THE ANNALS OF "DUNAREA DE JOS" UNIVERSITY OF GALATI
FASCICLE III, 2009, Vol.32, No.1, ISSN 1221-454X
ELECTROTECHNICS, ELECTRONICS, AUTOMATIC CONTROL, INFORMATICS

CHANNEL ALLOCATION MANAGEMENT USING AGENTS TECHNOLOGY

Gabriel Sirbu[1], Ion Bogdan[2]

1. *Dept. of Electronics and Telecommunications, Dunarea de Jos University of Galati, Domneasca-47, Galati-800008, Romania, e-mail gabriel.sirbu@ugal.ro*
2. *Dept. of Telecommunications, Gheorghe Asachi Technical University of Iasi, Dimitrie Mangeron-63, Iasi-700050, Romania, e-mail bogdani@etc.tuiasi.ro*

Abstract: Channel allocation schemes are management techniques meant to solve the radio access problem in the telecommunications area. There are several known schemes namely: static, dynamic, hybrid and flexible channel allocation schemes. In our work we developed the channel allocation scheme using a decentralized structure, based on software agents. In order to create such a structure, some functions must be carried out by some specialized functional entities. Each functional entity is implemented with an agent, having a specific role. Using this manner of implementation, different functions of the system can be implemented at different sites, located apart and also the structure can be modularized. In our work we have created the agents structure and the communications between agents, on the use of Multiagent Systems Engineering Technology. We also involved the AgentTool platform to automate the creation of the agents structure and the code resulted was in Java. In order to realize the test of our system we considered a shape of 21 radio areas, each including a number of users. Each wireless user can initiate a number of calls per hour and each call last a holding time. The communications between agents were realized for the TCP/IP protocol, based on socket connections.

Keywords: Agents, AgentTool Platform, Dynamic Channel Allocation Scheme, Mase Technology, Radio Resources Management, Wireless Communications..

1. INTRODUCTION

1.1. Channel Allocation Schemes.

Channel allocation schemes are strategies meant to solve the conflicts between multiple carriers in radio communications systems. Splitting the service area into cells and giving each cell the permission to use a set of specific radio channels is a solution to this problem (Proakis, 1995; Rappaport, 2002).

In the dynamic allocation scheme there is a total number of channels which is allocated each cell in the service area on a Carrier to Noise + Interference ration (CNIR) measurement basis as follows, Harada (2002):

$$(1)\quad R_{cni} = \frac{A \cdot P_0 \cdot d_0^{-\alpha} \cdot 10^{\frac{\xi_0}{10}}}{N + \sum_{i=1}^{m} A \cdot P_i \cdot d_i^{-\alpha} \cdot 10^{\frac{\xi_i}{10}}}$$

This paper was recommended for publication by Dorel Aiordachioaie

where P_0 and P_i are the transmitting powers of users u_0 and u_i respectively, $\propto$ is the path loss exponent, ξ_0 and ξ_I are the standard deviation of the log-normal fading (shadowing) associated to users u_0 and u_i respectively, N is the thermal noise power and A is a network specific propagation coefficient. We illustrate the situation in Fig. 1.

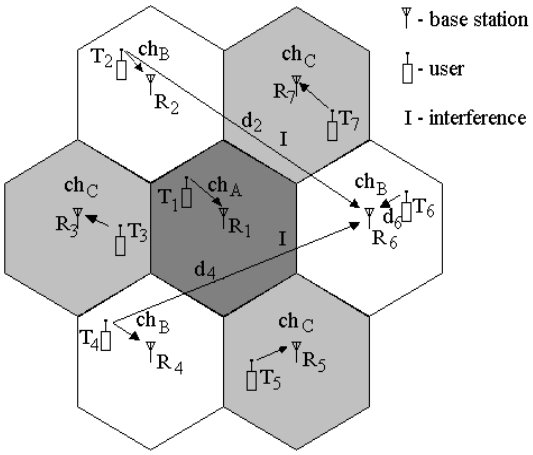

Fig.1. Interferences between radio cells

When the radio channel A is allocated to cell R_1, we need to calculate the interference that occurs. If the interference is larger than a threshold value, the current allocation of the channel is rejected. In order to take in account all the cells from the total radio covered area we used a wrapping technique, as in Fig. 2.

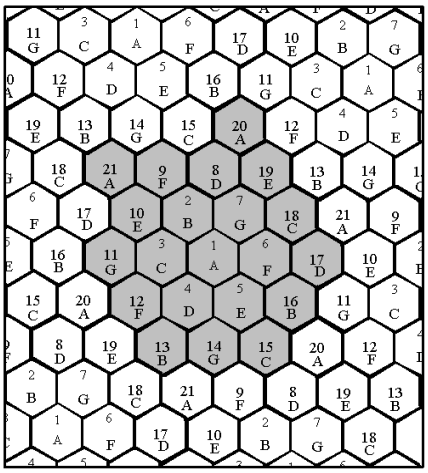

Fig.2. The wrapping method

Thus, if *n* is the total number of cells from the cluster in Figure 2 and *m* is the total number of considered cells, then W[m,n] is the wrapping matrix, containing all neighbors of the cluster's cell. When a call is initiated, the user occupies one of the randomly generated mesh position into the cell, thus, both the distance between users and base station and the interference can be precisely evaluated, Harada (2002). After that a channel is allocated to the call if the interference ration is lower than the threshold value. When the holding time is finished, the channel is released and the communication is stopped.

1.2. MaSE (Multiagent Systems Engineering)

The agent concept is being used as a part of the service architecture of next generation telecommunications networks. There are two types of agents: static and dynamic. In our system we developed static agents. The structure we used is created by the AgentTool platform, which generates the AgentMom framework. It is implemented in Java and provides the basic building blocks for building agents, conversations between agents and the messages that are passed through these conversations. AgentMom is capable of using five types of conversations:

a) unicast conversations using TCP/IP
b) secured unicast conversation using Secure Socket Layers over TCP/IP
c) multicast conversation using multicast socket and datagram packet.
d) secured multicast conversation using multicast socket and datagram packet with symmetric key algorithm.
e) broadcast conversation using datagram socket and datagram packet.

An overview of how conversations work is shown in Fig. 3.

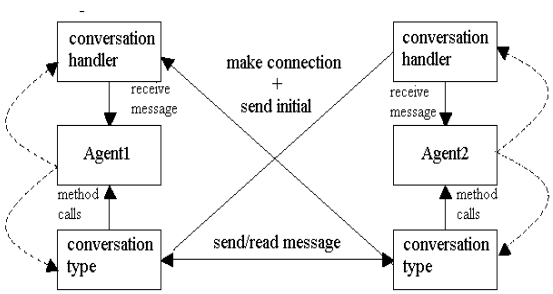

Fig.3. Conversations in AgentMom framework

In order to communicate with other agents, an agent starts one of its conversations as a Java thread. The communication establishes a socket with other conversation handler and sends the initial message. When the handler receives a message, it passes the message to the receive message method that validate the conversation. If the conversation is valid the agent starts it's appropriate conversation as a separate Java thread. After this, the conversation thread from each agent controls all communication.

There are two basic configurations for agent conversation methods:

- Agent based conversations
- Component based conversations

We implemented in our system the component based conversations, and the structure is referred in Fig. 4.

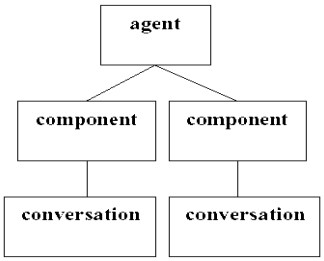

Fig.4. Component oriented Agent

2. SYSTEM IMPLEMENTATION

In order to create the skeleton of the final system, we followed the specific steps in MaSE:

- capturing the Goal Hierarchy
- defining the Sequences Diagram
- creating the role Diagram
- generating the Agent Template Diagram

The Goal Hierarchy stage is described by the Fig. 6 and consists in capturing the general structure of the Agent system. At this stage we defined the following modules:

- dynamic channel allocation, which is the higher class level
- call manager and channel manager, which implement the mid-level class
- call generator, call administrator, channel interference check administrator, blocking probability administrator and GUI (Graphical User Interface) administrator, which implement the methods used in lower classes.

The call generator block contains basically a set of methods necessary to supply the Poisson distributed randomly generated calls, with its mean arrival rate for each user of lambda = 24 calls/hour.

The cells are inspected at a "timestep" period, in order to pick up a new call, and to finalize the call with duration greater than holding time. In our system, holding time *ht* = 120 seconds. If we visualize Fig. 7, we can extract the precise order of all operations, described in Sequences Diagram.

In the horizontal dimension there are five sequences that our system iterate among. In the vertical dimension we can extract the temporal order the sequences flow. CallManager sequence takes the message "call" and sends forward another two messages, called "initiated_call" and "allocated_channel". We can better understand the actions if we analyze the diagram in Fig. 8.

Considering the Role Diagram, each role interacts with the corresponding one by mean of tasks. Each task contains a set of routines that basically execute all the actions described in that agent. For more details, see DeLoach (2001). For instance, the "call_administration" task has the structure described in Fig. 9 and basically represents a finite automaton, including an initial and a final state, and more intermediate states. The transition between these fixed states can be made with or without any conditional selectors, and also with or without any associated actions.

As in Fig. 9, the "interference_verification" task tries to find out if there is an available channel to associate with the new call. If the answer is "yes", the protocol "intercepted_call" will carry the answer. The communications between these tasks are of unicast type, which means they flow in one way and only between two tasks. The "probability_computation" task obtains the ratio:

$$(2)\quad \text{prblock}=\frac{\text{blocknum}}{\text{callnum}}$$

which is delivered to "graphical_display" task. Blocknum represents the number of blocked calls if there is no available channel for the incoming call, and callnum is the total number of initiated calls. All the protocols are of the Socket Connection type, based on the TCP/IP protocol.

Finally, one or more roles can define an Agent. In our work all agents were defined as fixed, consequently we obtained a structure of 5 agents, each having a single role. In Fig. 10 we illustrated this final step of code generation. As an instance, the class structure of the "call_administration" task generated by the AgentTool platform is shown in Fig.11, and the structure for the "Conversation 18_1" is shown in Fig.12.

3. THE RESULTS

Simulating with a number of 21 radio cells tested the system. For each cell we considered a number of 25 wireless users, each user having a holding time *ht* = 120 seconds. We considered the dynamic channel allocation scheme and the results are similar with those obtained in centralized manner in Sirbu (2003, 2005) simultation, in the same conditions. In Fig. 5 we illustrate the evolution of blocking probability versus time.

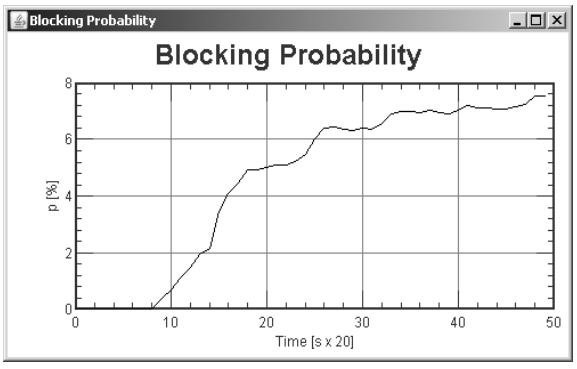

Fig.5. The blocking probability evolution, for 25 users per cell

There are also some differences regarding the centralized case, which are reflected in the gradient of variation.

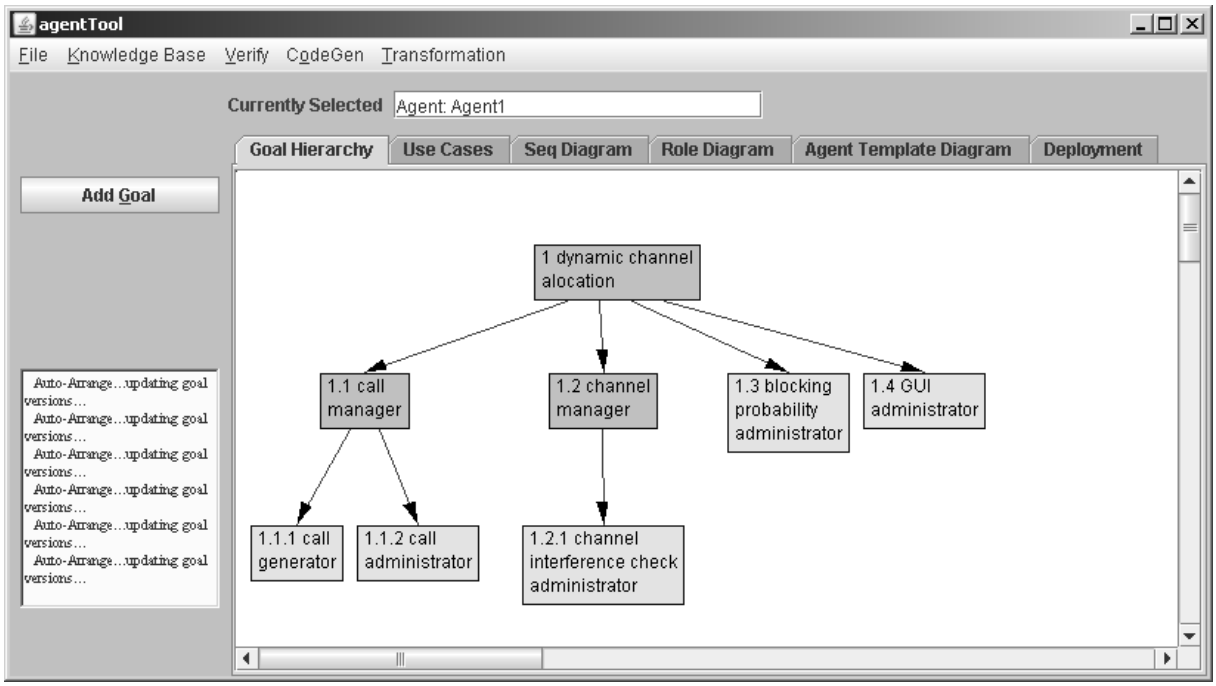

Fig.6. The Goal Hierarchy

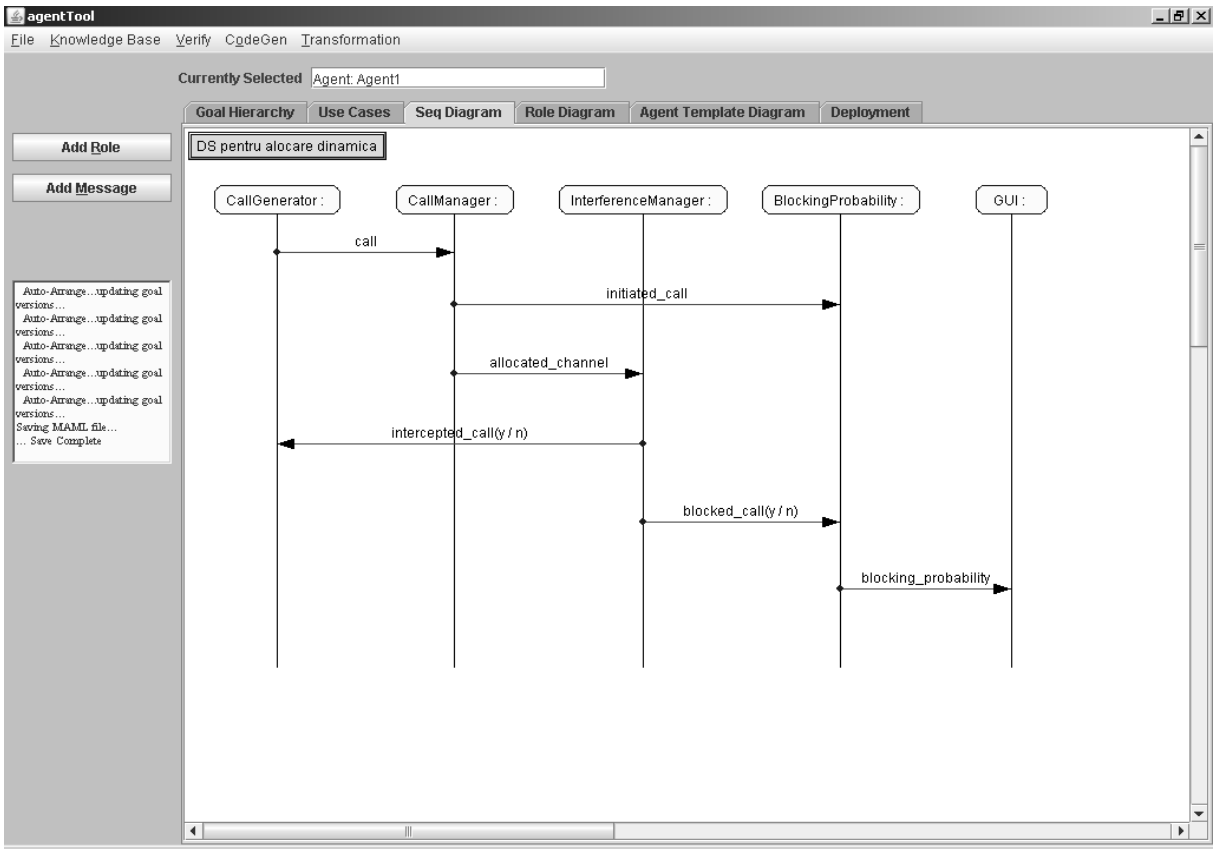

Fig.7. The Sequences Diagram

Our system has the capability to run on multiple machines. Although we didn't yet tested this implementation, the computational effort can be reduced, by sharing this effort among multiple systems.

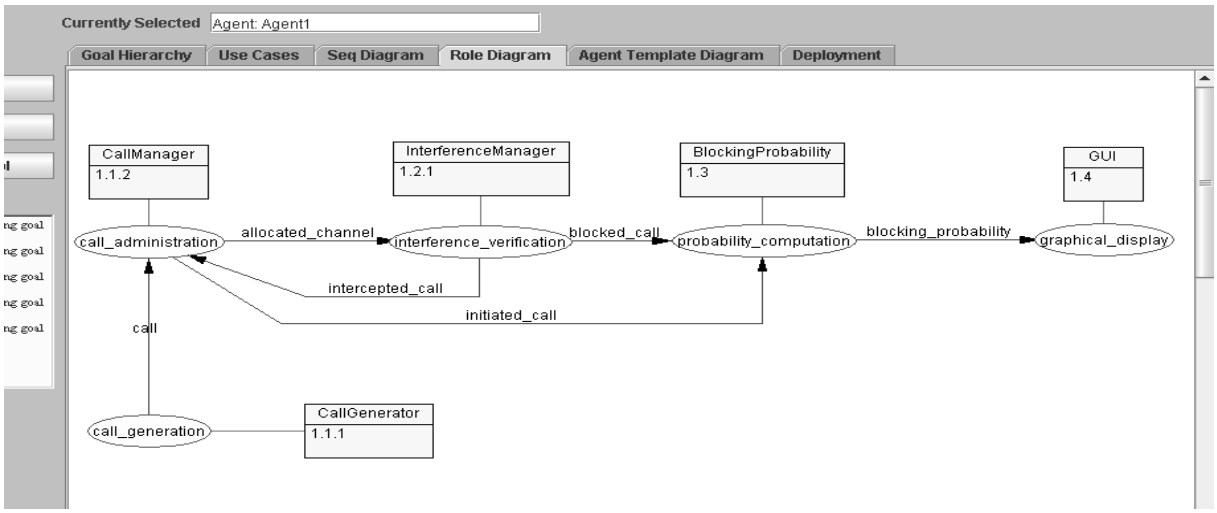

Fig.8. The Role Diagram

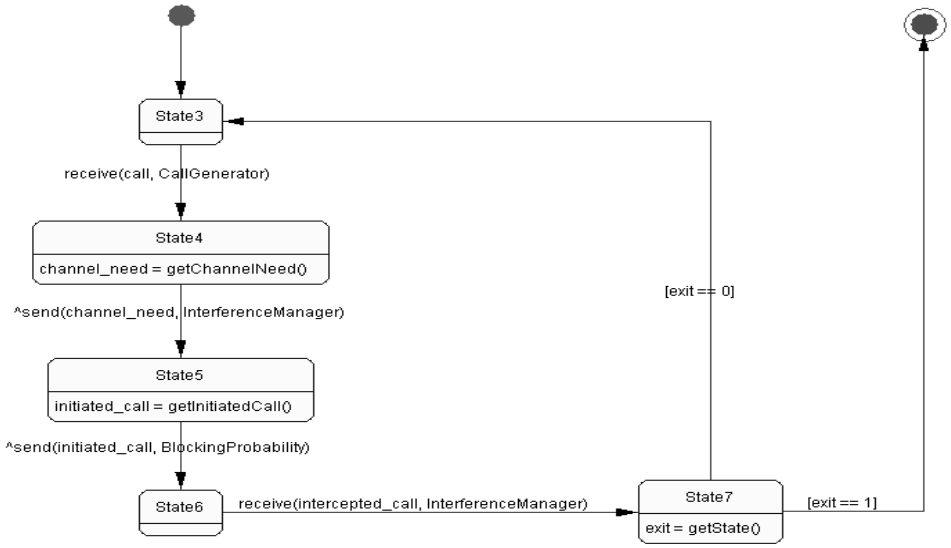

Fig.9. The "call_administration" task structure

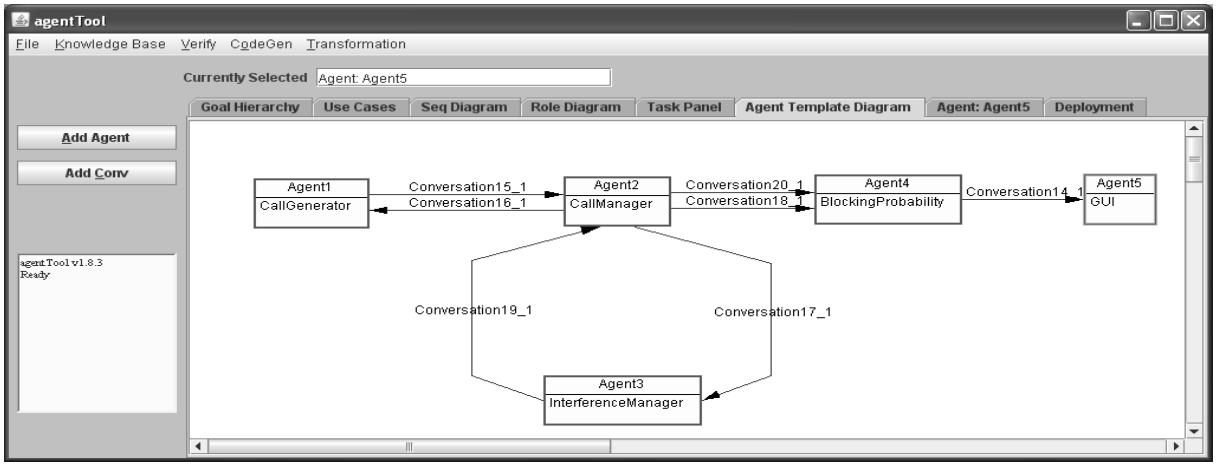

Fig.10. The agents structure

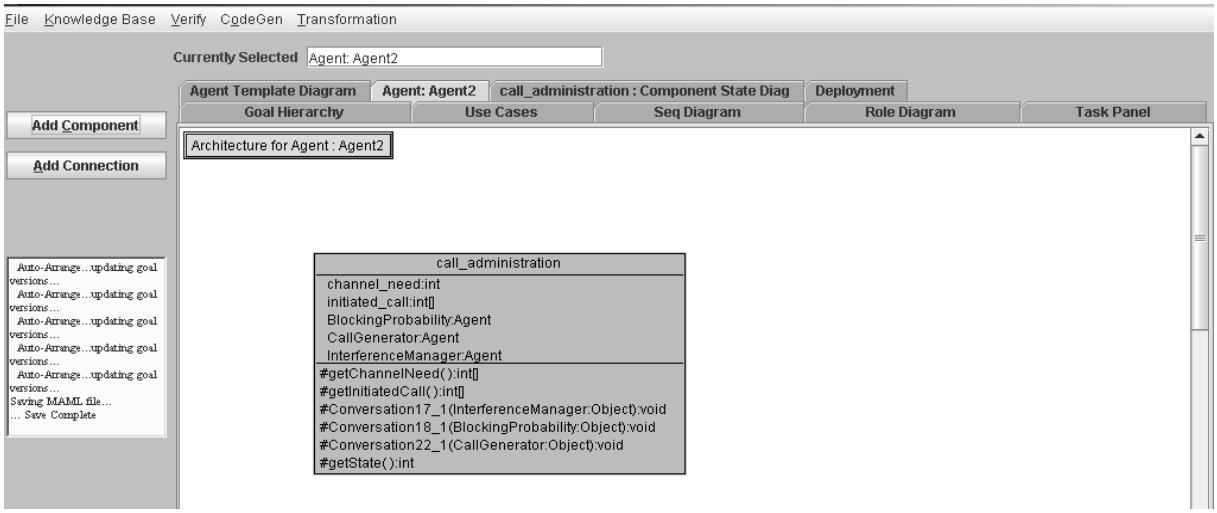

Fig.11. The structure for "call_administration" class

Fig.12. The generated conversation, initiator (left) and responder (right)

4. CONCLUSIONS

Our agent-based system is aligned to the MaSE technology, the structure is easy to implement, using a code-generator tool, namely AgentTool. The resulted code is in Java language, so it is platform independent, this system can be implemented on different machines, having different operation systems. In this work we used with good results the structure, which implemented the dynamic channel allocation scheme, using fixed agents.

Finally, the conversations between agents can be implemented in broadcast manner and also in secured manner.

We have to implement all these features and to draw further conclusions.

5. REFERENCES

DeLoach A. S., (2001), *Analysis and design using MaSE and AgentTool* (Proceedings of the 12th Midwest Artificial Intelligence and Cognitive Science Conference, MAICS 2001). http://www.macr.cis.ksu.edu/projects/agentTool

Harada H., Prasad R., (2002), *Simulations and Software Radio for Mobile Communications* (Artech House, London)

Proakis J. G., (1995), *Digital Communications*, (New York)

Rappaport S. T., (2002), *Wireless Communications Principles and Practice* (Prentice Hall)

Sirbu G., Bogdan I., (2003) *Analysis of Channel Allocation Schemes for Cellular Systems* ,Proceedings of SCS 2003, Iasi, Romania, Vol.II, pp533-536

Sirbu G., Bogdan I., (2005) *Performances Analysis of Different Channel Allocation Schemes for Personal Mobile Communication Networks*, Proceedings of the WSEAS Conferences, Control '05, Venice Italy, November 2-4, CD volume