EXAMINATIONS — 2001

END-YEAR

COMP 462

OBJECT-ORIENTED PARADIGMS

Time Allowed: 3 Hours

Instructions:

- *Read each question carefully before attempting it.*
- This examination will be marked out of **120** marks.
- Answer all questions.
- You may answer the questions in any order. Make sure you clearly identify the question you are answering.
- Many of the questions require you to express and justify an opinion. For such questions, you will be assessed on your *justification*.
- Some of the questions ask for examples from object-oriented languages. Your answers need only refer to object-oriented languages discussed in the course, but you may refer to other languages if you wish.
- Non-electronic foreign language-english dictionaries are permitted.

Question 1. Use Cases [20 marks]

(a). Define the term "use case", and discuss whether use cases are object-oriented.

(b). Outline what Constantine argues are the advantages of "essential" use cases over conventional use cases, and discuss whether there are any disadvantages.

Question 2. Design [20 marks]

(a). Explain what is meant by a "responsibility" in Responsibility-Driven Design, and outline the case for how it leads to a good design.

(b). Compare using *heuristics* with using *metrics* as ways of evaluating an object-oriented design. Use at least one example of a heuristic from Riel, and at least one example of a metric from Chidamber and Kemerer's paper.

Question 3. Frameworks [20 marks]

(a). Explain the differences between a class library and an object-oriented framework.

(b). Describe Hans Albrecht Schmid's approach to the development of object-oriented frameworks. Suggest a role for one of the The Gang of Four (Gamma, Helm, Johnson & Vlissides) patterns that supports this approach.

Question 4. Patterns [20 marks]

Alexander's patterns are organised into a *Pattern Language*. The Gang of Four (Gamma, Helm, Johnson & Vlissides) *Design Patterns* do **not** form a pattern language.

(a). Explain why the *Design Patterns* do not form a pattern language. Describe what would be required to turn them into a pattern language.

(b). Imagining the *Design Patterns* could be turned into a pattern language, do you think this would be a good thing? Why or why not?

Question 5. Language [20 marks]

The programming language Yøbø 1.0 is like Java, but does not have either *subclassing* (i.e. `extends`), or *subtyping* (i.e. `implements`).

The following list of language features has been suggested for inclusion in Yøbø 2.0. Present brief arguments why programmers may want these features. In your arguments, indicate the relative importance of each of the features.

(a). Single Inheritance — like Java `extends`

(b). Multiple Inheritance — like C++ or Eiffel

(c). Subtyping — like Java `implements`

(d). Generic (Parametric) Classes — like C++ templates or Eiffel's Generic classes

(e). Reflexive Metaclasses — like Smalltalk

Question 6. Object-Oriented Paradigms [20 marks]

Doing *Extreme Programming* using a *prototype-based reflexive language* will result in a *Big Ball of Mud*.

Discuss.

****