Using Simulation for Decision Support: Lessons Learned from FireGrid

Gerhard Wickler
Artificial Intelligence Applications Institute, University of Edinburgh
G.Wickler@ed.ac.uk

George Beckett
EPCC, University of Edinburgh
G.Beckett@ed.ac.uk

Liangxiu Han
NeSC, University of Edinburgh
liangxiu.han@ed.ac.uk

Sung Han Koo
SEE, University of Edinburgh
S.Koo@ed.ac.uk

Stephen Potter
AIAI, University of Edinburgh
S.Potter@ed.ac.uk

Gavin Pringle
EPCC, University of Edinburgh
G.Pringle@ed.ac.uk

Austin Tate
AIAI, University of Edinburgh
A.Tate@ed.ac.uk

ABSTRACT

This paper describes some of the lessons learned from the FireGrid project. It starts with a brief overview of the project. The discussion of the lessons learned that follows is intended for others attempting to develop a similar system, where sensor data is used to steer a super-real time simulation in order to generate predictions that will provide decision support for emergency responders.

Keywords

data acquisition, model-based simulation, high-performance computing, decision support

INTRODUCTION

One of the aims of the FireGrid project (Berry *et al*., 2005) was to develop a next-generation software system that can support command and control for large-scale fire-fighting in the built environment. In this paper we shall describe the resulting system and an experiment involving a real fire that was used to evaluate the system. However, the main contribution of this paper is a summary of the lessons learned during the system's development. This should provide general guidance for others designing a similar emergency response system.

The development of the FireGrid system involved a number of experts from various fields, including Fire Modelling, High-Performance Computing (HPC), Grid Computing, and Artificial Intelligence (AI). Integrating the different technologies brought to the project by these experts resulted in a highly complex system.

The FireGrid Approach

For obvious reasons, fire-fighters will rarely be aware of the exact conditions that hold within a building during a fire incident and, consequently, they will be compelled to make intervention decisions based on the information provided by their senses, on their training and on their past experience of fires. Furthermore, since fire is a complex phenomenon, the interpretation and prediction of its physical manifestations is a difficult task. Advances in several technologies when taken together suggest a possible solution to the problem:

- Developments in sensor technology, and a reduction in unit cost, offer the prospect of deploying large-scale, robust and cost-effective sensor networks within buildings;
- Advances in the understanding of fire and related phenomena have resulted in sophisticated models which might be used to simulate a fire, and thus provide predictions;
- The availability of Grid infrastructure for distributed HPC and data processing suggests a platform on which these models could be run in super-real time, making their use in emergencies a practical proposition.

The FireGrid approach aims to improve the information available to fire-fighters, and, more specifically, that available to the incident commander who has overall strategy control of the response. The emphasis of the project lies firmly on the integration of existing technologies in the areas mentioned above rather than on the development of new ones. In practical terms, this would involve the loose coupling of diverse computational simulations, seeded and steered by real-time sensor data, and processed using HPC resources accessed across the Grid. An agent-based command-and-control layer using AI approaches to knowledge representation and reasoning would give end users access to the generated information. An initial architecture along these lines has been developed, implemented, and applied to a number of real fires.

The complexity of fire incidents, the difficulties of interpreting and especially predicting fire conditions within buildings, and the urgency with which fire-fighters have to make critical decisions all conspire to ensure that the FireGrid project presents an exceptional challenge for the use of computer simulations – but, equally, this challenge provides a rich source from which to draw lessons about simulations and their applicability to emergency response.

Evaluating the FireGrid System

For reasons that should be obvious, validating the FireGrid architecture presents a number of practical difficulties. The project has adopted an incremental approach, gradually increasing the number of implemented components. As a result of this process, a number of different FireGrid systems have been implemented that conform in part or in whole to the architecture outlined in figure 1, and culminating in a 'complete' system that was applied to provide real-time information during an experiment involving a real fire (under controlled conditions) that was conducted before a select audience at the Building Research Establishment (BRE), near London, in October 2008. The FireGrid system deployed in this case was intended to be used by (using UK fire service terminology) a Fire Incident Commander, or – more likely – a support officer detailed to assist the Incident Commander (HM Fire Service Inspectorate, 2002). The Incident Commander is responsible for the management of the incident, including tactical planning, coordination and resource deployment.

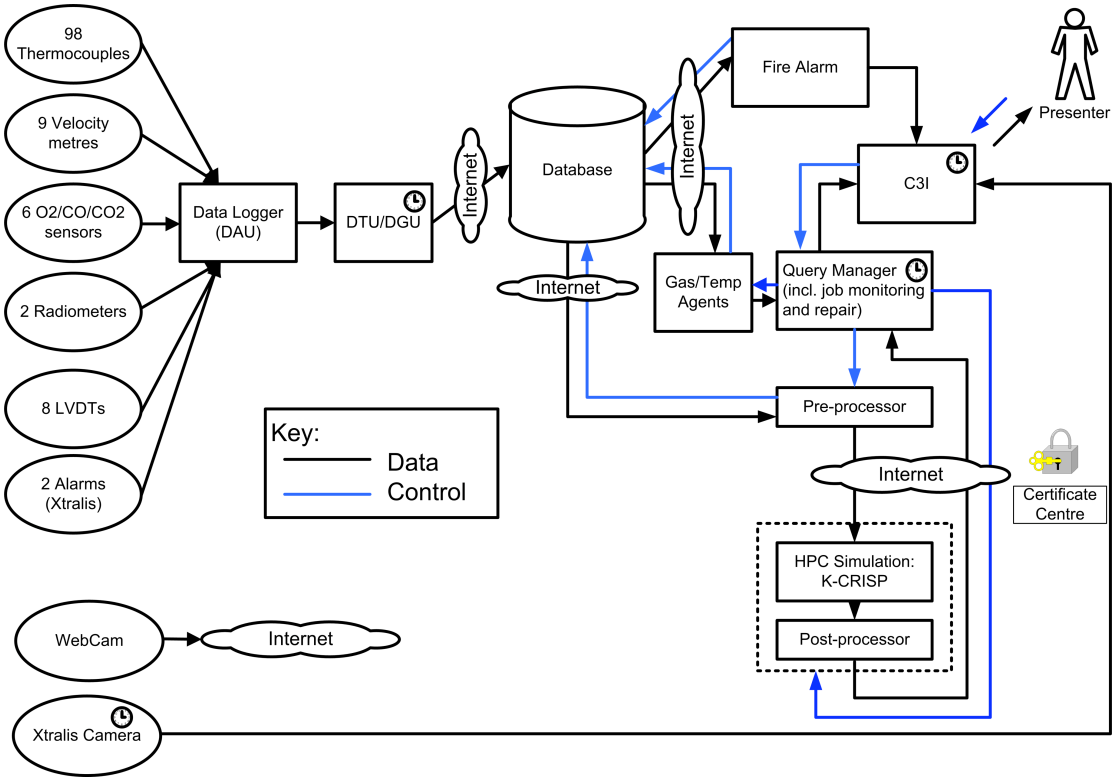

Figure 1. FireGrid architecture overview.

This experiment involved a fire initiated in a specially constructed rig, a small 3-room apartment. A total of 125 sensors placed throughout the rig measured temperatures, heat flux, gas (O_2, CO, CO_2) concentrations and deformation of structural elements. Values from each of these sensors were polled at roughly 3-second intervals, and fed to a database housed off-site. This rig and its contents were intended to produce flashover in a relatively short time (in the event the whole experiment, from ignition to manual extinguishment lasted around one hour). Flashover typically occurs when the gases produced by a fire in some enclosed space reach temperatures high

enough to ignite all combustible matter in the vicinity. The conditions required for flashover are controversial among the fire engineering community, but it usually occurs at temperatures above 500°C and, from the perspective of responders, represents a potential transition from a contained fire to an uncontrolled fire. In addition, certain structural elements of the rig were expected to deform and fail during the fire; the potential collapse of ceilings and floors is, of course, a major hazard for fire-fighters. In this experiment, the Commander's decision would be whether or not to send officers into the rig to search for occupants trapped by the fire or overcome by fumes; however, there would be no direct intervention in the actual fire.

The model/simulation that was used was K-CRISP (Fraser-Mitchell, 1994; Koo *et al*., 2008). This is a sensor-linked zone model that adopts a Monte Carlo approach, with multiple alternative scenarios for the development of the fire after ignition time generated from a set of initial conditions. The simulation was supplied with real-time sensor readings, which were used to determine the current best scenarios (that is, those found to most closely match the readings seen thus far). As well as giving some indication of the predicted evolution of the fire, the best scenarios were used to implement a feedback loop to modify the parametric space from which initial conditions for new scenarios are drawn so as to steer the simulation towards the real fire conditions.

For this FireGrid system, based on the content of the current best scenarios, K-CRISP output would be interpreted to provide predictions for the values of different state parameters (maximum temperatures, smoke layers, visibility, etc.) in each of the rooms at two-minute intervals into the future, as well predictions of any flashover or collapse events. While K-CRISP can run on a PC, its Monte Carlo approach means that generating more scenarios increases the likelihood of generating better scenarios; to try to provide quality information rapidly, the code was deployed on a HPC machine launched over the Grid.

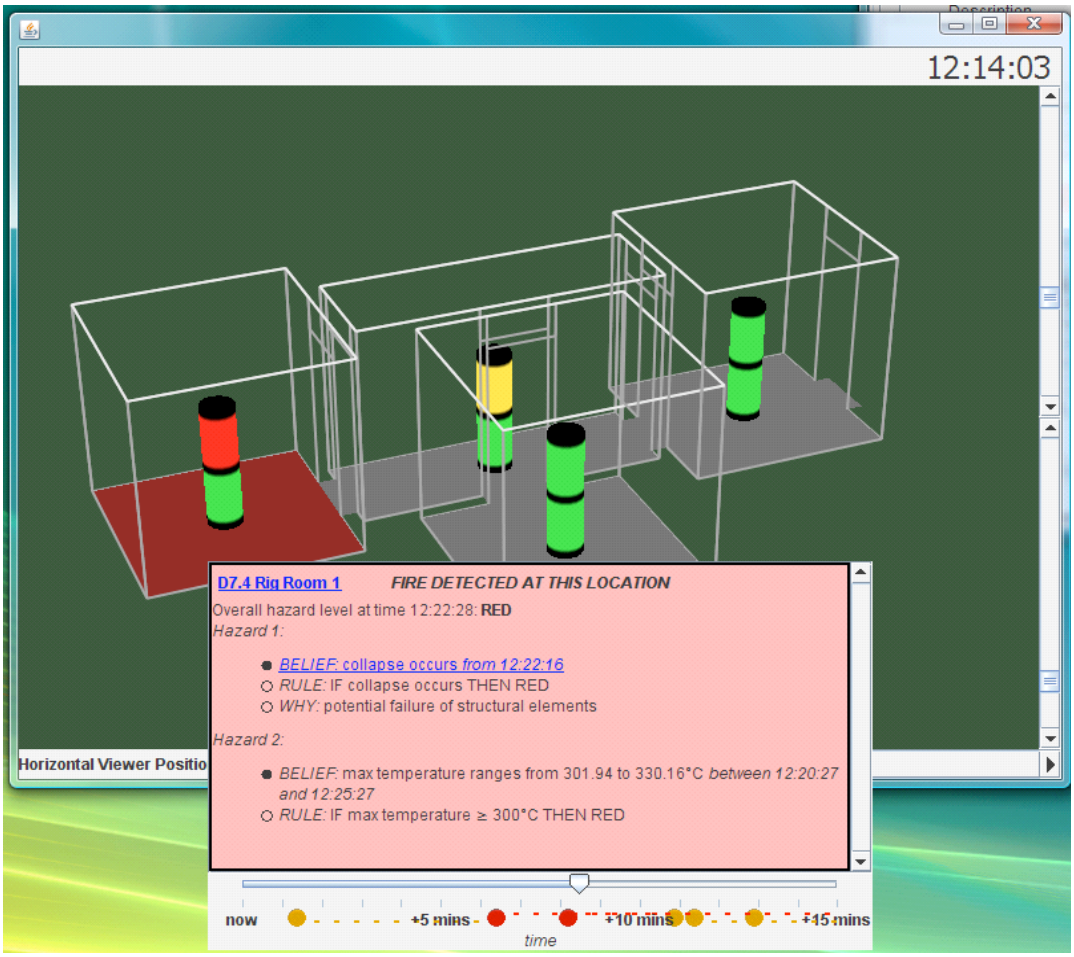

Figure 2. FireGrid system interface around 15 minutes after detection of the fire (the red floor shows the location of the fire). The user has clicked on this room to see further details. The 'traffic lights' are intended to show the current (lower colour) and projected (upper colour) hazards in each location.

This simulation was automatically launched upon detection of a fire and controlled from a command-and-control layer, which also provided the principal user interfaces onto the system, in this case, based on consultation with serving fire officers from UK Brigades, a graphical interface with a traffic-light indication of

the current and predicted 'hazard levels' in each of the rooms of the rig was developed (shown in figure 2). These hazard levels are determined by applying a set of hazard rules (expressing, for instance that flashover indicates a 'red' hazard) to the processed information supplied by the K-CRISP simulation and other models applied to the sensor data. With a member of the FireGrid team playing the role of support officer to the Incident Commander (but with a senior fire officer in the audience), the experiment was conducted and was felt, in general terms, to be a success, with the system behaving as envisaged.

LESSONS LEARNED

To analyze the lessons learned the following fictional scenario was used: the authors had just presented the FireGrid system at a conference. Someone in the audience has a model for a different kind of scenario, e.g. an oil-spill or disease spread scenario. Recognizing the potential of the FireGrid system they wonder whether a similar system to aid emergency responders could be built around their model. Note that we assume that there already exists a model on which a similar system is to be built. We also assume the applicability of the general FireGrid approach: *using the model and input from sensors to perform super-real time simulations, providing emergency responders with a glance into the future that should help making better decisions*.

The lessons learned described in this paper are given in the form of questions that should provide food for thought before the development of a similar system. While the FireGrid prototype employed a number of models that could be invoked to generate information, the models that provided the most promise were the ones that would allow the emergency responder to look into the future, i.e. models that require simulation.

The discussion of the lessons learned in this section is structured along the lines of the simplified data flow depicted in figure 3. At the heart is the simulator, which is an implementation the model. The simulator runs the model on an HPC resource accessed via the Grid. It takes dynamic sensor data as input and generates predictions as output. These predictions are task-neutral and must be interpreted to provide decision-support. For a different scenario not all the lessons may be applicable, e.g. if HPC is not required, but it is hoped that other lessons remain relevant.

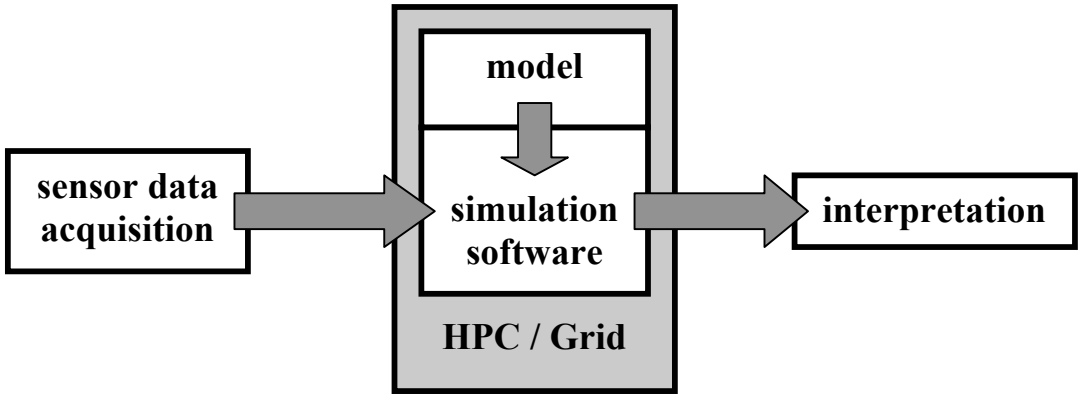

Figure 3. FireGrid abstract data flow overview.

Note that a lot of the lessons learned may appear obvious to experts in the respective fields, especially in hindsight, but they are anything but obvious to experts in different fields.

Data Acquisition from Sensors

The aim of data acquisition is to collect the raw data from available sensors such that the simulator has access to the information and can make accurate predictions about the future. Some limitations apply to the lessons discussed in this section:

- The sensors used in the experiment described above were all relatively simple, measuring a single quantity at a specific location. Sensors such as video cameras resulting in image data were not used and thus, the insights gained may be of limited applicability if the sensor data does not consist of point values.
- The sensors used in the FireGrid system were all pre-installed, meaning their exact number and location was known when the simulation was run. For other domains such as an oil spill this may not be possible and mobile ad-hoc sensor networks must be used. Not all lessons learned from FireGrid may be applicable in such a scenario.

Is all the data required by the models actually available?

If the input required by a model cannot be known during an emergency then the simulation cannot be expected to produce accurate predictions. Some fire models require detailed information about the furniture in a room including the exact locations and heat release rates over time. This information is almost impossible to obtain during an actual emergency. In addition, potential problems may arise from the number of required sensors or the locations of these. For example, at the level of individual sensors, in an office building it would be difficult to install sensors in the middle of a room – the natural location for measuring air temperatures – as they would simply be in people's way; or, at a higher level of granularity, the dispersal of sensors might not give an adequately representative data set as a whole.

Can the sensor data be channelled to and processed by the simulator?

If the required sensor readings cannot get to the simulator quickly, and/or the data is not in a usable format, the model cannot produce accurate predictions. The sensor rig that was used during the FireGrid experiments was originally set up to pull readings taken from sensors and write these to a local file only. This is common when the aim is to analyze the data after a controlled experiment. While not ideal, it was possible to construct a workable solution on top the existing rig software. However, other more complex, commercially available sensors generate data in proprietary formats, making it impossible to extract the relevant values.

At what frequency can sensor values be expected?

The frequency with which sensor readings can be taken must be sufficiently high for the data assimilation to work. This turned out not to be an issue for the FireGrid system, but was recognized as a potential problem. Furthermore, sensor readings were synchronized resulting in a complete snapshot in order to simplify subsequent processing.

Is there an ontology that describes the required sensor types?

An ontology that describes the available sensor types is helpful but not strictly necessary. In FireGrid, sensor readings were stored in a database where the underlying schema could have been more appropriate if a sensor ontology had been available. However, the lack of an ontology did not cause problems because experts were usually at hand.

Is there a reliable way of grading the sensor output?

Detecting sensors that are failing is a difficult problem that, if not addressed appropriately, will lead to poor predictions from the simulation. In FireGrid, we have developed an algorithm that takes as input a set of constraints that are expected to hold for valid sensor values. The algorithm identifies potentially failing sensors based on the constraints their readings violate (Wickler and Potter, 2009). Related to the previous point, a rich ontology might have contained failure models for the different sensor types used, which could have made possible more sophisticated techniques for sensor grading.

High-Performance Computing

The aim of using high-performance computing equipment is primarily to speed up computation. This is often important when simulators are used that require significant computational resources and, for this reason, are too slow to generate predictions in super-real time.

How fast does the simulation run on a "normal" computer?

If the linear speed-up that can be provided by HPC is not sufficient to make the simulation run in super-real time, then this simulation is not suitable for generating predictions. HPC resources achieve their computing power by providing multiple CPUs for processing. Each individual CPU may be powerful, but it is unlikely to be significantly faster than that of a state-of-the-art PC. Thus, the expected speed-up comes mostly from the number of processors that can be used in parallel, and it is important to keep in mind that the achievable speed-up is at best proportional to the number of processors used. Thus, a model that takes 24 hours on a single processor will take at least 15 minutes on 100 (equivalent) processors (and in practice may take longer).

More specifically, the authors believe that current CFD models are not suitable for super-real time simulation of large-scale fires as they tend to scale poorly and, thus, not achieve the speed-up required.

What is the execution bottleneck for the simulation?

If the execution bottleneck is not the CPU then using multiple CPUs may not address the performance problem. Any simulation will do more than only compute, i.e. data must be read in/out from file or from remote databases. When running such simulations on HPC platforms, the user must be aware that this I/O bottleneck

will cause the parallelised simulation to loose efficiency. Indeed, efficiency may also be lost by running on an excessive number of CPUs, as time may be dominated by necessary communications between these CPUs, thereby stopping the computation from proceeding.

Is the model implementation suitable for running on a (parallel) HPC resource?

If the current simulator is not already implemented for running on a parallel computer, parallelizing the simulation software will require significant effort and access to an HPC expert. Computational models are often developed by domain experts (e.g. engineers) who cannot be expected to be experts in parallel programming. As a result, an existing implementation may be highly serial. This may be made even more difficult if the original implementation is inherently serial and/or in a programming language that is not suitable for HPC platforms (use of bookkeeping features of C++, codes relying on non-standard C/Fortran, etc).

The simulator underlying the FireGrid system was a task farm, where each slave executed a serial Fortran code. Each of the tasks simulated the entire fire scenario and many tasks were run concurrently, where each task employed different values for unknown model inputs. This method of ensemble computing is applicable if a limited number of model inputs are unknown and a single run of the simulator is sufficiently fast.

Can the existing implementation be compiled on the HPC resource?

If the simulation code has not yet been compiled on the HPC resource on which it must run, this will require effort. Firstly, it requires the availability of an appropriate compiler for the specific resource. Despite the fact that many programming languages are (ostensibly) standardized, it is often quite difficult to transfer code from one compiler to another, and it may take some work to get code to run properly on a platform different from the one it was developed on. Furthermore, if there are specific libraries required, these may not be available for the HPC resource.

How quickly do simulators need to start running?

Using remote HPC resources may lead to a delay of up to several minutes before the simulation will start due to the HPC resource's batch system.

The Grid

The aim of using a computational Grid in FireGrid is simply to provide access to HPC resources. Such supercomputers are expensive to purchase and maintain. For a given emergency response task they might be rarely used. Thus, it makes sense to share HPC resources and the Grid is one way of gaining on-demand access to such general purpose computing power.

Note that computing Grids in general offer far greater functionality than what is exploited in FireGrid. Grid computing is:

> "… a form of distributed computing whereby a "super and virtual computer" is composed of a cluster of networked, loosely coupled computers, acting in concert to perform very large tasks. […] What distinguishes grid computing from conventional cluster computing systems is that grids tend to be more loosely coupled, heterogeneous, and geographically dispersed.[1]"

It should be clear from the discussion related to HPC that the running of a special-purpose simulator on a set of *heterogeneous* computers will be work intensive as the code will need to be compiled, installed, and subsequently maintained on every computer it is meant to run on. A simulator based on some standard software that is widely available may be able to better exploit the Grid, but we do not know what this software might be.

How many (heterogeneous) computing resources should be available through the Grid?

Given the issues discussed in the previous section, it is a good idea to start with a small number of relatively similar resources. In FireGrid one default and one spare resource were used. One of the biggest advantages of the Grid is that it provides unified access to computing resources that may be based on quite different hardware and software (operating systems). This should make it simple to switch from one Grid resource to another in case one resource is not available during an emergency.

Is there a Grid expert available?

A Grid expert will almost certainly be necessary to develop a FireGrid-like system. The Grid is a relatively new layer of infrastructure that is running on computational resources. To access the Grid it is necessary to install a

[1] http://en.wikipedia.org/wiki/Grid_computing

Grid certificate and run a Grid client on the emergency responder's computer, as well as open firewall(s) for Grid access, especially for militarised zones. These tasks are time-consuming and difficult to debug.

Can the simulator be adapted to the resource it is running on?

Information about the specific computing resource employed is necessary to set parameters of the simulation such that predictions are available when needed. The simulator will run at different speeds depending on input parameters and where exactly it is being run. Decisions often need to be taken at fixed times. Setting parameters for the simulation to give results in time requires information about the underlying computing resource. While the unified interface of the Grid simplifies access to resources, it may also hide detail that is necessary to set parameters for the simulation. The solution adopted for FireGrid was to have an HPC expert port and optimise the code *a priori* on all platforms of interest, and a Grid expert create and maintain scripts for running the simulations on the different HPC resources.

Models and Simulation

The aim of using models is to generate information that is required by the decision makers but is not directly available either from the sensor data or from elsewhere. As mentioned before, the predictions generated from simulations afford the emergency responders a glance into the future, which is perhaps the most appealing aspect of the FireGrid system from an end user perspective, which is why we shall restrict the discussion to predictive models.

Have the models ever been used to generate predictions?

If a model has only been used post-hoc, i.e. to verify whether the model can accurately re-generate some real events, the predictive capabilities of the model should be questioned. The post-hoc use of models is typical in a research context, where often only relatively few experiments can be used to validate a model. In the case of fire modelling, it is obviously not possible to burn down a number of large buildings to assess the accuracy and limitations of a model. The model used in the FireGrid system turned out to be capable of predicting the development of a fire up to the point when flashover occurs, but not beyond that stage.

Can the simulation be "calibrated on the fly"?

If the simulation cannot assimilate sensor data while it is running, the results are likely to diverge from reality and therefore not be useful for decision makers. Mathematical models, however good they may be in principle, almost always need to be calibrated to accurately reflect reality, a process usually performed post-hoc. In an emergency response scenario the model needs to be calibrated as the incident unfolds. That is, model parameters need to be adjusted using sensor readings while the model is still running, a process called data assimilation.

In the FireGrid system, the sensor readings were used to identify the scenario generated by the parameter sweep that best fits the actual readings. The parameters used for this best-fitting scenario were then used to concentrate the parameter sweep in that area. If convergence occurs, this can be seen as evidence that calibration was successful, but this is not necessarily the case.

Can the model be used to address "what-if" questions?

If the model can only be used to simulate reality as it "naturally" unfolds, it will be of limited use to an emergency responder considering one or more possible courses of action. Emergency responders are interested in intervening in the situation and will try to change the natural course of events with their actions. Thus, it would be most useful if they could use the model to run a simulation of multiple courses of action and compare the results. This leads to the requirement of running a model, and giving it the initial state as input as well as set of planned actions. The model could then be used to answer the "what-if" question: What if these actions are used to tackle the emergency? Even if the model cannot take future actions into account, it should at least take into account any actions emergency responders are already implementing; otherwise predictions might be very inaccurate. This question was not addressed within FireGrid.

Can the model assess the accuracy of its own results?

Ideally, the predictions generated by the simulation should be annotated with some value indicating the system's confidence in these predictions. In FireGrid this is done by counting the number of scenarios generated in which a given prediction (e.g. flashover temperature has been reached) holds. However, it is not immediately clear what this percentage means to an emergency responder, an issue discussed in the next section.

Intelligent Decision Support

The aim of information interpretation is to take the output from the various models available and transform this information in a way that makes it directly relevant to the decisions an emergency responder is deliberating.

Note that this depends on the current situation and is therefore dynamic, i.e. model outputs may require different interpretations for different users in different situations.

Are the model outputs in terms the emergency responders can understand?

If the outputs of a model are not context sensitive or intelligible to the decision maker they will need to be interpreted. Models are often the result of academic work and the outputs they produce are those that interest their creators. The type of information that is interesting for a fire engineer is not necessarily the type of information an emergency responder is interested in. One way of addressing this issue is to include an interpretation module into the system. Another option is to have a domain expert interpret the model results for the decision maker, which requires the presence of such an expert during an emergency.

In FireGrid we have developed an AI system to interpret the result of the simulation for the user. A rule-based system was used to extract information from the scenario generated by the simulator and transform it into knowledge relevant for the decision maker (Wickler and Potter, 2009). There are a number of rule interpreters available that can be used to generate consequences given some initial facts. However, certain features of the domain can make this reasoning process significantly more difficult to the point that existing rule engines can no longer be applied. Sophisticated reasoning about time and space belongs to this category of features. An AI expert is necessary to integrate the relevant techniques into a functional interpretation module.

Is there a set of standard operating procedures available?

Standard operating procedures are a useful source of information requirements. Usually they will have been developed by experts in the field and reflect best practices that have been tested in similar circumstances. Standard operating procedures usually list the different ways in which a given task can be accomplished, and indicate which way should work best under which conditions. Such conditions then represent the kind of information in which a decision maker should be interested during an emergency.

Can uncertainty about the model results be conveyed to the user in a useful way?

As mentioned above, the outputs of the simulation can only be accurate to a certain degree. However, presenting this uncertainty to the user as a probability (percentage) is usually not very useful as it can be hard to interpret such values. On the other hand, leaving out the uncertainty may give the wrong impression to the user and result in suboptimal decisions, potentially putting lives at risk.

CONCLUSIONS

The main contribution of this paper is a discussion of some lessons learned during the development of the FireGrid system. The aim of describing these lessons was to assist others considering the development of a similar system, pointing out potential pitfalls and problems. Hopefully this will lead to a new generation of emergency response systems that will provide decision makers with better information based super-real time simulations.

ACKNOWLEDGMENTS

The work reported in this paper has formed part of the FireGrid project. This project is co-funded by the UK Technology Strategy Board's Collaborative Research and Development programme, following an open competition. The authors acknowledge the contribution of their colleagues in the FireGrid project. The University of Edinburgh, project partners and project funding agencies are authorized to reproduce and distribute reprints and on-line copies for their purposes notwithstanding any copyright annotation hereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of other parties.

REFERENCES

1. Berry, D., Usmani, A., Torero, J., Tate, A., McLaughlin, S., Potter, S., Trew, A., Baxter, R., Bull, M. and Atkinson, M. (2005) FireGrid: Integrated Emergency Response and Fire Safety Engineering for the Future Built Environment, *UK e-Science Programme All Hands Meeting (AHM-2005)*, Nottingham, UK, Sept. 19-22. See also http://www.firegrid.org/.
2. Cowlard A., Jahn W., Empis CA., Rein, G., and Torero JL. (2009) Sensor Assisted Fire Fighting, *Fire Technology* (in press), 2009.
3. Fraser-Mitchell, J. N. (1994) An object-oriented simulation (CRISP II) for fire risk assessment. *Fire Safety Science*, 4:793–804.

4. HM Fire Service Inspectorate (2002) *Fire Service Manual*, Volume 2 Fire Service Operations, Incident Command, HM Fire Services Inspectorate Publications, London: The Stationary Office.
5. Koo, S.H., Fraser-Mitchell, J., Upadhyay, R. and Welch, S. (2008). Sensor-linked fire simulation using a Monte-Carlo approach, *Proc. 9th IAFSS International Symposium on Fire Safety Science*, Karlsruhe, Germany, September 2008.
6. Potter, S. and Wickler G. (2008) Model-Based Query Systems for Emergency Response, *Proc. 5th Int. Conf. on Information Systems for Crisis Response and Management (ISCRAM 2008)*, F. Fiedrich and B. Van de Walle (eds.), Washington DC, USA, May 2008.
7. Upadhyay, R., Pringle, G., Beckett, G., Potter, S., Han, L., Welch, S., Usmani, A., and Torero, J. (2008) An Architecture for an Integrated Fire Emergency Response System for the Built Environment, *Proc. 9th IAFSS International Symposium on Fire Safety Science*, Karlsruhe, Germany, September 2008.
8. Wickler, G. and Potter, S. (2009) Information-gathering: From sensor data to decision support in three simple steps. In: *Intelligent Decision Technologies*, vol. 3.