System of Systems Hazard Analysis using Simulation and Machine Learning

Robert Alexander, Dimitar Kazakov, and Tim Kelly

Department of Computer Science
University of York, York, YO10 5DD, UK.
{robert.alexander, dimitar.kazakov, tim.kelly}@cs.york.ac.uk

Abstract. In the operation of safety-critical systems, the sequences by which failures can lead to accidents can be many and complex. This is particularly true for the emerging class of systems known as systems of systems, as they are composed of many distributed, heterogenous and autonomous components. Performing hazard analysis on such systems is challenging, in part because it is difficult to know in advance which of the many observable or measurable features of the system are important for maintaining system safety. Hence there is a need for effective techniques to find causal relationships within these systems. This paper explores the use of machine learning techniques to extract potential causal relationships from simulation models. This is illustrated with a case study of a military system of systems.

1 Introduction

Large-scale military and transport Systems of Systems (SoS) present many challenges for safety. The term 'SoS' is somewhat controversial — attempts at definitions can be found in [1] and [2]. It is easy, however, to identify uncontroversial examples, Air Traffic Control and Network Centric Warfare being the most prominent. These examples feature mobile components distributed over large areas, such as regions, counties or entire continents. Their components frequently interact with each other in an ad-hoc fashion, and have the potential to cause large-scale destruction and injury.

It follows that for SoS that are being designed and procured now, safety has a high priority. This is particularly true for SoS incorporating new kinds of autonomous component systems, such as Unmanned Aerial Vehicles (UAVs).

This paper is concerned with one aspect of the safety process for SoS, specifically hazard analysis. This is an important first step in any risk-based safety process. Unfortunately, performing hazard analysis on SoS is not easy. Quite apart from the novelty of these systems, and the commensurate lack of examples to work from, the characteristics of SoS raise serious difficulties. For example, ad hoc communications mean that information errors can propagate through the system by many, and unpredictable, routes.

The following section describes the problems faced in SoS hazard analysis, then section 3 proposes multi-agent simulation as a possible solution. An approach to performing hazard analysis, using simulation combined with machine learning, is outlined

in section 4, and the results of a case study are presented in section 5. Section 6 compares the work with existing applications of simulation in safety and section 7 discusses the issue of model fidelity.

2 The Problem of SoS Hazard Analysis

A definition of the term 'SoS hazard' was given by the authors in [3] as *"Condition of an SoS configuration, physical or otherwise, that can lead to an accident."* It follows that SoS hazard analysis is the process of finding those conditions that can lead to accidents.

The problems faced by safety analysts when attempting to perform hazard analysis on SoS fall into two key categories: the immediate issue of failure effect propagation, and the more pernicious category of 'System Accidents'. It has been noted by Kelly and Wilkinson, in [4], that these problems are present in conventional systems, too, but the characteristics of SoS exacerbate them.

2.1 Deriving the Effects of a Failure

In a conventional system, such as a single vehicle or a chemical plant, the system boundary is well-defined and the components within that boundary can be enumerated. When a safety analyst postulates some failure of a component, the effect of that failure can be propagated through the system to reveal whether or not the failure results in a hazard. This is not always easy, because of the complexity of possible interactions and variability of system state, hence the need for systematic analysis techniques, automated analysis tools, and system designs that minimise possible interactions. To make the task more tractable, most existing hazard analysis techniques (such as FFA and HAZOP) deal with only a single failure at a time; coincident failures are rarely considered.

In an SoS, this problem is considerably worse. The system boundary is not well defined, and the set of entities within that boundary can vary over time, either as part of normal operation (a new aircraft enters a controlled airspace region) or as part of evolutionary development (a military unit receives a new air-defence system). Conventional tactics to minimise interactions may be ineffective, because the system consists of component entities that are individually mobile. In some cases, particularly military systems, the entities may be designed (for performance purposes) to form ad-hoc groupings amongst themselves. Conventional techniques may be inadequate for determining whether or not some failure in some entity is hazardous in the context of the SoS as a whole.

2.2 System Accidents

Perrow, in [5], discusses what he calls 'normal accidents' in the context of complex systems. His 'Normal Accident Theory' holds that any complex, tightly-coupled system has the potential for catastrophic failure stemming from simultaneous minor failures. Similarly, Leveson, in [6] notes that many accidents have multiple necessary causes. In such cases it follows that an investigation of any one cause *prior to the accident* (i.e. without the benefit of hindsight) would not have shown the accident to be plausible.

An SoS can certainly be described as a 'complex, tightly-coupled system', and as such is likely to experience such accidents. This line of reasoning can be taken slightly further, however, to note that a 'normal accident' could result from actions by each of two entities that were safe in themselves, but that are hazardous in combination with each other and the wider SoS context.

This latter issue is more immediate when we consider that many SoS will incorporate systems drawn from multiple manufacturers, developed at different times, and operated by multiple organisations. The evolutionary and dynamic nature of SoS structures means that a system designer will not necessarily ever have a clear picture of the entire SoS context.

2.3 Dealing with these Problems

It follows from the above that in order to perform effective hazard analysis for SoS, there is a need for a hazard analysis approach that can find the hazards in a system containing multiple autonomous entities that interact in complex and continually changing ways.

To some extent, this situation is comparable to that faced by the military modelling and simulation community when they attempted to build models that incorporated explicit modelling of entity behaviour rather than only high-level mathematical abstractions. Their solution was the development of multi-agent simulation, which is discussed in the next section.

3 Multi-agent Simulation

Ferber, in [7] provides the following definition of multi-agent simulation: *"Multi-agent simulation is based on the idea that it is possible to represent in computerised form the behaviour of entities which are active in the world, and that it is possible to represent a phenomenon as the fruit of the interactions of an assembly of agents with their own operational autonomy."*

Similarly, Ilachinski, in [8] offers *"[Multi-agent simulations] consist of a discrete heterogenous set of spatially distributed individual agents, each of which has its own characteristic properties and rules of behaviour."*

Typically, the value of multi-agent simulation is asserted in comparison to the mathematical models that have traditionally been used in biology, economics and military analysis. Ferber notes that agent-based models allow the integration of quantitative variables, differential equations and symbolic rules into agent behaviour, thereby providing a means to exploit qualitative observations as well as quantitative information [7]. He also notes that such 'micro-worlds' allow analysts to experiment by modifying agent behaviour and adding new agent types, which is not possible with high-level mathematical models. Most significantly for our purposes, Ferber comments that such simulations *"make it possible to model complex situations whose overall structures emerge from interactions between individuals"*.

Ilachinski, in [8] makes a similar point: in a multi-agent simulation, different levels of behaviour can be observed. Analysts can examine both the top-level emergent behaviour and the low-level interactions between individual agents. That is, the simulations can both predict overall behaviour and explain *why* it occurs.

4 Hazard Analysis Method

The approach described in this paper combines simulation and machine learning. The SoS to be analysed is represented by a model in which each major component of the system (such as an aircraft or radar station) is represented by an agent. Each agent is described in terms of its physical capabilities (such as the ability to fly at a certain speed) and its rules of behaviour. The simulated system is then placed in a simulated environment (containing, for example, terrain and hostile forces) and given orders and objectives for an appropriate military mission.

The resulting simulation model will have dynamics that are too complex to understand merely by watching it run. It is therefore necessary to derive other models that characterise its behaviour and that *are* simple enough for humans to read and understand.

This derivation is achieved by defining a set of deviations over the simulation model, and exploring the set of combinations of these deviations. Machine learning allows an automated analysis of the resulting output data, resulting in a set of comprehensible rules that relate deviations to accidents occurring in the simulation. The intention is that these rules will guide analysts towards identifying some hazards in the system that they otherwise would have missed.

The five steps of the method are described in the following five sections.

4.1 Build Model

The approach is potentially open to multiple modelling approaches, but the effectiveness of the analysis process will hinge on the type of model used. Models that are used for traditional performance analysis can focus heavily on capturing the overall *functionality* of the system. Models for hazard analysis need to capture much of the *mechanism* of the system's operation. This is for two reasons:

- The system needs to be manipulated in ways which the original designers might not expect (particularly in ways which relate to implementation rather than functional specification), and the model has to respond appropriately.
- Deviations will be derived by studying how the system works and applying a set of heuristics. Mechanical detail that is not modelled cannot be used in this process.

Inter-agent mechanisms (communications protocols, operating procedures, roles) are more important than intra-agent mechanisms (since the internal behaviour of agents will already be well understood). In order to capture the necessary inter-agent mechanisms, agent actions and communication must be made explicit. Also, as discussed by Hall-May in [9], an important aspect of the system model is that of 'policy' or 'operational doctrine', the set of rules or procedures that attempt to constrain the global behaviour of the system.

Further discussion of modelling approaches is outside the scope of this paper. The important issue of model fidelity, however, is addressed in section 7.

4.2 Specify Deviations for Model

Given the model specified in the previous step, a set of possible deviations must be derived. There are many ways to do this, but the one that is used in this paper is to identify the *channels* over which *interactions* can occur between entities. Examples of channels include network wires, radio transmissions or simply being located in the same airspace. A set of *guide words* is then used to hypothesise some *failure modes* of these channels. Each combination of some entity exhibiting some failure mode on some channel provides a single distinct 'deviation'.

The use of guide words for deriving deviations is based on their usage in HAZOP [10]. By combining the words used in HAZOP with those from the computer-system analysis method SHARD [11] we can derive the set shown in Table 1.

Table 1. Channel deviation guide words

Guide Word	Interpretation
Omission	The interaction does not occur
Commission	The interaction occurs when not expected
Early	The interaction occurs too early
Late	The interaction occurs too late
Too much	A parameter associated with the interaction is increased
Too little	A parameter associated with the interaction is decreased
Conflicting	The interaction conflicts with another interaction on the channel

4.3 Run Simulation to Explore the Effects of Deviations

Given a model and set of possible deviations, the simulation must now be run and the results recorded. In an ideal world, a run would be performed for every possible combination of deviations, but this is not realistic because of the number of such combinations entailed by even a small deviation set. An efficient approach is to work through the low-order subsets of the deviation set. Given a priori knowledge of the probability of each deviation, it will be possible to show that the higher-order subsets represent wildly improbable circumstances.

4.4 Learn Rules

The task of machine learning can be viewed as one of function approximation from a set of *training instances* expressed as input-output pairs; given a function specification (a set of named input parameters (the 'features' used for learning) and a particular form of output value), the algorithm learns the relationship between combinations of parameter values and the output of the target function for those values.

For our purposes, the features represent causes and the output values are the consequences within the simulation. All the features used in the current work are explicit parameters that are given to the model, and the target function is the set of accidents that occurs during the simulation run. For example, in an air traffic scenario, the analysis might determine that when the parameter "collision warning distance" is reduced below 8km, it becomes possible for the accident "mid-air collision" to occur.

Many machine learning algorithms are described in the literature; a summary is provided by Mitchell in [12]. Machine learning is used here to produce *descriptive* rather than *predictive* models, i.e. models are learned, but they are not then used to classify any new instances; rather, they are studied by human analysts who wish to understand how the system behaves under various failure conditions. Learning approaches that produce 'black box' outcome models (i.e. models that are not very amenable to human comprehension), such as Bayesian Learners or conventional Neural Networks, are therefore not very helpful for this purposes of this work..

Learning approaches that *do* produce comprehensible models are more valuable in that they allow the engineer or analyst to inspect the learned model to discover *why* it considers a particular parameter combination to be hazardous. This is analogous to the analyst observing the hazardous result produced by the original (simulation) model and then inspecting the event log, or watching the run via visualisation, in order to determine why the model produced the result that it did.

4.5 Investigate Rules

Once some set of rules has been learned from a system model, safety analysts need to study and make use of them. This is a two stage process:

In the first stage, the analysts must try to understand the rules, and how they relate to the modelled system (rather than the model itself). Important questions at this stage include "Why did the algorithm learn this rule?", "Is this realistic, or is it merely a simulation artifact?", "Why (in terms of the mechanisms of the model) are these particular features/causes so important?"

The output of this first stage includes a revised set of rules, which have been manually filtered (to remove rules that have no apparent correspondence to the real system) and augmented with explanations and references to particular simulation runs which express the behaviour well.

In the second stage, analysts must consider the implications of the identified hazards for the real system. Key questions are "Is this hazard serious and plausible enough to warrant our attention?" and "What are we going to do about it?".

5 Example

Our hypothesis is that the machine learning algorithm will learn rules that cover all the hazards that were identified by manual analysis. The learning tool that has been used here is a decision tree learner using the C4.5 algorithm as described by Quinlan in [13]. The algorithm was chosen because it is fast, stable implementations are readily available, and the resulting rules are human-comprehensible. The implementation used was that provided by the data mining tool WEKA (described by Witten and Frank in [14]) under the name of 'J48'.

The example uses a simulation model of a military unit engaged in anti-guerilla operations. An overview of the elements in the system is shown in figure 1. Notionally, it contains four types of entities: Unmanned Air Vehicles (UAVs), Unmanned Self-Propelled Guns (UGVs), transport helicopters and infantry sections. As the infantry

move only by air, a helicopter with troops on board is represented by a single entity. Infantry sections never appear on the map on their own.

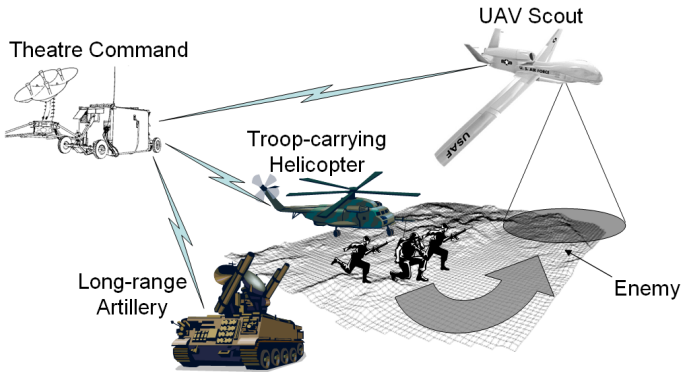

Fig. 1. The System

A single scenario has been implemented for this model, in which the units in the system must detect and neutralise a number of static enemy positions. The UAVs move on pre-defined search paths, and when they detect an enemy presence they contribute this to a shared picture which is available to all friendly entities. Responding to this shared picture, the artillery entities fire on the enemy until the UAVs report that it is adequately weakened. Once such weakened enemies are identified, the helicopters move in to take control of the areas on the ground. It is at this stage that the safety risk manifests, as the manned helicopters move across the terrain and engage the enemy.

5.1 Hazards in the Model

For a given system it is relatively easy to determine the types of accidents that can occur, since the set of entity types is finite and there are only a few ways in which an entity can be involved in an accident. Simple examination of our model reveals that the following accidents are possible:

- Accident 1 — Helicopter collides with another helicopter
- Accident 2 — Helicopter collides with a UAV
- Accident 3 — Landed helicopter is hit by artillery fire
- Accident 4 — UAV collides with a UAV
- Accident 5 — Helicopter hit by enemy fire

By running the model with all combinations of the possible entity-failure pairs (some 260000 in number), and studying the results manually, we have been able to identify the following hazards in the system:

- Hazard 1 — Friendly forces in field of fire of inaccurate artillery

– Hazard 2 — A UAV is in shared airspace with no ability to detect other airborne entities
– Hazard 3 — A helicopter moves into anti-aircraft range of a strong enemy unit

Hazard 1 can cause accident 3, hazard 2 can cause accident 4, and hazard 3 can cause accident 5. In the runs that we have performed, given the deviations that we have implemented (in this case only entity-level failures), there are no instances of accidents 1 and 2. We have not, therefore, been able to identify any hazards that would lead to them.

5.2 Learning Rules from the Model

In section 4.5, we noted that heuristics can be used to identify and respond to issues implicit in the learned rules. In this case, the heuristic could be described as 'single point of failure' (i.e. the failure of one entity allows a variety of accidents to occur), and an appropriate response would be to re-evaluate the roles in the system to redistribute some of the functionality of the entity, or to insulate other systems from the effects of its failures.

The model has approximately a quarter of a million combinations of possible deviations. For this simple model we can learn from the complete set of runs, but this will not be practical for larger examples, so it would be misleading to do so here. Therefore, only the first 8000 runs were performed. Most of these include only small numbers of failures, which is appropriate given that larger numbers of failures become increasingly improbable.

As noted above, it was possible to determine by examination the accidents that were possible. These then provided learning 'targets'; for each such target, a decision-tree model was learned to predict the combinations of failures that would cause that accident to occur. The failures that provide the learning features for the decision tree were expressed as entity-failure pairs; one example would be "UAV4 has suffered the failure loss_of_communications". The list of the accidents that were used as learning targets, together with the labels used for them in the learning tool, was as shown in Table 2.

Table 2. Possible accidents

Accident	Label
Helicopter X hit by enemy fire	ehX
Helicopter X hit by friendly artillery fire	ghX
UAV X collides with UAV Y	cuXuY
Helicopter X collides with UAV Y	chXuY

Enumerating these combinations gave 36 targets to learn models for. In practice, many of these accidents were not manifest in the available runs and so no models were to generated for them. For each accident, the 8000-run input set was processed into a table with boolean values for each of the 18 entity-failure pairs and a label of either 'safe' or the code for the target accident. Those runs that contained accidents, but not

the current target accident, were discarded. This ensured that the data set contained positive and negative examples. The data set was then given to the learner, which was told to learn a decision tree for predicting the label from the parameters.

As a simple example, consider the accident 'cu1u4' i.e. where UAV 1 collided with UAV 4. Of the 8000 runs, 192 were safe and another 2048 contained cu1u4. These runs were labelled appropriately while the rest were discarded, and the resulting data set was given to the tree learner.

The resulting decision tree is shown in figure 2. The rule expressed here is that if UAV 4 exhibits the failure 'noairsensors' then this collision will occur, otherwise it will not. This learned rule is 100% accurate with respect to the training data; it perfectly captures the implicit model that was fed to it. Whether this is the optimal *inductive* inference will require further testing effort; see below.

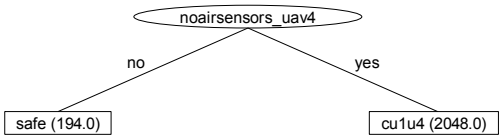

Fig. 2. Decision tree learned for accident 'cu1u4'

A more complex model is that for the accident 'gh1' (One of the UGVs hits helicopter 1). The learned tree is shown in figure 3.

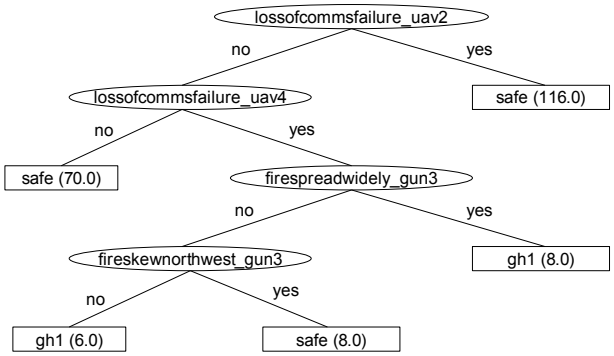

Fig. 3. Decision tree learned for accident 'gh1'

Although the tree notation is attractive for simple models, it becomes increasingly unwieldy as models become larger. As noted by Quinlan in [13], a decision tree can be 'flattened' into a set of production rules. The tree for figure 3 has five leaf nodes and therefore corresponds to the following five rules:

1. $\neg lossofcommsfailure\_uav2 \wedge \neg lossofcommsfailure\_uav4 \rightarrow safe$

2. $\neg lossofcommsfailure\_uav2 \wedge lossofcommsfailure\_uav4 \wedge \neg firespreadwidely\_gun3 \wedge \neg fireskewnorthwest\_gun3 \rightarrow accident$
3. $\neg lossofcommsfailure\_uav2 \wedge lossofcommsfailure\_uav4 \wedge \neg firespreadwidely\_gun3 \wedge fireskewnorthwest\_gun3 \rightarrow safe$
4. $\neg lossofcommsfailure\_uav2 \wedge lossofcommsfailure\_uav4 \wedge firespreadwidely\_gun3 \rightarrow accident$
5. $lossofcommsfailure\_uav2 \rightarrow safe$

In order to assess the importance of each rule, we can assess the probability of it occurring in practice by applying a priori probabilities to the individual failures and setting a 'threshold of concern' beyond which a rule will be considered too improbable to be worthwhile investigating. This is similar to the 'incredibility of failure' concept used in the nuclear industry; the probability that is used for this is given in [15] as 10^{-7} per year of operation (equivalent to 10^{-11} per hour).

We will consider all failures to have a probability 10^{-3} of being present in any given instance of the scenario, and set a lower threshold of 10^{-11}. Rules with a lower probability than that will be discarded as implausible.

It can be noted that this approach is somewhat naïve; for example, we have assumed complete independence between the failures. As the authors noted in [3], the nature of systems of systems is such that many apparently independent failures have common causes. It can also be noted that, within an entity, one failure may cause (or indeed mitigate the effects of) another. We have assumed a simple flat probability for each individual failure; these probabilities would much better justified if they came from entity-level safety analyses. For the context of this paper, however, these assumptions suffice for illustration.

Table 3 summarises the results of this process. For each accident, it shows the number of instances that contained that accident, the number of rules in the learned model (in total/rules that led to the accident occurring), percentage accuracy of that learned model (over the training set), the number of rules above the plausibility threshold and the highest estimated probability of any of the rules occurring.

Note that the table only contains those accidents for which examples were found in the first 8000 runs. Many of the accidents that could potentially occur (as apparent from a simple examination of the simulation model) did not manifest in this result set.

For the five accidents identified in section 5.1, we have rules that correspond to three of them. (Accidents 1 and 2, involving helicopters colliding with UAVs or other helicopters, do not occur in any of the runs we are working with). These three accidents correspond to the three hazards that were previously identified, as shown in Table 4.

At the beginning of this section, we gave our experimental hypothesis as "the machine learning algorithm will learn rules that cover all the hazards that were identified by manual analysis". It can be seen that the example supports this hypothesis, in that we have at least one rule that describes a way to cause the accidents corresponding to each hazard.

The question remaining is whether a safety engineer studying this simulation and the learning results would be lead directly to discovering the hazards (as opposed to merely noting that the accidents could happen). It is certainly plausible that the engineer would discover the hazards, but to give a more affirmative answer in practice (with

Table 3. Summary of the learned rules

Accident	#runs	#rules	#plausible rules	highest prob	% accuracy
eh1	6657	54/33	19	9.98×10^{-4}	96.7
eh2	6966	42/28	14	1×10^{-3}	99.3
eh3	6738	56/35	21	1×10^{-3}	99.1
eh4	6842	56/35	21	1×10^{-3}	99.2
gh1	14	5/2	2	9.97×10^{-4}	99.5
gh3	14	5/2	2	9.97×10^{-4}	99.5
cu1u4	2048	2/1	1	1×10^{-3}	100
cu4u3	3904	2/1	1	1×10^{-3}	100
(other)	0				

Table 4. Accidents found and the corresponding hazards

Label	Accident	Hazard
gh1, gh3	3	1
eh1-4	5	3
cu1u4, cu4u3	4	2

systems of realistic complexity) will require the application of the approach to larger scale industrial case studies.

5.3 Investigating these Rules

The preceding discussion has looked at the experimental results from the perspective of function approximation from failures to accidents. We can also look at how these rules relate to the behaviour of the system as observed through visualisation of the simulation runs. This is a necessary step in any case because these rules have only been learned in terms of explicit simulation parameters, rather than in terms of the actual mechanisms of the simulated system; they tell us (as simulation operators) how we can cause an accident to manifest, but they don't tell us what events within the simulation model lead to that accident. This requires additional analysis, but the analyst has an advantage in that he is aware of these learned rules and can look first at those runs that implement the rule preconditions, knowing in advance what overall result he expects to see.

In the current example, we have derived a number of rules from the simulation that describe how accidents can occur. For purposes of illustration we will follow up the accident 'gh1' (UGV fires on helicopter 1). The rules for this are given in section 5.2 and are shown in figure 3 as a decision tree. The rules specify that the necessary conditions for this accident are that UAV 4 has lost all communications and that UAV 2 has *not* suffered any loss of communications. Furthermore, UGV 3 must either be (a) firing accurately (i.e. not skewed) or (b) have its aim spread widely.

Observing some of the runs in which this accident does occur, it is apparent that removing UAV 4 from the data fusion loop (through communications loss) has an effect on the order in which enemy positions are detected, and that this affects both the order in which the guns target the enemy positions and the order in which the helicopters fly

out to them. In order for this to be dangerous UAV 2 must be functioning normally; if UAV 2 has suffered a communications failure then this changes the ordering of target selection, and the corresponding outcome of airspace deconfliction actions, such that the result is a safe state. (The failures specified for UGV 3 merely mean that it must be able to hit the square it aims at).

Given this interpretation of the rules, is it plausible that this could occur in the real world, or are we merely seeing a simulation artefact? Superficially, it would seem to involve a rather unlikely combination of events (the helicopters being in just the right position at just the right time) but it can be observed that the fire from the UGVs and the movement of the helicopters are concentrated around specific locations (those occupied by the enemy positions) and specific times (when the enemy are first revealed as valid targets).

Finally, what changes can we propose to prevent this accident occurring in the future? One apparent issue is that this accident depends heavily on a failure of UAV 4. We studied the assigned flight path for UAV 4, and it was apparent that UAV 4 is particularly significant in this context in that it covers a large number of enemy positions. One viable option would be to change the UAV roles to ensure a more even distribution of coverage, perhaps by introducing an additional UAV.

6 Existing Applications of Simulation in Safety

In that the current work uses simulation for safety-related analysis, it is similar to the work of Blom *et al.* in airspace system safety [16] and Johnson in hospital evacuation [17]. Both of those, however, use Monte Carlo techniques to acquire quantitative statistical measures of the overall safety of a system under specified conditions. By contrast, the work described in this paper attempts to determine the relationship of simulation parameters to distinct (undesirable) modes of behaviour of the system; the aim is to acquire a *qualitative* understanding of system behaviour.

Computer system simulation approaches (such as the DEPEND tool described by Goswami *et al.* in [18]) generally focus on the interaction of software processes running on networked processors. Our work is distinct from that in that it explicitly deals with mobile physical entities interacting in physical space.

Perhaps the closest work described in the literature is that of Platts *et al.* in [19], in which rules are learned which relate the behaviour of an unmanned aircraft to success in a particular mission. The approach described in this paper is similar in that it involves learning rules which relate entity behaviour to unwanted hazardous consequences.

7 Model Fidelity

In this work, the aim of the simulation is to identify ways in which hazards (and hence accidents) could reasonably occur; in this respect, it is comparable to existing hazard analysis techniques. Any hazards that are identified through simulation will require further manual investigation — the simulation result is valuable in that it has drawn the analyst's attention to the hazard and 'made a case' for its plausibility by means of the recorded event trace.

A standard objection to the use of simulation for analysis is to question the fidelity of the model with respect to the real system that it purports to represent. In this context, it is important to note that almost all hazard analysis is performed with respect to some model of the real system; it cannot be said to be performed on the system itself. This is partly because the complexity of a real system is unmanageable (and much of it irrelevant) but also because hazard analysis is important very early in the safety life cycle, before the detail of the final system design is available.

Whilst there will always be concerns with the fidelity of the models we use, the use of models and approximations remains an inevitable part of real-world hazard analysis. One difference when using models for simulation, rather than for manual analysis, is that in manual analysis there is great opportunity for pragmatic human interpretation, thereby covering a multitude of deficiencies in any modelling approach adopted.

The use of simulation in our work is for what Dewar *et al.* describe in [20] as 'weak prediction'. They note that *"subjective judgement is unavoidable in assessing credibility"* and that when such a simulation produces an unexpected result *"it has created an interesting hypothesis that can (and must) be tested by other means"*. In other words, when a simulation reveals a plausible system hazard, other, more conventional analyses must be carried out to determine whether it is credible in the real system. Therefore, the role of the simulation analysis is to narrow down a huge analysis space into one that is manually tractable.

8 Summary and Future Work

This paper demonstrates an approach to performing hazard analysis for complex systems of systems using a combination of multi-agent simulation and machine learning. This was motivated by the successful use of multi-agent techniques in other fields of modelling and analysis. As illustrated in the example in this paper, we have been able to show that the approach can be used to identify hazards.

Challenges that remain to be tackled include the application of this technique to a wide variety of systems and scenarios, and combining the results of simulation and analysis across multiple scenarios and system configurations. There is also scope for further experimentation with different machine learning algorithms and different techniques for introducing deviations into simulation runs.

Acknowledgements The work described in this paper was funded under the Defence and Aerospace Defence Partnership in High Integrity Real Time Systems (Strand 2).

References

1. Maier, M.W.: Architecting principles for systems-of-systems. In: 6th Annual Symposium of INCOSE. (1996) 567–574
2. Periorellis, P., Dobson, J.: Organisational failures in dependable collaborative enterprise systems. Journal of Object Technology **1** (2002) 107–117

3. Alexander, R., Hall-May, M., Kelly, T.: Characterisation of systems of systems failures. In: Proceedings of the 22nd International Systems Safety Conference (ISSC 2004), System Safety Society (2004) 499–508
4. Wilkinson, P.J., Kelly, T.P.: Functional hazard analysis for highly integrated aerospace systems. In: IEE Seminar on Certification of Ground / Air Systems, London, UK (1998)
5. Perrow, C.: Normal Accidents: Living with High-Risk Technologies. Basic Books, New York (1984)
6. Leveson, N.: A new accident model for engineering safer systems. In: Proceedings of the 20th International System Safety Society Conference (ISSC 2003), System Safety Society, Unionville, Virginia (2002) 476–486
7. Ferber, J.: Multi-Agent Systems: an Introduction to Distributed Artificial Intelligence. Addison-Wesley (1999)
8. Ilachinski, A.: Exploring self-organized emergence in an agent-based synthetic warfare lab. Kybernetes: The International Journal of Systems & Cybernetics **32** (2003) 38–76
9. Hall-May, M., Kelly, T.P.: Defining and decomposing safety policy for systems of systems. In: Proceedings of the 24th International Conference on Computer Safety, Reliability and Security (SAFECOMP '05). Volume 3688 of LNCS., Fredrikstad, Norway, Springer-Verlag (2005) 37–51
10. Kletz, T.: HAZOP and HAZAN: Identifying and Assessing Process Industry Hazards. 3rd edn. Institution of Chemical Engineers (1992)
11. McDermid, J.A., Nicholson, M., Pumfrey, D.J., Fenelon, P.: Experience with the application of HAZOP to computer-based systems. In: Proceedings of the Tenth Annual Conference on Computer Assurance, IEEE (1995) 37–48
12. Mitchell, T.M.: Machine Learning. McGraw-Hill (1997)
13. Quinlan, J.R.: C4.5: Programs for Machine Learning. Morgan Kauffman (1993)
14. Witten, I.H., Frank, E.: Data Mining: Practical machine learning tools and techniques. 2nd edn. Morgan Kaufmann, San Francisco (2005)
15. Ammirato, F., Bieth, M., Chapman, O.J.V., Davies, L.M., Engl, G., Faidy, C., Seldis, T., Szabo, D., Trampus, P., Kang, K.S., Zdarek, J.: Improvement of in-service inspection in nuclear power plants. Technical Report IAEA-TECDOC-1400, International Atomic Energy Agency (2004)
16. Blom, H.A.P., Stroeve, S.H., de Jong, H.H.: Safety risk assessment by Monte Carlo simulation of complex safety critical operations. In Redmill, F., Anderson, T., eds.: Proceedings of the Fourteenth Safety-critical Systems Symposium, Bristol, UK, Safety-Critical Systems Club, Springer (2006) 47–67
17. Johnson, C.: The Glasgow-hospital evacuation simulator: Using computer simulations to support a risk-based approach to hospital evacuation. Technical report, University of Glasgow (2005) Submitted to the Journal of Risk and Reliability.
18. Goswami, K.K., Iyer, R.K., Young, L.: DEPEND: A simulation-based environment for system level dependability analysis. IEEE Trans. Comput. **46** (1997) 60–74
19. Platts, J.T., Peeling, E., Thie, C., Lock, Z., Smith, P.R., Howell, S.E.: Increasing UAV intelligence through learning. In: AIAA Unmanned Unlimited, Chicago IL (2004)
20. Dewar, J.A., Bankes, S.C., Hodges, J.S., Lucas, T., Saunders-Newton, D.K., Vye, P.: Credible uses of the distributed interactive simulation (DIS) system. Technical Report MR-607-A, RAND (1996)