CHALLENGES FOR DESIGNING INTELLIGENT SYSTEMS FOR SAFETY CRITICAL APPLICATIONS.

Authors: **Chris Harper** **cjharper@avian-technologies.co.uk**

Intelligent Autonomous Systems Engineering Lab

University of the West of England (UWE)

TABLE OF CONTENTS.

1 INTRODUCTION.

This paper aims to discuss some themes on the use of Intelligent Systems in Safety Critical Applications. The author (Harper) is currently researching a PhD on the subject of Design Methodologies for Behaviour Based Systems, with a view to their eventual application in safety critical problem domains.

This work is being carried out at the Intelligent Autonomous Systems Engineering Lab (IAS Lab) at the University of the West of England (U.W.E.). The aim of the Lab is to research and develop engineering principles and methods for designing Intelligent Systems based on newly emerging technologies such as Adaptive Neurocontrol, Collective or Swarm Systems, and Behaviour Based Systems.

The research programme takes the view that the design of large and complex systems involving many interacting agents is best approached by dispersing the responsibility for the processing of information as widely as possible throughout the agents being designed. The "Intelligence" (or any other behavioural property) of the system is thus achieved by the emergent interactions between the agents, allowing the design complexity of individual agents to be reduced, and their inherent reliability to be improved. The research programme is aimed at developing methods of analysis and synthesis that allow this goal to be achieved. The PhD study described above is a part of this overall programme.

1.1 Scope of Paper.

This paper has been written as a response to an e-mail debate that took place in the Safety Critical Systems Mailing List, maintained by the University of York, on or around the period of the 29th September to 2nd October 2000. The subject of the discussion thread was initially titled "Intelligent Systems", but individual messages also had the titles "Intelligent Systems (safety of AHS)", "Safety of Automated Vehicles", "Intelligent Systems (safety of computer controlled cars)", "Automated Highway System Demo", and "Safety of AHS". A number of contributions were made, covering topics such as:

1. The problems facing current Safety Critical Systems (SCS) methodologies in handling the subject of Intelligent Systems.
2. Suggestions for safety management strategies for Intelligent Safety Critical Systems (including the option of backing of such projects altogether).
3. Examples of an industrial problem, which was an experiment in a U.S. Automated Highway Systems research programme.

This paper has been written in the context of e-mail messages that were contributed by the following people (in surname alphabetical order):

- Joseph Ambrosio
- Mike Ellims
- Ken Frith
- Peter Ladkin
- Nancy Leveson
- Bev Littlewood
- James Bret Michael
- Martyn Thomas

This paper has been written in the context of the above persons' letters, and does not address any other contributions to this subject that may have been made.

1.2 Advisory Note.

The ideas presented in this paper should be considered FOR RESEARCH PURPOSES ONLY. My intention is to offer my opinions to those people who are looking at future applications whose entry into service is many years away (e.g. 10 years or more), but who may be looking at the development of technology demonstration prototypes in the short to medium term. I make the assumption that any such prototype systems are kept well away from the public, and that those who use the prototypes do so at their own risk. If your prototype does not meet these criteria, in my opinion you should think long and hard about putting it into operation (for example, trials of car auto-drive systems on public roads).

If you are faced with the requirement for building Artificial Intelligence technology into systems right now (October 2000), then I can only re-iterate what Ken Frith, Neil Hudson, Mike Ellims and Peter Ladkin have suggested between themselves, namely:

(a) You should consider re-designing the system solution to remove the AI component(s), and do something else instead.
(b) If your customers won't let you choose option (a), then you should attempt to isolate or interlock the AI-technology subsystem with non-AI components, based on system-level hazard and safety analyses as per conventional safety engineering practice.
(c) If the application problem is such that you cannot achieve option (b), i.e. if there are no reasonable means to interlock or protect the AI components, then you should withdraw from the project, quoting Codes of Practice, as suggested by Peter Ladkin (and repeated below), if your customers attempt to sue you for breach of contract.

From Peter's message:

> "As far as I interpret the Code of Practice of, say, the British Computing Society, not only is there no obligation to suggest to a customer that one can do the impossible, but rather a positive obligation to be clear that it cannot be done in the current state of the art. Likewise, there is no obligation to accept a contract that asks for the impossible, and a positive requirement instead to say it can't be done at the state of the art. If the customer can't find a safety analyst (and in fact shouldn't be able to do so, if they all subscribe to professional codes of conduct), he/she will either have to give up, or face jail."

The remainder of this paper should be read with the above caveats firmly in mind.

1.3 Background and Structure of the Paper.

Ken Frith began the on-line debate with what was clearly a cry from the heart:

> "The thought of having to apply formal proofs to intelligent systems leaves me cold. How do you provide satisfactory assurance for something that has the ability to change itself during a continuous learning process?"

This paper takes the position that there are two principal reasons why current safety engineering practices do not deal well with Artificial Intelligence technologies such as IKBS, neural networks, and other adaptive system technologies. The first has to do with the way Structured and/or Formal Methods deal with the problems of design and implementation of software. The second has to do with certain traditional assumptions underlying many certification requirements, and the safety engineering methods that have been developed to meet them.

Section 2 contains a discussion of some reasons why current structured and formal methods for software design do not address the subject of intelligent systems.

Section 3 contains a discussion of some topics that affect the perceived dependability of intelligent systems, and some constraints that should be taken into consideration when performing risk assessment on intelligent systems.

Section 4 is a review of the two principal formulations of the Artificial Intelligence problem, namely “Classic AI” and “Modern AI”.

Section 5 discusses the research that is needed to allow analysis of the safety properties of Modern AI systems, with the ultimate view of producing a safety argument for such systems.

Section 6 presents the major conclusions of this paper, Section 7 contains references and finally, Section 8 contains a Bibliography for further reading.

2 SOME REASONS WHY STRUCTURED AND FORMAL METHODS MAY NOT ADDRESS THE DESIGN OF INTELLIGENT SYSTEMS.

Some of the contributions to the discussion on Intelligent Systems, especially a contribution from Peter Ladkin, have focused on the apparent inability of current software development methods, whether they are Structured Methods or Formal Methods, in developing and assuring Intelligent Systems. The aim of this section is to offer an explanation about why this might be so. In order to provide an example of the issues, a technical paper is reviewed that is reasonably typical (in the author's opinion) of research and application of a Formal Method.

2.1 What Aspects of (Software) System Design do Structured and Formal Methods Cover – And What Do They Not Cover?

In order to gain some understanding about why it is not well understood how to develop Intelligent Systems, especially with current system and software development methods, it is worth reviewing the aims and objectives of these methods, and what the current state of the art can and cannot achieve.[1] This review sets some of the scene for later discussions of different paradigms for Artificial Intelligence, and subsequent discussion about working towards production of safety assessments/arguments for such systems.

2.1.1 What Structured/Formal Methods Achieve...

For safety critical systems, errors in the design and programming of software can have very severe consequences, in terms of the number of people injured/killed, or the amount of infrastructure and/or environment damaged or destroyed. For this reason, it is extremely important to have the assurance that software-based systems do not contain any design or programming errors that could cause such accidents. The introduction of inconsistencies, internal contradictions, and other defects, into the software design can cause the system to behave in an unintended and possibly hazardous manner.

Structured Methods, such as the Yourdon Method or JSP, assist software developers by providing a development process framework, which reduces the chance that they introduce errors. However, the notations and procedures defined in structured methods are largely non-mathematical in nature, with the intention of being performed manually by developers, although CASE tools supporting particular methods can offer improved assurance using automated "semi-formal" capabilities for tasks such as consistency checking between software design diagrams.

Formal Methods take the design assurance process even further, by providing a mathematical framework, in which computer programs can be specified in formal logic, allowing the internal consistency and completeness of a specification to be analysed mathematically. The intention is that this should reveal all logical errors in the specification, allowing them to be corrected. Furthermore, many Formal Methods, for example the "B Method", support the formal refinement of specifications into lower level software design descriptions without introducing any further errors (at least as far as the lowest design level that is handled by the method). This capability has made the use of such methods highly recommended for software with very high integrity requirements (SIL levels 3 or 4).

2.1.2 …And What They Do Not Achieve.

The capabilities of Structured Methods, and particularly Formal Methods, are important achievements. The complexity of microprocessor-based electronic hardware, and the complexity of the functional requirements often found in modern systems, makes it is very easy to make mistakes in their construction and programming, and the process of gaining assurance of error-free design extremely difficult. Hence, the significance of these methods (particularly Formal Methods) should not be dismissed. However the power of these methods can sometimes lead one to overestimate the range of tasks to which they are suited. There are some important considerations in (safety critical) systems engineering that they do not address, and for which they may never really have been intended. It is

[1] Much of the content of this section has been inspired by another discussion within the Safety Critical Systems Mailing List, titled "Whether Computer Science has 'failed' Software Engineering". The author would like to acknowledge the contributors to this discussion for their insights.

important to define these considerations, because the principal subject of this paper (how to design and analyse the safety of an “intelligent system”) falls precisely into this domain.

In plain language, Structured and Formal methods tend to instruct developers in how to define a system, not what system to define. Design methods are notational devices. They define languages for the expression of various different levels of design abstraction, rules for ensuring that designs are well-formed in the logic associated with the language, and rules for transformation of one design level into the next without destroying the design properties established in the prior stage. This is commendable, for the reasons put forward in Section 2.1.1. However, these languages and rules have little to say about whether designs being expressed using the methods are going to fulfil their requirements.

By concentrating only on the prevention and removal of errors in the description of a system structure, the methods do not address the problem of removing errors in its behaviour, i.e. how it should interact with its environment. Within many Formal Methods schemes, the problem of behaviour validation (a.k.a. validation of functional requirements) is one of interpretation of models and it is well understood that formal logical systems require a set of basic axioms that must be taken as true without logical proof. In system design problems, some of these basic axioms form the initial statement of its required behaviour. Hence, the 'correctness', however it may be defined[2], of a system's behaviour cannot be formally proven within the method.

What this means is that while one can take a system specification whose behaviour might be known in advance to be satisfactory, and then develop (and verify) an error-free design using a formal method, the inverse operation cannot be done. One cannot take a system, developed using a formal method so that the implementation is consistent with the specification, and then satisfy oneself about its behaviour using the formal method alone.

2.2 Why Current Software Methods Do Not Address System “Intelligence”.

Where all this discussion fits in with the debate on Intelligent Systems is that "intelligence" is generally thought of as a property of a system's behaviour, not its structure (see Section 3 for the two major prevailing formulations of the Artificial Intelligence problem). Hence, conventional design methods, which aim to assure properties in the notation of system design descriptions, cannot address the problem of whether a system's behaviour possesses "intelligence" of sufficient quantity or quality to satisfy owners, customers, certifiers, etc. This issue was revealed most clearly by one particular comment in the e-mail discussion, made by Peter Ladkin:

> "No one has come up with any method yet of doing so that even begins to approach what the word ‘verification’ means in its FM usage." [i.e. with respect to verifying or validating that the “intelligence” of the system does not have properties that introduce safety risks]

An alternative view on this issue was provided by Mike Ellims, who wrote:

> "I feel in general that embedded software [of an AI nature] of itself does not create a hazard. It creates a hazard when it interacts with the outside world (this may not be the case with financial systems). Therefore it is the effects that software errors have on the system as a whole (or some defined system boundary) that have to be taken into account."

This view reiterates the notion that intelligence is an attribute or aspect of system behaviour, and hence outside the problem domain those software design methods (of any kind) are intended to address.

Since they do not explicitly address the issue of behaviour validity, there remains the question of what to do instead to predict and validate the behaviour of an (intelligent) system. The answer to this question is an expansion of the hypothesis of this paper, namely that the reason why current methods are not easily applicable is that they implement “design theory” whereas the question of the definition of intelligence is a problem of “natural science theory”.

2.2.1 “Natural Science” vs. “Design Science”.

This view taken in this paper is that the general nature of system or application problems can be split into two distinct theoretical domains, here called (for want of better expressions) the “natural science” and “design science”

[2] In this case, 'correctness' refers to the behavioural property that one wishes to establish, for example "safety" or “intelligence”.

of a problem. "Design science" is concerned with the issues of how to express problems and refine abstract definitions of problems into implementations. "Natural science" is concerned with the development of theories that allow the explanation and prediction of real-world phenomena, independently of any specific technological issues of how application solutions might be developed and implemented. The term "natural science" has been coined in this paper, because in most cases, the basic body of knowledge for a given problem comes from physics. As may be seen in Section 4, Artificial Intelligence tends to draw from other sciences, such as psychology or biology. These branches of science are often called "The Natural Sciences", so the term has been adopted in this paper as well.[3]

Having a well-developed natural-science theory for a problem is generally a pre-requisite for the development of a dependable application solution, because it is from this domain of knowledge that an understanding of required, valid, and safe/dangerous system behaviour is developed. The study discussed in Section 2.2.2 provides an excellent example of this. The natural science of a problem generally forms the basis for the majority of the functional requirements for a system, which is usually the initiating process for development of a system implementation (with which the "design science" domain is concerned).

The SCS community is naturally concerned with the design science issue of avoiding programming errors in software-based systems, and the development of Formal Methods is the logical conclusion of such concerns. However, the position of this paper is that the sheer complexity and difficulty of developing methods for formal specification and verification of software, and the effort required to achieve such goals in practice, has tended to become a preoccupation for the SCS academic/research community. As a result of this, subjects with more of a natural science flavour, such as AI theories, have been put to one side. No one is to blame for this - it is just a natural result of the research priorities put to the community over the last 20 years or so. However, the problem remains that, whenever customers of a system begin to ask for behaviour that might be termed intelligent, there is no available methodology for dealing with such matters. The opinion offered in this paper is that the research community needs to concentrate on understanding what is needed in the natural science of AI that can allow safety critical systems to be developed.

A research programme for Safety-Critical Intelligent Systems should concentrate on the development of theories that explain and predict "intelligence", such that the behaviour those systems is 'intelligent, and dependably so'. As a starting point down this path, Section 4 provides a review of the two competing bodies of theory that attempt to explain and predict Artificial Intelligence, and Section 5.1 discusses current research that may serve to provide methodologies acceptable for safety critical systems.

Having taken the view that current SCS/FM research does not address issues in the natural science theoretical domain, it would be wasteful if the development of natural science theories for Artificial Intelligence did not exploit the hard work of the last two to three decades. Research aimed at developing natural science theories for system intelligence ought to have a secondary goal of ensuring that existing design science can be used to implement the results. Section 5.2 provides a brief overview as to how well this has been achieved so far.

2.2.2 An Illustrative Example of Recent Software Methods Research and Application.

A good example of how natural science and design science issues are typically blended together in current methodologies is provided in the paper "Requirements Specification for Process-Control Systems" by Nancy Leveson et al. [Ref.10]. This paper was chosen as an example for two reasons: first, Nancy has been kind enough to put this paper on her Web Site[4]. Second, the section structure of the paper actually provides quite a good demarcation between the two theoretical domains. The paper describes a research project into statechart-based languages for specification of systems and analysis of such specifications for properties such as consistency and completeness. The paper then goes on to present an example application, namely a specification for TCAS II, the second generation of Traffic-alert and Collision Avoidance Systems for use in commercial aircraft.

Apart from the specifics of the statechart method (which itself is an interesting subject), the contrast between natural science and design science topics is illustrated by the separation of these two concerns into Sections 3 and 4 of the report. Section 3 covers the subject of "Specifying Requirements for Process-Control Systems", and lays down the natural science basis of the method and/or the application problem (TCAS). There is a discussion of how to model

[3] The names "natural science" and "design science" are only used for want of better terminology. If any readers can think of an improved nomenclature, then suggestions or proposals would be considered.

[4] Go to http://sunnyday.mit.edu/papers.html and look for the above title.

the behaviour of aircraft collision avoidance, and which aspects of the collision avoidance behaviour should be captured as system requirements. A process model was presented for the problem (the traditional closed-loop feedback diagram from control system theory), and the issues surrounding how the process model represents the requirements are addressed. All this discussion sets out the natural science of aircraft collision avoidance problems.

Section 4 covers the subject of "Specification Language", the first few pages of which define what I call the design science basis of the method. The first few pages discuss "design criteria" for the language, providing a list of the properties of the system design description that the method is intended to maintain when transforming requirements into specifications, and specifications into implementations. Such criteria include semantic simplicity, coherence, consistency, and other properties that the authors consider desirable. The remainder of Section 4 goes on to explain in detail how the statechart specification language embodies those epistemic criteria, and lists its specific features and language constructs. This is the design science of the method[5].

2.3 How "Rigorous" Should the Natural Science of Safety Critical Systems Be?

One issue that might be a psychological stumbling point for researchers from the CS/FM domain is the notion that there seems to be no way to achieve the level of rigour in the proof of AI-type systems that is possible for conventional systems constructed using Formal Methods. In a conventional system, one can specify a system, and logically prove that detailed designs and implementations are consistent with that specification. The fact that full mathematical rigour has been brought to bear on the design gives designers, analysts, and certification authorities greater confidence in the claims of safety that are made about such a design.

However, this achievement, while being a laudable one, may have set something of a false standard when it comes to the understanding of how Intelligent Systems should be applied to safety critical problems. The reason for this (which is the recurring theme of the essay) is that Formal and Informal Methods address the design science of systems, not their natural science. These methods provide tools for the translation of descriptions from one form to another (specifications to implementations) and the analysis of epistemic/notational properties of that translation. Once it has been proven that an implementation and a specification are logically equivalent, then that property has been proven for all time (note: the proof applies to the specific version of the system; design changes generate new specifications/implementations, which require new proofs to be generated). Whilst it may be possible to achieve the full rigour of logical proof for the design science purposes (epistemic translation/analysis), in my opinion it is neither possible nor necessary to achieve the same level of rigour when considering natural science issues (theories of situated behaviour).

The view taken in this paper has been derived from the ideas of Popper [Ref.14]. Natural science theories are often, by necessity, partially empirical in nature. That is to say, they are theories based on observations of the real world, and attempt to explain and/or predict the behaviour of objects in the real world through some form of conceptual model (which forms the theory itself). It may be that someone has been clever or fortunate enough to develop a theory from 'first principles', in which case that theory may contain logical proofs deducing its theorems from more 'fundamental' axioms. Nevertheless, the fact remains that the end result is a hypothesis about the real world, whose validity can only be assured by testing, and only remains assured for as long as such tests produce results that agree with the theory. Hence, proofs of natural science theories cannot achieve the permanency of proofs in design science, where proofs remain valid for all time (see pervious paragraph). The most one can ever really do to "prove" a natural science theory is to keep it valid for another day, because nobody can ever know whether tomorrow's test will falsify it.

Whilst one cannot provide a permanent proof for a particular natural science theory, in my opinion such a level of 'absoluteness' is not necessary. Many such theories, whilst not absolutely true, are nonetheless still valuable tools for the provision of solutions to application problems. Natural science theories remain useful even though more complete ones may formally have superseded them. For example, the flight control laws of modern airliners still use algorithms based on Newtonian classical dynamics. This is still done today, even though experiments have shown that Einstein's Theory of Relativity has been proven to be the more complete theory. Yet flight control algorithms based on classical dynamics are still used (and accepted) in the latest design of airliners.

[5] NOTE: It is noticeable that the number of pages devoted to discussion of the design science of the statechart method (Section 4) was approximately 20, whilst the number of pages devoted to the discussion of the natural science of aircraft collision avoidance (Section 3) was approximately 5. This is a crude metric, but may be indicative of the prioritisation of design science over natural science.

The position of this paper is not suggesting that all FCS algorithms should be changed, but rather that the notion of proof (and subsequent acceptability) of natural science theories should be different to our notion of proof and acceptability of system design descriptions. Hence, we must be careful not to reject useful natural science theories for systems by mistakenly transferring the proof requirements of FM (design science) into AI (natural science). What would be a very useful contribution from the Safety Critical Systems and Computer Science communities is a discussion about what level of empirical evaluation (i.e. attempted falsification tests) is needed for a natural science theory to be acceptable for application to safety critical problems. The issue here is about qualitative evaluation of theories of behaviour rather than quantification of statistical parameters such as reliability. The need for (and the difficulty of) obtaining statistical information is accepted fully, but there is also the issue of deciding when one accepts a natural science theory as being dependable, because it has successfully resisted attempts at falsification[6].

[6] Perhaps this discussion could take place via the mailing list?

3 SOME REASONS WHY SYSTEM SAFETY ENGINEERING HAS DIFFICULTY HANDLING INTELLIGENT SYSTEMS.

3.1 Problems with Assumptions Underlying Typical Certification Requirements, and Associated Safety Engineering Methods.

The issue of how to develop Intelligent Systems for safety critical applications raises a number of issues over how certification requirements are currently developed for conventional systems. The 'conventional wisdom' often applied when setting certification requirements for systems (now being thought of as High-Level Arguments for Safety Cases [Ref.11]) may need to be superseded by different arguments for Intelligent Systems. One particular example is an argument from current certification requirements for commercial aircraft. Advisory Material (Joint) AMJ 25.1309 [Ref.9], published by the European Joint Airworthiness Authorities (JAA), offers advice on the interpretation of one of the airworthiness requirements for large aircraft systems (JAR 25.1309). Section 4a sets the overall probability of failure requirement for Catastrophic hazards by using an argument relating to the fraction of large aircraft accidents that are caused by system failures. The essence of the argument is:

> In general, there is one aircraft accident (causing fatalities) approximately every million flying hours accumulated by large aircraft across the world. Of all these accident events, approximately ten percent are caused by failures directly related to systems. The other ninety percent (approximate) of accidents are caused by human error, natural environmental occurrences (e.g. severe weather), and other factors. Given an assumption that there are approximately one hundred possible system failure conditions on a proposed aircraft type, which could cause an accident, if the probability of each single failure condition was less than (1E-6 x 10%) / 100 = 1E-9 per flying hour, then the risk of fatality generated by the proposed aircraft type should not be greater than that which exists in types currently in service. If this probability of failure can be demonstrated for potentially Catastrophic failure conditions, then the proposed aircraft type can be accepted for entry into service.

The first thing to point out about this argument is a major implicit assumption about the nature of the aircraft, namely that it is manually controlled. The systems of the aircraft are servo-actuator systems that respond to commands issued by the pilot(s) via control interfaces such as joysticks, pedals, etc. This can be derived from the fact that the 'system-related failures' category is defined as consisting of failure events other than failures that are related to errors in the interaction between the aircraft and its environment, which are attributed to human error, natural occurrences, etc. The conclusion drawn from this statement is that it is assumed implicitly that the pilot(s) are responsible for all interactions between the aircraft and its environment. Therefore, the systems on board the aircraft are responsible only for transmission of pilot commands from the cockpit to the various actuator mechanisms. This is done using methods such as feedback control to ensure that the command signals are transmitted as faithfully as possible (e.g. minimising position or rate errors or some such measure).

It should hardly be surprising that such an argument has been the traditional argument for establishing the probability of failure requirement for potentially Catastrophic system failure conditions. At the time that this argument came into widespread use, aircraft control systems were largely hydromechanical in nature, employing mechanical linkages from the cockpit controls to drive hydraulic servomechanisms, or employing analogue electrical / electronic mechanisms for functions such as airspeed measurement. Hence, when attempting to define requirements for probability of failure, it was considered appropriate to extract that fraction of accidents that were caused purely by faults internal to systems.

Modern aircraft systems employ digital computers to perform the transmission of command signals to electro-hydraulic servomechanisms, which allows additional functionality (e.g. diagnostics and monitoring) to be inserted into the system. It also allows the introduction of automatic systems, which actually generate aircraft commands on their own "initiative"[7]. Hence, these systems are now capable of generating failure conditions that could only have been generated, in purely manual control systems, by Pilots and/or other persons such as Air Traffic Controllers. Given the introduction of this degree of autonomy, it may be necessary to reassess whether the implicit assumption of manual control underlying the AMJ 25.1309 high-level argument still applies.

[7] Note: this is in fact one definition of an "Intelligent System" (see Section 4), so if current opinion is that there is no way of certifying such systems, what does this say about safety cases for current auto-flight systems?

3.2 Fundamental Limits to the Achievable Integrity of Intelligent Systems.

The opinion expressed in this paper is that it is an open question as to whether it will really be possible to claim a probability of failure of 1E-9 per hour for the autonomous behaviour of an Intelligent System involved in aircraft flight management/control. One of the issues anticipated to emerge from Behaviour-Based (Modern) AI research is an appreciation that the 'correctness' of autonomous behaviour is dependent not only on the 'correctness' of the internal decision-making of the Intelligent System, but also on the 'boundedness' of disturbances imposed on that system by its environment. The behaviour of any system is an emergent property of the interaction between a system's actions and the subsequent feedback from the environment onto the system's state. Unbounded disturbances from the environment may force the system into an 'unsafe' state, even if the system selected a reasonable action at the time. Therefore, even though the system makes no error of judgement, the hazard may still occur.

3.2.1 An Illustrative Example: The Lufthansa A320 Incident at Warsaw, 14 September 1993.

If it really is the case that hazardous behaviour is dependent on factor(s) purely related to environmental disturbances, then there are limits to how much the risks of a system can ever be reduced, even if the system design contains no decision errors. One notable example of a real accident that may be in this category is that of an Airbus A320, which landed at Warsaw Airport on 14th September 1993 [Ref.19] under severe weather conditions, and overran the end of the runway. During the event, the aircraft's braking system did not begin to apply braking force to the landing gear wheels, because the weight of the aircraft was distributed unevenly across the wheels. In addition, load sensors on the wheels, which measure the weight, act as enabling signals for the 'Ground Spoiler' function, which will deploy the spoilers to act as an airbrake when the aircraft has settled onto the ground. Since the weight was not evenly applied to the wheels, neither the wheel brakes nor the spoilers deployed for a considerable part of the aircraft's travel along the runway. By the time these systems did deploy correctly, there was insufficient runway distance remaining for the aircraft to avoid overrunning the end of the runway.

There was some initial speculation that this might be a design flaw in the logic of the brakes/spoiler systems. However, it is also the case that uneven braking force applied to the wheels can cause an aircraft to veer off the side of a runway, and at high speeds this is every bit as severe as overrun at the end. Hence, if wheel braking had been applied during the event in question, another hazard (veering off the side of the runway) could well have occurred. Since one possible escape from this situation is for the aircraft to take off again and attempt to land for a second time, the correct decision for the braking system to take is not to apply the brakes until weight is established evenly across the wheels. Therefore, in my opinion, there was no flaw in the system design.

The reason why this event is so pertinent to this discussion is the reason why the aircraft weight was unevenly distributed in the first place, and the interactions between the aircraft and its environment that led to this condition. The reason for the uneven weight distribution was that the relative airspeed of the aircraft was unusually high. This meant that there was considerable residual aerodynamic lift being generated from the aircraft wings, even though the aircraft was touching the ground. This meant that the aircraft had not settled properly onto its wheels, thus enabling the braking system in the intended manner. This is an example of environmental feedback (the residual aerodynamic lift) generating a disturbance to the system state (a modification to the load measured by the wheels' weight sensors) that exceeded some bound, and even though the braking system made the correct decision at the time (not to apply the brakes until weight distribution is even), a hazard occurred. The principal cause of this event was an error made by Air Traffic Control, in which the aircraft pilots were misinformed about the prevailing wind speed and direction at the airport. As a result they configured the aircraft's speed and bank angle incorrectly for the true conditions on the ground. As a result, the ground speed at landing exceeded the maximum landing speed for which the aircraft was certified, and the bank angle led to the aircraft not settling evenly onto its wheels. This led to events unfolding as described above[8].

[8] This is a synopsis based on the Internet material of [Ref.6], in addition to the author's memory of media reports of the event. If there are any specific errors in the description, corrections would be welcomed. However, the intention in using this example is as an illustration of how environmental disturbances may override the actions of a system, even though there is no error in their design. Therefore, readers are asked to send specific corrections to the author personally, rather than through the Safety Critical Systems Mailing List, unless the correction affects the point of the argument. This will avoid unnecessary e-mail traffic through the List.

3.2.2 Assessment of Environmental Limitations of Intelligent System Behaviour.

One of the benefits of building Intelligent Systems using the Behaviour-Based Systems approach described in Section 3.2 is that it may be possible to get an explicit model of the tolerable limits of environmental disturbance for a given system. In the ongoing PhD studies of the author, a methodology is being developed, in which certain stability properties can be used as measures of the 'dependability' of a system's functionality. One feature of the stability theorems is the property that the stability of a function is preserved as long as disturbances to the function remain within certain bounds, and if the disturbances exceed those bounds then stability can no longer be proven. This may be a mechanism for assessment of how environmental disturbances might be assessed. If models of the relative frequency vs. the magnitude of environmental disturbances can be prepared (by analysis of historical data, for example) then it might be possible to understand what constraints on the achievable level of risk might be for Intelligent Systems. If such a limit exists, then it may not be worth attempting to build an Intelligent System with internal failure probabilities that are lower than say 10% of this environmental constraint, since at this point the environmental risk dominates the system risk. This may be an Intelligent Systems variant of the ALARP concept.

The author’s PhD studies are aimed at development of the basic methodology of how to construct Behaviour-Based Systems using stability properties as the underlying mathematical basis. Therefore, it is not likely that the subject of environmental risk constraints will be studies until the after the Thesis is submitted, which is anticipated to happen next year. However, the subject of statistical modelling of environmental disturbances, with respect to my proposed methodology, is on the list of future research subjects.

4 "CLASSIC" AND "MODERN" FORMULATIONS OF THE ARTIFICIAL INTELLIGENCE PROBLEM.

In section 2, the current difficulties in the problem of developing Intelligent Systems for safety critical applications was identified as a lack of adequate natural science theory for Intelligent Systems, it is necessary to review the current theoretical basis for Artificial Intelligence. There are two major prevailing views. “Classic” AI has been available since the 1950’s, and forms the theoretical basis for Expert Systems/IKBS. “Modern” AI developed in the mid-1980’s, principally from the mobile robotics field. These two formulations of Artificial Intelligence constitute the majority of the natural science theory available to date for Intelligent Systems.

4.1 "Classic" AI: Symbolic Reasoning via Logical Inference.

The traditional formulation of the problem, largely motivated by psychology and related sciences, views "intelligence" as a specific method of computation, namely symbol processing by logical inference and reasoning. The hypothesis is that all real-world problems can be transformed into abstract forms, e.g. planning or game playing, for which the computational mechanisms purported to lie at the heart of any intelligent system (including humans) are especially well suited. The problems of tying sensory inputs to symbols, and the generation of actions, were considered to be merely implementation details to be sorted out in the 'hardware design' (or the equivalent detailed design stage of a system's development). They are not thought to be of essential importance to the generation of intelligent behaviour in any system.

From this philosophical standpoint comes the standard system architectures that characterise the Expert System (or KBS, IKBS, etc.) and its counterpart in Mobile Robotics, often called the Sense-Model-Plan-Act (SMPA) architecture. Broadly speaking, and there are many variations, these architectures process input information and generate responses or actions by a series of general processes, described as follows. The "Sense" process receives inputs from sensors, as in many 'non-intelligent' systems. The "Model" process manipulates these signals (using data fusion techniques) into some form of "world model", a unified set of symbols that represented the state of the system's environment, and on which all subsequent reasoning would be based. The "Plan" process uses a set of logical theorems or axioms about planning, basic axioms for performing logical inference and axioms about what the required behaviour of the robot, to establish a sequence of actions (the 'plan') to perform in order to achieve whatever the required ('intelligent') behaviour. Finally, the "Act" process converts the logical action predicates into whatever physical signals are necessary to drive actuators (again, much the same as non-intelligent systems). Non-mobile expert systems have similar architectures, except that the Sense and Model processes are usually merged into a User Interface Module (or similar terminology), and the Plan process is often split into two processes, the Inference Engine (containing axioms about logical reasoning) and the Rule Base or Knowledge Base (containing axioms about the problem domain).

Symbolic Reasoning has proven successful for a variety of narrowly defined application problems, where the purpose of the system was to manipulate knowledge of some sort (e.g. advisory systems, diagnostic systems, etc.). However, when the approach came to be applied to tasks of a more dynamic nature, such as control of mobile robots, some key problems arose (this overview is based on Brooks [Ref.4]). Mobile robots built with a SMPA architecture worked quite well when placed in a highly uniform environment, such as a room with monotone coloured walls, and were required to perform simple tasks (e.g. stacking blocks) with objects whose shape was simple (e.g. cubic, spherical or wedge-shaped blocks). Robots built with world modelling and action sequence (plan) generation subsystems could perform such tasks using the hardware of the day (and this point will be revisited later) even if they performed somewhat slowly. However, a lot of this success was inadvertently due to the careful engineering of the environments in which they performed their tasks. When the robots were placed in a dynamic environment (i.e. taken outdoors) and asked to perform the same tasks with more complex objects that were typical of everyday objects, they almost completely failed to operate successfully. Typical errant behaviour included remaining motionless, blundering into objects, or some combination of the two. The principal reason for this lay in the (hitherto unappreciated) computational complexity of maintaining accurate and reliable symbolic representations of the real world, when sensory inputs to the robot are changing rapidly. Highly dynamic input patterns forced the robots to one of two conditions. First, they would remain motionless whilst the world modelling subsystem kept revising its models, forcing the robot into a constant state of re-planning. Alternatively, they would develop plans based on a specific symbolic world model, generate a sequence of actions (e.g. movement in a certain direction at a

certain speed), and assume that the symbolic model remained valid in spite of the changing input patterns, which could subsequently render the models and associated plans mutually inconsistent. The result would be that from time to time, a robot would fail to recognise an obstacle and collide with it.

It must be said at this point that these robot experiments were performed in the 1960s and 1970s, at a time when the computational power of microprocessors was (to say the least) a lot lower than their modern counterparts. If one were to build a robot today to perform the same task, using processors such as Pentium III's for example, then it undoubtedly would fare a great deal better, if only because the sheer speed of the processor is so much greater. However, in general, it has been shown that the question of whether plans can be generated in a finite time given arbitrarily changing inputs is computationally intractable (NP-complete), and only becomes bounded if one makes certain assumptions, for example about processor performance. Hence, factors such as processing power will constrain the complexity of domain (in terms of the 'dynamism' of sensory information obtained from such an environment) in which one can reliably use a SMPA-style solution. This is particularly interesting with respect to a particular failure condition that affected the original robot experiments, and may have a modern-day counterpart in some contemporary applications.

The feature of interest in the original robot experiments was that the robots would often be fooled by their own shadows. When attempting to classify an object by its visual image, the shadow cast by that image on the ground would cause the perceived shape characteristics of the object to change, thus prompting the robot to misclassify the object type. This phenomenon would vary with the passage of the day; when the sun was overhead, and the shadows cast were small, a robot might recognise/classify the object correctly. In the early morning or late afternoon, when the shadows cast were long, the robots often misclassified objects quite significantly. Hence, at these times of day, it seem to the robots that objects were spontaneously appearing and disappearing (as they classified or misclassified the visual images), causing even more inconsistency within their world models. This particular failure condition is very interesting when compared with Nancy Leveson's first letter on Intelligent Systems:

> "I was at the demonstration of the automated highway system demo. In the car in which I was driving, the safety driver (sitting in the driver's seat in case something happened) had to grab the steering wheel when the car suddenly turned to the left (the track was a straight stretch of highway that had been closed to all traffic). We asked what had happened. He explained that the vision system got confused when it passed through a shadow (the shadow was maybe 4 inches wide). We asked what that implied if we had travelled through a tunnel or under an overpass -- the reply was that they had not tackled that problem yet. In fact most of the difficult problems were not included in the demonstration, including the problem of merging traffic with non-automated traffic. I was also on the committee that recommended after the demonstration that the automated highway program be killed."

Although the dangers of drawing conclusions from such general descriptions must be acknowledged, Nancy's description of the event suggests that the cause was a highly dynamic change to the image information obtained by the car vision system. This is reminiscent of the original experimental failures with the robots. Assuming that the car system was based on a "classic AI" type of world modelling system[9], it sounds as though the car's intelligent system might have been unable to update its symbolic world model fast enough to cope with the changes to the image. Could this have led to decision errors in its planning system, causing it to generate a steering error?[10]

What this suggests is that, whilst contemporary hardware technology might easily be capable of handling the manipulation of blocks in enclosed arenas, even when outdoors, the requirements of application problems, namely the problem of navigation/automatic control of cars moving at 100kph or thereabouts, have moved on as well. It may be that the processor performance requirements of modern application problems outstrip even modern (128-bit, 4GHz+) microprocessors.

4.2 "Modern" AI: Task-Based Situated Behaviour via Adaptive Control.

The failure of SMPA-based systems to succeed in performing tasks in complex environments prompted some AI researchers to review the underlying theories and assumptions behind "Classic AI", and Brooks [Ref.4] provides a good overview of this process. What became clear was the existence of competing explanations for how intelligent behaviour is generated in animals and humans.

[9] Note that "data fusion" subsystems fall into this category.

[10] Perhaps Nancy (or others) can provide further information about the system architecture in this experiment.

Classic AI was essentially derived from psychology, in which experiments had shown that humans appeared to use logical reasoning processes in tasks that were thought to require high intelligence, and from this came the theory that intelligence was the result of logical inference based on symbolic models. However, the problem with this approach was that many such experiments depended on introspection on the part of an experimental subject (who might be asked to tell the researcher what they were thinking, for example) and the objectivity of such data has been highly suspected. Research into Neurophysiology has shown that the real way in which a biological brain processes information follows a highly decentralised model. For example, experiments with human responses to optical illusions have revealed that the human visual cortex processes an object's form independently from its motion and colour without combining them into a single integrated representation [Ref.17]. A particularly interesting example of this decentralisation is the investigation of "blindsight". This is a visual impairment in which affected people cannot recognise objects, and therefore consider themselves blind. Nevertheless, they can still react on a more primitive level to visual objects, when performing simple reflex-oriented tasks such as flinching from an object that appears suddenly in front of them. When asked to explain how they are capable of avoiding potential visual obstacles, whilst being nonetheless incapable of identifying the specific nature of an object, they can provide no rational answer. This example shows the problem of relying on introspection as a mechanism for revealing how intelligence might be implemented within our brains.

The above experiments also reveal something of the true nature of biological implementations of intelligence. The reason why people with the blindsight impairment could use visual information for one type of task and not another is that the visual cortex connects to different areas of the brain via different paths. The nature of the lesions they received during the accidents that damaged their brains caused one pathway to be severed, whilst the other remained intact. Hence, the brain centres with intact visual information paths could still function, leaving the person able to "see" for the purposes of the tasks performed by the intact brain centres, but blind for the purposes of the tasks performed by the damaged brain centres (or their connections). This reveals two things about the nature of information processing. First, there is no centralised representation of the environment within the brain. If there were, then the visually impaired persons would either still be able to see (but possibly with less "confidence") or alternatively would not be able to see for any task whatsoever (if the brain damage completely wrecked their unified world representation). The fact that they possess partial visual abilities denies the existence of unified symbolic world models within the brain. The second revelation of these investigations is that the brain displays task-oriented modularity in its structure. Different brain centres appear to perform all the processing necessary for a particular task, and none for anything else. This contrasts sharply with the transformation-oriented modularity of the Classic AI paradigm, in which all tasks are processed using the same generalised information transformations (Sensing, Modelling, Planning, and Action Generation).

This task-oriented modularity has been noted in other biological disciplines as well. Investigations in the field of Ethology (the study of the behaviour of animals in their environment) has revealed this architecture, when researchers have attempted to describe the nature of, and inter-relationships between, the various observed behaviour patterns of animals (e.g. nesting seagulls [Ref.18]). Ethological studies have shown that animal behaviour can be modelled as a layered hierarchical architecture of reflexive behaviours. The term "reflexive" means that actions related to a task can be triggered by specific stimuli, with a fixed relationship between the stimulus and its associated response. This effectively defines a fixed transfer function (stimulus-response relationship, or input-output map) for the discrete behaviour pattern. The array of different behaviour patterns is hierarchical in that discrete groups of primitive behaviours have been observed simultaneously when an animal displays a behaviour that is considered to be more intelligent (or more complex). The conclusion from this is that the brains of the animals are organised in a layered hierarchical manner.

These stimulus-response mechanisms display a property that AI researchers call *situatedness* - their actions only produce intelligent behaviour when applied in the appropriate environment (the "situation"). The behavioural properties of the mechanism are *emergent* - they are not explicitly encoded within the transfer function of the mechanism. This property allows the internal complexity of the machine to be reduced (even down to state-less functions for simple reflexive behaviour patterns). Whilst this might appear to be a weakness, in that changes to the environment could invalidate behavioural properties (e.g. intelligence), the brains of humans and animals are adaptive, and can change their transfer functions to restore these properties.[11]

This alternative view of how intelligent behaviour is generated forms the basis of what is called "Modern AI" in this essay, but which the AI community refers to by a number of names - Behaviour Based AI, Simulation of Adaptive

[11] Note that attempting to overcome the above weakness by building state machines with state variables that encode the nature of the environment re-introduces the concept of world modelling using logical symbols, leading to the symbol grounding problems experienced by the Classic AI community (see Section 4.1).

Behaviour, Animats, and others. Rodney Brooks wrote one of the earliest papers to espouse this idea, by proposing a new system architecture called Subsumption Architecture [Ref.5]. This is a layered architecture, in which a system is comprised of modular components, called Behaviour Modules, each of which performs a specific global system task. This is in contrast to many conventional system architectures, in which each modular subsystem/component performs a specific type of processing (e.g. input processing, command generation, output processing), for all the global system tasks. By allocating global system tasks to separate modules, specific assumptions about the nature of information required for the task can be made, and the module designed to obtain only the information it needs. Hence, processing of input information can be optimised to the needs of each task, allowing real-time responses to be achievable. Behaviour Modules operate concurrently, and sensor information is broadcast to all behaviour modules, and since each module is performing a separate task, there is no need for modules to communicate to achieve the goals. This decoupling of modules allows different modules to operate without any need for synchronisation, removing the need for many of the mechanisms in a conventional computer system (typically within the operating system) that make them so complex. It is an opinion of this paper is that this complexity reduction is one of the principal reasons this architecture might be very beneficial for safety critical applications. Conflicts between different Behaviour Modules (where they might need to control the same actuators, for example) are resolved by an Arbitration Network, in which modules belonging to higher layers within the Subsumption Architecture suppress the outputs of lower-level modules, and transmit their signals to the actuators instead. Note that arbitration only occurs between the outputs of modules; the network does not force modules to communicate or synchronise their activity. This works because the capabilities of modules in higher layers have been built to 'subsume' the capabilities below them in the hierarchy. This means that higher layer modules achieve more "intelligent" global system behaviour by allowing lower level modules to control the system most of the time (thus subsuming their behaviour into their own). Suppression of lower modules by higher ones only occurs in the situations when different behaviour is needed to achieve the higher global capability. The global system behaviour displayed at any given time is therefore an emergent property of the set of modules that are active.

Many variations of architectures have been proposed (see the Bibliography and References), and are too numerous to mention here. However, two particularly interesting variants are an improved version of Subsumption Architecture proposed by Jonathan Connell [Ref.6], and a neural architecture proposed by Randall Beer et al. [Ref.3]. Connell's improved Subsumption Architecture achieves a number of properties that, whilst advocated by Rodney Brooks in his original version, were never completely realised. In particular, the original architecture still contained some transformation-oriented design features, where information was processed by several modules in sequence within a given layer. Some modules also retained a few symbolic representations. Connell's version eliminated these holdovers from traditional approaches, and produced an architectural scheme with full decoupling of behaviour modules, and almost completely reflexive behaviour modules. He built a mobile robot that could wander around an office environment, looking for empty soda cans, which it picked up with a manipulator arm, and returned to a home base (where presumably it dumped the can into a bin). In no way was the office environment simplified; people would wander into and out of the office at random intervals, and the robot had to (and did) avoid them. According to [Ref.6], you could even hand an empty can to the robot and it would then take it away to the waste bin. According to the reports, the robot suffered none of the problems typically encountered with SMPA-based robots. This experiment remains the most sophisticated achievement of Subsumption Architecture to date. For various reasons (with which the author does not completely agree) the Adaptive Behaviour community has abandoned this line of enquiry and pursued methods based on machine learning.

The architecture that Randall Beer and colleagues [Ref.3] have proposed is a behaviour-based approach to the design of neuro-controllers for artificial creatures. The main experiment described in this reference is a software simulation of locomotion and feeding in an insect, but at the SAB '96 conference (see Bibliography) he presented a hardware implementation of a six-legged insect-like motion platform, based on the earlier work. The approach used was to define neuron-like processing elements that generated specific motion or feeding actions in the simulated insect, and then tie them directly to specific states of its simulated sensory organs (thus producing reflexive behaviours). Neurons could also excite or suppress/inhibit the activity of others, thus generating arbitration between different behaviour patterns. Paired connections between leg sensorimotor neurons could generate cyclic waves of activity in the simulated insect legs, thus producing "tripod gait" leg movement patterns typically found in real insects. This behaviour was reproduced in the hardware platform, along with balance control reflexes and other quite sophisticated capabilities.

5 METHODOLOGICAL RECOMMENDATIONS FOR ARTIFICIAL INTELLIGENCE IN SAFETY CRITICAL APPLICATIONS.

Consideration of the parameters of SCS research into the "natural science" of Intelligent Systems generates two immediate questions:

1. What "natural science" is used by SCS engineers/researchers in order to establish the safety properties for current systems?
2. Is current natural science still relevant for Intelligent Systems?

The answer to Question 1 as indicated so clearly in [Ref.10] is that conventional control theory is the science chosen for (in my experience) the majority of safety critical applications. One notable exception to this might be Medical Diagnostic Systems and problems of a similar nature.

The opinion of this paper on the answer to Question 2 is that traditional control theory (current natural science) is not the best natural science for the development of Intelligent Systems (safety critical or otherwise). Furthermore, "Classic" AI theory (based on Symbolic Reasoning) as described in Section 4.1 is not the best natural science for Intelligent Systems either, at least for problems related to interaction with complex dynamic environments (e.g. vehicle controllers). The opinion is that "Modern AI" theory as described in Section 4.2 appears to be a more suitable natural science for safety critical Intelligent Systems. Where systems are used in a secondary role for advisory functions or information/knowledge manipulation (e.g. medical diagnosis), then extensions of Classic AI (see Section 5.1.2) may be acceptable. These methods allow for explicit symbolic representations of “safety” or “risk” in the knowledge used, and inferences made, thus allowing the possibility of assessment of the causes of unsafe decisions.

5.1 Natural Science Recommendations for Intelligent Safety Critical Systems.

Future projects involving the development of Intelligent Systems should take care to define the theoretical basis for intelligent behaviour that an appropriate natural science of the application problems being solved. This section contains recommendations for selection of such theoretical models.

5.1.1 Intelligent Information Systems (IKBS/Expert Systems).

For applications such as Intelligent Medical Diagnostic Systems, Hazard Advisory Systems, and similar applications, Classic AI theory may still be an acceptable natural science. The problem is one of knowledge manipulation rather than adaptive behaviour, and whilst the Modern AI formulation may argue that the majority of behaviour of intelligent creatures is not based on symbolic reasoning, it is true of humans at least that some behaviour is generated in this way[12]. Therefore, established theories for symbolic reasoning may still be valid in such applications. Classic AI systems generally require a domain knowledge base on which to base their inferences, and this must be defined by information models that have come from the relevant application domain. A great deal has been written on the development of expert systems (IKBS), so readers are recommended to consult the established literature for further reading.

5.1.2 Intelligent Autonomous Systems (Mobile Robots).

The majority of systems considered safety critical are considered to be so because they drive physical machinery that is capable of causing damage to people and/or the environment. Any such application is by nature an embedded control application, in which systems may have to respond in real time, otherwise an accident may occur. For any

[12] For example, this paper itself a symbolic representation used for the task of communicating abstract concepts, and the act of writing it is clearly typical of human intelligent behaviour [at least the act is, if not the author himself :-)].

Intelligent System to be developed for such an application, the opinion of this paper is that Modern AI is the better theoretical model to apply as the natural science basis for the system.

Modern AI framework produced more successful results for those aspects of intelligence that imply interaction with the real world (instead of pure knowledge manipulation), and the body of experience of the poor real-time performance of Classic AI weighs in favour of the newer theory.

The recommendation of this paper to anyone working on current projects *(N.B. prototypes not products! - see section 1)* is that they look at the bibliography and references to get a feel for the general ideas, and then try to define the system as a hierarchy of task-based modules. Hopefully, this should produce more dependable results, and be easier to analyse for safety (at least in qualitative terms), than traditional Classic AI architectures. However, it must be noted that Modern AI is a much newer field than Classic AI, and that a sound theoretical basis (i.e. based on well-founded mathematical theories) for the field is still emerging. The most promising theoretical basis appears to be the use of proofs of stability as the basis for system design. Three particular studies provide a reasonable overview of what the author believes to be the most rigorous theoretical approaches:

5.1.2.1 The Method of Steinhage, Menzner, and Erlhagen.

Axel Steinhage, Rainer Menzner, and Thomas Erlhagen, of the Ruhr University of Bochum, Germany, have proposed one method for the design of system architectures for intelligent autonomous behaviour in robots [Ref.12]. The methodology aims to construct robots with stable behavioural properties, allowing arbitration for actuator control, between different behaviour modules, to be configured into a variety of schemes (e.g. mutual exclusion, summation of actuator state vectors). The transfer function of each behaviour module is designed such that the motion of the system state trajectory is stable with respect to one or more attractor states, and unstable with respect to one or more repulsor states. This is an implicit form of Lyapunov stability proof for the transfer functions of each behaviour module. These states are selected to generate the desired behaviour for the system being proposed. The methodology, and its supporting implementation on a mobile robot, has been developed, but based on the description in [Ref.12] there does not yet appear to be any scheme to identify specific attractor/repulsor states from more generalised specifications of behaviour (i.e. functional requirements). However, the mathematical basis of this work does make it one of the more promising emerging methods.

5.1.2.2 Methodologies based on Viability Theory.

Viability Theory, developed by Ashby in the 1950's [Ref.1], has been proposed as a theory for Animat Design both by Jean Arcady Meyer [Ref.13] and Jean-Pierre Aubin [Ref.2]. The theory is in effect an extension of Lyapunov stability theory, to cope with more generalised problems than might be typical of traditional control theoretic applications. The aim of viability is to design a transfer function whose resultant state trajectories converge on specified goal states, whilst avoiding specified hazard states. Viability Theory proposes some mathematical properties of the transfer functions that yield the required goal convergence and hazard avoidance properties, whilst identifying the boundary conditions in state space, within which the properties remain valid. Some application of Viability Theory to animat simulation design has been carried out, but the method as defined in [Ref.2] appears to be very intensive computationally, and may require more refinement before it becomes practical as a design tool. However, fact that Viability Theory is a more generalised form of Lyapunov Stability Theory means that it may become the preferred basis for the design of behaviour based systems.

5.1.2.3 The Space-time Distance Design Methodology.

Space-time Distance Design is a methodology being developed by the author at the University of the West of England (UWE) [Ref.8]. The methodology will address the problems of behaviour based systems design on two levels. First, the design of individual behaviour modules is achieved by using a modified form of Lyapunov stability analysis. Traditionally, Lyapunov stability analysis has been used to analyse the behavioural properties of an existing transfer function. In this methodology, Lyapunov functions are used as the basis for defining a function in the first place, i.e. the theory is used as a synthesis procedure instead of an analysis procedure[13]. The advantage of

[13] A first attempt to develop this methodology was made by the author as part of the DTI ISSAFE Project. However, this original work was not particularly successful. The PhD study has corrected the problem by developing an extension to the basic Lyapunov stability theorem, which makes the practical application of the method much more feasible. Requests for information about the project should be made to the author, via the e-mail address on the front page of this report.

this method is, like the other two methods in the preceding sections, that a rigorous mathematical property is established for the system behaviour patterns expressed in the modules.
The second part of the approach is the use of an emerging concept, called the Space-time Distance Principle, as a scheme for explaining how the overall behaviour of a system can be defined as a set of specific behaviour patterns of individual modules. The general principle is that the higher priority behaviour modules within Behaviour Based Architectures are responsible for behaviour at longer spatial distances or time-scales (note that 'space' can mean state space as well as physical space). The concepts of spatial distance and time are handled as different parameters of the same state vector, which is why the name of the principle refers to 'space-time'. Longer spatial distances and time-scales are then represented as Euclidean Distances in the space-time vector for the system.

The use of the Space-time Distance Principle to identify behaviour modules and their place in an overall architecture allows that architecture to be developed from a more general set of customer requirements. This capability is the distinctive feature of the methodology, which sets it apart from the others. In principle, it seems quite feasible to substitute the detailed design methodology of Steinhage et al., or the Viability Theory methods of Aubin and Meyer, for the Lyapunov-based design techniques in this method. However, the Space-time Distance concepts complete the overall design methodology, and this appears to be absent from the other two approaches.

Work on validation of the Lyapunov-based techniques, and on the Space-time Distance Principle, is still in progress. Simulations of a bench-top experiment (the Inverted Pendulum or "Cart-Pole" problem) have been successful, and provide some measure of confidence in the general principles, and the development of a physical version of the experiment is currently being completed. But the work is still far from complete, and it would not be reasonable to say that the principles are properly validated. Future work will move from the bench top to experiments with mobile robots in the laboratory, and the intention over the next year or two is to move to more complex machines (e.g. model helicopters) and with increasing complexity of behaviour patterns. Where the limits of applicability of this methodology are is still unknown.

5.2 Design Science Requirements for Intelligent Safety Critical Systems.

Recommendations about the natural science basis for intelligent behaviour in safety critical systems must be accompanied by recommendations for appropriate design notations when specifying the system and writing programs to be loaded into the computing machinery of an Intelligent System.

5.2.1 Intelligent Information Systems (IKBS/Expert Systems).

This section refers to the use of artificial intelligence techniques as the base technology for information, advisory, or decision support systems, which tend to be the classical applications of expert systems (IKBS). A significant amount of material has been written about the application of structured and formal methods to the design of expert systems or IKBS. Since the aim of this paper is to focus more on behaviour based systems, the reader is referred to the literature for further information on this subject.

In general, the type of computing machinery used for Expert Systems/IKBS is a conventional microprocessor computer, so the issue of whether existing notations being appropriate to the computing machinery of the application does not usually arise. Predicate logic is the typical notation used, and is even embedded into programming languages themselves, as in the Prolog language, for example. Analysis of the software design for logical consistency and completeness should therefore be possible using existing methods. However, readers should be cautioned that poor requirements definition for expert system problems can cause severe difficulties, even if (or perhaps especially if) Formal Methods are used (see [Ref.16] for an example of this). If any readers are looking for guidance on possible methods for development of Intelligent Systems in this domain then John Fox's book "Safe and Sound" [Ref.7] is highly recommended.

5.2.2 Intelligent Autonomous Systems (Mobile Robots).

The subject of which structured and formal methods are appropriate to the design of behaviour-based safety critical systems has not yet really been addressed by the research community, as the basic natural science of behaviour based systems is still the current focus of research. Furthermore, there is considerable variation in the basic computing machinery used in implementations, ranging from microprocessors to gate arrays to hardware neural networks. The design notations used tend to reflect the chosen implementation scheme. Design notations that could be used include:

- Use of propositional logic for specification of behavioural rules, if gate arrays are used as the computing machinery as per Rosenschein and Kaelbling [Ref.15].

- Use of state transition notations if Asynchronous Finite State Machines are used as the computing machinery, as used by Brooks [Ref.5], Connell [Ref.6], and others.

- Dynamical equations modelling the behaviour of neural networks, as used by Beer [Ref.3].

Specifications could be written in these notations, allowing rigorous verification between different stages of development. However, current research has not yet addressed the general lifecycle models for how one might transform requirements into implementations.

6 CONCLUSIONS.

As a result of the earlier discussion in the paper, the following conclusions can be made:

- In the analysis of an engineering problem, a distinction can be drawn between the knowledge of how to solve the problem itself, and the knowledge of how to represent that solution in a notation that is used to develop a system implementation. The former knowledge domain has been called the “natural science” of the problem, and the latter its “design science”. The question of how to design intelligence into a system is a natural science issue rather than a design science one. The problem facing the Safety Critical Systems community is that most existing Computer Science theory that is used for SCS development (e.g. Formal Methods) applies to the issues of manipulating system descriptions (design science). The natural science of application problems is often buried within the development of design science methods, as exemplified by [Ref.10]. Hence, it should not be too surprising that current safety critical system engineering methodology is having trouble dealing with the subject of Intelligent Systems.

- Natural science theories of Intelligent Systems need not be, and cannot be, “absolutely proven” in the same sense as the consistency, completeness, etc. of the description of their design and implementation. The two concepts form different parts of the system solution, and it would be a mistake to attempt to apply the proof requirements of one part to the other.

- Having stated the above point, one should also be careful not to reject the advances in design science just because we are building Intelligent Systems. If, after having defined the requirements and architecture of a system according to natural science theories from AI, it turns out that specifications are produced that are amenable to rigorous design and implementation using Formal Methods then they should be used.

- There may be limits to the reduction in risk that intelligent autonomous systems may ever be able to achieve. Any system interacting with its environment receives a feedback from it, and system disturbances generated through that feedback may overwhelm the intelligent (or safe) behaviour generated by the system, regardless of whether the behaviour is “correct” (in terms of intelligence or safety) or not.

- Existing experiments in Intelligent Systems may be using the wrong underlying AI theories, as discussed in Nancy Leveson's letter to the Safety Critical Systems Mailing List on the Intelligent Highway Systems project. Behaviour-Based (Modern) AI solutions may offer more dependable system behaviour, and may allow rigorous analysis and design of safety properties.

- Design methods for Behaviour-Based AI are still emerging, and for this reason their application to practical industrial projects should be considered to be some years away (perhaps as much as 10 years). However, to date the prospect is good that methodologies with sufficient rigour for application to safety critical problems will eventually be developed.

7 REFERENCES.

[1] Design for a Brain
W. Ashby
Wiley 1952

[2] Elements of Viability Theory for Animat Design
Aubin, Jean-Pierre
From Animals to Animats 6
Proc. 6th Intl. Conf. Simulation of Adaptive Behaviour (SAB) 2000
MIT Press 2000

[3] Intelligence as Adaptive Behaviour: An Experiment in Computational Neuroethology
Randall D. Beer
Academic Press 1990
ISBN 0-12-084730-2

[4] Intelligence Without Reason
Rodney A. Brooks
Cambrian Intelligence: The Early History of the New AI
MIT Press 1999 ISBN 0-262-02468-3

[5] A Robust Layered Control System for a Mobile Robot
Rodney A. Brooks
Cambrian Intelligence *(op.cit.)*

[6] Minimalist Mobile Robotics: A Colony-style Architecture for an Artificial Creature
Jonathan H. Connell
Academic Press 1990
ISBN 0-12-185230-X

[7] Safe and Sound: Artificial Intelligence in Hazardous Applications
John Fox and Subrata Das
AAAI/MIT Press, 2000
ISBN 0-262-06211-9

[8] Designing Behaviour Based Systems Using The Space-time Distance Principle
Christopher Harper and Prof. Alan Winfield, Univ. of the West of England
Proc. Conf. Towards Intelligent Mobile Robots (TIMR) 2001, *to appear*

[9] Advisory Material Joint (AMJ) 25.1309 (Revision 4)
Joint Aviation Authorities, October 1989
Available in the UK from: Civil Aviation Authority, Gatwick

[10] Requirements Specification for Process-Control Systems
Leveson N.G., Heimdahl M.P.E., Hildreth H., Reese J.D.
IEEE Trans. SE, Vol.20, No.9 (September 1994), pp 84-107
Also available for viewing online at Nancy Leveson's web site: http://sunnyday.mit.edu/papers.html

[11] Safety Case Management
John McDermid and Tim Kelly
UK Safety Critical Systems Club Seminar
15 October 1999 London, UK
SCSC Web site: http://www.safety-club.org.uk/safeclub.html

[12] Generating Interactive Behaviour: A Mathematical Approach
Menzner, Steinhage & Erlhagen
From Animals to Animats 6 (*op.cit.*)

[13] Simulation of Adaptive Behaviour in Animats: Review and Prospect
Jean-Arcady Meyer and Agnes Guillot
From Animals to Animats, Proc. 1st Intl. Conf. On Simulation of Adaptive Behaviour, pp2-14
MIT Press 1991

[14] The Logic of Scientific Discovery
Karl R. Popper
Routledge 1980
ISBN 0-415-07892-X

[15] Action and Planning in Embedded Agents
Leslie Pack Kaelbling and Stanley J. Rosenschein
Designing Autonomous Agents: Theory and Practice from Biology to Engineering and Back (ed. Pattie Maes)
Pages 35-48
MIT/Elsevier 1990
ISBN 0-262-63135-0

[16] Quality Measures and Assurance for AI Software
John Rushby
Contractor Report 4187
NASA, 1988

[17] The Visual Image in Mind and Brain
Semir Zeki
Scientific American, Vol.267 No.3, pp42-51
September 1992

[18] The Use of Hierarchies for Action Selection
Toby Tyrell
Adaptive Behaviour, Vol.1 No.4
MIT Press 1993

[19] Report on the Accident of Airbus A320-211 Aircraft in Warsaw

- Transcript available at:
http://www.rvs.uni-bielefeld.de/publications/Incidents/DOCS/ComAndRep/Warsaw/warsaw-report.html
- See also the Aviation Safety Network web site: http://aviation-safety.net/database/1993/930914-2.htm

8 BIBLIOGRAPHY.

Many of the References are anthologies or conference proceedings, and contain many interesting papers that are not referenced by this paper. Readers are encouraged to explore these other articles. If readers are interested in finding out more about the Adaptive Behaviour field of Artificial Intelligence, I recommend the following books, in addition to the references:

- From Animals to Animats: Proceedings of the International Conferences on the Simulation of Adaptive Behaviour (SAB'90 – SAB2000):
  From Animals to Animats, MIT Press 1991, ISBN 0-262-63138-5
  From Animals to Animats 2, MIT Press 1992, ISBN 0-262-63149-0
  From Animals to Animats 3, MIT Press 1994, ISBN 0-262-53122-4
  From Animals to Animats 4, MIT Press 1996, ISBN 0-262-63178-4
  From Animals to Animats 5, MIT Press 1998, ISBN 0-262-66144-6
  From Animals to Animats 6, MIT Press 2000, ISBN 0-262-63200-4

- Understanding Intelligence
  Rolf Pfeifer, Christian Scheier
  MIT Press 1999
  ISBN 0-262-16181-8

- Behaviour-Based Robotics
  Ronald C. Arkin
  MIT Press 1998
  ISBN 0-262-01165-4

- Vehicles: Experiments in Synthetic Neuropsychology
  Valentino Braitenburg
  MIT Press 1984
  ISBN 0-262-52112-1

- Computational Theories of Interaction and Agency
  Eds. Agre P.E., Rosenschein S.J.
  MIT Press 1996
  ISBN 0-262-51090-1