Student ID#: _____

CPSC 410 — Advanced Software Engineering
Midterm (Term I 2002-2003) with Solutions
Instructor: Gail Murphy

Do NOT start until you are informed you can start!

This examination has 6 questions (with multiple parts). The question and answer space is from page 2 to page 9. Check that you have a complete paper when you are told that you can start the examination.

You have **80 minutes** to complete this exam. Before you start, you must write your student id number on the top of **every** sheet 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 and in **ink** (not pencil). 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 one of the invigilators 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!

Print Name:_____

Signature:_____

Student ID#: _____

1 point for T/F correct. If T/F correct, then 1 point for justification if it is reasonable.

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

(a) The scenarios used in SAAM are equivalent to the use cases determined during the requirements and analysis activities of the system's development.

T or F. Justification:

SAAM scenarios must account for more kinds of stakeholders; for instance, the developers who make changes to the system and the system administrators are included.

(b) The 4+1 view model for expressing a system's software architecture exemplifies the separation of concerns design principle.

T or F. Justification:

Each view provides a different, and separate, perspective on the system. For example, the deployment view shows how processes will be mapped to processors, whereas the logical view describes the functionality of the system.

(c) An association between two classes in an object-oriented design can be directly implemented in an object-oriented language such as C++ or Java.

T or F. Justification:

Object references, possibly in collections, must be used to implement associations.

(d) A description of the software architecture of a system can help a software developer reason about several kinds of non-functional requirements for the system.

T or F. Justification:

Architectural views can help reason about performance, availability, and reliability as well as other non-functional requirements.

(e) All design decisions are architectural.

T or F. Justification:

A software designer must also make choices about how a user interacts with a system (the usability), which is not an architectural decision.

(f) In a system whose architecture exhibits the batch sequential architectural (sub-)style, one component need not complete execution before another component begins execution.

T or F. Justification:

In this style, one component must complete execution before another starts. This style was used in early computer systems with limited memory.

(g) A plug-in for the Eclipse platform can be written in any programming language.

T or F. Justification:

Eclipse plug-ins must be written in Java.

(h) UML distinguishes between compositions and aggregations.

T or F. Justification:

Yes, a composition is an aggregation in which there is strong ownership by the whole of the parts, when the whole is deleted, all of the parts are deleted as well (i.e., the whole and parts have coincident lifetimes).

(i) The Factory Method pattern is useful when a developer knows when an object should be created, but is not able to specify exactly which kind of object should be created.

T or F. Justification:

In an abstract class, the factory method can encode when an object should be created by delegating the construction to another (abstract) method; the abstract method is then overridden in subclasses to create an object of a suitable type.

(j) Constraints placed on the architecture of a system at design time may not be violated in an implementation.

T or F. Justification:

Constraints are most often violated for reasons of performance. For example, layer bridging may be used in a layered architecture to ensure adequate performance.

2. (Total of 10 points) Interface Segregation Principle

(a) (5 points) Briefly describe (in 2-3 sentences maximum) the Interface Segregation Principle.

The idea in this principle is that a class should provide access to its functionality through many client-specific interfaces, rather than exporting only one general interface. Each client-specific interface should provide access to a cohesive set of functionality.

5 marks – Class exports many client-specific interfaces, each with cohesive set of functionality
4 marks – Class exports many client-specific interfaces
3 marks – Class exports many interfaces

(b) (5 points) How does the use of the Interface Segregation Principle improve an object oriented software system?

Consider that the class A exhibits the Interface Segregation Principle, exporting functionality through Interfaces B, C and D. The use of A can improve an object-oriented software system because a region of the system can declare a variable of the type corresponding to the functionality needed from A, perhaps declaring the variable to be of type B instead of A because only a subset of functionality is needed. As a result, the coupling between the region of the program and A can be narrowed to just that functionality declared in B.

5 marks – coupling between a part of a program and class offering service is reduced/narrowed

3. ***(Total of 25 points)*** Architectural Style

(a) (10 points) Compare and contrast the batch sequential and pipe-and-filter architectural (sub-)styles according to the kinds of components, kinds of connectors, control topology, binding time of control, and data topology in each (sub-)style.

The kinds of components differ in each sub-style: for the batch sequential sub-style, the components are (typically large) batch programs; for the pipe-and-filter sub-style, the components are filters (typically a small program that does one thing).

The kinds of connectors differ in each sub-style; for the batch sequential sub-style, the connectors are batch data files; for the pipe-and-filter sub-style, the connectors are data streams.

The control topologies also differ: The batch sequential sub-style control topology is linear where the pipe-and-filter topology is an acyclic graph.

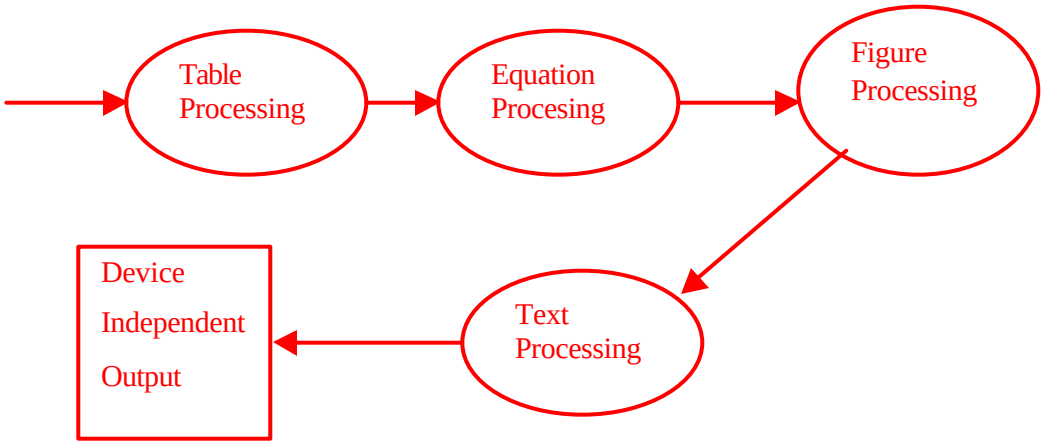
The binding time of control also differs: In the batch-sequential sub-style, the binding time is run-time; in the pipe-and-filter style, the binding time is compile or run-time (for Unix pipe-and-filters especially).

The data topologies are the same as the control topologies.

There are five characteristics on which you were to compare and contrast the two sub-styles. Each characteristic is worth 2 points: 1 point for each sub-style.

(b) E (15 points) You have been contracted to build a text processing system. The system is to support the inclusion of tables, equations, and simple figures. The system produces device-independent output. (It is outside the scope of your system to convert the device-independent output to screens, particular printers, etc.). You have been told that your users will be sophisticated computer users, capable of writing descriptions of the tables, equations, and figures they want if given an appropriate language in which to express the table, equation, or figure.

Design an architecture for the system that conforms to the pipe-and-filter architectural style. You do not need to fully elaborate the architectural design, but you must describe the components, the connectors, and how the system will basically work. State any assumptions you make.



The components in this architectural design are filters (small programs). Each filter reads in a stream of characters, looking for mark-up commands that the filter knows about. When a filter recognizes a region of the stream that is marked-up, it replaces that region with suitable device independent output according to the mark-up. A filter ignores regions of the input stream that contain device independent output. The connectors between the filters are streams of characters. This design assumes that equations are not embedded in tables, and that figures do not contain either tables or equations. (The troff system had this basic architectural style.)

2 marks for each filter (total of 8 marks).

3 marks for connectors as streams of characters.

4 marks for description of how it works.

4. (Total of 20 points) Software Architectural Views

(a) (12 points) Name and briefly describe the role of each of the 4 (not the +1) views in the 4+1 view model. (Briefly describe here means 2-3 sentences for each of the views.)

The logical view describes how the system will support the functional requirements.

The process view shows how the functionality from the logical view is mapped onto threads of control (processes, tasks, and threads).

The development view shows how the software is packaged into chunks that can be developed by more than one developer.

The deployment (or physical) view shows how the software maps onto physical processing nodes.

3 points for each view: 1 for the name and 2 for the role.

(b) (8 points) What component(s) is (are) in each view?

Components in the logical view are typically classes, in the process view are tasks and processes, in the development view are packages and subsystems, and in the deployment view are processing nodes.

2 marks for each view. You get 2 marks for a view if you name at least one component that appears in the view. You get 1 mark if you include more than 1 inappropriate component in a view.

5. ***(Total of 10 points)*** Implicit Invocation. You have been asked to build a system that supports the description and visualization of a graph (with nodes and edges). You have been asked to apply the Event Notification Pattern to support the implicit invocation of the visualization component of the system when the graph description is changed. Sketch a class diagram for the system and explain how you have used the Event Notification Pattern.

A solution will be sketched in class (its too painful to draw in Word!).

1 mark for identifying the 4 classes involved: Graph (Subject), GraphDisplay (Observer), StateChange and EventStub. You did not have to name the classes exactly these names, your design just had to convey that you had classes that played each of these roles.

3 marks for your explanation of how you used the pattern to solve the stated problem. Many people failed to convey how the registration of the stub to the statechange worked, and how the eventstub knew what to invoke on the observer.

1 mark for getting the connections between the classes basically right.

1 mark for having registration capabilities on the StateChange.

1 mark for having notification capabilities on the StateChange.

6. (Total of 15 points) Architectural Style II

(a) (5 points) What differentiates a 3-tier client-server architecture from a 2-tier client-server architecture.

A 3-tier client-server architecture has a middle tier that encompasses the business (logical) functionality of the system.

5 marks for middle tier with business functionality.

3 marks for middle tier.

2 marks for one more tier.

(b) (5 points) What are two advantages of a 3-tier client-server architecture over a 2-tier client-server architecture.

A 3-tier client-server architecture is easier to administer, typically can have better security, provides better encapsulation of data, has better performance, can handle more clients, provides better application reuse, is better at legacy system integration, provides good support of the Internet, supports heterogeneous databases, and has better architectural flexibility.

3 marks for the first of any of the above reasons.

2 marks for the second of any of the above reasons.

(c) (5 points) When would you choose a blackboard architectural (sub-)style over a repository architectural (sub-)style

When you want to be able to easily introduce new clients of the data into a system where the clients need to share results amongst them.

5 marks – easily introduce clients who need to share data (or who need to be informed of data changes)

3 marks – easily introduce clients (2 marks if you say scale and aren't clear about in which way the system needs to scale)