Towards a framework for agent coordination and reorganization, AgentCoRe

Ghijsen, M.; Jansweijer, W.N.H.; Wielinga, B.J.

Published in:
Lecture Notes in Computer Science

DOI:
10.1007/978-3-540-79003-7_1

Link to publication

Citation for published version (APA):
Ghijsen, M., Jansweijer, W., & Wielinga, B. (2008). Towards a framework for agent coordination and reorganization, AgentCoRe. Lecture Notes in Computer Science, 4870, 1-14. https://doi.org/10.1007/978-3-540-79003-7_1

Towards a Framework for Agent Coordination and Reorganization, AgentCoRe [⋆]

Mattijs Ghijsen, Wouter Jansweijer, and Bob Wielinga

Human Computer Studies Laboratory, Institute of Informatics, University of Amsterdam, email:{mattijs, jansw, wielinga}@science.uva.nl

Abstract. Research in the area of Multi-Agent System (MAS) organization has shown that the ability for a MAS to adapt its organizational structure can be beneficial when coping with dynamics and uncertainty in the MASs environment. Different types of reorganization exist, such as changing relations and interaction patterns between agents, changing agent roles and changing the coordination style in the MAS. In this paper we propose a framework for agent **Co**ordination and **Re**organization (AgentCoRe) that incorporates each of these aspects of reorganization. We describe both declarative and procedural knowledge an agent uses to decompose and assign tasks, and to reorganize. The RoboCupRescue simulation environment is used to demonstrate how AgentCoRe is used to build a MAS that is capable of reorganizing itself by changing relations, interaction patterns and agent roles.

1 Introduction

The quality of organizational design of a MAS has a large influence on its performance. However, this is not the only factor that determines MAS performance. It is the combination of the organizational design together with the nature of the task performed by the MAS and the characteristics of the environment in which the MAS is embedded that determines the performance of a MAS [1]. A MAS that operates in a dynamic environment can mitigate or reduce negative effects of dynamics in this environment by changing its organization [2].

In this paper we present the architecture of a framework that enables agents in a MAS to coordinate and reorganize. The goal of such an architecture is not to improve existing work on coordination by providing more efficient task/goal decomposition or task allocation, but rather to integrate several different aspects of reorganization into a single framework. To ensure a generic design, we describe our framework at the knowledge level [3], by providing the knowledge items and inferences an agent requires to coordinate and reorganize.

The organizational design of a MAS involves many aspects such as authority relations between agents, interaction patterns, agent roles and coordination

[⋆] The research reported in this paper is part of the Interactive Collaborative Information Systems (ICIS) project, supported by the Dutch Ministry of Economic Affairs, grant BSIK03024.

style. Research by Mintzberg has shown that the structure of human organizations and the coordination mechanisms used by its managers are closely related [4]. This, and other notions from the field of organizational design have already been applied in the area of MAS design [5, 6]. Since organization design and coordination are so closely related we believe that a framework for agent coordination should also provide the ability to reorganize.

Before we describe the AgentCoRe framework we discuss theory and related work on MAS reorganization. After a description of AgentCoRe, we will show how AgentCoRe is used to design and implement a MAS in the RoboCupRescue simulator [7]. We end with discussion, conclusions and directions for future work.

2 Theory and Related Work

In this paper we use definitions based on [8] and [5]. A task is defined as an activity performed by one or more agents to achieve a certain effect or goal in the environment. A task can be decomposed into subtasks and, in the case a task cannot be decomposed any further, it is called a primitive task. We define a role as a set of tasks that an agent is committed to perform when it is enacting that role. Capabilities are defined as a set of roles the agent is capable of enacting.

We define a MAS organization as a group of distributed agents, pursuing a common goal. The design of a MAS organization consists of relationships and interactions between the agents [9], agent roles [5] and coordination style [10]. Thus we define reorganization of a MAS as changing one or more of these organizational aspects. We assume that reorganization is triggered by the agents of the MAS, and not by a system designer or "human in the loop" as in [11].

A generic definition of coordination is given by [12] who define coordination as managing dependencies between activities. Research in the area of coordination in MAS has resulted in frameworks such as GPGP/TÆMS [13] and COM-MTDP [14] and both have been used in research on reorganization.

Nair et al. [15] extend [14] and change the composition of teams of agents to perform a rescue task in a highly dynamic environment where tasks can (de)escalate in size and new tasks are formed. Horling and Lesser change the relations and interaction patterns between agents in TÆMS structures but do not allow for role changes [16]. Their work is recently being extended by Kamboj and Decker [17] who use agent cloning [18] to allow for role changes in the organization. Barber and Martin present dynamic adaptation of coordination mechanisms as a mechanism for dealing with a dynamic environment [19].

The approaches described above all involve different aspects of reorganization; changing relations and interactions, changing agent roles and changing coordination style. In the next section we present the AgentCoRe framework which gives a knowledge level description of a framework for coordination and reorganization. We extend existing work by incorporating all aspects of reorganization described above into a single framework. By giving a knowledge level description, we refrain from computational details and focus on the required knowledge and reasoning for combined coordination and reorganization.

3 The AgentCoRe Framework

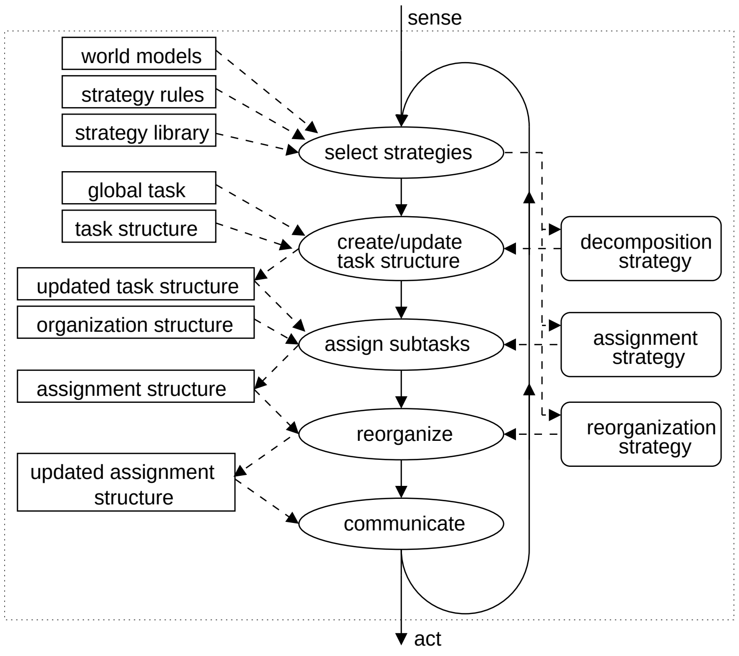


Fig. 1. AgentCoRe.

A global overview of the AgentCoRe framework is shown in figure 1. Oval shapes are sub-processes of the coordination process, rectangles depict declarative knowledge and rounded rectangles depict procedural knowledge. In a single iteration, an agent selects a coordination strategy, decomposes tasks, allocates tasks, reorganizes and communicates.

Strategy selection is based on the current state of the environment and strategy rules which prescribe the use of a coordination strategy in a certain situation. We see a coordination strategy as a combination of a task-decomposition strategy, an assignment strategy and a reorganization strategy. Each of these strategies are used as input for the sub-processes in the coordination process. Strategy selection is an important aspect of our approach because it enables an agent to use different coordination strategies during its lifetime.

The next step in the coordination process is to create and update the task structure. Based on sensory input – which can be observations by the agent and messages from other agents – the agent will decompose the global task into subtasks. The structure of decomposed tasks is called a task structure which is a simplified version of the goal trees in the TÆMS framework [13]. The decomposition strategy describes how the global task decomposes into subtasks.

Relations and interaction patterns between agents and agent roles in the MAS are described in the organization structure. In the task assignment sub-process, subtasks of the task structure are connected to the agents in the organization structure. The task structure combined with the organization structure by means of assignments is called the assignment structure. Which agents are assigned to which tasks is determined by the assignment strategy that is used.

When assignment is completed, the agent can reorganize the assignment structure (which contains the task structure as well as the organization structure). A reorganization strategy describes when and how reorganization takes place. Based on the final assignment structure the agent communicates changes in the organization and task structure to the agents affected by these changes.

3.1 Declarative ingredients

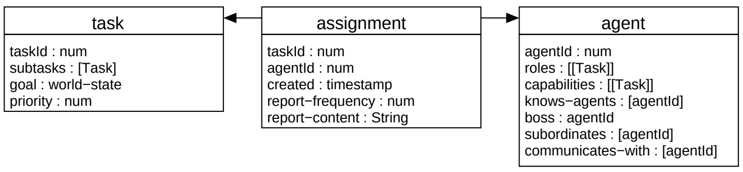


Fig. 2. Basic AgentCoRe declarative concepts.

The basic declarative components of the framework (see figure 2) are `task`, `agent` and `assignment`. A `task` has a set of subtasks and a description of the goal that is to be achieved by performing the task. Furthermore each `task` has a priority. Using the `task` concept, task structures can be created that show how tasks are decomposed into subtasks.

An `agent` has a set of roles the agent is currently enacting (in section 2 a role is defined as a set of tasks) and a set of capabilities which is the set of all roles the agent is capable of enacting. Furthermore, `agent` has relations with other agents. These relations describe which other agents the agent knows, communicates with, is boss of and which agent is its boss. Using the `agent` concept an organization structure can be created where agents have relations with each other, have roles and capabilities.

An `assignment` is a reified relation between `task` and `agent`. It has a timestamp that indicates when the assignment is created, a report frequency that defines when status reports on the progress of the task should be sent, and a specification of the content of the reports. The `assignment` concept connects task structures and organization structures to form assignment structures.

The decomposition, assignment and reorganization strategies are mostly domain specific procedural descriptions of how a specific task should be decomposed or what types of roles should be assigned when reorganizing. Examples of these strategies are given in section 4 of this paper.

3.2 Subprocess description

A knowledge level description of the internal structure of the subprocesses shown in figure 1 is given by using the CommonKADS notation of inference structures [20]. Rectangles depict dynamic information, ovals represent elementary reasoning processes and arrows indicate input-output dependencies. Two thick horizontal lines depict static information used as input for the reasoning processes. For clarity purposes we depict the starting point of the inference structure by a thick squared rectangle.

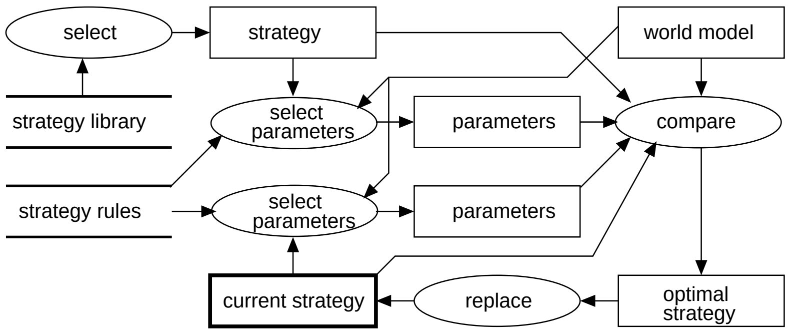


Fig. 3. Strategy selection inference structure.

Figure 3 shows the inference structure of strategy selection in which the current coordination strategy is compared to other available strategies in the strategy library. To obtain the most optimal strategy, parameters are used which represent selection criteria of a coordination strategy (e.g. "required time", "required resources" or "required capabilities"). Strategy rules define in which situation a coordination strategy is optimal by indicating which parameters should be used to compare the strategies. Examples of strategy rules are; "always use the cheapest strategy" and "use the cheapest strategy but when lives are at stake, use the fastest strategy". In the first case, the parameter that indicates cost will be selected. In the second case, the parameters for cost and required time will be selected. The value of a parameter is determined by the current state of the world.

Figure 4 shows the inference structure for task decomposition. First, one of the tasks is selected from the task structure and based on the current model of the world, it is determined whether the task is still valid; is the goal still a valid goal or has a report been received that the task is finished. In the case the task is not valid, the task structure is updated immediately. Otherwise, the task is decomposed as prescribed by the decomposition strategy. The task and the generated subtasks are then used to update the task structure. This continues until each task in the task structure has been validated and decomposed.

In figure 5 the task assignment inference structure (based on the assignment inference structure in [20]) is shown. The assignment strategy determines se-

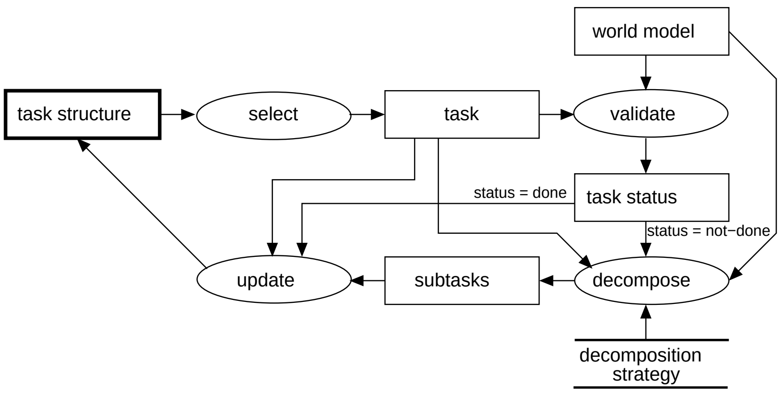


Fig. 4. Task decomposition inference structure.

lection of a set of tasks and a set of agents based on the current assignment structure. Grouping of tasks and agents can be used if multiple agents are assigned to a single task, or one agent is assigned to a group of tasks, or a group of agents is assigned to a group of tasks. If and how grouping is done, depends on the assignment strategy. If no grouping takes place the assign inference will use the task and agent sets that have been selected. The assign inference couples the sets or groups of tasks to the agents which results in a set of new assignments. This continues until all agents are assigned or no tasks are left to perform.

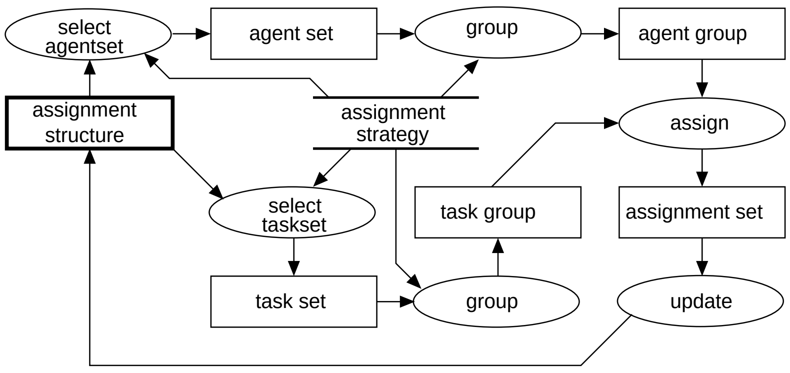


Fig. 5. Task assignment inference structure.

In the reorganize inference structure in figure 6, a reorganization strategy gives a set of available triggers. Triggers are rules that initiate change in organization. Some examples of triggers are detecting an unbalanced workload over the agents, sudden changes in priority of one of the unassigned subtasks while all available agents are already allocated to other tasks, or an event in the environment that requires two teams to work together. Triggers are tested on the assignment structure and if they fire, a set of matching change rules is selected and applied to the assignment structure. Possible change rules are to assign agents to different roles and creating and/or removing relations between agents, but also taking an agent away from the task it is currently performing and assigning it to a task with a higher priority. Applying change rules results in a partial new assignment structure which is used to update the current assignment structure. Trigger selection continues until all triggers have been tested on the assignment structure.

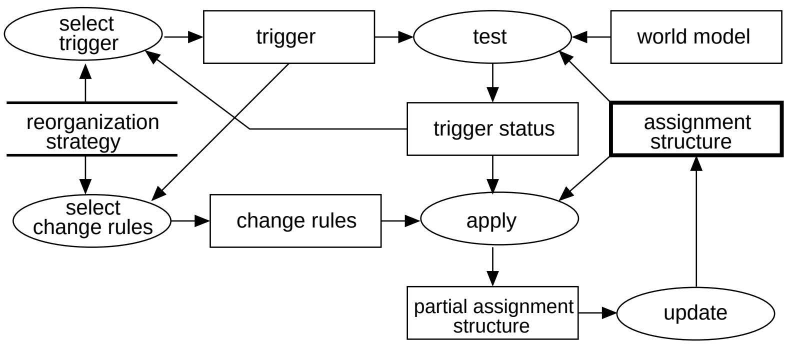


Fig. 6. Reorganize inference structure.

4 A MAS implementation using AgentCoRe

To demonstrate how AgentCoRe is used, the RoboCupRescue simulator [7] is used. In RoboCupRescue, agents are deployed that jointly perform a rescue operation. When the simulation starts, buildings collapse, civilians get injured and buried under the debris, buildings catch fire and fires spread to neighboring buildings. Debris of the collapsed buildings falls on the roads causing roads to be blocked. For this rescue operation, three main tasks can be distinguished and for each of these tasks a type of agent with appropriate capabilities is available. Fires are extinguished by fire brigade teams, blocked roads are cleared by police agents and injured civilians are rescued by ambulance teams.

For the purpose of demonstrating the use of AgentCoRe, we focus on the task of rescuing injured civilians. Thus we have build a MAS that consists of ambulance teams. The tasks for this MAS are the following:

- `SearchAndRescueAll` is the main task of searching the complete map and rescuing all injured civilians. This task has no additional attributes.
- `SearchAndRescueSector`, is the same task as the main task but is restricted to a single sector on the map (the map is divided into 9 sectors). The additional attribute for this task is a `sectorId`.

– `SearchBlock` is the task of searching all houses in a block for injured civilians. Blocks are small groups of houses of which there are 361 on the map. The additional attribute for this task a `blockId`.
– `RescueCivilian` is the task of rescuing a civilian. The additional attributes for this task are a `civilianId` and a `civilianLocation`.
– `CoordinateWork` is the task of coordinating (by means of the AgentCoRe framework) another task. The additional attributes for this task is a `taskId` of the task that is to be coordinated.

Based on the tasks described above, we have defined the following roles in the MAS organization:

– `AmbulanceRole`: [`SearchAndRescueSector`, `SearchBlock`, `RescueCivilian`]
– `GlobalManagerRole`: [`CoordinateWork`]
– `LocalManagerRole`: [`CoordinateWork`, `SearchAndRescueSector`]

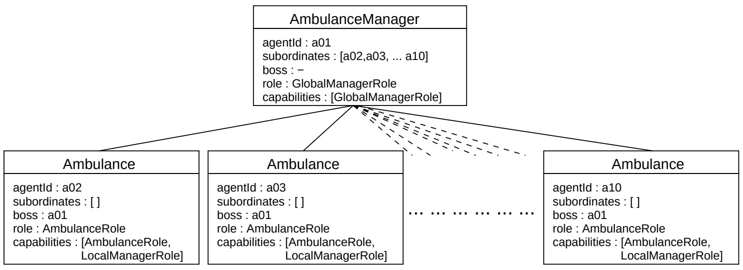
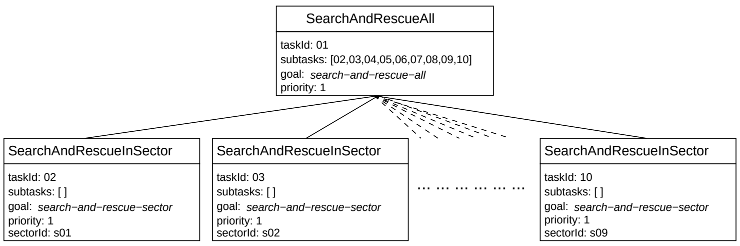


Fig. 7. Ambulance organization, initial structure

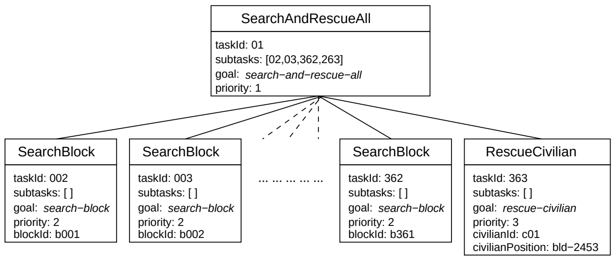
Based on these roles and tasks we have implemented an organization (see figure 7) that is controlled by an `AmbulanceManager` with a `GlobalManagerRole` who coordinates work on the `SearchAndRescueAll` task. The `AmbulanceManager` has 9 `Ambulance` agents at its disposal who are all capable of performing the `AmbulanceRole` and the `LocalManagerRole`. Intially all `Ambulance` agents will perform the `AmbulanceRole`. Because the `communicates-with` and `knows-agent` relations overlap with the authority relations only the authority relations are shown in figures 7 and 9. Authority relations also indicate how tasks are assigned and thus, the `AmbulanceManager` will assign tasks to its direct subordinates and it will receive status reports about the progress of these tasks on a regular basis. An `Ambulance` agent only performs the `LocalManagerRole` when ordered by its direct superior. An agent with the `LocalManagerRole` is able to assign tasks and order role changes to its direct subordinates.

4.1 Strategies

To be able to illustrate the use of different coordination strategies, we have implemented two task decomposition strategies. The first, named decomposition into



(a) decomposition into skills.



(b) decomposition into primitive tasks

Fig. 8. The result of different task decomposition strategies.

skills[1], decomposes the `SearchAndRescueAll` into 9 `SearchAndRescueSector` tasks which results in a task structure as in figure 8(a). The second strategy, named decomposition into primitive tasks, decomposes the `SearchAndRescueAll` into `SearchBlock` tasks for each housing block on the map. If any civilians are reported to be found during one of those `SearchBlock` tasks, a `RescueCivilian` task is generated which results in a task-structure as in figure 8(b). The priority of `RescueCivilian` tasks is based on a civilian's health status. As long as the injury of the civilian is not critical, the priority of the `RescueCivilian` task is lower than the `SearchBlock` tasks. As a civilian becomes more injured, the priority of the `RescueCivilian` task becomes larger than the `SearchBlock` tasks. By adjusting the priority of `RescueCivlian` tasks, the agents search the map as fast as possible but prevent civilians from dying.

For the assignment process we have implemented a strategy that selects tasks from the assignment structure that have not yet been assigned to an agent. From that subset the tasks with the highest priority are selected. The strategy also

[1] The name of this decomposition strategy is based on "coordination by standardization of skills" described by Mintzberg [4]. Coordination by standardization of skills can be characterized by assignment of large and complex tasks to the operator agents.

selects the agents from the assignment structure that are not assigned to a task. The strategy does not include grouping of agents or tasks.

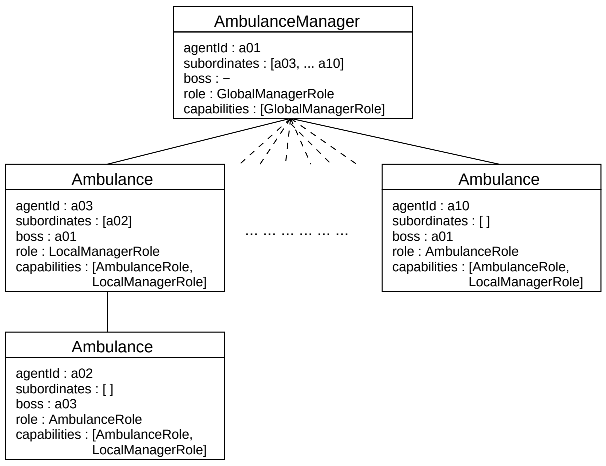


Fig. 9. Ambulance organization, after reorganization

A reorganization strategy has been implemented with one trigger and a set of change rules that are used when the trigger fires. The trigger fires if two conditions both hold; (1) there is at least one agent that has not been assigned to a task and (2) there is at least one task that is still being executed. The change rules specify that the `Ambulance` agent that is already working on that task has to switch from the `AmbulanceRole` to the `LocalManagerRole` (role change) and that the other agent will become a subordinate agent of the agent with the `LocalManagerRole` (structural change). The rationale behind this is that the first agent assigned to a task has acquired the most information on that task and is therefore most suited to coordinate work on this task when other agents are assigned to the same task.

5 Discussion and Conclusions

In the previous section we have described an implementation of a MAS in the RoboCupRescue environment using the AgentCoRe framework. By using a reorganization strategy, the MAS is capable of adapting agent relations and agent roles. Although AgentCoRe allows for switching between coordination mechanisms, the implemented MAS is not capable of changing its coordination style. However, a previous study in [21] has demonstrated the possibility of implementing different coordination mechanisms in an ambulance organization in the RoboCupRescue environment. The coordination mechanisms described in [21] have been composed out of the strategies as described in the previous section.

The design of the AgentCoRe framework enables an agent to use strategies for task-decomposition, task-allocation and reorganization. By using these strategies as input in the inference structures we have been able to distinguish domain dependent strategies from the domain independent reasoning for task decomposition, task assignment and reorganization. By providing the agent with these strategies, the agent will be able to cope with dynamics in the environment [21]. However, it may be the case that the environment or the nature of its task changes in such a way that these strategies – that are designed to enable the agent to cope with these dynamics – are not effective anymore. In this case the agent has the possibility to change its coordination strategy by selecting different strategies for task decompostion, task assignment and reorganization, that are better suited to cope with the current situation.

6 Future Work

As mentioned, the current MAS implementation does not have the capability of adjusting its coordination strategy. We have already shown that AgentCoRe can be used to implement multiple coordination strategies and in future work we will implement strategy rules that allow the agent in a MAS to change its coordination strategy. Future work will also focus on other domains that are more dynamic in nature. Furthermore we will study the applicability of the AgentCoRe framework in these domains to get a better understanding for which types of problem domains AgentCoRe is suited or not.

As also recognized by Dignum et al. [22], different reasons for reorganization exist. Our future research will continue to focus on the questions of *when* a MAS should reorganize and if such a situation occurs, *how* the MAS should reorganize. The first question involves identifying appropriate triggers for strategy selection and reorganization. The second question involves the identification of appropriate change-operators on a MAS organization and determine how these change-operators should be used by the agents in a MAS to achieve a more optimal organization structure.

References

1. So, Y., Durfee, E.: Designing organizations for computational agents. (1998) 47–64
2. Carley, K.: Computational and mathematical organization theory: Perspectives and directions. Journal of Computational and Mathematical Organizational Theory (1995)
3. Newell, A.: The Knowledge Level. Artificial Intelligence **18**(1) (1982) 87–127
4. Mintzberg, H.: Structures in fives: designing effective organizations. Prentice Hall, Englewood Cliffs, N.J. (1993)
5. Zambonelli, F., Jennings, N.R., Wooldridge, M.: Organisational abstractions for the analysis and design of multi-agent systems. In: 1st International Workshop on Agent-Oriented Software Engineering at ICSE 2000. (2000)

6. van Aart, C., Wielinga, B., Schreiber, G.: Organizational Building Blocks for Design of Distributed Intelligent System. International Journal of Human-Computer Studies **61** (2004) 567–599
7. Kitano, H., Tadokoro, S., Noda, I., Matsubara, H., Takahashi, T., Shinjou, A., Shimada, S.: Robocup-rescue: Search and rescue for large scale disasters as a domain for multi-agent research. In: Proceedings of IEEE Conference on Man, Systems, and Cybernetics(SMC-99). (1999)
8. Uschold, M., King, M., Moralee, S., Zorgios, Y.: The enterprise ontology. The Knowledge Engineering Review (1998)
9. Carley, K., Gasser, L.: Computational organization theory. In Weiss, G., ed.: Multi-Agent Systems, A Modern Approach to Distributed Artificial Intelligence. MIT-press (1999) 299–330
10. Jennings, N.: Coordination Techniques for Distributed Artificial Intelligence. In: Foundations of Distributed Artificial Intelligence. Wiley (1996) 187–210
11. Tambe, M., Pynadath, D.V., Chauvat, N.: Building dynamic agent organizations in cyberspace. IEEE Internet Computing **4**(2) (2000) 65–73
12. Malone, T.W., Crowston, K.: The interdisciplinary study of coordination. ACM Computing Surveys **26**(1) (March 1994)
13. Lesser, V., Decker, K., Wagner, T., Carver, N., Garvey, A., Horling, B., Neiman, D., Podorozhny, R., Prasad, M.N., Raja, A., Vincent, R., Xuan, P., Zhang, X.Q.: Evolution of the GPGP/TAEMS domain-independent coordination framework. Autonomous Agents and Multi-Agent Systems **9** (2004) 87–143
14. Pynadath, D.V., Tambe, M.: Multiagent teamwork: Analyzing key teamwork theories and models. In: First Autonomous Agents and Multiagent Systems Conference (AAMAS). (2002)
15. Nair, R., Tambe, M., Marsella, S.: Team formation for reformation. In: Proceedings of the AAAI Spring Symposium on Intelligent Distributed and Embedded Systems. (2002)
16. Horling, B., Benyo, B., Lesser, V.: Using self-diagnosis to adapt organization structures. In: Proceedings of the 5th International Conference on Autonomous Agents, ACM Press (June 2001) 529–536
17. Kamboj, S., Decker, K.: Organizational self-design in semi-dynamic environments. In: 2007 IJCAI workshop on Agent Organizations: Models and Simulations (AOMS@IJCAI 07). (2007)
18. Shehory, O., Sycara, K., Chalasani, P., Jha, S.: Agent cloning: An approach to agent mobility and resource allocation. In: IEEE Communications
19. Barber, K., Martin, C.: Dynamic reorganization of decision-making groups. In: AGENTS '01: Proceedings of the fifth international conference on Autonomous agents, New York, NY, USA, ACM Press (2001) 513–520
20. Schreiber, G., Akkermans, H., Anjewierden, A., de Hoog, R., Shadbolt, N., van de Velde, W., Wielinga, B.: Knowledge Engineering and Management: The CommonKADS Methodology. The MIT Press (2000)
21. Ghijsen, M., Jansweijer, W., Wielinga, B.: The effect of task and environment factors on m.a.s. coordination and reorganization. In: Proceedings of the Sixth International Joint Conference on Autonomous Agents and Multi-Agent Systems, short paper, to appear. (2007)
22. Dignum, V., Dignum, F., Sonenberg, L.: Towards dynamic reorganization of agent societies. In: Proceedings of CEAS: Workshop on Coordination in Emergent Agent Societies at ECAI 2004. (September 2004) 22–27