Multi-Agent Simulation of Disaster Response

Daniel Massaguer, Vidhya Balasubramanian, Sharad Mehrotra, and Nalini Venkatasubramanian

Donald Bren School of Information and Computer Science
University of California, Irvine
Irvine, CA 92697, USA
{dmassagu, vbalasub, sharad, nalini}@ics.uci.edu

ABSTRACT

Information Technology has the potential of improving the quality and the amount of information humans receive during emergency response. Testing this technology in realistic and flexible environments is a non-trivial task. DrillSim is an augmented reality simulation environment for testing IT solutions. It provides an environment where scientists and developers can bring their IT solutions and test their effectiveness on the context of disaster response. The architecture of DrillSim is based on a multi-agent simulation. The simulation of the disaster response activity is achieved by modeling each person involved as an agent. This finer granularity provides extensibility to the system since new scenarios can be defined by defining new agents. This paper presents the architecture of DrillSim and explains in detail how DrillSim deals with the edition and addition of agent roles.

Categories and Subject Descriptors

I.2.11 [**Computing Methodologies**]: Artificial Intelligence. Distributed Artificial Intelligence[Intelligent agents, Multi-agent systems]; H.1.2 [**Information Systems**]: Models and Principles. User/Machine Systems[Human information processing]; I.6.3 [**Computing Methodologies**]: Simulation and Modeling Applications; I.6.4 [**Computing Methodologies**]: Simulation and Modeling. Model Validation and Analysis

General Terms

Design, Algorithms, Experimentation

Keywords

Agent-based simulation and modeling, applications of autonomous agents and multi-agent systems, artificial social systems

1. INTRODUCTION

Efficacy of disaster response plays a key role in the consequences of a disaster. Responding in a timely and effective manner can reduce deaths and injuries, contain or prevent secondary disasters, and reduce the resulting economic losses and social disruption. Disaster response is basically a human-centric operation where humans make decisions at various levels. Information technologies (IT) can help in disaster response since improving the information management during the disaster–collecting information, analyzing it, sharing it, and disseminating it to the right people at the right moment–will improve the response by helping humans make more informed decisions.

While innovations in information technology are being made to enhance information management during a disaster [18], evaluating such research is not trivial since recreating crisis scenarios is challenging. The two main approaches to recreate crisis scenarios are simulations [6, 9, 5, 8, 19] and drills. Both approaches have their benefits and drawbacks, simulating a disaster entirely by software lacks realism; continuously running drills is expensive.

In DrillSim [4], we propose to take the best of both approaches. We have instrumented part of our campus with sensing and communication capabilities such that we input data from a real ongoing drill into a multi-agent simulation, and vice versa. This way, a simulated disaster response activity gains realism since it occurs within a real space with input involving real people, sensors, communication infrastructure, and communication devices. Simultaneously, the limited drills are augmented with virtual agents, sensors, communications, and hazards enhancing the scope of the response activity being conducted. This, along with the modularity of DrillSim, enables a framework where the impact of IT solutions on disaster response activities can be studied.

DrillSim is an augmented reality micro-simulation environment in which every agent simulates a real person taking part in the activity. Every agent learns about its environment and interacts with other agents (real and virtual). Agents execute autonomously and make their own decisions about future actions. In an actual real-world exercise, the decisions each person takes during a disaster response depend on a set of physical and cognitive factors as well as the role that person is playing. The modeling of DrillSim agent behavior considers these factors. Creating a scenario is now based on binding a small set of roles and physical and cognitive profiles to the large number of agents. One of the key advantages of a micro-simulation using a multi-agent system is the ability to bring new roles anytime and

AAMAS'06 May 8–12 2006, Hakodate, Hokkaido, Japan.

study their interaction with the old ones or even create a complete different scenario.

The rest of the paper is organized as follows. Section 2 compares our approach with related work. Section 3 presents the DrillSim environment. Section 4 describes the DrillSim agent model and Section 5 elaborates on how agent roles can be instantiated and edited. In Section 6 we illustrate the use of DrillSim through experiments conducted in the context of an evacuation simulation. The paper concludes with future work in Section 7.

2. RELATED WORK

The need for multi-agent models for emergency response that incorporate human behavioral aspects has been realized [16, 17]; an example of one such recent effort is at the Digital City Project at Kyoto University [19]. Other multi-agent simulators for disaster response include the efforts within Robocup-Rescue Simulation Project [8]. Our work is similar in spirit to those suggestions and enhances these initial efforts significantly. First, agent models within our simulator capture the sensing and communication capabilities of individual agents at very fine granularity. These models allow us to systematically model the information received by individual agents over time. Dynamic changes to information available to an agent results in behavior changes (at the agent) represented in our system using stochastic models based on neural nets. Second, our system consists also of a pervasive space [7] that captures a drill of the activity in the real space. This pervasive space consists of a variety of sensing, communication, and display technologies and is used to conduct and monitor emergency drills within our campus. This way, our simulations can replicate real drills captured within the pervasive space. That allows comparing the behavior of simulated humans with the behavior of real humans for validating and calibrating the simulated human models. Furthermore, the DrillSim augmented reality environment also allows integrating a simulation with an ongoing drill, enhancing the simulation with realism and augmenting the drill with simulated actors, hazards, resources, and so on.

3. DRILLSIM

DrillSim [4] is a testbed for studying the impact of Information Technology (IT) in disaster response. DrillSim provides a simulation environment where IT metrics (e.g., delay, call blocking probability) of a given IT solution are translated to disaster metrics (e.g., time to evacuate, casualties). This way, different IT solutions for disaster response such as [11, 21, 20, 12] can be systematically and consistently tested.

DrillSim is a plug-and-play system. It enables scientists and developers to (i) test the impact of one solution at a time and (ii) reconfigure the set of technologies used and evaluate the overall efficacy of integrated solutions. In addition, the system is designed to allow plug-and-play operation of different external simulators such as network, hazard, or traffic simulators. This way, available network, hazard, traffic simulators, etc developed by domain experts can be exploited in DrillSim. Within DrillSim, new agent behavior models can also be plugged in and the impact of IT on agent behavior can be observed.

The software architecture of DrillSim is shown in Figure 1. The core of DrillSim is the simulation engine. It is the principal component that drives the activity and it is composed of a multi-agent disaster response simulation. It consists of the simulated geographic space, the current evacuation scenario (i.e. where people are and what they are doing), and the current crisis as it unfolds. The engine keeps a log of every event, information exchange, and decision. Agents also keep an individual log, which is consistent with the global log.

The simulation engine interacts with the data management module, the input interfaces and external modules, VR/AR module, and output interfaces and visualization. The data management module manages the data exchange between components. It is responsible for handling a high rate of queries and updates coming from the agents, and logging the events of the activity. An important aspect of this module is the representation of the spatial and temporal data so as to adequately support functioning of the simulated activity.

The inputs to the simulation engine can be divided into configuration inputs and interaction with external modules. Configuration inputs create a scenario by initialize parameters regarding space, resources, crisis, infrastructure, agents location, agent profiles, and agent roles. External modules can be plugged to DrillSim so that crisis, communications, traffic, etc can be simulated in external simulators developed by domain experts. Mediators translate the interfaces among external simulators and DrillSim.

The VR/AR module is responsible for the Virtual Reality/Augmented reality integration. The activity takes place in a physical space instrumented with visualization, sensing, and communication infrastructure. This pervasive space includes multi-tile displays, video and audio sensors, people counters, built-in RFID technology, powerline, Ethernet, and wireless communications [7]. This provides an infrastructure for capturing the physical space and activities unfolding during a drill. This information is then input into the simulation engine, augmenting the simulation with real people, space, and also sensing and communication infrastructure. The real world is also augmented with the simulated world. This is achieved by projecting the simulated world into visualization devices (e.g, a PDA) and allowing users to interact with the simulated world.

Several visualization interfaces are supported: multi-tile display, workstation, PDA, and EvacPack. The multi-tile display and workstation interfaces allow a user to observe the activity and control it (e.g., configure and start a simulation). The PDA and EvacPack allow a user to take part of the activity. Location-aware information is sent to the PDA and its, simplified, interface allows its user to see the simulated scenario as well as interact with it. Evacpack is a portable computer composed of a Windows box with wireless internet connection, a pair of MicroOptical SV-6 VR glasses [2], a wearable keyboard, a wireless mouse, headphones, a microphone, and a webcam. With more resources than the PDA, Evacpack also gets a location-aware view of the simulated scenario and allows its user to interact with the simulated agents. A part from the visualization, the engine also outputs disaster response metrics regarding the activity.

4. DRILLSIM AGENT MODEL

Each agent simulates a person and agents are the main drivers of the response activity. We illustrate the agent

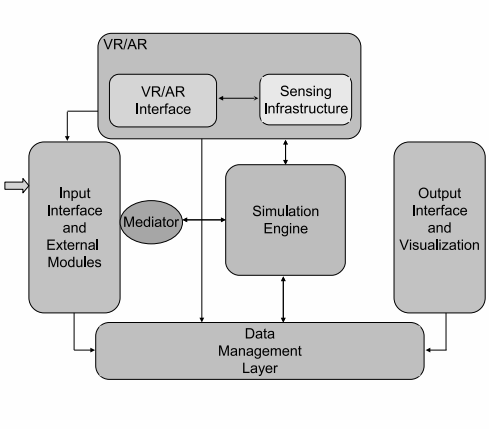

Figure 1: DrillSim architecture.

model in DrillSim through an evacuation activity following a crisis (e.g. fire). Each agent has a subjective view of the world it lives in. This view depends on the agent's sensing characteristics (e.g., how far it can see). Based on this observed world and the agent's cognitive characteristics, the agent takes decisions (e.g., exit the floor), which results in the agent attempting to execute certain actions (e.g., walking towards the floor exit). Every agent in DrillSim has the following attributes: state, observed world, role, profile, and social ties with other agents. These attributes dictate how an agent behaves and we describe each attribute in more detail below.

Agent Attributes

State. The state of an agent comprises its current location, health, and devices it carries. The state is also formed by the information the agent knows about the world, the decisions it has taken, and the plans it has generated to realize these decisions.

Observed world. An agent's observed world is what the agent knows about the world it lives in. It is composed of the information it knows a priori (e.g., a map of the building) and the information it gains during its life. An agent gains information about the world it lives in via the sensors it has access to (e.g., its own eyes, its cellphone). An agent's observed world is represented as a matrix *Obs* and a message queue. The matrix contains the representation of the geographic space and the localization of agents, obstacles, and hazards. Each cell in the matrix corresponds to an observation of a small region in the real world–that is, the real world is geographically divided in equal sized cells. Each cell contains a tuple of the form:

$$Obs_{i,j} =< time, obstacle, occupied, hazards > \quad (1)$$

where *time* corresponds to the time the observation was made, *obstacle* is a value between 0 and 1 and represents the difficulty an agent faces in traversing a cell, *occupied* contains a list of agents occupying that cell, and *hazard* contains a list of hazards present in that cell. Each agent updates its matrix based on their perceptual characteristics (specified in the agent's profile).

The message queue contains messages received from other agents. It is a finite and cyclic buffer queue–agents, as humans, can only remember a finite number of messages. The amount of messages an agent remembers is also specified in its profile. Each message m contains the time $m.time$ it has been received, its source $m.source$, its destination $m.destination$, receiving device $m.device$, and message contents $m.msg$.

Role. An agent's role dictates the decisions an agent makes. For instance, a student at school, on hearing a fire alarm, may decide to evacuate the building or to finish the paper he/she is working on. On the other hand, a fire fighter's decisions might involve enter the building on fire to look for people or report to the fire lieutenant. Therefore, the role an agent plays in the activity is a key element when modeling the agent's behavior. In fact, modeling the activity as a multi-agent simulation provides the flexibility to be able to change the scenario being simulated by changing the agent roles. Section 5 gives more details in role management in DrillSim.

Profile. An agent's profile includes the agent's perceptual and mobility characteristics, initial health, role, the communication and sensing devices carried, and some cognitive characteristics. The agent's perceptual characteristics along with the devices carried determine the information the agent can sense and communicate. The mobility characteristics include the speed of movement of the agent.

An agent's information abstraction is also influenced by other cognitive factors. To accommodate this, an agent's profile includes how often an agent takes a decision and the definition of an activation function s. The activation function expresses how the agent perceives information. The simplest function would be $s(x) = x$, where for any objective input x (e.g., temperature, risk), the agent perceives it objectively. Currently, we are using a sigmoid function , which is a common activation function used in artificial neural networks [15].

Social ties. Agents represent people and, as such, people develop social ties with each other. Social network analysis focuses on the relationships among social entities, and on the patterns and implications of these relationships [24]. Currently, we are modeling two basic social networks. In our case, the social entities are agents and the relations capture how much an agent trusts another agent or how much an agent would wait for another agent when evacuating.

The more an agent trusts another agent, the more the reliability associated with the information received from that agent. Agents also associate different degrees of trust to different devices. To represent this social network, each agent has a vector *Rel* where Rel_{a+d} contains the relevance the agent gives to a message from agent a received via device d.

The other social network is regarding the fact that, as also observed in several evacuations in our building, people tend to evacuate in groups. To represent this social network, each agent has a vector $MovingInf$ where $MovingInf_{(a)}$ represents how much an agent's decision to evacuate is influenced by another agent a that has decided to evacuate.

Agent behavior

Agent behavior in DrillSim is motivated by well-studied models of information processing in humans [22, 23]. These models are formed by four entities: Data, Information, Knowledge, and Wisdom. A data element is the codification of an observation. An information element is an aggregation of data elements, and it conveys a single and meaningful message. A knowledge element is the union of pieces of information accumulated over a large period of time. A wisdom element is new knowledge created by a person after hav-

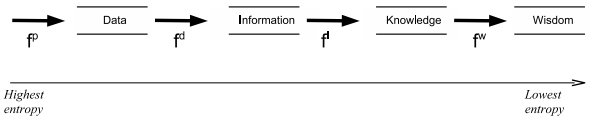

Figure 2: Basic entity model of information processing.

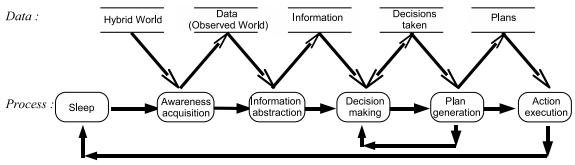

Figure 3: DrillSim Agent Behavior process.

ing gained sufficient knowledge. There exists a function for abstracting wisdom from knowledge, knowledge from information, and information from data (f^w, f^I, f^d). There also exists a function (f^p) that codes observations into data (Figure 2). The goal of each function is to gain a clearer perception of a situation by improving the orderliness (i.e., lowering the entropy); which enables further systematic analysis.

Agent behavior in our model is illustrated in Figure 3. We model agent behavior as a discrete process where agents alternate between sleep and awake states. Agents wake up and take some action every t time units. For this purpose, an agent acquires awareness of the world around it (i.e. event coding), transforms the acquired data into information, and makes decisions based on this information. Then, based on the decisions, it (re)generates a set of action plans. These plans dictate the actions the agent attempts before going to sleep again. For example, hearing a fire alarm results in the decision of exiting a floor, which results in a navigation plan to attempt to go from the current location to an exit location on the floor, which results in the agent trying to walk one step following the navigation plan. Note that the time t is variable and depends on each agent's profile. Furthermore, when an agent wakes up, it may bypass some of the steps from Figure 3. An agent acquires awareness every n_c time units, transforms data into information every n_d time units, makes decisions and plans every n_I time units, and executes actions every n_a time units. The relationship between these variables is: $t \leq n_a \leq n_c \leq n_d \leq n_I$ (e.g., $n_I = n_d = 2n_c = 4n_a = 4t$). This bypassing allows us a finer description of personalities and makes the system more scalable since there might be thousands of agents in one simulation.

5. AGENT ROLE EDITING

In DrillSim every agent simulates a real person. A scenario is recreated by binding agents to a small set of predefined roles, instantiating each agent's profile, and instantiating social networks among agents. For example, an evacuation of an office building is recreated by creating as many agents as people would work in the building and then binding most agents to the evacuee role and the rest to other roles such as floor warden (person in charge of evacuating one floor). Also, a profile distribution is instantiated for every role (e.g., every evacuee's age is set), and the underlying social networks present in an office building are instantiated. Factors such as the randomness involved in decision-making,

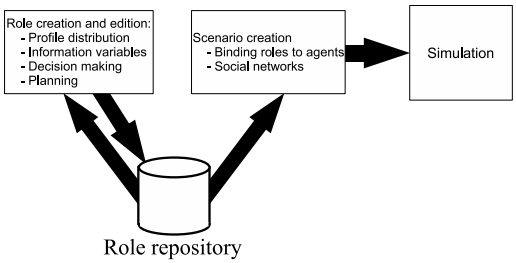

Figure 4: GUI for editing agent role.

the different initial situation of every agent, and the underlying social networks guarantee a certain degree of variability on the agents behavior.

DrillSim incorporates a few predefined roles. However, this is not sufficient to cope with all the scenarios that can be played out. Evacuating an office involves employees and first responders; evacuating a school involves students, teachers, and principals; and responding to a radioactive spill involves a hazardous material (hazmat) team. To support this kind of extensibility in DrillSim, a user should be able to define new roles as needed.

Figure 4 depicts the process of managing roles. A user can edit existing roles or create a new role based on another role or from scratch. This process is done before running the actual simulation and the new roles are stored in a role repository. When creating the initial scenario of a simulation, the user specifies how many agents are needed, the roles of each agent, and the social networks among the agents. For specifying a role, the user needs to specify information regarding profile, information variables, decision making, planning, and social networks.

Profile. For every role, the user indicates a profile distribution associated with it. The profile specifies several factors that influence what people sense, what decisions they take, and how they perform actions. Some of these factors include their visual and hearing range, their personality (e.g. risk takers), their health, and their speed of walking. In addition, attributes of other agents such as health, age, and sex may influence's a person's behavior. Defining the profile means providing a mean and a variance for each of the profile's parameters. The current prototype implementation supports a subset of these factors, i.e. visual acuity, personality, speed of walking. This subset will be revised and enhanced as and when they become relevant to the agent's behavior.

Information variables. As depicted in Figure 3, the world observed by an agent is abstracted into information variables, which are the input to the decision making module. Not all agents take the same decisions. For example, an evacuee might not decide to put on a floor warden's vest and safety helmet. Adding new roles involves sometimes adding new decisions. Some information important for this decision might have not been relevant for other roles. Hence, sometimes, a user might need to specify new information variables. Namely, the user has to specify how this new information variables are named and how they are abstracted from observed world and state (e.g., health). The user specifies the name of the new variable and the code that, based on the agents observed world and state, abstracts the new information variables.

Decision making. An agent's decision making is mod-

eled as a recurrent artificial neural network [15]. Briefly, the core of the neural net describes the importance of each input to the decision-making module (i.e. information variables, decisions already taken) and results in the probability of taking each decision. Another part of the neural net deals with, given this probability, randomly deciding whether the decisions are taken. Given the same input, agents with different roles may take different decisions. For example, on hearing a fire alarm, a floor warden will decide to put his/her vest and helmet on, whereas an evacuee may decide to exit the building. When defining a new role, a user has to set the weights of the decision-making neural network.

Plan generation. Once an agent has decided to take a decision, it computes a plan of actions to perform such decision. For instance, when an agent decides to put the floor warden's vest and helmet, it has to plan a set of steps from its current location to the vest and helmet. For each possible decision, the user has to specify the code that returns the corresponding plan of actions.

Social networks. Some of the decisions depend also on the underlying social networks. For example, recall that the importance an agent gives to a message also depends on the message source. Social networks are instantiated a posteriori, when defining a scenario. However, certain information variables depend on the social networks as well. Therefore, when defining a new role, a user needs to specify the dependencies with social networks.

6. CASE STUDY: EVACUATION SIMULATION IN DRILLSIM

This section exemplifies one of the advantages of using multi-agent simulation–adding new roles on-demand. In particular, the response activity being simulated is the evacuation of our building floor and the different roles are based on the emergency response plan of the University of California, Irvine (UCI) [3]. Based on this response plan and on the maps of our building floor in UCI, we ran a series of simulations on a DrillSim prototype. The rest of this section describes the different roles defined on the UCI emergency response plan, presents an implementation of DrillSim, and discusses some experiments and their results.

Note that the results here reported are only for illustration purposes. The validity of them depends on the validation of the agent behavior. Validating the current roles and calibrating them is part of our ongoing work.

6.1 Implementation

An initial DrillSim prototype has been implemented in Java. The multi-agent platform chose is JADE [10]. JADE seamlessly integrates with Java and provides a framework for easy development and deployment of multi-agent systems. The prototype provides a GUI for controlling the simulation that allows real humans to observe and interact with the drill simulation. Figure 5 shows a snapshot of the GUI. In particular, it allows a user to start the simulation, pull the fire alarm, input a hazard, get an arbitrary view of the simulation in 2D or 3D, get a 3D view of what an agent is viewing, send messages to other agents, control an agent, and get statistics of the evacuation.

In addition to this GUI, this first prototype also includes an interface that allows creating and editing agents roles (Figure 6). In particular, it allows specifying the mean

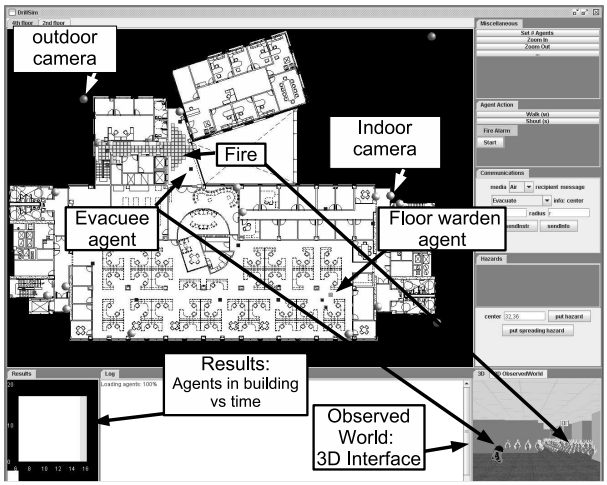

Figure 5: Snapshot of the prototype.

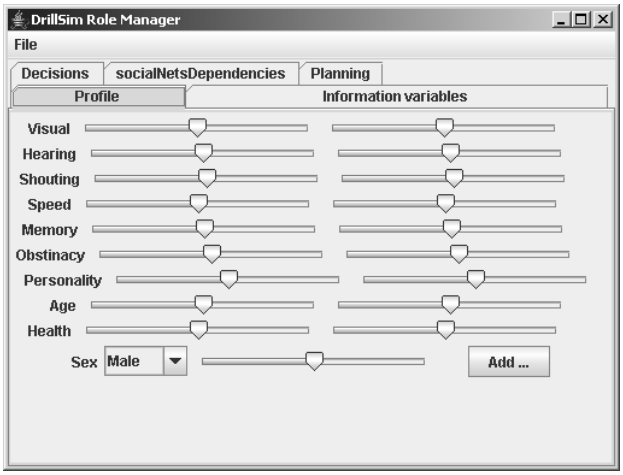

Figure 6: GUI for editing agent role.

and variance for different profile parameters, the information variables along with the software modules to compute them, the weights of the decision-making neural net, the social network dependencies, and the software modules for plan generation. The roles are then stored in XML format and loaded a posteriori from DrillSim.

6.2 Agent roles for evacuation

The experiments here realized with DrillSim are based on evacuating a floor of a building in UCI. Roles are used to represent both emergency personnel and average citizens of the public (visitors, employees). The emergency management plan for UCI defines the following three roles: zone captains, building coordinators, and floor wardens. These are regular UCI employees that take the roles of zone captains, building coordinators, and floor wardens during the event of an emergency.

In order to coordinate the evacuation of the campus or a shelter-in-place response, the campus has been divided into 13 zones with one zone captain per zone. Zone captains wear a red vest and are responsible for a zone, requesting resources, and relaying information between Emergency Op-

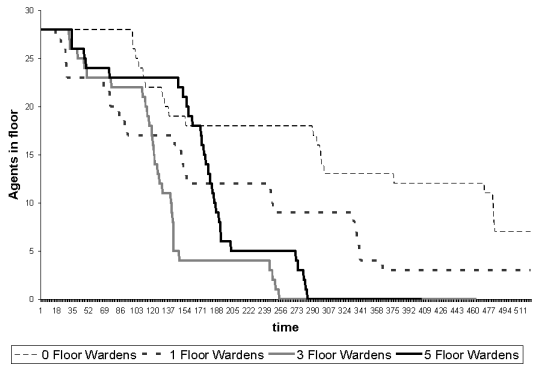

Figure 7: Impact of the new role.

eration Center (EOC) and the zone personal. Zones are further subdivided in buildings. There is a building coordinator for each building. They wear yellow vests and are responsible for evacuating their building, determining a head count, and reporting the building status. Floor Wardens (green vest) are responsible for evacuating their floor, and they assist building coordinators as needed. This way, a floor warden's decisions involve wearing his/her green vest and safety helmet, going into rooms to ask people to evacuate, calling to report people that are not evacuating, doing nothing, and exiting the floor.

The other relevant role is the evacuees. Evacuees represent average citizens of the public (visitors, employees) and they are supposed to exit the building and gather at the assembly areas. However, even though people rarely panic in disaster situations, they do not always comply with warnings [1] and they do not always exit the building.

6.3 Experiment results

With the current prototype, we realized several experiments to illustrate the addition of new roles–a key advantage of an agent-based simulation. These experiments also illustrate the capability of DrillSim as a testbed for emergency response. The results are depicted in Figure 7 and summarized as follows.

In these experiments two roles have been considered: evacuees and floor warden. We started our experiments with only one role: an evacuee. Twenty-eight evacuee agents were positioned in the floor and the fire alarm was triggered. All agents heard the fire alarm; but not all agents immediately decided to evacuate. In fact, the evacuation progressed slowly and some agents never evacuated. The weights of the decision-making neural net were set such that the presence of a hazard (e.g., fire), hearing the fire alarm, and being told by other agents to evacuate were the most important information variables that drive an agent to decide to exit a floor. The computation of these variables is based on the agent's observed world and on more subjective factors such as the memory an agent has and the reliability it associates to other agents and the fire alarm. The former was fixed for all agents. The latter was based on a social network randomly initialized. Planning involved computing steps from the agent current location to an exit. This was achieved by using an algorithm based on A* [13, 14].

On the rest of the experiments, a new role was added: the floor warden. In these experiments, we initialized the scene with also twenty-eight agents. This time, one, three, or five of the agents were assigned a floor warden role. Figure 7 shows the impact of the new role. Having one floor warden improves the evacuation, even though when the floor warden leaves the floor there are still 3 evacuees that have not evacuated the floor. Using three floor wardens improves the evacuation further. However, five floor wardens do not do better than three. For the new role, the weights of the decision-making neural net were set such that the same factors that were relevant for deciding to exit a floor are this time important for deciding to evacuate a floor instead. When an agent decides to evacuate a floor, the first thing he/she does is go to his/her office to pick up the floor warden's hat and vest and put them on. Afterwards it visits every room and asks agents to exit the building. Only when the floor warden has visited all rooms it decides to exit the floor. The reliability social network was similarly initialized as before. However, the importance that a floor warden would give to the fire alarm and the importance evacuees give to the floor warden was high.

7. CONCLUSIONS AND FUTURE WORK

Providing the right testbed for testing IT solutions in the context of disaster response is crucial for improving our response to disasters such as the World Trade Center terrorist attack. Traditionally, IT researchers would test their solutions in IT-oriented testbeds (e.g., a network simulator) that would evaluate their approaches based on IT metrics such as delay, call blocking probability, packet lost probability, and quality of the service just to name a few. However, when testing an IT solution for improving the efficiency of disaster response, we need a testbed that translates these IT metrics to disaster metrics such as evacuation time and casualties. DrillSim is such a testbed that allows plugging an IT solution to the simulation framework and obtaining disaster metrics.

This paper presented DrillSim, focusing on the Multi-agent simulator component that simulates a disaster response activity. One of the key features of such a multi-agent based simulation where agents simulate humans is that it allows the editing of existing roles and the addition of new roles on-demand. This enhances DrillSim and makes it an extensible framework where new scenarios can be created and executed on the fly. The methodology implemented in DrillSim for managing agent roles was also described and demonstrated with a series of experiments in the context of an evacuation.

Future work

A very important point for every simulator is to be able to discern to which extent it models reality. In our future work, we plan to calibrate and validate our agent behavior models. We have instrumented part of our campus with visualization, sensing, and communication infrastructure so that an activity can be captured. With this infrastructure, we can contrast a simulation of an activity in the multi-agent simulator with a real drill of such an activity. This way, we can calibrate our agent behavior model and validate it. Moreover, this infrastructure also allows us to merge the real drill with the simulation, achieving a very flexible and powerful testbed.

Our objective in DrillSim is to be able to simulate campus-wide disaster response activities involving a large amount of agents. With this goal in mind, scalability becomes another issue that needs to be tackled. A multi-agent simulation

provides us a natural way of distributing the computation, since agents can be seen as autonomous computation units for partitioning computation. However, the high rate of data queries and updates that need to be resolved in real-time poses still a challenge. Even worse, data cannot be statically partitioned based on location; in activities such as evacuation, agents would initially be distributed across several areas but as we start the simulation, most agents would be moving towards the same areas, overcrowding those areas (i.e. overloading those servers).

Apart from calibrating agent behavior and making it scalable, some of the other issues that will be tackled to move from the current prototype to the next stable DrillSim version will also involve the interaction between real people and agents and generation of a role repository.

8. ACKNOWLEDGMENTS

We would like to thank the rest of the DrillSim team for their dedication to the DrillSim project. This research has been supported by the National Science Foundation under award numbers 0331707 and 0331690.

9. REFERENCES

[1] TriNet Studies & Planning Activities in Real-Time Erthquake Early Warning. Task 2 - Lessons and Guidance from the Literature on Warning Response and Warning Systems.

[2] MicroOptical-SV-6 PC Viewer specification. http://www.microopticalcorp.com/DOCS/SV-3-6.pdf, 2003.

[3] Emergency Management Plan For the University of California, Irvine. http://www.ehs.uci.edu/em/UCIEmergencyManagementPlan rev5.htm, Jan 2004.

[4] Drillsim: Multi-agent simulator for crisis response. http://www.ics.uci.edu/ projects/drillsim/, 2005.

[5] EGRESS. http://www.aeat-safety-and-risk.com/html/egress.html, 2005.

[6] Myriad. http://www.crowddynamics.com, 2005.

[7] Responsphere. http://www.responsphere.org, 2005.

[8] Robocup-Rescue Simulation Project. http://www.rescuesystem.org/robocuprescue/, 2005.

[9] Simulex: Simulation of Occupant Evacuation. http://www.iesve.com, 2005.

[10] F. Bellifemine, A. Poggi, G. Rimassa, and P. Turci. "an object oriented framework to realize agent systems". In *WOA 2000*, May "2000".

[11] M. Deshpande. Rapid Information Dissemination. http://www.ics.uci.edu/ mayur/rapid.html, Aug 2005.

[12] A. Ghigi. Customized Dissemination in the Context of Emergencies. Master's thesis, Universita de Bologna, 2005.

[13] P. E. Hart, N. J. Nilsson, and B. Raphael. A formal basis for the heuristic determination of minimum cost paths. *IEEE transactionss on Systems Science and Cybernetics*, pages 100–107, 1968.

[14] P. E. Hart, N. J. Nilsson, and B. Raphael. Correction to "a formal basis for the heuristic determination of minimum cost paths", 1972.

[15] S. Haykin. *"Neural Networks - A Comprehensive Foundation"*. Prentice Hall, 1999.

[16] S. Jain and C. R. McLean. An Integrating Framework for Modeling and Simulation of Emergency Response. *Simulation Journal: Transactions of the Society for Modeling and Simulation International*, 2003.

[17] S. Jain and C. R. McLean. An Architecture for Modeling and Simulation of Emergency Response. *Proceedings of the 2004 IIE Conference*, 2004.

[18] S. Mehrotra, C. Butts, D. Kalashnikov, N. Venkatasubramanian, R. Rao, G. Chockalingam, R. Eguchi, B. Adams, and C. Huyck. Project rescue: Challenges in responding to the unexpected. *SPIE Journal of Electronic Imaging, Displays, and Medical Imaging*, (5304):179–192, 2004.

[19] Y. Murakami, K. Minami, T. Kawasoe, and T. Ishida. Multi-Agent Simulation for Crisis Management. *KMN*, 2002.

[20] N. Schurr, J. Marecki, M. Tambe, P. Scerri, N. Kasinadhuni, and J. Lewis. The future of disaster response: Humans working with multiagent teams using defacto. In *AAAI Spring Symposium on AI Technologies for Homeland Security*, 2005.

[21] M. Tambe, E. Bowring, H. Jung, G. Kaminka, R. Maheswaran, J. Marecki, P. Modi, R. Nair, S.Okamoto, J. Pearce, P. Paruchuri, D. Pynadath, P. Scerri, N. Schurr, and P. Varakantham. Conflicts in teamwork: Hybrids to the rescue (keynote presentation). In *AAMAS'05*, 2005.

[22] L. Thow-Yick. The basic entity model: a fundamental theoretical model of information and information processing. *Information Processing and Management*, 30(5):647–661, 1994.

[23] L. Thow-Yick. The basic entity model: a theoretical model of information processing, decision making and information systems. *Information Processing and Management*, 32(4):477–487, 1996.

[24] S. Wasserman and K. Faust. *Social Network Analysis: Methods and applications*. Cambridge University Press, 1994.