TE WHARE WĀNANGA O TE ŪPOKO O TE IKA A MĀUI

VUW **VICTORIA**
UNIVERSITY OF WELLINGTON

EXAMINATIONS — 2008

MID-YEAR

COMP 301
Software Engineering Principles

Time Allowed: 3 Hours

Instructions: Answer all questions
Total marks are 180.
Use the marks for each question as a guide as to how long to spend on it.
No calculators are permitted.
Paper dictionaries for translating between English and a foreign language are permitted.

continued...

Question 1. General Knowledge [30 marks]

(a) [6 marks]

Describe the relationship between the UML meta-model and the UML models that we create for our own software systems.

(b) [6 marks]

A project manager's team is about to start a software development project. The project manager wants to use an estimation tool to determine how much effort will be required for the software project. The tool requires that the size of the project and a vector of various cost factors are entered so that the tool can determine the amount of effort required. Discuss two problems concerning the tool's input that can seriously affect the validity of the calculated effort.

(c) [6 marks]

What is the difference between functional and non-functional requirements?

(d) [6 marks]

Describe one way that the Unified Process is commonly modified to create specific life-cycle models such as the Rational Unified Process or the Enterprise Unified Process.

(e) [6 marks]

What is the difference between a patent and copyright?

Question 2. Design [45 marks]

(a) [5 marks]

Compare and contrast patterns and anti-patterns.

(b) [5 marks]

Why might we want low coupling between objects in an object-oriented design?

(c) [10 marks]

A developer wants to record some details of an object's state at a particular point in time. These details must be kept for later restoration. However, the object's encapsulation should not be broken.

What is the name, solution and consequences of a pattern that can resolve this problem?

(d) [15 marks]

Consider the following simple problem description:

> A local community sports team wants a software system to manage their player roster and match schedule.
>
> The team captain should be able to add or remove players, and add or remove future (but not past) match details. Any other team member should be able to remove themselves from the system, and view the matches they are currently assigned to play.
>
> Team captains should also be able to assign players to future matches, and the system should help the team captain ensure that all players get a chance to play a match. In particular, if there are M future matches, and P existing players in the system, then a player cannot be assigned to a match if (at the time of assignment) that player has already been assigned to more than $((7M/P) + 1)$ future matches.
>
> Whenever a player is assigned to (or removed from) a match, they should be emailed. It can be assumed that a valid email address has been entered when the player details were entered into the system. Note that when a player is removed from the system, they should automatically be removed from all matches they are assigned to, and the team captain should be alerted that replacement players are required for those matches.

Identify a plausible cross cutting concern in the problem description. Use your cross-cutting concern as an example to discuss one way that an aspect-oriented design can be a better decomposition of the system than an object-oriented design.

(e) [10 marks]

Describe one possible aspect-oriented implementation of the Observer Pattern. Your description does not have to include the actual source code, but should state in general terms what aspects, join-points and advice would be needed.

Question 3. Implementation [20 marks]

(a)

The Java Native Interface (JNI) allows a developer to use C++ (or C) code in their Java software system.

(i) [5 marks]

Discuss one reason a developer might want to implement their Java software system using the JNI.

(ii) [5 marks]

Discuss one disadvantage of using the JNI with respect to a core principle of the Java programming language.

(b) [10 marks]

Discuss two distinct ways that coding standards can help a developer both understand and integrate a reusable Java software component (developed by someone else) into their own Java software system.

Question 4. Testing [25 marks]

(a) [5 marks]

What is an advantage of having separate testers and coders in a development team?

(b) [10 marks]

Compare and contrast the bottom-up and top-down integration testing strategies.

(c) [10 marks]

A development team wants to estimate how many unfound bugs there are remaining in their code. Describe and justify a technique for estimating unfound bugs. You may assume that the coder and the tester are different people.

Question 5. Deployment & Maintenance [17 marks]

(a) [10 marks]

Compare and contrast how stand-alone applications and plugins are updated post-installation.

(b) [7 marks]

Discuss the validity of the following statement: ″Polymorphism in object-oriented code ensures that it is easy to trace execution in the source code″.

Question 6. Software Lifecycles [22 marks]

(a) [7 marks]

Discuss the validity of the following statement: "The cycles in the spiral model will follow the order of activities in the waterfall model."

(b) [15 marks]

Identify and describe three practices used in Extreme Programming. In your answer you should also discuss how the three practices you chose are related to the core values of Extreme Programming.

Question 7. Free & Open Source Software [21 marks]

(a) [7 marks]

Discuss the validity of the following statement: ″the software industry is a manufacturing industry″.

(b) [7 marks]

Discuss the validity of the following statement: ″open source development is an agile methodology″.

(c) [7 marks]

Name two business models that could allow companies to make a profit based on open source principles. Name one real world example for each model.

****