Information-Gathering: From Sensor Data to Decision Support in Three Simple Steps

Gerhard Wickler and Stephen Potter

{g.wickler|s.potter}@ed.ac.uk

Artificial Intelligence Applications Institute,
School of Informatics, University of Edinburgh
Appleton Tower, Crichton Street, Edinburgh, EH8 9LE
Scotland, UK
Tel/Fax: +44 131 650 2732/6513

Abstract

In this paper we describe the information-gathering problem which can be characterized as transforming large amounts of data obtained from sensors into accurate, concise, timely and meaningful information that can be used by decision makers faced with a specific task and a number of options for performing that task. The approach to this information-gathering problem as described here consists of three phases: data validation, data aggregation and abstraction, and information interpretation. Each of these phases will be described in general, and for each of these phases we describe techniques that are reasonably generic to be applicable in many domains, but domain specific knowledge will of course always be needed too.

Keywords

Command-and-Control, Artificial Intelligence, Modelling, Information Systems

1. Introduction

Effective action in any domain is necessarily founded on the availability of effective, timely and accurate decision-making information. In complex domains, however, the acquisition of such information is itself a complex task, and one that requires the application of specialised information-gathering processes performed by information-gathering agencies. This need becomes particularly evident when the domain is highly dynamic, and raw data and the information derived from it must be filtered to extract just that information required to make the decisions the situation demands.

At a geopolitical level, in military contexts and even in the business world the need for such information-gathering agencies (where they are termed ‘intelligence agencies’ or ‘intelligence services’) has long been recognised; at a smaller scale, or in civilian contexts, while their need is no less pressing, lack of resources often leads to inadequate provision of these services. To redress this balance, the authors have, in the FireGrid project [1], been exploring the use of new technologies to provide (semi-) automated decision-making support for fire-fighters tackling an emergency incident within a complex, sensitive or otherwise high-value building.

In the next section we will give a brief overview of the FireGrid system and the problem it addresses. We will then go on to describe the information-gathering problem as encountered in FireGrid in more detail. The contribution of this paper is the three phase approach to the information-gathering described next: data validation, abstraction and interpretation. We shall describe general techniques that we expect to be applicable in the respective phases and illustrate these with examples from fire experiments conducted in FireGrid.

2. FireGird: Emergency Response Support for Complex Fires

The FireGrid project [1] represents a farsighted attempt to harness recent advances in a number of disparate fields for the express purpose of assisting responders to tackle emergency incidents, in particular (but not exclusively so), complex building fires. Currently fire-fighters, when they arrive on the site of an incident, generally have to rely on the information provided by their own senses, any information that can be provided by evacuated occupants of the building in question and their experiences of previous fires in order to decide on an intervention course. In the UK, the initial decision is one of choosing the appropriate tactical mode for tackling the fire: this may be either *offensive* or *defensive* [2]. The former often involves sending fire-fighters into the building, a decision that is taken if the potential benefits are felt to outweigh the risks – for example, if people are thought to be trapped within the building and fire-fighters are felt to have a reasonable chance of rescuing them while being exposed to an acceptable level of risk. Defensive mode, on the other hand, is adopted when the trade-off of the potential benefit against the likely risk of offensive mode is not thought favourable (or, in some situations, where the current lack of information means that the benefits or risks cannot yet be assessed).

Hence, the intervention decision can be based on incomplete or faulty information; in particular, for large-scale and complex buildings, fire-fighters are rarely aware of the exact conditions within the building. Moreover, the lack of experience of complex fires that many fire-fighters have (simply because such fires occur relatively rarely), can mean that, even when available, information is misinterpreted, and fire-fighters are placed in danger.

Obviously this is an unsatisfactory state of affairs. However, recent advances in three areas of technology, when exploited together, suggest a possible solution to this problem:

- Developments in sensor technology, along with 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 computer models which might be used to interpret sensor data;

- The availability of Grid resources and infrastructure promises to enable these (usually extremely time- and resource-hungry) models to be run in real-time, making their use in emergencies a practical proposition.

The FireGrid vision is to combine these technologies in a system, underpinned by concepts and techniques drawn from Artificial Intelligence, that essentially provides an ‘intelligence service’ for fire-fighters.

2.1 The FireGrid Software Architecture

The architecture of the FireGrid system is presented from a command and control (C2) perspective here as this is the aspect that is intended to directly support decision makers. The role of the C2 layer of a FireGrid system is, in brief, to provide a means for users to interact with the system and steer it towards achieving their goal – which, in a deployed system, would be to help with the safe and successful management of fire incidents in the building in question. The unique aspect of a FireGrid system is the capture of ‘live’ sensor data from the building and the use of this data by models to interpret the status and projected course of the incident for emergency responders. Figure 1 shows the components of the C2 layer.

There are two primary human interfaces onto the C2 layer, namely the *Building C2* (BC2) interface, and the *e-Response C2* (eRC2) interface. The role of these interfaces is to provide their human users with information about the current state of the system (and hence about the state of any incident and of the response to it), and to assist users to interact with system components to acquire additional information or actuate some response.

The two types of C2 interfaces differ in their applicability, coverage and scope. A BC2 interface is specific to a particular FireGrid system, and is tailored towards that system and the building it relates to. Its projected user is someone who has responsibility for monitoring the state of the building in question and, in the event of an incident, for instigating initial response activities (such evacuating the building), but would not be expected to tackle anything but the most trivial of fires.

The eRC2 interface, on the other hand, contains knowledge of agents (such as fire-fighters) and resources (such as standard operating procedures) that are external to any specific FireGrid system, and which may be required when the response to incident has to be escalated beyond the local (that is, BC2) level. The eRC2 interface is intended to be installed on, for instance, the computer system in an emergency response command vehicle; when the vehicle arrives at the site of the incident, it ‘taps into’ the *in situ*

FireGrid system to access and request information about the incident. The projected user of the eRC2 interface is (using UK terminology) a Fire Incident Commander, or – more likely – a Support Officer detailed to assist the Incident Commander. The Incident Commander is responsible for the management of the incident, including tactical planning, coordination and resource deployment [2].

2.2 Decision Support using Intelligent Agents

Underlying the C2 interface components in FireGrid are intelligent agents that are based on the I-X framework [4][7]. I-X provides a generic systems architecture (and tool suite) for multi-agent process support, structured upon an abstract activity-centred ontology for expressing information and communications within the system. While this approach has its foundations in work in AI planning, it is intended for use in systems of collaborating human and computer agents.

The “intelligence” of an I-X agent stems from a set of standard operating procedures encoded by domain experts. These procedures correspond to the structures called *methods* in the planning literature [3]. Methods formally describe how a specific task can be broken down into sub-tasks. The definition of a method consists of four main parts: *task pattern*, *name*, *constraints* and *network*.

The task pattern of a method is used for matching methods to items in the activity list, the “to-do” list of the user that describes the current problems that must be addressed. The name can be used to refer to the method and thus to distinguish the different methods available to address the same task. Methods applicable to the same task are options which require a decision from the user. The network contains the list of sub-tasks that will be added as activities when the method is chosen.

The constraints are used to decide whether a method is applicable in the current context. Hence, the constraints provide a formal way for the experts in the field to stipulate the conditions under which a method/standard operating procedure is valid, and as such they form part of the information about the environment and state of the task required by the decision maker in order to decide how to tackle a given problem. Furthermore, since these constraints have been provided by an expert we can assume that they will be at a level of abstraction that is relevant and meaningful to the task in hand. Thus, we shall assume that such constraints describe the kind of information that is the objective of information-gathering – in other words, it is the target output of the information-gathering process.

3. The Information-Gathering Problem

In our approach, the information that needs to be gathered is described by the constraints associated with different methods in an I-X agent's library of standard operating procedures. In FireGrid one source of the information that is available about the environment comes in the form of sensor readings that provide large amounts of dynamically changing data. In this case, then, the information-gathering process needs to bridge this gap, taking as input this sensor data and generating as output the information relevant to a decision maker.

The data generated by the sensors and the information required by the decision maker can differ in a number of ways that need to be addressed by the information-gathering process; specifically they can differ in terms of:

- **Accuracy**: decision makers require accurate information whereas sensors might fail and result in false readings. This is further complicated in a situation such as a fire incident in which sensors can be destroyed while information-gathering is taking place.

- **Concision**: decision makers require concise information whereas large numbers of sensors can result in large numbers of sensor readings that directly reflect the quantities the sensors are measuring.

- **Timeliness**: decision makers would like to base their decisions on the latest state of the environment; however, sensors operate only at a specific frequency, meaning that the data they provide might be out of date.

- **Meaningfulness**: decision makers usually require information that corresponds to some generalised and task-specific interpretation of the current state of the environment whereas sensors provide objective ‘point-data’ that effectively constitute task-neutral, specific and localised truths.

4. From Sensor Data to Decision Support

In this section, we will describe our view of this information-gathering process, and our approach to the information-gathering problem. The process will be described in the abstract, regardless of context or domain, or even whether the actors in this process are human or automated. We shall illustrate this approach through examples drawn from the FireGrid project and experiments conducted. In our approach the information-gathering process can be divided into three phases as follows:

- **Phase 1: Data validation**

  Observations or facts are collected; and these must be verified. In the context of FireGrid these ‘observations’ are provided by the individual sensors within the building; since sensor data may be noisy, and faulty and failed sensors can provide incorrect readings (which will invariably happen in devastating fires as sensors are destroyed), these observations need to be ratified before they are passed onto the next phase. Similarly, in other contexts, ‘facts’ may be provided by humans, and attempts such should be made to validate or corroborate these before accepting their veracity.

- **Phase 2: Data aggregation and abstraction**

  Usually – but not always – the observations/facts provided by phase 1 will need to be processed to provide information. In the context of FireGrid, the task during this phase is to condense the sheer quantity of sensor data into abstractions that are meaningful in the context. In computational terms, this is achieved by applying different analytical algorithms to the data; these algorithms can range from the simple – for example, selecting the maximum value from a contemporaneous set of readings from a co-located set of sensors – through more advanced data fusion algorithms, up to the highly complex – in the case of FireGrid, the application of models of the physics of fire spread to make predictions about the course of the fire. This phase is necessary when there is a discrepancy between the content or expression of the data from phase 1 and the content or expression of the information required for decision support in phase 3; in complex or open systems, this will invariably be the case. The algorithms applied at this phase will typically be domain-dependent, and may well be formulated with the general task (such as emergency response) in mind, but will be independent (and oblivious) of the wider context for which the information is required.

- **Phase 3: Information interpretation**

  The information derived from phase 2 (or, where appropriate, directly from phase 1) must be further interpreted in the context of the state of the current activity and the available choices in order to select viable courses of action. Hence the information needs to be interpreted (and presented) in such a fashion as its relevance to the decision-maker and the task in hand is readily apparent, and which takes into account his/her particular knowledge and capabilities along with the specific circumstances and pressures under which he/she operates. In FireGrid, we support this task by providing a custom interface and underlying reasoning engine tailored to the needs of a particular decision-maker, representing graphically the afflicted building and onto which we superimpose relevant information.

In any application, we would expect these phases to be applied cyclically (or, more likely, concurrently) until such time as all the goals of the decision-makers have been achieved. For our purposes we assume that the communication of data and information from phase to phase is noise- and error-free; obviously to make such an assumption in any real situation would be dangerous. The last phase should be seamlessly integrated with the I-X system and its operating methodology as described above.

4.1 Phase 1: Data Validation using Constraint Networks

The first phase involves the gathering of data about the situation. For building fire incidents, fire-fighters will be continually taking in direct sensory perceptions of the incident, collecting and cross-checking statements by eye-witnesses, and so on. To augment this, a FireGrid system is intended to provide information about the state of the incident based on readings supplied periodically by a number of sensors of different types located within the building. These sensors can include, for instance, fire alarms, smoke detectors, thermocouples (for reading temperatures), CO and CO_2 meters. Typically, these sensors will be polled in batch mode periodically by one or more *data loggers*, physical devices with which groups of sensors have some communications link; in this device and its accompanying software, the signals produced by the sensors will be converted into their corresponding quantities (so, for instance, the voltages read from the thermocouples will be converted into degrees Celsius). In modern systems, these steps are automatic, and at this point, these data values can be accessed and stored in a database. To do this, however, would be to mistakenly assume that all data values are correct. Since sensors (or their lines of communication) can be noisy or can fail because of manufacturing flaws or the extremes of the fire incident itself, the data first needs to be verified: this is the *sensor grading* task.

In this section we will describe an algorithm that can be used for real-time sensor grading. The expected input is a batch of ‘raw’ sensor readings at some specific point in time. The algorithm then uses previously asserted constraints on the values of the different sensors to find a set (which may be empty) of sensors that are considered to have failed or to be failing at that point in time (a sensor might cease to be reliable when conditions move outside its normal operating range; however, it is not necessarily the case that any subsequent reading from that sensor from that time onwards will be unreliable). Thus, the output of the algorithm is a binary value for each sensor, indicating whether the sensor reading is considered correct or not.

To illustrate the algorithm we shall use a controlled fire experiment that was conducted at the University of Edinburgh in February 2008. In this experiment a tray containing a fuel source was placed in the middle of a small room. Around this fire were placed four vertical ‘trees’, each equipped with 10 thermocouples, allowing the measurement of the gas temperatures at different heights. The layout of the experiment and the sensors is shown in Figure 2.

Thermocouple readings were taken at a rate of 2Hz and written to a file and database. The initial grading was performed by running some preliminary tests before the fire experiment and having a fire-engineering expert look at the resulting data. The expert then identified four of the 40 sensors as failing and they were excluded manually from further processing – in other words, in advance of the experiment these sensors were flagged so that their values would be ignored by the computational models that were to be applied to the data during the experiment. Clearly, this is not a satisfactory approach for a system that can support decisions during real fires, partly because it is not feasible to expect that sensors have been thoroughly tested just prior to a fire breaking out, and also because one should not assume sensors will remain undamaged during a fire.

The ungraded output from all the sensors at one time point consists of 40 floating point values representing temperatures in degrees Celsius. Table 1 illustrates the output from the sensors, the data arranged in columns corresponding to each sensor tree, and with each sensor identified by the unique label shown. Each tree contains 10 thermocouples, with the first sensor located 40cm above the ground and the others spaced evenly above this to the ceiling of the room (note how the values of the readings increase as we move ‘up’ each sensor tree, a result of the hot gases rising from the fire towards the ceiling).

In addition to the readings for each sensor, the data grading algorithm takes as input a constraint network that expresses the expected relations among the sensor readings at any particular time. The simplest type of constraint is the unary constraint which can be used to express ranges in which a sensor is expected to operate (it is a unary constraint since it refers to the values of that sensor alone). For example, each of the thermocouples used in our fire experiment was expected to return values confined to the range 0°C to 2000°C, and this can be expressed through the following constraint (where ‘s’ is the reading of any particular sensor, i.e. of s1001, s1002…s1040.):

```
(0 < s) and (s < 2000)
```

The data grading algorithm starts by first evaluating each of the unary constraints against the latest batch of readings. If one of the constraints is violated, the corresponding reading is graded as unreliable. In the example data set, the reading provided by sensor s1009 in the first tree can be identified in this way as unreliable since its value (9.9E+37) breaks this constraint.

In addition to the unary constraints the algorithm can take more complex constraints that express relations involving more than one sensor value. For example, since the trees were equidistant from the fire, it is expected that sensors at the same height in the room will provide similar temperature values. This can be expressed by a set of binary constraints such as:

```
(similar s1001 s1011) and
(similar s1011 s1021) and
(similar s1021 s1031) and
  (similar s1031 s1001)
```

where the sensor identifiers, s1001 etc., refer to the readings of those sensors at some particular point in time. This constraint expresses the knowledge that all the sensors that are 40cm off the ground should have similar values. Respective constraints can be added for the 9 other heights, resulting in 40 different constraints. Note that this partially exploits the transitivity of the similarity relation as there is no constraint connecting s1001 with s1021 or s1011 with s1031.

Another set of constraints can be used to express the fact that temperature is expected to rise as we move up each tree. This can be expressed by the following binary constraints:

```
(s1001 < s1002) and
(s1002 < s1003) and
        …
(s1008 < s1009) and
  (s1009 < s1010)
```

That is, the reading of s1001 at some point in time should be less than that of s1002, which in turn should be less than that of s1003 and so on. The data grading algorithm will now attempt to use these binary constraints to derive which readings are unreliable. While the last set of constraints appears to be sufficient, the transitivity of the '<' relation is not known to the algorithm and this leads to a minor issue. In the example above the values for the sensors s1002, s1003, and s1004 are 22.9, 21.5, and 26.2 respectively, meaning that temperature is not rising as expressed by the constraints. However, only one of the constraints is violated, namely (s1002 < s1003). The other constraint, (s1003 < s1004), is satisfied. In general, with only one constraint violated it is not possible to tell which of the two sensor readings involved is unreliable. This can be fixed by adding more domain specific knowledge or by adding more constraints. We have solved the problem by adding more constraints, making the transitivity relation more explicit:

```
(s1001 < s1003) and
(s1002 < s1004) and
        ...
(s1007 < s1009) and
  (s1008 < s1010)
```

After processing the unary constraints, the algorithm now proceeds with the *n*-ary constraints. It first evaluates all those constraints that do not involve sensors whose readings at this time have already been identified as unreliable. If one such constraint is violated the algorithm collects the sensor labels in a set along with the number of times each is involved in a constraint violation. The reading of the sensor that violates the most constraints is then graded as unreliable and the process is repeated until no more constraints are found to be violated.

In the example used here, after having identified the reading of s1009 to be unreliable using the unary constraints, the algorithm now finds the readings of sensors s1013, s1007, and s1040 to have violated 4 constraints each. After their readings have been graded as unreliable, the reading of sensor s1036 is found to violate 2 constraints and is graded unreliable. As a result no more constraints are violated and the algorithm terminates, with the overall result as shown in Table 2 where unreliable readings are highlighted.

This compares to sensors s1007, s1009, s1032, and s1040 that were identified by the expert as failing. The reading of sensor s1009 is clearly out of range and requires no further discussion. The readings of s1007, s1032, and s1040 appeared to the expert to be stuck at just over 20°C. The algorithm did not identify s1032 as faulty as this could well be correct in the situation given (but perhaps only by coincidence). The algorithm did however mark as unreliable the readings of two other sensors that were not identified by the expert. Comparing the reading of s1013 to the neighbouring values shows that it is indeed suspect. It may have worked better during the preliminary experiments or it may be correct and just show some unexplained random temperature peak – this cannot be verified now. Sensor s1036 only violates 2 constraints, indicating some lower degree of confidence on the part of the algorithm. The problem here is probably that this sensor is located at a height at which during this stage of the experiment the temperature is rapidly changing, making this reading look suspicious in the current context.

The overall algorithm can be summarized in pseudo-code as shown in Figure 3 (with a grading of 0 being used to indicate that a reading is unreliable).

This algorithm could be improved further by adding a dynamic component to it. Currently, it looks at each batch of readings independently of the ones before and the ones after. However, a sensor that has been destroyed in a fire at one point in time is likely to remain destroyed for the future. Also, there are trends over time, e.g. a rise in temperature at a given sensor that could be used to grade its readings. Thus, it would make sense to carry such information from one batch to the next.

4.2 Phase 2: Data Aggregation and Abstraction

The problem for data grading is to identify in real time sensor readings that are unreliable. While this may reduce the amount of data available, this is not the aim. Data aggregation aims to reduce the amount of data by eliminating redundancy and lifting it to a higher level of abstraction. What this means is that a number of functions will be applied to derive new features from the given data. Often these features will be – or be very close to – information that is meaningful to the user.

For example, one of the features that fire modellers (and fire-fighters) could be interested in is the smoke layer height in a room such as that used in the experiment described above. The smoke layer is the body of hot gas that collects near the top of a room. The smoke layer height is the distance from the ground to the bottom of the hot layer; when this descends too low, occupants are endangered and fire-fighting operations in the room become perilous. Given the sensor data above, the rise in temperature can be visualized as shown in Figure 4**Error! Reference source not found.** (before data grading) and Figure 5**Error! Reference source not found.** (after data grading).

In the case of the smoke layer height, then, the aim of data aggregation would be to process this data to derive a single number corresponding to the height of the layer. The first step in the data aggregation phase usually exploits and then eliminates any redundant data that is collected in order to corroborate values and so reduce noise.

In our example we can use each group of four sensors that are located at the same height to compute the average temperatures at the ten different heights (once again with the underlying assumption that the trees are equidistant from the fire). The result is shown in Figure 6. The height of the smoke layer corresponds to the transition from the cooler temperatures of the lower region to the higher temperatures of the hot gases accumulating at the ceiling. For this example this point is reasonably easy to pinpoint visually in the graph of the averages; and in practice, we are looking for a ‘significant’ rising inflexion in the graph. This point is estimated using the second derivative of the average values with respect to height; with allowances made for local deviations in the averages, in this example the ‘most significant’ zero crossing of the second derivative is found to occur at approximately 1.48m– which gives us a value for the smoke layer height.

This example illustrates how averaging can be used to remove redundancy from the data and the use of mathematical models to derive new features that are not directly available from the data. The result of this phase should still be numeric data, but the amount of data should be much lower and describe the situation in terms of features that are much closer to the features needed by the decision maker – the preconditions in the I-X activity model.

4.3 Phase 3: Information Interpretation for Decision Support

The third phase of our intelligence gathering model involves interpreting the information that is provided from the earlier phases in the very particular context of the task in hand. This interpretation is performed to establish what – if anything – this information *means* for the task, what effect it has, and how it constrains current and future actions. This task obviously requires intelligence (in information processing terms) in order to understand what the relevance of the information and the implications it has for activity in this domain. In many contexts this intelligence will be human in nature, relying on the knowledge and experience of the decision-makers and their support teams to relate the information to their own situation. An alternative or supplementary approach is to augment this human intelligence with computational tools; this becomes increasingly relevant as the amount or complexity of the incoming information grows. In this section we will illustrate this phase of the intelligence gathering task through the example of a computer system based on artificial intelligence ideas.

For the FireGrid project, this decision-support functionality is provided through a purpose-built Command, Control, Communications and Intelligence (C3I) tool intended for use by the fire incident commander (or, more likely, by a fire-fighter acting in a role supporting the commander). The interface provided by this tool, developed with advice from serving fire-fighters, displays the interpretations of incoming information in a manner that is intended to make their relevance to the task of continuously determining the most appropriate tactical mode immediately apparent. Note that other interfaces, intended for other groups of people (say, medical staff, or occupants of the building) with other tasks (treatment of casualties or safe evacuation of the building) could be developed that would interpret and present the same information in entirely different ways; this is an important point, and goes some way to justifying the division – in theory if not always in practice – between the second and third phases of the intelligence gathering model presented here.

The primary role of this interface, then, is to convey succinctly and rapidly to the incident commander the current 'hazard level' at each location within the building. 'Hazard level' is a concept we introduce that is intended to express an integrated measure of the degree of risk to which a fire-fighter operating within that location would be exposed. The hazard level is expressed using a 'traffic light' for each location, where a green light should be interpreted as "the system is unaware of any specific hazard to fire-fighters operating under normal safe systems of work at this location", amber as "additional control measures may need to be deployed to manage hazards at this location" and red as "this location may be dangerous for fire-fighters". Each of these indicators is relevant to a particular location; the definition and extent of a location will be determined by the standard practice and procedures deployed by the task in hand; these may not correspond to the physically differentiated spaces (rooms, corridors, stairwells) within the building itself. In addition to the current hazard level, a traffic light may also concurrently display a second colour, when information is received to the effect that the hazard level in that location is predicted to worsen. When this happens, the 'worse' of the two lights shown indicates the predicted future hazard level (so, for instance, a traffic light simultaneously showing both amber and red lights indicates that the current hazard level at that location is “amber”, and that it is predicted to become “red” at some time in the future). In this manner, the traffic light system adopted expresses both the current state of a location and how this state is expected to develop in time, vital information for assessing the appropriateness of current activities and for planning future activities. In addition to this, the floor of locations where fire has been detected is coloured red to provide the fundamental information of the fire position and spread. Figure 7 shows this interface.

The hazard level measure represents an abstract attribute integrating the various individual current and future hazards that can be inferred to exist from the incoming information. The secondary role of the interface is to provide textual (and hyperlinked) information about these individual hazards, and, more generally, about the state of each location in the building in a pop-up window. A ‘time-slider’ provides the user with a means of exploring predicted hazards; by dragging the slider to any point within a 15-minute timeframe, he/she can see the hazards that are predicted to be occurring at that time. (15 minutes is chosen for the timeframe in this case as this is the furthest time that the available models look into the future.) Figure 8 and Figure 9 show this pop-up display.

Consideration of these individual present and future hazards leads us to a consideration of the reasoning underpinning this interface. The reasoning engine operates with two basic concepts: *beliefs*, propositions about some time at some location which are held by the system to be true, and *rules*, general expressions of the inferences that can be deduced from believed premises. The reasoning works in the following manner:

1. models send messages to the C3I tool;

2. the C3I tool revises its beliefs in the light of the information contained in each new message;
3. the C3I tool applies its set of rules to its revised beliefs to draw new conclusions;
4. the C3I interface is updated to reflect any changes.

Alongside this continual cycle of revising and updating, the tool must periodically revise its beliefs in the light of the passage of time.

To explain what this all means, we first consider the content of these *messages*. A message describes its source and the time it was created along with some content which will be the description of the state of some location at some time. An example might be “message from smoke-layer-height-model at 12:54:32: smoke-layer-height = 1.2m in room-A at 12:54:32”. The state description here consists of a value (expressed in conventional units) for a given state parameter (*smoke-layer-height*), one of a number of such state parameters. Instead of a state parameter, this description might have referred to the occurrence of an event (such as *collapse*); state parameters and events are defined in the *ontology* for the system. An ontology is a formal, agreed definition of the concepts that occur in the context of the task in hand, along with descriptions of relationships between these concepts. For building a system such as that described here, which ranges over a number of different fields of expertise – sensor technology, fire modelling, fire fighting – an ontology becomes an almost vital tool for establishing the appropriate terminologies, for defining relationships between the different areas of expertise and later for integrating the various technologies into a coherent whole (for example by providing a formal language for expressing messages). In most cases, ontology construction requires manual knowledge engineering to establish the terms and achieve consensus among available experts.

For FireGrid, the ontology contains terms related to the physical phenomena surrounding fire, and, important in this case, how these relate to space and time. This gives us the high-level distinction between state parameters, quantities that are (in theory at least) continuously measurable for some place and time, and events, instantaneous occurrences at some location. The subclasses of these two categories (such as smoke layer height and collapse respectively) correspond to concepts that are both potentially of interest to fire-fighters and that can be derived from the available data with the use of models. In addition, the hazard levels (and their definitions) constitute part of this ontology, and the rules that relate them to values of state parameters or events represent "axioms" of the ontology.

The name and time of the message help the C3I tool to assess the effect that the message content should have on its current beliefs. A *belief* is a state description or hazard level that is taken by the tool to be true for some location and over some durations (hence, beliefs have start and end times). In addition, every belief must have one or more justifications, indicating the rationale for believing it. A justification might be a message (if that is the basis for the belief) or the combination of the rule and the beliefs which, taken together, allowed the belief to be deduced.

When a new message arrives, it has to be considered in the context of existing beliefs. If, for example, the tool currently believed nothing about the smoke-layer-height in room-A and received the example message given above at 12:54:32 (or, more likely, some time soon after this, since there will be delays due to information processing and message passing) and assuming that the source of the message (that is the smoke-layer-height-model) is trusted, this message would be the justification for the tool believing the contents of the message. Moreover, since nothing else is known about the values of this state parameter in this location, the reasoning would assign a duration to this belief stretching from the current time to some indefinite time in the future (that is, since the reasoning engine does not believe otherwise, it assumes that the values of state parameters persist, and hence in this case the smoke layer height is believed to remain at 1.2m indefinitely).

If, on the other hand, something is already believed either about the current or future values of the smoke-layer-height, then a more complex train of reasoning begins, which attempts to reconcile this message with the existing belief(s). This may involve adjusting durations of beliefs or, where there seems to be a contradiction, choosing to adopt one or other of the possibilities and disregarding the other. This might be done on the basis of, say, one source being ‘more trusted’ than another or due to the general principle of favouring beliefs based on more recent information as being more likely to be true. Contradictions of this sort occur when there exist inconsistent state descriptions about the same location at the same time; due to the inherent lags and delays between values being read at the sensors and processed information arriving at the C3I tool, the tool must adopt some fairly relaxed definition of what is meant by “same time” (in this case descriptions that have (start) times within 30 seconds of one another are assumed to be referring to the same time).

A further complexity arises since the content of a message may be a prediction – that is it purports to describe the state of some location at some future time. While this might be adopted as a belief with a duration as before, the inexorable flow of time will mean that, assuming this belief has not been retracted in the meantime, at some time the prediction will refer to the current time, and in the absence of other information a choice must be made about whether or not to accept the predicted value as an actual current value. While reasoning of this sort is difficult to justify on grounds of logical soundness, it can be justified on the basis of a cautious approach to the safety of fire-fighters.

Assuming that the set of beliefs has been revised and is consistent, the next step is to apply the set of rules to these beliefs. There are, generally speaking, two types of rules: *hazard rules* and *physical rules*. Rules represent expert knowledge about fire-fighting capabilities and practice (hazard rules) and the nature and progress of fire and associated physical phenomena (physical rules). An example hazard rule might be:

IF smoke-layer-height < 1.5m THEN hazard level = amber

An example physical rule might be:

IF smoke-layer-height < 1.0m THEN max-temperature > 80°C

In each case, a rule consists of one or more conditions and a single conclusion. In the case of a hazard rule, the conclusion is an interpretation of the conditions in terms of the hazard level for the time and place in question; in the case of a physical rule, the conclusion is a state description that should be consistent with the current set of beliefs for that time and place (where consistency, in this sense, may entail the addition or modification of beliefs). In addition, a hazard rule – especially one that refers to less commonly encountered hazards – may have an associated explanation and recommendations. So, for instance, a rule referring to excessive CO levels may offer the explanation that CO levels in that range can "cause headache, fatigue and nausea" alongside the recommendation to "avoid prolonged exposure or consider the use of breathing apparatus".

For each rule, then, a search is made in the set of beliefs for subsets that both satisfy the conditions and are contemporaneous (that is, which have overlapping durations). If such a subset exists, then the conclusion of the rule can be drawn. An inferred hazard level results in a new belief (or in the modification of an existing hazard level belief with an additional justification), with a duration delimited by the latest start time and earliest end time among the subset of beliefs satisfying the conditions. An inferred state description results in a similar modification to the existing beliefs about state descriptions.

Finally, since the application of the rules may have resulted in the inference of multiple simultaneous hazard levels, the inference engine must collate these into a single hazard level for each location at every time. This is a (relatively) straightforward matter of determining the 'worst' hazard level that is believed to apply. So, for instance, if from the state of room-A at the current time, two "amber" hazards and one "red" hazard had been inferred, then the current overall hazard level of room-A is "red", and this is displayed in the corresponding traffic light (A similar search through future states provides future hazards for concurrent display on the traffic light and on the time slider.)

5. Conclusions

In this paper we have characterized the information-gathering problem as bridging the gap between ‘raw’ sensor data and information used by decision makers. The input to the information gathering process is hence defined by the available sensors, and the desired output is defined by the precondition constraints to the standard operating procedures in an I-X agent’s domain model.

We have described the information-gathering process as a three-phase procedure that decomposes the overall problem into phases requiring different types of knowledge and information processing capabilities. The first phase, data validation, aims to remove incorrect information from the input data, thereby creating a consistent view of the current situation. The second phase, data abstraction and aggregation, applies mathematical models to reduce the amount of data, remove noise from the data, and derive features that are closer to the terminology of the user. The third phase, information interpretation, uses a belief revision and rule-based approach to make the information actionable for the decision maker.

In addition to this three-phase information-gathering process, we have identified reasonably general techniques that we expect to be applicable in general, not just in the FireGrid scenario we have used to illustrate our approach. In phase 1 the general idea is to specify constraints between sensors that must be satisfied for the resulting set of readings to be consistent. A greedy algorithm is used to grade sensor readings until maximally large and consistent set of values remains. Phase 2 requires mathematical modelling techniques that will often be domain specific, but statistical methods, for example, provide a toolset that can be expected to be applicable in many domains to remove redundancy and noise. Finally, phase 3 is based on a rule-based system that reasons over space and time, maintaining a set of beliefs and their justifications to supply an application-specific user interface with relevant information.

The evaluation of any system that provides support during large-scale emergencies is a difficult task, of course. This is because such emergencies happen relatively rarely and they tend to vary quite a lot. Setting up experiments on the same scale is very costly, if at all possible. Thus, we have used a number of small, controlled fires to evaluate our approach. The result shows that our system performs well and output corresponds to information generated by experts in hindsight.

We have argued that the results of the information-gathering process as described here, presented in the context of an activity-centric model of an incident, represent a vital source of the sort of accurate, concise, timely and meaningful information that decision makers need to make the right choices under difficult conditions.

Acknowledgements

This project is co-funded by the Technology Strategy Board's Collaborative Research and Development programme, following an open competition. 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, 2005.

[2] HM Fire Service Inspectorate (2002) Fire Service Manual, Volume 2 Fire Service Operations, Incident Command, HM Fire Services Inspectorate Publications, London: The Stationary Office. Crown Copyright.

[3] Ghallab, M., Nau, D. and Traverso, P. (2004) Automated Planning: Theory and Practice. Morgan Kaufmann.

[4] Potter, S., Tate, A. and Wickler, G. (2006) Using I-X Process Panels as Intelligent To-Do Lists for Agent Coordination in Emergency Response, Proceedings of the Information Systems for Crisis Response and Management 2006 (ISCRAM2006), Special Session on Multiagent Systems for Disaster Management and Response, Newark, New Jersey, USA, May 15-17, 2006.

[5] Rogers, A., Ramchurn, S.D., Jennings, N.R., Osborne, M.A. and Roberts, S.J. (2008) Information Agents for Pervasive Sensor Networks. In *Proc. 4th IEEE Int. Workshop on Sensor Networks and Systems for Pervasive Computing*, Hong Kong, China, March 2008.

[6] Schurr, N., Marecki, J., Lewis, J.P., Tambe, M. and Scerri, P. (2005) The DEFACTO System: Coordinating Human-Agent Teams for the Future of Disaster

Response. In *Multi-Agent Programming*, 15, Bordini, R.H., Dastani, M., Dix, J., El Fallah Seghrouchni, A. (Eds.), Springer, 2005.

[7] Wickler, G., Tate, A. and Potter, S. (2006) Using the <I-N-C-A> Constraint Model as a Shared Representation of Intentions for Emergency Response, Proceedings of the First International Workshop on Agent Technology for Disaster Management (ATDM), at the Fifth International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2006), Future University, Hakodate, Japan, May 8-12, 2006.

Height	*Tree 1 labels*	*Tree 1 readings (°C)*	*Tree 2 labels*	*Tree 2 readings (°C)*	*Tree 3 labels*	*Tree 3 readings (°C)*	*Tree 4 labels*	*Tree 4 readings (°C)*
0.4m	s1001	22.9	s1011	24.8	s1021	22.4	s1031	20.5
0.6m	s1002	22.9	s1012	24.9	s1022	22.2	s1032	20.2
0.8m	s1003	21.5	s1013	34.2	s1023	21.5	s1033	20.5
1.0m	s1004	26.2	s1014	29.4	s1024	26.2	s1034	23.5
1.2m	s1005	29.4	s1015	30.6	s1025	26.6	s1035	27.9
1.4m	s1006	42.2	s1016	35.9	s1026	40.9	s1036	31.8
1.6m	s1007	21.1	s1017	69.8	s1027	77.9	s1037	84.6
1.8m	s1008	72.1	s1018	76.9	s1028	79.4	s1038	85.4
2.0m	s1009	$9.9\text{x}10^{37}$	s1019	80.5	s1029	83.9	s1039	87
2.2m	s1010	82.2	s1020	90.6	s1030	88.7	s1040	21.9

Table 1. Sample thermocouple (sensor) readings at a particular point in time for each of the sensors in each ‘tree’ of the fire experiment.

Height	*Tree 1 labels*	*Tree 1 readings (°C)*	*Tree 2 labels*	*Tree 2 readings (°C)*	*Tree 3 labels*	*Tree 3 readings (°C)*	*Tree 4 labels*	*Tree 4 readings (°C)*
0.4m	s1001	22.9	s1011	24.8	s1021	22.4	s1031	20.5
0.6m	s1002	22.9	s1012	24.9	s1022	22.2	s1032	20.2
0.8m	s1003	21.5	s1013	34.2	s1023	21.5	s1033	20.5
1.0m	s1004	26.2	s1014	29.4	s1024	26.2	s1034	23.5
1.2m	s1005	29.4	s1015	30.6	s1025	26.6	s1035	27.9
1.4m	s1006	42.2	s1016	35.9	s1026	40.9	s1036	31.8
1.6m	s1007	21.1	s1017	69.8	s1027	77.9	s1037	84.6
1.8m	s1008	72.1	s1018	76.9	s1028	79.4	s1038	85.4
2.0m	s1009	9.9x10^{37}	s1019	80.5	s1029	83.9	s1039	87
2.2m	s1010	82.2	s1020	90.6	s1030	88.7	s1040	21.9

Table 2. As Table 1, but now highlighting those sensor readings that the data validation algorithm has graded as unreliable.

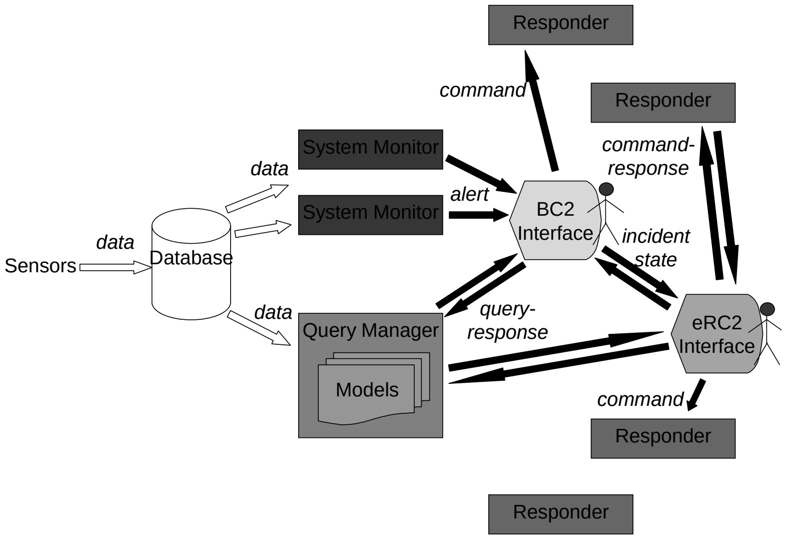


Figure 1: The FireGrid system architecture from a C2 perspective. Arrows show principal communication flows only, with inter-agent communications indicated by solid arrows.

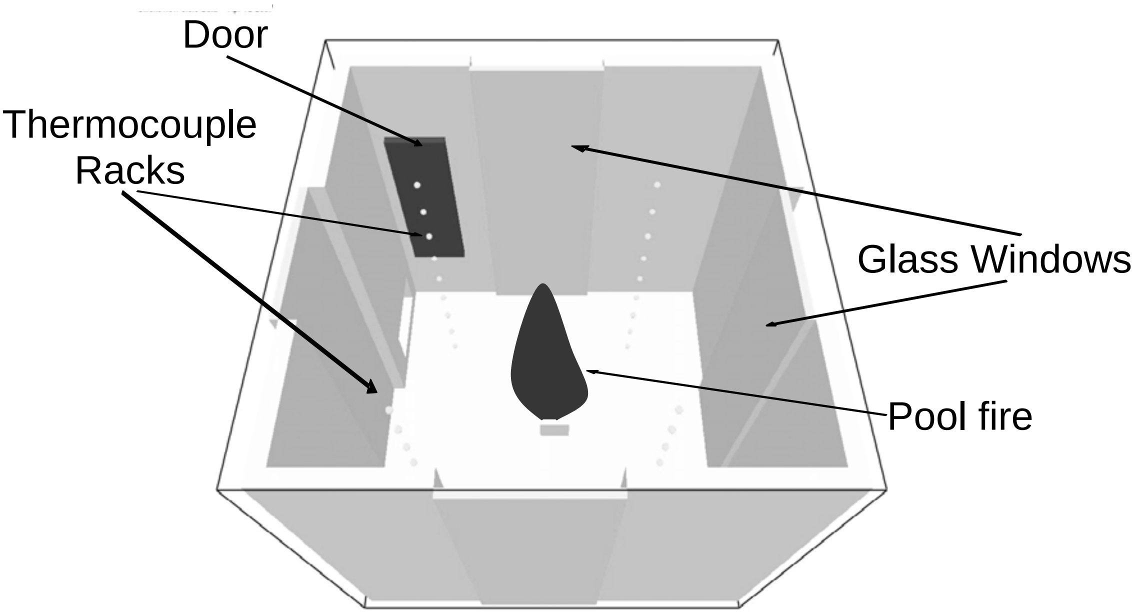


Figure 2: Layout of smoke-box with instrumentation.

```
for every unary constraint c($s$) do
    if not holds(c($s$)) then
        grade($s$) = 0
do
    for every $n$-ary constraint c($s_1$, …, $s_n$) do
        if holds(c($s_1$, …, $s_n$)) then
            continue with next constraint
        if exists $s$ in ($s_1$, …, $s_n$) such that grade($s$) = 0 then
            continue with next constraint
        for every $s$ in ($s_1$, …, $s_n$) do
            increase violation-count($s$)
    $s$ = sensor with highest violation-count
    if violation-count($s$) > threshold then
        grade($s$) = 0
until violation-count($s$) • threshold
```

Figure 3: Pseudo code for constraint-based data grading algorithm

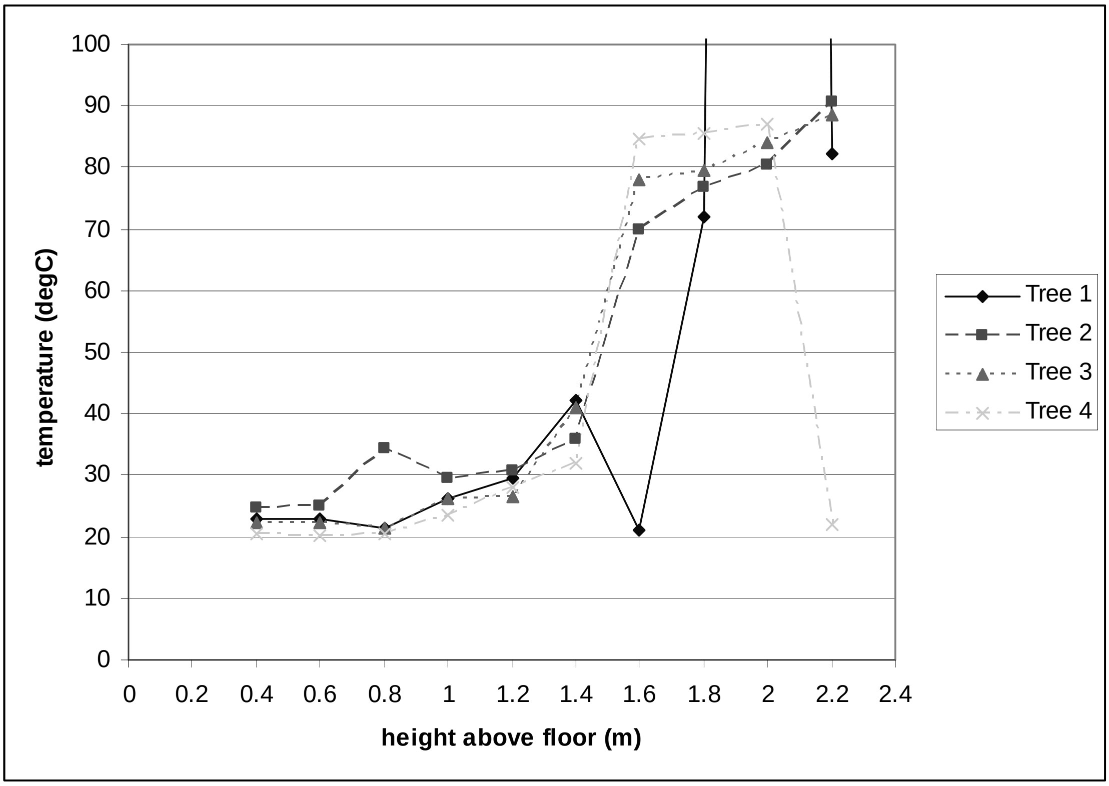


Figure 4: Sensor readings from each tree in Table 1Table 1 plotted against sensor height.

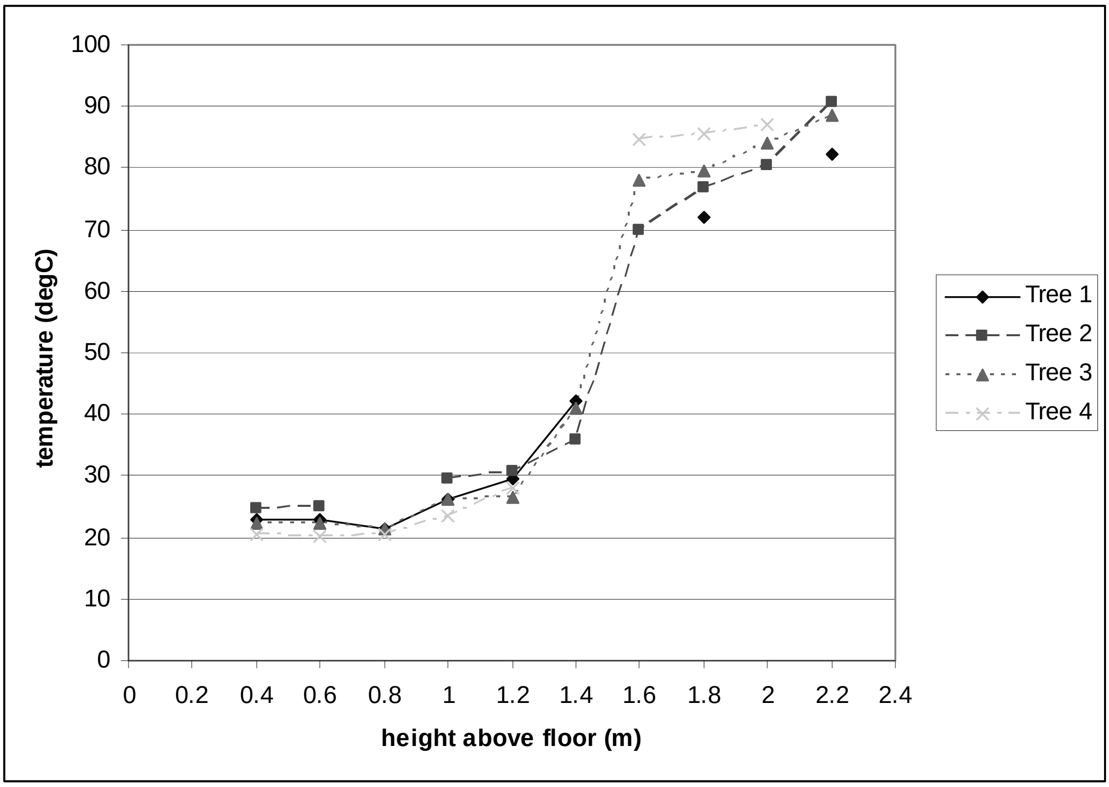


Figure 5: Graded sensor readings from each tree in Table 2 plotted against sensor height.

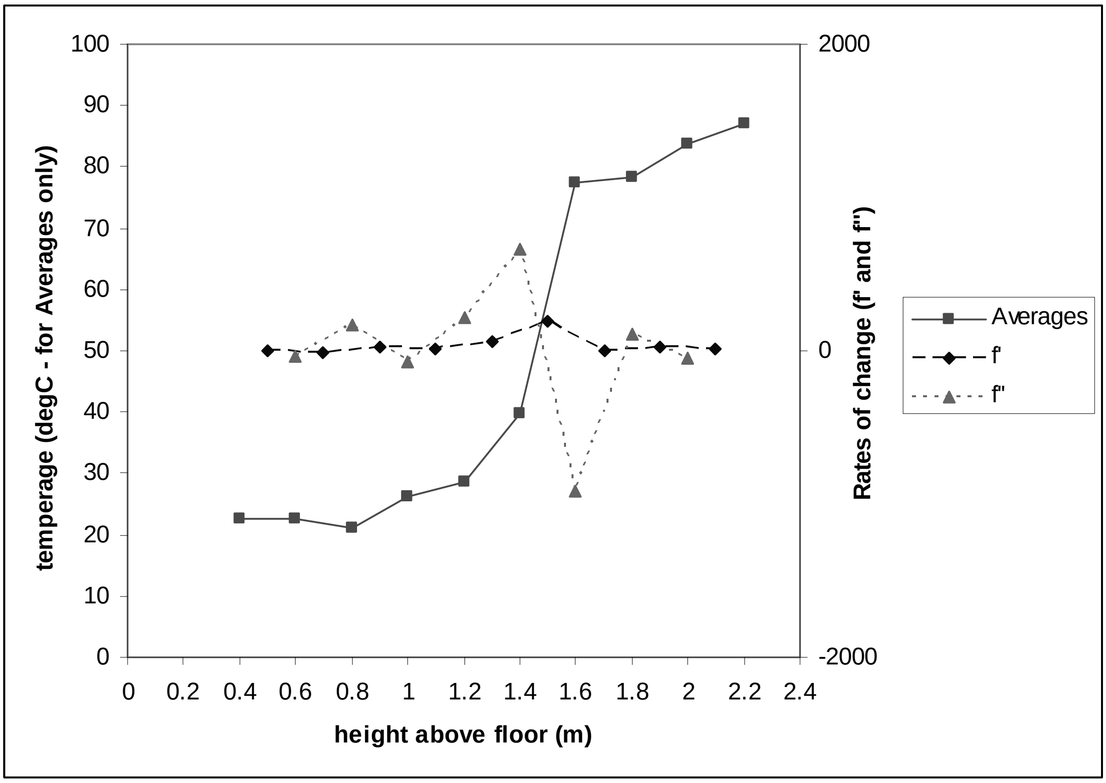


Figure 6: Processed sensor readings showing averages, first derivative and second derivative.

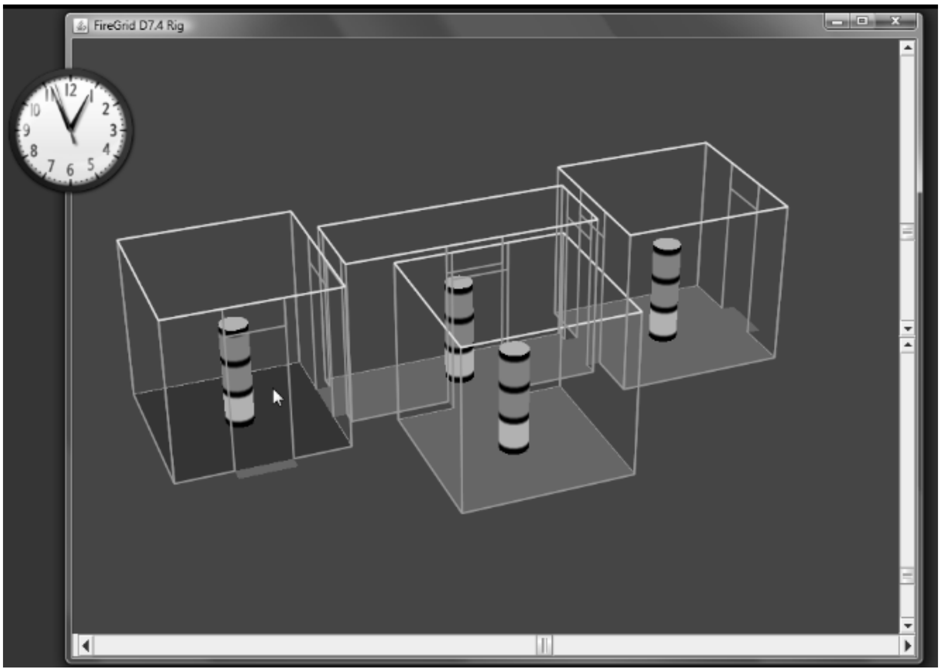
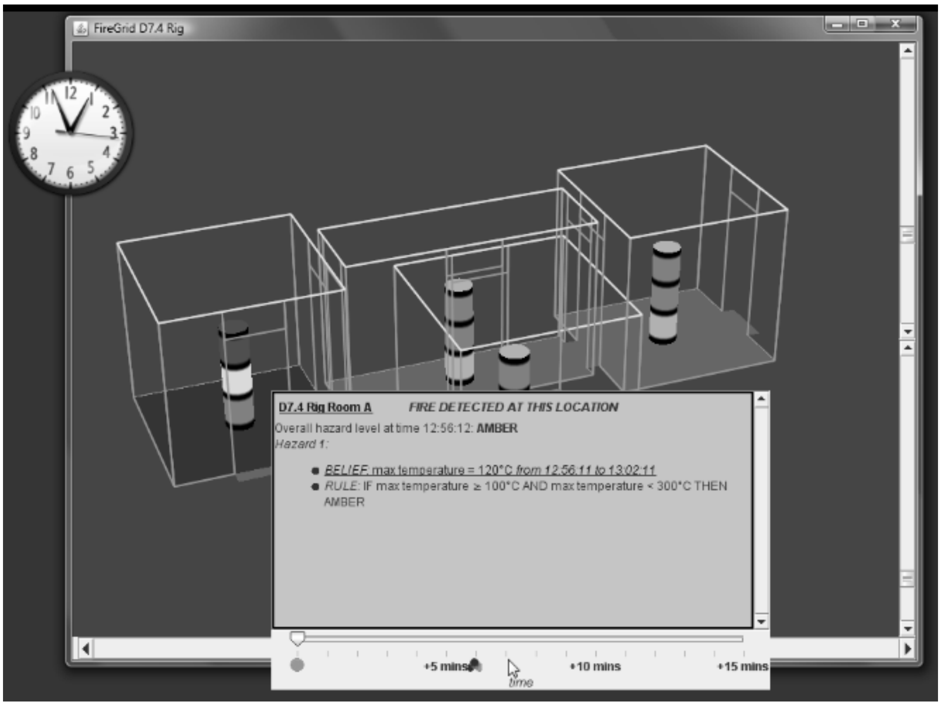


Figure 7: C3I tool interface for 4-room (and 4-location) building, with "traffic light" for each location: in this case the green traffic lights indicate that fire-fighters can operate in every location; however, the red floor in the room under the cursor indicates that a fire has been detected there.



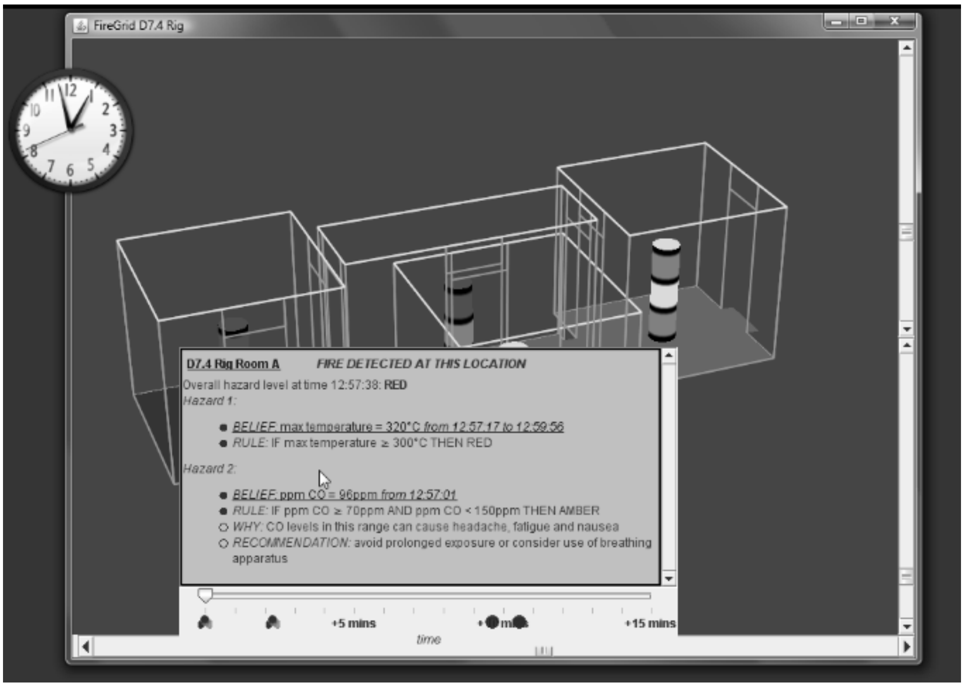
D7.4 Rig Room A *FIRE DETECTED AT THIS LOCATION*

Overall hazard level at time 12:56:12: **AMBER**

Hazard 1:

- *BELIEF:* max temperature = 120°C *from 12:56:11 to 13:02:11*
- *RULE:* IF max temperature ≥ 100°C AND max temperature < 300°C THEN AMBER

Figure 8: At a later time, the user has selected more details about the state of the ‘fire’ room: the current hazard level is now “amber”, due to the combination of belief and rule shown. Moreover the traffic light (and the time slider) indicate that there is a future “red” hazard level predicted (for approximately 6 minutes into the future).



D7.4 Rig Room A *FIRE DETECTED AT THIS LOCATION*

Overall hazard level at time 12:57:38: **RED**

Hazard 1:

- *BELIEF:* max temperature = 320°C *from 12:57:17 to 12:59:56*
- *RULE:* IF max temperature ≥ 300°C THEN RED

Hazard 2:

- *BELIEF:* ppm CO = 96ppm *from 12:57:01*
- *RULE:* IF ppm CO ≥ 70ppm AND ppm CO < 150ppm THEN AMBER
- ○ *WHY:* CO levels in this range can cause headache, fatigue and nausea
- ○ *RECOMMENDATION:* avoid prolonged exposure or consider use of breathing apparatus

Figure 9: At a still later time, the hazard level is now "red" (and has deteriorated in the other locations), and the presence of multiple hazards is indicated by the rules. Note the explanation and recommendations attached to one of the rules.