University of Glasgow

Wednesday XX XX 2017
XX.XX-XX.XX
(Duration: 2 hours)

DEGREES OF MSci, MEng, BEng, BSc, MA and MA (Social Sciences)

Safety-Critical Systems Development (H)

(Answer 3 out or 4 questions)

This examination paper is worth a total of 60 marks

The use of a calculator is not permitted in this examination

INSTRUCTIONS TO INVIGILATORS
Please collect all exam question papers and exam answer scripts and retain for school to collect. Candidates must not remove exam question papers.

1. (a) Briefly explain how perception, cognition and physiology influence the safety of complex, interactive software systems.

[4]

[Seen problem]

Perception involves the detection of change in the environment using our senses; if operators do not perceive critical information in their context then they are unlikely to accurately identify the state of complex, interactive software systems (1 mark). Cognition involves the processing of information perceived in the environment, it covers decision making and planning (answers do not necessarily need to adopt a pipeline model) (1 mark). If operators detect information but cannot understand it or do not form correct intentions for action then mistakes can occur (1 mark). Other answers might refer to slips and lapses (1 mark). Finally, physiology refers to the physical characteristics of the operators. If individuals cannot physically operate interface devices (reach controls, see display panels) then errors are likely to occur (1 mark). Another mark could be awarded for answers that identify interconnections.

(b) The UK Office of Rail Regulation publishes guidance on Managing Rail Staff Fatigue. This stresses that fatigue increases the likelihood of errors and adversely affects performance, especially in tasks that require vigilance and monitoring; decision-making; awareness; fast reaction time; tracking ability; memory.

Briefly explain how software might be used to mitigate the impact of fatigue on each of these types of task.

[6]

[Seen/unseen problem]

Software can be used to support vigilance and monitoring by requiring periodic input leading to an alarm or automatically putting the system into a safe state if the operator does not respond within a limited time period. The 'dead mans handle' on rail systems is an example (1 mark).

Software can be used to support decision-making by offering context dependent help. Other answers might refer to feed forward, prediction systems that show operators the anticipated consequence of particular commands through simulation. These planning tools are used in Air Traffic Management (1 mark).

Software can be used to support awareness – by highlighting key changes in the underlying state of an application, including trends that take significant periods of time to develop (1 mark).

Software can be used to support fast reaction time – users can be reminded to provide input if they are slow in responding. Automated and default responses can be pre-programmed to ensure the safety of a complex application (1 mark).

Software can be used to support tracking ability – as in many of the previous examples, training and simulation can help operators improve and provide feedback on previous performance (1 mark).

Software can be used to support memory in many different ways, context sensitive menus of commands can guide operators so that they do not have to remember which commands

can be applied in a particular state, other answers might refer to help functions etc (1 mark).

(c) Automation is frequently cited as a way of mitigating the impact of human error, by reducing the operator's ability to intervene with safety-critical applications. Explain some of the problems that have been encountered when fully autonomous systems are integrated with more conventional applications.

(Hint: illustrate your answer using case studies that we have met on the course in the safety engineering of either autonomous cars or Unmanned Airborne Systems)

[10]

[Unseen problem]
There are many different ways of answering this question. We have looked at some of the issues involving both autonomous cars and RPAS (Remotely Piloted Airborne Systems)/UAS. In both cases there are similar concerns. The software must form a model of different possible human behavior [2 marks]. Conversely, human operators must be able to form accurate predictions about the likely behavior of an autonomous system [2 marks]. In both domains, the human operators of other vehicles can act in ways that are hard for an autonomous system to anticipate [1 mark]; drivers routinely violate road traffic rules so autonomous systems cannot assume that other human drivers will always follow the rules of the road [1 mark]. The autonomous application must account for human error and negligence but also the spontaneous human response to degraded modes and emergency situations that would then have knock-on effects for any automated system[1 mark].

On the other hand, human operators must be able to anticipate the behavior of an autonomous system. In the aviation domain, we must support predictions by other aircrew and also by ground-based air traffic controllers [1 mark]. For autonomous vehicles, it should be possible for other drivers to adjust their driving to accommodate the likely behavior of this new class of vehicle [1 mark]. Equally, pedestrians must also be able to anticipate the ways in which an autonomous vehicle will respond to evolving traffic patterns [1 mark]. Ideally, it should not be possible to distinguish between the performance of human and autonomous systems [1 mark]. However, as noted above, this can be extremely difficult to achieve in degraded modes or emergency situations especially when this may involve autonomous systems being programmed to violate legal requirements in order to save life [1 mark] (there are a host of further ethical dilemmas that might also be mentioned). Marks totaled together up to a maximum of 10.

2. (a) What are non-functional requirements?

[6]

[Seen problem]

Functional requirements specify what a system should do [1 mark]. Non-functional requirements such as safety and cyber-security [1 mark] provide additional constraints that help determine the quality of any implementation [1 mark]. They are problematic because they are relative and not absolute concepts [1 mark]. It is impossible to be totally safe or totally secure and yet still satisfy functional requirements [2 marks]; for instance, a totally safe aircraft would be grounded [1 mark]. Other solutions are possible,

(b) Explain the problems that arise when integrating cyber security concerns into safety-related risk assessments.

[7]

[Seen/unseen problem]
In conventional safety assessments risk = probability x consequence. Cyber risk assessments usually introduce additional terms into this formula – for instance, vulnerability [1 mark]. In the context of safety, we can use statistical data to guide these assessments by looking at previous incidents and accidents we can ground our predictions [1 mark]. This is far more difficult in cyber-security where the problems of under-reporting are compounded by the possibility of new forms of attack (zero day exploits [1 mark]) that are the result of human agency actively seeking vulnerabilities [1 mark]. Safety related risk assessments usually assume random stochastic probabilities with a high degree of independence [1 mark] – in other words the probability of a second failure is not significantly influenced by any other failure [1 mark]. This is clearly not the case in security where blended attacks explicitly depend upon coordinated threats [1 mark]. Much work remains to be done in the mathematics of combined safety-cyber assessments [1 mark]– for example, helping us to assess the probability that an attack might be launched to coordinate with a degraded mode or other form of random failure [1 mark]. More sophisticated answers could go on to argue that without considering cyber threats existing safety related risk assessments are unlikely to accurately capture the potential hazards to complex, critical systems [2 marks], hence there is an urgent need to resolve some of the issues identified in this question.

(c) The US Food and Drugs Administration (FDA) recently issued a warning about the cyber-security of infusion pumps; these deliver drugs to a patient at a pre-programmed rate for fixed periods of time. Some infusion pumps can be programmed remotely through a health care facility's Ethernet or wireless network. An unauthorized user with malicious intent could access the pump remotely and modify the dosage it delivers, which could lead to over- or under-infusion of critical therapies.

Goal Structuring Notation (GSN) has been proposed as a means of integrating cyber-security concerns into the engineering of safety-critical systems. Describe how this approach might be used to combine evidence about safety and security to improve our confidence in the software engineering of the infusion pumps examined by the FDA.

[7]

[Unseen problem]

It is not necessary to sketch a GSN diagram but if one is provided it should attract up to three marks if correct and also if it is linked to the question [3 marks]. Only one mark should be allowed if it simply dumped into the solution without addressing the integration of safety and cyber concerns into a more unified approach to design/development [1 mark].

There are a number of different approaches here and each would attract full marks – one would be to maintain two separate GSNs for safety and security [2 marks]. This could still be argued to provide a degree of integration because the same notational technique is being used for both concerns [1 mark]. GSN has been used for both purposes and there are numerous examples of this approach on the web.

An alternate approach would be to use cyber-security concerns to challenge or undermine the evidence that is included as leaf nodes in a convention safety GSN [2 marks]. For instance, any assertion that particular safety risks have been mitigated could then be challenged by evidence that a cyber attack might undermine the techniques used to protect an application [1 mark]. In our example, lab tests to demonstrate that the infusion device could be attacked in the manner described by the FAA would be associated with the leaf nodes claiming that hazards associated with the design of the pump had been mitigated. The best answers are expected to form these links between the use of GSN and the FAA case study [up to 3 marks for this].

A final alternative would be to develop a security oriented GSN and then to consider the safety impact of an attack [1 mark]. I am less keen on this approach because it is unclear whether it could be extended to capture a broad range of hazards that are not directly connected to cyber threats. Also, I would expect some attempts to show what the syntax of this might look like as there are few (no?) documented examples [2 marks for this].

3. (a) How do safety kernels improve the reliability of critical software applications?

[4]

[seen problem]
Safety kernels provide an architectural means of grouping critical code [1 mark]; this is helpful in numerous ways. Development resources are allocated in proportion to the criticality of the code, hence it is to be expected that more attention be paid to the kernel [1 mark]. Regulators and IV&V teams similarly know where to focus most closely [1 mark]. There may be particular development procedures and requirements triggered by changes to the kernel in order to maintain the quality of the code (eg additional reviews etc) [1 mark]. Kernels also simplify traceability between safety requirements and their implementation [1 mark]. A range of other reasons can be given – marks to be allocated for accuracy and quality of the argument in support of the kernel approach.

(b) Safety kernels are often resident in non-volatile ROM or in another form of protected memory. There is also often a requirement that the kernel is implemented so that it cannot be corrupted, delayed or substituted by any other program in the system.

Briefly explain the importance of each of these requirements for the implementation of safety kernels in high reliability software systems.

[6]

[unseen problem]
Safety kernels are often resident in non-volatile ROM or in another form of protected memory – this is important to prevent critical code from becoming over-written by application code or data [2 marks]. There is also often a requirement that the kernel is implemented so that it cannot be corrupted – if a kernel were to be corrupted then it is likely that significant safety requirements would not be met [1 mark]. This critical code should not be delayed – if kernel functions are delayed then real-time safety requirements would be violated with necessary recovery actions being taken too late to protect an application process [2 marks]. It is important that the kernel is not substituted by any other program because as mentioned in part a) development resources are focused on the kernel and any other program may not achieve required levels of reliability to meet the requirements for the safety functions [1 mark].

(c) Software testing is 'necessary but not sufficient' for the verification and validation of safety-critical systems. Justify this statement and explain how a range of other techniques can be used to ensure that software meets safety requirements in complex, safety-critical applications.

[10]

[seen/unseen problem]

During the course we have discussed Dijkstra's maxim 'testing proves the presence of bugs and not their absence' on many occasions [2 marks]. We have looked at issues of coverage [1 mark]; where the scale and complexity of modern software prohibits the use

of exhaustive testing [1 mark], leaving aside the ethical issues in safety domains [1 mark]. We have also looked at particular techniques that help to address some of the associated concerns; including fault injection [1 mark].

Testing is necessary but it cannot provide guarantees that code will be totally correct. One way to develop the answer would be look at other forms of IV&V activity – including the use of formal methods and peer reviews to add assurance to the delivery of safety-critical software [2 marks].

Another important set of points could be made around the relationship between requirements and testing [1 mark]. If the requirements are incorrect then tests are unlikely to identify these problems – especially if verification focuses on demonstrating that software satisfies the requirements [2 marks]. In other words, additional forms of validation must be used to support the verification of particular requirements [1 mark].

4. 'Incident analysis provides greater insights into future risks than accident investigations involving safety-critical software.'

State whether or not you agree with this statement and illustrate your answer with examples drawn from the course.

(Hint: do not attempt this question unless you have done some of the preparatory reading in this area)

[20]

[Unseen problem/essay]

There are many different approaches to this question. Marks will be awarded in proportion to the quality of the argument rather than to particular opinions about the topic as both accidents and incidents contribute to the development of safety-critical systems and there is no systematic way to determine whether one or the other class of adverse events contributes more in any objective sense.

Accidents are important because they act as focus of public and government attention [1 mark]; usually providing a catalyst for safety improvements [1 mark] that might otherwise have waited months or years to be implemented [1 mark]. Often they trigger changes that go well beyond the narrow causes of a particular failure [1 mark]. For instance, the Piper Alpha accident revolutionized the safety regulation of offshore oil and gas production with responsibility for safety being separated from the promotion of the industry [2 marks].

On the other hand, accidents are often argued to be atypical [1 mark]. The low frequency of adverse events means that we have very little feedback on the safety of complex

software systems [1 mark]. Hence to make any safety improvements it is better to look at the large number of near misses [1 mark]. This provides additional benefits because simply reacting to previous accidents may not prepare us for new forms of future failure [1 mark]. Near miss incidents help us to intervene BEFORE an accident occurs [1 mark].

In the course we covered precursor indicator models that combine both incidents and accidents [1 mark]– the causal and contributory factors of previous major accidents are monitored even when they do not lead to a major failure (ie they are near miss incidents) because they provide an indication of the potential for a repetition of that accident in the future [2 marks]. An example would be the enumeration of Signals Passed at Danger (SPADs) where no accident occurred [1 mark]. In terms of software, we might study the performance of exception handlers as a trigger to improve particular routines [1 mark].

First class answers might extend the argument to consider resilience engineering and the promotion of techniques derived from instances where things went right in everyday operation rather than a very small number of accidents or near misses [2 marks]. In the course, we have looked at the application of resilience engineering with the International Space Station, this could be brought into these more advanced answers [2 marks].