UNIVERSITY OF LONDON
IMPERIAL COLLEGE OF SCIENCE, TECHNOLOGY AND MEDICINE

EXAMINATIONS 2004

BEng Honours Degree in Computing Part III
BEng Honours Degree in Information Systems Engineering Part III
MEng Honours Degree in Information Systems Engineering Part III
BSc Honours Degree in Mathematics and Computer Science Part III
MSci Honours Degree in Mathematics and Computer Science Part III
for Internal Students of the Imperial College of Science, Technology and Medicine

This paper is also taken for the relevant examinations for the Associateship of the City and Guilds of London Institute
This paper is also taken for the relevant examinations for the Associateship of the Royal College of Science

PAPER C302=I3.8

SOFTWARE ENGINEERING - METHODS

Tuesday 4 May 2004, 10:00
Duration: 120 minutes

Answer THREE questions

Paper contains 4 questions
Calculators required

Section A

1 **(Alloy as a constraint-based declarative language)**

a Alloy uses `fun`-statements and `assert`-statements for analysis. Given the `fun`-statement

```
fun ReplaceComp(G,G' : GAC, c1,c2:  Comp){
 c1 in G.components && not c2 in G.components
 G'.components = G.components - c1 + c2
} run ReplaceComp for 3
```

Translate `ReplaceComp` into an `assert`-statement `A` such that the analysis of the `fun`-statement `ReplaceComp` and the analysis of the `assert`-statement `A` always render the same result. Justify that correspondence. Why would, or wouldn't, this translation be of use for invocations of `ReplaceComp`?

b Multiplicity constraints, e.g. "*at most one* failure occurs," are important declarative mechanisms but they may give rise to inconsistencies. Write a short Alloy module that uses three *different* explicit multiplicity keywords such that their interaction causes an inconsistency. Identify the location(s) of that inconsistency and suggest a way of resolving that inconsistency.

c i) Consider the two `fun`-statement patterns

```
fun P(s,s':State) { C1 => C2 } run P for 3
fun Q(s,s':State) { C1 && C2 } run Q for 3
```

where `Ci` are constraints with `s` and `s'` as only free variables. Explain in what senses these patterns differ and give a prominent example use for each such pattern, justifying the use of the pattern for each example.

ii) A directed graph is *2-colourable* if, and only if, each of its nodes can be colored with either "blue" or "red" such that no nodes connected by an edge have the same colour. Given the declarations

```
sig Element {}
sig Graph {
 nodes : set Element,
 edges : nodes -> nodes }
```

write a consistent constraint in Alloy whose analysis can only generate a 2-colourable directed graph with exactly five nodes.

The three parts carry, respectively, 25%, 25%, and 50% of the marks.

2 **(Root contention protocol for IEEE 1394 ("FireWire"))**

The IEEE 1394 high-performance serial bus is "hot-plugable," allowing quick and dependable transfer of multi-media data.

a Given the Alloy declarations (*nodes* are any components connected to the bus)

```
sig Node { to, from :  set Link }
sig Link { target, source :  Node, reverse :  Link }
```

Add constraints that ensure "Each node has links connecting to other nodes. Each link has at most one connection."

b Messages (sent along outgoing links) are stored in queues of length at most one with a possible overflow where `Msg` is a partition of scalars `Req` and `Ack`:

```
    sig Queue { slot:  option Msg, overflow:  option Msg }
```

Write a parameterized constraint named `SameQueue` specifying that two queues are identical.

c Given that `Op` is a partition of scalars `Init, AssignParent, ReadReqOrAck, Elect, WriteReqOrAck, ResolveContention, Stutter` consider the state declaration

```
sig State {
part waiting, active, contending, elected:  set Node,
parentLinks :  set Link,
queue :  Link ->!  Queue,
op :  Op } -- the operation that produced the state
```

i) Write a parameterized constraint `SameState` stating that its two argument states are the same.

ii) In a transition function `fun Trans(s,s' :  State) { ...  }` write constraint(s) for the case that the transition is caused by stuttering, where "stuttering" means that the state is idle.

d Considering that `s.elected` denotes the nodes elected as roots of the tree in state `s`, write a `fun`-statement `Evolution()` for linearizing the state space and use it to declare, for all protocol runs, `assert`-statements `AtMostOneElected` and `OneEventuallyElected`.

The four parts carry, respectively, 25%, 15%, 25%, and 35% of the marks.

Section B

3 **(Software Ethics)**

a i) State Kant's categorical imperative.

ii) State Kant's practical imperative.

b Consider and carefully read the following "Big Brother" scenario:

"Malvin works as a computer systems operator for a local police force in Warwick. Janice, a patrolling officer, calls into the station from her patrol car. She has just observed a driver that looks suspicious. Although the driver was not breaking the law, he was driving close to the speed limit and seemed to be continuously looking over his shoulder. Though it was hard to tell from a distance, he appeared to be unkempt and unshaven. Janice took down his registration number and is asking Malvin to check the driver out on the system. Malvin uses the registration number to retrieve a file on a certain Andrew Forster, age twenty-five, who was recently fired from his job at a local bank. He is married, with two children under the age of five, and lives in a middle-class, predominantly Jamaican, neighborhood. Although he is in good standing with the local gas, electricity, and telephone companies, he is two payments behind on his mortgage. He and his wife have had marriage counselling in recent years and he was cleared in a lie detector test, carried out after the occurrence of an internal theft in his bank past year. Malvin passes all this information on to Janice who decides to follow the driver for a while longer."

Was Janice's decision to follow the driver for a while longer ethical according to

i) Kant's categorical imperative?

ii) Kant's practical imperative?

iii) Mill's principle of utility (greatest happiness for all involved)?

iv) You?

Please ensure that you justify each answer compactly but adequately.

The two parts carry, respectively, 20%, and 80% of the marks.

4 **(Software Testing)**

a Software testing, if done methodologically, requires a good fault model.

i) Define the term “fault model.”

ii) Give an example of a particular fault model.

iii) Sketch how the particular fault model of the previous item ii) may impact or determine the activities that occur in software testing.

b Conducting a test on a piece of software may be easy to do, but it may be quite difficult to determine whether such a test reveals an error which deserves to be reported or analyzed. In this context:

i) Explain the active role of *test oracles* and define that concept.

ii) Give an example of a test which has a reliable and easily obtainable test oracle.

iii) Give an example of a test which is likely to have neither a reliable nor an easily obtainable test oracle.

c The vast portion of in-house versions of commercial software is merely code for testing the commercial product. Creating and using such “test stubs” is expensive and challenging. In this context:

i) Describe four *different* challenges of software testing and illustrate them with small code fragments, if appropriate.

ii) Explain why software testing alone is considered insufficient for the *certification of critical systems*, e.g. a smart card platform or the small operating system of a Mars rover, and sketch alternative solutions.

iii) *Programming by contract* refers to the idea of stating assumptions (preconditions) and guarantees (postconditions) in the interface of services so that services can validate their own interface specifications based on those of services they call.

Explain one principal advantage and one principal disadvantage of such an approach to design and implementation. Then discuss how programming by contract may help software testing and vice versa.

The three parts carry, respectively, 30%, 35%, and 35% of the marks.