

Modeling Mobile Agent Network with Parallel Communicating Agents

Gordan Jezic, Mario Kusek, Ignac Lovrek and Vjekoslav Sinkovic
Department of Telecommunications
Faculty of Electrical Engineering and Computing
University of Zagreb
FER Unska 3, HR-10000 Zagreb, Croatia
{gordan.jezic, mario.kusek, ignac.lovrek, vjekoslav.sinkovic}@fer.hr

Abstract

The paper deals with modeling of mobile agent network. The model includes a multi-agent system consisting of parallel communicating agents, a set of processing nodes in which the agents perform services and a network that connects processing nodes and allows agent mobility. Parallelism in mobile agent network is based on parallel agents allocated to a single task requested from the environment. During migration and execution, the model allows agent communication and result sharing between the simultaneously operating agents and improves the overall system performance. Mobile agent network is defined as queuing system where the agents represent information units to be served.

1. Introduction

Agent paradigm is a promising choice for network-centric applications because it is intrinsically communication and co-operation oriented [1, 2]. Multi-agent systems and especially the systems containing mobile and intelligent agents will provide full mobility to the users and services.

This paper elaborates a modeling of mobile agent network with parallel communicating agents and related performance evaluation framework. The mobile agent network consists of a multi-agent system and its agents co-operate, communicate and travel in the network. The users interact with the mobile agent network by requesting the services. Each user request activates one or more parallel agents with a set of elementary services that correspond to users' request. The agents migrate autonomously from node to node, performing some processing on behalf of a user. Communication of the agents during migration

and execution allows co-operation and result sharing between the simultaneously operating agents and improves the overall system performance.

The paper is organized as follows: model of mobile agent network is described in Section 2. Parallel communicating agents are elaborated in Section 3, performance evaluation is presented in Section 4, and Conclusion is given in Section 5.

2. The Model of Mobile Agent Network

The model of mobile agent network is represented by the following triple:
 $\{A, S, N\}$,

where A is a multi-agent system consisting of co-operating and communicating mobile agents, S is a set of the processing nodes at which the agents perform services, and N is a network that connects the processing nodes and allows agent mobility [3, 4].

Each agent performs a set of tasks called elementary services, defined by the user request. Service types of the tasks determine migration of the agent. Each task can be executed only at appropriate nodes, depending on the task service type. In case of the complex user request with many and/or dependent elementary tasks, it is useful to create and employ a team of communicating agents in order to execute the tasks in parallel [5]. The service provided by the $agent_k$ comprises ns_k elementary services, es_{kl} , as follows:

$$ES_k = \{es_{k1}, es_{k2}, \dots, es_{kl}, \dots, es_{k(ns_k)}\}$$

Each node, S_i , is characterized by the set of service types, s_i , it supports, as follows:

$$s(S_i) = \{s_1, s_2, \dots, s_l, \dots, s_m\}$$

At each node the agent can execute elementary services that satisfy $es_{kl} \in s(S_i)$, i.e. es_{kl} can be executed only at the nodes that serve its service type.

The mobile agent represents a user in the network. It can migrate autonomously from node to node, to perform some processing on behalf of the user. The agent's owner is defined by its identification and service request. Node data consist of actual node address wherefrom the agent is executing at the point of time. Task part is a list of tasks to be performed by the agent. Data part is the field in which the agent stores the results.

Parallelism in the mobile agent network is based on the parallel agents allocated to a single request from the environment. The mobile agent network is defined as the queuing system where parallel communicating agents represent information units to be served. With the introduction of the agent communication it is possible to reduce the processing time.

User request is presented by a directed acyclic graph $G = (L, E)$ where L denotes the list of elementary tasks (services), and E represents the set of directed edges that define precedence relations between the tasks. Figure 1 shows the example of a task graph.

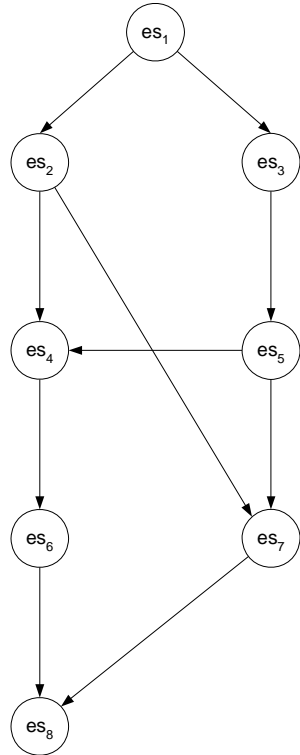


Figure 1 – Example of a task graph

Each task from the list L is defined by the triple $l_i = \{i, s_i, [p]\}$. The element i represents service es_i , s_i its service type, and $[p]$ the list of predecessors. For the example in Figure 1, the list is the following:

$$L = [\{1,1,[]\}, \{2,1,[1]\}, \{3,2,[1]\}, \{4,2,[2,5]\}, \{5,1,[3]\}, \{6,3,[4]\}, \{7,1,[2,5]\}, \{8,3,[6,7]\}]$$

Dependent tasks es_3 and es_5 (start of es_5 depends on the result of es_3), where $es_3 > es_5$ defines the precedence relation (es_3 must precede es_5). The type of services for es_1, es_2, es_5 and es_7 is s_1 , es_3 for es_4 is s_2 and for es_6 and es_8 is s_3 . For example, service es_8 depends on execution of the services es_6 and es_7 .

The agents forming multi-agent system are grouped into domains, according to service types of their elementary tasks. The agents from the same domain execute the tasks with the related (equal, dependent or similar) service types. Therefore, $agent_k$ belongs to the domain d_j from the set of domains $D = \{d_1, d_2, \dots, d_j, \dots, d_n\}$, $agent_k \in d_j$. The agents registered within the same domain are the candidates for mutual communication.

Agent communication between the two agents, $agent_a, agent_b \in d_j$, can be accomplished in the following ways:

- $agent_a$ and $agent_b$ contain es_{aj} and es_{bl} of the same service type, in which case the agent can take over the result from other agent who has completed the execution;
- $agent_a$ and $agent_b$ perform dependent tasks es_{aj} and es_{bl} (start of es_{bl} depends on the result of es_{aj}). When $agent_a$ completes es_{aj} and communication with $agent_b$, $agent_b$ can start es_{bl} ;
- $agent_a$ creates and starts a new $agent_b$ which executes delegated elementary tasks or merely transports the results to the selected node (transport agent).

Agent communication can be performed locally, on the same node, or remotely, between the agents placed at different nodes. In case of remote agent communication, the communication is based on the messages exchanged through the network.

A message sent by an agent communication consists of three parameters, $\{sender, receiver, content\}$, where $sender$ represents the name of an agent which sends the message/creates new agent, $receiver$ is the name of an agent which receives the message/newly created agent and elementary service to which the content should be sent. The $content$ is the result/set as an input parameter for execution to continue. Each agent has a unique name which corresponds to communication address $name@hostname:port$ where the agent is created. The $name$ is a unique expression for the agent within the domain, $hostname$ is the name (IP address) of the node S where an agent is placed, and $port$ is the port number of that node from which the messages are sent and to which they are received. In order to provide an open environment, FIPA proposals of agent communication and agent communication language, ACL, are followed [6].

The number of agents needed for tasks execution is determined by the complexity of the user request. Creation of agents can be performed before processing (static) or dynamically during processing. The model includes dynamic generation and creation of the agents based on *the earliest parallelism of agent creation*. The number of successor tasks of a particular task defines the number of newly created agents, and the number of predecessor tasks defines the number of destroyed agents, according to precedence relations.

The model includes the following facts:

- Having completed an elementary service es_{kn} , $agent_a$ continues execution at the same node (es_{kn+1}) or moves to another in order to perform next elementary service, es_{ln+1} . The agent moving is determined by the service type of the next elementary service.
- If task es_{kn} , executed by an agent, has z successors, then the agent continues execution of one of the successors (es_{kn+1}) and creates $z-1$ new agents that execute other successor tasks of task es_{kn} .
- Having completed all tasks, the agent, which executes the last one, sends the response to the user.

3. Parallel Communicating Agents

The proposed model of the mobile agent network with parallel communicating agents is based on the parallel agents allocated to a single request from the environment (user), or parallel agents handling simultaneous requests from different users. Related work includes different aspects of parallel processing with mobile agents [7-9]. The following example explains migration, parallel execution, creation and communication of the mobile agents.

If two agents, $agent_a$ and $agent_b$, that execute dependent tasks, es_{ij} and es_{kl} , are allocated to different nodes, S_i and S_k , agent communication c_{ab} between agents must be performed. Three types of agent communication are possible:

- Direct agent communication,
- Indirect agent communication, and
- Remote agent communication.

Figures 2 and 3 show the diagrams of possible agent communication types in AUML notation [10]. *Direct agent communication* is the local communication between the agents that execute dependent tasks. *Indirect agent communication* supposes creation of the new agent, which migrates to the node with the agent who executes successor task in order to communicate locally

(Figure 2). *Remote agent communication* allows communication between the agents placed at different nodes (Figure 3). In this case, communication is based on message handling through the network.

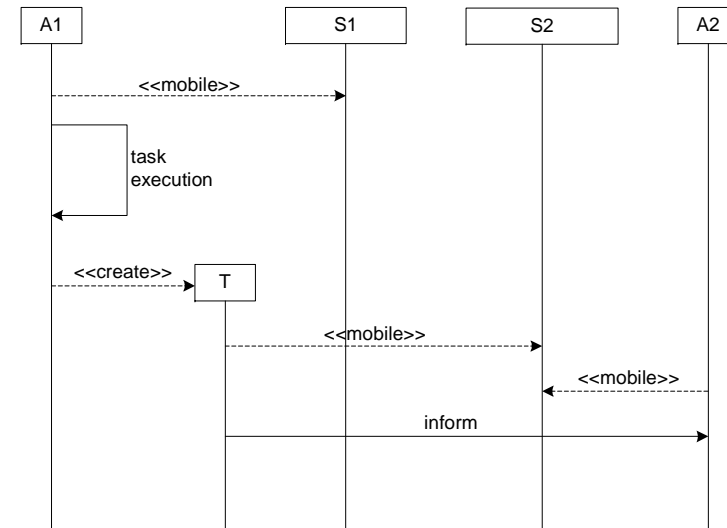


Figure 2 – Indirect agent communication

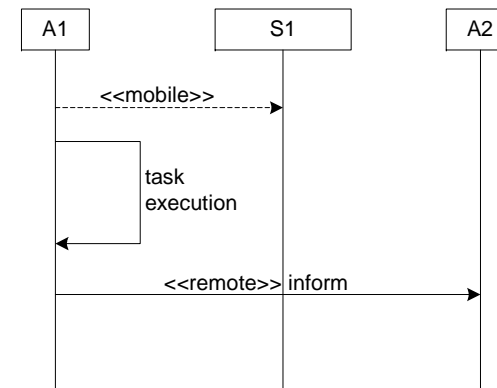


Figure 3 – Remote agent communication

The principle of agent execution is shown in Figure 4. When an agent is created and the elementary service list is set, the agent moves to the node where the first elementary service should be executed. After its execution, the agent tries to send the result remotely to the agent(s) that execute(s) successor tasks. If the agent executes the last elementary service, after successful receipt of the result (successful remote communication) it goes home and reports to the owner. If this is not the last elementary service, the agent moves to the node of the next listed elementary service.

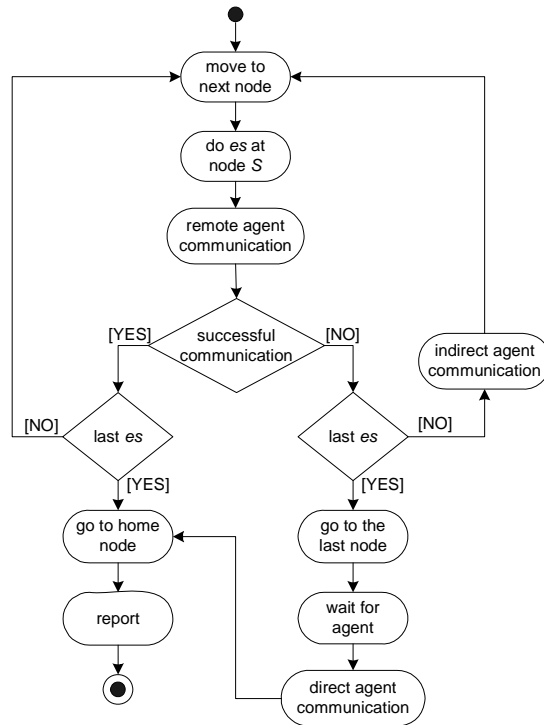


Figure 4 – Activity Diagram of Agent Execution

When the remote communication fails, the agent checks whether this task is the last elementary service to be executed. If so, the agent moves to the node where the result should be transferred and waits for the agent that must receive the data. When that agent arrives to the same node, direct (local)

communication is performed and then the agent goes home and reports to the owner.

In another case, when the remote communication fails and is not the last listed elementary service, the agent starts indirect agent communication. It creates a new agent (transport agent), which goes to the node and locally sends the result to other agent.

4. Performance Evaluation

In order to introduce performance evaluation capabilities, the mobile agent network is described as the queuing system where the agents represent information units to be served. The nodes are the servers capable of hosting, parallel execution and communication with the agents (Figure 5). When hosted by the same network node, the agents from the same domain can exchange the results by performing local agent communication and, thus, reduce the number of non-executed elementary services.

The node S_i with processing capacity B_i receives the input agent flow G_i and network agent flows X_{ji} . Those flows are summarised as L_i and sent to the queue Q_i organised according to the domain principle. When it is $agent_k$ from Q_i turn, the processor executes es_{ki} . After completing it, the $agent_k$ either executes $es_{k_{n+1}}$ on the same node (U_i flow) or migrates to another node and executes next elementary service there. The migrating agent is placed in the migration queue TQ_i . When it is the $agent_k$'s turn to migrate, it is serialised and transferred to node S_j . If the $agent_k$ has completed the last elementary service, it is placed in the output flow R_i .

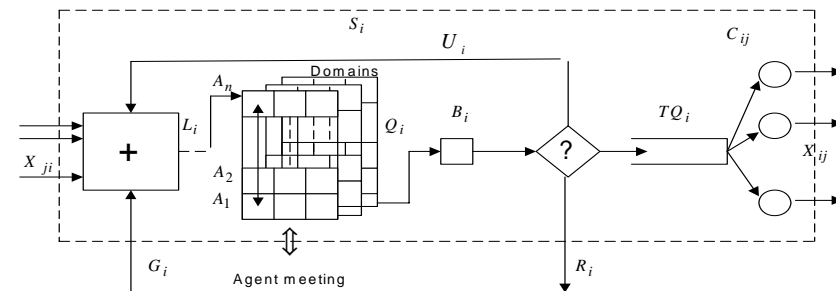


Figure 5 – Node Structure

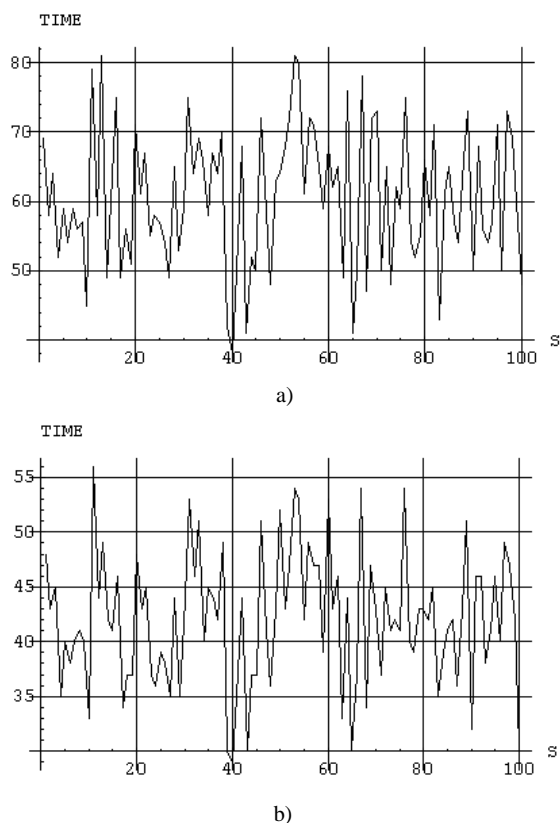


Figure 9 - Influence of agent communication

5. Conclusion

Formal model of the parallel communicating agents in the mobile agent network is elaborated. Parallel operations within multi-agent system are specified. Agent domain and agent precedence relation are used for recognizing related agents that are the candidates for communication. The related agents perform equal, dependent or similar tasks.

Future work will include improvements of formal model and performance evaluation of parallel agent execution with agent communication.

References

- [1] W. Cockarine, M. Zyda, *Mobile Agents*, Manning, Greenwich, 1998.
- [2] W. Brenner, R. Zarnekow and H. Wittig, *Intelligent Software Agents - Foundations and Applications*, Springer, Berlin, 1998.
- [3] V. Sinkovic and I. Lovrek, Generic Model of a Mobile Agent Network Suitable for Performance Evaluation, *Proceedings KES'2000 Fourth International Conference on Knowledge-Based Intelligent Engineering Systems & Allied Technologies*, Brighton, UK, 2000, pp. 675-678.
- [4] I. Lovrek and V. Sinković, Performance Evaluation of Mobile Agent Network, *Proceedings KES'2001 Fifth International Conference on Knowledge-Based Intelligent Engineering Systems & Allied Technologies*, Osaka, Japan, 2001, pp. 924-928.
- [5] V. Sinkovic, I. Lovrek, M. Kusek, Parallelism in Mobile Agent Network, *MPCS 2002 International Conference on Massively Parallel Computing Systems*, Accepted for publication, Ischia, Italy, 2002.
- [6] FIPA Agent Management Specification, Foundation for Intelligent Physical Agents, 2000, URL: <http://www.fipa.org/specs/fipa00023/>
- [7] Scott A. DeLoach, Specifying agent behaviour as concurrent tasks, *Proceedings of the fifth International Conference on Autonomous Agents*, Montreal, Canada, 2001, pp. 102-103.
- [8] L. M. Silva, V. Batista, P. Martins, and G. Soares, Using Mobile Agents for Parallel Processing, *Proceedings of the International Symposium on Distributed Objects and Applications*, Edinburgh, United Kingdom, 1999.
- [9] C. Panayiotou, G. Samaras, E. Pitoura, and P. Evripidou, Parallel Computing Using Java Mobile Agents, *Proceedings of the 25th Euromicro Conference (EUROMICRO '99)*, Milan, Italy, 1999
- [10] J. Odell, H. V. D. Paraunak, B. Bauer: *Extending UML for Agents*, In Proc. of the Agent-Oriented Information Systems (AOIS) Workshop at the 17th National conference on Artificial Intelligence (AAAI), 2000.

Biography

Gordan Jezic received his B. Sc. and M. Sc. degrees from the University of Zagreb in 1995 and 1999, respectively. He is with the Faculty of Electrical Engineering and Computing of the University of Zagreb where he works as an assistant at the Department of Telecommunications. His research interests include mobile software agents, mobile computation and formal methods for specification and verification of mobile processes.

Mario Kusek received his B. Sc. and M. Sc. degrees from the University of Zagreb in 1997 and 2001, respectively. He has been working at the Faculty of Electrical Engineering and Computing as

an associate assistant since September 1997. His research interests include mobile software agents, distributed computing and Internet technologies.

Ignac Lovrek received a B. Sc., M. Sc. and Ph. D. from the University of Zagreb in 1970, 1973 and 1980, respectively. He is with the Faculty of Electrical Engineering and Computing of the University of Zagreb where he works as a professor at the Department of Telecommunications. His research interests include call/service modelling and processing, telecommunication system architectures, and software engineering and new soft technologies for telecommunications.

Vjekoslav Sinkovic received the B.S. degree in electrical engineering from the University of Zagreb, Faculty of Electrical Engineering in 1962, and the M.S.E.E. and Ph.D.E.E. degrees from the University of Zagreb in 1966 and 1968, respectively. Since 1963 he has been employed at the Faculty of Electrical Engineering, Telecommunications Department at the University of Zagreb, assistant 1963, assistant professor 1969, associate professor 1974, professor 1979. His research interests focus on performance evaluation of high-speed networks and parallel and distributed systems.