Definition of formal infrastructure for perception of intelligent agents as »problem solvers«

Prof. dr. sc. Marijan Kunštić, dipl. inž. Marina Bagić
Faculty of electrical engineering and computing, Department of telecommunications
Zagreb, Croatia
E-mail: marijan.kunstic@fer.hr, marina.bagic@fer.hr

mr.sc. Oliver Jukić
ICE systems
Slatina, Croatia
E-mail: oliver.jukic@icesystems.hr

Abstract: Transport platform development and softwaresation of telecommunications have triggered process of intelligent agents penetration into the telecommunication area. Since management can be perceived as process of sequential problem solving, we have tried to make formal infrastructure for perception of intelligent agents as "problem solvers". Creation of set of agents required for particular problem solving, as well as their schedule within the network are of great importance within our formal infrastructure. At the end, basic concepts of programming verification by simulation tool are given.

1. INTRODUCTION

In this paper we will try to present possible formal infrastructure for presentation and perception of intelligent agents as »problem solvers«. It means we will try to perceive agent's behavior as problem driven, what is naturally to do.
There are two pre-conditions that we've mentioned as crucial for introduction of intelligent mobile agents model in telecommunication. Those are, but not limited to:

1. Transport platform development
   Current telecommunications transport platform provide us with a very high level of flexibility during communication process.
   If agent is perceived as (part of) software system, we can say agent is portable and transportation time between network nodes will not be necessary greater then time needed for problem solving.
2. »Softwaresation« of telecommunications

If we have decomposed process of network management to the smaller problems, we are able to introduce agents as "problem solvers", e.g. assistants during system management process, which are able to "travel" across the network and make management tasks for us.
Combined with previous pre-condition, these provides us with very flexible and powerful mechanism of distribute network management.
As we can consider above-mentioned preconditions fulfilled, it is possible to define or, better said, perceive agent as problem's solver within telecommunication domain.

2. FORMAL MODEL INTRODUCTION

Staring point of functional specification for every particular agent object (or group of agent objects working together on problem solving) we can find within "problem space" domain.
Spheres of this space penetrate within two important segments of telecommunications: telecommunications resources production and telecommunications' resources management.
Let us define "problem space":
<Definition 1>
Problem space is finite unordered set of problems, generated by problem environment of specific domain. Problem set is ordered subset of problem space.
Every problem within problem space can be paired with probability of appearing at particular network location at particular moment.
Since every problem needs solver to solve it (or group of solvers), we can try to define solvers space. Let us define it:
<Definition 2>
Solver space is finite, unordered set of problem solvers, generated by network and node infrastructure and available resources. Solver set is ordered subset of solver space.

Defining relations between members of solver space we must consider way of solving every problem, which represents inter-solver communication protocol, globally speaking.

Very important factor during definition of problem solver is location of problem as well as solver location. So, it is necessary to find optimal solution regarding to transportation time, efficient problem solving, minimization of total solving time and meaningless communication.

Very important feature of the system is its changeability. It can be changeability of system functionality, or changeability of relations between system' components (agents). Hence we can consider system as a state of solver space (subset), and we define:

<Definition 3>

Agent system (AS) is ordered relational state within collection or subset members of solver space.

So, agent system is set of solvers with exactly defined relations between them.

Problem space environment or problem environment (Πo) generates problem space elements (Πp). Over this set problem set can be considered also ($P\zeta$). Elements of problem set implicate solver set creation ($Rj\zeta$), which is ordered subset of solver space.

$$(\Pi o)\circ \rightarrow (\Pi p) \supset (P\zeta) \Rightarrow (Rj\zeta) \quad (1)$$

Problem environment examples are:

1. Telecommunication market
   It generates new resources requirements (equipment, programming modules, tools, …) and new service requirements. So, problem space is made of requirements and industrial programmes while solver space is made of intelligent developer environment. By choosing of concrete tools, techniques and languages from solver space we create subset that represents solver set.
2. Telecommunication resources management
   Problem environment is made of customers that have needs in terms of new services creating as well as existing service support (quality of service, technical support, new information, flexibility, etc.)

Let us define problem space of network management:

$$UM(\Pi o)\circ \rightarrow UM(\Pi p) \supset UM(P\zeta) \Rightarrow UM(Rj\zeta) \quad (2)$$

where is:

$$UM(P\zeta) = \{\gamma_1, \gamma_2, ..., \gamma_r\} \quad (3)$$

set of primary problems

Let us define primary set of problems:

<Definition 4>

Primary set of problems is that one where no one problem can be decomposed to the sub-problems.

Since there is no sense to consider is it possible to decompose problem or not, we can talk is it necessary to decompose problem or not. When all problems within problem set are on that stage of complexity, we can consider it as primary set of problems.

Consider that set (3) is not primary set by default, rather is necessary to make it primary by problem decomposition. Mechanism of decomposition requires methodology that is out of scope of this paper.

Further, by definition exist:

$$UM(Rj\zeta) = \{R_1, R_2, ..., R_s\} \quad (4)$$

which represents primary set of solvers e.g. (intelligent) agents (R_s).

Since problem solving is distributed process, it is reasonable to define set of locations

$$L = \{\lambda_1, \lambda_2, ..., \lambda_m\} \quad (5)$$

This is set of all possible locations within network on which elements of sets (3) and (4) can exist.

If we assume probability of problem appearance at particular node as constant during the time (and it is true for short time intervals), probability that one problem (γ_r) appears at particular location (λ_m) can be considered as element of matrix m*r, matrix of problem arising probability:

$$\sigma(\lambda,\gamma) = \begin{matrix} \sigma(\lambda_1,\gamma_1) & \sigma(\lambda_2,\gamma_1) & ... & \sigma(\lambda_m,\gamma_1) \\ \sigma(\lambda_1,\gamma_2) & \sigma(\lambda_2,\gamma_2) & ... & \sigma(\lambda_m,\gamma_2) \\ ... & ... & ... & ... \\ \sigma(\lambda_1,\gamma_r) & \sigma(\lambda_2,\gamma_r) & ... & \sigma(\lambda_m,\gamma_r) \end{matrix} \quad (6)$$

Let us define priority of every problem during solving process. Priority is defined indirectly, by maximum allowed time to solve the problem, so called maximum allowed reaction time:

$$\omega(\lambda,\gamma) = \begin{matrix} \omega(\lambda_1,\gamma_1) & \omega(\lambda_2,\gamma_1) & ... & \omega(\lambda_m,\gamma_1) \\ \omega(\lambda_1,\gamma_2) & \omega(\lambda_2,\gamma_2) & ... & \omega(\lambda_m,\gamma_2) \\ ... & ... & ... & ... \\ \omega(\lambda_1,\gamma_r) & \omega(\lambda_2,\gamma_r) & ... & \omega(\lambda_m,\gamma_r) \end{matrix} \quad (7)$$

Namely, some problems may have very small probability of arising at one location, but very high solving priority, e.g. very low reaction time. It is necessary to have solver that can reacts within predefined, very small, time interval, regardless on small probability of problem.

Finally, it is reasonable to define schedule of solvers, which defines how they are scheduled within the network, by locations:

$$\delta(R,\lambda)\begin{array}{c|cccc} & R_1 & R_2 & \ldots & R_s \\ \lambda_1 & 1 & 1 & \ldots & 0 \\ \lambda_2 & 0 & 0 & \ldots & 1 \\ \ldots & \ldots & \ldots & \ldots & \ldots \\ \lambda_m & 1 & 0 & \ldots & 1 \end{array} \quad (8)$$

If matrix element $\delta(R_4,\lambda_2)=1$, it means solver R_4 exist on location λ_2, while value 0 means solver doesn't exist on that location.

Based on set (3) we approach to definition and specification of solvers' set, which contains problem solvers (set (4)).

Every member of set (3) requires an authentic solver from the set (4). During the process of solver choosing it is necessary to take into the account about solver schedule within the network (matrix (8)), as well as probabilities of problem existing at particular network location (matrix (6)).

3. PROBLEM SOLVING

Since agents from the set (4) have been chosen from the solver space according to the set (3) and tables' states, we can conclude that election is indirectly done by problem environment within domain of problem space.

Hence, it is reasonable that agents from the set (4) will be "united" on the task of goal reaching; they will make agent system (definition 3). Even if agents will have goals that are in conflict with each other, it is necessary to define minimal amount of communication between them. Protocol defines agent schedule and conversation rules between agents; hence it is necessary to be focused on:

- Common goals recognition
- Common tasks recognition
- Avoiding of undesired conflicts
- Putting knowledge and resources "together" in order to solve the problem optimally
- Minimization of "undesired" communication (moving of agents between network nodes which can be avoided by optimal solvers' schedule)
- Tracing and updating data from (6), (8) and (7), which means re-grouping of agents within the network.

3.1. Programming simulation

Definitions that are recommended can be verificated very efficiently by programming simulation. Namely, object-oriented programming techniques give an opportunity to make a model of every process participant as one object, with environmental interface that is very clearly defined. Behavior of every object is predefined independent, but after systems startup results can be presented and perceived in global term. To reach that goal it is necessary to develop application with following features:

Parameters are given in advance, or can be changed during runtime, but results are presented according to parameters. Results are:

- Problem space and problems' sets created from it
- Solver space and solvers' sets with features and specifications which solver has resources to solve every problem or set of problems that arises.
- Location list within the network with predefined relations (connectivity, distance, …)
- Probabilities of problems arising at locations, for every problem and every location
- Maximum reaction times for every problem at every location

Every management resource contains knowledge that controls management decision-making process. Every decision controls only one small segment of distributed system; hence it is necessary to communicate.

There are several performances that can be measured by simulation depending on solving problem way and inter-agents communication, as well as agents' schedule within the network:

- Average solving time
- Maximum/minimum solving time
- Number of maximum time violations
- Amount of communication
- …

3.2. Programming simulation results

Here is an example of programming simulation results, which are presented in non-graphic way at the moment.

Input file, ulaz.txt, contains all input parameters:

```
10                          total number of nodes in the network
0 1 2 1 2 2 2 1 2 2
1 0 1 1 1 2 2 2 2 1
2 1 0 2 1 2 2 1 1 2     network topology, matrix presenting
1 1 2 0 2 2 1 2 2 1     how nodes are connected:
2 1 1 2 0 1 1 1 1 1       "1" → nodes i i j are connected
2 2 2 2 1 0 1 2 2 1       "2" → nodes i i j are not connected
2 2 2 1 1 1 0 1 2 1       "0" → i=j
1 2 1 2 1 2 1 0 1 2
2 2 1 2 1 2 2 1 0 1
2 1 2 1 1 1 1 2 1 0
```

7	total number of agents in the network
5 0 2 4 5 10 3	agent schedule in the network (by nodes)
10	total number of primary problems

1 0 1 0 0 1 0 0 1 1	
0 1 1 0 0 0 1 1 1 0	
0 1 0 0 0 0 0 0 0 0	
0 0 0 1 0 1 1 0 0 1	matrix of problems solved by
0 1 1 0 0 0 0 1 0 0	particular agent
0 0 0 0 0 0 1 0 0 1	
0 1 0 0 0 1 0 0 0 1	

This is an output file, izlaz.txt, after simulation running, based on input file, ulaz.txt (txt based results, in Croatian language):

Warning! Problem in the node 2
Problem code: 0101011011
Date: 29.3.2002.
Time: 19:05:00:083

Minimal distances between node 2 and all other nodes:

Distance between nodes 2 and 0: 2
Distance between nodes 2 and 1: 1
Distance between nodes 2 and 2: 0
Distance between nodes 2 and 3: 2
Distance between nodes 2 and 4: 1
Distance between nodes 2 and 5: 2
Distance between nodes 2 and 6: 2
Distance between nodes 2 and 7: 1
Distance between nodes 2 and 8: 1
Distance between nodes 2 and 9: 2

Agent(s) in the network that can solve the problem:
1) Agent 2 (at node 2)
2) Agent 3 (at node 4)
3) Agent 0 (at node 5)

It's needed 3 agents in order to solve the problem

4 CONCLUSION

In this paper we've presented the "infrastructure" for formal presentation of agent model as "problem solving agent".
Hence we've defined problem environment, problem space and problem set as well as solver space and set of problem solvers.

Every segment of network management process contains only segment of knowledge required for management decision-making it's necessary to ensure mechanisms of communication and knowledge (information) transfer. It is also necessary to define optimal way of communication and knowledge transfer, depends on predefined parameters. One way of solution is programming simulation that will be presented in future works in more details.

REFERENCES

[1] Wooldridge, M., N.R. Jennings, Intelligent agents: Theory and practice, The Knowledge Engineering review, 10(2):115-152, 1995.
[2] Russell, S., P. Norvig, Artificial Intelligence: A Modern Approach, Prentice-Hall, 1995.
[3] Weiss, G., Multiagent Systems, The MIT Press, Cambridge, Massachusetts, London, England
[4] Kinny, D., M. Georgeff, Commitment and effectiveness of situated agents Proceedings of the Twelfth International Joint Conference on Artificial Intelligence (IJCAI-91), pp. 82-88, Sydney, Australia, 1991.
[5] Fisher, M., A survey of Concurrent METATEM – the language and its applications, In D.M.Gabbay and H.J.Ohlbach, editors, Temporal Logic – Proceedings of the First International Conference (LNAI Volume 727), pp. 480-505, Springer-Verlag, Berin, Germany, July 1994.
[6] Austin, J. L., How to do Things with Words, Clarendon, Oxford, United Kingdom, 1962.
[7] Searle, J. R., Speech Acts: An Essay in the Philosophy of Language, Cambridge U. Press, 1970.
[8] Wellman, M. P., A Computational Market Model for Distributed Configuration Design, AI EDAM 9:125-133, 1995.
[9] Kunštić, M., F.Pogačnik, N.Sandri, " A Global Model for Solving the Management Problems in Telecommunication Systems", Proc. WMC"97, Phoenix, Arizona, 1997.
[10] Tkalčić, H.: "Modeli pokretljivosti agenata u procesima upravljanja mrežom", diplomski rad, april 2002, Fakultet elektrotehnike i računarstva Sveučilišta u Zagrebu