allow the system to achieve the designed goals.

The inter-agent communication in ZEUS language is used to communicate with the Agent Name Server, the Facilitator and other agents. The communication requires a shared representation and understanding of common domain concepts, i.e., a common ontology (ZEUS provides tools to create new ontologies). The communication protocol used by the agents is TCP/IP and the language supported is FIPA ACL [8].

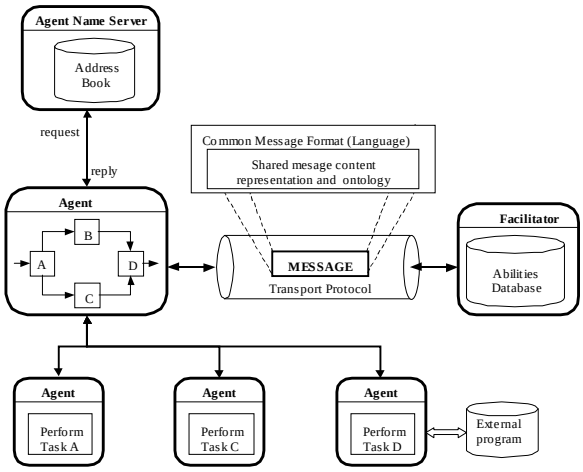

Fig. 1: Multi-agent architecture defined by ZEUS to implement a system.

2.2 SkeletonAgent

This framework is built by a set of Java libraries. SkeletonAgent tries to wrap the "*agent concept*" into a set of reusable libraries [5]. Once the agents have been built by means of the libraries, it is possible to develop the MAS that will be used to solve the problems. SkeletonAgent requires a set of predefined agents:

- Control agents: the SkeletonAgent architecture is based in the "***team***" concept. All the agents in the system belong to one team, which is managed by the CoachAgent. All of the CoachAgents are managed by the ManagerAgent. The control agents manage different problems, like the insertion or deletion of them in the system.
- Execution agents: these agents are involved in solving the task. It is possible to define different kinds of agents, like UserAgents (that deals with the users), WebAgents (specialized in retrieving information from the Web), etc…

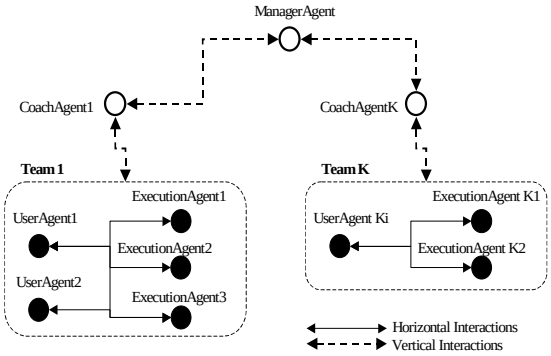

Fig. 2: Multi-agent architecture defined by SkeletonAgent to implement a System.

The architecture provided by SkeletonAgent (see Fig.2) divides the management problem in a MAS in two ways:

- First, with a vertical control of the insertion and deletion problems
- Second, with a horizontal relation among the execution agents that belong to a specific team.

This architecture tries to minimize the problems when any agent in the system is unreachable. At this point, its main disadvantages are:

- The framework does not have graphical programming interfaces, and the debugging of the agents is hard.
- There is a poor documentation about the reusable libraries.

The concept of agent is well encapsulated and it is possible to reuse a lot of code. The inter-agent communication in Skeleton implements a reduced version of KQML [7]. This language provides to the system agents flexible protocols that allows them to coordinate and to share information and tasks in a cooperative way. The communication protocol used by the agents is TCP/IP like in ZEUS.

3 *SimpleNews*: A Meta-Search Web Engine

The aim of this section is to present the generic description of a MAS, named SimpleNews. This MAS will be implemented in the two frameworks so that they can be compared.

3.1 SimpleNews: MAS Architecture

SimpleNews is a simple meta-search engine that allows, by means of a UserAgent, to search for news in a set of electronic newspapers[2]. In this paper a very simple topology was used (see Fig. 3), where all of the Web agents solve the queries sent by the UserAgent. The *SimpleNews* engine is built using a set of specialized agents that are able to retrieve information from a particular electronic newspaper. SimpleNews can retrieve from all the previous electronic sources, filter the different answers from the specialized agents and show them to the user. The meta-search engine includes a UserAgent and six specialized WebAgents. The Web specialized agent can be classified in the next categories. Two of them offer information supplied by newspaper specialized in *financial information* (*Expansion_WebAgent*, *CincoDias_WebAgent*). Two in *sports information* (*Marca_WebAgent*, *Futvol_WebAgent*). And finally two agents specialized in *general information* (*ElPais_WebAgent*, *ElMundo_WebAgent*).

This multi-agent system uses the six previous specialized agents and a UserAgent that make up the meta-search engine. The UserAgent has a graphical user interface (Fig.4) for making queries. The number of solutions requested, and the agents that will be consulted. The interface used by this agent allows to the user to know: The actual state of the agents (active, suspended, searching, finished), and the messages and contents sent between the agents. Finally the entire requests retrieved by the agents are analyzed (only different requests are taken into account) and the UserAgent builds an HTML file, which is subsequently displayed to the user.

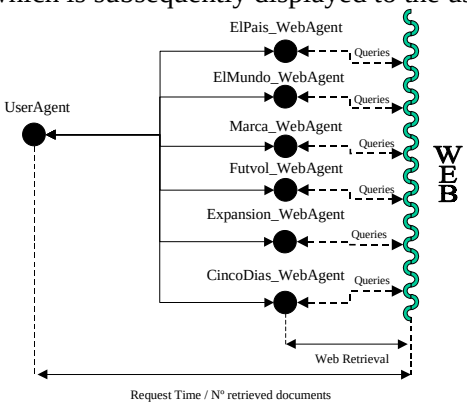

Fig. 3: Multi-agent architecture of the meta-search news engine.

Fig. 4: UserAgent Interface for interacting with the specialized Web Agents

[2] All the electronic newspapers used are Spanish newspapers: www.expansion.es, www.cincodias.es, www.marca.es, www.futvol.com, www.elpais.es, www.elmundo.es.

3.2 Meta-search engine differences

Because the previous meta-search engine has been built using the different multi-agent frameworks, they have different features. All of them need one UserAgent and the six Web agents, but the different architectures may need to use other different agents to work correctly. The main differences are summarized as follows:

- **ZEUS** meta-search engine needs to use the *ANServer agent* that implements the yellow pages for all the connected agents, and a *Facilitator* agent that provides the abilities of the agents. Other visual agents could be used like: the *Agent Viewer*, *Society Viewer*, etc… But for the experimental evaluation only the essential agents will be used.
- **SkeletonAgent**, as we describe briefly in the previous section, two control agents are needed: the ManagerAgent and the CoachAgent. Only one team will be used, so only two control agents will be necessary to test SimpleNews.

4 Multiagent System Analysis

The purpose of this section is to compare several characteristics from the previous frameworks, to do that SimpleNews will be used like a *benchmark* system. The characteristics that will be evaluated should be summarized in:

- *Development time*: it includes both the design and implementation effort, for the development of the first agent, and for the whole MAS.
- *Reusability of the Software*: the lines of code (LOC) and the number of classes which can be reused will be measured.
- *Performance evaluation*: a simple performance evaluation of SimpleNews will be done for each framework.

Different evaluations have been obtained from the experience of several undergraduate students that built the multi-agent system.

4.1 Development time

This subsection analyzes the time effort to implement (for the different development phases considered) SimpleNews agents.

4.1.1 Development Time: The First Agent

The measured phases considered were:

- *Analysis and Design*. To implement the SimpleNews multi-agent architecture for each technology.
- *Code generation for the first agent*. It measures the time required to modify the agents provided by the tools, so that it is adapted to the required MAS. For instance, the code necessary to extract the information from the Web, or to build the graphical interfaces for the UserAgent.
- I*ntegration with external code*. It evaluates the effort to integrate previous code (which implements the specific skills) in the agents.
- *Tests and debugging.*

Table 1 shows the measured time for the first agent implementation using ZEUS and SkeletonAgent frameworks respectively.

Phase	ZEUS	SkeletonAgent
Analysis and Design	10h	10h
Code generation	12h	30h
Integration of external code	12h	30h
Test and debugging	6h	15h
TOTAL	**40h**	**85h**

Table1. Development Time (first agent) for ZEUS and SkeletonAgent

As Table 1 shows, the development time for the first agent is more than twice for SkeletonAgent technology. This is due to the poor documentation that the latter framework provides.

4.1.2 Time Development: Rest of the Agents

When the first agent is built and is fully deployed, other characteristics were measured. When the new agents appear in the multi-agent system it is necessary:

- To modify the graphical UserAgent interface, now it is necessary to display the information about the new agents.
- To generate the code (specific skills or abilities) for the new agents.
- To integrate the new code.

Table 2 shows the measured time for the development of the new agents in SimpleNews.

Phase	ZEUS	SkeletonAgent
Modification of GUI (UserAgent)	0.5h	0.5h
Modification of UserAgent	1h	1h
New agent code generation	0.5h	2h
Integration of external code	2h	2.5h
TOTAL	**4h**	**6h**

Table2. Development Time (new agents) for ZEUS and SkeletonAgent.

The previous tables show how when the different technologies are understood by the engineers, the time effort to implement similar agents reduces to 10 percent or less.

4.2 Software Reusability

This subsection measures the LOC (lines of code), the new Java classes and the reusable classes that will be necessary to implement the UserAgent and the six specialized Web agents. These characteristics measure the programming effort for the technologies evaluated. Figures 5 and 6 show the LOC and the number of reused and generated classes for the seven SimpleNews agents.

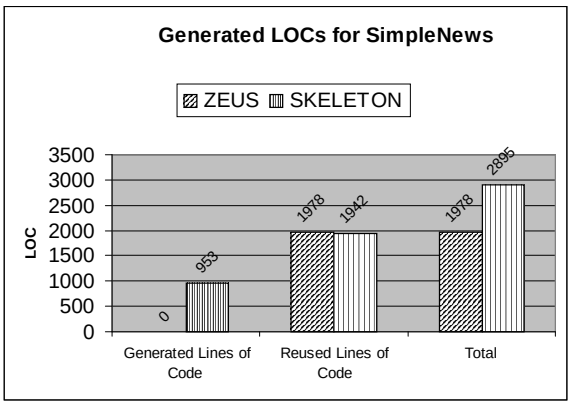

Fig.5. LOC comparation for the seven agents implemented.

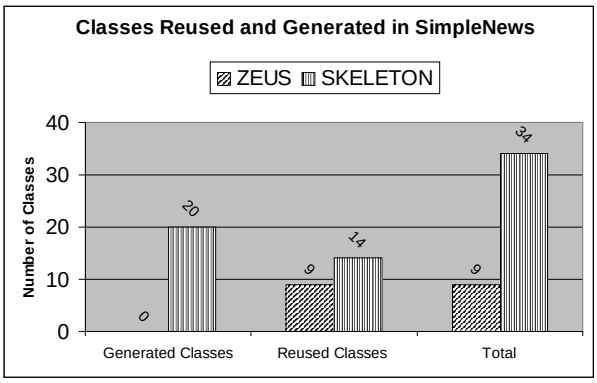

Fig.6. Implemented and reused Java classes for the seven agents implemented.

4.3 Performance Evaluation

The following performance measures will be taken into account:

- *Number of requested documents*: from only one document to fifty documents to any active agent in the system (1, 5, 10, 15, 20, 30, 40 and 50 news).
- *Request time*: this is the time the user has to wait.
- *Number of articles retrieved.*

4.3.1 Empirical Evaluation

The experiments in this section display the average time to answer questions and the number of articles retrieved for each of the architectures.

Figures 7 and 9 display the performance for each architecture when any (but only one) of the Web agents implemented is used. Figures 8 and 10 show the MAS performance when the number of Web agents grows.

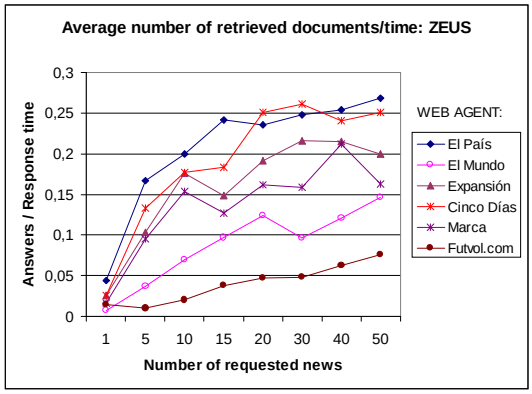

Fig. 7. Performance of each Web agent in ZEUS.

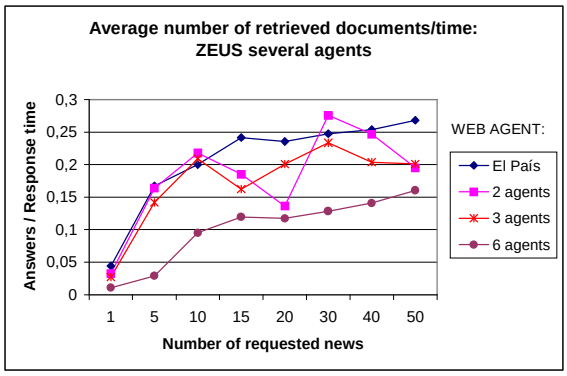

Fig. 8. Performance with several Web agents in ZEUS.

From Figures 7 and 8, it can be concluded that the ZEUS architecture has an irregular behavior when more than two Web agents are used. This is due to ZEUS communication technology. When a Web agent finds in the retrieving process any non-allowed character, the communication fails and the system forgets those results.

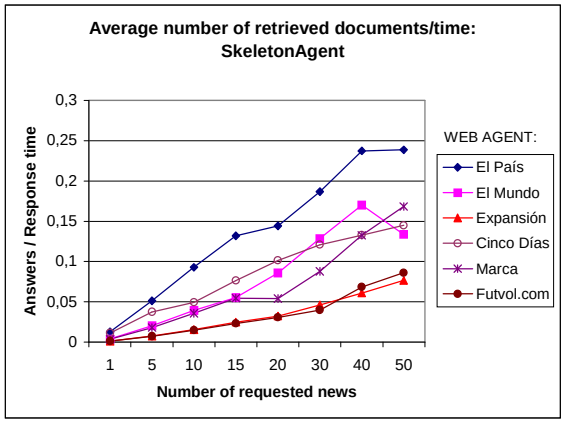

Fig. 9. Performance for each Web agent in SkeletonAgent.

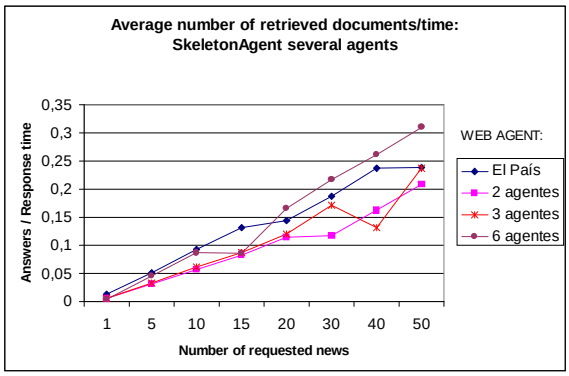

Fig. 10. Performance with several Web agents in SkeletonAgent.

Figures 9 and 10 show that the evaluation of MAS built with SkeletonAgent has a similar performance to ZEUS. However, its behaviour is more regular (linear) because the fault tolerance to possible mistakes in those agents is more robust.

5 Results and Conclusions

Although there are many different frameworks to build multi-agent systems, not much effort has been spent in comparing them empirically to help engineers choose the most appropriate toolkit for a given task. In this paper, we present the SimpleNews domain (retrieving news from Web newspaper servers). By using it as common ground, two multi-agent frameworks (ZEUS and SkeletonAgent) have been tested in a common domain:. Our conclusions are that:

1. The ZEUS architecture allows a quick implementation of MAS and provides excellent helping tools and documentation. However the agents built show a low fault tolerance.
2. SkeletonAgent has a similar performance to Zeus, but displays a more regular (linear) behavior, and a good fault tolerance. However, currently development is slower because of its poor documentation and development support tools.

In the future, we will use the SimpleNews domain to compare other architectures. We encourage other researchers to test their architectures in this domain to have comparative results in a real task.

References:

[1] Azarmi N., Thompson S. ZEUS: A Toolkit for Building Multi-Agent Systems. Proceedings of fifth annual Embracing Complexity conference, Paris April, 2000.

[2] F. Bellifemine, A. Poggi, G.Rimassa. JADE-A FIPA-Compliant agent framework. Proceedings of the 4th International Conference on the Practical Applications of Agents and Multi-Agent Systems (PAAM-99). pp. 97--108, 1999.

[3] Berners-Lee, T., Hendler, J., Lassila, O. The Semantic Web. Scienfic American, May2001.

[4] Camacho D., Molina J.M., Borrajo D., Aler R. *MAPWEB*: Cooperation between Planning Agents and Web Agents. Information & Security: An International Journal. Volume 7. 2001

[5] Collis J., Ndumu D., Nwana H., Lee L.. The Zeus Agent Building Tool-Kit. BT Technology Journal 16(3), July 1998, p60-68.

[6] Ferber J. Multi-agent systems: an introduction to distributed artificial intelligence. Ed. by Addison Wesley, 1999.

[7] Finin, T., Fritson, R., Mackay D., McEntire, R. KQML as an agent communication language. In Proceedings of the Third International Conference on Information and Knowledge Management (CIKM94), pages 456-463, 1994.

[8] Foundation for Intelligent Agents (FIPA). Specifications 1998. Available from http://www.fipa.org/specifications.

[9] Wagner T., Rana O. Infrastructure for Agents, Multi-Agent Systems, and Scalable Multi-Agent Systems. International Workshop on Infrastructure for Scalable Multi-Agent Systems. Ed. by Springer-Verlag. Lecture Notes in Artificial Intelligence (LNAI 1887). Barcelona, Spain,2000.

[10] Wooldridge M. Multi-Agent Systems: an Introduction. Ed. by John Wiley, 2001.