ELSEVIER

Reliability Engineering and System Safety 70 (2000) 29–39

RELIABILITY ENGINEERING & SYSTEM SAFETY

www.elsevier.com/locate/ress

Safety analysis of autonomous excavator functionality

D. Seward[a,*], C. Pace[a], R. Morrey[b], I. Sommerville[b]

[a]*Department of Engineering, Lancaster University, Lancaster LA1 4YR, UK*
[b]*Department of Computing, Lancaster University, Lancaster LA1 4YR, UK*

Received 19 April 1999; accepted 15 March 2000

Abstract

This paper presents an account of carrying out a hazard analysis to define the safety requirements for an autonomous robotic excavator. The work is also relevant to the growing generic class of heavy automated mobile machinery. An overview of the excavator design is provided and the concept of a safety manager is introduced. The safety manager is an autonomous module responsible for all aspects of system operational safety, and is central to the control system's architecture. Each stage of the hazard analysis is described, i.e. system model creation, hazard definition and hazard analysis. Analysis at an early stage of the design process, and on a system that interfaces directly to an unstructured environment, exposes certain issues relevant to the application of current hazard analysis methods. The approach taken in the analysis is described. Finally, it is explained how the results of the hazard analysis have influenced system design, in particular, safety manager specifications. Conclusions are then drawn about the applicability of hazard analysis of requirements in general, and suggestions are made as to how the approach can be taken further © 2000 Published by Elsevier Science Ltd.

Keywords: Excavator; Mobile machinery; Hazard analysis; Safety manager

1. Introduction

Over the past four years, a team in the computing and engineering departments at the Lancaster University has been involved in developing the safety case for an autonomous robotic excavator. Lancaster University Computerised Intelligent Excavator (LUCIE) is based on a commercial manual hydraulic excavator, but has an on-board computer system in place of a driver to control the hydraulics and therefore the machine. It is being developed with one particular task in mind: the digging of foundation trenches on a building site. The ultimate aim is to develop a machine that will be able to accept a program of trench locations and dimensions and then traverse a building site and dig a series of trenches meeting these specifications. It should do this autonomously without the need human intervention.

Although the technical hurdles in the way of achieving this basic task are significant, they are by no means insurmountable. The greater challenge—one which faces the designers of all commercial robots to be employed in a working environment—is to ensure that the machine will always achieve its basic task *without accident*. One of the main aspects of the research therefore has been to carry out a safety analysis on the system.

1.1. Problem definition

This particular safety problem domain differs from most other problem domains on which safety analyses have previously been carried out. These domains have almost always involved structured environments. That is, they are concerned with systems that control man-made environments, particularly chemical production plants and nuclear installations. Even in robotic systems, hazard analysis has been applied mostly to robotic manipulators, where the manipulator's environment is generally structured and can be controlled, in most of the cases [1–4]. In this case however, the system is required to interact directly with a natural environment: a building site. During normal operation it will need to establish the state of that environment—the locations of obstacles, its own location etc.—, and take actions to change that environment—move to a different location, dig a bucketful of earth etc. Thus our problem domain is unusual in the field of safety analysis, because it involves direct interaction between the system and an *unstructured environment*.

* Corresponding author. Tel.: +44-1524-5-93010; fax: +44-1524-3-81707.

E-mail address: d.seward@lancaster.ac.uk (D. Seward).

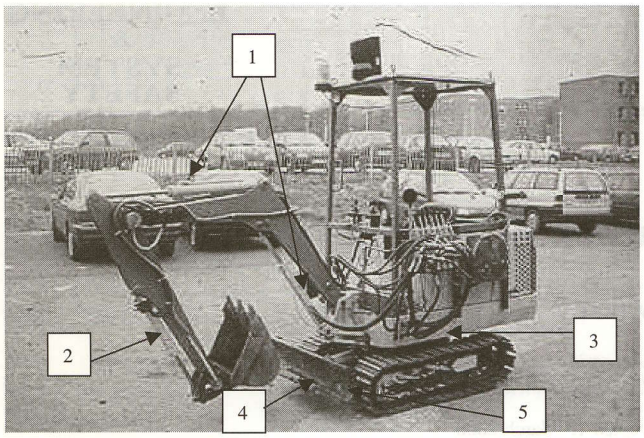

Fig. 1. The autonomous excavator, LUCIE.

1.2. An analysis based on the requirements specification

There have been many studies that show that a high percentage of system errors are due to deficiencies in requirements specification [5–7]. Many accidents occur, not because the design and implementation fail to deliver the desired functionality reliably, but because situations arise that have not been foreseen and for which no system action has even been defined. Systems that interface to unstructured environments necessarily have more complex functionality, which is harder to define, so in this case there was a higher risk of dangerous lapses in definition of functionality. Safety analysis therefore needs to be applied early in the development life cycle, prior to completion of the design, in an attempt to ensure that a safe system action has been defined for all dangerous real-life situations. This paper is an account of such an analysis.

2. The excavator system

Although the analysis has to be carried out at an early point in the development process, some initial design and functional specification still needs to be done. This section gives an overview of the information available when the analysis was carried out, and therefore gives a context for the analysis that follows.

The first major source of information is the design of the commercial digger on which LUCIE is based. This digger, shown in Fig. 1, is a JCB 801.4 mini excavator, having one boom arm at the front. All of the excavator's movements are hydraulically driven, these being:

1. The movement of the arm in an (x,y), that is vertical, plane, using two rams.
2. The rotation of the bucket at the end of the dipper, in the same vertical plane, using another ram.
3. The rotation of the cab at its connection to the undercarriage, effectively providing movement for the arm in a horizontal plane (slew).
4. The movement up and down of a dozer blade at the front of the undercarriage.
5. The movement of two parallel caterpillar tracks independently, backwards and forwards.

The excavator is automated by installing a distributed computer architecture to control its hydraulics and therefore all movement. A target hardware system for LUCIE is based around three processors:

- An activities manager (AM) to issue high-level commands for digging and navigation. This contains the knowledge base extracted from theoretical studies and the observation of actual expert operators.
- A low-level controller (LLC) for the arm and tracks. This simply converts the movement demands from the AM into drive signals for the electro-hydraulic valves.
- The safety manager (SM) to monitor the environment and ensure safe behaviour.

The PC104 format is used with the processors communicating via a CAN bus as shown in Fig. 2. The following sensory equipment is also provided:

- Four potentiometers on the joints for angle measurement.
- A two axis tilt sensor.
- A Leuze RotoScan RS 3 optical laser distance sensor, for obstacle detection up to a range of 15 m.
- A Trimble 7400MSi series satellite GPS for location and navigation.

The entire architecture is built around the concept of a SM as an autonomous processor ultimately responsible for all aspects of system safety. This can be seen as a development of the concept of a software safety executive [8] and has several clear advantages over architectures in which responsibility for safety is spread across the whole system:

- The design of the other modules can be simplified as they need focus only on their principal tasks. This reduces both development and validation costs.

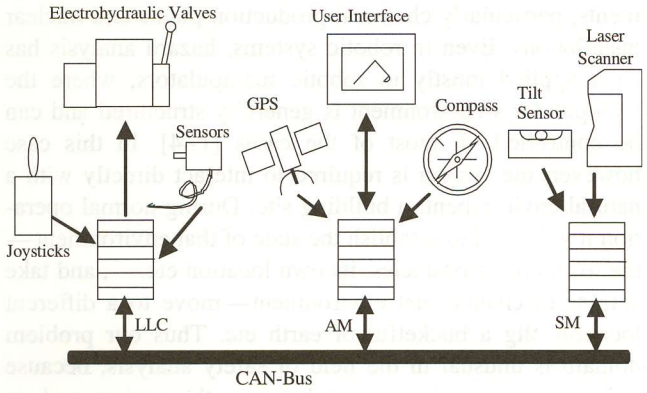

Fig. 2. Hardware architecture for excavator control.

- Other modules are free to use non-deterministic techniques such as fuzzy logic, artificial neural networks and other Artificial Intelligence design approaches, which are considered to have a low safety integrity [9].
- The SM should be able to ensure the system is returned to a safe state in the event of failure or malfunction of one of the other modules.
- If one module can guarantee safety, whatever the status of the rest of the system, then that module only will need to have a high safety integrity. Concentrating rigorous design methods on that module will have a good chance of creating a safe system.

This architectural framework has similarities with Visinsky's et al., expert system supervisor [10], although an autonomous planner is represented in this architecture through the introduction of the AM, rather than a human operator.

With such a design, it is inevitable that any safety analysis will be largely concerned with the functionality of the SM itself; although the question of the substance of information provided to it, together with the means of provision, is also very important.

In addition to this basic hardware design, a prototype algorithm has been proposed that carries out its basic task of digging trenches. This algorithm is expressed as a finite state machine and gives a reasonable idea of the kinds of movements the digger will have to perform and in what order.

3. Analysing system safety

3.1. Using hazard analysis early in the development process

A requirements specification is an attempt to describe the desired functionality for a system. It usually consists of an imprecise combination of diagrams, tables and text descriptions that give a general impression of how the system should operate. The problem is that this description has, in the end, to map onto the functionality of the system; and the functionality of any system driven by software is very complex. It is therefore very difficult to ensure that:

1. the document is unambiguous and does not contradict itself;
2. the document specifies what the system should do in *every* situation, i.e. is complete.

For true completeness a system action needs to be defined for every combination of *external and internal* state, i.e. not only the external operational state, but even the state of internal parameters, both hardware and software. Obviously it is unrealistic to expect any document in practice to be so thorough; and completeness in requirements specification is accepted as meaning something other, and less than mathematical completeness. Nancy Leveson [11] uses the term 'sufficient completeness' and defines how a safety requirement may be 'sufficiently complete with respect to safety'. Here, appropriate system actions must be specified in the event of *all* dangerous situations. To achieve this completeness, it is clearly necessary to establish first all possible dangerous situations—external/internal state combinations that have the potential for causing an accident.

Therefore the centre-piece of the safety analysis should be a hazard analysis, carried out at an early stage in development. The objective is to ensure that the outcome of such an analysis would define the requirements specification for the system as a whole and more specifically for the SM. Thus, through such a process, the requirements specification for the whole system is expected to be complete sufficiently with respect to safety.

General hazard analysis techniques necessitate a clear definition and description of the system, and this generally entails a completed design (including system components and their inter-connections) [3,10]. Through this approach, it is then considered possible to identify all possible hazards and analyse the mode in which each of these hazards occur, i.e. their causes, and furthermore establish ways and means of avoiding or, at least, reducing the occurrence of such hazards to within acceptable limits. In all such cases the aim is to establish if the system *will do* anything hazardous. However in applying hazard analysis at an earlier stage in the development process, as was done in this case study, the aim is to establish what the system *should do* and that such a functional definition should be hazard-free. Indeed, the hazard analysis in this case, rather than serving to outline potentially hazardous system features has to serve as a means of defining how the system is to behave safely from a functional perspective.

3.2. Defining basic functionality

The overall aim of the hazard analysis was to discover what parts of LUCIE's desired functionality are likely to give rise to a hazard, so that extra functionality could be added, or the basic functionality adjusted to prevent the hazard or occurrence of any subsequent accident. Therefore, the first and essential stage of the analysis is the definition of the basic functionality. It should be noted that this is not the same thing as a requirements specification. The point in providing the functional definition is precisely to aid the hazard analysis process and not to aid the mainstream requirements definition; although the two may be compatible.

So, what does the analysis require of this functional definition? Primarily, the description is required to establish when hazards are likely to occur. Therefore, the functional definition should describe the actions of LUCIE at a physical level. Furthermore, it should describe in detail the type and sequence of movements LUCIE will make in the process of digging a series of trenches. If it did these two, it would in effect be an informal model of LUCIE in action.

A 'deconstruction' was therefore performed from a high

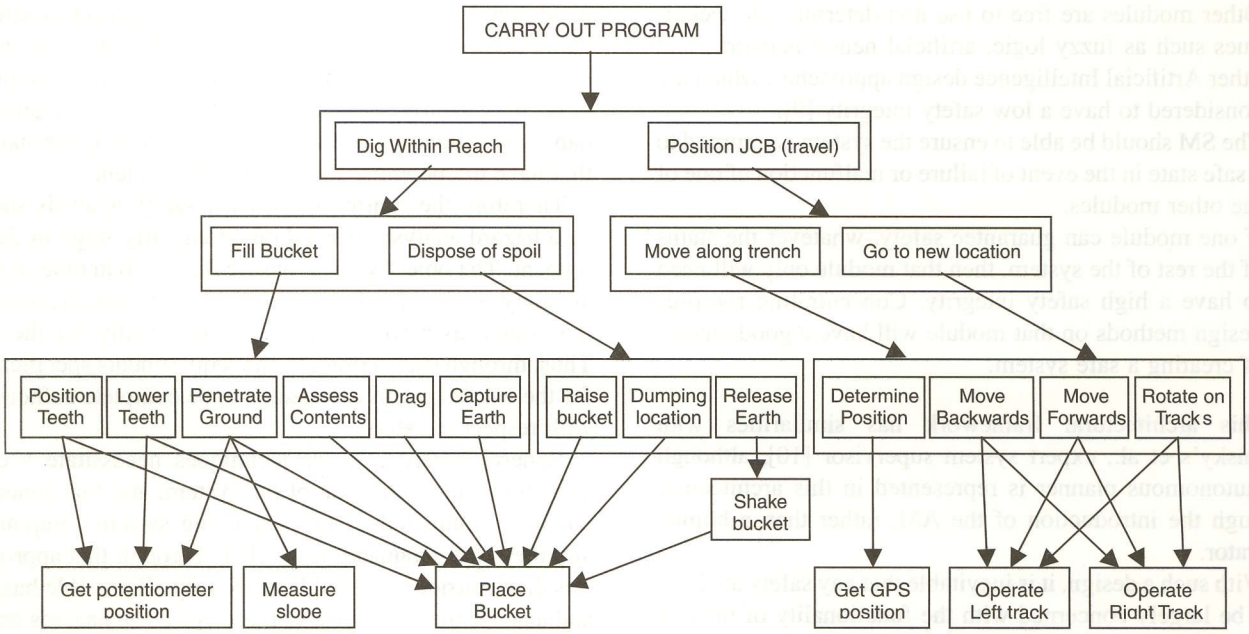

Fig. 3. LUCIE's functional deconstruction.

level description of what LUCIE will do. The most general functional description, 'carry out program', is taken as the starting point and encompasses the complete system functionality. The generic action is then deconstructed into more specific actions that it entails. Each of these new actions is then broken down into more rudimentary actions and so on. This process was repeated until no further deconstruction was possible. Such a deconstruction is presented in Fig. 3.

Of course there are many other established ways of organising a requirements description. There are more formal techniques such as RML [12] and much of the information in the deconstruction was based on a finite state diagram, which had previously been worked out. It is felt, however, that there are two advantages to the method described above:

1. It encourages the definer to think about the complete system functionality, because the starting point (the root node 'carry out program') embraces the complete functionality. Although completeness in actual definition is not guaranteed, the tendency to concentrate on the known details and skirt around unknown details of functionality, is offset.
2. It allows the functionality of the system to be examined at whatever level of abstraction is appropriate to the problem in hand.

3.3. Defining hazards

Hazard definitions, in general, suffer from the same problem as any representation of reality. The intricacies of a real-life situation can never be described precisely. In practice, a representation can only ever comment on a small sub-set of the factors that combine to make up the essence of a situation. Thus, a hazard definition must necessarily be the definition of a whole *class* of real situations, i.e. all those situations that the definition does not exclude. The difficulty is particularly acute in this problem domain, where definitions of dangerous situations must include the state of a complex unstructured environment.

The problem of defining such classes boils down in practice to one of choosing the right criteria for hazard definition and sticking to them when defining individual hazards. The choice of hazard classification is also fundamental from the SM task perspective. Providing a coherent hazard classification allows for a sound basis on which SM requirements specifications may be decomposed. This would then enable the development of operational patterns, which would suit the different hazard classes.

Following the necessary considerations from a SM hazard containment point of view, a similar attitude to that used when defining basic functionality has been adopted for hazard definition. Hazards are therefore defined in the most generic way possible (i.e. large class sizes), to avoid making any initial assumptions about system and environment or any interactions between the two, and to make sure no hazardous situations were omitted from the analysis before starting. The following have been adopted as the principal hazard definitions:

1. Collision with an object on the surface.
2. Collision with an underground object.
3. Toppling of the excavator.

Certainly, there is an element of 'physical effect' in the

use of the terms 'collision' and 'toppling'; and this seems one of the best ways of creating hazard definitions that are all inclusive. However, the separation between collisions with surface and underground objects also embodies some knowledge about 'system action'.

Section 4 provides an account of the analysis for determining hazard causes. Collision with underground obstacles is not evaluated further; mostly due to the lack of available equipment for underground obstacle detection.

4. The hazard analysis

The purpose of this step is to work back from a set of hazard definitions to establish possible causes of hazards; and then with reference to the functional description, to establish situations in which they might arise. The outcome is to have a two-fold application:

1. defining means of avoiding hazards through the general development of the system;
2. identifying the operational requirements for the SM for hazard containment, in order to provide an acceptable level of system safety.

As an initial step towards a deeper understanding of hazard causes, the generic hazard classification outlined in Section 3.3 is further subdivided according to the main functional deconstruction, that is digging and travelling. Such a division is considered necessary due to the inherently different operational states in which hazards are to occur. Furthermore, due to the diverse dynamics for the occurrence of hazards, a further deconstruction is made for collision occurrences with static and dynamic obstacles.

4.1. Defining hazard dynamics

Following an initial attempt to apply Fault Tree Analysis (FTA) to hazard definition, it was determined that a preliminary analysis involving the physical dynamics of the identified hazards had to be carried out. Such a decision was taken as it was noted that:

- it was hard to decide whether each level of deconstruction should have represented a temporal change or a more detailed description of the event above;
- it was impossible to represent the physical reasons why the hazard occurred, as discrete events with a fault tree. Yet, these reasons had to be understood in order to determine the hazard causes.

FTA's are considered to be a well-established and widely-used technique and indeed such an analysis technique has been applied in several circumstances to robotic systems safety analysis [2,4]. Yet in such cases, the FTA's are not utilised as a basis for defining the dynamics of a system's interaction with its environment. Rather, the FTA's are applied as a tool for defining the mode in which system elements interact to give rise to potential hazards. Hence, in this case, where the concern is on defining safe system operation, such an analysis was deemed unsuitable unless a more detailed understanding of the excavator's interaction with the environment was available.

Hence, an exercise to express excavator and environment interactions in more formal terms was carried out in order to identify:

- at which point during operation there is danger of a hazard occurrence;
- how the hazard might best be avoided from a system dynamics point of view, or failing that, detected and an accident avoided.

It was found that such an activity was particularly useful in the case of the toppling hazard event, where a mathematical evaluation of the conditions for toppling was necessary. In the specific case for toppling, this mathematical analysis gives rise to the following concerns, with respect to potential hazard occurrence:

1. the slope of the ground;
2. the position of the excavator itself—particularly the distribution of the mass in the plane of the boom arm and the width of the base in that plane;
3. any external rotation force (e.g. wind pressure) that might be applied to the excavator;
4. terrain consistency, with concern to the ground's ability to withstand the load imparted to it by the excavator.

Such concerns are also greatly influenced by the type of operation carried out by the excavator, whether digging or travelling.

A similar exercise was carried out in the event of collision hazards where concerns range from the obstacle position and relative motion with respect to the excavator to the identification of certain obstacle features. Again, such concerns are to be influenced by the operational state of the excavator.

Actions were also identified, within the scope of this exercise, which were to take the excavator away from potential hazardous situations. Typical examples for the toppling hazard include folding the boom inwards against the body of the machine, rotating the cab to make it facing forwards using the boom arm as an extra support, and emptying the bucket to reduce the toppling moment. Similar activities were outlined for collision hazard avoidance, where in particular, slowing down the operational speed according to the estimated real-time system reaction limitations, was considered.

4.2. Application of FTA

Having defined the physics of such hazard occurrences and the expected physical system reactions and following

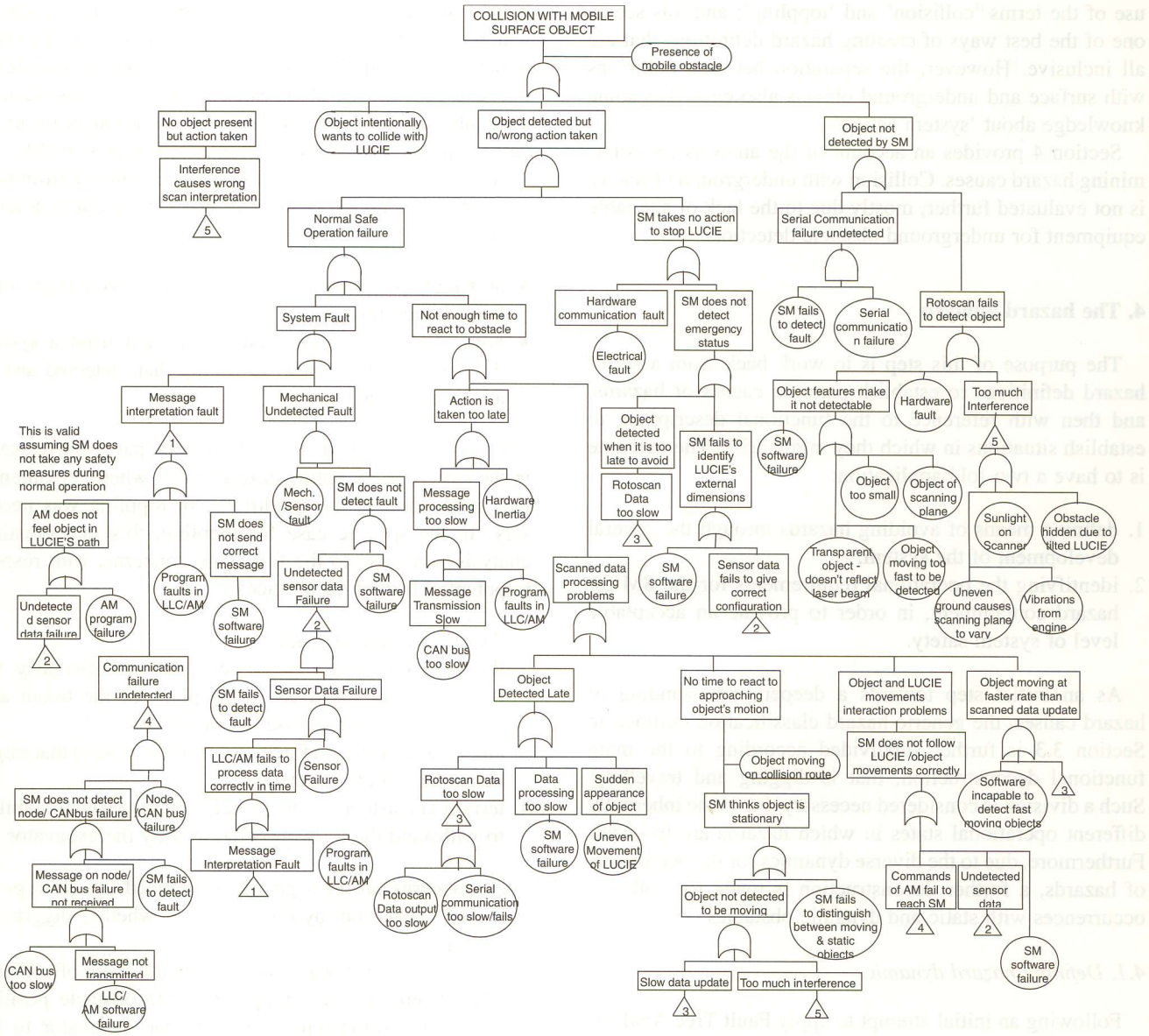

Fig. 4. Fault tree for collision hazard with mobile surface obstacle while travelling.

the initial functional definition, a more robust and complete functional description that included hazard-handling was achieved. With such a functional definition available, FTA is considered to be useful for identification of internal system interactions, which might have given rise to the hazardous event. This is different from the requirement to identify the physics of the event itself, for which fault trees where initially considered to be impractical. Thus a FTA, was developed only after a clear functional description of the mode of interacting with the environment was available.

Therefore, the combination of the FTA and the physical definition of the system's interaction with its environment was to form the basis for defining the system SM requirements for ensuring acceptable operational safety.

To further aid in the development of the FTA, a number of assumptions on the SM's intended mode of operation have been made, amongst which are the following:

- The SM is capable of communicating and interacting with both the AM and Low-level Controller to an extent where actions would only be performed if sanctioned as safe by the SM.
- The SM is considered to be 'a highly safety critical' element within the system. This does not mean that other elements are not safety critical, but rather that a reliable SM is essential to ensure that at best, all actions performed are not to cause any hazard occurrences and at worst, the system would fail safe.
- Actions to re-establish safety when an unsafe state results are to be directed by the SM, in order to ensure that a system restart would not in itself cause further hazard potentials.

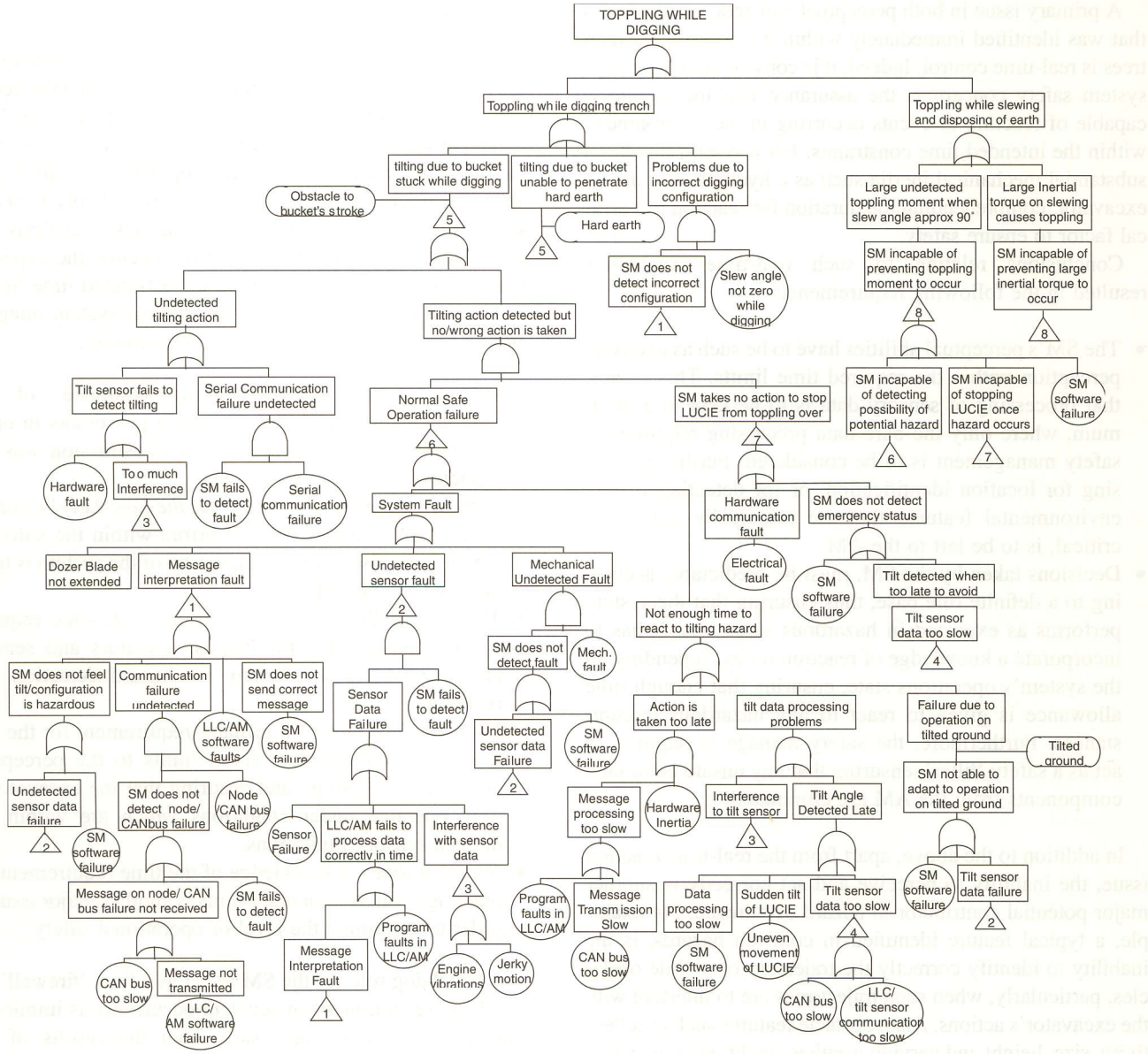

Fig. 5. Fault tree for toppling hazard while digging.

Such assumptions are required to ensure a framework under which fault trees can be constructed in a useful manner.

Figs. 4 and 5 illustrate a sample of the fault trees constructed to describe the system failure propagation to the extent of hazard occurrence. These fault trees represent the analysis for a dynamic collision hazard with the excavator travelling throughout the construction site (Fig. 4), and a toppling hazard while the excavator is in a digging operational state (Fig. 5). Similar fault trees were developed for static and dynamic collision and toppling hazards for all travelling and digging operations, including for slewing and dumping situations. The whole analysis data and description is provided in Ref. [13].

5. Hazard analysis results

5.1. Environmental interaction limitations

From the FTA's performed on the various hazards, there were two major causes for hazard occurrence, these being:

1. *Perceptual deficiencies,* which inhibit the system from detecting the occurrence of certain events, such as obstacle presence and other environmental features, resulting in the inability to perceive a potential hazardous situation.
2. *Action deficiencies,* which inhibit the system from taking the necessary action on perceiving a potential hazardous situation, resulting in the system's inability to react to the occurring events or hazard itself.

A primary issue in both perceptual and action deficiencies that was identified immediately within the constructed fault trees is real-time control. Indeed, it is considered that a major system safety concern is the assurance that the system is capable of reacting to events occurring in the environment, within the intended time constraints. For a system that has a substantial mechanical inertia such as a hydraulically driven excavator, comprehending the duration for reaction is a critical factor to ensure safety.

Conclusions relating to such real-time constraints resulted in the following requirements:

- The SM's perceptual abilities have to be such as to ensure perception within the required time limits. This means, that processing of sensory data is to be kept to a minimum, where only the bare data processing required for safety management is to be considered. Further processing for location identification, or for detecting certain environmental features that are principally operation-critical, is to be left to the AM.
- Decisions taken by the SM, must be predictable according to a definite rule base, thus ensuring that the system performs as expected in hazardous situations. It has to incorporate a knowledge of reaction times, depending on the system's operations state, ensuring that enough time allowance is given to react to any hazardous circumstances. Furthermore, the safety manage is required to act as a safety 'filter', ensuring that any unsafe command components from the AM are eliminated.

In addition to the above, apart from the real-time reaction issue, the inability to perceive and act correctly is another major potential contributor to hazard occurrence. For example, a typical feature identified in collision hazards, is the inability to identify correctly the trajectory of mobile obstacles, particularly, when such trajectories are to interfere with the excavator's actions. Also, obstacle features such as reflectivity, size, height and varying weather conditions can lead to errors in the comprehension of environmental features.

This clearly outlines the necessity of identifying the system's sensory limitations in comprehending the environment. Otherwise, constraints must be imposed on the environment in order to ensure the required operational safety integrity. In this regard the FTA provided an aid to identifying the perceptual performance necessary for hazard aversion. The analysis also helped to identify where further mathematical rigour was necessary in identifying environmental features. For example, in avoiding collision, the requirement to distinguish mobile obstacles provides the necessity to be able to compute relative object motion with respect to the excavator, taking into consideration the excavator's own motion. Similarly in the case of toppling, the necessity to detect dug trenches and other terrain characteristics have been identified. Furthermore, definitions of obstacle motion characteristics and other obstacle features have been identified also through the FTA.

5.2. Internal safety aspects

The FTA, apart from interpreting the interaction between the system and the environment, further identified the necessary procedures for assessing internal system integrity. This was a constantly recurring theme within each FTA for every identified hazard in every identified operational state. Assurance that the internal system is operating reliably is necessary in order to ensure that hazardous situations are perceivable, and that once perception occurs, the expected reaction would take place within the estimated time limit.

The major areas of concern for internal system integrity arising from the FTA's are outlined hereunder:

- *Inter-controller communication*—assurance of no communication breakdown between processors in order to ensure that data reception and transmission was not hindered.
- *Software integrity monitoring for the Low-Level Controller and AM*—any systematic errors within the software which gives rise to erratic operation of the system is to be detected by the SM.
- *Mechanical System and Sensor Integrity*—the requirement to ensure that mechanical actuators and sensors are operative is a necessity for hazard detection and reaction.
- *Sensory Limitations*—a primary requirement for the SM is to be knowledgeable about limits to the perceptual ability of the system, and ensuring that the excavator is only operated under conditions which are within the defined sensory limitations.
- *Reaction time*—a knowledge of the time requirement for reacting to environmental events is another major issue in order to determine the level of operational safety.

A watchdog role for the SM to act as a final 'firewall' for accident prevention when accident occurrence is imminent, was also considered necessary from the results of the analysis.

5.3. SM functionality requirements

The Hazard Analysis provides a comprehensive and early understanding of the required SM functionality.

The requirements for managing safety were based on the generation and maintenance of safe excavator operational states, which were to be achievable through the SM's operation. Two general safe states were defined, an internal safe state and an external safe state, both of which had to be attained for safe system operation. The internal safe state identified the internal system status, such as communication requirements between processors and the mechanical and sensory operation requirements for safety integrity. The external safe state identified the environmental safety aspects, including environmental conditions for which safety integrity was achievable.

The internal safety assessment task was based on the

following subtasks:

1. Communication assessment to ensure communication bus integrity and avoid overloading the communication system.
2. Software assessment for ensuring, in as simple and straight forward a manner as possible, the AM and low-level Controller Integrity.
3. Actuator and sensor assessment, including mechanical integrity and sensory data reliability and noise impact on the sensory data.
4. Assessment of the integrity of the shut-down safety action, for ensuring shut-down availability in the event of such an operational requirement.

The external safety assessment task was to be based on the following subtasks:

1. Environmental condition monitoring in order to obtain a reliability estimate of other sensory data such as distance measurement and obstacle detection.
2. Determination of obstacle distance and trajectory.
3. Determination of potential interference between obstacle trajectory and excavator trajectory and hence defining safe operational zones.
4. Determination of terrain tilt trends and the resulting assessment of potential toppling conditions.

Both tasks were managed through a safety decision making process. Such a decision making process was implemented through a decision network based on the fault tree structures developed within the FTA. Furthermore, the decision network provided the method by which the excavator is returned to a safe state from an unsafe operating situation.

6. Lessons learnt

In this section it is first necessary to discuss an issue that pervaded our whole analysis, but which has not yet been mentioned: the necessary trade-off between safety and viability. The overall aim in performing the safety analysis was to come up with a strong case for how an automated excavator could be developed that would be both viable and operate safely. The raw materials for this were:

- a reasonable idea of the algorithm the excavator would use to carry out its basic task;
- a knowledge of the sensor hardware that would be available;
- a knowledge of the environment in which the excavator must work.

During the hazard analysis, it was found that there was always a trade-off to be managed, trying to come up with a scenario in which available sensors could be deployed in order to avoid the occurrence of an accident. This was founded on the fact that there are two ways of ensuring safe operation:

1. Add extra functionality to handle a dangerous situation which may occur.
2. Place restrictions on the environment, such as banning humans from the work area, in order to ensure that a dangerous situation does not occur.

If (1) is not scientifically possible or is impractical given the resources of the developers, then (2) must be invoked. However, (2) reduces the usefulness of the system and care must be taken to ensure that viability is maintained. Thus, when having to make decisions between (1) and (2), it has been found useful to follow the following procedure. Each hazard begins by assuming an environment with no restrictions, and attempt to deploy available resources to prevent the occurrence of any accident arising from the hazard, given that environment. If safety cannot be guaranteed, then the least possible restriction to the environment is to be added and tried again, until safe operation can be guaranteed. At this point, the restrictions are examined to see if they preclude viable operation. If they do, then recommendations are made for extra sensing capabilities. In short, placing restrictions on the environment is a last resort, in order to have the greatest possible functionality while still maintaining safety.

This type of trade-off situation occurred during the analysis of every hazard. For the toppling hazard there was a trade-off between allowing operation on land that was not completely level and making sure that toppling was not a serious danger. For the collision hazard, there was a trade-off between allowing mobile obstacles, such as other vehicles and people on the building site and making sure a collision could not occur. These and various other trade-off decisions gave another reason for making the initial hazard definitions so vague. Part of the aim of the hazard analysis became the *specification* of the environment in which LUCIE could work safely. If the hazards had been described in more detail initially, assumptions would have been made inevitably about the environment that would have distorted and restricted the analysis.

The results of the safety analysis also led to a necessary revision of the initial ideas about the concept of a SM. It was stated earlier that the safety requirements specification were concerned mainly with the functionality of the SM itself; and indeed the hazard analysis report contains many proposals about what the SM has to be aware of in order to detect hazards, and actions it should take in the event of a hazard to avoid accident. However, in implementing these proposals the SM depends very much on:

- the information to which it has access—it can only make decisions on the presence of a hazard that is provided with enough information from the rest of the system;
- the amount of control it has over the actions of other

modules in the system—it can only carry out corrective manoeuvres if provided with the necessary supervisory and control abilities.

For these reasons, a major question to address, after completing the hazard analysis, apart from the details of the SM's functionality, is the manner and substance of communications between it and other modules. A safe communication protocol, or at least a communication philosophy, is required for the whole system. Factors influencing this philosophy include:

1. The hazard analysis, the results of which indicate the type of information the SM requires about the actual position of the excavator and state of the immediate environment and the intended imminent actions of other modules.
2. The type of information that needs to be passed between modules, particularly the AM and the track and boom controllers, to achieve basic function.
3. The fact that the SM must maintain safety in the event of component failure and therefore needs to be aware of the status of all other modules in the system.

In short, the results of the hazard analysis must form a part of the input to the entire design process, not just the software design of the SM, if its proposals are to be implemented successfully.

Finally, the hazard analysis provides sufficient information to enable the SM to act as a safety net for the AM, defining where SM intervention is required and how.

7. Conclusions

This paper describes how the safety analysis of LUCIE progressed in practice, and how this fitted into the SM concept around which the design of LUCIE is based. Looking back on the experiences, it is now possible to draw some firm conclusions about the process of hazard analysis.

Most importantly, it is felt that there is certainly a place in the problem domain for hazard analysis at an early stage in the development process, i.e. analysis of functionality. The applicability of a preliminary hazard analyses during early design stages is a well-established concept in safety critical system design [5,6]. In the case of LUCIE, the results of the analysis contain the essence of the SM's functional specifications and hence provide the basis on which operational safety is to be ensured. Furthermore, the analysis was required at such an early stage because the SM was developed as an integral component of the control architecture, rather than as a later add-on for safety assurance.

LUCIE's complex functionality is largely down to the nature of its interface to an unstructured environment. When dealing with systems that interface to unstructured environments, basic functionality is clearly going to be complex and sensitive to the state of the environment. Therefore it is generally expected in these cases, that hazard analysis of functionality will be useful.

In general, it should be emphasised that analysis of functionality can in no way be a replacement for analysis of the design (although the two processes may be able to share some information). In fact the two analyses—of hazards due to gaps in basic functionality and hazards due to internal fault—are complementary, because different results should arise from each. Indeed, the early FTA exposed, internal deficiencies that were to be catered for.

The following observations can be made:

1. It is dangerous to define hazards too precisely at an early stage of the analysis, because this can cause the lack of consideration of hazards that lie outside those definitions, and may lead to assumptions being made about the environment and system functionality that restricts the rest of the analysis.
2. It is essential to have a full understanding of the actual physical causes of a hazard, before going on to suggest ways in which those physical causes might arise given the actions of the system. This suggests that a good comprehension of the dynamics of hazard occurrence should be well understood before any structured analytical technique such as FTA is applied.
3. It is essential to know the basic actions of the system in real time. It is only by knowing what actions are going to be performed and in what circumstances, that one can predict when a hazardous situation might arise.

References

[1] Visinsky ML, Walker ID, Cavallaro JR. Robotic fault tolerance: algorithms and architectures. In: Jamshidi M, Eicher PJ, editors. Robotics and Remote systems for hazardous environments, Englewood Cliffs, NJ: Prentice-Hall, 1993.
[2] Khodabandehloo K. Analyses of robot systems using fault and event trees: case studies. Reliability Engineering and System Safety 1996;53:247–64.
3 Walker ID, Cavallaro JR. Failure mode analysis for a hazardous waste clean-up manipulator. Reliability Engineering and System Safety 1996;53:277–90.
[4] Dhillon BS, Fashandi ARM. Safety and reliability assessment techniques in robotics. Robotica 1997;15:701–8.
[5] Lutz Robyn R. Analyzing Software Requirements Errors in Safety-Critical, Embedded Systems. In: IEEE International Symposium on Requirements Engineering, IEEE Computer Society Press, 1993.
[6] Storey N. Safety critical computer systems, Addison-Wesley, 1996.
[7] Gaskill SP, Went SRG. Safety issues in modern applications of robots. Reliability Engineering and System Safety 1996;53(3):301–7.
[8] Leveson N. Software safety. In: Anderson T, editor. Resilient Computing Systems, London: Collins, 1985. p. 122–43 (chap. 7).
[9] International Electrotechnical Commission IEC 61509—Functional Safety: Safety-Related Systems, Parts 1–7.
[10] Visinsky ML, Cavallaro JR, Walker ID. Expert system framework for

fault detection and fault tolerance in robotics. Computers in Electrical Engineering 1994;20(5):421–35.
[11] Leveson Nancy G. Safeware—system safety and computers. Reading, MA: Addison-Wesley, 1995.
[12] Greenspan Sol, Mylopoulos J, Borgida A. On formal requirements modelling languages: RML revisited. In: Proceedings of 16th International Conference on Software Engineering, IEEE, 1995. p 135–47.
[13] Pace Conrad, Seward Derek. Development of a Safety Manager for an Autonomous Mobile Robot. In: Proceedings of 29th International Symposium on Robotics, Birmingham, 1998.