Victoria University of Wellington

DEGREE EXAMINATIONS — 1995

(MID-YEAR)

COMP 462

Object-Oriented Paradigms

Time Allowed:

THREE HOURS

Instructions:

- *Read each question carefully before attempting it.*
- This examination will be marked out of **150** marks, so allocate approximately 1 minute per mark. The marks per question are specified in square brackets "[]". The marks for parts of questions are specified in parentheses "()".
- You must answer each of the "top-level" questions however you have some choice within question 5. Do not answer more than is asked — *the extra answers will be ignored.* Cross out any solutions you do not wish to have considered.
- 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.*

1. [30 marks]

You have just joined a software development organization that does not use object technology in any form. The CEO has become very enthusiastic and wants the company to convert to object technology for all phases of the software development process. The Director of the division that you work for has been charged with the conversion process. Being a practical person, she doesn't buy all the hype that the CEO has been fed. She has asked you to prepare a report that describes, in clear terms, what object technology is, how it affects the different parts of software development, and what the costs and benefits of object technology really are. Due to the time constraints (about 30 minutes) she has placed on you, she does not expect a detailed discussion — just the important issues. She also does not expect perfect prose but does expect something readable!

2. [20 marks]

An important concept in most object-oriented programming languages is *encapsulation*.

(a) Explain what encapsulation is and why is it important to the object paradigm. (5 marks)

(b) Discuss the support for encapsulation provided by CLOS. Your discussion should also cover why the CLOS designers chose the form of encapsulation they did. (7 marks)

(c) In Smalltalk, no object may directly access the instance variables of another object, even if both objects belong to the same class. In C++, objects may access instance variables (data members) of other objects if they belong to the same class. Discuss the tradeoffs between these two views of encapsulation. (8 marks)

3. [20 marks]

Booch regards *hierarchy* as an important feature of the object model. The two forms of hierarchy he discusses are *aggregation* (also known as *composition*) and *inheritance*.

(a) Explain what the two hierarchies are and why they are useful. (10 marks)

(b) There are two views as to what inheritance means. Compare these two views. Your discussion of each view should include an example of a language whose design was influenced by that view. (10 marks)

4. [20 marks]

In object-oriented programming, *polymorphism* includes the notion of dynamic binding, that is, the decision as to which code will be executed is made a run-time. In most languages, this kind of polymorphism is fundamental to the language. However in C++ and Ada9X, the programmer must explicitly distinguish between static versus dynamic binding of operations.

(a) Discuss the pros and cons of having to explicitly request polymorphism, compared with always having it available. Your discussion should cover both ease of use and ease of implementation. (7 marks)

(b) Compare and contrast the support for polymorphism in C++ and Ada9X. (8 marks)

(c) Dynamic binding in C++ and Ada9X only applies to classes that are related (ancestors and descendents) in a statically-specified inheritance hierarchy. Emerald's equivalent hierarchy (based on types) is not static. This allows for more flexible relationships between objects, and hence more opportunity for dynamic binding. However it can also result in inappropriate behaviour. Give an example where the inappropriate behaviour can happen. (5 marks)

5. [20 marks]

Answer any **two** of the following questions.

(a) Some languages that claim to be object-oriented do not have inheritance, but instead have *delegation*. Explain what delegation is and compare it to inheritance. (10 marks)

(b) Craig Chambers has created Cecil, a language designed around *multi-methods*. Explain what multi-methods are, the problem this feature is supposed to solve, and how Chambers justifies the inclusion of this feature in the object model. (10 marks)

(c) Smalltalk-80 has support for concurrency in the form of being able to send a `fork` message to a block, as well as semaphore objects. Discuss the problems associated with this form of support for concurrency and describe one variant of Smalltalk that tries to overcome these problems. (10 marks)

(d) It has been suggested that an object-oriented database management system (OODBMS) is just an object-oriented language that supports persistence. In fact this is not the case. Discuss the forms of support beyond that of the basic object model that are typically provided by OODBMSs. You should also discuss how the object model is typically modified for OODBMSs.

6. [40 marks]

Consider building software for managing an Automatic Teller Machine (ATM). ATMs provide the following services for bank accounts: *Balance*, *Deposit*, *Withdrawal*, and *Transfer* (which is just a withdrawal followed by a deposit). The kinds of bank accounts that are available are: *cheque*, *savings*, *super savings*, and *term deposit*. All kinds of accounts except cheque accounts earn interest. Only the balance service is available for term deposit accounts from ATMs (other services may be available at a branch office). Savings accounts charge a penalty fee for each withdrawal, and super savings accounts do not allow withdrawals at all (including transfer). Deposits that are not part of a transfer operation are subject to a clearance period before accounts will be credited. (Note: This behaviour by ATMs may not be representative of that by commonly available ATMs in New Zealand.)

(a) The above description does not specify what the ATM software is to do. Two possibilities are: the software is intended to be a complete reimplementation of the existing software, or the software is intended to simulate existing services so that new services may be tested.
Discuss the difference in the software design required for these two applications. Your discussion should be in terms of the different classes or different responsibilities of classes. (10 marks)

(b) For each of the classes below, write a paragraph justifying whether or not having that class would be useful. If the existence of the class depends on the purpose to which the ATM software is to be put, describe the kind of application that may require it. (20 marks)

i. CASH
ii. INTEREST
iii. CARDREADER
iv. CUSTOMER
v. BANK

(c) The description above lists 4 different kinds of accounts. Discuss the appropriateness of representing these as an inheritance hierarchy. Your discussion should include a description of the benefits you expect to gain from such a hierarchy (should you decide it is a good idea) or an explanation of the problems associated with a hierarchy (should you decide it is a bad idea). (10 marks)

****