Print Name:__________________________

Student ID#: __________________________

SOLUTIONS

CPSC 410 — Advanced Software Engineering
Mid-term Examination (Fall 1998)
Instructor: Gail Murphy

Do NOT start until you are informed you can start!

This examination has 7 questions. The question and answer space is from page 2 to page 8. Check that you have a complete paper when told that you can start the examination.

You have **80 minutes** to complete this exam. Before you start, you must write your name and student id number on the top of **every** page of this exam. We will not mark your exam without this information. You must also sign your name in the space provided below.

Write legibly. We also have to be able to read it to mark it.

This exam is closed book and closed neighbour. Notes, book, or other materials are not allowed. Candidates guilty of any of the following, or similar, dishonest practices shall be immediately dismissed from the examination and shall be liable to disciplinary action:

- Making use of any books, papers or memoranda, calculators or computer, audio or visual cassette players, or other memory aid devices, other than those authorized by the examiners.
- Speaking or communicating with other candidates.
- Purposely exposing written papers to the view of other candidates. The plea of accident or forgetfulness shall not be received.

If you have a question during the exam, raise your hand and I will come and answer your question.

The exam is out of 100 marks. The value of each question is indicated. Use your time wisely. If you do not know the answer to a question, move on to the next question and come back to the question at the end.

The exam is double-sided. There are questions on **both** sides of the page.

Good luck!

Signature:___________________________________

Print Name:__________________________

Student ID#: __________________________

1. ***(Total: 20 points. 2 points each.) True or false with justification. For each statement below, circle T if the statement is true and F if the statement is false. In the space provided for each statement, write at most two sentences explaining your choice.***

   (a) The term information hiding is used solely to describe the hiding of data representations behind an interface.

   T or [✓ F.] Justification: Information hiding pertains to hiding a secret behind an interface. The secret may be algorithm or cultural in nature; it need not be restricted to a secret about data representation.

   (b) The uses relation between modules is different than the calls relation between modules.

   [✓ T] or F. Justification: One module may use (i.e., rely on the correct behaviour of another module without calling it. This can be true of interrupt handlers.

   (c) The batch sequential architectural style and the pipes-and-filters architecture style share many similarities.

   [✓ T] or F. Justification: They are both sub-styles of a data-flow architectural style

   (d) The layered architectural style is used when the ease of porting a system is a desired property.

   [✓ T] or F. Justification: A layered system may typically be ported by changing only the lowest layers, since each layer may, in essence, define a virtual machine

   (e) COM/OLE interfaces have unique identifiers that are separate from the identifiers for COM/OLE components.

   [✓ T] or F. Justification: Interfaces are uniquely identified by IIDs; components are uniquely identified by CLSIDs. The same interface may be implemented by more than one component.

   (f) A function call may be a connector in the blackboard style.

   [✓ T] or F. Justification: Components in the style may be functions or modules. The main distinguishing feature of the system is when control and data are passed, not how.

Print Name:__________________________

Student ID#: __________________________

(g) Messages (method calls) are the primary means of supporting communication between Java Beans.

T or [✓ F.] Justification: Events are the primary recommended form of communication because they increase the possibility of reusing beans.

(h) An algebraic specification is oriented around the state of the system being specified.

T or [✓ F.] Justification: Model-oriented specifications are based on state. Algebraic specifications focus on the relationships between sets.

(i) A formal approach to specification ensures that a produced specification is complete.

T or [✓ F.] Justification: A formal specification may be incomplete as we saw in the lecture on algebraic specifications.

(j) Statecharts are an example of a descriptive approach to formal specification.

T or [✓ F.] Justification: Statecharts are an operational approach to formal specification.

2. *(Total: 10 points) Describe two ways in which software architecture can benefit the software development life-cycle.*

A variety of answers were possible here. Marks were based on the identification of the benefit as well as how it aids development. Here are three possible answers (5 marks for each of two).
1. Software architecture improves the communication amongst stakeholders to promote a common understanding. This may ease acceptance testing activities and/or simplify design and implementation.
2. Software architecture enables the early analysis of system properties, possibly preventing costly errors by ensuring the appropriate system is being built.
3. Software architecture supports a transferable abstraction of a system possibly enabling the identification of product lines and increasing the possibility of reuse.

Print Name:__________________________

Student ID#: __________________________

3. ***(Total: 10 points) Characterize the object-oriented architectural style according to the following features: kinds of components, kinds of connectors, control topology, the binding time of control, and the purported development benefits of the style.***

Components: Objects (or Classes) [Either answer is acceptable. 2 marks]

Connectors: Primarily method calls [2 marks]

Control Topology: Arbitrary [2 marks]

Control Binding Time: write-time, compile-time, and run-time. [1 mark if one of these three, 1.5 if two, 2 marks if all three]

Benefits: Modifiability, scalability, reusability [1 mark each to max of 2 marks]

4. ***(Total: 5 points) What features of Sun's JavaBeans helps ensure a component provides (is compatible with) behaviour that is desired in an application?***

   Java interfaces are used to ensure compatibility. [3 marks] This handles compatibility for non-persistent beans. However, when beans persist, a Java-level interface may be changed in an incompatible way. In these cases, the developer is responsible for ensuring compatibility, typically via a stream unique identifier. [2 marks]

   Introspection could also consider to be involved [1 mark if total of 5 not already reached].

   [

Print Name:__________________________

Student ID#: __________________________

5. ***(Total: 15 points) Compare and contrast the following two integration approaches: Microsoft's COM/OLE and OMG's CORBA.***

[Here is an abstracted version of an answer to make the marking clearer. Other answers were possible and were evaluated on a case-by-case basis.]

Both define standards to achieve component inter-operability. Both enable inter-application (inter-process) [2 marks] inter-operability. CORBA is inherently distributed [1 mark]; DCOM must be used to integrate distributed applications in COM/OLE case.

Offering Services:
Both use interfaces [1 mark] to describe available services. Both use IDLs (Interface Definition Languages) [1 mark] to define interfaces, although the IDLs differ. Both [1 mark] support the definition of interfaces separate from the language used to define the component. (This means you can control a component from a programming language that is different from the language used to create the component.)

Find Component:
In CORBA, ORB dynamically finds an object meeting the desired service commitments [1.5 marks]. In OLE, the COM layer interacts with the registry to find a component [1.5 marks].

Compatability:
CORBA ensures a component is compatible with the desired behaviour through interfaces and the ORB. [1 mark] In contrast, in OLE, a component interacts with interfaces (through QueryInterface) to determine if the component is compatible [2 marks].

Interact:
In both cases, component interaction occurs through method calls. But in CORBA, the calls are through the ORB. The calls are direct in the COM/OLE case [3 marks]

6. (Total: 25 points) Write a Z specification for the system described by the following natural language description. You need only specify the correct operation of the system (i.e., you can ignore error handling). The last page of this examination booklet provides some Z syntax you may find helpful (but not necessarily all you may wish to use)..

The Student Registration System shall track the students who have registered for a particular course offered by the University. There shall be a limit placed on the number of students who can register for any particular course. This limit will be defined by a member of the registrar's office. A student shall only be allowed to register in a particular course if the limit for that course has not yet been reached. The system shall also permit a student to withdraw from a course, cancelling that student's registration in the course. Each course shall have only one section.

A "real" Z specification would include natural language text describing the Z. Given the time constraints on the exam, I did not expect any of this descriptive text. Given that the question only asked for the correct operation of the system to be specified, we won't do any validity checking on courses and students.

We use a static Z schema to describe the state space of the system. First, we introduce some basic types (STUDENT and COURSE) that we don't need to describe in detail. The static schema defines an invariant that all courses in which a student may register have a set limit.

[STUDENT, COURSE]

```
┌ RegistrationSystem ─────────────
│ limits: COURSE → N (partial function. Appropriate symbol not available.)
│ registrations: COURSE ↔ STUDENT
├─────────────────────────────────
│ dom registrations ⊆ dom limits
│
└─────────────────────────────────
```

We then need to describe the initial state of the system. Note that the invariant above is satisfied.

```
┌ InitRegistrationSystem ─────────────
│ RegistrationSystem
├─────────────────────────────────────
│ dom limits = ∅
│ dom registrations = ∅
└─────────────────────────────────────
```

There needs to be a way for the registrar to set limits on courses. You didn't have to specify a schema necessarily, as long as you had some means of the limits being defined.

SetLimit
Δ RegistrationSystem
courseId?: COURSE
limit?: N

limits' = limits $\oplus$ { courseId? $\mapsto$ limit? }

Now the main operations of Register and Withdraw.

Register
Δ RegistrationSystem
studentId?: STUDENT
courseId?: COURSE

StudentId? $\notin$ registrations (| courseId? |)
#(registrations (| courseId? |)) < limits(courseId?)
registrations' = registrations $\cup$ { courseId? $\mapsto$ studentId? }

Withdraw
Δ RegistrationSystem
studentId?: STUDENT
courseId?: COURSE

studentId $\in$ registrations (| courseId? |)
registrations' = registrations \ { (courseId?, studentId?) }

Here's an outline of how this question was marked…

10 marks were allocated to the description of the system state (something like RegistrationSystem) broken up as 3 marks for the static schema formulation, 3 marks for associating limits with courses, 3 marks for tracking registrations, and 1 mark for the specification of invariants.

2 marks were allocated to the initialization of the system state.

3 marks were allocated to the providing some way of setting limits (it didn't have to be through a separate schema as above).

5 marks were allocated for registering a student in a course.

5 marks were allocated for withdrawing a student from a course.

Print Name:__________________________

Student ID#: __________________________

7. ***(Total: 15 points) Identify three problems, in the style of Meyer's problems with natural language specifications, in the following requirements specification. Your identification should include the text causing the problem as well as the type of problem (i.e., remorse).***

   The system shall record people's birthdays and shall issue a reminder when the day comes round, stating the names of those whose birthday is on the given day. Those persons whose birthday falls on February 29 will be treated specially. An error will be issued if a birthday is specified for a person whose birthday has already been recorded.

   A user shall specify the birthday for a given person by stating the person's name followed by a space followed by the date of the person's birth followed by a newline character. If the person's name has been seen before by the system, than the newly specified birthday will overwrite the old birthday.

   The reminder service provided by the system will report the names of those persons with birthdays on the current day. The user will be responsible for invoking the reminder service once each day.

1. "Those persons whose birthday falls on February 29 will be treated specially." This statement is an example of SILENCE. What should happen on February 29? Should the birthdays be reported on Feburary 28 on a non-leap year? (I'll also accept AMBIGUOUS for those statement).
2. "An error will be issued if a birthday is specified for a person whose birthday has already been recorded." "If the person's name has been seen before by the system, than the newly specified birthday will overwrite the old birthday." This statement is an example of CONTRADICTION. The second statement suggests overwrite, while the first specifies an error if a birthday is entered twice.
3. "space followed by the date of the person's birth followed by a newline character." This statement is an example of OVERSPECIFICATION. Whether or not a newline is needed depends on how lines in the system work.

[Marks were deducted if you stated more than three problems on an increasing scale for the number of answers given.]