Xday, XX XXX 2010

9.30 am - 11.15am (check this!)

University of Glasgow

DEGREES OF BEng, BSc, MA, MA (SOCIAL SCIENCES).

COMPUTING SCIENCE - SINGLE AND COMBINED HONOURS
ELECTRONIC AND SOFTWARE ENGINEERING - HONOURS
SOFTWARE ENGINEERING - HONOURS

SAFETY-CRITICAL SYSTEMS DEVELOPMENT

Answer 3 of the 4 questions.

1.

a) The development of the European Geostationary Navigation Overlay Service (EGNOS) and Galileo systems will revolutionise the critical infrastructures of many European states. The encrypted Galileo Public Regulated Service (PRS) and Safety of Life Service (SoL) provide reliable services to a range of end-users including the police and military as well as transport applications, such as air-traffic management. However, there are significant safety concerns. For example, the additional accuracy offered by overlay architectures has supported the development of EGNOS Controlled Rail Applications in which the opening and closing of railway crossings is driven by a precise knowledge of the location of each train in the system. The intention is to minimise the delays to rail and road traffic. Rather than closing the gates for as long as it takes for the slowest train to pass the junction, they can be closed as soon as the EGNOS application reports that the train has moved on. In other words, the provision of accurate satellite based positioning systems enables the gradual erosion of previous safety margins by increased performance and reduced breaking.

Briefly explain how you would use safety integrity levels to guide the development of safety-critical applications such as the EGNOS Controlled Rail Application.

[5 marks]

[Seen/unseen problem]

The key point of many existing approaches to the use of risk assessment in support of safety-critical software engineering is not to associate probabilisties with the likelihood of bugs in the code. This is because software does not wear out in the same way that the probability of hardware failures increased over time, in line with the bath-tub distributions that characterize most components [2 marks]. Another way of expressing this is to say that coding errors are deterministic – if they are present then they should be removed. The second key issue is that software itself will not kill or injure anyone as it is an abstraction for low level signals in the underlying hardware. In contrast, it is the 'equipment under control' that is the focus of risk assessment [1 mark]. Failure modes and error conditions are associated with the train and the signaling equipment, including the EGNOS infrastructure mentioned above. Where the likelihood and consequence of the hazards that together lead to a conjoint risk are considered to be 'unacceptable' then software is one of several mitigation techniques that can be used to reduce the risk. The integrity or assurance level of the software provides a crude measure of the extent of the risk reduction that is to be achieved by that software, again with respect to the equipment under control [2 marks]. The integrity or assurance level acts as a metric to determine appropriate development practices – hence at the higher levels of assurance it may be necessary to use formal methods and trusted compilers whereas this would not be appropriate or cost effective for lower criticality software. The assignment of an integrity level or the use of particular development techniques does not guarantee the correctness of the eventual code [1 mark up tom a total of 5].

b) In the past, there was relatively little on-board information processing by satellites. Instead they acted as a form of router – forwarding data to the ground based processors for further analysis. This is changing and owners are becoming increasingly interested in the use of redundancy to harden satellites to processor failures. Briefly identify at least three different ways in which redundancy might be used to address the problem of on-board processor failures.

[5 marks]

[Unseen problem]

A range of novel architectures are being developed for satellite architectures. One is to develop two parallel pipelines – these are sometimes referred to as the 'A side' and the 'B side' of a satellite. Some of these proposals have suggested that one side could be reserved for military use rather than as a redundant processing channel [2 marks]. However, for systems such as Galileo this approach would provide the resilience needed to support SOL applications. A more obvious technique is to deploy redundant satellites [1 mark]. This reduces the complexity of dual pipeline architectures but has important implications for launch costs given the additional mass and the recognition that redundant satellites may not buy significant additional functionality until a failure occurs. The satellite market seems to be favoring multiple micro satellites rather than a smaller number of larger units [2 marks]. This has implications in terms of software complexity – focusing the vehicles on a smaller number of specialist communications functions that simplifies the operation and maintenance of the satellite [an extra mark for mentioning the implications for software development, including software redundancy and diversity].

c) Global Navigation Satellite Systems, such as Galileo, have three major components:

a. A ground control system responsible for maintaining the operation of the satellite;
b. Communications ground stations that send and receive data to the satellite.
c. The satellite itself that is controlled by the ground control system and which provides data to, and receives data from, the communications ground control system.

A company is hoping to develop and market a variant of the EGNOS Controlled Rail System. Even though you might have reservations about this approach, you have been asked to write a report explaining how one might use estimates of failure probabilities to derive numeric risk assessment for the failure of software modules in these three components.

[10 marks]

[Seen/Unseen problem]

This question will stretch the more able students because they should recognize a possible conflict with part a) The earlier section explains why risk assessments and software assurance levels are associated with equipment under control not with the probability of failure in the software itself [2 marks for mentioning this]. In this question, we ask about the numeric assessment of failure probabilities for modules. So one answer would be to say – don't even try to assess the probability of failure in a particular module [2 marks for making this argument].

If people do want to argue in favor of this approach then they might mention the Musa formula that includes terms for the numbers of lines of code, the execution rate of the code, the expansion ration for the language used (for example increasing the probability of failure from compiler errors in higher level languages) and so on [1 mark for mentioning Musa, 1 marks for mentioning each of the parameters to the formula and for assessing the credibility of that parameter]. I would expect answers to show some skepticism about the application of these formulae and also to consider issues of safety culture and software process management – in other words the training and competence of the programmers may have more of an impact than any of the other terms in the Musa formula [2 marks for mentioning competency and 2 marks for mention the managerial/safety culture issues].

It is important to stress that we may not know in detail what the failure modes or associated probabilities would be for these software components given that we are end-users of the Galileo/EGNOS service. Hence, we may be forced to rely on system level guarantees from the infrastructure providers – in other words, in addition to the theoretical objections to the probabilistic approach, there will be practical and contractual barriers to obtaining the information that would be necessary to inform the application of techniques such as the Musa formula [3 marks for making an argument along these lines about information disclosure].

2.

a) A recent report published by a group of 7 nuclear regulators, including the UK Health and Safety Executive, summarized the problems of software development for nuclear reactors in the following way:

> "It is widely accepted that the assessment of software cannot be limited to verification and testing of the end product, i.e. the computer code. Other factors such as the quality of the processes and methods for specifying, designing and coding have an important impact on the implementation. Existing standards provide limited guidance on the regulatory and safety assessment of these factors. An undesirable consequence of this situation is that the licensing approaches taken by nuclear safety authorities and by technical support organizations are determined independently and with only limited informal technical coordination and information exchanges. It is notable that several software implementations of nuclear safety systems have been marred by costly delays caused by difficulties in coordinating the development and the qualification process".

Briefly explain why regulators find it so difficult to develop appropriate means of certifying safety-critical software for use within the nuclear industry.

[5 marks]

[Unseen problem] There is a lack of consensus over the best methods and techniques to use in the regulation (as opposed to the development) of safety-critical software. As the citation suggests, regulators are often forced to rely on ad hoc and informal communications mechanisms to determine whether organizations are following appropriate development and maintenance techniques [1 mark]. Existing standards that describe development processes and techniques do not typically describe how regulators should audit the application of these standards [1 mark]. Where these issues are considered it is principally from the standpoint of an independent safety auditor rather than the ultimate regularity authority that may paradoxically have more limited resources or technical background than an ISA [1 mark]. This raises the more general issue about the competence and resourcing of regulatory authorities in complex and high-demand areas such as the development of safety-critical software [2 marks]. Many organizations including the US NRC and the9 European regulators are urgently reviewing these issues as part of the drive towards the 'nuclear renaissance' [1 mark].

b) The same report recommended that:

"Before a system is taken into operation, the licensee should submit written proposals to the Regulatory Authority on the methods to be employed (change control, configuration management, maintenance, data entry etc.) to ensure that the required level of integrity of the safety system will be maintained throughout its operational life".

Explain the importance of traceability and configuration management in the life-cycle of safety-critical software.

[5 marks]

[Seen/unseen problem] Software is often the most flexible element of a complex safety-critical system [1 mark]. The ease with which code can be edited or reconfigured stands in contrast to the many problems of installing and updating new hardware in hazardous environments [1 mark]. However, this flexibility creates considerable safety concerns. In the past, it has been hard for safety related organization to identify the code that is actually running on any particular processor (cf MARS SOHO mission) [1 mark]. This in turn complicates the diagnosis and correction of any future bugs [1 mark]. A related issue is that without detailed configuration management it can be difficult to ensure that integrity constraints that are identified in the earlier stages of development will also hold after subsequent modifications [2 marks].

c) The 7 European nuclear regulatory agencies identified a number of benefits that can be derived from the use of mathematics and formal methods in the specification and analysis of complex, software systems. However, they also enumerated a number of concerns:

> "…when used inappropriately, formal methods may be dangerous as:
>
> - their lack of legibility may lead to difficulties in understanding and verification, especially for plant specific application software which must be understood by different branches of engineering,
> - no method is universal, and the impossibility of expressing some types of requirements important to safety (e. g. non functional requirements or aspects of real time behavior) may lead to incompleteness or inconsistencies,
> - they might be used by insufficiently trained personnel or without support of adequate tools. In addition, the training effort required by the use of a formal method can be disproportionate to the benefits that can be expected".

Write a brief technical report explaining to the managers of a reactor operating company how these limitations might be addressed to support the use of formal techniques in software development.

[10 marks]

[Seen/unseen problem] This is an open-ended question intended to ensure that people have seen the connections between different areas of this course. Ideally, they will take each of the points in the enumeration and consider them in turn [1 mark for this]. There are no 'perfect' answers to each of the problems but some level of familiarity with the application of formal methods should be shown. For instance, the legibility issues could be addressed by using graphical formalisms such as Statecharts or Petri Nets or semi-formal techniques [2 marks]. One of the key strengths of the Z approach was the insistence on an informal commentary [1 mark] – other solutions might mention the ability to derive partial simulations from specifications [1 mark].

The second point could be addressed using different techniques for particular areas of their system that are identified using software assurance levels mentioned in the answer to question 1 – for instance a real time core could be analyzed using methods that are appropriate for these non-functional constraints[1 mark]. However, the additional costs associated with many of these approaches need not be justified for code that lies outside the real time kernel [1 mark].

It is hard to ensure that personnel meet sufficient standards for the application of formal methods because, unlike other engineering disciplines, there is a lack of specific competency standards for safety-critical software development [2 marks]. Audit and review by external agencies can increase assurance – there are also commercial organizations such as Praxis that offer specific services in this area [1 mark]. The issues of cost effectiveness can be addressed by appropriate tool support and by the targeting of resources using risk assessments to identify software assurance levels [1 mark].

3.

a) Human Factors distinguishes between "skill-based" performance, "rule-based" performance, and "knowledge-based" performance. Skills are routine behaviors that we perform almost without thinking. Rule based performance involve more conscious thought as we seek to apply procedures that we have been taught. Knowledge based performance often requires wider forms of inference and general knowledge when we cannot find an applicable procedure.

Briefly explain why these three concepts are important for the design of safety-critical systems with reference to one of the accidents or incidents that you have studied during this course.

[6 marks]

[Unseen problem] The skills, knowledge, rules, hierarchy is controversial but can be used to provide a crude approximation of the stages by which a novice might learn to interact with a safety-critical system [2 marks]. Hence, as people grow more accustomed to using safety-related software and associated processes the nature of the potential errors may change [1 mark]. However, there are cases where expert users may be forced to rely on general knowledge when faced by an error in which they have no expertise and which is not covered by the SOPs [2 marks] – an example would be the Kegworth crash when the crew tried to find a task card to restart their one functioning engine in flight – the card did not exist and hence they had no available rules for how to complete this task [up to 3 marks for the relevance of the example].

b) Slips and lapses are errors of skill based behaviour. In contrast, mistakes are associated with rule or knowledge based performance. Distinguish between these different forms of human error and explain what measures you might take to reduce the frequency of each type of error in the design of the safety-critical system mentioned in your answer to part a).

[6 marks]

[Unseen problem] A lapse is an omission from a skilled task – you forget to do something. For example, you forget to lower the landing gear during an approach [1 mark]. A slip is where, for example, you insert an additional action into a skilled task. For instance, the pilot might attempt to lower the landing gear during normal flight [1 mark]. A mistake is where you have an error of intention.- for example, you might try to land the aircraft with the undercarriage raised [1 mark]. Slips and lapses are often best addressed by introducing forms of checks – for instance by have a co-pilot monitor the pilot's actions [2 marks]. As these are skilled activities it can be particularly difficult for the individual concerned to notice a potential problem [1 mark]. Training and the use of simulators as well as procedural cues including task cards and SOPs can be used to address the more obvious mistakes [1 mark].

c) Performance shaping factors (PSFs) can have a major impact on the frequency of human error in complex, safety-critical systems. Write a brief report describing how PSFs might affect the occurrence of slips, lapses and mistakes. You should then explain how testing might be conducted to assess the impact of PSFs on the operation of safety-critical user interfaces.

[8 marks]

[Unseen problem] Performance shaping factors include fatigue, heat, alcohol and drug abuse, noise, stress etc [2 marks]. They describe external conditions that can influence human behavior [1 mark]. They make skill based mistakes more likely to occur because they can interrupt practiced behaviors [1 mark]. Similarly, they can distract crew members so that they forget previously learned SOPs that would otherwise guide rule based behaviors [1 mark]. PSFs can also be argued to consume finite perceptual and cognitive resources – for example, individuals

must concentrate to filter out distracting conversations that might otherwise have profound consequences for knowledge based behaviors [2 marks]. These PSFs create particular problems for the testing of safety-critical interactive systems – for example, it may be unethical to allow someone who has consumed alcohol to use a particular system [1 mark]. These ethical issues can be avoided by the use of simulators [1 mark]. However, in such circumstances behaviors can change because individuals know that they are using a simulator away from their usual working context [1 mark]. Other factors such as noise or heat may expose the operators to dangerous conditions [1 mark]. It can be difficult to know how to simulate some stressors; such as divorce or other forms of external social pressure that have been shown to degrade operator performance [1 mark].

4. The Haddon-Cave review into the loss of Nimrod MR2 Aircraft XV230 advocated the development of 'risk cases' that are to be maintained throughout the life of a safety critical system. The QC argued that 'a paradigm shift is required away from the current verbose, voluminous and unwieldy collections of text, documents and GSN diagrams to Risk Cases which comprise succinct, focused and meaningful hazard analysis which stimulate thought and action'.

   Write a brief technical report that describes how the Haddon-Cave recommendations might affect the development of complex safety-critical systems that integrate software components.

   [20 marks]

[Unseen problem/bookwork/Essay]
The Haddon-cave review is one of the most important reports in the area of safety critical systems development to have been published over the last decade. It describes how faults were introduced when the Airborne Refueling system of the Nimrod aircraft was installed as an urgent requisition item during the Falklands War and during subsequent permanent modifications [2 marks]. The companies involved had produced a safety case but this did not consider previous incidents that provided warnings about potential problems and there is a perception that the safety cases had become an end in themselves – safety was achieved through the maintenance of the safety case rather than the argument reflecting the underlying safety of the application [3 marks].

Haddon-Cave goes on to argue that safety cases are not adequately maintained through the life of a complex system and that risk cases should provide a more light weight 'living document' that is altered and updated through design modifications and in the light of incident or accident reports [2 marks]. It should be easier for operational staff to review and understand safety arguments [1 mark].

The Haddon-cave review does not explicitly deal with software systems but as noted in previous sample solutions, software is often the focus for modification and maintenance activities because of the low perceived cost of updates [2 marks]. Hence many of the arguments mentioned in the Nimrod review are directly applicable to software development [1 mark]. Issues of traceability and of configuration management, mentioned at the start of this paper can again be mentioned here [2 marks]. Stronger students may also refer to the previous questions raising issues about the readability and modification of formal methods following changes in the design or requirements of software systems [2 marks].

I would expect the very best solutions to point out some of the challenges in the Haddon-Cave recommendations – how can we do this and yet maintain the cost-effect development of complex safety-critical systems [2 marks]. We need methods and techniques to do what the QC proposes – he explicitly mentions GSN but this is the present 'state of the art' so how will we push his recommendations forward in practice? [2 marks] In the course, I have mentioned how this is a 'living field of research' and we have very few answers to many of the problems of safety-critical development [1 mark].