ENGINEERING TRIPOS PART IIA

Thursday 11 May 2.30 to 4

Module 3F6

SOFTWARE ENGINEERING AND DESIGN

*Answer not more than **three** questions.*

All questions carry the same number of marks.

*The **approximate** number of marks allocated to each part of a question is indicated in the right margin.*

There are no attachments.

STATIONERY REQUIREMENTS	SPECIAL REQUIREMENTS
Single-sided script paper	None

You may not start to read the questions printed on the subsequent pages of this question paper until instructed that you may do so by the Invigilator

1 The POSIX PThreads standard provides four core functions to support multi-threaded programming:

```
void pthread_mutex_lock(pthread_mutex_t *mutex);
void pthread_mutex_unlock(pthread_mutex_t *mutex);
void pthread_cond_wait(pthread_cond_t *cond, pthread_mutex_t *mutex);
void pthread_cond_signal(pthread_cond_t *cond);
```

The mutex type (`pthread_mutex_t`) is a binary variable used to control access to a critical section. A thread calling `pthread_mutex_lock(mutex)` will block until `mutex` is in the unlocked state, at which point `mutex` will be locked and the thread will proceed into the critical section. On leaving a critical section, `pthread_mutex_unlock(mutex)` is called to unlock `mutex` and allow other threads access.

Condition variables are used to signal events to threads waiting inside critical sections. When `pthread_cond_wait(cond,mutex)` is called, `mutex` is unlocked and the calling thread is blocked, when it is unblocked it automatically calls `pthread_mutex_lock(mutex)`. When a thread calls `pthread_cond_signal(cond)`, one (and only one) of the threads waiting for `cond` is unblocked. The calling thread is unaffected.

(a) Explain what is meant by multi-threading and describe the principal hazards that it introduces. [15%]

(b) Define the term 'critical section' and explain why some form of operating system support (such as `pthread_mutex_lock`) is needed to allow safe access to critical section. [15%]

(cont.

(c) In a multi-threaded C++ program, a bounded buffer class is required to allow messages to be sent asynchronously between one thread and another. The class has two principal methods:

```
void Put(Message m);  // put message m into buffer, block if full
Message Get();        // get next message from buffer, block if empty
```

Assuming that messages are stored in a circular array indexed 0..N-1, provide implementations of Put and Get using the POSIX primitives given above. [Hint: you will need one mutex to protect access to the array and two condition variables to deal with the buffer full and empty states.] [40%]

(d) What problems arise if a thread needs to collect messages from several input buffers? Describe a solution which does not require polling. [30%]

2 (a) What is a virtual function? Give an example of how one could be used. [20%]

(b) A local ambulance authority is commissioning a software system to manage its fleet of ambulances and the answering of emergency calls. Part of the specification for the software reads as follows:

Specification:

The software running in the ambulance control room should allow the user to view a list of emergencies. The user should be able to select an emergency and view details of the emergency, in particular the address to which an ambulance should be sent and the nature of the emergency. The software should display the current locations of all ambulances in the fleet on a computerised map of the local region as well as a list of ambulances by identification number. The user should be able to select an ambulance from this list, or by clicking on its location on the map. When an ambulance is selected, the user should be able to see details of that ambulance including the details of any emergency it may be attending and the names of the crew, together with any special skills or equipment they may have. The user should be able to place a radio call to the ambulance from this display. The user should be able to assign an emergency call to an ambulance. When an ambulance is responding to a call the colour with which it is represented on the map display should change, as should its appearance on the list (e.g. through the use of a red background behind the identification number).

There should be a separate system on-board each ambulance which can interface to an existing gps navigation system. This on-board system should also interface with the software in the control room. In particular, it must relay its location (as measured from the gps) to the control room software and should accept emergency assignments from that software. When an assignment is received, it should assign the location of the emergency as the destination in the gps navigation system which provides turn-by-turn instructions to the driver.

Using good design principles, draw a UML class diagram which shows the main classes that will be needed for this software, the relationships between these classes and the main attributes and operations that they should support. Identify any classes which have to expose an external interface to a network and show the CORBA `idl` that would be needed to describe this. You do not need to give any pseudocode in your answer. [55%]

(cont.

(c) The ambulance authority now wishes to extend the software to allow emergencies to be prioritised and to integrate the system with scheduled activities (such as carrying patients to outpatient clinics).

Show what changes will be required to the software in order to support this additional functionality, drawing any modified parts of your class diagram and identifying any design patterns used. [25%]

3 Figure 1 shows a UML class diagram for software which plays music on a portable player.

(a) Identify the classes present and the relationships between them. You do not need to describe any attributes or operations. Identify any design patterns that are present. [25%]

(b) Draw a sequence diagram showing what happens when the `update()` function is called on the `CurrentlyPlaying` object in the case when the current track has come to an end and there are unplayed tracks in the queue. [25%]

(c) The software must now be modified to allow two additional pieces of functionality:

(i) The user should be able to play internet radio stations. These radio stations offer a network interface using the following CORBA `idl`

```
interface RadioSation {
  void stream_audio();
  string name();
  string current_track();
};
```

The radio stations should appear in the library like normal tracks, but when one is played, the display should show the name of the radio station followed by the name of the track.

(ii) The user should be able to create and play playlists. These should also appear in the library like normal tracks, but when a playlist is playing, the display should show the name of the current playlist before the name, artist and time remaining for the specific track currently playing from the list.

Show what changes will be required to the software in order to support this additional functionality, drawing any modified parts of the class diagram and identifying any design patterns used. [50%]

(cont.

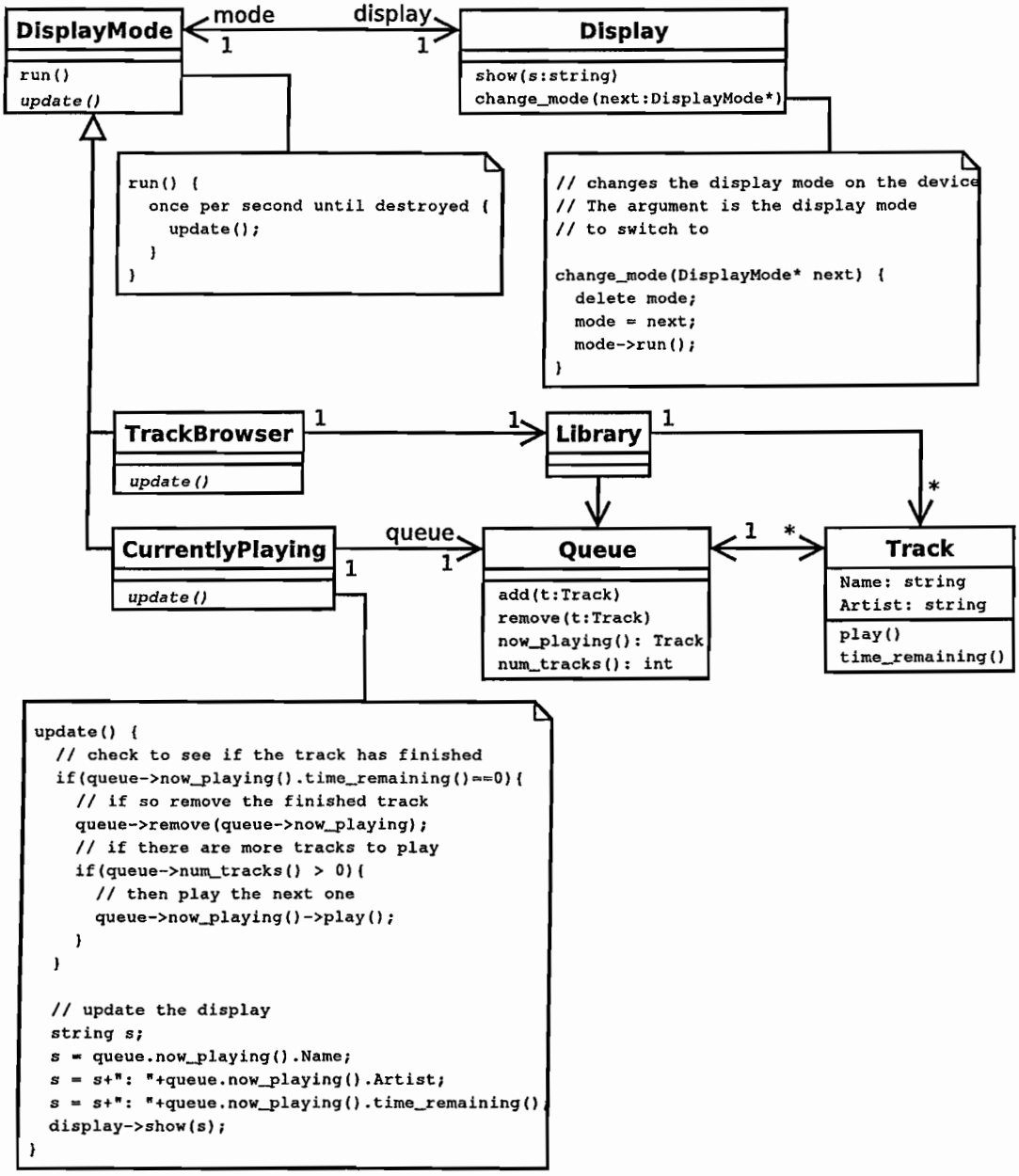

Fig. 1

(TURN OVER

4 An aircraft manufacturer is developing a new system to control and monitor the flaps on a new high-performance wing. The flaps are moved by electrical actuators, and it is vital for safe take-off and landing that their position is always set and known. Intelligent monitoring of the system will reduce maintenance costs and time. A top-down analysis has already identified three major subsystems:

1) The command and control system for the flaps, which interfaces to the existing flight controls and displays in the cockpit.

2) The monitoring and data storage system to record flap movements, actuator loads, etc.

3) A maintenance and condition monitoring system for use by technicians, which will run on a laptop computer connected to the data store during servicing. Trends, counts and out-of-band errors will be displayed and manipulated.

(a) Formal methods and prototyping are two very different approaches to software design. For each, identify one of the subsystems where the approach would be suitable and discuss why. [20%]

(b) (i) Describe three techniques for fault minimisation during development which could be applied to this project.

(ii) Critical parts of the command and control subsystem must be tolerent to remaining faults. Describe how N-version programming provides fault tolerance.

(iii) What steps could be taken during design, implementation and testing to maximise the advantages of N-version programming?

[50%]

(c) Discuss the considerations which govern the design of user interaction and information presentation in relation to the maintenance and condition monitoring interface. How can the software engineers ensure that the interface meets the needs of the technicians who will use it? [30%]

END OF PAPER