AGENT ORIENTED TECHNOLOGY OF DECENTRALIZED SYSTEMS BASED ON THE M-AGENT ARCHITECTURE

Ewa Cetnarowicz Edward Nawarecki
Krzysztof Cetnarowicz

Institute of Computer Science
University of Mining and Metallurgy - AGH
E-mail: cetnar@uci.agh.edu.pl

Abstract: An idea of autonomous agents that arises as an extension of the object and process concepts may be applied to distributed and decentralized systems development. In the paper the authors have undertaken an attempt to describe formally the architecture of multi-agent systems (called M-agent architecture) that may be considered as a starting point to develop a decentralized multiagent system technology such as Agent Oriented Analysis, Design and Programming.

Keywords: agents, decentralized systems, systems design, system analysis

1. INTRODUCTION

Although multi-agent systems have been studied for nearly ten years the technology of such systems creation has not been proposed yet. There are some approaches ((Crowley J. L., 1993)), or others known as BDI architecture ((Kinn Y. and A., 1996), (Rao, 1996)), that make some attempts to the practical model of multi-agent world definition, but the research has not been completed yet and the results are not always practically applicable.

An idea of M-agents that arises as an extension of the object, process ((Cetnarowicz K., 1995)) and multi-model control systems concepts ((Binder Z. et al., 1974) (Binder Z., 1974)) may be applied to distributed and decentralized systems development.

In the paper a proposition of a formal definition of the M-agent architecture, multi-agent world including the definition of agents, agents’ space, agents’ environment and the relations between agents and environment, and among agents themselves is presented. The approach proposed, based on the concept of successive refinements of created constructions, gives tools that enable to define multi-agent system for a given problem without using any programming language (Cetnarowicz K., 1995), (Nawarecki E., 1993).

2. ABSTRACTION LEVELS AS A TOP-DOWN ANALYSIS AND DESIGN METHOD FOR MULTI AGENT SYSTEMS (MAS) CREATION

For a given problem a corresponding multi-agent system may be created. To design the multi-agent system we have to define:
- living space for agents,
- agents of several types (kinds),
- relations:
  * between agents and the living space,
  * among agents themselves.

The agent architecture may be considered from two different points of view:
- as an intellectual profile describing the agents ability to resolve a given problem,
Fig. 1. Abstraction levels that are used for MAS development

- as an energetical profile that describes the "living energy" of the agent and enables the elimination of redundant agents.

2.1 Definition of multi-agent systems by the abstraction levels

The proposed approach using abstraction levels for successive improvement of the system definition gives a tool to map a given real problem in the notion of the MAS (fig. 1).

The top-down MAS construction with the use of the abstraction levels follows the rules:

- Definition of the system is done by the levels of specifications, called abstraction level, from the top - most general to the bottom - lowest level. The lowest abstraction level which defines the final version of the system, may be specified in programming languages.
- At every abstraction level the formal model of the MAS is described. It enables to define the model of the created system taking into consideration the properties of the system corresponding to this abstraction level.
- Passing from a given abstraction level to the lower one we adopt a more complicated model of the system taking more properties into consideration. It is possible to apply several approaches at a given abstraction level (several versions of abstraction may be deduced). It allows to analyze a given problem from different points of view.

The method proposed may be considered as an application of the Top-Down analysis and design method to the MAS creation. With the use of the method the following advantages may be available:

- It is possible to apply the proposed method to a wide range of MAS very simple (composed of several abstraction levels) to a more complex (composed of several dozens of levels) containing simple - reactive agents and complicated - cognitive agents.
- Applications of various mathematical formalism for the system specification are possible. It is reasonable in case of such complicated and large systems as MAS.
- Considering several different abstraction levels deduced from a given one enables to create different types of agents. It makes possible to create MAS with cooperation of different kinds (reactive and cognitive) of agents.
- The approach is very flexible for the analysis and design work organization. It enables decomposition of tasks at every abstraction level following the organizational needs.

The approach proposed, based on the concept of successive refinements of created constructions, gives tools that enable to define multi-agent system for a given problem without using any programming languages (Cetnarowicz K., 1995), (Nawarecki E., 1993). The programming environments may be defined at the lowest abstraction level with the implementation environment.

3. DESIGN OF MULTI-AGENT SYSTEM BASED ON THE ABSTRACTION LEVELS APPROACH AND THE IDEA OF M-AGENT ARCHITECTURE

As an example a definition of multi-agent world for a graph-like space is given. There is a graph like space. In every node there is a defined amount of resource r. The resource may be produced, and consumed in the node and transferred between the nodes, so the quantity of the resource in a given node varies and is not predictable. For every node two limits are defined: rmax and rmin. The goal of the multi-agent system is to keep the quantity of the resource between rmax and rmin in every node.

3.1 Abstraction level 0 and level 1

3.1.1. level 0

A given agent a remains and acts in an environment called V (fig. 2).

For any created agent the following sets are defined: set of strategies S (s - strategy s ∈ S), set of goals Q (q - goal q ∈ Q), and set of models of the environment M (m - model of the environment, m ∈ M).

The main idea of the agent functioning is as follows:

- Agent observes the environment around and builds a model m of the environment in its mind. For this purpose it uses the observation (or imagination) operation I.
Fig. 2. Principle of the M-agent architecture. I - imagination operation, X - execution operation, m, m' - models of the environment, s - strategy, q - evaluation function, e_i - events taking place in the environment.

- Agent performs a forecasting of its possibilities using the strategy s. Applying a strategy s to model m it obtains model m' of the modified environment.
- Agent compares in its mind models m and m' using function q that determines the goal of the agent functioning.
- If the evaluation of m and m' is satisfactory the agent realizes strategy s that is represented by operation X - execution: V' = X(s, V)

The first approach to the algorithm of a given agent may be expressed in three stages:

Observation - creation of the model m of environment V:

\[ m = I(M, V) \] \hspace{1cm} (1)

Decision - selection of optimal strategy s*:

\[ s^* = F(M, Q, S, V) \] \hspace{1cm} (2)

Action - realization of optimal strategy s* in the environment V:

\[ V' = X(s, V) \] \hspace{1cm} (3)

where:

M, Q, S - sets of models, goals, strategies represent knowledge of agent a, V - environment (structure and state of environment), I - observation function, F - decision function enabling a selection of optimal strategy s*. X - function representing operation of the realization of agent’s strategies.

3.1.2 level 1

Abstraction level 1 describes M-agent architecture, structure of M, Q, S, operations I, X, general algorithm of agent a functioning with autoadaptation operations. The formal definition of M-agent architecture may be found in: ((Cetnarowicz, 1996)).

3.2 Design and Realization of Multi Agent System abstraction level 2 and level 3

3.2.1. design - level 2

Definition of the space \( E = (T, R) \) where: \( T = \{ (t_{i,j}: i \in I_n, j \in I_{n_2}, B) \} \)

\[ B = \{ (t_{i,j}, t_{k,l}) : ((-1 \leq (i - k) \leq 1) \cup ((i = 1) \land (k = n_1)) \cup ((i = n_1) \land (k = 1)) \}
\]

\[ \land ((-1 \leq (j - l) \leq 1) \cup ((j = 1) \land (l = n_2)) \cup ((j = n_2) \land (l = 1)) \land ((i \neq k) \land (j \neq l)) \} \}

\[ R = \{ r : (r : t_{i,j}, i \in I_n, j \in I_{n_2}) \rightarrow \mathbb{R} \} \}

\[ r(t_{i,j}) = r_{ij} \]

\[ r = \{ r_{ij} \} \]

\[ r_{max} = \{ r_{max(i,j)} \} \]

\[ r_{min} = \{ r_{min(i,j)} \} \]

\[ \forall i \in I_n, j \in I_{n_2} \hspace{0.5cm} r_{max(i,j)} \leq r_{ij} \leq r_{min(i,j)} \]

\[ E = \{ r_{ij}, r_{max(i,j)}, r_{min(i,j)} \} \]

\[ V_a = (E, A_a, C_a) = \]

\[ \{ E = \{ r_{ij}, r_{max(i,j)}, r_{min(i,j)} \} \}

\[ A_a = \phi, \]

\[ C_a = \{ (i_a, j_a) \} \]

There are three types of agents in the designed Multi Agent world: \( G = \{ 0, 1, 2 \} \).

Definition of agent \( g = 0 \):

Definition of \( M^0 \):

\( M^0 = \{ m^0: m^0 = (\mu^0, \alpha^0, \beta^0, \delta^0, \gamma^0) \} \),

\( \alpha^0, \beta^0, \delta^0, \gamma^0 \in \mathbb{R} \),

\( \mu^0: I_{I_1} \times I_{I_1} \rightarrow \mathbb{R} \)

where: \( I_{I_n}, I_{I_{n_2}} - indexing sets, \)

\( I_{I_n} = (-n, -n + 1, \ldots, -1, 0, +1, \ldots, n - 1, n) \)

\( \alpha^0 \) is the amount of resource that may be obtained from one of the neighbouring nodes.

\( \beta^0 \) is the amount of resource that may be sent to one of the neighbouring nodes.

\( \delta^0 \) is the number of agents of type \( g = 1 \) created up to now by the considered agent.

\( \gamma^0 \) is the number of agents of type \( g = 2 \) created.
Definition of $s^0$: $\forall (i, j) \in I_1 \times I_1 \setminus \{(0,0)\}$:

$$s^0 = \{ s_{ij}^0, s_{ij}^0, a_{k}^0 \}$$

$s_{ij}^0$-strategy aiming to obtain some resource from one of the neighbouring nodes.

$$s_{ij}^0(m) = s_{ij}^0((\mu^0, \alpha^0, \beta^0, \delta^0, \gamma^0)) = \begin{cases} \{ (\mu^0, \alpha^0, \beta^0, \delta^0, \gamma^0) \text{ for } \mu^0(i,j) > 0 \\
(\mu^0, \alpha^0, \beta^0, \delta^0, \gamma^0) \text{ for } \mu^0(i,j) \leq 0 
\end{cases}$$

where $\alpha^0 = \alpha^0 + \mu^0(i,j)$

$s_a^0$-strategy aiming to send some resource to one of the neighbouring nodes.

$$s_a^0(m) = s_a^0((\mu^0, \alpha^0, \beta^0, \delta^0, \gamma^0)) = \begin{cases} \{ (\mu^0, \alpha^0, \beta^0, \delta^0, \gamma^0) \text{ for } \mu^0(i,j) < 0 \\
(\mu^0, \alpha^0, \beta^0, \delta^0, \gamma^0) \text{ for } \mu^0(i,j) \geq 0
\end{cases}$$

where $\beta^0 = \beta^0 + \mu^0(i,j)$

$s_k^0$-strategy aiming to create an agent of type $g = 1$.

$$s_k^0(m) = s_k^0((\mu^0, \alpha^0, \beta^0, \delta^0, \gamma^0)) = (\mu^0, \alpha^0, \beta^0, \delta^0, \gamma^0)$$

where $\delta^0 = \delta^0 + 1$

$s_l^0$-strategy aiming to create an agent of type $g = 2$.

$$s_l^0(m) = s_l^0((\mu^0, \alpha^0, \beta^0, \delta^0, \gamma^0)) = (\mu^0, \alpha^0, \beta^0, \delta^0, \gamma^0)$$

where $\gamma^0 = \gamma^0 + 1$

Definition of $Q^0$: $Q^0 = \{ q^0_k, q^0_0 \}$.

$q_0^k$ goal to obtain some resource for the considered node.

$$q_0^k(m^0, m_{\text{out}}^0) = q_0^k((\mu^0, \alpha^0, \beta^0, \delta^0, \gamma^0), (\mu^0, \alpha^0, \beta^0, \delta^0, \gamma^0)) = \begin{cases}
\alpha^0 - \alpha^0 \text{ for } \mu^0(0,0) < 0 \\
\delta^0 - \delta^0 \text{ for } \mu^0(0,0) < 0 \\
0 \text{ for } \mu^0(0,0) \geq 0
\end{cases}$$

$q^0_k$ goal to send from some resource the considered node.

$$q^0_k(m^0, m_{\text{out}}^0) = q^0_k((\mu^0, \alpha^0, \beta^0, \delta^0, \gamma^0), (\mu^0, \alpha^0, \beta^0, \delta^0, \gamma^0)) = \begin{cases}
\beta^0 - \beta^0 \text{ for } \mu^0(0,0) > 0 \\
\gamma^0 - \gamma^0 \text{ for } \mu^0(0,0) > 0 \\
0 \text{ for } \mu^0(0,0) \leq 0
\end{cases}$$
$r_{k+i,i+j} = s_0((\mu^1, \tau^1)) = \begin{cases} (\mu^1, \tau^1') & \text{for } \mu^1(0,0) > 0 \\ (\mu^1, \tau^1) & \text{for } \mu^1(0,0) \leq 0 \\ \end{cases}$

where $\tau^1' = \tau^1 + \mu^1(0,0)$

$s_{i,j}^1(m) = s_{i,j}^1((\mu^1, \tau^1')) = (\mu^1, \tau^1)$,

$\mu^{1,i,l} = \mu^{1,i,l}(k,l) = \begin{cases} \mu^{1}(k+i,l+j) & i f \quad -1 \leq k + i \leq 1 \\ \mu^{1}(0,0) & \text{otherwise} \end{cases}$

where $k \in I_1, l \in I_1$, $(i,j) \in (I_1 \times I_1) \setminus \{(0,0)\}$, $\nabla$ - undefined.

Definition of $Q^1$: $Q^1 = \{q_i^1, q_j^1\}$

$q_i^1(m, m^{1'}) = q_i^1((\mu^1, \tau^1'), (\mu^{1,i,l}, \tau^{1,i})) = \begin{cases} 1 + (\mu^{1}(0,0) - \mu^1(0,0)) & i f \quad \mu^1(0,0) \leq 0 \\ \mu^1(0,0) & \text{otherwise} \end{cases}$

$\mu^{1,i,l} = \mu^{1,i,l}(k,l)$

where: $\tau^1 = 0$

$\forall i \in I_1, j \in I_1: \mu^{1}(i,j) = \begin{cases} r(k+i,l+j) - r_{\text{max}}(k+i,l+j) & i f \quad r(k+i,l+j) > r_{\text{max}}(k+i,l+j) \\ r(k+i,l+j) - r_{\text{min}}(k+i,l+j) & i f \quad r(k+i,l+j) < r_{\text{min}}(k+i,l+j) \\ 0 & \text{otherwise} \end{cases}$

Definition of $X^1$:

$X^1(s_0^1, (r, (k,l)))$ - FROM the NODE $t(k,l)$ TAKE AMOUNT $r_{\text{max}}(k,l) - r(k,l)$ of the RESOURCE AND SEND IT TO the HOME NODE THEN STOP.

$X^1(s_{i,j}^1, (r, (k,l)))$ - GO FROM the NODE $t(k,l)$ TO the NODE $t(k+i,l+j)$.

Definition of agent $g = 2$:

Definition of $M^2$: $M^2 = \{m^2 : m^2 = (\mu^2, \tau^2)\}$

$\tau^2 \in \mathbb{R} \quad m^2 : \mathbb{I}_1 \times \mathbb{I}_1 \rightarrow \mathbb{R}$

where:

$\mathbb{I}_n = (-n, -n+1, \ldots, -1, 0, 1, \ldots, n-1, n)$

Definition of $\mu^2$ is like for agent $g = 1$ where upper index $1$ is replaced by $2$. Definition of $S^2$: for $(i,j) \in \mathbb{I}_1 \times \mathbb{I}_1 \setminus \{(0,0)\}$:

$S^2 = \{s_0^2, s_{i,j}^2\}$

where $s_0^2$ - put to $t(k,l)$ some resource taken from home node. $s_{i,j}^2$ - go from $t(k,l)$ to $t(k+i,l+j)$,

$s_{i,j}^2(m) = s_{i,j}^2((\mu^2, \tau^2)) = \begin{cases} (\mu^2, \tau^{2,i}) & \text{for } \mu^2(0,0) < 0 \\ (\mu^2, \tau^2) & \text{for } \mu^2(0,0) \geq 0 \end{cases}$

where $\tau^{2,i} = \tau^2 - \mu^2(0,0)$

Definition of $s_{i,j}^2(m)$ is like for the agent of type $g = 1$ where upper index $1$ is replaced by $2$.

Definition of $Q^2$: $Q^2 = \{q_i^2, q_j^2\}$

$q_i^2(m, m^{2'}) = q_i^2((\mu^2, \tau^2), (\mu^{2,i,l}, \tau^{2,i})) = \begin{cases} 1 + (\mu^2(0,0) - \mu^2(0,0)) & i f \quad \mu^2(0,0) \geq 0 \\ 1 & \text{otherwise} \end{cases}$

Definition of $P^2$:

$P^2(M^2, V = \{r, r_{\text{max}}, r_{\text{min}}, (0, (k,l))\}) = m^2 = (\mu^2, \tau^2)$

where: $\tau^2 = \text{resource taken from home, and for } i \in \mathbb{I}_1, j \in \mathbb{I}_1 \mu^2(i,j)$ is defined like for agent $g = 1$ where upper index equal is replaced by $2$.

Definition of $X^2$:

$X^2(s_0^2, (r, (k,l)))$ - PUT TO the NODE $t(k,l)$ AMOUNT $\tau^2$ of the RESOURCE THEN STOP.

$X^2(s_{i,j}^2, (r, (k,l)))$ - GO FROM the NODE $t(k,l)$ TO the NODE $t(k+i,l+j)$

3.2.2. practical realization of the system - level 3

The model defined above has been applied to the resolution of load balancing problem with the use of the environment composed of 400 processors joined together in the form of torus.

Every processor of the structure is equipped with the operating system that provides execution of the tasks and the task transfer by the channels and by the bus. The goal of the multiprocessor system is to compute a group of tasks. The main function of the operating system proposed above is to ensure such load a load distribution among the processors of the structure so that the execution of all the tasks could be completed in the shortest possible time. For this purpose the above specified multi-agent system may be applied. The resource distributed around the multi-processor structure represents the tasks to be executed, the resource consumption corresponds to the tasks’ execution and resource generation to the tasks creation (Cetnarowicz K., 1995), (Nawarecki E., 1993).
In the realized multi-agent system the agent of type \( g = 0 \) that resides at a given processor which, for instance, needs tasks (or has excess of tasks) can create the agent of type \( g = 1 \) (or \( g = 1 \)) that has a mission to find some tasks (or find a processor that has a lack of tasks). The generated agent \( g = 1 \) (or \( g = 2 \)) navigating around the whole multiprocessor structure looks for a processor which has an excess (or a lack) of tasks and when it finds it, the agent initiates the transmission process with the use of the bus.

In practical verification of the system proposed the efficiency \( E \) of the multiprocessor system attained 70 percent (Cetnarowicz K., 1995).

4. CONCLUSIONS

The top-down MAS construction with the use of the abstraction levels and the model of the agent presented in this paper has the following advantages:

- Possibility to define multi-agent system independently of programming the language to be applied.

- Description of the multi-agent system is general enough to give a free choice of particular formalism to describe different parts of the system and then the way of practical realization.

- General methods of agent functioning in the environment defined in the proposed model make the whole agent algorithm specification more simple.

The proposed top-down MAS construction with the use of the abstraction levels and the architecture of the M-agent enables to create the knowledge representation method for the decentralized system specification. It may be a useful approach to the technology of MAS creation - the Agent Oriented Analysis and Design processes.

5. REFERENCES


