A HIGH RESOLUTION DISTRIBUTED AGENT-BASED SIMULATION ENVIRONMENT FOR LARGE-SCALE EMERGENCY RESPONSE

Glenn I. Hawe, Graham Coates, Duncan T. Wilson, Roger S. Crouch

School of Engineering and Computing Sciences, Durham University, United Kingdom

g.i.hawe@durham.ac.uk, graham.coates@durham.ac.uk, d.t.wilson@durham.ac.uk, r.s.crouch@durham.ac.uk

ABSTRACT

This paper describes the architecture of an agent-based simulation environment for large-scale emergency response. In an effort to increase "model payoff", it uses two representations for both the environment and the agents. Around each incident, where topographical information is necessary, an *operational level simulator* program models the environment using the OS MasterMap topography and Integrated Transport Network (ITN) layers. At these incident sites, first responder agents are modelled with a rich repertoire of actions. The remaining area of interest, encompassing the locations of relevant resource bases (e.g. ambulance stations, hospitals and fire stations) which are outside of the incident sites, is modelled using only transport network information by a *tactical level simulator* program. This program also simulates the tactical level agents, communicates with each operational level simulator, and provides a viewer. A separate Pre-Simulator program allows new scenarios to be set up with ease.

Keywords: agent-based simulation, emergency response

1. INTRODUCTION

The simulation of emergency scenarios is an important part of the preparedness stage of the emergency management cycle (Haddow 2010). *In silico* simulation in particular finds numerous applications within emergency preparedness and response (Jain and McLean 2003, Longo 2010).

Agent-based models are a popular way of simulating responses to large-scale emergencies (Khalil *et al* 2009). Roberts (2010) describes an agent-based model (ABM) design as:

"...one in which analogs of those real-world entities that are to be modeled are represented as software agents, or objects, *at a level of detail and resolution necessary to address the questions the model is required to answer*."

The importance of finding the appropriate level of resolution in an ABM is emphasized by Grimm *et al.* (2005):

"Finding the optimal level of resolution in a bottom-up model's structure is a fundamental problem. If a model is too simple, it neglects essential mechanisms of the real system, limiting its potential to provide understanding and testable predictions regarding the problem it addresses. If a model is too complex, its analysis will be cumbersome and likely to get bogged down in detail."

This leads to the concept of the "Medawar zone" (Grimm *et al*. 2005), a region of complexity in which the ABM is not only useful for its intended purpose, but is also structurally realistic. Together, the usefulness and structural realism of an ABM determine the "model payoff". The Medawar zone may be seen as an application of the Aristotelian notion of "the golden mean", the most desirable region between two extremes, to ABM design.

Whilst North and Macal (2007) assert that "realistic agent behaviors are the key to agent-based modeling", they also state that "properly specified agent environments are critical for correct agent operation." Thus it is important that both the agents and their environment are appropriately represented.

In the remainder of this section, we briefly discuss different representations of agents and their environment in ABMs for large-scale emergency response, and consider their relation to model payoff. We also briefly discuss the high-level software organization of existing ABMs. Then in Section 2 we propose the use of two different representations for both the environment and the agents: one for around incident sites, and another for elsewhere. In Section 3 we describe our software which uses these two representations. Section 4 provides a summary, and describes future planned developments and use of the software.

1.1. Agent representations for large-scale emergency response

The range of complexity which is possible when implementing agents, from simple production systems to cognitive architectures, is discussed by Gilbert (2007). An agent consists of a state (variables) and behavior (methods). The behavior of an agent manifests itself through the actions it decides to make. At the highest level, decision making may be *descriptive* or *normative* (Peterson 2009). Depending on which is used to implement the behavior of the agents, one of two conceptually different ABMs may arise:

1. *Descriptive*: An ABM in which the behaviors of agents are designed to mimic that of their real-life counterparts.
2. *Normative*: An ABM in which agents have behavior which is not based on reality. Instead their behavior is designed to be optimal, according to some criteria.

An example of an ABM which has agents whose behavior is defined by normative decision making is the RoboCup Rescue Simulation (RRS) (Skinner and Ramchurn 2010). The aim in the annual RRS competition is to design *ex novo* behaviors for police, fire brigade and ambulance agents, along with their control centers, to optimize an objective function which combines the health of civilians and damage to property. As Carley *et al* (2006) says, "it is concerned with designing smart algorithms, not with investigating a current human social system as it exists and designing a public policy for it." Such "smart algorithms" are often quite complex. A simple, yet crude, measure of the complexity of agent representation could be the number of lines of code taken to implement it. For example, thousands of lines of Java code were used to implement agent behaviors in RoboAkut (Akin 2010), the winning entry in the RRS 2010 competition. An interesting approach which yields agents of different complexities is described by Runka (2010). Agents are represented by decision trees, which evolve using genetic programming (Koza 1992). Different fitness functions yield different sized trees, corresponding to agents of different complexity.

This paper is concerned with agents whose behavior is designed to mimic that of their real-life counterparts. A variety of ABMs exist which implement such agents, and complexity can vary greatly. For example, the rescuer agents in the ABS SimGenis (Saoud 2006) are quite simple in their implementation (despite being described as having "perceptive and cognitive intelligence"). A set of heuristic rules determine their behaviors. The casualty agents are even simpler, having only a discrete-valued health state, the evolution of which is modelled using a Markov chain. In PLAN-C (Narzisi 2007), rescue agents are also quite simple. For example, the pseudo-code in (Mysore 2006) is only a few lines long. Although pseudo-code, it is low-level enough to suggest that the actual (Java) code would not be significantly longer. More complex are the rescue agents in the AROUND project (Chu 2009), which learn their behavior from their human counterparts through interactive sessions. Weights in a utility function, which combines multiple objectives, are adjusted so as to select actions in a manner which is most consistent with that of the human experts. State of the art cognitive architectures, such as Soar (Lehman 2006) and ACT-R (Anderson 2007), do not appear to have yet been applied to large-scale emergency response ABMs.

Just from these examples, it is evident that a range of complexities are possible for representing agents, and different research groups differ in what they deem appropriate. Müller (1999) gives eleven general guidelines for choosing the right agent architecture to apply to a specific problem, one of which is "Do not break a butterfly upon a wheel" (i.e. do not waste effort in developing complex agents when simpler agents suffice). While Sun (2006) points out that most social simulation tools "embody very simplistic agent models, not even comparable to what has been developed in cognitive architectures", Gilbert (2006) questions when cognitive architectures are needed anyway. Grimm *et al* (2005) point out that many ABMs "try only one model of decision-making and attempt to show that it leads to results compatible with a limited data set", and point out the flaws in doing so.

One characteristic which existing ABMs for large-scale emergency response do share however is that within each ABM, the representation of any particular agent is *constant*. For example, whether a firefighter agent is leaving the fire station, travelling to an incident scene, or inside the inner cordon, it is modelled using the same code, and thus at the same level of complexity, throughout the simulation.

1.2. Environment representations for large-scale emergency response

Some large-scale emergencies, such as earthquakes, may cause widespread damage to the environment, whilst others, such as terrorist bombs, may cause localized damage. Some may even cause no damage to the environment, e.g. human pandemics. Thus, the most appropriate representation for the environment is dependent on the type of large-scale emergency being simulated, and in particular the damage it causes.

For example, in the ABM EpiSimS (Del Valle 2006), which models the effect of different policies on the spread of human pandemics, only the transport network is modelled. RRS on the other hand, which simulates the response to an earthquake, models the buildings as well, as many will be damaged and need to be considered during the response effort. The representation of the environment in RRS is explored by Sato and Takahashi (2011). They found that the representation of topography influenced simulation results: when modelled at a lower resolution, buildings were found to take a longer time to burn; at a higher level of resolution, gaps between the smaller buildings prevented fire from spreading.

In the context of military simulations, which use triangulated irregular networks (TIN) to model terrain, Campbell *et al*. (1997) point out that "users naturally favour high resolution, high fidelity models because of the realism they offer, but computers that run the simulations may not be able to store and process the amount of data that is associated with these high resolution models." They propose "by identifying tactically significant (and insignificant) terrain, we can more effectively manage the TIN budget by suggesting areas of terrain that should be modelled at high and low fidelity", i.e. the use of *different resolutions* within a single model.

1.3. Software organization of ABMs for large-scale emergency response

Many ABMs are single-process programs. However, some do make use of multiple programs, and in particular distributed memory parallelism.

RRS makes use of *functional* parallelism. Different sub-simulators, each of which simulates one aspect of the earthquake response scenario (such as a fire sub-simulator, and a flood sub-simulator), run on separate processors. Using more processors enables more functionality to be modelled; however it does not allow larger areas to be simulated. As well as functional parallelism, the IDSS ABM (Koto 2003), which is also designed for earthquake response simulation, uses data parallelism: large geographical regions are split into smaller ones, which are simulated in parallel. The use of up to 34 machines is reported for modelling an area affected by an earthquake.

Data parallelism is also used in EpiSimS to split the large environment, spanning five US counties, into smaller regions, which are then simulated in parallel following a master-slave model. Del Valle *et al.* (2006) report distributing a single simulation over 106 processors.

EpiSimS also reports the use of separate programs for "enhanced pre- and post-processing" (Del Valle *et al* 2006). An "*InitializeHealth*" program allows the user to specify different probability distributions on the population being modelled. A "graphic user interface enables it to be used by nonprogrammers". This program is part of a suite of programs that are combined into a "*Set-up Wizard*". A set of scripts, which call programs such as gnuplot and Excel, are then used for carrying out post-processing on the output files generated by the simulation.

2. IMPROVING MODEL COMPLEXITY FOR LARGE-SCALE EMERGENCIES

In this section, we propose the use of more than one representation for both agents and their environment, when simulating large-scale emergency response.

2.1. Agent representation

In the vicinity of an incident site, where the casualties are (possibly trapped), first responders carry out a wide range of actions. Using the National Occupational Standards for firefighters in the U.K. (U.K. Firefighter NOS 2005) as an example, four broad groups of activities may be identified (out of nine) as being directly relevant at the time of an emergency. These four groups are highlighted in bold in Table 1.

Using the detailed descriptions of these four activity groups, twelve distinct actions may be identified, as shown in Figure 1. Of these twelve distinct actions, only one is relevant away from the actual incident sites: "driveVehicle" (the action necessary to arrive at the incident site). Thus, away from the incident sites, it is unnecessary to model the full repertoire of twelve actions for firefighters. In this regime, the behavior of firefighters reduces to movement

Table 1: Activities in the National Occupational Standards for FireFighters in the U.K.

Ref	Activity
FF1	Inform and educate your community to improve awareness of safety matters
FF2	Take responsibility for effective performance
FF3	**Save and preserve endangered life**
FF4	**Resolve operational incidents**
FF5	**Protect the environment from the effects of hazardous materials**
FF6	Support the effectiveness of operational response
FF7	Support the development of colleagues in the workplace
FF8	Contribute to safety solutions to minimize risks to your community
FF9	**Drive, manoeuvre and redeploy fire service vehicles**

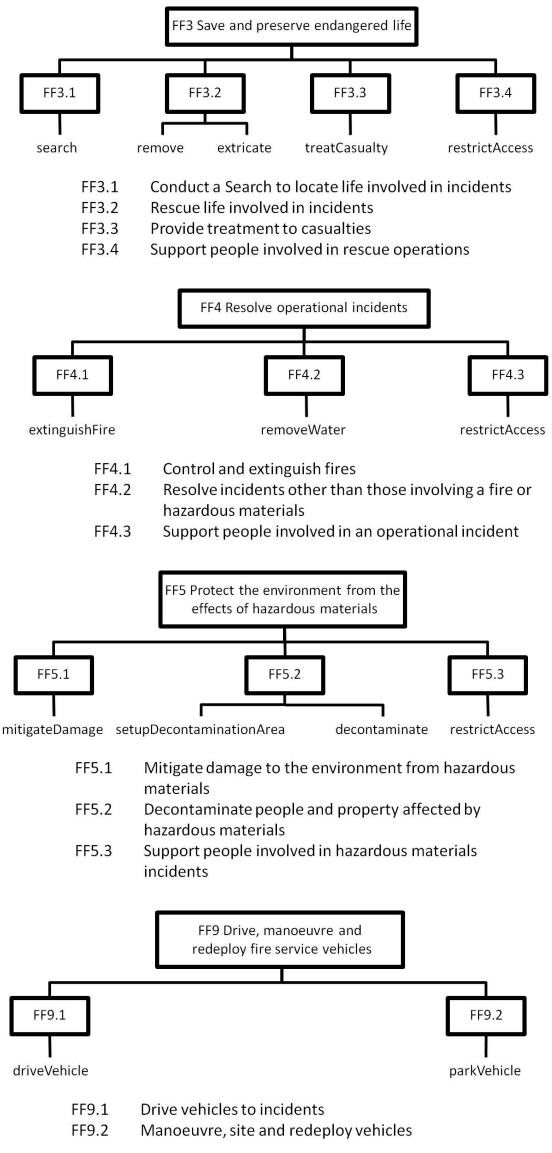


Figure 1: Identification of FireFighter agent actions

along the transport network, and the ABS is more akin to a traffic simulation. The same arguments hold for other first responder agents, such as paramedics and police.

2.2. Environment representation

The preceding discussion leads us to two different representations of the environment also. Around incident scenes, information is required (cues) in order for agents to discern which action to perform. The Ordnance Survey MasterMap (OS MasterMap 2011) topography and Integrated Transport Network (ITN) layers are used to model the topography and transport network of rectangular regions centred around each individual incident scene.

Away from the incident scenes however, as the only action which agents are performing is the action of moving, only the transport network needs to be represented. Thus, only the ITN layer is used to model the larger area which surrounds the incident sites. This area is sufficiently large to capture all the hospitals, fire stations and police stations that may be involved in resolving the incident.

Figure 2 illustrates our approach for the case of the London 2005 bombings. The locations of the bomb explosions are modelled using the topography and ITN layers. Edgware Road (Figure 2 (a)) and Liverpool Street (Figure 2 (b)) are modelled as separate 1 km^2 regions. As the Tavistock Square and King's Cross/Russell Square incidents were quite close, they are modelled together in a larger 3 km^2 (1.5 km x 2 km) region (Figure 2 (c)). To capture all the hospitals and fire stations used, the road network in the larger 60 km^2 area is modelled using the ITN layer as shown.

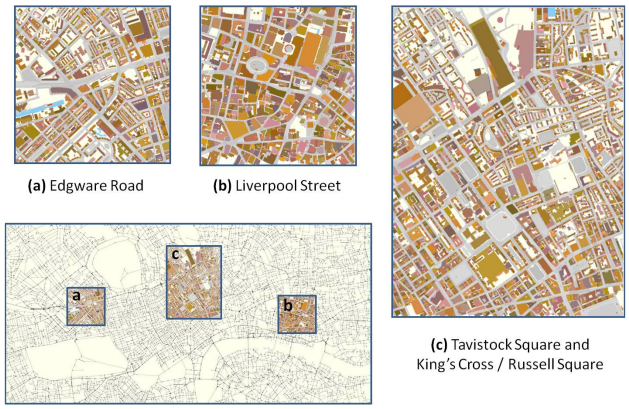


Figure 2: Two representations for the environment

3. SOFTWARE ORGANIZATION

In this section, we describe the high-level organization of the software used to set up and perform simulations, using the different representations mentioned in the previous section.

Three separate programs make up the agent-based simulation environment:

1. A Pre-Simulator program, which is used to set up the emergency to be simulated.
2. A Tactical level simulator program, which:
   a. simulates tactical level agents,
   b. simulates first responders when they are travelling along the road network, to and from incident sites,
   c. provides a viewer for the simulation,
   d. communicates with the operational level simulator programs.
3. An Operational level simulator program, one instance of which simulates one individual incident site.

Figure 3 shows how these programs are organized. The Pre-Simulator program is used to set up the scenario to be simulated. The details are written to xml files which then serve as input to the Tactical level Simulator program.

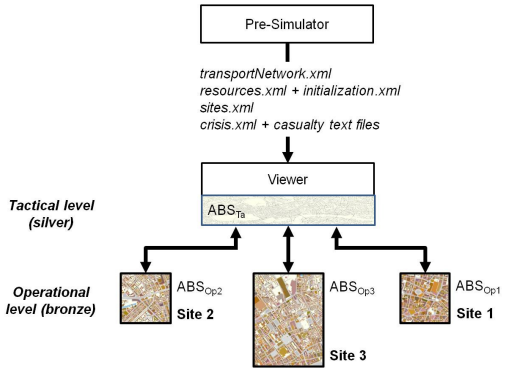


Figure 3: High-level software organization

More precisely, the Pre-Simulator is used to:

1. Specify the OS ITN file defining the transport network, and (optionally) specify a folder containing OS StreetView raster files (OS StreetView 2011) covering the same area. This information is saved to *transportNetwork.xml*.
2. Identify nodes on this transport network which represent locations of resource bases (hospitals, fire stations, ambulance stations and police stations). This information is saved to *resources.xml*.
3. Set the resources available at each resource base (e.g. the number of ambulances at each ambulance station). This information is saved to *initialization.xml*.
4. Initialize the positions and crew of each individual resource. This information is also saved to *initialization.xml*.
5. Specify the individual incident sites, and the OS MasterMap topography file(s) which define the topography in each. This information is saved to *sites.xml*.
6. Set up the incidents (including casualty information held in text files) at each incident site. This information is saved to *crisis.xml*.

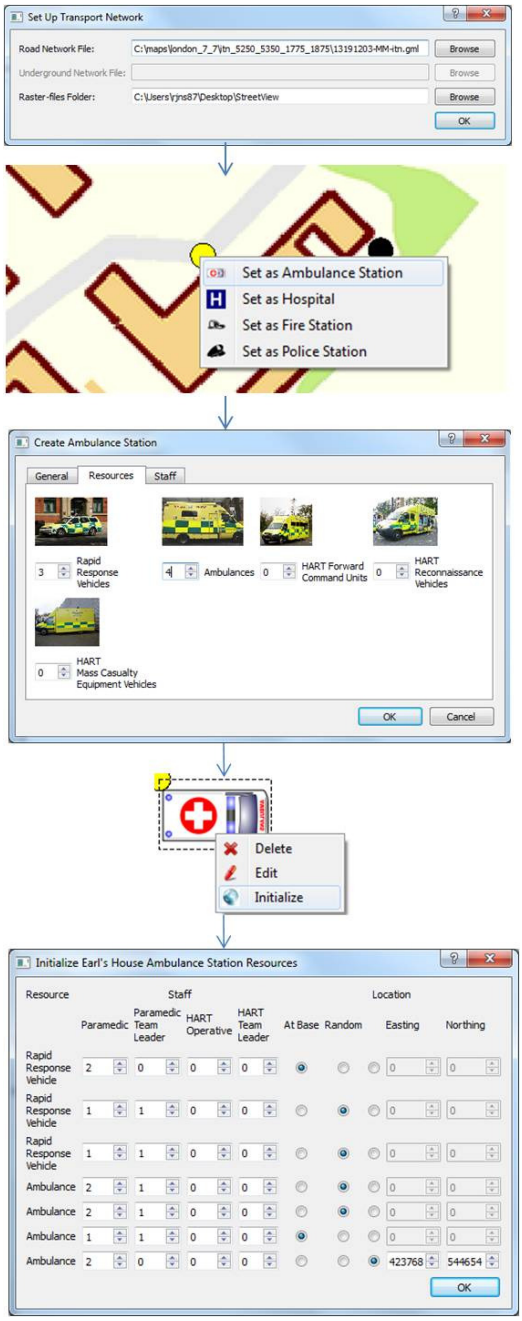


Figure 4: Setting up the transport network, identifying resource bases, and initializing resources in the Pre-Simulator.

Steps 1-4 of this process are illustrated in Figure 4, whilst steps 5-6 are illustrated in Figure 5. Note that, once the road network is loaded and displayed (Step 1), the user must select nodes as resource bases (Step 2). As it is difficult to identify the appropriate nodes using the road network alone, the first dialog in Figure 4 allows the user to (optionally) specify a folder containing OS StreetView raster files. If specified, these are superimposed onto the Pre-Simulator view which shows the transport network, allowing the user to easily identify the nodes of interest.

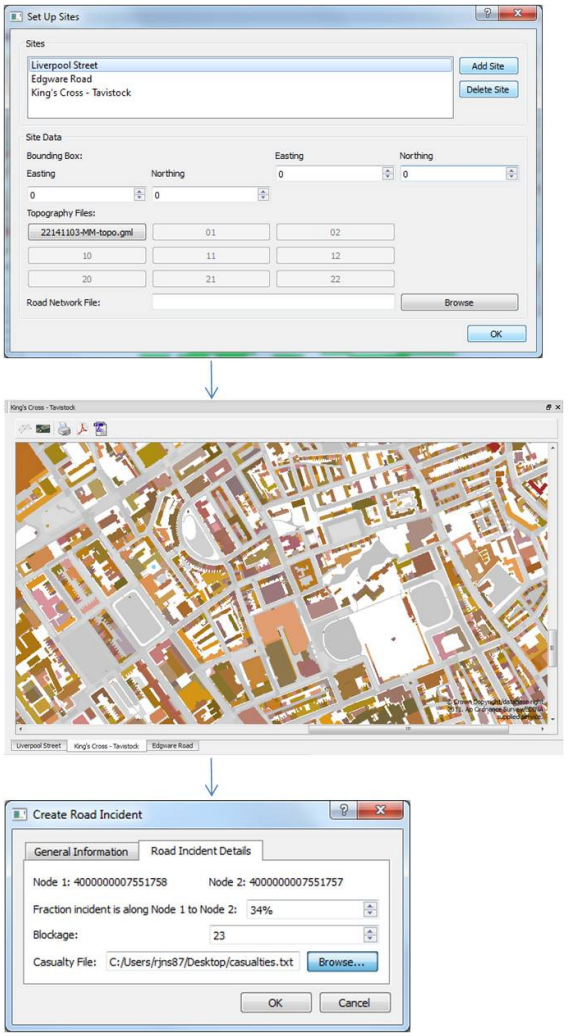


Figure 5: Setting up the incident sites, and creating incidents in the Pre-Simulator.

The tactical level simulator takes the xml input files written by the Pre-Simulator as command line parameters, and uses them to create the virtual environment and populate it with agents. It also provides a viewer of the environment, as shown in Figure 6. Each incident site has its own dockable window showing the topography of the area as defined by its OS MasterMap topography files. These windows are docked in region "**a**" in Figure 6. The area outside the incident sites is represented by the transport network, but is visualized using the OS StreetView maps. This window is labeled "**b**" in Figure 6.

The tactical level simulator also simulates the tactical level agents (in the strategic-tactical-operational command structure used for major incidents in the U.K. (LESLP 2007)). These agents are responsible for issuing a plan (usually a predetermined attendance) to the available resources. This is represented in the form of an evolving Gannt chart, which is shown in the window labeled "**c**" in Figure 6. Finally, the window labeled "**d**" in Figure 6 shows how the estimated total number of fatalities evolves with time.

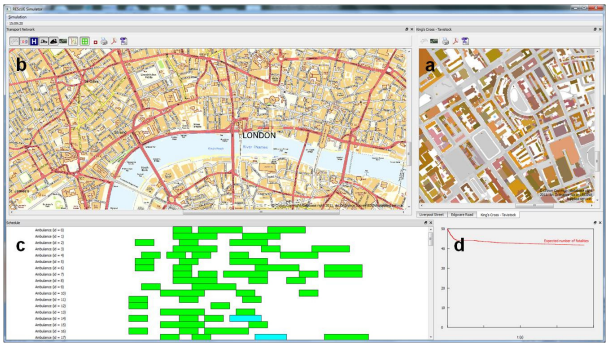


Figure 6: Screenshot of the viewer

The road network is represented as a graph in the tactical level simulator, using the Boost graph library (Siek, Lee and Lumsdaine 2002). This allows the use of Dijkstra's algorithm, to determine responder agents' paths to and from the incident sites (once issued with their part of the plan).

When an agent enters an incident site, it stops being simulated in the tactical level simulator, and starts being simulated inside the appropriate operational level simulator. Its representation changes from a basic agent which can merely move along a transport network, to a more sophisticated agent which can perceive its environment, as shown in Figure 7, and select among a wide range of actions to perform. The actions for FireFighters have already been given in Figure 1. In a similar manner, eight actions for Paramedic agents and fourteen actions for Police agents have been identified from their National Occupational Standards and Major Incident Plans.

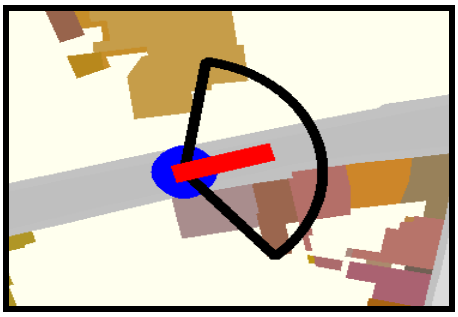
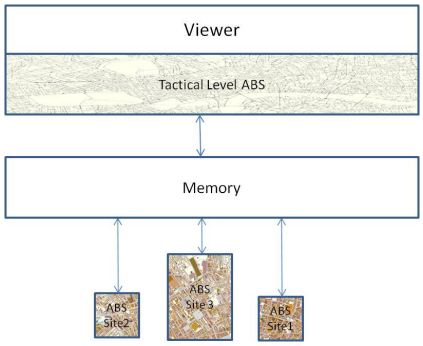
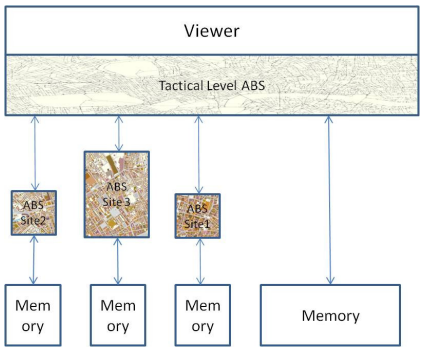


Figure 7: An agent perceiving its environment in the operational-level simulator

Finally, the tactical level simulator and operational level simulator(s) communicate with one another. The way they do this depends on whether the programs are running on the same machine or different machines, as shown in Figure 8. When on the same machine (Figure 8 (a)), they communicate using shared memory, using the QSharedMemory class from Qt (Qt 2011). When on different machines (Figure 8 (b)), they communicate using sockets, using the QTcpServer and QTcpSocket classes from Qt.



(a) Shared memory



(b) Distributed memory

Figure 8: Inter-process communication

4. SUMMARY AND FURTHER WORK

The high-level architecture of an agent-based simulation environment, designed specifically for emergency response has been described. In an effort to target the most appropriate level of model complexity, it uses two different representations for both the agents and their environment. Here we describe four further ongoing developments.

First, although agents have a rich repertoire of actions available in the operational level simulators, their action selection mechanism is still basic. Efforts are underway to model these mechanisms using naturalistic decision making (Klein 2008), in particular using a recognition-primed decision (RPD) model (Klein 2003, Warwick *et al* 2001). This has been shown to correspond well to the decision making of emergency first responders, such as firefighters (Burke and Hendry 1997, Klein *et al* 2010).

Second, parallel to the implementation of an RPD model of decision making for operational level agents, validation and verification will be carried out. Practitioners from local Emergency Planning Units, involved from the initial stages of the project, will provide face validation, whilst past case studies, such as the London 2005 bombings, will be used for retrodiction.

Third, a post-processor program will be developed to enable the analysis and understanding of simulation results.

Finally, the agent-based simulation environment is just one of two software components in the "REScUE" project (Coates *et al* 2011), being carried out at Durham University. The other component is a decision support system (DSS). The DSS has a two way communication with the tactical level simulator. It receives information about the emergency from tactical level agents as it becomes available, and uses this to generate plans for the responder agents. It then communicates these plans back to the tactical level agents, who may or may not decide to issue them to the operational level agents (who may or may not decide to adhere to the plan, depending on their most up-to-date knowledge of the emergency situation). It is a goal of the REScUE project to identify how to formulate near-optimal plans quickly, especially in the case of rapidly evolving, large-scale, unprecedented events where the practice of predetermined attendances and adhering to standard operating procedures may be far from optimal.

ACKNOWLEDGMENTS

The authors gratefully acknowledge the funding provided by the UK's EPSRC (EP/G057516/1).

Further, the authors thank practitioners from the Emergency Planning Units from Cleveland and Tyne & Wear, Co. Durham & Darlington Civil Contingencies Unit, Government Office for the North East, Fire and Rescue Services from Co. Durham & Darlington and Tyne & Wear, North East Ambulance Service, and Northumbria Police.

REFERENCES

Akin, H. L., Yilmaz, O. and Sevim, M. M., 2010. RoboAKUT 2010 Rescue Simulation League Agent Team Description. Available from: http://roborescue.sourceforge.net/2010/tdps/agents/roboakut.pdf

Anderson, J., 2007. *How Can the Human Mind Occur in the Physical Universe?* Oxford University Press.

Burke, E. and Hendry, C., 1997. Decision making on the London incident ground: an exploratory study, *Journal of Managerial Psychology*, 12 pp. 40-47.

Campbell, L., Lotwin, A., DeRico, M. M. G. and Ray, C., 1997. The Use of Artificial Intelligence in Military Simulations, *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics, Vol. 3,* pp. 2607-2612, October 12-15, Orlando, Florida.

Carley, K. M., Fridsma, D. B., Casman, E., Yahja, A., Altman, N., Chen, L.-C., Kaminsky, B. and Nave, D., 2006. BioWar: scalable agent-based model of bioattacks. *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans* 36 (2) pp. 252-265.

Coates, G., Hawe, G.I., Wilson, D.T. and Crouch, R. S., 2011. Adaptive Co-ordinated Emergency Response to Rapidly Evolving Large-Scale Unprecedented Events (REScUE). *Proceedings of the 8th International Conference on Information Systems for Crisis Response and Management – ISCRAM 2011*. May 8-11, Lisbon, Portugal.

Chu, T.-Q., Boucher, A., Drougal, A., Vo, D.-A., Nguyen, H.-P. and Zucker, J.-D., 2008. Interactive Learning of Expert Criteria for Rescue Simulations, *Lecture Notes in Artificial Intelligence,* 5347 pp. 127-138.

Del Valle, S., Kubicek, D., Mniszewski, S., Riese, J., Romero, P., Smith, J., Stroud, P. and Sydoriak, S., 2006. EpiSimS Los Angeles Case Study. Technical Report LAUR-06-0666, Los Alamos National Laboratory.

Gilbert, N., 2006. When Does Social Simulation Need Cognitive Models? In: R. Sun, ed. *Cognition and Multi-Agent Interaction*, Cambridge University Press, pp. 428-432.

Gilbert, N., 2007. Computational Social Science: Agent-based social simulation. In: D. Phan and F. Amblard, eds. *Agent-based modeling and Simulation*. Bardwell Press, Oxford, pp. 115-134.

Grimm, V., Revilla, E., Berger, U., Jeltsch, F., Mooij, W. M., Railsback, S. F., Thulke, H-H., Weiner, J., Wiegand, T. and DeAngelis, D. L., 2005. Pattern-Oriented Modeling of Agent-Based Complex Systems: Lessons from Ecology, *Science*, 310 (5750) pp. 987-991.

Haddow, G. D., Bullock, J. A. and Coppola, D. P., 2010. *Introduction to Emergency Management*, Butterworth-Heinemann.

Jain, S. and McLean, C., 2003. A Framework for Modeling and Simulation for Emergency Response, *Proceedings of the 2003 Winter Simulation Conference,* pp. 1068-1076. December 7-10, New Orleans, Louisiana, USA.

Khalil, K.M., Abdel-Aziz, M. H., Nazmy, M. T. and Salem, A. M., 2009. The Role of Artificial Intelligence Technologies in Crisis Response. *Proceedings of the 14th International Conference on Soft Computing*, pp. 293-298. June 18-20, Brno University of Technology, Czech Republic.

Klein, G., 1998. *Sources of Power: How People Make Decisions*. MIT Press.

Klein, G., 2008. Naturalistic Decision Making, *Human Factors* 50 (3), pp. 456-460.

Klein, G., Calderwood R. and Clinton-Cirocco, A., 2010. Rapid Decision Making on the Fire Ground: The Original Study Plus a Postscript, *Journal of Cognitive Engineering and Decision Making*, 4 pp. 186-209.

Koza, J. R., 1992. *Genetic Programming: On the Programming of Computers by Means of Natural Selection*, MIT Press.

Koto, T. and Takeuchi, I., 2003. A distributed disaster simulation system that integrates sub-simulators. *Proceedings of the First International Workshop on Synthetic Simulation and Robotics to Mitigate Earthquake Disaster*. July, Padova, Italy.

Lehman, J. F., Laird, J. and Rosenbloom, P., 2006. A Gentle Introduction to Soar, an Architecture for Human Cognition: 2006 Update. Available from:

http://ai.eecs.umich.edu/soar/sitemaker/docs/misc/GentleIntroduction-2006.pdf

LESLP, 2007. *London Emergency Services Liason Panel Major Incident Procedure Manual*, The Stationery Office. Available from: http://www.leslp.gov.uk/docs/Major_incident_procedure_manual_7th_ed.pdf

Longo, F., 2010. Emergency Simulation: State of the Art and Future Research Guidelines, *SCS M&S Magazine*, Vol. 1.

Müller, J., 1999. The Right Agent (Architecture) to do the Right Thing. *Lecture Notes in Artificial Intelligence*, 1555, pp. 211-225.

Mysore, V., Narzisi, G. and Mishra, B., 2006. Agent Modeling of a Sarin Attack in Manhattan. *Proceedings of the First International Workshop on Agent Technology for Disaster Management*, pp. 108-115. May 8-12, Hokkaido, Japan.

Narzisi, G., Mincer, J., Simth, S. and Mishra, B., 2007. Resilience in the Face of Disaster: Accounting for Varying Disaster Magnitudes, Resource Topologies, and (Sub) Population Distributions in the PLAN C Emergency Planning Tool. *Lecture Notes in Computer Science* 4659, pp. 433-446.

North, M. J. and Macal, C. M., 2007. *Managing Business Complexity: Discovering Strategic Solutions with Agent-Based Modeling and Simulation*. Oxford University Press.

OS MasterMap, 2011. Ordnance Survey MasterMap Available from: http://www.ordnancesurvey.co.uk/oswebsite/products/os-mastermap/index.html

OS StreetView, 2011. Ordnance Survey StreetView Available from: http://www.ordnancesurvey.co.uk/oswebsite/products/os-streetview/index.html

Qt, 2011. A Cross-Platform Application and UI Framework. Available from: http://qt.nokia.com/

Peterson, M., 2009. *An Introduction to Decision Theory*, Cambridge University Press.

Roberts, D., 2010. Distributed Agent Based Modeling, *Linux Journal*. Available from: http://www.linuxjournal.com/content/distributed-agent-based-modeling

Runka, A., 2010. *Genetic Programming for the RoboCup Rescue Simulation System*. M.S. Thesis, Brock University, Ontario.

Saoud, N. B-B., Mena, T. B., Dugdale, J., Pavard, B. and Ahmed, M. B., 2006. Assessing large scale emergency rescue plans: An agent-based approach, *The International journal of Intelligent Control Systems,* 11 (4) pp. 260-271.

Sato, K. and Takahashi, T., 2011. A Study of Map Data Influence on Disaster and Rescue Simulation's Results In: Q. Bai and N. Fukuta, ed. *Advances in Practical Multi-Agent Systems*. Springer Berlin /Heidelberg, pp. 389–402.

Siek, J. G., Lee, L-Q. and Lumsdaine, A., 2002. *The Boost Graph Library*, Addison Wesley.

Skinner, C. And Ramchurn, S., 2010. The RoboCup Rescue Simulation Platform, *Proceedings of the 9th International Conference on Autonomous Agents and Multi-Agent Systems*, pp. 1647-1648. May 10-14, Toronto, Canada.

Sun, R., 2006. Prolegomena to Integrating Cognitive Modeling and Social Simulation, In: R. Sun, ed. *Cognition and Multi-Agent Interaction*, Cambridge University Press, pp. 3-26.

U.K. Firefighter NOS, 2005. Skills for Justice - National Occupational Standards. Available from: http://www.skillsforjustice-ipds.com/nos.php

Warwick, W., McIlwaine, S., Hutton, R. and McDermott, P., 2001 Developing computational models of recognition-primed decision making, *Proceedings of the 10th conference on computer generated forces*. May 15-17, Norfolk, VA, USA.

AUTHORS BIOGRAPHY

Glenn I. Hawe is a Research Associate in the School of Engineering and Computing Sciences at Durham University, where he is currently developing an agent-based simulation environment for large-scale emergency response. He has a PhD in Electronics and Electrical Engineering from the University of Southampton, and an MPhys in Mathematics and Physics from the University of Warwick.

Graham Coates is a Senior Lecturer in the School of Engineering and Computing Sciences at Durham University. He has a PhD in Computational Engineering Design Coordination from Newcastle University, and a B.Sc. in Mathematics from Northumbria University. Recently, his research interest in coordination has been extended into the area of emergency response, and an EPSRC (UK) grant has enabled a small team to be brought together to carry out a three year study.

Duncan T. Wilson is a PhD student in the School of Engineering and Computing Sciences at Durham University, where he is currently working on a decision support system for large-scale emergency response. He has a B.Sc in Mathematics and an M.Sc in Operational Research from the University of Edinburgh. Prior to Durham he worked for the U.K. Government Operational Research Service.

Roger S. Crouch is Head of the School of Engineering and Computing Sciences at Durham University. In 1994 he took up a lectureship (then senior lectureship) at Sheffield University where he set up the Computational Mechanics Group. In 2005 he moved to Durham. His research work has focused on four areas: (i) structural integrity of nuclear reactor vessels under elevated temperature and over-pressure, (ii) computational plasticity of geomaterials, (iii) wave slamming on breakwaters and (iv) simulating reactive processes in groundwater transport.