Sample answers for tutorial #1 (for week 4) (Jan 30, 2004)

17th February 2004

The following are just examples, and wide variations are expected...

Q1: Find an example of a GUI blooper, or hall-of-shame entry from any NUS web site. Briefly describe both *it*, and how it could be improved. (Pictures/screenshots and brief description).

Answer: *Well - when I am asked to change a student's tutorial selection, I am not sure if I should use the CORS system or the SOC system. But in either case, the applications are cryptic and painful. For example - even finding the SOC application involves each of the following - each leading to its own different screen:*

```
Login to MySOC
  MySOCAcademic
    OnLineReg - undergraduate (why?)
      SOC Tutorial/request/swapping
        Click here for Tutorial swapping
          Lecturer-tutor login
            Submit
              Login again (sometimes more than once for some reason)
                Enter
                  Join/request accept reject
```

Q2: RESEARCH: Hick's law is an empirical law which can be used to make a time estimate for how quickly a person can operate a GUI. It can be applied to individual tasks such as choosing a menu item. Research Hick's law, and discuss how it can be applied to the design of faster menu systems. Does the law suggest you should use fewer-longer menus, or more-shorter menus?

Answer: *Hick's law is an empirical law which relates the number of targets/items in (say) a menu system, and the time T taken to choose one. The relation is logarithmic:*

$$T = k \log_2(n + 1)$$

where n is the number of targets, and k is some constant. The law suggests that you can use fewer-longer menu items.

Q3: A claim was made in class that there is some *link* between OO technology and GUI programming. As justification for this *link*, the notes mention the use of libraries and reuse of code. However this sort of use of libraries is *generally* useful - it is just as useful for *any* software development, not just GUI programming. Can you identify any aspect of OO technology that is *particularly* useful in GUI programming?

Answer: *In the GUI programming model, events are processed by the top-most widget, and if not, by the parent enclosing widget, and so on. The inheritance/class hierarchy is used in GUI programming: an inner widget is always a specialization of an outer widget. If you do not override an event-processing method when defining an inner widget, the next widget out will process it. So, in some sense, the inheritance model of OO directly supports the event model most often used in GUI programming.*

Q4: Briefly outline a suitable system architecture for each of the following GUI applications:

(a) A personal time management system for use by one person on a single PC.

Answer: *A standalone architecture might be sufficient. The simpler architecture minimizes the software, and hopefully as a result the number of bugs.*

(b) A personal time management system for use by one person on either their work, home, portable PCs, or on a PDA, or remotely.

Answer: *A web server based application might be sufficient. Many modern PDAs support web browsers.*

(c) A ticket/sales application for a small private bus operator BUS-U-THERE-ALIVE, which will be used by the three people who work at the counter, and can be used to book trips only on a BUS-U-THERE-ALIVE bus.

Answer: *A shared database application might be sufficient. Each counter sales-person uses a client program which uses a shared database.*

(d) A ticket/sales application for a small private tour operator, which will be used by the three people who work at the counter, and can be used to book trips using any airline, bus operator etc.

Answer: *The shared database application could be extended. Each counter sales-person uses a client program which uses any of a number of shared databases, perhaps with a proxy.*

(e) A ticket/sales application for a major international airline, which will be used by anyone with a credit card.

Answer: *A web server based application might be sufficient.*

Q5: If you were arranging for a GUI application to have a record-and-replay option, what things would have to be recorded? If on the other hand, you wanted to develop some tool that would allow record and replay for ANY application, in what way(s) would it have to be different?

Answer: *You may need to catch all keypresses, and mouse-clicks that are directed to the application. The recorder will have to also record what widget captured each event. In the event that the GUI application uses mouse-movement directly (for drag and drop and so on), then the mouse-movement will have to be recorded. This record-and-replay software could be part of the application, capturing all events for recording before they are passed onto the correct widget.*

When the tool is to work for all applications, instead of just one, then you have to capture events before they get to the application.