INTERNATIONAL CONFERENCE ON
SOFTWARE, TELECOMMUNICATIONS AND
COMPUTER NETWORKS

SoftCOM 2002

Split - Dubrovnik, Croatia
Venice - Ancona, Italy
October 8-11, 2002

Edited by:

Nikola Rožić

Dinko Begušić

SoftCOM
library

Published 2002 by

Faculty of Electrical Engineering, Mechanical Engineering and Naval Architecture
University of Split
R. Boškovića b.b.
HR-21000 Split
Croatia

Editorial assistant: Hrvoje Dujmić

Cover design: Zoran Perdić Lukačević

Printed in Croatia by Slobodna Dalmacija, Split

CIP – Katalogizacija u publikaciji
Sveučilišna knjižnica u Splitu

UDK 621.39:004>(063)

INTERANTIONAL Conference on Software, Telecommunications and Computer Networks (10;2002;Split<etc.>
SoftCOM 2002 / International Conference on Software, Telecommunications and Computer Networks, Split-Dubrovnik, Croatia, Venice-Ancona, Italy, October 8-11, 2002; edited by Nikola Rožić, Dinko Begušić – Split; Faculty of Electrical Engineering, Mechanical Engineering and Naval Architecture, 2002. – XVI, 808 str.: graf. Prikazi; 24 cm

Prema predgovoru, knjiga je zbornik radova 10. međunarodne konferencije.
- Bibliografija uz svaki rad - Kazalo

ISBN 953-6114-52-6

1. Rožić, Nikola
2. Begušić, Dinko

ISBN 953-6114-52-6

INTERNATIONAL CONFERENCE ON
SOFTWARE, TELECOMMUNICATIONS AND
COMPUTER NETWORKS

SoftCOM 2002

FACULTY OF ELECTRICAL ENGINEERING, MECHANICAL ENGINEERING AND NAVAL ARCHITECTURE - FESB SPLIT

TECHNOLOGY CENTER SPLIT

CROATIAN TELECOM, TELECOMMUNICATIONS CENTER SPLIT

Sponsored by:

IEEE COMMUNICATIONS SOCIETY (COMSOC)
Technical Committee of Communications Software
Technical Committee of Communication Switching and Routing

MINISTRY OF SCIENCE AND TECHNOLOGY
REPUBLIC OF CROATIA

UNIVERSITY OF SPLIT

October 8-11, 2002
Split - Dubrovnik, Croatia
Venice - Ancona, Italy

Feasible Agent Communication Architecture

Maja Štula, Darko Stipaničev, Mirjana Bonković
University of Split, FESB
R. Boškovića bb, 21000 Split, Croatia
E-mail: {maja.stula, dstip, mbonkovic}@fesb.hr

Abstract: **Current efforts in standardization of agent architecture together with standardization of agent communication languages, interaction protocols and ontologies will facilitate agent systems develop and integration. Feasible agent communication architecture is described in this paper. Described architecture is based on existing web or ftp server applications, existing agent platforms and experimental experience gained in process of building multiagent system for monitoring and control of dislocated greenhouse.**

1. INTRODUCTION

In this paper overview of current standardization process in the agent technology and description of potential agent communication architecture is given.

Chapter Two contains summary on current standardization efforts in agent communication languages, ontologies and interaction protocols.

Agent communication techniques used in existing agent systems are described in Chapter Three. In this chapter is also described multiagent system developed for monitoring and control of dislocated greenhouse.

In Chapter Four is given potential agent communication architecture emerged from existing agent systems and experimental experience.

Future work, that includes testing of proposed agent communication architecture, and conclusion is given in Chapter Five.

2. CURRENT STANDARDIZATION EFFORTS

The Foundation for Intelligent Physical Agents (FIPA) is an international organization that is dedicated to promoting the industry of intelligent agents by openly developing specifications supporting interoperability among agents and agent-based applications.

The core mission of the FIPA standards consortium is to facilitate the interworking of agents and agent systems across multiple vendors' platforms. [2]

The World Wide Web Consortium has more than 500 Member organizations. W3C's long term goals for the Web are to make the Web accessible to all by promoting technologies that take into account the vast differences in culture, education, ability, material resources, and physical limitations of users on all continents, to develop a software environment that permits each user to make the best use of the resources available on the Web and to guide the Web's development with careful consideration for the novel legal, commercial, and social issues raised by this technology. [4]

As stated the standardization is aiming on making possible interoperability among different agent platforms and agent applications.

Probably the easiest part will be standardization of agent communication languages or ACL's. Agent communication language is a language with precisely defined syntax, semantics and pragmatics that is the basis of communication between independently designed and developed software agents as defined by FIPA. Widely used Knowledge Query and Manipulation Language (KQML) and FIPA ACL share a very similar syntax and semantics. [6]

There is also done progress in standardizing ontologies by the W3C Web Ontology working group. Majority of existing agent systems use implicitly defined ontologies for special problem which that system is covering. Such ad hoc ontologies are usually not reusable, they are hard to change or expand and they oppose obstacle in connecting those systems with another systems. W3C Web Ontology working group has produced working draft on requirements for a Web Ontology Language. This document specifies usage scenarios, goals and requirements for a web ontology language. Ontologies figure prominently in the emerging Semantic Web as a way of representing the semantics of documents and enabling the semantics to be used by web applications and intelligent agents. [2]

2.1 KQML AND FIPA ACL

A FIPA ACL message contains a set of one or more message elements. The only element that is mandatory in all ACL messages is the *performative*, although it is expected that most ACL messages will also contain *sender*, *receiver* and *content* elements. [5]

The Knowledge Query and Manipulation Language has been developed as part of the Defense Advanced Research Projects Agency (DARPA) Knowledge Sharing Effort (KSE)

project. KQML message consists of a *performative*, and also can contain *sender*, *receiver* and *content* information. [1]

One of the ACL message elements can be, and often is ontology. The ontology is used in conjunction with the language element to support the interpretation of the content expression by the receiving agent. [7]

FIPA ACL and KQML share common syntax but they are also conceptually very much alike. Both languages are based on the speech act theory which was developed by philosophers and linguists (J. L. Austin, John Searle) in an attempt to understand how humans use language in everyday situations.

Leaning ACL on speech act theory provides a limited but well defined set of interaction protocols or more precisely a limited set of message types shareable among agents.

2.2. INTERACTION PROTOCOLS

Exchanging messages between agents often fall into typical patterns. In such cases, certain message sequences are expected, and, at any point in the conversation, other messages are expected to follow. These typical patterns of message exchange are called *interaction protocols*. Interaction protocols define series of massages that will be exchanged between agents.

Agents in a system can be made so they are sufficiently aware of the meanings of the messages and the goals, beliefs and other mental attitudes the agent possesses, and that the agent's planning process causes such interaction protocols to arise spontaneously from the agents' choices.

Another solution is to specify the interaction protocols. All agents should have knowledge about specified interaction protocols, so that a simpler agent implementation can engage in meaningful conversation with other agents, simply by following the known interaction protocol. [8]

Interaction protocols can be divided to coordination protocols, cooperation protocols and negotiation protocols.

2.3. ONTOLOGIES

Similar to any other application, an agent application is connected with problems solving in a certain domain. To be able to share their knowledge, agents need to share common ontology, or they have to be able to translate between different ontologies because ontology provides the basic structure for building knowledge base. That basic structure contains set of concepts and terms for describing a certain domain.

3. MULTIAGENTS SYSTEM COMMUNICATION

Majority of existing agent software and existing agent applications use a kind of centralized management of communication. This statement relates to the multiagent systems in which agents are distributed spatially on connected computers [3]. The term centralized implies that part of communication is carried out by some kind of ANS (agent name server) or AMR (agent message router) or something similar. That part of communication can be only establishing the initial link between two agents or in a case of agent message router routing all massages between agents as shown on Figure 1.

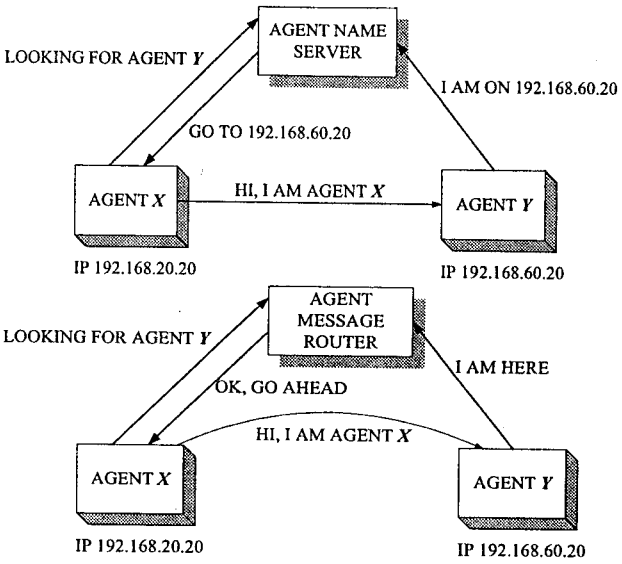

Figure 1 - Agents communicating through ANS and AMR

Although one of agent main properties is autonomy practical reasons have lead to centralized management of communication. It is unpractical and would consume lot of agent time and memory to take care of other agents IP addresses or to look for other agents without any centralized system. It can be concluded that former stated examples of agent communication will become standard. This means that agent system would be developed with communication architecture including standard communication agent (AMR or ANS) that would relieve agent from a quit a bit of communication problems. This would also impose possibility to standardize communication agents on application level protocol like FTP or Web server.

3.1. NETWORK PROTOCOLS USED BY AGENTS

Agents in their communications relay on standard network protocols like TCP/IP [Fig. 2.].

ANS and AMR are usually carried out as server/client applications hanging on a port and waiting for agent messages like agents itself. Agents are also carried out as applications listening on some port waiting to get an ACL message from another agent or ANS or AMR.

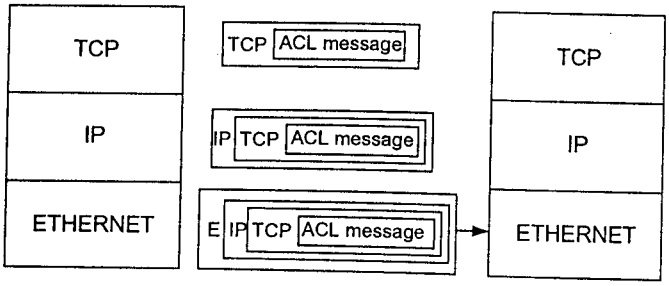

Figure 2 - Transferring ACL message over TCP/IP protocol

For establishing connection and exchanging messages they relay, as already stated, on a standard network protocol TCP/IP and protocols on application level like HTTP, FTP or SMTP protocols.

3.2. MULTIAGENT SYSTEM FOR MONITORING AND CONTROL OF DISLOCATED GREENHOUSE

In our case we have developed multiagent system for monitoring and control of dislocated greenhouse using Jatlite agent platform and KAPI software. User interface agents are used for communication with user. They were made like Java applets so that user could access the system throughout browser. [9] That way user could access system from wherever he had access to the Internet. [Fig. 3.]

User interface agents are using HTTP protocol to send and receive messages.

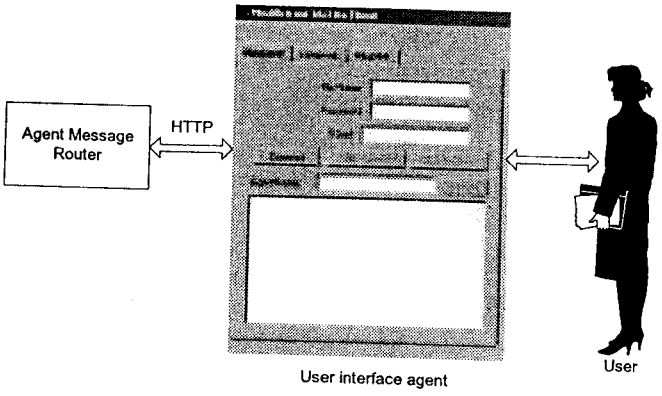

Figure 3 - User interface agent communicating with user and AMR

On the side of greenhouse we have couple of agents connected to sensors and actuators. They are working on lower level then application network level and using TCP/IP protocol for sending and receiving messages. AMR used was application siting on port 4444 and 4445 and accepting incoming messages from agents or sending it own messages to the agents. Each agent has established just one socket connection with AMR. [Fig. 4.]

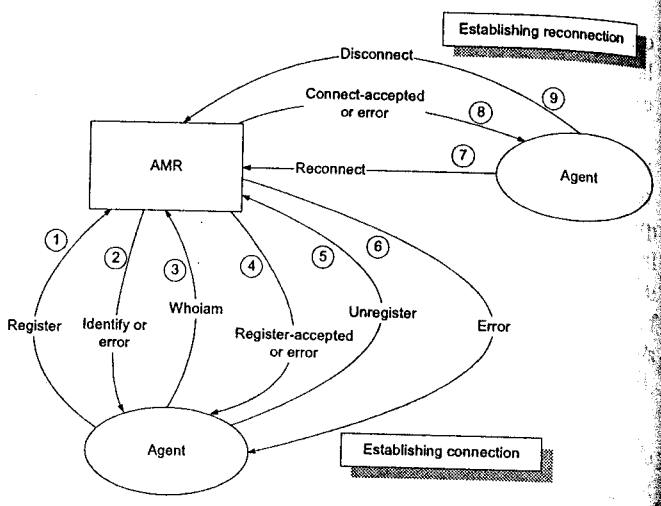

Figure 4 - Process of establishing connection and reconnection between agent and AMR

All communication in the system was carried out through AMR. Agents have knowledge about AMR IP address and port number. That is the only IP address agent has to know except it's own address.

When agents are started they connect to AMR. They exchange messages after that either by broadcasting messages or by sending messages to a particular agent. Broadcasted messages are directed by AMR to all agents connected to the AMR. To send message to a particular agent, agent has to know name of that particular agent. Agent has to have some kind of knowledge about other agents if it wants to engage in point-to-point communication (from one agent to another agent). That knowledge in system with AMR is reduced to knowing just agent name.

This is easier, particularly if agents can migrate (mobile agents) from one IP address to another IP address. That is specially the case with user interface agents.

Another advantage is that AMR can store messages. If agent is not connected to the system the message to that agent will be stored and delivered to agent when it reconnects.

The idea of standardizing AMR will relieve agent of knowledge about AMR IP address and port number. Yes, of course, agents will have that knowledge but it will be standardize and not susceptible to changes.

4. AGENT COMMUNICATION ARCHITECTURE

Application-level protocols are built on top of TCP/IP protocol which consists of lower-level protocols that provide the application-level protocol with a mechanism for reliable data transmission between computers. Every protocol has a

ort number which is used to decide what protocol is used for onnection.

The table 1. shows some standard application protocols nd standard ports used with those protocols:

Protocol	Port
FTP	21
TELNET	23
SMTP	25
FINGER	79
HTTP	81

Table 1 - Standard application protocols and standard ports

4.1. AMR SERVERS

If the ANS or AMR are to be standardize like web or ftp or mail server applications to have standard set of commands nd listen to standard ports it would become considerably easier to combine different agent application and platforms nd to make them interoperable. Extended table would look something like this:

Protocol	Port
FTP	21
TELNET	23
SMTP	25
FINGER	79
HTTP	81
AMR	4444

Table 2 - If AMR would be added as standard service on port 4444

We could develop agent and let it connect to the nearest ANS which would provide our agent information about other agents for interaction. So when we develop an agent system we would not have to take care about agent communication system. If we are using standard agent communication language our agents would be able to connect to standard ANS and through them communicate with other agents.

It would be convenient to provide some other information to AMR like ontology. AMR would make decision whether to accept or to deny connection to an agent. That decision can be based on number of currently active connections to prevent AMR overload. It could also be based on agent ontology. For example, AMR can maintain knowledge about other agent ontologies and refuse connection to agents dealing with totally opposite ontologies.

Standard ACL messages for establishing communication could go like this, based on ACL messages used in our agent system for monitoring and control of dislocated greenhouse:

1. (register-agent :receiver AMR_NAME :sender AGENT_NAME :ontology computer)
2. (identify-self :sender AMR_NAME :receiver AGENT_NAME)
3. (whoiam :sender AGENT_NAME :receiver AMR_NAME :content (host 192.168.22 :port 6721))
4. (connection-accepted :sender AMR_NAME :receiver AGENT_NAME)

or

5. (connection-refused :sender AMR_NAME :receiver AGENT_NAME :content (ontology not supported))

or

6. (connection-refused :sender AMR_NAME :receiver AGENT_NAME :content (AMR_NAME overload))

AMR can have integrated another functionality. It could contain knowledge about other AMR's domains, that is, ontologies they are covering. That knowledge would be at disposal to each agent connected to a AMR. AMR could reconnect agent to another AMR dealing with agent domain or it could just route agent messages. [Fig. 5.]

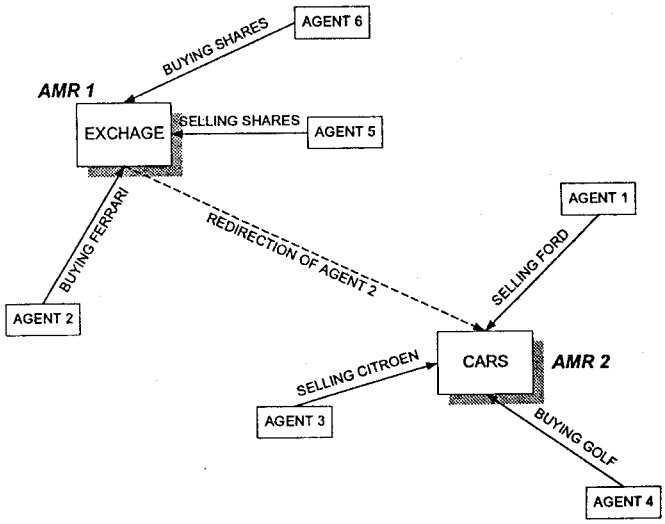

Figure 5 - Connecting to the nearest AMR and reconnecting to wanted AMR

This global knowledge would be dynamically updated. New AMR connecting to the net would exchange its knowledge with existing AMR's. New agent would connect to the nearest AMR he is aware of and gain access to AMR knowledge about other agents connected to that AMR and about other AMR's and they domain of interest.

5. CONCLUSION

Standardization of agent communication language, ontologies and interaction protocols will in the end make

possible to develop standard agent systems capable to interact even using different programming languages and agent developing tools.

Proposed agent communication architecture together with adding new application protocols for agent communication could make easier current standardization efforts. Existing agent applications as well as developing tolls, as shown, already have support for proposed architecture.

So it will be possible and advisable to use such architecture to facilitate developing agent systems open for integration and cooperation with other agent systems.

Future work will provide multiagent system with described agent communication architecture. Through this system will be tested described agent communication architecture feasibility and advantages.

REFERENCES

[1] Michael Wooldridge, Nick Jennings: Intelligent Agents: Theory and Practice, *Knowledge Engineering Review*, Volume 10, No. 2, *Cambridge University Press*, 1995.

[2] http://www.w3.org/TR/2002/WD-webont-req-20020307

[3] Jasques Ferber: Multi-agent Systems, An Introduction to Distributed Artificial Intelligence, *Addison-Wesley*, England, 1999.

[4] http://www.fipa.org

[5] http://www.w3.org/Consortium

[6] http://www.fipa.org/specs/fipa00061

[7] Venu Vasudevan: Comparing Agent Communication Languages, *OBJS Technical Note*, July 1998.

[8] http://www.fipa.org/specs/fipa00025/XC00025E.htm

[9] Heecheol Jeon, Charles Petrie, Mark R. Cutkosky: JATLite: A Java Agent Infrastructure with Message Routing, *IEEE Internet Computing*, March-April 2000