Proceedings of The 2004 International Conference on Artificial Intelligence (IC-AI'04)
Las Vegas, Nevada, USA. June 21 - 24, 2004.

Integrating Robotic Sensor and Effector Capabilities with Multi-agent Organizations

Eric Matson, Scott DeLoach
Multi-Agent and Cooperative Robotics Lab
Department of Computing and Information Sciences
Kansas State University
Manhattan, KS 66506
matson,sdeloach@cis.ksu.edu

Abstract

Robots possess many effectors and sensors of various capability. It is often difficult, not only to integrate these numerous capabilities, but also to organize them to accomplish a set of goals or tasks. Even more difficult is how to reorganize on the condition that a sensor/effector is damaged or incapacitated during operation. In this paper, we show how a organization-based multi-agent system (OMAS) adapts to control the sensor and effectors within a robot. The OMAS can also be extended to model the sensor and effectors for a team of cooperative robots. The use of integrating sensors/effectors via a OMAS will produce a self-reorganizing, fault tolerant control architecture capable of allowing a robot, or a team of robots, to overcome obstacles and faults within dynamic domains where sensor and effector failure rates may be high. Another positive effect of an OMAS is simplification of system complexity where a non-trivial number of sensors and effectors exist.

Keywords: Multiagent Systems, Organization

1. Introduction

A common obstacle with robotics is the development of control programs that allow sensors and effectors to work in harmony to accomplish a wide range of tasks. As the demands of the environment become more complex and the number of goals increase, the sensors and effectors required to meet the goal and environmental demands will also grow. Growth in sensors and effectors will increase the interaction required to be managed by the robotic control software. A way of managing and organizing the growth of the robotic control is through a multiagent system extended with an organization theoretic model. In our research, we have developed an organization-based MAS (OMAS) model that can be applied to the problem of sensor and effector organization for a robot or teams of robots. We propose that the use of an OMAS will significantly reduce the complexity of sensor and effector interaction and provide a model to create an efficient self-reorganizing team.

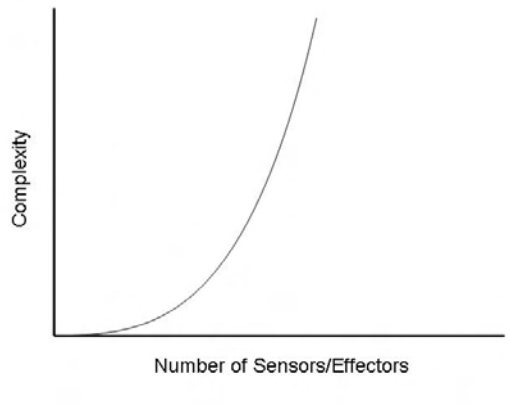

Figure 1. Complexity of Control

Complexity is a common problem in robotics research and implementation. The number of sensors and effectors for non-trivial systems can be quite large and, therefore, very complex. As related in Fig. 1, as the number of sensors and effectors increase, the complexity of the software control system will increase at an increasing rate. While this curve will be differentiated by the number, type and complexity of each individual type, it merely represents the general trend of com-

plexity. Reduction of the complexity, or simply managing and controlling it, is potentially gained using an organization-based multiagent system. An organization based model, using capability, enables a robot's sensors and effectors to be modularized, which allows effective relationships to be constructed between them. The integration of the organization model additionally allows for specialization of labor around roles that reduces the relationship complexity shortcoming of many multiagent peer-to-peer systems.

The goal of this research is to show the viability of applying organizational models and MAS for use in robotics. Specifically, we want to show that the resulting organization-based multiagent systems (OMAS) are a highly useful and functional alternative to traditional teamwork schemes and formalisms [1] [2]. Complimentary work in this area has proposed the use of networked robotics without a self-reorganizing multiagent concept [3]. Our research takes into consideration fault tolerant systems and architectures that deal with detecting and handling sensor failure and faults [4], and calibration of sensors to adapt to unknown environmental conditions [5]. Our model tolerates faults by managing the available hardware sensors as a cooperative, intentional system, focusing on managing their entire set of capabilities instead of simple "brute force" approach to sensor switching in cases of failure.

This paper defines our organization model in Section 2. Section 3 details the application of the OMAS model to a robotic instance with a non-trivial number of sensors and effectors. Section 4 provides conclusions for this model and Section 5 details future directions and plans.

2. Organization

To implement teams of autonomous, heterogeneous robots, we created an organizational model, which defines and constrains the required elements of a stable, adaptable and versatile team. While most people have an intuitive idea of what an organization is, there are no standard definitions. However, in most organizational research, organizations have typically been understood as including agents playing roles within a structure in order to satisfy a given set of goals. Our proposed organizational model (O) contains a structural model, a state model and a transition function. Fig. 2 shows the combined structural and state models using standard UML notation.

$$O =< O_{structure}, O_{state}, O_{transition} >$$

2.1. Structure

The structural model includes a set of goals (G) that the team is attempting to achieve, a set of roles (R) that must be played to attain those goals, a set of capabilities (C) required to play those roles, and a set of rules or laws (L) that constrain the organization. The model also contains static relations between roles and goals (achieves), roles and capabilities (requires), individual roles (related), goals and subgoals(subgoals) and a unary relation for conjunctivity between subgoals of a goal(conjunctive). Formally, we model the organization structure as a tuple:

$O_{structure} =< G, R, L, C, achieves, related, requires,$
$subgoal, conjunctive >$
$where:$
$achieves : R, G \rightarrow [0..1]$
$related : R, R \rightarrow Boolean$
$requires : R, C \rightarrow Boolean$
$subgoal : G, G \rightarrow Boolean$
$conjunctive : G \rightarrow Boolean$

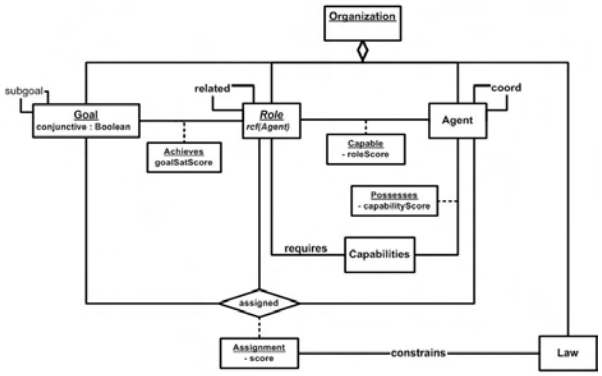

Figure 2. Organization Model

The team goals include the goal definitions, goal-subgoal decomposition, and the relationship between the goals and their subgoals, which are either conjunctive or disjunctive. Roles define parts or positions that an agent may play in the team organization. In general, roles may be played by zero, one, or many agents simultaneously while agents may also play many roles at the same time. Each role requires a set of capabilities, which are inherent to particular agents and may include sensor capabilities (sonar, laser, or video, etc.), actuator capabilities (movement type, grippers, etc.),

or computational capabilities (processing power, algorithms, communications, etc.). Robots are unique in the area of capabilities versus software agents; robot's physical capabilities may improve or degrade over time, which can often cause the team to reorganize. Organizational rules are used to constrain the assignment of agents to roles and goals within the organization. Generic rules such as "an agent may only play one role at a time" or "agents may only work on a single goal at a time" are common. However, rules are often application specific, such as requiring particular agents to play specific roles. The structural model relations define mappings between the structural model components described above. A role that can be used to satisfy a particular goal is said to achieve that goal, while a role requires specific capabilities and may work directly with other roles, thus being related to those roles. Achieves is modeled as a function to capture the relative ability of a particular role to satisfy a given goal.

2.2. State

The second element of the Organization Model is its state. The Organizational State (O_{state}) is an instance of the organizational structure at a point in time. As the Organization Model is a template, the state is an instance of the model. In an instance of an organization state, each of the elements will be bound to a set of values that represent the organization attributes. An organization will possess at least one goal, one role to accomplish the goal, and one agent to play the role where the agent will possess capabilities required by the role. Not every organization state element is required to be populated by an instance variable for creation of a valid organization. The constraints and laws of an organization will govern the requirements of a specific state.

The organizational state model defines an instance of a team's organization and includes a set of agents (A) and the actual relationships between the agents and the various structural model components.

$O_{state} =< A, possesses, capable, assigned, coord >$
$where:$
$possesses : A, C \rightarrow [0..1]$
$capable : A, R \rightarrow [0..1]$
$assigned : A, R, G \rightarrow [0..1]$
$coord : A, A \rightarrow Boolean$

An agent that possesses the required capabilities for a particular role is said to be capable of playing that role. Since not all agents are created equally, possesses is modeled as a real valued function, where 0 would represent absolutely no capability to play a role while a 1 indicates an excellent capability. In addition, since agent capabilities may degrade over time, this value may actually change during team operation. The capable function defines the ability of an agent to play a particular role and is computed based on the capabilities required to play that role. During the organization process, a specific agent is selected to play a particular role in order to satisfy a specific goal. This relationship is captured by the assigned function, which includes a real valued score that captures how well an agent, playing a specific role, can satisfy a given goal. When an agent is actually working directly with another agent, it is coordinating (coord) with that agent. Thus, the state model defines the current state of the team organization within the structure provided by the structural model.

2.3. Transition

The Organization Transition Function ($O_{transition}$) defines the ability of the organization to reorganize from an instance state to the next instance state over the organization life span. From the initial organization, through its termination, the organization may transition its state model numerous times.

The organization transition function defines how the organization may transition from one organizational state to another over the lifetime of the organization. Since the team members (agents) as well as their individual capabilities may change over time, this function cannot be predefined, but must be computed based on the current state, the goals that are still being pursued, and the organizational rules. In our present research with purely autonomous teams, we have only considered reorganization that involves the state of the organization. However, we have defined two distinct types or reorganization: state reorganization, which only allows the modification of the organization state, and structure reorganization, which allows modification of the organization structure (and may require state reorganization to keep the organization consistent). To define state reorganization, we simply need to impose the restriction that:

$$O_{trans(O)}.O_{structure} = O.O_{structure}$$

Technically, this restriction only allows changes to the set of agents, A, the coord relation, and the possesses, capable, and assigned functions. However, not all these components are actually under the control of the organization. For our purposes, we assume that agents may enter or leave organizations or relationships, but that these actions are triggers that cause

reorganizations and are not the result of reorganizations. Likewise, possesses (and thus capable as well) is an automatic calculation on the part of an agent that determines the roles that it can play in the organization. This calculation is totally under control of the agent (i.e. the agent may lie) and the organization can only use this information in deciding its organizational structure. Changes in an agent's capabilities may also trigger reorganization. That leaves the two elements that can be modified via state reorganization: *assigned* and *coord*. Thus, we define state reorganization as:

$$
\begin{aligned}
&O_{trans(state)} : O \rightarrow O \\
&where \\
&O_{trans(state)}(O).O_{struct} = O.O_{struct} \\
&O_{trans(state)}(O).O_{state}.A = O_{state}.A \\
&O_{trans(state)}(O).O_{state}.possesses = O_{state}.possesses \\
&O_{trans(state)}(O).O_{state}.capable = O_{state}.capable
\end{aligned}
$$

Organization and Reorganization Processes The initial step in organizing an multi agent or robotic team is to use the existing information production goals to establish the organizational roles required to produce the appropriate information. At the same time, the team of agents or robots making up the team must assess their individual and collective capabilities to determine whether they can fulfill the required roles [MD03]. If the required roles can be filled, then the capabilities exist to satisfy the information production goals and the team assigns the necessary roles to agents (effectively defining the state of the team's organization). Once the assignments are made, the team may initiate action to satisfy the team information production goals.

$$O_{state(0)} \rightarrow O_{state(1)}$$

The reorganization process follows the same basic steps as the organization process; however, it differs in the point of initiation. Reorganization is initiated by a trigger event, such as sensor loss, during the execution of an already existing organization. When such an event occurs, the team must determine if it still has the capabilities to satisfy team information production goals or whether it must reorganize to do so.

$$O_{state(0)} \rightarrow O_{state(n+1)}$$

Organizational Outcomes The outcomes of the organization and reorganization processes are equivalent. The three available outcomes are goal satisfaction, goal relaxation or goal abandonment. Goal satisfaction indicates that the capabilities exist within the remaining team to accomplish all critical and non-critical goals. Goal relaxation indicates that capabilities exist within the remaining team to meet all critical goals, but some or all non-critical goals may not be met and will have to be "relaxed". Goal abandonment indicates the remaining member's capabilities do not allow the organization to continue because not all critical goals can be satisfied and success is not achievable.

2.3.1. Capability Robots are defined by the physical and computational capabilities they possess. The robot's capabilities define the roles they can play in meeting a team goal. For robots, there are two levels of capabilities: computational and physical. The computational capabilities are defined by the level of intelligence built into the robot. The physical capabilities are defined by the range of sensors and effectors included as part of the robot's design. In this research, we will focus on modeling the function of the physical sensors with the capabilities of agents.

So far, we have used the term capability generically. However, we must define it more precisely to demonstrate how we utilize it to integrate robot sensors and effectors. A capability's existence is based on the collective sense in which it is viewed. To specify this, we further define capabilities in relation to agent and roles that exist within a self-reorganizing multiagent team. As described above, an agent possesses specific capabilities while roles require particular capabilities, each with specific scores.

The capability set of an agent, C_a, varies from the empty set, if the agent possesses no capability, to a complete set of the capabilities that the agent intrinsically possesses. Typically, although not always, even a simple agent has multiple capabilities.

$$C_a(a) = \{c \mid possesses(a, c) > 0\}$$

Likewise, the capability set of a role, C_r, is the set of capabilities required to play that specific role. The capability set formally describes what capabilities are required for agents potentially to enact and play the role [7]. All non-trivial roles must have at least one capability in order to accomplish some task or goal.

$$C_r(r) = \{c \mid requires(r, c)\}$$

An agent is capable of playing a role if $Cr(r) \subseteq Ca(a)$. How well agent a can play role r is determined by the *role capability function (rcf)* that is part of each role definition. The *rcf* is part of the role and defines a role-specific computation based on the capabilities possessed by an agent. If an agent does not possess one of

the required capabilities, then the agent has no capacity to play that role and $r.rcf(a) = 0$. Thus, the capability score of an agent playing a particular role is defined by:

$$capable(a, r) = r.rcf(a)$$

During the organization process, a specific agent is selected to play a particular role in order to satisfy a specific goal. This relationship is captured by the assigned function, which includes a real valued score that captures how well an agent, playing a specific role, can satisfy a given goal. This score is computed by

$$assigned(a, r, g).score = capable(a, r) * achieves(g, r)$$

When an agent is actually working directly with another agent, it is coordinating *(coord)* with that agent. Thus, the state model defines the current state of the team organization within the structure provided by the structural model.

The agents that form a team have a collective capability. Similarly, the set of roles required to achieve the overall organizational goal also have a set of required capabilities. We define these as team capabilities, C_A, and required capabilities, C_R.

$$C_A(O) = \{c \mid \exists a : A \bullet c \in C_a(a)\}$$

$$C_R(O) = \{c \mid \exists r : R \bullet c \in C_r(r)\}$$

To form a viable organization, these sets must be minimally overlapping such that the capabilities required are contained in the capabilities available from the agents, with respect to the organization, such that:

$$C_R(O) \subseteq C_A(O)$$

3. Robotic Implementation

In this section, we will use a robotic instance to demonstrate how the organization-based concept and how formalizations can be used to integrate sensors and effectors.

An important measure of a robot is its physical abilities, each with a specific set of capabilities to play a role within an organization. Whereas a robot is defined by the combination of its computational and physical characteristics and capabilities, we will focus on the physical attributes; sensors and effectors. Each sensor and effector binds to a capability and the capabilities tie to an agent contained within an organization. Our demonstration robot is a Nomad Scout [8] as shown in Fig. 3. The effectors of the Nomad Scout are trivial, but the use of the sensors, as shown in Figure 4, are sufficient to capture and apply to organization theoretic concept.

Figure 3. Nomad Scout

An integral concept in the development of an organization, to satisfy a set of goals, is the specialization of labor. In our case, the specialization is captured by the roles defined to accomplish the given goals. Specialization will allow the organization to avoid complex individual coordination but it will call for coordination between individuals[10]. Specialization, using roles, limits the interactivity to only the agent peers that must communicate by definition. The organization model formalizes this by the coordination between agents $coord : A, A \rightarrow Boolean$. The reduction of coordination, between all individuals, will allow us to reduce the increase in complexity as the number of individual sensors grow. In the case of the Nomad Scout robot, we no longer need to work with 22 different sensors, we will now organize around two roles that require the capability to manage a sonar or a tactile bump, respectively.

The Nomad Scout robot has sonar and tactile bump sensors. Collectively, there are 22 sensors consisting of 16 sonar sensors and 6 tactile bump sensors. The sonar ring is a Sensus 200 consisting of 16 Polaroid 6500 sonar ranging modules fixed in 22.5° increments in a full 360° configuration. The Polaroid 6500 module can accurately measure distances from 6 inches to 35 feet,±1%. There are six bump sensors configured on the front and rear arcs of the robot. The bump sensors are tactile, so physical contact must be made with a physical object to be enacted. The bump sensors, unlike the sonar, do not provide a 360° range of detection, as shown by the graphical comparison of the sonar and bump sensor configurations.

Example: Environment Scanning In this example, we show how the organization model can be used in a sim-

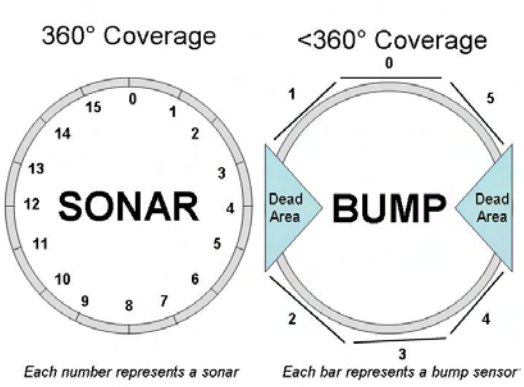

Figure 4. Nomad Sensors

ple scenario, stationary environmental scanning, with a single Nomad Scout robot. To create the organization for integrating the Nomad's sensor and effector capabilities, goals, roles and agents must be created that capture the essence of the specified task. For integration of sensor capabilities, the structural model must be fully developed. The organization structural element $O_{structure}$ is captured by:

$Organization : Scan\ the\ environment$
$Goal = \{scan_{22.5^\circ}, bump, integrate, manage, scan_{360^\circ}\}$
$Role = \{scanner_{sonar}, scanner_{bump}, integrator, manager\}$
$Law = \{\}$
$Capability = \{detect_{22.5^\circ}, detect_{touch}, integrate_{sonar}, integrate_{bump}, manage_{integrator}\}$
$The\ achieves\ relation :$
$achieves(scanner_{sonar}, scan_{22.5^\circ}) \rightarrow 1$
$achieves(scanner_{bump}, bump) \rightarrow 1$
$achieves(integrator, integrate) \rightarrow 1$
$achieves(manager, manage) \rightarrow 1$
$The\ related\ relation :$
$related(scanner_{sonar}, integrator) \rightarrow true$
$related(scanner_{bump}, integrator) \rightarrow true$
$related(integrator, manager) \rightarrow true$
$The\ role\ capabilities\ required\ are\ as\ follows :$
$C_r(scanner_{sonar}) = \{detect_{22.5^\circ}\}$
$C_r(scanner_{bump}) = \{detect_{touch}\}$
$C_r(integrator) = \{integrate_{sonar}, integrate_{bump}\}$
$C_r(manager) = \{manage_{integrator}\}$
$The\ subgoal\ relation :$
$subgoal(scan_{22.5^\circ}, scan_{360^\circ}) \rightarrow true$
$subgoal(bump, scan_{360^\circ}) \rightarrow true$
$subgoal(integrate, scan_{360^\circ}) \rightarrow true$
$subgoal(manage, scan_{360^\circ}) \rightarrow true$
$The\ conjunctive\ relation :$
$conjunctive(scan_{360^\circ}) \rightarrow true$

To complete the organization the O_{state} will need to be defined for this scenario. The *state* is defined by O_{state} where the following relationships form:

$The\ agent\ definitions\ for\ this\ scenario\ are :$
$A = \{agent_1, agent_2 \ldots agent_{25}\}$
$The\ capabilities\ possessed\ by\ each\ agent\ are :$
$C_a(agent_1) \ldots C_a(agent_{16}) = \{detect_{22.5^\circ}\}$
$C_a(agent_{17}) \ldots C_a(agent_{22}) = \{detect_{touch}\}$
$C_a(agent_{23}) = \{integrate_{sonar}\}$
$C_a(agent_{24}) = \{integrate_{bump}\}$
$C_a(agent_{25}) = \{manage_{integrator}\}$
$The\ capabilities\ for\ an\ agent\ to\ play\ a\ role\ are :$
$capable(agent_1, scanner_{sonar}) \ldots capable(agent_{16}, scanner_{sonar}) \rightarrow 1$
$capable(agent_{17}, scanner_{sonar}) \ldots capable(agent_{25}, scanner_{sonar}) \rightarrow 0$
$capable(agent_1, scanner_{bump}) \ldots capable(agent_{16}, scanner_{bump}) \rightarrow 0$
$capable(agent_{17}, scanner_{bump}) \ldots capable(agent_{22}, scanner_{bump}) \rightarrow 1$
$capable(agent_{23}, scanner_{bump}) \rightarrow 0$
$capable(agent_{24}, scanner_{bump}) \rightarrow 0$
$capable(agent_{25}, scanner_{bump}) \rightarrow 0$
$capable(agent_1, integrator) \ldots capable(agent_{22}, integrator) \rightarrow 0$
$capable(agent_{23}, integrator) \rightarrow 1$
$capable(agent_{24}, integrator) \rightarrow 1$
$capable(agent_{25}, integrator) \rightarrow 0$
$capable(agent_1, manager) \ldots capable(agent_{24}, manager) \rightarrow 0$
$capable(agent_{25}, manager) \rightarrow 1$

$The\ coordinating\ agents\ are :$
$coord(agent_1, agent_{23}) \ldots coord(agent_{16}, agent_{23}) \rightarrow true$
$coord(agent_{17}, agent_{24}) \ldots coord(agent_{22}, agent_{24}) \rightarrow true$
$coord(agent_{23}, agent_{25}) \rightarrow true$
$coord(agent_{24}, agent_{25}) \rightarrow true$

The assignment of agents, roles and goals completes the set of relations required to form an organization.

In this case, there are singular agents capable of playing each role and thus satisfying each of the main goals conjunctive subgoals. While there are trivial assignment alternatives for this organization, such as assigning $agent_2$ to a different sonar unit, overall the resulting organization instance $O_{state(1)}$ or alternatively, $O_{scan_{22.5^\circ}}$ is by default optimal. If there are assignment alternatives, the organization computation score $assigned(a, r, g).score$ will be employed to determine the optimal organization instance based on the capability of an agent a to play a role r, where the role r satisfies some goal g.

$$
\begin{aligned}
&\textit{The assignments are}: \\
&\quad assigned(agent_1, scanner_{sonar}, scan_{22.5^\circ}) \ldots \\
&\quad assigned(agent_{16}, scanner_{sonar}, scan_{22.5^\circ}) \rightarrow 1 \\
&\quad assigned(agent_{17}, scanner_{bump}, bump) \ldots \\
&\quad assigned(agent_{22}, scanner_{bump}, bump) \rightarrow 1 \\
&\quad assigned(agent_{23}, integrator) \rightarrow 1 \\
&\quad assigned(agent_{24}, integrator) \rightarrow 1 \\
&\quad assigned(agent_{25}, manager) \rightarrow 1
\end{aligned}
$$

In this scenario the capability constraint is satisfied as $C_R(O) \subseteq C_A(O)$ is satisfied. All C_R are possessed by C_A:

$$
\begin{aligned}
C_R(O) = \{&detect_{22.5^\circ}, detect_{touch}, integrate_{sonar}, \\
&integrate_{sonar}, manage_{integrator}\} \\
C_A(O) = \{&detect_{22.5^\circ}, detect_{touch}, integrate_{sonar}, \\
&integrate_{sonar}, manage_{integrator}, coordinate\}
\end{aligned}
$$

Because the constraints required to form a valid organization have been satisfied, the initial organization will be instantiated with the given assignments based on the goals, roles and agents.

4. Conclusions

In this paper, we show how our organization theoretic model can be applied to the problem of sensor and effector integration. By using an organization model, intrinsic capabilities of sensors and effectors can be enabled to cooperatively work together. A second important feature allows growth in the number of sensors and effectors without exponential growth in the complexity to control the sensors.

Advantages There are many advantages in using organization models. Application to many domains, the ability to limit or reduce complexity, reduction of overall necessary agent interaction, assumed intent, cooperation, fault tolerance and recovery are just a few of the advantages. Our model is generic, and purposely designed with the intent of making it applicable in a number of problem domains. With the ability to specialize labor, the complexity can be reduced by the nature of simple planning and interaction in relation to roles that will satisfy the system goals. Reducing complexity and specializing roles will reduce overall interaction between agents in comparison with strict peer point-to-point agent implementations. This complexity reduction will allow us to construct larger robotic teams, with less development effort, than is currently possible. Because the organization is created around a specific set of goals, the system becomes intentional by nature, and cooperative by design. Our organizational model allows the transition to a new organization if there is a failure by a agent working within the team. Reorganizing also can be triggered by a suboptimal state occurring or developing in the current organization. Self reorganization allows for the implemented systems to be fault tolerant and recoverable.

Limitations Even though the organization model is designed generically, with many domains considered, it will have to be applied to the specific domain problem effectively. If the model is not applied thoughtfully and correctly, the expected results may be less than the effective level desired. We are continuing to further refine the model to capture all domains where organization is needed or can be applied. Tools that guide the planner through the process of developing, asking the valid questions and capturing all necessary organization information will reduce these issues significantly.

Possible Applications Because of the generic design, our model is enabled for use in a wide range of applications. As of the time of submission, we have used the model in the Adaptive Information Systems (AIS) [12], robotics, and agent-oriented software engineering domains. As we have shown here, there are many goal-based robotic applications where our organization model can be applied using an OMAS approach.

5. Future Work

This work is part of a larger effort to more fully define the usefulness of an organizational theoretic approach to building a multi-agent system. In the near future, we plan to add new sensor types and thus assign more, different types of agent capabilities. This will allow us to more fully evaluate the scalability of the organizational model and the effectiveness of our organizational reasoning techniques. We will also integrate the organization model into the Multiagent Software Engineering (MaSE) tool agentTool[13]. This will assist in the construction of valid and useful organiza-

tion based application while reducing the problems in designing non-trivial organizations from scratch.

6. Acknowledgments

This work is sponsored by the Air Force Office of Scientific Research (AFOSR) under grant number F49620-02-1-0427.

References

[1] Tambe, M. and Zhang, W., Towards Flexible Teamwork in Persistent Teams. Second International Conference on Multi-Agent Systems, 1996.

[2] Far, B.H., Hajji, H., Saniepour, S., Soueina, S.O., Elkhouly, M.M., Formalization of Organizational Intelligence for Multiagent System Design. IEICE Transactions on Information and Systems. Volume E83-D, No. 4, April 2000.

[3] McKee, G., Schenker, P., Baker, D. Networked Robotics Concepts for Space Robotics Systems, Proceedings of the Robosphere 2002 Workshop, NASA Ames Research Center, Moffett Field, California, November 14-15, 2002.

[4] Roumeliotis, S., Sukhatme, G., Bekey, G., Sensor Fault Detection and Identification in a Mobile Robot, Proceedings of 1998 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS '98), Victoria, Canada, Oct. 13-17, 1998, pp.1383-1388.

[5] Soika, M. Grid Based Fault Detection and Calibration of Sensors on Mobile Robots. Proceedings of the 1997 IEEE International Conference of Robotics and Automation (ICRA '97). Albuquerque, New Mexico. April 1997.

[6] Matson, E., DeLoach, S. Organizational Model for Cooperative and Sustaining Robotic Ecologies, Proceedings of the Robosphere 2002 Workshop, NASA Ames Research Center, Moffett Field, California, November 14-15, 2002.

[7] Dastani, M., Dignum, V., Dignum, F. Role Assignment in Open Agent Societies. Proceedings of the Second Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS '03), Melbourne, Australia, July 14-18, 2003.

[8] Nomadic Technologies, Inc. July 12, 1999. Nomad Scout User's Manual, Mountain View, CA

[9] Matson, E., DeLoach, S. Using Dynamic Capability Evaluation to Organize a Team of Cooperative, Autonomous Robots, Proceedings of the 2003 International Conference on Artificial Intelligence (IC-AI '03), Las Vegas, Nevada, June 23-26, 2003.

[10] Houthakker, Henrik, S., Economics and Biology: Specialization and Speciation, Kyklos, 1956, 9,(2), 181-187.

[11] DeLoach, S., Matson, E., Li, Y. Exploiting Agent Oriented Software Engineering in the Design of a Cooperative Robotics Search and Rescue System. The International Journal of Pattern Recognition and Artificial Intelligence,17 (5), pp. 817-835, August 2003.

[12] Matson, E., DeLoach, S. Organization-based Adaptive Information Systems for Battlefield Situational Analysis, Proceedings of the IEEE International Conference on Integration of Knowledge Intensive Multi-Agent Systems (IEEE KIMAS '03), Boston, MA, October 1-3, 2003.

[13] DeLoach, S., Wood, M. Developing Multiagent Systems with agentTool. in Intelligent Agents VII. Agent Theories Architectures and Languages, 7th International Workshop (ATAL 2000, Boston, MA, USA, July 7-9, 2000), C. Castelfranchi, Y. Lesperance (Eds.). Lecture Notes in Computer Science. Vol. 1986, Springer Verlag, Berlin, 2001.