PAPER • **OPEN ACCESS**

Modelling a Swarm of Delivery Drones for Disaster Relief Utilizing an Organization Approach

To cite this article: Valeriy Kolesnikov 2022 *J. Phys.: Conf. Ser.* **2224** 012115

View the article online for updates and enhancements.

You may also like

- Challenges in Scientific Data Communication from Low-mass Interstellar Probes
David G. Messerschmitt, Philip Lubin and Ian Morrison
- A Robust Backstepping Sliding Mode Controller with Chattering-Free Strategy for a Swarm of Drones
Adeel Zaidi, Muhammad Kazim and Hui Wang
- From honeybees to robots and back: division of labour based on partitioning social inhibition
Payam Zahadat, Sibylle Hahshold, Ronald Thenius et al.

ECS Membership = Connection

ECS membership connects you to the electrochemical community:

- Facilitate your research and discovery through ECS meetings which convene scientists from around the world;
- Access professional support through your lifetime career:
- Open up mentorship opportunities across the stages of your career;
- Build relationships that nurture partnership, teamwork—and success!

Join ECS! Visit electrochem.org/join

ECS

This content was downloaded from IP address 193.34.92.189 on 03/06/2022 at 12:03

Modelling a Swarm of Delivery Drones for Disaster Relief Utilizing an Organization Approach

Valeriy Kolesnikov

Computer Science Department, Sumy State University, 2 Rymskogo-Korsakova St., Sumy, 40007, Ukraine

Email: v.kolesnikov@cs.sumdu.edu.ua

Abstract. This paper describes an organization-based approach to modelling an autonomous distributed system. This approach simplifies design of such systems and allows for a better understanding of the system overall. Such an approach is applicable for modelling a variety of distributed systems. We demonstrated it for a swarm system of autonomous drones used for disaster relief aid delivery.

1. Introduction

New advances in technology allow using autonomous drones to work as a team to achieve a specific goal. Such teams are often called swarms. Each drone in a swarm is autonomous, but drones can communicate among themselves in a distributed fashion to get to a specific goal as a team.

A study of autonomous systems like swarms of AI-enabled autonomous drones has become important not only for military applications but also, and increasingly so, for civilian use, such as humanitarian aid and disaster relief, search and rescue and search and track of a target with minimal outside direction, to name a few. Smart-swarm autonomous systems should be self-directing, with reliable communication among its parts, resilient to various interferences and self-adjusting in the face of failures of some of its parts.

It is a difficult task to develop autonomous distributed systems that are robust, flexible and fault tolerant. This paper describes how a design of such systems can be done using an organization model approach for multi-agent systems. The design stage is the first step in developing autonomous distributed systems and is a part of a bigger ecosystem for design, development, verification, validation and deployment of such systems that the author is working toward.

The approach to design autonomous distributed systems using organization model presented here is based on an organization as a social construct that provides a framework for defining goals that the organization tries to achieve, entities to achieve these goals and relationships among the entities. More specifically, we draw from the Multiagent Systems Engineering methodology (MaSE) [1,2] to derive goals from the system description, and then design roles, capabilities and agents to achieve the organization goals.

We present a concrete example utilizing this approach for a swarm of autonomous drones for disaster relief aid delivery and discuss the place of this approach in the framework for development of autonomous distributed systems.

The remainder of this paper is organized as following: first, we discuss related work to designing autonomous distributed systems; then, we present an organization model that we utilize to model autonomous distributed systems; we follow with an example, in which we design a model for a swarm

Content from this work may be used under the terms of the Creative Commons Attribution 3.0 licence. Any further distribution of this work must maintain attribution to the author(s) and the title of the work, journal citation and DOI.
Published under licence by IOP Publishing Ltd

of autonomous drones for disaster relief aid delivery and present on organization instance for this example; finally, we end with conclusion and directions of our future work.

2. Related Work

Research into designing and implementing autonomous distributed systems has taken many directions. For us, an organization-based approach is of interest for modelling such systems as well as the development of frameworks for the ease of developing, testing and deploying autonomous distributed systems.

Research into an organization-based approach to modelling distributed systems shows that: 1) organizational design is easier to understand as it is close to human organizations [3], 2) organization-based systems tend to be efficient and can efficiently adapt to changing conditions [3,4], 3) organizations are flexible [3,4], and 4) best designs utilizing organization-based approach are application and situation dependent [5].

Organizations can be centralized or decentralized. In a distributed scenario, decentralized organizations tend to exhibit higher performance and reliability [1].

Several attempts have been done at creating and formalizing systems or frameworks to capture the concept of teamwork within an organization [6-11]. Although valuable for design of autonomous distributed systems, they fail to provide a wide accepted use.

In our previous work we proposed InDiGO, an infrastructure for development of distributed systems [12]. InDiGO allows design of generic but customizable algorithms and provides tools to customize such algorithms for distributed applications [12,13]. We demonstrated advantages of using such a framework for distributed systems development [12,14] and plan to merge this research into the InDiGO framework.

3. Organization Model

3.1. Model Description

To ease the design of autonomous systems, we propose an organization model (O). This model describes the structure of the organization and combines all entities of the autonomous system and the relationships among them. Figure 1 shows organization model using standard UML notations.

Organization model O contains a set of goals (G) that the system is trying to achieve, a set of roles (R) that achieve the stated goals, a set of agents (AG) that are capable of playing specific roles, a set of capabilities (C) that agents possess and that are required by the roles to achieve the goals, a set of assignments (A) that the organization performs to assign agents to play specific roles to achieve specific goals and a set of policies (P) that constrain the assignments.

The organization model also describes relationships that exist among entities of the autonomous system. Among them are the following relationships: achieves relationship that exists between roles and goals, capable relationship that exists between agents and roles, possesses relationship that exists between agents and capabilities, required relationship that exists between capabilities and roles, constrains relationship that exists between policies and assignments and assigned relationship that exists between agents, roles and goals.

Formally, organization model O is a tuple

$$O = < G, R, AG, C, A, P, achieves, capable, possesses, required, constrains, assigned > \quad (1)$$

where

$$achieves: R \times G \rightarrow Boolean \quad (2)$$

$$capable: AG \times R \rightarrow Boolean \quad (3)$$

$$possesses: AG \times C \rightarrow Boolean \quad (4)$$

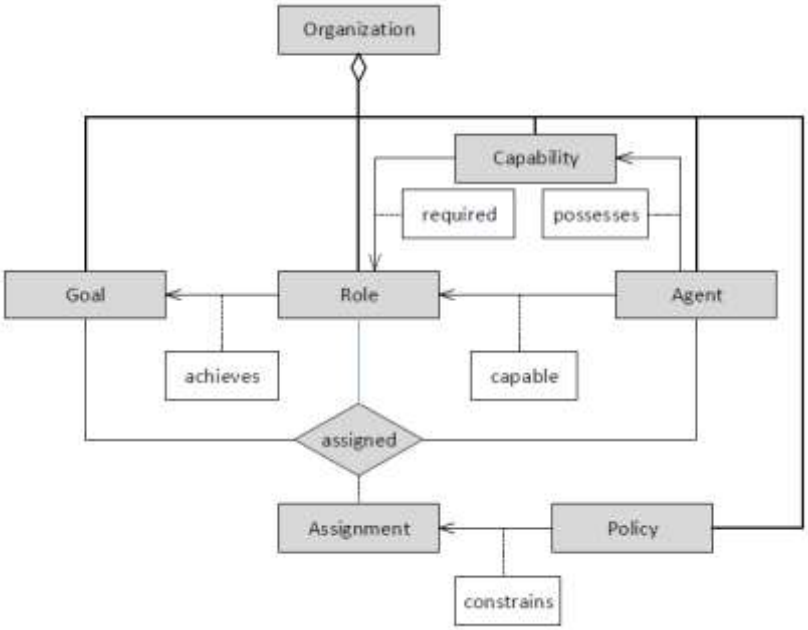

Figure 1. Organization model.

$$required: C \times R \rightarrow Boolean \quad (5)$$

$$constrains: P \times A \rightarrow Boolean \quad (6)$$

$$assigned: AG \times R \times G \rightarrow Boolean \quad (7)$$

Every autonomous system is built to achieve certain goals. In our model each goal has a definition. All goals are placed in a set G. Goals can be critical or noncritical. Function critical determines whether a goal is critical. At this point we do not consider various relationships among goals, for example, such as subgoals, parent and child goals, conjunctive and disjunctive goals and precedence between goals and leave this to our future work.

$$critical: G \rightarrow Boolean \quad (8)$$

We define completeness and incompleteness of a model with respect to organizational goals. The model is complete if all the goals can be achieved, whereas the model is incomplete if at least one goal cannot be achieved. We discuss completeness, validity and viability of the model in the following section.

A goal is directly achieved by a role or a set of roles that each agent is capable of playing. Therefore, each goal is indirectly achieved by an agent or a group of agents through the roles, which those agents are capable of playing. An agent can be capable of playing several roles depending on the inherent capabilities that the agent possesses. Each role requires one or several capabilities that agent/s possess.

Completeness of the model is validated through assignments. Assignments are used to capture information about which agents play which roles to achieve which goals. Assignments are constrained by the organization policies that dictate which assignments are allowed and which are not.

3.2. Completeness, Validity and Viability of the Model

In this section we introduce the concepts of completeness, validity and viability of an organization model.

If all the goals can be achieved, the model is said to be complete. If at least one goal cannot be achieved, the model is said to be incomplete. It is certainly desirable for a model to be complete and in some cases it is absolutely necessary. There are non-critical situations though, in which a model can be incomplete but sufficient for a system. We will research such scenarios in our future work. Complete and incomplete predicates are defined below:

$$complete\ (O) \Leftarrow \forall g \in G\ (\ \exists a \in AG, r \in R \mid capable\ (a, r) \wedge (achieves\ (r, g))) \quad (9)$$

$$incomplete\ (O) \Leftarrow \exists g \in G\ (\forall a \in AG, r \in R \mid \neg capable\ (a, r) \vee \neg (achieves\ (r, g))) \quad (10)$$

A model can be incomplete but still viable. If all critical goals are achieved, the model is said to be viable. A complete model is de facto viable. If at least one critical goal cannot be achieved, the model is said to be nonviable. Viable and nonviable predicates are defined below:

$$viable\ (O) \Leftarrow \forall g \in G \mid critical\ (g)\ (\ \exists a \in AG, r \in R \mid capable\ (a, r) \wedge (achieves\ (r, g))) \quad (11)$$

$$nonviable\ (O) \Leftarrow \exists g \in G \mid critical\ (g)\ (\forall a \in AG, r \in R \mid \neg capable\ (a, r) \vee \neg (achieves\ (r, g))) \quad (12)$$

If all constraints of the organization model are satisfied, the model is said to be valid. For example, one general constraint that we define on an agent is that each agent must possess at least one capability. If an agent possesses no capability, it cannot play any role and, therefore, is useless to the system. This constraint is formalized below, where # is the cardinality of the set:

$$\forall a \in AG, c \in C \mid \#\{\ c \mid possesses\ (a, c)\} \geq 1 \quad (13)$$

Another general constraint is applied to roles and it states that each role has to require a capability. If a role does not require any capability, it means that no agent is capable of playing this role, therefore, each goal that the role achieves cannot be achieved. This constraint is formalized below:

$$\forall r \in R, c \in C \mid \#\{\ c \mid required\ (c, r)\} \geq 1 \quad (14)$$

In order for an agent to be assigned to play a certain role to achieve a certain goal, the agent must possess all the capabilities that are required by the role that achieves the goal. This is formalized in the two general constraints below. Constraint (15) specifies that an agent has to possess all the capabilities that are required of a role to be able to play that role. Constraint (16) specifies that an agent has to be capable of playing a role before the agent can be assigned to play that role.

$$\forall a \in AG, r \in R \mid capable\ (a, r) \Rightarrow \exists c \in C \mid required\ (c, r) \subseteq possesses\ (a, c) \quad (15)$$

$$\forall a \in AG, r \in R, g \in G \mid assigned\ (a, r, g) \Rightarrow capable\ (a, r) \quad (16)$$

3.3. Organization Policy Constraints

Along with general constraints that each organizational model must satisfy, an organization might impose additional constrains through policies. An example of such a policy constraint could be one that states that certain goals must be achieved by several agents to provide redundancy. This redundancy might be important for agent's capabilities that degrade under certain conditions, such as capabilities that depend on a battery life for autonomous drones.

4. Organization Instance

In this section we present examples of what various entities of an organization model could be in a case of a system of autonomous drones and then apply organization model approach to designing a swarm system of autonomous drones that are used for disaster relief aid delivery.

Examples of agents could be autonomous drones, parts of autonomous drones or whole groups or swarms of autonomous drones.

Examples of capabilities are data acquisition capabilities, data access capabilities, data manipulation capabilities, data transmission capabilities, computational capabilities, sensor

capabilities, GPS capabilities, etc. During the system life span capabilities can degrade or improve. For example, sensors usually degrade over time and sometimes their accuracy becomes an issue. Another example of a degrading capability, albeit for possibly a short period of time, would be a limited vision capability during a cloudy or rainy weather. On the other hand, learning algorithms can improve a certain capability. In these cases, the system reorganization could be advisable or necessary.

Each organization has a set of policies that constrain the assignment of agents to roles to goals. These policies could be general in nature or very specific to the autonomous system. An example of a general policy could be a policy that states that each role must be played by at least one agent. An example of a system-specific policy could be a policy that states that no fewer than three agents ought to be involved in working on a specific goal.

4.1. A Swarm System of Autonomous Drones Used for Disaster Relief Aid Delivery

In this section we apply the organization model approach to designing a swarm system of autonomous drones that are used for disaster relief aid delivery. This example is intentionally kept to a minimum for simplicity and clarity of presentation.

4.1.1. Defining Goals. The purpose of this system is to deliver disaster relief aid to the disaster area. Therefore, we define the following two goals for this system:

- g1: deliver disaster relief aid.
- g2: navigate to site.

4.1.2. Defining Roles. Goals are achieved by roles that various agents play. For simplicity we define only two roles that will be directly mapped to achieve the two goals stated above. In general, an organization can have many roles that achieve various goals.

- r1: carrier.
- r2: navigator.

4.1.3. Defining Capabilities. Each role requires a set of capabilities that an agent must possess in order to be capable to play the role. For this system we define three capabilities and list them below:

- c1: carrying capability.
- c2: navigation capability.
- c3: communication capability.

4.1.4. Defining Agents. We also define five agents with various capabilities and list them below:

- a1-a4 – carrier agents.
- a5 – navigator agent.

4.1.5. Organization Instance. After we define goals, roles, capabilities and agents, the organization algorithms make assignments taking into consideration policy constraints to create an organization instance that would achieve all the stated goals. We call such an organization instance a system. For this example, we use a simple greedy assignment algorithm that first tries to find roles to achieve goals, then it tries to match roles with existing capabilities, and then it matches existing agents to capabilities. We consider the study of assignment algorithms an extensive and important aspect of this work not only for initial assignments but also for reorganization of the system when faced with failures and leave it to future work. The organization also produces information to the user whether the system is complete. If a complete system cannot be formed, the organization tries to create a viable system. In all cases, the initial system has to be valid. An organization instance for our example is shown in Figure 2. It is both complete and valid. It is complete because all goals can be achieved. It is also valid because all organization policies are satisfied.

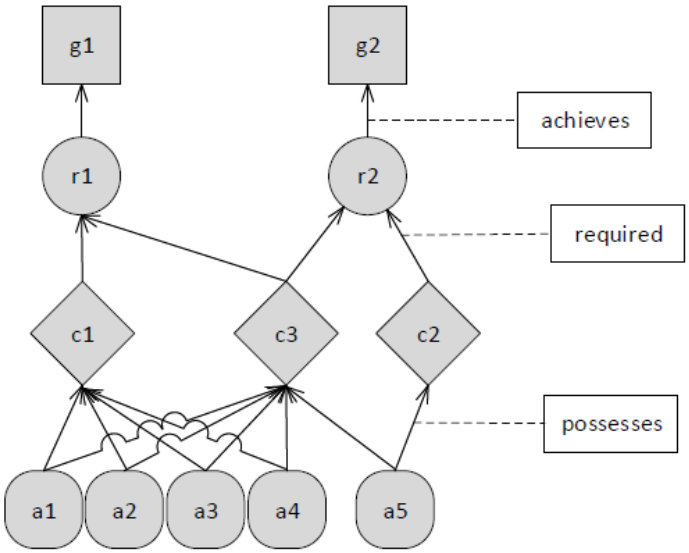

Figure 2. Organization instance.

5. Simulation of the Organization Instance

We simulated the work of the organization instance from Figure 2 on a 1000x1000 grid in a program written in Java. Part of the grid in the simulation is shown in Figure 3. The simulation's task was to move carrier agents a1–a4 from the initial position to the destination – dark grey area on the grid. The simulation worked in time increments. During each time increment communication between agents happened according to the following communication protocol: navigator agent a5 generated next moving instructions for itself, moved accordingly and sent a broadcast message to all other agents with its movement information. Upon receiving the message and based on the information in it, carrier agent generated a move for itself, moved accordingly and sent an acknowledgement message back to the navigator agent. The same sequence of events repeated during next time increment. An intermediate position of agents during the simulation is shown in Figure 3. The simulation continued until all agents a1-a5 reached the destination.

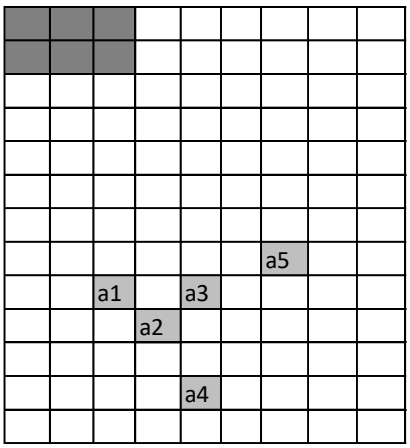

Figure 3. Simulation of an organization instance.

6. Conclusions and Future Work

In this paper, we presented an organization-based approach to modelling autonomous distributed systems. This approach simplifies modelling of such systems and allows better understanding of the overall system. This approach is applicable for modelling a variety of distributed systems. We

demonstrated it for a swarm system of autonomous drones used for disaster relief aid delivery. Even though the example of an organizational instance for a swarm system of autonomous drones used for disaster relief aid delivery is simplified, it still shows the benefits of using such an approach.

In future work, we plan to enhance the organization model with additional features that are useful for the swarm of autonomous drones type of applications and generalize these enhancements to be applicable for autonomous distributed applications in general. We also plan to include organization-based approach to modelling autonomous distributed systems into InDiGO framework and provide tools to extract optimization information from the model for middleware distributed algorithms of a distributed systems. We also have plans to research a possibility of expressing communication protocols through the organization-based modelling so that optimizations for communication in a distributed system could be realized based on the information from a design stage of a distributed system. And lastly, we plan on working on a simulator for a system, developed with the use of organization-based modelling in the InDiGO framework.

7. References

[1] DeLoach S, Wood M and Sparkman C 2001 Multiagent Systems Engineering *The International Journal of Software Engineering and Knowledge Engineering* **11(3)** 231–258
[2] DeLoach S and Wood M 2001 Developing Multiagent Systems with agent Tool *Intelligent Agents VII. Agent Theories Architectures and Languages 7th International Workshop (ATAL 2000)* vol 1986 (Berlin: Springer Verlag)
[3] Carley K 1995 Computational and Mathematical Organization Theory: Perspective and Directions *Computational and Mathematical Organization Theory* **1(1)** 39–56
[4] Carley K 1998 Organizational Adaptation *Annals of Operations Research* **75** 25–47
[5] Lawrence P and Lorsch J 1997 Organization and Environment: Managing Differentiation and Integration *Journal of Artificial Intelligence Research* **7** 83–124
[6] Cohen P and Levesque H 1990 Intention is Choice with Commitment *Artificial Intelligence* **42(3)**
[7] Jennings N 1993 Commitments and Conventions: The Foundation of Coordination in Multiagent Systems *The Knowledge Engineering Review* **8(3)** 223–250
[8] Jennings N 1995 Controlling Cooperative Problem Solving in Industrial Multi-Agent Systems Using Joint Intentions *Artificial Intelligence* **75(2**) 195–240
[9] Grosz B and Kraus S 1996 Collaborative Plans for Complex Group Action *Artificial Intelligence* **86(2)** 269–357
[10] Kinny D, Ljungberg M, Rao A, Sonenberg E, Tidhar G and Werner E 1992 Planned Team Activity *Fourth European Workshop on Modeling Autonomous Agents in a Multi-Agent World (MAAMAW-92)* **830** 226–256 (Springer-Verlag)
[11] Kota R, Gibbins N and Jennings N 2012 Decentralized Approaches for Self-Adaptation in Agent Organizations *ACM Transactions on Autonomous and Adaptive Systems – TAAS* pp 1–28
[12] Kolesnikov V and Singh G 2008 InDiGO: An infrastructure for optimization of distributed algorithms *7th International Symposium on Parallel and Distributed Computing (ISPDC 2008)* pp 401–408
[13] Singh G, Kolesnikov V and Das S 2008 Methodologies for optimization of distributed algorithms and middleware *Proceedings of the 22nd IEEE International Parallel and Distributed Processing Symposium*
[14] Kolesnikov V 2010 Realizing optimization opportunities for distributed applications in the middleware layer by utilizing InDiGO framework *9th International Symposium on Parallel and Distributed Computing (ISPDC 2010)* pp 85–92