UNIVERSITY OF LONDON
IMPERIAL COLLEGE OF SCIENCE, TECHNOLOGY AND MEDICINE

EXAMINATIONS 2002

MSc in Computing for Industry
MEng Honours Degree in Information Systems Engineering Part IV
MSci Honours Degree in Mathematics and Computer Science Part IV
MEng Honours Degrees in Computing Part IV
MSc in Advanced Computing
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 C475=I4.16

SOFTWARE ENGINEERING - ENVIRONMENTS

Thursday 9 May 2002, 14:30
Duration: 120 minutes

Answer THREE questions

Paper contains 4 questions
Calculators not required

Section A *(Use a separate answer book for this Section)*

1a In most programming languages conventionally, the binary compatible changes depend on the language implementation. C++ programs may risk a *fragile base class* problem.

i) What is the fragile base class problem? Describe the C++ implementation mechanism that causes it.

ii) Give an example (concrete syntax is not important) that demonstrates that a change in your code would cause a wrong result.

iii) Give an example (syntax not important) that demonstrates that a change in your code would cause the program to crash.

b The Java programming language specification states that correct Java implementations must exhibit the fragile base class problem.

i) Give four changes that are binary compatible changes according to the specification. For each of your changes give the justification for why the modification is binary compatible.

ii) Unfortunately, it is possible to make a binary compatible change, which is not visible to the running program. Give an example that shows this.

iii) Unfortunately, it is possible to make a binary compatible change, which will link to the running program, but the running program cannot be built by recompiling a single set of sources. Give an example that shows this.

The two parts carry, respectively, 40% and 60% of the marks.

2a What are the general characteristics of peer-to-peer (p2p) systems? Why might the name *collaborative networking* be more appropriate than the name peer-to-peer (p2p)?

b What are the technical advances that have led to the rise of p2p?

c Describe briefly two p2p applications, including their overall architecture.

d Querying a decentralized network requires finding a path through a network graph. This is just a distributed search problem (same problem as IP routing). Different p2p systems use different search algorithms. Assume that you have an application where you wish to preserve anonymity and that needs to be scaleable. Describe an algorithm that you might use. (You can draw a diagram or write pseudocode.)

The four parts carry, respectively, 15%, 15%, 30%, 40% of the marks.

Section B *(Use a separate answer book for this Section)*

3a Write an Alloy model for the DOC Spring Exams. Specify the entire state of the model, including students, staff, helpers; and oral and written exams. Each student takes at least one exam. Each exam is invigilated by at least one person. Each exam has a first and second marker. There are cheats as well.

b Specify the basic assumptions of this model, expressed in at least five invariants, e.g. "Invigilators are staff."" These invariants need to contain plain English comments that state what is being formalized.

c Make use of appropriate preconditions and postconditions to specify, in Alloy,

i) the assignment of a first and second marker to an exam;

ii) the assignment of a set of exams to an invigilator;

iii) for all exams, that some assigned marker invigilates; and

iv) that cheats never, ever mark.

d i) Express the specification in Alloy:

"If a helper marks an exam, then he or she invigilates that exam and the other marker is a member of staff."

Describe a snapshot of the Alloy model's state space (of 3(c)) that violates this specification. (In the event that this specification is an invariant of your model in 3(d), prove that fact mathematically.)

ii) Suppose that you add to your model of 3(c) the invariant

"All helpers invigilate some exam."

For this modified Alloy model, either provide a counter-example to the specification of 3di, or sketch a proof that the modified Alloy model satisfies that assertion, or show that the resulting model is inconsistent.

The four parts carry, respectively, 20%, 20%, 20%, and 40% of the marks.

4a Explain what knowledge you may infer from the (edited) replies

i) "No instance exists within given scopes."

ii) "No instance found within given scopes."

iii) "Instance found within given scopes."

when analyzing an **op** schema in Alloy.

b Explain what knowledge you may infer from the (edited) replies

i) "No instance exists within given scopes."

ii) "No instance found within given scopes."

iii) "Instance found within given scopes."

when analyzing an **assert** schema in Alloy.

c Consider the Alloy model below for naming in the worldwide web. Complete the `empty bodies indicated with *** ... ***`

```
// Model of naming in the WWW

// Accounts for relative URLs, but not for caching or for
// implicit extension with default relative URLs (eg,
// index.html). URLs are structured as (domain name,
// resource name), each of which (but not both of which)
// may be missing. Resource names have some minimal
// structuring to account for basenames: for a resource
// name r, r.prefix corresponds to the name obtained by
// trimming of the last element of the URL, and is the
// same as the basename. For example,
// "sdg.lcs.mit.edu/~dnj/alloy.html" has domain name
// "sdg.lcs.mit.edu", resource name "~dnj/alloy.html",
// basename "sdg.lcs.mit.edu/~dnj".

// Each domain name is assumed to translate to at most one
// site. Directory structure within sites is not modelled.
// The translation performed by a web server is modelled
// by the indexed relation selects. At a given site s,
// r.selects[s] is the document obtained by looking up
// resource name r. Interpretation of a relative URL in
// the context of an absolute URL is modelled by the
// indexed relation interprets: for a given basename b,
// b.interpret [u] is the URL that results from
// interpreting the relative URL u (by concatenation).
```

```
model WWW {

domain
{Browser,Document,URL,DomainName,ResourceName,Site,Copy}

state {
  partition Relative, Absolute: URL    // two kinds of URLs
  mentions: Document -> URL
  // u in d.mentions if u is mentioned in d
  dom: URL -> DomainName?
  // u.dom is the domain name part of URL u
  resource: URL -> ResourceName?
  // u.resource is the rest of URL u
  site: DomainName -> Site?
  // DNS: maps domain name to at most one site

  selects [Site]: *** ... ***
  // Web server: maps resource name to document

  basename: *** ... ***
  // maps absolute URL to its basename (maybe itself)

  interpret [URL]: *** ... ***
  // b.interpret [u] is interp of rel u under basename b

  identifies: *** ... ***
  // a.identifies is document resulting from URL a

  loaded: *** ... ***
  // each browser has some set of copies loaded

  doc: *** ... ***        // a copy is of a document

  fetchedas: *** ... ***
  // a copy was fetched with a particular absolute URL

  fetches *** ... ***
  // c.fetches[u] is doc fetched from link u in copy c

  prefix, last: *** ... ***
  // r.prefix is r with the last element chopped off
  }

inv BasisURLStructure {
  *** ... *** // an absolute URL must have a domain name
  *** ... *** // a relative URL has no domain name
  *** ... *** // all URLs have a domain name or a resource
  }

// Basic properties of URL structure
inv URLStructure {
  all a: Absolute | a.basename.dom = a.dom &&
     a.basename.resource = a.resource.prefix
  all r: Relative, base, result: Absolute |
    result = base.interpret [r] ->
      (result.dom = base.dom
      && result.resource.prefix = base.resource
      && result.resource.last = r.resource)
  }

// Definition of identifies: to find a document, use DNS
// (the dom relation) to find the site, then use the web
// server (the selects relation) to get the document.
def identifies {
  *** ... ***
  }
// Definition of fetches: to fetch a document from a
// relative link, interpret the link in terms of the
// basename of the URL with which the containing document
// was fetched.
inv def_fetches {
  *** ... ***
  }

// A copy of a document was fetched with a URL that
// identifies it
inv ConsistentFetch {
  *** ... ***
  }


// An absolute name refers to at most one document
// Valid because DNS and web server are functional
assert A1 {
  *** ... ***
  }

// At most one document results from interpreting a URL
// Valid because DNS and web server are functional
assert A2 {
  *** ... ***
  }

// Translation of relative URLs is consistent:
// A relative URL in two copies of the same document
// results in fetching the same document even if the
// copies are in different browsers and were fetched with
// different absolute URLs. Invalid, because nothing
// requires the web server to return a document for a URL,
// so in one case a document may be returned, but not in
// the other.
assert A3 {
  *** ... ***
  }

// Like A3 but only considers the case in which two
// documents are actually fetched
assert A4 {
  *** ... ***
  }
}
```

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