Proceedings of the Fourth IASTED International Conference
COMPUTATIONAL INTELLIGENCE
July 4-6, 2005, Calgary, Alberta, Canada

APPLICATION AND TESTING OF A COUGAAR AGENT-BASED ARCHITECTURE

Mike Emery, John Paxton, Rick Donovan
Montana State University, Montana State University, Montana Tech
Bozeman MT, Bozeman MT, Butte MT
USA, USA, USA
emery@cs.montana.edu, paxton@cs.montana.edu, rdonovan@mtech.edu

ABSTRACT
The maturation of multi-agent systems permits increasingly sensitive and security related problems to be addressed. In particular, the Cougaar architecture has demonstrated itself to possess significant scalability and survivability features. In this paper, a wireless door-entry security system built using Cougaar is discussed. The key contribution of this paper is to begin a discussion regarding how such a system can be tested for reliability and accuracy. Testing multi-agent systems is a very beneficial, yet challenging endeavour.

KEY WORDS
Agent Based Architectures, Cougaar, Testing

1. Introduction & Problem Statement

The work described in this paper is part of an effort by The Rocky Mountain Agile Virtual Enterprises Technical Development Center (RAVE) [1]. The goal of RAVE, directed by Rick Donovan, is to connect universities with light and moderate manufacturers in the state of Montana in order to cooperate on large projects beyond the scope of a single entity.

RAVE and its partners are funded to develop an agent based approach to detecting the presence of unauthorized personnel in the Secure Area of regional airports. The solution being constructed requires airport personnel to carry biometrically enabled (fingerprint ID) radio frequency identification cards (RFID cards). Intelligent sensor nodes (ISNs) will be deployed on various doors throughout the airport that lead to secure areas. When a card is activated, the closest door receiving a strong enough signal will open.

Currently, the hardware that is required for a solution to the above stated problem is being investigated and developed. While the hardware is being developed, a software simulation has been constructed in order to test whether the intelligent sensor nodes (ISNs) operate accurately. The remainder of this paper is organized as follows. In section two, the underlying agent architecture for the proposed solution, Cougaar, is introduced. In section three, the underlying system architecture for the solution is explained. In section four, validation techniques are covered. Finally, in section five, future work is discussed.

2. Cougaar Overview

Cougaar, an abbreviation of COGnitive Agent Architecture, is a Java-based agent architecture that provides a survivable base on which an application can be built. A survivable base is one that can withstand man-made hostile environments without suffering an abortive impairment of its ability to accomplish its designated mission. Cougaar is capable of handling large-scale distributed applications and was developed as part of the solution to the DARPA UltraLog project [2], a distributed logistics application consisting of more than 1000 agents distributed over 100 hosts. The resulting system is completely open-source and covered by a BSD-equivalent license [3]. Cougaar was selected over other agent architectures [4] due to its affiliation with DARPA, the fact that it is open source, the importance of security for this particular project and its general applicability to the problem at hand.

Each Cougaar node is composed of support services and at least one agent with its own component plug-ins, as illustrated in Figure 1. These plug-ins communicate using a provided blackboard which supports standard publish/subscribe semantics [5, 6].

Agents communicate with each other using a built-in, asynchronous message-passing protocol called the Message Transport Service (MTS) [7]. Agents may also be grouped into a community based on a common purpose or function. By combining a logical grouping with the provided Community Service, additional functionality, such as broadcast messaging to all members of a given community is possible [5, 6].

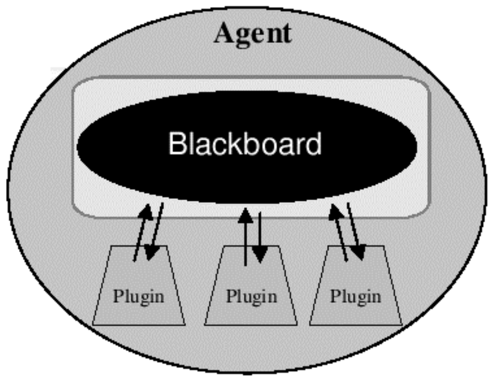

Figure 1. Cougaar Agent Internal Structure

3. System Architecture

These features of Cougaar make it an attractive base system upon which to develop this application. One Cougaar agent is assigned to each intelligent sensor node (ISN).

In the simulation, there are two key assumptions regarding the hardware. First, each restricted access door has an ISN associated with it. Second, when an RFID card is activated near an ISN, the closest ISN is the intended target.

Using these two assumptions, the system functions as follows. First, a person carrying an RFID card approaches a restricted access door and activates the RFID card. Second, each ISN within range picks up the intensity of the signal and adds a timestamp, provided that the intensity is above some minimal threshold. Third, each ISN that picks up a signal broadcasts the signal that it received. Fourth, after a certain amount of time elapses, each ISN determines whether it received the strongest RFID signal. Fifth, the ISN that received the strongest signal will open its restricted access door, provided that the authentication information is sufficient.

In the current simulation, all agents are contained in a single computer. As Cougaar is not limited by the hardware on which it runs, an entire society can function normally while contained within a single system. This allows for easier initial development and testing.

3.1 Node Design

Each ISN is an independent Cougaar agent with several plug-ins. Cougaar provides the base system, behaving much like an operating system in terms of functionality. Additional capabilities and behaviours must be added in the form of plug-ins. To modularize the solution, three plug-ins were developed.

The Sensor Plug-in is responsible for interfacing with the ISN and sending and receiving signals to other agents in the community.

The Comparator Plug-in receives signals from the local agent and from non-local agents. After allowing sufficient time for other agents to respond to an RFID signal, the Comparator Plug-in compares the local signal with non-local ones. If the local signal's intensity is the highest, the signal is relayed to the Authenticator Plug-in. Otherwise, the signals are discarded.

The Authenticator Plug-in functions as a gatekeeper. It sends a signal to an authentication server to verify the access rights of the corresponding carrier. If the carrier is allowed, the Authenticator Plug-in authorizes the restricted access door to be opened.

3.2 Community Design

ISN agents are grouped into communities based on physical location or function. For example, nodes in a hallway containing several doors could be arranged into a community. Other nodes providing various support services are located in their appropriate community. These communities are further contained by the greater society that is the overall system. Figure 2 shows a simple society layout involving a single community of ISNs and the supporting administrative community. Each ISN is on an individual machine, but the agents in the administrative community could share a machine or be distributed across many.

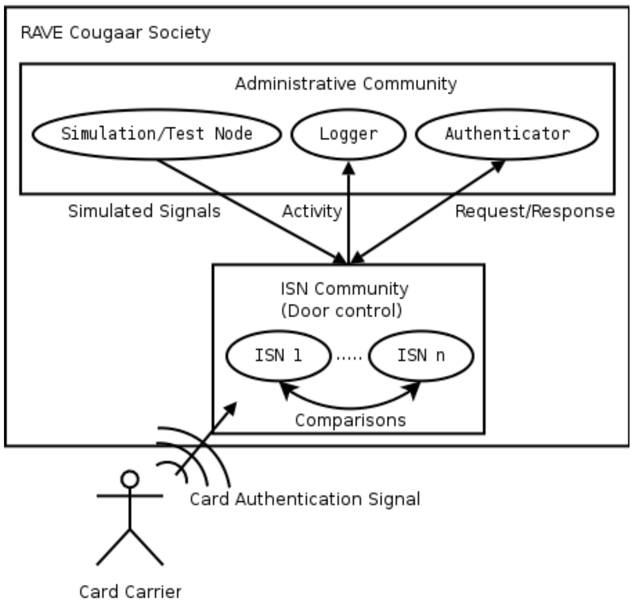

Figure 2. Society Diagram

3.3 Graphical Simulation

An interactive, graphical simulation was implemented to demonstrate the behaviour of the system visually. This simulation is composed of a GUI that allows a user to generate carrier signals and set the location of ISNs. The other component is a back-end which interfaces directly to the Cougaar system via a specialized node. Figure 3 shows the simulation GUI running. The six doors might depict a hallway and Zone 1 might depict an isolated ISN that could be a door or restricted area.

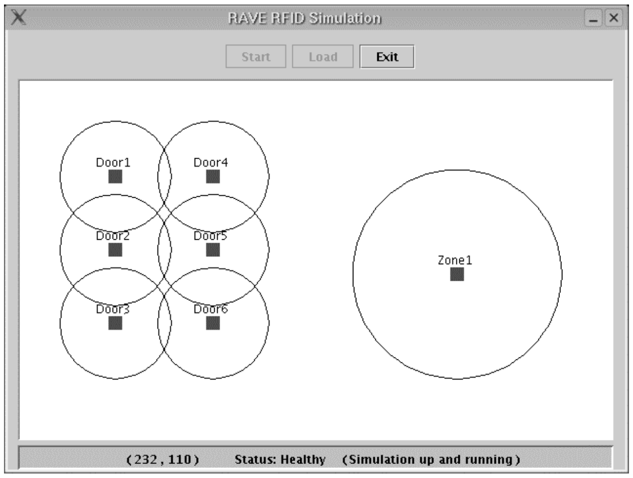

Figure 3. Simulation Screenshot

The back-end takes signals generated by the GUI and forwards them to the appropriate agents. These signals are generated by a mouse click and their intensity is determined by the modified inverse distance formula shown in Figure 4:

$$I(x,y)=\frac{1000}{\sqrt{(x_1-x_2)^2+(y_1-y_2)^2}}$$

Figure 4. Simulation Intensity Equation

(x1, y1) gives the coordinates of the ISN and (x2, y2) gives the coordinates of the RFID signal. The underlying behaviour of the agents is logged and the eventual results of the system, such as a door being unlocked, are displayed.

4. Test Plan and Results

The GUI based simulation is very useful for basic functionality testing and flaw discovery. However, a non-graphical, file-based testing method was developed to permit faster, more elaborate testing without the need for user interaction. This file-based test method was used for all trials to determine system correctness.

4.1 File-based Testing

The file tester more closely mimics the actual behaviour of the system as it supports any configuration of ISNs and RFID card activations. Test events consist of four pieces of information generated by the RFID card: the identity of the carrier, the x position of the card activation, the y position of the card activation, and the time in milliseconds at which the event occurred. For example,

Mike-Emery 100 130 1000

A complete, sample file is shown in Figure 5. The file begins with information regarding each ISN. An ISN consists of four pieces of information: the name of the ISN, the x position of the ISN, the y position of the ISN, and the minimum threshold of intensity that is required for the ISN to recognize a signal. Following the ISN information is the RFID signal information, ordered by the time that the signal is generated.

```
ISN-Name     X-pos   Y-pos   Threshold
ISN-#1        100     100       15
ISN-#2        100     200       15

Carrier-Name   X-pos   Y-pos   Time (ms)
Mike-Emery      100     130      1000
John-Paxton     100     160      1000
Rick-Donovan    125     140      2000
```

Figure 5. Sample Test File

The results are sent to a separate file for analysis. Of primary interest are the eventual results of the simulated signals. In Figure 6, the responses to the sample file are shown. For example, in Figure 5 at time 1000, both Mike and John activate their cards, but each person is closer to a different node. In Figure 6, Mike is granted access to ISN-#1 at time 1299, and John is granted access to ISN-#2 at time 2211. The difference between the time of activation and the time of authorization is discussed further in section 4.2.

Carrier	*ISN*	*Intensity*	*File Time*	*Actual Time*
Mike-Emery	ISN-1	25.00	1000	1299
John-Paxton	ISN-2	33.30	1000	2211
Rick-Donovan	ISN-1	22.30	2000	3283

Figure 6. Sample Test Results File

A special testing agent is used for the file-based testing. Testing information is provided to this agent via a file, loaded when the system first starts. The testing agent is responsible for sending signals to each ISN at the appropriate time as well as collecting and logging the results.

We devised a test methodology consisting of eight different tests. The first two tests examined the performance of a single ISN and the next six tests examined the performance of two ISNs. The goal of these eight tests is to validate the correctness of the solution by providing the system with a set of boundary and expected use authentication signals.

Test #1 consists of one ISN and one RFID carrier. This is the most basic, proof-of-functionality test. Signals are generated inside the ISN's range, on the boundary of the ISN's range and outside of the ISN's range. Please see Figure 7 where squares denote ISNs, circles show the boundary for an RFID signal to be detected, and asterisks show locations where RFID cards are activated. This test demonstrates the stand-alone capability of one ISN. Test #2 consists of one ISN and two carriers. Signals are generated (1) when one carrier is inside the range and one carrier is not, (2) when both carriers are within range, but at different distances, (3) when both carriers are within range and at the same distance, and (4) when both carriers are at the sensor. To avoid repeating test cases from Test #1, card activations by the two carriers always occur simultaneously.

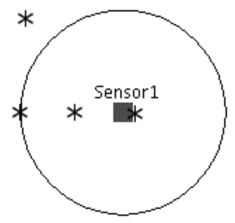

Figure 7. Single ISN

The next six tests all investigate the performance of two ISNs. These tests focus on the ability of the ISNs to interact with each other correctly. Note: in tests where two RFID carriers are present, they generate their RFID signals simultaneously.

Test #3 (one RFID carrier) and Test #4 (two RFID carriers) consider the case where two ISNs do not have intersecting detection areas. This arrangement of ISNs should be a common one in a regional airport and is depicted in Figure 8. The carriers are situated in the same manner as in Test #1 and Test #2.

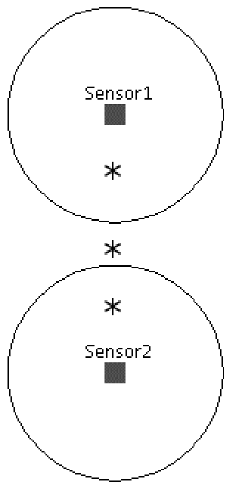

Figure 8. Separate ISNs

Test #5 (one RFID carrier) and Test #6 (two RFID carriers) consider the case where the two ISNs have intersecting detection areas as depicted in Figure 9. A single RFID carrier who is in range of each ISN should only be admitted to the closer door. This ISN arrangement is also quite common, especially in hallways containing many doors in close proximity to one another. In Test #5, the carrier is placed (1) within the range of both ISNs but at different distances to each, (2) within the range of both ISNs and at the same distance to each, (3) within range of one ISN, but on the edge of the other ISN and (4) on the edge of each ISN. Test #6 adds a case where one carrier is in the overlapping area while the other carrier is either outside both ranges or only inside one.

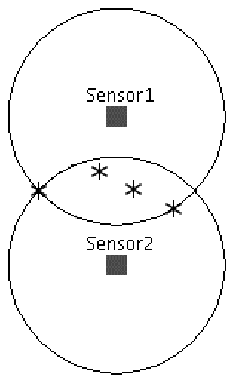

Figure 9. Intersecting ISNs

Test #7 (one RFID carrier) and Test #8 (two RFID carriers) are both purely theoretical. In these tests, the two ISNs occupy the same location, as depicted in Figure 10. Any and all access attempts should be rejected by the sensors, as neither ISN is closer to the RFID carrier.

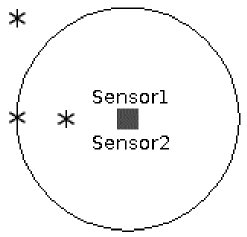

Figure 10. Identically Situated ISNs

4.2 Results

In order to evaluate the performance of the system, an oracle program was created. The oracle takes the test file and the results of the simulation and determines whether the simulation performed correctly or not.

Each test was performed 25 times. Tests 1, 2, 3, and 4 all performed correctly 100% of the time. Tests 5, 6, 7, and 8 performed correctly 88% - 96% of the time. When the results were incorrect, processor load was higher than normal. Errors occurred during sustained periods of 100% processor usage, such as when multiple instances of the entire simulation were running simultaneously. This resulted in ISN-to-ISN messages being delayed, causing an ISN to make a decision before all relevant messages were received. In these cases, more than one ISN granted access to a single carrier request.

To consider how an error might occur, consider the situation depicted in Figure 9 where a carrier is standing in the intersecting area between the sensors, but is closer to Sensor 1 then Sensor 2. This carrier attempts to authenticate during a period of 100% processor load. Both sensors relay their received signals to each other, but the relay from Sensor 1 to Sensor 2 is delayed. Sensor 1 receives the relayed signal from Sensor 2, determines that it is closer to the carrier, and authorizes access. Sensor 2, however, has not yet received the relayed signal from Sensor 1 before the timer expires. Consequently, it also decides that it is closest to the carrier and grants access. Later, the relayed signal from Sensor 1 is finally processed by Sensor 2, but there is no associated local signal, so it is discarded. The problem is caused primarily by the asynchronous nature of Cougaar's message transport service and thread handling [8] which does not easily lend itself to time-sensitive operations. Cougaar makes no guarantees as to the order of execution or to the timely delivery of messages. When the solution is placed on multiple machines (see Section 5), it is expected that these time delays will significantly decrease, as the processor load will be much lower per machine.

Several solutions to this problem exist, but at the cost of additional time. For instance, an ISN could require each other ISN to respond to every signal that it broadcasts. This causes an increase of traffic within the system by *n*m* messages, where *n* is the number of ISNs broadcasting a signal and *m* is the number of ISNs in the community. Another solution is to increase the time allowed before an ISN makes a decision. The maximum time is limited by how long users are willing to wait for door access.

5. Future Directions

As additional hardware becomes available for the project, the software focus will shift to running the application on multiple machines. This will come first in the form of simulation, followed later by integration of functional hardware allowing for a proof-of-concept demonstration.

With the addition of multiple machines, a central logging service will also be added. This will allow events relevant to each ISN to be gathered and stored in a central management node [5]. A tool such as Lumbermill [9] can then provide a UI front-end for analyzing the log files and for archiving the results.

Another future direction is to investigate how the architecture can be modified to allow one ISN to control two doors. This scenario is cheaper from a hardware standpoint, but will complicate the software. In this scenario, we are allowed to make three assumptions. First, each ISN is located at a door. Second, the other door that is controlled by the ISN is the next closest door to the ISN. Third, when a person activates an RFID card, that person is standing in front of a door. With these three assumptions, the problem is conceptually solvable.

Thus, there are many avenues that remain to be explored. We are excited to continue our investigations.

6. References

[1] RAVE Overview.
http://www.umt.edu/urelations/rview/1000/virtual.htm

[2] UltraLog Web site, http://ultralog.net.

[3] Cougaar Web site. http://www.cougaar.org

[4] Network Agents Project.
http://sourceforge.net/projects/networkagent/

[5] Cougaar Developer's Guide, Version 11.4. BBN Technologies. Available on [3]. 2004.

[6] Cougaar Architecture Document, Version 11.4. BBN Technologies. Available on [3]. 2004.

[7] Helsinger, A., Thome, M. & Wright, T. Cougaar: *A Scalable, Distributed Multi-Agent Architecture.* Proceedings of IEEE SMC on Agent Architectures, The Hague, 2004.

[8] Snyder, R. & MacKenzie, D. *Cougaar Agent Communities*. Proceedings of Open Cougaar 2004, New York, 2004.

[9] Lumbermill Web site.
http://sourceforge.net/projects/lumbermill.