The University of Melbourne
Department of Computer Science and Software Engineering
433-254 Software Design
Semester 2, 2003
Solutions - Tutorial 2
Week 3

1. Explain the object-oriented paradigm and how does it differ from the structured paradigm of software development?

   Sample Answer:

   The object-oriented paradigm is based on the idea that *data* is central to a program's existence. In such a paradigm, the world is not viewed as a set of functions alone; the data that the functions act upon is equally important. Data and the functions that operate on the data are coupled together to form objects. The blueprints for these objects are known as classes which form the basis for the creation of an object. For example, you may view different cars as separate objects but they are all instances of a class known as car, which defines the essential attributes and services that a car provides.

   In the structured paradigm is important is given to data or the procedures but not both.

2. What is meant by a *software crisis*? Many argue the need for a new programming paradigm, why is this? Explain how the object-oriented paradigm attempts to overcome this *software crisis*.

   Sample Answer:

   The software crisis hit long ago and has not been resolved. It has been observed that by far the majority of software projects go over time, over budget and at the end of the day, are of a low quality.

   It is worth noting that this results from not just the complexity of the problem, but also the difficulty in understanding the requirements of the system to be developed. In the long term, some are predicting (or hoping) that much higher level, declarative languages (functional/logic) will become the solution to the crisis; however these languages are still mostly limited to academic use. There advantage lies in the fact that they abstract away from the underlying nuts and bolts, allowing the programmer to focus purely on the problem to be solved.

   OO attempts to beat the crisis by supporting modularity through classes. Modularity allows for isolated development of modules in parallel. OO design attempts to maintain a direct mapping from the modeled problem domain, to the software. It supports code re-use through polymorphism and inheritance.

3. Explain the terms object and class for OO programming (use examples)

   Sample Answer:

Class is the implementation of an Abstract Data Type. An abstract data type is definition which contains attributes and a set of methods which operate on the attributes.

Examples: Vehicle, VehicleOwner

Object is a specific instance of a class.

Examples: Vehicle with registration number xxx yyy defined as vehicleA is an object of class Vehicle.
John who is the owner of vehicleA is an object of class VehicleOwner.

4. Define the following OO related terms:
   (a) Encapsulation
   (b) Data abstraction
   (c) Inheritance
   (d) Multiple Inheritance
   (e) Polymorphism

   Sample Answer:
   Refer your notes.

5. What is software reuse? What is the difference between reuse and porting? What are the factors influencing software re-use?

   Sample Answer:
   Software reuse typically refers to the idea of not duplicating code in a software system. This is important for many reasons beyond simply reducing the size of source files. If changes are needed in code that has been duplicated many times, then that change will need to be made for every duplicated instance of that code fragment.

   This differs from porting software which is exporting code from a particular system, and if need be, modifying it to either run on a different platform or in a different system. Portability of code is a completely separate Software Engineering aim (one of the major features of programs written in Java).

   One of the factors that influence the level of software use in your program is the choice of classes. Classes should attempt to define as much of what is common to all the objects being classified in order to avoid the duplication of these things in each sub class. The choice of objects also influences the sorts of categorizations that are made, which in turn influences code re-use.

6. Identify reusable components in software and discuss how OOPs help in managing them.

   Sample Answer:
   You should be able to come up with some examples of code re-use from your own experiences. Examples of re-usable components include software things like graphics or math libraries where classes offer services to a system. In Java, a

huge array of classes is available which allows the re-use of code. OOP's give us the functionality to support modularity in our code through classes and inheritance which allows for more re-usable code.

7. What are the pros and cons of web-based applications compare to stand-alone applications?

   Sample Answer:

   Pros: availability, platform independence, easy to use (GUI).
   Cons: Security, performance.