A Holonic Approach for Providing Composite Services

Irina Diaconița, Florin Leon, Andrei Dan Leca
Department of Computer Science and Engineering
Technical University “Gheorghe Asachi” of Iaşi
Iaşi, Romania
irina.diaconita@gmail.com, fleon@cs.tuiasi.ro, andy_leca@yahoo.com

Abstract—**Holonic systems are a promising development of multiagent systems, where a holon is simultaneously a whole, composed of sub-structures, and a part of a larger entity, thus demonstrating a self-similar or fractal configuration. In this paper, we present a holon-based approach for complex multimedia processing based on elementary services that can self-organize in order to perform complex tasks. The reputations of agents are taken into account and a protocol is described that demonstrates the formation and stability of holonic coalitions that offer high quality services.**

Keywords-holonic multiagent systems; coalitions; composite services; agent reputation.

I. INTRODUCTION

A multiagent system (MAS) consists of a collection of individual agents, each of which displays a certain amount of autonomy with respect to its actions and perception of a domain [1]. While the traditional specification of a problem solving method at design time can be difficult or sometimes even unfeasible, MAS focuses on the interaction of the individual agents at run-time, and is more concerned about the way in which a solution can emerge from the interactions.

A further development on the theory of MAS is the concept of a holonic multiagent system. A holon is a self-similar or fractal structure that is stable and coherent and that consists of several holons as sub-structures [2]. Thus a holon is a complex whole that consists of substructures, and it is as well a part of a larger entity. It uses recursively nested self-similar structures which dynamically adapt to achieve the design goals of the system. In a holonic MAS, autonomous agents group together to form holons. However, in doing so they do not lose their autonomy completely. The agents can leave a holon and act autonomously again or they can rearrange themselves as new holons. According to this view, a holonic agent consists of sub-agents, which can separate and rearrange themselves and which may be holons themselves [3].

Following the general characterization of a holon, some principles can be ascertained [4]: • A holonic system possesses a tree structure, or it can be seen as a set of interwoven hierarchies; • A holon obeys precise principles, but is able to adopt different strategies according to its need; • The complex activities and behavior are situated at the top of the hierarchy, while the simple, reactive acts are to be found at the base of the holarchy; • The communications must follow the hierarchy. Messages are only possible between a holon and its responsible agent, or between holons on the same layer.

Holons have found applications in a variety of domains. A comprehensive survey [5] identifies real industrial applications such as: shipboard automation distributed control and diagnostics, production planning of engine assembling, air traffic control, and RFID-enabled material handling control. In the following paragraphs, we briefly describe some typical applications of holonic systems.

The TeleTruck system [6] is an example of an online order dispatching system for a transport company. Its task is to compute routes for a fleet of trucks for a given set of customer orders, and also handle online scheduling requests, in which new orders can arrive at any moment, and problems in the execution of the computed plans can appear.

A problem for which holons are very fit is train coupling and sharing [7]: a set of train modules are able to drive on their own on a railway network; however, if all the train modules drive separately, the capacity utilization of the railway network is not acceptable. The idea is that the module trains join together and jointly drive some distance, thus the overall goal is to reduce the cost for a given set of transportation tasks.

Another application is manufacturing scheduling, involving the allocation of jobs to machines over time, within a short temporal horizon and according to a specific criterion, such as cost or tardiness [8]. The authors suggest the use of three types of holons to handle the scheduling and control at shop floor level: task holons (production orders), operational holons (physical resources or operators available), and supervisor holons (which provide co-ordination and optimization services to the holons under their supervision).

In our paper, we analyze the use of holonic intelligent architecture to develop a cluster-based distributed application for complex media processing on demand, where the core services are libraries used in various combinations.

The structure of the article is as follows. In Section II, we describe the proposed protocol for holon interaction, including a simplified model of the underlying physical network infrastructure. In Section III, we present some case studies regarding the actual implementation of a holonic multiagent system whose behavior follows the above

 ISBN: 978-1-61208-155-7

mentioned design principles. Section IV presents the conclusions of our work.

II. Protocol Description

The development of computer systems has always had the problem of the optimal use of hardware resources. The main reason for this constraint used to be the higher price of hardware. For dedicated applications that require high amounts of computer power this restriction remains true. But in the present information society computer applications enter each aspect of our lives, and even change the way we interact or work. As a result, large amounts of high and medium power computing nodes begin to be available both at the organizational level and at personal level. This involves changes in thinking software development itself. New approaches such as service-oriented architectures arise. This has many advantages because the granularity of the system can be quickly modified in accordance with the architecture chosen for the development, but the basic elements used in software construction remain the same. Also, the security aspects can be much better handled due to the inherent encapsulation on each level of the model. Thus, no matter the solution offered by the combination of hardware and operating system (such as grid, cloud, or other solutions), there is a support for the service-based applications.

For the application of the proposed model for holon collaboration and agent transfer, we consider a homogenous network with a constant transfer rate, where each machine initially hosts one holon, containing agents which offer specific elementary services. The restriction of having one top-level holon on each machine was enforced in order to avoid load balancing problems, e.g. when all the agents gathered on one or a few computers, and also to reveal that even if at the beginning there could be a positional bias for holons placed on machines closer to the client gateway, in time the holons with a better quality of service are favored.

The client agents send requests for composite services to the gateway agent, which will make calls for offers and choose the best ones. Thus, the clients only specify the combination of elementary services they need and the order of complexity of the tasks. The offer selection is provided by the gateway agent and the required service is controlled by the holons.

A holon consists of a representative agent and other agents or holons specialized in providing specific elementary services (we call a type *i* agent an agent specialized in providing the *i* service). A holon may contain one or more agents of the same type. Each service agent has its own quality, not known by others beforehand, for the service it provides, which is expressed as a percentage. Each holon has its own estimates for the quality of all service agents, also known as the reputations of the agents. Initially, the holons consider the quality of all service agents to be 100% and update the quality of the agents they use based on the feedback they receive from the client agents, after providing the requested services.

The representative agent is responsible for all the reputation estimation, communications, bidding, and agent transfer on behalf of the holon, as well as for the supervision of the tasks carried out by the service agents. Each representative knows the representatives of the other holons and the gateway agent, but does not know the clients.

In the following, we explain the way in which a request received by the gateway is treated.

In the first stage, the gateway agent sends a call for offers to all holons. When a holon receives such a request, its representative checks if there are local available agents for all the elementary services needed, with a reputation of at least ρ_{min}. If this condition is met, it sends an offer with the estimated processing time to the gateway:

$$t_p^{estimated} = \frac{C_{in}^2}{res \cdot n} + t_{dt}, \quad (1)$$

where C_{in} is the order of complexity of the tasks (as estimated by the client), *res* represents the resources of the machine, *n* is the estimated number of agents that will work simultaneously on the machine, and t_{dt} is the data transfer time from the gateway to the holon, defined as:

$$t_{dt} = \frac{C_{in}^2}{v_d \cdot n_{hi}}, \quad (2)$$

where v_d is the data transfer rate and n_{hi} the number of hops between the gateway and the holon (on the least cost path, as calculated by Dijkstra's algorithm [9]).

Several clarifications are needed concerning these formulas. The complexity of a task C_{in} and the resources of a machine *r* are generic. The complexity is squared because we assume that the processing would mainly follow an $O(n^2)$ complexity, which is not unusual when dealing with multimedia processing. Of course, it could also be greater, and in this case the equations should be changed accordingly. The resources mainly refer to the processing power of the machine; different real-world configurations could be eventually reduced to a number *r* that reflects the speed of the processing, affected for example by the processor performance, memory capacity, and harddisk size.

If the holon has suitable agents for all services, but one or more of them are busy at the moment, the representative sends a message to inform the gateway that it cannot offer the required services.

In the case that the holon does not have one or more types of agents needed (with a reputation greater than ρ_{min}) for the processing, its representative sends agent transfer requests to the other holons. For each type of agent, it waits until it receives an answer from all holons and then, if there are holons that have agreed to the agent transfer request, it chooses the one with the best reputation. After the holon receives an answer from all holons or after the wait time expires, if it has agents for all the required services, it sends an offer to the gateway agent with an estimated processing time of:

$$t_p = \frac{C_{in}^2}{res \cdot n} + t_{dt} + \sum_{i=1}^{m} \frac{t_a}{n_{hi}}, \quad (3)$$

where m is the number of transferred agents, t_a is the transfer rate for an agent, and n_{hi} is the number of hops between the machines involved in the transfer.

If, on the contrary, the holon still has at least one missing type of agents, its representative sends a message to the gateway, telling that it cannot offer the required service.

When a holon receives a transfer request for a specific type of agent, it checks if it has any available agents of the required type. If this condition is met and the average estimated reputation of the holon is less than $\rho_{cohesion}$, the representative accepts the transfer. Otherwise, it rejects the transfer request.

In the second stage (after the gateway received an answer from all holons or the waiting time expired), the gateway agent chooses the best offer for the given service. If no holon made an offer, the gateway informs the client that the service is unavailable. In the other case, it chooses the best offer – the one with the best processing time – and informs the holon that its bid was accepted and that it should begin processing the data.

When a representative receives a message saying that its offer was accepted, it sends a message to the first agent from the list of agents involved, which contains the size of the task, the composite service, the ordered list of the agents involved in solving the task and the processing time (initially 0).

The moment when a service-providing agent receives a message asking it to do a processing, depending on whether it is busy or not, it adds the request to the waiting list or checks if it belongs to the holon responsible for the task and solves the required task. If it belongs to a different holon, it moves and joins the one that made the request. A holon can send a service or transfer request to an agent only if this action was previously approved by the representative of the holon to which the agent belongs. When an agent finishes a task, it takes the next one from the waiting list, it moves if necessary, and then provides the required service. Finally, the agent adds the time while the request was in the waiting list and the processing time to the current/received value of the computation time, and sends the request to the next agent from the list of agents involved in the processing. If the agent is the last one in the list, it sends a message to its representative that the task has been carried out successfully, and what was the processing time. The representative will forward this message to the gateway agent.

Therefore, the final processing time is:

$$t_p^{final} = t_{dt} + \sum_{i=1}^{n} p_i \cdot t_{p_i} + \sum_{i=1}^{m} t_{w_i} + \sum_{i=1}^{l} t_{m_i}, \quad (4)$$

where t_{dt} is the data transfer time between the gateway and the holon that provides the service, p_i is the number of type i tasks and t_{p_i} is the corresponding processing time, n is the total number of agents involved in solving the task, t_{w_i} represents the waiting time for agent i, m is the number of agents that caused delays, i.e. they were busy when they received the processing request, $\sum_{i=1}^{l} t_{m_i}$ is the sum of the times for moving agents that belong to other holons, t_{m_i} is the sum of times for the moving agent i (after the agent had joined the holon and made a processing it could have moved again to another holon to do another processing, before it had to do another processing in the given holon), and l is the number of the agents that were transferred to the holon.

The hierarchic and reflexive properties that define the holonic models are preserved in our approach. Thus, the holon representative is the one which decides if it will use an agent or if the agent is unfit, and therefore it will search for another to offer the same service. The representative of a holon makes requests for the agents or holons it needs to the representatives of the holons they are part of. Not only the individual agents, but also their representatives must agree to the movement.

After receiving the processed data from the selected holon, the gateway agent sends it to the client agent, which evaluates the quality of the service (as a percentage) and sends this value to the gateway agent, which forwards it to the responsible holon. Then, the holon updates its reputations for the agents involved in the processing as it follows:

$$\rho_{hi}(k+1) = (1-\alpha) \cdot \rho_{hi}(k) + \alpha \cdot \frac{e}{n_a} \cdot n_i \quad (5)$$

where $\rho_{hi}(k)$ represents the reputation of the service agent i estimated by the holon h after the k^{th} time it provided a service as a part of holon h, α is the learning rate, e is the evaluation provided by the client, n_a is the number of agents involved in the task, and n_i is the number of processings agent i did within the given task.

The choice of this function is based on a learning model encountered in other adaptive AI algorithms, such as Kohonen's self-organizing map or Q-learning. Basically, a fraction $(1-\alpha)$ of the old value is replaced with a fraction α of the new one. In our case, the evaluation of the client is equally distributed to all the agents involved.

An important aspect is the use of a variable learning rate, which depends on the number of processings an agent performs:

$$\alpha = \frac{\alpha_0}{n_a} \cdot n_i, \quad (6)$$

where α_0 is a constant.

TABLE I. THE QUALITIES OF THE AGENTS

Agents													
Holon1			Holon2			Holon3		Holon4				Holon5	
A1 (A)	A2 (B)	A3 (C)	A4 (A)	A5 (D)	A6 (E)	A7 (A)	A8 (F)	A9 (A)	A10 (C)	A11 (D)	A12 (F)	A13 (A)	A14 (E)
80	72	63	40	93	87	61	84	89	88	59	77	68	95

TABLE II. THE REPUTATIONS OF THE AGENTS (BEFORE RECEIVING THE REQUEST)

	Agents													
Holons	Holon1			Holon2			Holon3		Holon4				Holon5	
	A1 A	A2 B	A3 C	A4 A	A5 D	A6 E	A7 A	A8 F	A9 A	A10 C	A11 D	A12 F	A13 A	A14 E
Holon1	77	73	68	34	96	88	60	82	87	90	57	60	62	96
Holon2	80	69	71	38	92	86	57	79	88	95	62	67	64	98
Holon3	87	70	61	42	90	81	64	86	82	93	53	68	70	91
Holon4	74	80	68	45	93	85	65	84	90	97	59	65	67	92
Holon5	81	72	65	39	89	82	61	81	85	98	60	70	60	89

TABLE III. HOLONS AND THEIR MEMBERS AT DIFFERENT MOMENTS OF TIME

	Epochs		
	Epoch 2	**Epoch 3**	**Epoch 10**
Holons	Machine 1: H1 (A1, A3)	Machine 1: H1 (A3)	Machine 1: H1 (A3)
	Machine 2: H2 (A4, A5, A6)	Machine 2: H2 (A4, A5)	Machine 2: H2 (A4, A6)
	Machine 3: H3 (A7)	Machine 3: H3 (A1, A2, A6, A7)	Machine 3: H3 (H6 (A1,A2),A5,A7)
	Machine 4: H4(A2,A8,A9,A10,A11)	Machine 4: H4 (A8, A9, A10,A11)	Machine 4: H4 (A8, A9, A10)
	Machine 5: H5 (A13, A14)	Machine 5: H5 (A13, A14)	Machine 5: H5 (A11, A13, A14)

In case when only one agent does all the processing, it is certain that the client evaluation can only refer to that agent's quality. In case when more agents work on a task, the client evaluation is reflected in a different manner on the agents' reputation, according to their contribution to solving the task. The learning rate should be small enough to ensure convergence, therefore if we consider that 0.1 is an acceptable value and the composite services require approximately 3 agents on average, we could choose $\alpha_0 = 0.3$.

III. CASE STUDIES

We will first consider the network and the holons presented in Fig. 1 at the moment when a request arrives. We assume that all agents are free. The qualities of the services provided by the agents are presented in Table I and the reputations of the service agents for each holon are presented in Table II.

In both case studies, we chose $\rho_{min} = 50\%$ and $\rho_{cohesion} = 85\%$.

We used a sequence of requests (*ABFDAB, FAB, DCABAB, DCABDC, ABABAB, DABDAB, EADFAA, DDEB, FAADB, CEDAFBB*), over 1000 epochs to study the formation of new holons and the evolution of the agents' reputations. Table III presents as an example the holons and their distribution over the network at different moments as well as the formation of the first new holon (made up of agents 1 and 2), belonging to holon 3 at the moment. After collaborating for a few services and receiving good evaluations from their clients, agents 1 and 2 form a new holon. From this moment on, they move together and act as a distinct entity, which is still a member of holon 3.

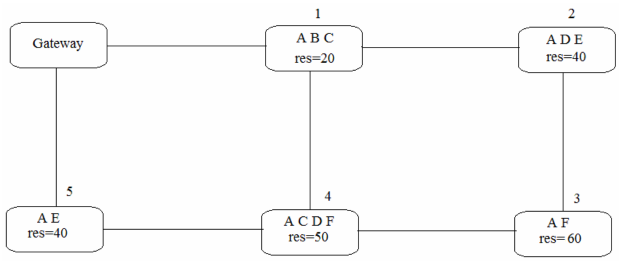

Figure 1. Simple network topology and the holons

The first request is for the service *ABFDAB* (with an order of complexity of 50). In the following, we present the steps taken to solve the request. The gateway agent sends requests for offers to all holons. The holons check whether they have all the types of agents needed. This is not the case for any of the holons, so they all send requests for agent transfers to the other holons. These transfers will be made only if the agent wins the auction, therefore a holon can approve more transfer requests for the same agent. The only criterion for choosing the transfer agent is the agent's reputation. Holon 1 chooses agents 5 (*D*) from holon 2 and 8 (*F*) from holon 3 as possible transfer agents and makes an offer of 93.38 to the gateway agent. Holon 2 chooses agents 1 (*A*) and 2 *(B)* from holon 1 and 8 (*F*) from holon 3 as possible transfer agents and makes an offer of 48.42 to the gateway agent. Holon 2 does not use its own *A* agent because of its low reputation. Holon 3 chooses agents 2 *(B)* from holon 1 and 5 (*D*) from holon 2 as possible transfer agents and makes an offer of 37.90 to the gateway agent. Holon 4 chooses agent 2 (*B*) from holon 2 as a possible transfer agent and makes an offer of 36.75 to the gateway agent. Holon 5 chooses agents 2 *(B)* from holon 1, 5 (*D*) from holon 2 and 8 (*F*) from holon 3 as possible transfer agents and makes an offer of 65.04 to the gateway agent. After receiving all offers, the gateway chooses and asks holon 4 to offer the service (because it has the smallest estimated time). Holon 4 sends a message to the first agent - *A* - to do the first task (and the ordered list of the IDs of the agents solving this request). Agent 9 (*A*) is free; it processes the data and sends a request to agent 2 (*B*). Agent 2 (*B*) is free, but it belongs to holon 1. Therefore, it announces the representative of holon 1 of its transfer, and then moves to holon 4. Afterwards, it processes the data and sends a request to agent 8 *(F)*. Agent 8 (*F*) is free, but it belongs to holon 3. Therefore, it announces the representative of holon 3 of its transfer, and then moves to holon 4. Afterwards, it processes the data and sends a request to agent 11 (*D*). Agent 11 (*D*) is free; it processes the data and sends a request to agent 9 (*A*). Agent 9 (*A*) is free; it processes the data and sends a request to agent 2 (*B*). Agent 2 (*B*) is free; it processes the data and, because it is the last agent from the list, it informs the representative of the holon that the task was carried out successfully. The holon announces the gateway agent that the task was carried out successfully and sends the processed data. The gateway agent sends the processed data to the client agent. The client sends the evaluation of the quality of the service (77.5 %) back to the gateway, and the gateway forwards it to holon 4. Holon 4 updates its reputations for the agents involved; thus, the new reputations will be: $\rho_1 = 87.5$, $\rho_9 = 96.5$, $\rho_{11} = 60.85$ and $\rho_{12} = 66.25$.

It was quite easy to predict that holon 4 would win the auction because it had the smallest number of missing agents – and transferring agents is the most time consuming task. As agent 4 from holon 2 has a bad reputation among all holons because of its low quality of service, it wasn't requested by any holon, and even holon 2 refused to use it and made a request for another agent of the same type. Each holon chose its possible transfer agents from the holons that accepted the transfer, based on the reputation of the agents; different holons chose different transfer agents for the same services, because each holon had its own reputation estimates. After the data was processed, the reputations were updated using (5) with a learning rate of 0.1. As expected, agents with a lower reputation get an undeserved reputation increase because of the agents with higher reputation.

We analyze the variation of the reputation estimates of holon 3 for agents 1 and 2, after they have moved to holon 3 and formed a new holon, over 1000 epochs, when holon 3 wins the auctions and provides services involving both agents.

Fig. 2 shows the fact that the holon's reputation estimates quickly converge to the value of the quality of service of the agents. Initially, the quality of the agents is unknown, and the holon over-estimates it. However, by collaborating with the agents, the estimation quickly converges to a value close to the actual quality of the agent, and remains around that value. Even if it also works with other lower quality agents, its overall reputation does not vary much.

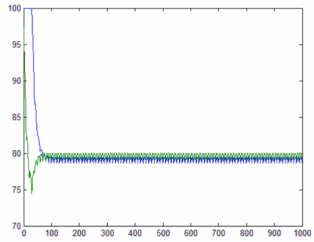

Figure 2. The variation of the reputation estimates made by holon 3 for agents 1 and 2 over 1000 epochs

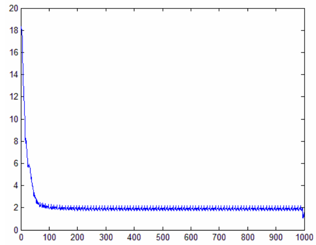

Figure 3. The evolution of the difference between the client's evaluation and holon 3's estimation over 1000 epochs

Fig. 3 presents the evolution of the mean difference between the client's evaluation and the service evaluation (based on the agents' reputation estimates) done by holon 3, over 1000 epochs. At the beginning, the difference between the two evaluations is 18%, but it decreases exponentially and converges to 2% in less than 100 epochs, i.e. the self-evaluation made by the holon converges to the actual quality of the service with a 2% error.

Next, we will consider the network presented in Fig. 4 and we analyze the way the reputations are adjusted for holon 5, when it solves a greater number of requests. There are 37 agents (with their IDs from 0 to 36, given in the order of the holons and the agents) and 8 elementary services. This network is more complex than the previous one and therefore the distances and connections between different machines will play a bigger role in the evolution of the holons. Due to the larger number of agents specialized on each service, the offers are more complex and the agents move a lot more across the network. Also, more new holons appear.

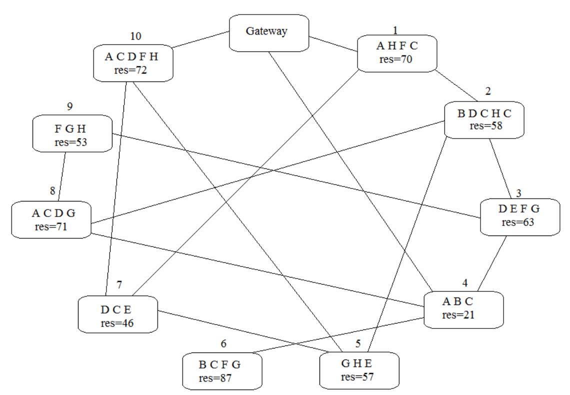

Figure 4. Complex network topology and the holons

Fig. 5 presents the evolution of the mean quality for a given service (*ABFDGFCFC)*, as it is evaluated by the client. In the first epochs the quality decreases because the agents with a better quality of service move to different holons for different tasks or are busy. However, over time, as the holons' estimates for the agents' quality improve and new holons appear, the quality of the composite services improves and remains high throughout the following epochs.

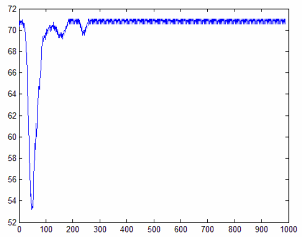

Figure 5. The evolution of the mean quality of a given service

It should be noted that the graphs, the reputations, and the initial positioning of the agents in the case studies were generated in a completely random manner. Therefore, we believe that the interaction model can be useful in any other context that complies with our basic assumptions presented in Section II.

There are many reputation and trust models currently available in the literature. Some approaches aggregate direct experience and indirect recommendations [10] or use probability theory [11]. Others apply typical artificial intelligence techniques such as: Bayesian networks [12], Dempster-Shafer theory [13], or reinforcement learning [14]. Compared to other reputation models, we consider that simplicity and rapidity are the main advantages of our approach. As each holon learns the quality of the other agents at run time, these estimations converge quite fast and the quality of the composite services themselves is shown to increase. Therefore, for the given problem of task allocation and service composition, it seems that our model is efficient.

IV. CONCLUSIONS

We have presented a model for holonic coalition formation and cooperation for providing different services over a homogenous network. The main elements that determine the holonic coalitions are the estimated reputations of the composing agents. At the beginning, all agents are assumed to provide their services with 100% quality, so the agent transfers are determined mainly by the distances between the holon machines, and, to a lesser degree, by the resources of these machines. After a few more epochs, the agents group themselves into holons based on the estimated quality of each other.

Also, at the beginning, the auctions could be won by holons with lower quality agents, because the winners are decided based on the estimated execution times. Over time, the estimated reputations of their agents decrease until they are no longer allowed to bid and the good agents leave as well. After collaborating more times for providing a service, if the feedback from the client is favorable, and their reputations do not suffer much variation, the agents remain together for the following epochs. If the overall reputation of a holon is over a given value, $\rho_{cohesion}$, its composing agents refuse to split. Also, if a group of agents collaborate and receive positive feedback from the client, after a few epochs they unite to form a new holon within the holon they were members of.

ACKNOWLEDGMENT

This work was supported by CNCSIS-UEFISCSU, project number PNII-IDEI 316/2008, *Behavioural Patterns Library for Intelligent Agents Used in Engineering and Management*.

REFERENCES

[1] Fischer, K., Schillo, M., and Siekmann, J.: Holonic Multiagent Systems: A Foundation for the Organisation of Multiagent Systems, Proceedings of the First International Conference on Applications of Holonic and Multiagent Systems, HoloMAS'03, LNAI vol. 2744, pp. 71-80, Springer Verlag, 2003.

[2] Koestler, A.: The Ghost in the Machine, Hutchinson & Co, London, 1967.

[3] Fischer, K.: Holonic Multiagent Systems – Theory and Applications, Proceedings of the 9th Portuguese Conference on Artificial Intelligence: Progress in Artificial Intelligence, EPIA'99, LNCS vol. 1695, pp. 850-864, Springer Verlag, 1999.

[4] Adam, E., Mandiau, R., and Kolski, C.: Application of a Holonic Multi-Agent System for Cooperative Work to Administrative Processes, Journal of Applied Systems Studies, no. 2, pp. 100-115, 2001.

[5] Pěchouček, M. and Mařík, V.: Industrial deployment of multi-agent technologies: review and selected case studies, Autonomous Agents and Multi-Agent Systems, vol. 17, no. 3, pp. 397-431, 2008.

[6] Bürckert, H.J., Fischer, K., and Vierke, G.: TeleTruck: A Holonic Fleet Management System, Proceedings of the 14th European Meeting on Cybernetics and Systems Research, vol. 2, pp. 695-700, 1998.

[7] Lind, J., Fischer, K., Böcker, J., and Zirkler, B.: Transportation scheduling and simulation in a railroad scenario: A multi-agent approach, Proceedings of the 4th International Conference on The Practical Application of Intelligent Agents and Multi-Agent Technology, pp. 325-344, London, UK, 1999.

[8] Leitão, P., and Restivo, F.: A holonic approach to dynamic manufacturing scheduling, Robotics and Computer-Integrated Manufacturing, no. 24, pp. 625-634, 2008.

[9] Dijkstra, E.W.: A note on two problems in connexion with graphs, Numerische Mathematik, no. 1, pp. 269-271, 1959.

[10] Zacharia, G.: Collaborative Reputation Mechanisms for Online Communities, Master's thesis, Massachusetts Institute of Technology, 1999.

[11] Schillo, M., Funk, P., and Rovatsos, M.: Using Trust for Detecting Deceitful Agents in Artificial Societites, Applied Artificial Intelligence, vol. 14, no. 8, pp. 825-848, 2000.

[12] Esfandiari, B. and Chandrasekharan, S.: On How Agents Make friends: Mechanisms for Trust Acquisition, Proceedings of the Fourth Workshop on Deception, Fraud and Trust in Agent Societies, Montreal, Canada, pp. 27-34, 2001.

[13] Yu, B. and Singh, M.P.: Distributed Reputation Management for Electronic Commerce, Computational Intelligence, vol. 18, no. 4, pp. 535-549, 2002.

[14] Sen, S. and Sajja, N.: Robustness of Reputation-based Trust: Booblean Case, Proceedings of the first international joint conference on autonomous agents and multiagent systems (AAMAS-02), Bologna, Italy, pp. 288-293, 2002.