Methods and models for the design and study of dynamic agent organizations
Ghijsen, M.

Citation for published version (APA):

General rights
It is not permitted to download or to forward/distribute the text or part of it without the consent of the author(s) and/or copyright holder(s), other than for strictly personal, individual use, unless the work is under an open content license (like Creative Commons).

Disclaimer/Complaints regulations
If you believe that digital publication of certain material infringes any of your rights or (privacy) interests, please let the Library know, stating your reasons. In case of a legitimate complaint, the Library will make the material inaccessible and/or remove it from the website. Please Ask the Library: http://uba.uva.nl/en/contact, or a letter to: Library of the University of Amsterdam, Secretariat, Singel 425, 1012 WP Amsterdam, The Netherlands. You will be contacted as soon as possible.
Simulation tools can be used to gain a better understanding of the effects of the task-environment and organization of agents on their performance. Current simulation environments often lack sufficient control over the environment or lack the ability to systematically vary a number of task-environment and organizational parameters. In this chapter we present a methodology for the design of simulation environments for Multi-Agent Systems. This methodology consists of a theoretical framework, the Extended Organization Design (EOD) model, and an approach for using the EOD to design a simulation environment. The EOD model categorizes and describes aspects of Multi-Agent Systems: their organization, their task-environment, and a set of performance metrics. We show how we use the EOD to develop a parameterized model of the Search and Rescue domain. This model of Search and Rescue is implemented in a prototype and used to analyze the performance of a Multi-Agent Organization in two different evaluations. In these evaluations we show how we design and perform controlled experiments for studying different aspects of performance by systematically varying parameters of both the task-environment and the Multi-Agent Organization.

An earlier version of this work has appeared in “BNAIC 2012: Proceedings of the 24th Benelux Conference on Artificial Intelligence” (Ghijsen, Jansweijer, and Wielinga, 2012).

7.1 INTRODUCTION

In the previous chapters, we have described how to design agents that have the ability to reason about the organization in which they are operating. How well agents are able to reason about their organization will in part determine the performance of the organization. In order to study and evaluate this organizational performance, we focus in this chapter on the environment in which Multi Agent System (MAS) organizations operate. More specifically, we show how to design MAS simulation environments, in such a way that they enable us to study MAS organization performance.

In this chapter we present a methodology to design simulation environments for studying the performance of Multi-Agent Systems. The aim of the methodology is to provide a structured approach for the design of multi-agent systems, such that we provide a MAS experimenter with insight in and control on the simulation environment. We apply our methodology to the design of a Search and Rescue (S&R) simulation environment that allows for a systematic analysis of MAS performance and more specifically, how this performance is influenced by different task-environment and organizational factors. We have
chosen the search and rescue domain due to its distributed and cooperative nature, high degree of uncertainty and dynamics. Furthermore S&R provides interesting challenges to MAS researchers in areas such as robotics (Balakirsky, Carpin, Kleiner, Lewis, Visser, Wang, and Ziparo, 2007), distributed information networks (Teacy, Chalkiadakis, Rogers, and Jennings, 2008), coordination and organization (Paquet, Tobin, and Chaib-draa, 2005) and game theory (Hemaissia, Seghrouchni, Labreuche, and Mattioli, 2007; Chapman, Micillo, Kota, and Jennings, 2009).

A number of simulation platforms have already been developed for simulations in the disaster management domain. Well known are the RoboCup Rescue simulation system (RCRSS) (Kitano et al., 1999), the Urban Search And Rescue simulator (USARsim) (Carpin, Lewis, Wang, Balarkirsky, and Scrapper, 2007) and the distributed building evacuation simulator (Filippoupolitis and Gelenbe, 2009; Dimakis, Filippoupolitis, and Gelenbe, 2010). These simulation environments have in common that they are aimed at providing realistic simulation environments but lack sufficient control to manipulate the task environment and conduct structured experiments.

So and Durfee (1996, 1998) present a possible basis for a more structured approach to studying MAS performance. More specifically, they introduce an organization design model in which the performance of a MAS is influenced by the task-environment and the organizational factors of the MAS. Moreover, they recognize that interaction effects exist between the task-environment and MAS organization factors. Their model however does not provide specific task-environment factors and MAS organization factors. Dignum, Dignum, and Sonenberg (2005a); Dignum, Dignum, Furtado, Melo, and Sonenberg (2005b) present their approach for the design of a simulation tool for studying MAS reorganization. They first identify the factors that determine the need for organization. Then they explore the different ways of reorganization and finally they identify the different triggers for reorganization. Based on this generic framework for reorganization a simulation environment for reorganization is defined. In our approach we combine the basic framework presented by So and Durfee (1996, 1998) with the design approach by Dignum et al. (2005a,b) to describe a methodology for designing MAS simulation environments that can be used for a systematic analysis of MAS performance.

In this chapter we present a methodology that consists of a theoretical framework, the Extended Organization Design (EOD) model, and an approach for using the EOD to design a simulation environment. In Section 7.2 we discuss the EOD model which is based on the organization design model by So and Durfee. The EOD extends the organization design model with a vocabulary to describe the MAS organization and the task-environment in which the MAS organization operates. Furthermore, we provide a more detailed performance model that distinguishes between effectiveness and efficiency and provide a set of performance metrics. In Section 7.3 we demonstrate our approach by operationalizing the generic factors of the EOD in a Search and Rescue
Simulation environment. A prototype implementation of this environment is discussed in Section 7.4. Next, in Section 7.5 we show how this simulator is used in two different case studies. Related work on simulation environments and simulating MAS organizations in the crisis management domain is discussed in Section 7.6.

7.2 EXTENDED ORGANIZATION DESIGN MODEL

The model of organization design by So and Durfee (1996, 1998) explains the interaction between a MAS organization and its task-environment and the effect of this interaction on the performance of a MAS. In this section we present the Extended Organization Design (EOD) model which we will use in the next section to construct a simulation environment for the search and rescue domain. The scope of the EOD model is MAS domains in which agents pursue a common goal in a distributed environment.

![Organization Design Model by So and Durfee (1996, 1998).](image)

The model of organization design by So and Durfee (1996, 1998) is shown in Figure 52. It consists of three main elements: task-environment factors, MAS organization factors and performance metrics. Together, the task-environment and organization factors determine the organizational design space. An implemented MAS organization represents a single point in the organizational design space. Furthermore, the model contains performance metrics which are the criteria for evaluation of the MAS organization’s performance. Thus, a performance measure can be seen as a function over the organizational design space to the performance metrics. In the remainder of this section we will extend this model by So and Durfee (1996, 1998).

Figure 53 shows the core of the Extended Organization Design Model. In the center, depicted by rectangles, the figure shows how different result metrics influence task efficiency and task effectiveness. A plus sign shows a positive influence between two factors and/or performance metrics while a minus sign is used to show a negative influence. On the left and right side of the figure, depicted by ellipses, we distinguish between agent capabilities and organization factors on one side and task and environment factors on the other side. In the following sections we discuss different types of performance metrics and the four main factors that influence performance.
One of the three components in the organization design model is the performance of a MAS. In figure 54 we show a performance taxonomy that describes a set of performance metrics. First, we distinguish between costs and result metrics. Result metrics indicate how well a MAS is performing in a certain task or subtask, while cost metrics indicate the costs made while performing a task. The four generic result and cost metrics described below are based on the selection criteria for coordination mechanisms presented in Chapter 6.

Solution quality is a type of result metric that focusses on the quality of the result when a goal is achieved. Solution quality metrics are especially useful in domains where the amount of time is limited and the performance of the MAS is measured in terms of the quality of work that is achieved within that time. A reward received by agents is an often used operationalization.

Time-to-goal-achievement is a result metric that expresses performance related to the amount of time consumed by agents until a goal is achieved.
This metric is used by Excelente-Toledo and Jennings (2004) to dynamically select a coordination mechanism.

The amount of resources that are consumed by an organization is an important cost metric. Many different consumable and reusable resources can be used while performing a task such as: fuel consumption, CPU load, amount of memory and amount of agents. Note that the amount of time used to complete a task is not seen as a resource but as the time-to-goal-achievement result metric.

Communication-costs is a cost metric that expresses the volume of the amount of information that is exchanged between agents in an organization. This is a useful metric when communication resources are scarce or when costs are involved for using a communication resource.

The performance of organizations can not only be described in terms of the absolute performance metrics as shown in Figure 54, but also in terms of the more relative concepts effectiveness and efficiency. Effectiveness can be defined as the obtained results (i.e. a result measured in terms of one or more result metrics) compared to the maximum obtainable result (see equation 2). Efficiency can be defined as the the ratio between one or more result metrics and one or more cost metrics (see equation 3).

\[
effectiveness = \frac{\text{obtained results}}{\text{max. result}} \quad (2)
\]

\[
efficiency = \frac{\text{obtained results}}{\text{costs}} \quad (3)
\]

7.2.2 Agent Capabilities

The first factor in the EOD model in Figure 53 describes the capabilities of the individual agents out of which the MAS is composed. The different types of agent factors are shown in Figure 55. The two main aspects of the individual agents are their physical capabilities and their intelligence. How (well) does the agent observe its environment and which actions the
agent is able to perform on the environment. The intelligence of an agent is determined by the knowledge of the agent: the declarative knowledge and the procedural knowledge (this distinction is also made in the AgentCoRe framework in Chapter 4).

7.2.3 Organization Factors

The different organization factors of the MAS are shown in Figure 56. The organization of a MAS consists of three aspects: the size, the heterogeneity of the organization and the structure of the organization (see Figure 56).

Organization size determines the potential amount of work that can be done by an organization. Large organizations may achieve better results and thus be more effective than a smaller organization, but its size may increase costs in such a way that its efficiency is lower than in a smaller organization.

We define the heterogeneity of a MAS organization by the heterogeneity of the agents that form the organization. Agents may have different physical capabilities as well as different knowledge. As the population of the agents in the organization becomes more heterogeneous, an organization may be able to handle a wider variety of tasks on the one hand, but may also face more difficult assignment of agents to different subtasks on the other hand.

Following Dignum (2004), we identify the following structural factors of a MAS organization: the communication structure (the language used to communicate and its semantics), the normative structure (the expectations and boundaries of agent behavior in the organization), the social structure (agent roles and relations between roles) and the interaction structure (how should agents interact).
7.2.4 Task Factors

For the task-environment it is obvious to distinguish between task and environment factors (as shown in Figure 53). In Figure 57 we further distinguish between factors that determine the size of the task, factors that determine the complexity of the task, the reward that can be received by performing the task and the task dynamics.

Task size relates to the amount of work that needs to be performed.

In our extended model of organization design, we identify three aspects related to task complexity. The first aspect describes how easy it is to decompose a task into subtasks. In the case a task is easier to decompose into subtasks, the task might become easier to manage (i.e. divide and conquer). The second task complexity factor is the heterogeneity of the subtasks. For example, subtasks may require different effort or have different rewards or priorities. As this heterogeneity increases, task complexity also increases. The third aspect related to task complexity is the amount of relations between subtasks. Such relations can be temporal relations (e.g. task A needs to be completed before task B) or input/output relations (e.g. the knowledge that is produced by executing task A is used as input for task B). As the number of such relations increases a task becomes more complex. The final task complexity factor is the distribution of tasks. As subtasks become more distributed, it will increase the complexity of a task because “distributivity stresses a coordination strategy because it increases agents uncertainty about which agents are currently sharing the task environment and what (if anything) each is, or should be, doing.” (Durfee, 2001).

A reward can be obtained by an agent or its organization after successfully completing a task. This can be a static reward, regardless of how well the task is performed or a dynamic reward coupled to a performance metric.

Dynamics in the task size, complexity or reward, require the organization and agents in the organization to constantly adjust their planning.
namics may also lead to more uncertainty in the organization when agents are not able to keep up with dynamics in their tasks.

7.2.5 Environment Factors

![Figure 58: Extended Environment Model.](image)

The different types of environment factors are shown in Figure 58. The first main environment factor is communication. In order to coordinate their actions agents may have to communicate. Specific properties such as the capacity and the reliability of the communication infrastructure in the environment will influence the organization of a MAS (Stone and Veloso, 1998).

The availability of resources in an environment, is another important factor which affects the ability of an organization to perform its tasks. When resources are distributed over the environment and the location of these resources is unknown, agents will have to spend time and effort on gathering these resources. Coordination is required to determine by which agent and for which task a resource should be used (Malone and Crowston, 1994). Depending on the scarceness of resources, different coordination strategies for allocating these resources are required (Savit, Brueckner, Van Dyke Parunak, and Sauter, 2005). Furthermore, different types of resources can be available. Each type of
resource (e.g. consumable or reusable) causes different types of coordination problems (Crowston, 2003).

As the size of a topological space increases this increases the possibility for tasks and agents to become more distributed. As mentioned before, this increase in distributivity increases the complexity of coordination. The next topological factor is the accessibility of the space, i.e. how much time and effort do agents have to spend to move from one location to another location. Multiple routes to a target location, possible obstacles on those routes and dynamics in these routes, pose many challenges for the agents in an organization (Vigorito, 2007).

The general behavior of the environment can be described by the following factors. The observability of the environment – which can be full or partial – is related to whether or not relevant information for decision making can be observed by the agents in the environment (see Russel and Norvig (2003)). The determinism factor indicates if the outcome of an agent action in a certain state will always result in the same next state or not. Dynamics in the environment determine how the environment changes “spontaneously” without any agent action causing the change.

7.2.6 Using the EOD for Simulation Environment Design

The EOD factors presented in this section provide a framework for the designer of a MAS environment. It covers a wide range of factors and helps the designer to be explicit about the design choices that are made. By being explicit about the design choices, the designer provides more insight in the environment to the user. In the next section we discuss the application of the EOD model to design a search and rescue simulation environment.

To translate the EOD factors to parameters in a specific domain we identify three steps:

• The first step is to select the EOD factors that need to be part of the simulator. For example, is the presence of resources required or does the simulation environment require dynamics? And if so, which aspects of those EOD factors need to be selected?

• The second is the operationalization step in which an EOD factor is represented by a more specific concept. For example the task size factor can be operationalized as the number of actions that need to be performed in order to complete the task.

• The third step is the implementation step in which an operationalized factor is implemented by one or more parameters. For example, in the search and rescue domain, the number of actions to complete a task can be implemented by two parameters: the number of victims that need to be rescued and the size of the search area.
7.3 MODELING THE SEARCH AND RESCUE ENVIRONMENT

In this section we describe the design of our Search and Rescue (S&R) simulation environment. We use the Extended Organization Design (EOD) model to identify the different types of parameters in the simulator and also show how these parameters influence the EOD model factors. The goal of this simulation environment is to provide a parameterized simulation environment that allows for systematic variation of (mainly) task-environment and (partly) MAS organization parameters. It is not the intention to provide a complete instantiation of the EOD model. A complete implementation of the EOD model, i.e. at least one parameter for each task-environment and organization factor and at least one performance metric for each of the generic performance metrics, is beyond the scope of this research.

7.3.1 Operationalization of EOD Factors

In applying the EOD model to design a search and rescue simulation environment we have to operationalize the EOD factors. Because we aim to design a controllable environment we often had to apply two operationalization steps at once. First we introduce search and rescue factors to operationalize the EOD factors. At the same time these operationalized S&R factors are often also a simplification of the real world search and rescue domain. Before we discuss the simulation environment in detail, we first give a short overview of the main features of our simulation environment.

The S&R simulator is a discrete-time simulator. The main motivation for discrete-time is that this makes it easier to implement a system with reproducible results. The environment consists of a rectangular grid topology on which a number of victims are distributed. We have chosen for a rectangular grid to limit agent movements to just 4 directions and make the speed in which the agents move around more controllable. The victims have a certain health status which may decline over time. The initial health state and the decline of health represent how serious a victim is injured. Victims have a fixed location and cannot move. In order to rescue a victim, agents first have to find the victim and then cooperate to rescue the victim by jointly performing a rescue action in the same time-step. In order to find victims, agents can move around on the grid. In a single time-step, an agent can move either one grid cell up, down, left or right. When an agent is moving around, it is able to observe the grid cells that are within its viewing range. These observations are always accurate. Actions related to rescuing a victim and moving around the search area are deterministic. In order to cooperate, agents may need to communicate with each other. To facilitate communication, the simulator provides the agents with a wireless communication infrastructure. Actions related to communication are non-deterministic due to possible failures in the communication infrastructure. The simulator environment is partly observ-
able, i.e. agents cannot see all relevant information needed for their decision making. For example, agents cannot observe whether a communication tower is operational or not. In the remainder of this section, we discuss the design of the simulator in terms of the parameters that influence the factors of the EOD model. An implementation of this simulation model is described in Section 7.4.

7.3.2 Search and Rescue Environment

The environment factors (see Figure 58) that are included in the design of the simulator are communication, topology and behavior. Because environment behavior is only influenced by parameters of communication infrastructure these influences are therefore discussed in the section on communication.

7.3.2.1 Topology

![Topology diagram](image)

Figure 59: EOD model factors for the simulator topology.

The topology of the search and rescue environment can be manipulated by two parameters: the size of the grid on which the search and rescue task takes place, and the degree of obstacles that determine the accessibility of the grid (see Figure 59). Obstacles on the grid are placed between adjacent grid cells. This prevents agents to move between those two grid cells but the obstacles do not block agent observations. Note that in Figure 59 and subsequent figures, we use a rounded rectangle to identify an environment parameter.

The advantage of using a grid structure is that it is easy to scale and its size can directly be related to the effort to search the entire area. When using grids of different size, the number of obstacles also has to scale accordingly to the size of the grid. To achieve this, we model the accessibility of search areas with a degree of obstacles ranging between 0\% and 100\%. 0\% indicates no obstacles and in the case of 100\%, the maximum amount of obstacles will be placed on the map while maintaining a unique route between any two points on the grid.

Figure 60 shows search areas with different degrees of accessibility. Each location on the map is kept accessible to guarantee that all victims can be found and rescued. If victims would become unreachable by randomly placing obstacles, this could cause undesirable effects on the performance of an organization because we would be unable to tell whether differences in performance
were caused by the organization’s behavior or by victims that are simply not reachable.

7.3.2.2 Communication

The EOD factors of the S&R simulator, shown in Figure 61, describe the parameters of the simulator communication infrastructure. It shows that some parameters of the communication infrastructure are related to both communication and environment factors. This communication infrastructure is a simplified wireless communication network. It consists of one or more transmitting/receiving towers that are connected to each other via cables. Each of these towers forms a circular communication cell (see Figure 62 for examples of single-cell communication networks). Whenever an agent sends out a message and it is within the covered area of such a communication cell, the message is picked up and sent to the receiving agents. Each tower acts as a relay station to its neighboring towers. This enables agents to communicate even when they are located in different communication cells. Three types of messages are available to the agent; unicast, multicast and broadcast messages. A broadcast message is sent to all agents that are within range of the communication network of the sender. In the case a directed (unicast or multicast) message is used, the sender has to specify one or more receivers of that message. Each time-step, an agent is allowed to send one message.
Figure 61: EOD model factors for the communication infrastructure.

The uptime of the communication infrastructure also influences environment dynamics and determinism. When the uptime is set to 0% or 100%, the environment is static and deterministic. For any value in between the communication factors of the environment become dynamic and non-deterministic.

Figure 62: Coverage of a single cell network on a 24 × 24 search grid

The capacity of the network is manipulated by the degree of coverage of the network and the maximum message size that individual agents are able to send. Figure 62 shows different levels of network coverage of a single-cell communication network. Outside the coverage area, no communication is possible and within the coverage area, the capacity is determined by the maximum message size. As every agent is allowed to send one message per discrete time-step, the maximum message size is a useful abstraction of the bandwidth limitations of an actual physical communication infrastructure.

The reliability of the communication infrastructure is influenced by the uptime of the communication network and the granularity of the communication infrastructure. The uptime of the communication network ranges between 0% and 100%. Every discrete time-step and for each tower separately we decide with a certain probability if a tower is operational or not. This probability is
based on the specified uptime, such that 0% uptime corresponds to a probability of 0.0 that a tower is operational and 100% uptime corresponds to a probability of 1.0 that a tower is operational. Because the decision if a tower is operational is made for each tower separately, this operationalization of network uptime allows for individual towers to fail in a certain time step while other towers remain operational. This can result in a partial failure of the communication infrastructure. The operational aspects of a tower include receiving messages from and sending messages to agents and also relaying messages from and to other towers.

The granularity of the network is determined by the number of towers. Figure 63 shows a varying degree of granularity for a given network coverage. Granularity affects the reliability of the network in two ways. First, as the number of towers is larger and one tower fails, the other towers can easily bypass the failing tower by re-routing messages over the network. Second, the gaps between the cells become smaller as the granularity increases. This will make it easier for agents to get within the range of a communication tower.

7.3.3 Search and Rescue Task

The four main task factors in the EOD model (see Figure 57) are task size, task reward, task complexity and task dynamics. Each of these factors has one or more parameters in the simulator and is described below.

A rescue task is located at a grid cell and it is defined by the effort that is required to rescue the victim and the reward that will be given to the agents that have rescued the victim. A grid cell can only contain one victim.

It could be argued that some of the task properties can also be described as properties of the environment. For example, health decline of victims is one of the task properties related to task dynamics but it could well be seen as a property of environment dynamics. We have chosen to regard all properties that are directly related to the execution of tasks as task properties.
7.3.3.1 Task Size

Figure 64: Simulator Task Size Model.

The search and rescue task size is determined by the EOD model factors as shown in Figure 64. Search and rescue consists of two different sub-tasks, a search task and a rescue task. The size of the search task is influenced by the size of the search area and the degree of obstacles on the search area. In a larger grid, the agents have to travel larger distances. These distances may become larger by increasing the amount of obstacles in the grid and thus the size of the task becomes larger. The size of the rescue task is influenced by the number of victims that need to be rescued and the effort that is required to rescue a single victim.

7.3.3.2 Task Reward

Figure 65: Simulator Task Reward Model.

A reward received by an agent can be coupled to any of the solution quality oriented performance metrics discussed in section 7.3.6. A high initial health and low health decrease causes victims to live longer and have a high health value in the initial phase of a simulation. Thus, if reward is coupled to any of the solution quality oriented metrics, these two parameters will influence the reward that can potentially be received by an agent. The number of victims in the simulation increases the potential amount of victims that can be rescued.
If more victims are indeed rescued, the summed health may also increase. It should be noted that the task reward parameters only increase the potential reward that can be received. How much reward is actually received depends on how well the agents perform.

Note that using victim health as a reward is a rational implementation of reward, i.e. it serves as an incentive to maximize the overall health state of the victim population. This is contrary to a more subjective implementation of reward in a real-world earthquake example where heavily injured victims are rescued a long time after the earthquake has taken place. The rescue of such victims seems to be rewarded higher by the amount of (media) attention given to the victim and rescue workers than the early rescue of a victim with high health. This more subjective implementation of reward unfortunately does not serve as incentive to maximize the health of the overall victim population and therefore we have decided to take the former approach.

7.3.3.3 Task Complexity

![Figure 66: Simulator Task Complexity Model.](image)

Task complexity is influenced in the simulator by the parameters shown in Figure 66. Subtask heterogeneity is influenced by three parameters in the simulator. The first two parameters are heterogeneity in the initial health of the victims and heterogeneity in the amount of effort that is needed to rescue victims. An increase of these two parameters will result in some victims for which a large reward can be obtained with only a small effort, while for other victims a lower reward can be received while a larger effort is required. The third parameters that influences subtask heterogeneity is heterogeneity in health decline.

Subtask distribution is influenced by the clustering of victims. When this parameter is set to 0% clustering, it means that all rescue tasks are randomly distributed over the search area. As the clustering value increases, the rescue tasks are positioned more around one grid cell in the search area. When clustering is set to 100%, all rescue tasks are positioned as close as possible to
one single grid cell. Figure 67 shows examples of different degrees of clustering. During a simulation, smaller clusters of victims may emerge due to the search and rescue behavior of the agents.

The EOD task-complexity factors, task-decomposability and inter-subtask relations, are not included in the simulator.

7.3.3.4 **Task Dynamics**

![Task Dynamics](image1.png)

Figure 68: Simulator Task Dynamics Model.

Task dynamics is influenced by the health decline of victims. Health decline is linear but can be different for each victim (depending on heterogeneity of the victim population).

7.3.4 **Search and Rescue Agent Capabilities**

![Physical Capabilities](image2.png)

Figure 69: Simulator Agent Physical Capabilities Model.

As shown in Figure 55, the two main factors that describe an agent are its physical capabilities and its intelligence. The simulator only has parameters that influence the physical capabilities of agents: the viewing range of the agent
and the maximum amount of messages the agent can receive per time-step (see Figure 69). The simulator does not impose any constraints and does not influence any of the intelligence factors of an agent.

Agents observe the world depending on their position and their viewing range. An agent with a viewing range of 0 cannot see anything. An agent with a viewing range of 1 can only see the grid cell on which it is standing. A viewing range of 2 results in a $3 \times 3$ square observable area around the agent, and so forth. In their observations, agents are not hindered by obstacles.

The other parameter influencing the physical capabilities of agents is the amount of messages an agent can receive in a single time-step. As mentioned in Section 7.3.2.2, agents can only send one message per time-step. When multiple agents send a message to one agent, the amount of messages that an agent is able to handle is limited.

Agents capabilities to move around the search area are restricted by moving in a single time-step only from its current position to one of its directly adjacent grid cells, excluding diagonal movements. Thus, moving direction and speed have become fixed agent capabilities instead of being parameterized.

### 7.3.5 Search and Rescue Organization

![Organization Model](image)

Figure 70: Simulator Organization Model.

The simulator influences the MAS organization by setting its size (i.e. the number of agents) and by the heterogeneity of the agent population. The level of heterogeneity of the agent population is determined by the differences between the agents in terms of their physical capabilities. The environment does not contain parameters to manipulate any of the structural factors of a MAS organization.

### 7.3.6 Search and Rescue Performance Metrics

For the Search and Rescue domain, many different types of performance metrics are possible. In our simulator we support three result metrics: the total reward that is received (i.e. the summed health of all victims at the end of a simulation), the amount of victims that are rescued and the amount of time
taken to rescue all victims (see Figure 71). Furthermore, the simulator supports two cost metrics that both focus on communication-costs: the amount of bytes that are sent and the amount of bytes received by agents.

7.4 SIMULATOR PROTOTYPE

Algorithm 1 simulation($C, s, t_{max}$)

1: Initialize the environment according to $C$
2: $t=1$
3: while there are victims to be rescued and $t \leq t_{max}$ do
4: Send observations to agents
5: while waiting for all agent to send an action do
6: Collect agent actions and communication acts
7: end while
8: Update communication model
9: Update world model
10: $t=t+1$
11: end while

The simulator has been designed as a client-server architecture. The simulator acts as a server and each agent is a stand-alone client that connects to the server using a socket connection. The main routine of a single simulation is shown in Algorithm 1. Based on configuration $C$ and a random seed $s$, a simulation is loaded and initialized. At the beginning of each time-step (line 4 in the algorithm), all agents observe the state of the environment at that time-step. On lines 5 to 8, the simulation waits until it has received an action from all the agents. This allows the agents to use as much time as they need within a single time-step for their reasoning processes and it ensures that
results are not influenced by system load, network traffic or any other external factors. In the meantime, the simulator also collects communication acts from the agents. Once all agents have responded, the simulator will continue on line 9 by updating the communication model by processing all communication acts and update the status of the communication towers. Finally at line 10, the world model will be updated by processing agent actions and updating the health status of victims and increasing the time.

The simulation process will continue until all rescue tasks have been performed or the maximum simulation time $t_{max}$ has been reached. The design of the simulator is such that when we assume deterministic agent behavior, identical results are obtained when a simulation is initialized with the same random seed and configuration.

To illustrate the interaction between the simulator and the agents, Figure 72 shows an example interaction pattern in a single time-step of a simulation with three agents. The sequence diagram shows that Agent2 sends his communication message to the simulator before sending its act message. This is to ensure that the communication message is processed by the simulator within the same time-step. Any communication act that is received after all actions have been received is stored and processed in the next iteration of the simulation loop.
7.4.1 Initializing the Environment

Initialization of the environment starts with creating a rectangular search grid of given dimensions. The maximum amount of obstacles on a search grid is generated using Prim’s algorithm (Prim, 1957) to create a minimum spanning tree between the elements of the search grid. This results in a search area with unique routes between any two grid elements. Then, obstacles are removed randomly until the desired level of obstacles, as specified in the configuration, is obtained. The communication infrastructure is initialized by distributing the given amount of communication towers evenly over the search area. Grid cells are added to the covered area of each communication tower such that the covered area for each tower is approximately the same and the desired overall network coverage is achieved.

<table>
<thead>
<tr>
<th>range</th>
<th>V1</th>
<th>V2</th>
<th>V3</th>
<th>V4</th>
<th>V5</th>
<th>V6</th>
<th>V7</th>
<th>V8</th>
<th>total</th>
</tr>
</thead>
<tbody>
<tr>
<td>5 ± 0</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>40</td>
</tr>
<tr>
<td>5 ± 1</td>
<td>5</td>
<td>5</td>
<td>4</td>
<td>6</td>
<td>5</td>
<td>5</td>
<td>4</td>
<td>6</td>
<td>40</td>
</tr>
<tr>
<td>5 ± 2</td>
<td>5</td>
<td>5</td>
<td>4</td>
<td>6</td>
<td>3</td>
<td>7</td>
<td>5</td>
<td>5</td>
<td>40</td>
</tr>
<tr>
<td>5 ± 3</td>
<td>5</td>
<td>5</td>
<td>4</td>
<td>6</td>
<td>3</td>
<td>7</td>
<td>2</td>
<td>8</td>
<td>40</td>
</tr>
</tbody>
</table>

Table 3: Varying required effort heterogeneity in a set of 8 victims.

Based on the ranges of victim parameters, the set of victims is created. In the case the victim set is heterogeneous one or more parameters are specified by a range. This range defines the lower and upper bound values of an approximate uniform distribution of values for that parameter. Table 3 shows an example of 8 victims where the heterogeneity in effort to rescue is varied but the average effort value is always 5. Thus, by decreasing or increasing the range of a parameter, heterogeneity of that parameter is manipulated without affecting the summed value of that parameter over all tasks. Values for different parameters are set independently from each other.

Each victim is positioned on the search grid as follows. One grid element is selected randomly as the center of the victim cluster. Based on the degree of clustering, a set of grid elements around this center is selected as potential target locations for a victim. Within this set of grid elements victims are placed randomly.

Based on the ranges of the agent parameters, the agent population is generated. Similar to generating heterogenous task values, the values for agent parameters are also created by generating an approximate uniform distribution of values based on a range specifying the lower and upper bound of those values. Again, values for different parameters are set independently from each other. Agents are then positioned randomly on the search grid. When an agent client connects to the simulation server it is given its physical constraints and its initial position.
7.4.2 Observing the environment

Agents observe the world based on their position and their viewing range. The content of a single observation includes, as shown in Figure 73, the current time, the current status of the agent itself, the grid-cells the agent is able to observe and other agents that are observed. The status of the agent itself includes its id (which is unique), its position, and its capabilities (how far it can see and how many messages it can receive per time step). A grid-cell is described by its coordinates, whether there are any obstacles from this grid-cell to its neighbor cells (obst-north, obst-south, obst-east and obst-west are boolean values that indicate the presence of an obstacle to a neighboring grid-cell) and a description of the victim located on that grid-cell. If no victim is present the state is set to no-victim, otherwise the state indicates whether the victim has already been rescued or whether it still needs to be rescued. In the latter case, the effort required to rescue and the current health state of the victim are also given. Finally the observation contains the id and position of other agents that can be observed.

7.4.3 Communication and action messages

The structure of a communication message is shown in Figure 74. A communication message can either be a broadcast message or a directed message. In
the former case, the message only contains the content that is communicated to the other agents. In the latter case, the message also contains one or more receivers of the content. The simulator does not impose any restrictions on the syntax and semantics of the \(<content>\) part of a communication message. Thus the content of messages is left fully to the designer of the organization’s communication protocol.

\[
\begin{array}{l}
\langle \text{act} \rangle ::= \langle \text{rescue} \rangle \mid \langle \text{move} \rangle \mid \langle \text{noop} \rangle; \\
\langle \text{rescue} \rangle ::= x, y; \\
\langle \text{move} \rangle ::= \text{“north”} \mid \text{“south”} \mid \text{“east”} \mid \text{“west”}; \\
\langle \text{noop} \rangle ::= ;
\end{array}
\]

Figure 75: Action message (agent to simulator).

The structure of an action message is shown in Figure 75. An action message can be a rescue action, a move action, or a do nothing action. In the case of a rescue action, the message contains the coordinates of the victim on which the rescue action should be performed. In the case of a move action, the message contains the direction in which the agent wants to move. A \textit{noop} message is used to indicate the agents does not want to do anything.

7.4.4 Updating the communication model

Once all actions have been received from the agents, the simulator will start to update the communication model. As shown in Algorithm 2 this update consists of two steps: handling failure of communication towers and processing agent communication.

For each tower \(tw\) in the set of towers \(T^s\), we determine with a probability of \(u\) that the tower is operational where \(u\) is the desired uptime of the communication network (see Section 7.3.2.2).

Processing messages is done such that it takes one discrete time step for a message to arrive at the receiver(s). First the set of attempted broadcast messages \(B\) is processed, then all directed messages \(D\) are processed. For a broadcast message \(b(c)\) only its content \(c\) has to be specified. A broadcast message is delivered to an agent if that agent is within reach of the network and it has not exceeded its receive capacity. For a directed message \(d(R, c)\) an agent has to specify the receivers \(R\) and the content \(c\). In the case of directed communication, a message is only delivered if all recipients are within reach of the network and none has exceeded its receive capacity.

In the case a message cannot be delivered, the sender will receive a notification which indicates the reason of the failure. In this prototype, a directed message will only be delivered if it can be received by all the receivers. Thus

1 Capital letters are used for sets, lower case letters denote an element of a set.
we create the situation that a message is either received by all its receivers or not. This prevents a sender having to keep record of each individual recipient whether it has received a message or not. However, if needed this simplification can easily be modified such that directed messages can be received by only a subset of the intended recipients.

7.4.5 Updating the world model

Updating the world model consists of two stages. First, all the agent actions are processed and second, the health decline of victims is calculated. The possible actions are the following:

- A rescue action succeeds only if in a single time step the following conditions hold: (1) Agents participating in rescuing a victim all have to be at the same grid element as that victim at the same time and (2) the amount of agents performing a rescue action is at least the amount of effort required.
• A move action succeeds only if there is no obstacle between the current position of the agent and the grid node the agent wants to move to. The edges of the search area are also modeled as obstacles.

• A do nothing task always succeeds.

The order in which these actions of different agents are processed in a single discrete time-step $t$ does not influence the state of the environment at $t = t + 1$. This is because these actions cannot interfere with each other.

After the agent actions have been processed, the health will be decreased linearly for each victim that is not yet rescued and for which the health is higher than zero.

7.5 Evaluating MAS Organization Performance

To illustrate the use of the simulator we have performed two case studies on the performance of a MAS organization. In these case studies we focus on the effect of organizational and task-environment factors on the collaboration between agents. First we describe a MAS organization that uses coordination by mutual adjustment. Next, we describe the design, data gathering and analysis of two performance evaluation studies. The first study focusses on the impact of network reliability on collaboration and organization performance and the second study investigates the effect of limited communication capacity of agents on collaboration and organization performance.

7.5.1 Organization Design

The organization that we have designed uses a coordination mechanism that can be characterized as mutual adjustment (Mintzberg, 1993). This means the agents form a decentralized organization in which agents mutually adjust their actions to each other in order to perform their tasks. The interaction mechanism that is used is similar to a Contract Net (Smith, 1980). The Contract Net provides a generic mechanism for communicating bids for cooperating on a task, the content of those bids and the offers that other agents can send.

In this organization, an agent can rescue a victim in two ways: the agent can decide to form a coalition for rescuing the victim, or the agent can decide to join a coalition. The process for forming a coalition is shown in Figure 76 and it consists of the following steps: The agent (in this case agent1) sends a request message for forming a temporary coalition to all other agents. Other agents can respond to this request by sending an offer message to join the coalition (in this case agent2, agent4 and agent5 respond). If the initiating agent accepts the offers, the coalition is formed and the agents will rescue the victim at the agreed time. In the example, a coalition is formed between agent1, agent2 and agent4.
The messages that are used in this interaction are \texttt{<request>}, \texttt{<offer>} and \texttt{<accept>} and the content of these messages is shown in Figure 77. A \texttt{<request>} message consists of an expiration time which indicates how long the request is valid, an action-window which is the time-window in which the action should take place and the coordinates of the victim. When the expiration time (\texttt{exp-time}) time has expired, the sender and receivers of this message will no longer consider this request. The action-window allows other agents to decide if they will be able to join the coalition in time at the given location of the victim.

If the agent decides to join a coalition, the agent will respond to the request by sending an offer with an expiration time and an action window. This action window indicates the availability of the joining agent and the action window of the offer should fit within the action window of the request (at best, it is the same as the requested action window). An agent will only reply to requests when complying with that request does not conflict with other commitments.

When the requesting agent has received sufficient offers, the agent will then send the accept message to the agents that will form the coalition. This accept message contains the time-step in which the rescue action should take place.

In this communication scheme, \texttt{<request>} messages are broadcasted while \texttt{<offer>} and \texttt{<accept>} messages are directed messages (unicast and multicast respectively). Furthermore, to prevent agents from flooding the communication...
infrastructure by broadcasting requests, each agent is only allowed to have one valid outstanding request.

### 7.5.2 Performance model

![Performance model diagram]

In this study we will investigate the influence of a number of task-environment and organization factors on the performance of the MAS organization. More specifically, we focus on their influence on the collaboration between agents. As shown in Figure 78, we see two main aspects that determine the performance of an organization. The first is the individual capabilities of agents. When agents walk faster or see more, they can rescue more victims. The second is the collaboration between agents. When agents collaborate better, they will be able to combine their individual capabilities to form a better performing organization. In the following two case studies we will focus only on the latter.

### 7.5.3 Studying the Effect of Communication Reliability

The goal of this first case study is to investigate how the reliability of the communication infrastructure affects the collaboration between agents and the performance of the MAS organization described in the previous sections. In this case study we vary two parameters we know from the EOD simulator model: network uptime and the number of victims. We measure solution quality by the total amount of health of the rescued victims at the end of a simulation and also by the total amount of victims that are rescued.

When the network uptime is less than 100%, two types of events can occur in the MAS organization’s interaction pattern due to communication failure.

1. When the communication network is down for one or only a few time-steps, agents are still able to respond to each others messages before these messages expire. This type of communication failure will cause a relatively small delay in the rescue of a victim that is equal to the network downtime during the interaction sequence.
2. When the communication network is down for longer periods of time, agents will not be able to respond to each others messages before they expire. If this type of communication failure occurs during an interaction sequence, the initiating agent (i.e. the agent sending the <request> message) will have to re-initiate the sequence when a message is expired and no response has been received. This will cause a much larger delay in the rescue of a victim. If the periods of network downtime become too large, no communication sequence can be successfully completed and no victims are rescued.

Based on these two delays we hypothesize that when the uptime of the network decreases, the first type of delay will start to occur in the MAS organizations’s interaction pattern and performance will drop. Then, when we further decrease the uptime of the network, the second type of delay will also start to occur. This will cause a more severe drop in performance. Once the uptime of the network reaches 0%, performance will also drop to 0.

Data for this evaluation was gathered by varying the network uptime between 0% and 100% with a 2% step size. Each simulation was done on a 30 \times 30 search area without obstacles. The communication network consisted of a single communication cell with 100% coverage. The number of victims was varied from 15, to 60 to 120 and victims were evenly distributed over the search area. The effort needed to rescue each victim was set to 3 (i.e. 3 agents are needed to rescue a single victim). The initial health state of a victim was set to 100 and the health decreased with 0.2 every time-step. Furthermore, the organization consisted of 30 S&R agents, each with the same observability range (5 \times 5 range) and the same receive capacity of 100 messages per time-step. Each simulation was initialized with a different random seed causing a different distribution of victims, a different initial agent positions, and a different timing of network failure. Appendix A.1 shows the complete simulator configuration for this experiment.

![Figure 79: Influence of network uptime on performance with 15 victims.](image)
The results of the simulations are shown in Figures 79, 80 and 81. We measure the effectiveness by measuring two performance metrics, the number of victims rescued during a simulation and the total victim health at the end of a simulation. For the first performance metric, effectiveness is obtained by dividing the total number of victims that are rescued at the end of the simulation by the total number of victims in the simulation. For the second performance metric, effectiveness is obtained by dividing the total victim health at the end of the simulation with the total initial health of all victims.

The variation observed in performance between simulations with the same uptime is mainly caused by randomness in the timing and duration of network failures. In simulations with the same uptime, network failures will randomly occur at different moments. When a communication failure happens at a critical moment during the coordination process, that specific communication failure will have a large influence on the performance of the MAS organization. This explains why, even when the overall uptime during two simulations is the same, large differences in performance can be found.
When we look at the number of victims being rescued, it is clear that the uptime of the network influences the maximum number of victims that can be rescued. 15 victims can still be rescued when the uptime is around 15%, 60 victims can be rescued when the uptime is around 30%, while 120 victims can still be rescued when the uptime is around 50%. This indicates a non-linear relation between the uptime and the number of victims that can be rescued.

When we look at the total victim health, it shows that the initial decrease in effectiveness is relatively slow. This can be explained because at high uptime values, communication failures mostly cause small delays. At a certain point however, larger delays are caused by larger periods of network downtime. When the workload per agent is relatively low, the agents still manage to rescue a lot of victims despite the network downtime. However as the workload increases, the delays caused by network downtime prevent the agents from rescuing victims quickly and their total health decreases.

![Graph showing the relationship between network uptime and number of victims rescued](image)

Figure 82: Results of simulations with different numbers of victims and different rescue workloads per agent. Organization size ($a$) is constant at 30 agents.

The results also show that as the initial amount of victims in a simulation becomes larger, performance only reaches its maximum score for higher uptime values. Moreover, when we plot the absolute results of the number of victims rescued for each of the experiments in a single graph (see Figure 82), we would expect that because all other factors remain constant, the decrease in performance would be the same in the three different numbers of victims. Although this is not immediately clear from the data because of the the large
variation, the data suggests that the decrease in performance does not overlap for the different number of victims. For the simulations with 60 victims the decrease in performance occurs at lower uptime values than for the simulations with 120 victims. For the simulations with 15 victims, this decrease occurs even at lower uptime values. Because the variation in the data is large we are unable to conclude that the decreases in performance for all different numbers of victims can be described by linear functions with the same slope.

Figure 83: Performance model showing the influence of network uptime and workload per agent on collaboration.

Nevertheless, we are able to explain the observed differences in performance decrease when we take the workload per agent into account as shown in Figure 83. This workload becomes lower as the number of victims becomes lower. A low workload per agent increases the probability that multiple coalitions will try to rescue the same victim. Thus, while one coalition is delayed or fails due to a communication failure, another coalition might be less influenced by that same communication failure and still be able to rescue the victim in time. In general this shows that organizations with a low workload per agent are able to mitigate negative effects caused by communication failure.

7.5.4 Studying the Effect of Limited Communication Capabilities

The goal of this second case study is to investigate the impact of the limited communication capability of agents in combination with different sizes of organizations on the performance. Limited communication capability of an agent is operationalized and implemented as the maximum amount of messages an agent can receive in one time-step. Organization size is operationalized and implemented by the number of agents in the organization. Performance is measured in terms of effectiveness by the amount of victims rescued during the simulation and the total health of victims at the end of a simulation. Efficiency can be measured by taking either one of the result metrics and using the number of agents as a cost metric.

We expect that by limiting the maximum amount of messages an agent can receive, the organization will become vulnerable to flooding the communica-
tion channel with request broadcasts. Due to the design of the communication infrastructure, broadcast messages will be delivered before directed messages. Thus, if the number of broadcast messages is high and the receive capacity of agents is low, directed messages won’t be delivered. For this organization, the number of broadcast messages depends on the amount of agents that send out a request to rescue a victim.

We hypothesize that large organizations will be more effective. We know from the previous case study that the maximum possible performance is determined by the initial number of victims in the simulation. We expect that for a high number of victims, small organizations will not be able to achieve this maximum performance but if we create sufficiently large organizations, maximum performance will be reached. Furthermore, we expect that introducing limited communication capabilities will also limit performance. In the case of large amounts of victims, the collaboration in large organizations could potentially be more affected by agents with a limited receive capacity because there are more agents that can send requests. Moreover, because agents are not allowed to have multiple valid outstanding requests, an increase in organization size is likely to lead to more outstanding requests.

Data for this evaluation was gathered by running simulations with organizations consisting of 10, 20, 40, 70 and 110 agents. The receive capacity was varied from 1 message per time-step to 12 messages per time-step. Furthermore, each simulation was done on a 40 × 40 search area without obstacles. The communication network consisted of a single communication cell with 100% coverage and 100% uptime. The rescue-task set consists of 150 victims that were evenly distributed over the search area. The effort needed to rescue each victim was set to 3 (i.e. 3 agents are needed to rescue a single victim). The initial health state of a victim was set to 100 and the health decreased with 0.2 every time-step. Furthermore, each agent had the same observability range (5 × 5 range). For each combination of parameters, 10 simulations were run and the average value of those simulations was taken. Each simulation is initialized with a different random seed which causes a different distribution of victims and different initial agent positions. Appendix A.2 shows the complete simulator configuration for this experiment.

Figures 84, 85 and 87 show the results of the simulations. Figures 84 and 85 show that indeed, as organizations get larger, their effectiveness improves, but at the same time the effectiveness of large organizations is also more influenced by the receive capacity of the agents. The organization consisting of 10 agents already starts performing at its maximum capacity when agents have a low receive capacity. As the organizations become larger, the negative influence of limited receive capacity becomes larger. Ultimately, when the agents reach a level of sufficient receive capacity, the larger organizations are most effective.

What these results show is that collaboration in organizations improves when agents are not flooded by messages. Message flooding occurs more in large organizations where the receive capacity of agents is low. We use these
results to update the performance model shown in Figure 83 to the model shown in Figure 86.

Because we vary the organization size in this experiment, it becomes interesting to also study the efficiency of the organization. In this case, we choose the victim health as result metric and the number of agents in the organization as a cost metric. Figure 87 shows that only for small values of receive capacity, the organization consisting of 10 agents is most efficient. For all other values,
the organization consisting of 20 agents is the most efficient. This indicates that adding more agents does not always result in more civilians that can be rescued. For the chosen map size, number of civilians and other simulator parameters in this experiment, an organization of around 20 agents appears to be the optimal organization size.

Therefore, from the perspective of efficient use of resources, one could conclude that deploying larger organizations is senseless. However, from the perspective of the search and rescue domain, ultimately the number of victims

Figure 86: Performance model showing the influence of network uptime, workload per agent and message flooding on collaboration.

Figure 87: Effect of limited receive capacity and organization size on organization efficiency measured by the obtained victim health after a simulation per agent.
that are rescued can be considered the most important performance indicator. More generally one could conclude that Search and Rescue organizations are expected to use all their available agents and (communication) resources as much as they can in order to obtain the highest level of effectiveness possible.

As explained before, a possible explanation for the decrease in performance for larger organizations is the number of requests that are flooding the communication infrastructure. To exclude other possible influences like for example organizational overhead we ran another series of experiments. In these experiments, we vary the size of the search area while keeping the organization size (70 agents) and number of victims (50 agents) fixed. For the search area, we use a $15 \times 15$, $25 \times 25$, $35 \times 35$ and $45 \times 45$ grid size. All other parameters are still the same as with the previous experiment.

By increasing the search area, we increase the search workload of the agents but at the same time we also decrease the victim density on the search area. We expect that for a lower average victim density (with victims randomly distributed), while keeping the rescue workload constant, the number of requests that agents want to send in a single time-step will be reduced. This is because the search effort will increase while the rescue effort remains constant. This search effort requires no communication and therefore, the communication channel will no longer be flooded and the interaction sequence for forming a coalition will get a chance to be completed.

Figure 88: Effect of victim density and limited receive capacity on organization effectiveness measured by the relative number of victims alive after a simulation.
The results of these simulation are shown in Figure 88\(^2\) and it shows that the effectiveness is indeed positively influenced by a decreasing victim density on the map.

![Performance model showing the influence of network uptime, workload per agent, message flooding and victim density on collaboration.](image)

Based on these results we can now update the organization performance model to include victim density as shown in Figure 89. This updated performance model shows us that a higher victim density makes it more difficult for the agents to collaborate. This is because a high victim density increases the probability for coalitions to work on the same victim. Because there is no coordination between the coalitions on which victim to rescue, valuable time can be lost because coalitions will waste time on trying to rescue victims that are already being rescued by other coalitions.

### 7.6 Related Work

A large number of simulation platforms have been developed and most of them focus on a specific sub-domain within the area of multi-agent simulations. The most well know simulation environment for disaster management is the RoboCup Rescue simulation system (RCRSS) (Kitano et al., 1999) which is used in the Agents Competition of the RoboCup Rescue simulation league. RCRSS\(^3\) provides a detailed scenario of urban search and rescue after an earthquake has hit a city. The aim of the simulator is to compare the performance of a

---

2 Measurements for relative total victim health have been omitted because the resulting graphs were almost identical to the results shown in this figure.

3 In this paper we discuss RoboCup Rescue version 0.48.
number of different MAS organizations in exactly the same setting. Therefore, similar to our approach, the behavior of the environment has been decoupled from the behavior of the agents. However, due to its focus on varying MAS organizations, the simulator does not provide easy manipulation of the environment itself.

The RCRSS is aimed at providing a realistic model of the environment. Because there is no explicit model of this environment, it is difficult to understand the consequences of manipulations on the environment. For example when the location of a victim is manipulated, the change of location may also impact the speed with which the health of that victim decreases as well as the effort that is needed to rescue that victim. In the design of our simulation environment we have tried to reduce such effects by removing influences between, for example, the location of a victim, the speed of its health decrease and the amount of effort needed to rescue the victim. This has resulted in an environment that is perhaps less realistic but easier to manipulate and allows the effects of manipulations to be easier understood.

Another difference with our approach is the communication system of the RCRSS which restricts broadcast messages to agents of the same type. Furthermore, agents can only use directed communication when they are within a certain range of each other. Our simulation environment improves on this by implementing a broadcast that is received by all agents. Also, our system does not impose any limitations to directed communication by allowing one or more recipients to be specified. By allowing more freedom in communication, we also allow for the design of a wider variety of MAS organizations.

The Urban Search And Rescue simulator (USARsim) is a multi-robot simulation platform (Carpin et al., 2007) for robots in the search and rescue domain. Similar to the RCRSS, the aim is to provide a realistic simulation environment and due to its application in a competition, the design of the simulation system is not suited for easy manipulation of environment parameters. Also, because of its focus on small teams of robots with limited interaction between the robots, this environment is not suited for implementing different types of MAS organizations.

Another simulation environment in the disaster management domain is the distributed building evacuation simulator (Dimakis et al., 2010). The focus of this simulation environment is mostly on the computational challenges encountered when running large scale and distributed simulations of building evacuation. In Filippoupolitis and Gelenbe (2009), the building evacuation simulation environment is used to evaluate the impact of a decision support system for emergency response. In this study, the impact and performance of a decision support system in different settings is evaluated. However, instead of analyzing the behavior and performance of their decision support system, the goal of this evaluation is mainly to compare the use of their system to a situation in which no decision support system is used.
The Brahms simulation model (Sierhuis, Clancey, and Hoof, 2009) assumes a BDI model for its agents and aims at simulation and analysis of the behavior of small teams of human agents. Brahms has also been applied in the disaster management domain to study the effects of adaptive information distribution system on the performance of a crisis management organization (Netten, Bruinsma, van Someren, and De Hoog, 2006). The main difference with our work is that the Brahms simulator and its application in the crisis management domain is aimed at finding and studying the cause of erroneous behavior in organizations while our work on the EOD is aimed at understanding the influences of the task-environment and organization factors on the performance of an organization.

A more generic simulation system that does not assume a specific agent architecture or environment model is MASON (Luke, Cioffi-Revilla, Panait, Sullivan, and Balan, 2005). MASON is a discrete-event multi-agent simulation toolkit implemented in Java and it’s aim is to provide a generic platform for multi-agent simulations, ranging from swarm robotics to machine learning to social complexity environments. One of its requirements is the ability to produce identical results independent of platform which is achieved by using Java to produce identical cross-platform results. Due to its generic applicability, the simulation platform itself does not provide environment models. The design and implementation of controllable and repeatable experiments is left to the user of the MASON toolkit.

An example of a simulation environment that provides more control to the user is the predator-prey pursuit simulation system (Alcazar and Garcia, 2006). In this system, an explicit mathematic model is provided to describe the predator prey domain. Control is provided by manipulating the different parameters in the mathematical model and the predator prey model is evaluated while systematically varying a number of different parameters. In this work on the predator prey domain however, the simulator integrates the behavior of the environment and the agents in one single model, while in our work the behavior of the environment and the behavior of the agents has been decoupled. Our search and rescue simulator only simulates the behavior of the environment and thus we leave room for the study of many different behaviors of the agents.

Another example of a more controllable environment is a system for simulating software evolution (Stopford and Counsell, 2008). Although no explicit environment model is presented and the application domain is completely different from ours, their simulator allows for the systematic variation of a number of parameters and the authors use a clear methodological approach to analyze the results. The latter is a good example of the type of experiments we envision for our simulation environment.
7.7 Conclusions

In this chapter we presented a methodology for the systematic design of simulation environments. Our methodology consists of the Extended Organization Design model which is a domain-independent model to describe organizations of agents, the task-environment in which they operate and how performance is influenced by task-environment and organization factors. The EOD model provides a structure and vocabulary for task-environment factors, MAS organization factors and performance metrics. Furthermore, we provide a three-step approach for operationalizing and implementing the EOD factors as parameters in a simulation environment.

We have used our methodology to design an agent simulation environment for the Search and Rescue domain. The main aim was to create a controllable experimentation environment for conducting experiments on the performance of Multi-Agent Organizations. Based on this design a Search and Rescue simulation environment has been implemented. The distributed nature of Search and Rescue - agents have to locate victims that are spread over a search area - and the agents sharing a common goal, makes Search and Rescue suitable to apply the EOD to.

In two case studies we have conducted a number of experiments to illustrate how the simulation environment is used. The first study focussed on analyzing the effect of communication failure in combination with different levels of workload on the performance of a MAS organization. In the second case study we investigated the effect of limited communication capabilities of agents and the size of the organization on performance. Both evaluation studies show that by using EOD concepts we are able to describe the dependencies between task-environment and organization factors and the performance of a multi-agent organization.