Management & Marketing (2008) Vol. 3, No. 4, pp. 73-80.

ADAPTIVE MULTIAGENT SYSTEM FOR SEISMIC EMERGENCY MANAGEMENT

Florin LEON, Mihai Horia ZAHARIA, Gabriela M. ATANASIU
„Gh. Asachi" Technical University of Iaşi

Abstract. *Presently, most multiagent frameworks are typically programmed in Java. Since the JADE platform has been recently ported to .NET, we used it to create an adaptive multiagent system where the knowledge base of the agents is managed using the CLIPS language, also called from .NET. The multiagent system is applied to create seismic risk scenarios, simulations of emergency situations, in which different parties, modeled as adaptive agents, interact and cooperate.*

Key words: adaptive systems, risk management, seisms.

1. Introduction

The modern societies have many problems related to risk management of metropolitan areas. Due to the inherent number of inhabitants, any larger disaster either natural or artificial can have a deep impact on society. The information society gives the instruments than can reduce this impact. This is accomplished by using both prevention plans and disaster management plans. Probably, the most efficient way of minimizing losses is to create a model of the reaction, response or of the disaster itself. At community level, tool already exist to handle some problems. Unfortunately, great natural disasters like earthquakes have an impact at national or international levels. The only way of creating a global image from discrete images is to analyze all existent data about the disasters. However, the quantity of information is large and modern computing has two approaches to solve this problem. The first is a centralized manner that involves data to be stored into a few computing centers with great computing power. These centers usually have clusters of supercomputers which are very expensive. The second is given by the Internet growth in both quality and quantity of handled datasets, it is the distributed approach. Here we have many models but the most efficient ones from the point of view of the involved costs is to use intelligent agents (Atanasiu, Leon & Zaharia, 2008). They can travel to any point that has the needed data and will make exchanges of knowledge. As a result, even the countries with medium developed economies can use it to try to solve a part of this great problem.

According to Wooldridge (2000), an *agent* is a computer system that is *situated* in its environment and is capable of *autonomous* action in order to meet its design objectives.

Agent Oriented Programming (AOP) is a fairly new programming paradigm that supports a societal view of computation. In AOP, objects known as agents interact

to achieve individual goals. Agents can exist in a structure as complex as a global internet or one as simple as a module of a common program. Agents can be autonomous entities, deciding their next step without the interference of a user, or they can be controllable, serving as an intermediary between the user and another agent.

One of the differences between traditional OOP and AOP is *coupling*. Interfaces are a way of improving good design which can decrease coupling, but usually the call of a specific method is made with specific arguments, thereby coupling the two classes in code. The same method invocation has to occur for agents, but with one difference: there is effectively just one method on each agent, with one argument, containing all the information of the call.

Enhanced decoupling in an AO environment also comes from a semantics-related reason. When an agent interacts with another, the recipient of the "call" is free to act as it wishes and should not be expected to blindly perform a request, like objects usually do.

Intelligent agents retain the properties of autonomous agents, and in addition show a so-called „flexible" behavior (Wooldridge & Jennings, 1995):

- *reactivity* (the ability to perceive their environment, and respond in a timely fashion to changes that occur in it);
- *pro-activeness* (the ability to exhibit goal-directed behavior by taking the initiative);
- *social ability* (to interact with other agents and possibly humans).

2. JADE.NET Framework

JADE (Java Agent DEvelopment) framework has become one of the most wide spread agent-oriented middleware, based on the peer-to-peer intelligent autonomous agent approach. It is a completely distributed middleware system with a flexible infrastructure and its goal is to facilitate the development of complete agent-based applications by means of a run-time environment implementing the life-cycle support features required by agents, and the core logic of agents themselves (Bellifemine, Caire & Greenwood, 2007).

FIPA (the Foundation for Intelligent Physical Agents) is an IEEE Computer Society standards organization that promotes agent-based technology and the interoperability of its standards with other technologies. The FIPA standard is based on the principle that only the external behavior of system components should be specified, leaving internal architecture and implementation details to the developers of individual platforms. This ensures the interoperation between compliant platforms.

JADE provides complete compatibility with the FIPA specifications (communication, management and architecture) that offer the framework within which agents can exist, operate and communicate, while adopting a unique internal

architecture and implementation of key agent services (Bellifemine, Caire & Greenwood, 2007).

LEAP (Lightweight Extensible Agent Platform) is an extension of JADE to enable it to run on wireless devices and PDAs such as cell phones and palm computers. S. Rusitschka has ported the LEAP version of JADE to .NET using Visual J#. Therefore, the resulting DLL can be used from any .NET language, such as Visual C#, to build multiagent systems.

3. Clips

CLIPS (C Language Integrated Production System) is an expert system shell developed by the Software Technology Branch, NASA. It is thus specifically designed to facilitate the development of software to model human knowledge or expertise (Giarratano, 2002).

There are three ways of representing knowledge in CLIPS: *rules* (which are primarily intended for heuristic knowledge based on experience), *generic functions* (which are primarily intended for procedural knowledge), and *objects-oriented programming* (also intended for procedural knowledge, supporting classes, message-handlers, abstraction, encapsulation, inheritance, and polymorphism).

The CLIPS shell provides the basic elements of an expert system: the *fact list* and *instance list*, which represent the global memory for data, the *knowledge base* or the *rule base* that contains all the rules, and an *inference engine* that controls the overall execution of rules, based on an implementation of the Rete algorithm (Forgy, 1982). Using the inference engine, rules match patterns on objects and facts.

4. The Model of an Adaptive Multiagent System

In JADE, a custom agent class is derived from the provided *Agent* base class. Agents run in the so-called *containers*, as shown in Figure 1. The methods that are automatically called by the platform during the agent lifecycle, such as *setup()*, must be overridden. For each agent a *behavior* can be defined, which describes the actions that the agent performs. Agent communication is implemented using *ACL messages*.

Each agent defined in this way has its own knowledge base corresponding to its type and previous experience. We chose to encode it with the CLIPS language due to the high expressive power of representation using facts (data) and rules (conditions and actions). Beside its built-in pattern matching capabilities, another advantage is the simplicity with which a CLIPS program *can modify itself*. There are two possible types of changes. First, changes in facts – this is the usual way of controlling the workflow of the program: adding (*asserting*) new facts and deleting (*retracting*) old ones.

```
(defrule add-fact
=>
(assert (data 0)))

(defrule delete-fact
?f <- (data ?)
=>
(retract ?f))
```

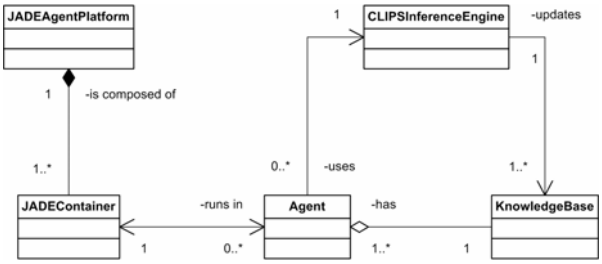

Figure 1. The UML diagram of the adaptive multiagent system

The second type refers to the changes in rules themselves. Most programming languages have a set of functions that remain unchanged during the execution of the program. Its behavior is given by the particular values of the variables, which define the program states. In CLIPS, rules can be dynamically added and deleted, i.e. the program can dynamically change itself at run-time. To add a new rule the following method can be used:

```
(defrule add-rule
(rule-text ?rt)
=>
(build ?rt))
```

The new rule is introduced as a string in a fact, and then it is converted into a rule *per-se* using the *build* function:

(assert (rule-text „(defrule dynamic-rule => (printout t \„The dynamically created rule has fired\" crlf))"))

Also, there is a simple function, *undefrule*, to delete a rule:

```
(defrule delete-rule
(rule-name ?rn)
=>
(undefrule ?rn))
```

The name of the rule to be deleted can be given as a fact:

(assert (rule-name dynamic-rule))

These features make CLIPS a good choice for implementing adaptive agents that can take into account the new information regarding a situation but also the new ways of reaching the same goal, the changes in behavior that can occur, for example, in case of a catastrophic seismic event.

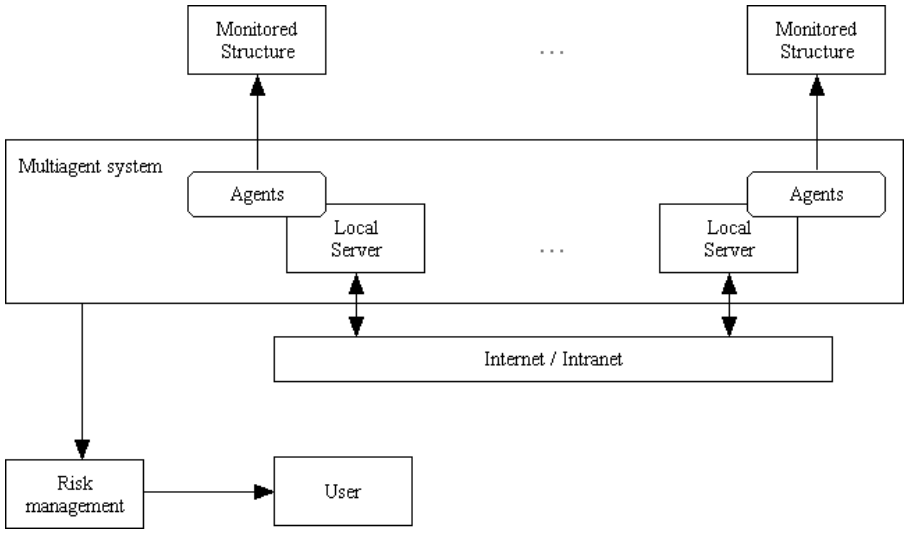

Figure 2. The general architecture of the agent-based risk management system

5. Creating Scenarios for Seismic Emergency Management

A fundamental principle of risk assessment is that risk due to natural hazards such as earthquakes, hurricanes and floods is location dependent. The process of risk assessment involves hazard evaluation and vulnerability analysis of built infrastructure. The probability of earthquake occurrence varies depending on location, and local site conditions also play a vital role in determining the intensity of the earthquake. Seismic risk scenarios are intended to provide local, state and regional officials with the tools necessary to assess the risks from earthquakes. GIS (Geographical Information Systems) can be used for all the disaster prevention phases, and especially for physical planning in the mitigation phase, by taking the disaster risk into consideration (Atanasiu, Brătianu & Leon, 2008; Atanasiu, Leon, et al., 2008).

The general architecture of the agent-based risk management system is presented in Figure 2. For the simulations of the seismic risk scenarios, three classes of agents were used:

- vulnerability assessment agents;
- transport network agents;
- emergency response agents.

5.1. Agents for Vulnerability Assessment

A key concept for the evaluation of vulnerability, developed primarily for seismic events, is the structural *fragility curve*. Fragility curves can be used for modeling the effects of a possible natural hazard event on structures, as a method of analyzing the behavior of built existing infrastructure in urban areas under different hazard scenarios. The fragility curve is defined as the mathematical expression that relates the conditional probability of reaching or exceeding a particular structural safety level, given a particular level of the hazard. HAZUS (FEMA, 1999) specifies the safety levels in terms of four damage states: slight, moderate, severe, and complete damage state. The GIS map of the vulnerability of buildings in a urban sample of Iaşi city, Romania, is displayed in Figure 3. Given a certain seismic event, green stands for minor damage, blue means moderate damage, yellow represents major damage, and red stands for near-collapse.

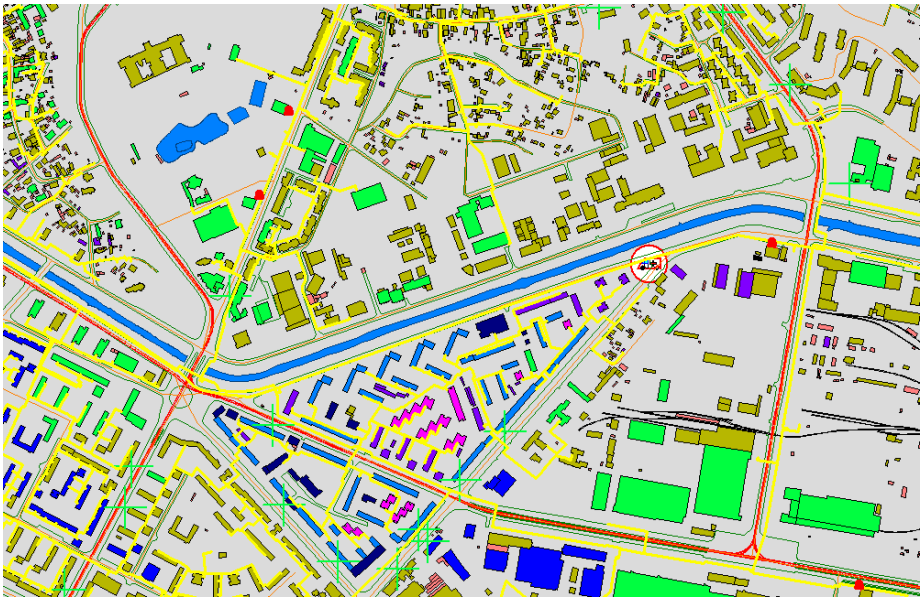

Figure 3. Using GIS mapping for the assessment of building vulnerability

The vulnerability assessment agents have information related to the fragility curves of buildings and critical infrastructure. These can be considered as starting points for the beginning of the emergency operations. As more real data come from the agents involved in emergency response, the information regarding the damage of the buildings is updated and the knowledge base of the agents is changed.

5.2. Agents for Transport Network

Similarly to the vulnerability assessment agents, the transport network agents evaluate the status of the transport channels, such as roads. They compute and communicate the shortest path between two locations, taking into account the current state of degradation. Beside the updated on-site information, their algorithms may be changed as well in order to offer better solutions, which is accomplished by dynamically changing their rules.

5.3. Agents for Emergency Response

Emergency response agents model the intervention forces that are active during emergency: fire squads, police cars, civil protection teams. They rely on the information from the vulnerability assessment and the transport network agents, and compute a risk map by identifying the critical areas. Such information can be the geographical locations of gas stations and gas pipelines, that may be affected after an earthquake (figure 3). The possible changes that can occur for an emergency response agent are both in the facts (information received from other agents and perceived on-site), and rules (better ways of action in emergency situations).

Figure 4. GIS visualization of urban risk control facilities

This class of agents provide in turn actual information to the other two classes of agents based on their field experience. A cooperation mechanism for the subtypes of the agents in this class may be devised. They may also exchange rules of actions, based on a common ontology that defines the goals of the agents.

6. Conclusions

This paper describes the model of a multi-agent system with complex and heterogeneous agents. Each type of agent was designed with an adaptive knowledge-based and different goals. An alternative approach to changing the behavior of the agents is to use inductive learning or reinforcement learning methods. The underlying learnt models have to be transformed into proper rules for the inference engine, as a pre-processing step. In this respect, the representation of the decision trees, reinforcement learning matrices and generalized exemplars (Leon, 2006; Leon, Atanasiu & Gâlea, 2006) can be converted into first-order logic rules. The multi-agent approach is particularly useful for simulating the emergency response actions for a set of seismic scenarios linked with urban locations, organized in a modular and cooperative manner.

References

Atanasiu, G.M., Brătianu, C., Leon, F. (2008) *Decision based risk assessment model for existing damaged infrastructure, application to Iasi city*, in J. Rostum, V. November, J. Vatn (eds.) - COST Action C19, Proactive Crisis Management of Urban Infrastructure, pp. 211-218, SINTEF Byggforsk, COST Office, European Science Foundation

Atanasiu, G.M., Leon, F., Popa, B.F., Doniga, C. (2008) *A Computational Approach for the Assessment of Seismic Vulnerability Based on the National Data Infrastructure*, *Advanced Materials Research*, **33-37**: pp. 789-794, Trans Tech Publications, Switzerland

Atanasiu, G.M., Leon, F., Zaharia, M. H. (2008) *Intelligent Agents for Life Cycle Management of Structures and Infrastructures in Seismic Areas*, in F. Biondini, D. M. Frangopol (eds.) - Life-Cycle Civil Engineering, pp. 921-928, CRC Press, Taylor & Francis Group

Bellifemine, F.L., Caire, G., Greenwood, D. (2007) *Developing Multi-Agent Systems with JADE, Wiley Series in Agent Technology*

FEMA (1999) *HAZUS, Earthquake Loss Estimation Methodology*, Technical Manual, National Institute of Building Sciences for the Federal Emergency Management Agency

Forgy, C. (1982) *Rete: A Fast Algorithm for the Many Pattern/Many Object Pattern Match Problem*, *Artificial Intelligence*, **19**: pp. 17-37

Giarratano, J.C. (2002) *CLIPS User's Guide*, retrieved from: www.ghg.net/clips/download/documentation/usrguide.pdf

Leon, F. (2006) *Intelligent Agents with Cognitive Capabilities*, Tehnopress, Iaşi, Romania

Leon, F., Atanasiu, G.M., Gâlea, D. (2006) *Using Data Mining Techniques for the Management of Seismic Vulnerability*, Key Engineering Materials, **326-328**: pp. 501-504, Trans Tech Publications, Switzerland

Wooldridge, M. (2000) *Intelligent Agents*, in G. Weiß (ed.), Multiagent Systems – A Modern Approach to Distributed Artificial Intelligence, The MIT Press, Cambridge, Massachusetts.

Wooldridge, M., Jennings, N.R. (1995) *Intelligent agents: Theory and practice*, The Knowledge Engineering Review, 10(2), pp. 115-152