Faculty of Arts and Science
University of Toronto

Midterm Test

Department: Computer Science
Instructor: Steve Easterbrook
Date and Time: 10:10am, Thursday Nov 10, 2005

Conditions: Closed Book
Duration: 50 minutes

This test counts for 20% of your final grade

Name: _____
(Please underline last name)

Student Number: _____

Question Marks

1 _____/20

2 _____/20

3 _____/20

4 _____/20

Total _____/80 = _____%

1. [Short Questions; 20 marks total]

(b) [Operational Feasibility – 5 marks] Operational feasibility refers to the likelihood that the proposed system, if built, can actually be used by its intended users to solve the original problem. What types of issues can affect the operational feasibility of a proposed system?

User resistance – e.g. the users decide the system has a negative impact on them personally, and so find ways of avoiding using it

Legal and social factors – e.g. the system would violate legal or social norms, for example by invading people's privacy or data protection laws

Labor objections – e.g. unions protest that the system causes an unacceptable change to working conditions, and threaten to strike in protest

(etc)

(b) [Systems Theory – 5 marks] In systems theory, a system may be described as a soft system or as a hard system. Explain each type of system, and give an example of each.

Soft systems are systems that are difficult to define precisely, because the system depends on the viewpoint of the person describing it. If it is difficult or impossible to come to agreement on the boundaries of the system and its behavior, then the system is considered to be soft. All human activity systems are soft systems. For example, a banking system is a soft system.

Hard systems are well-defined and it is relatively easy to get agreement on where the boundaries to the system are, and what the purpose of the system is. The key difference between soft and hard systems is the amount of consensus that can be reached. The mechanical operation of a car is an example of a hard system.

(c) [Formal Inspections – 5 marks] Name two different approaches for structuring the defect collection meeting during a requirements inspection. What are the advantages and disadvantages of each?

Checklists – the inspectors take each item on the checklist in order, and discuss defects of that type no matter where they occur. This approach ensures the inspection covers all the different types of defect normally found in a document, so that important classes of defect are not neglected. However, it may mean some parts of the document are not covered well; if the checklist omits certain types of defect, these will not be discovered.

Round Robin – the inspectors take it in turn to describe one of the defects they have found. This approach ensures that all inspectors get to contribute equally, so that the particular expertise of each inspector is taken into account. However, it means that the inspectors might not cover the whole document very well, or might miss certain classes of error. If the expertise/experience of one particular inspector is more important, this technique does not all the moderator to give that inspector more time.

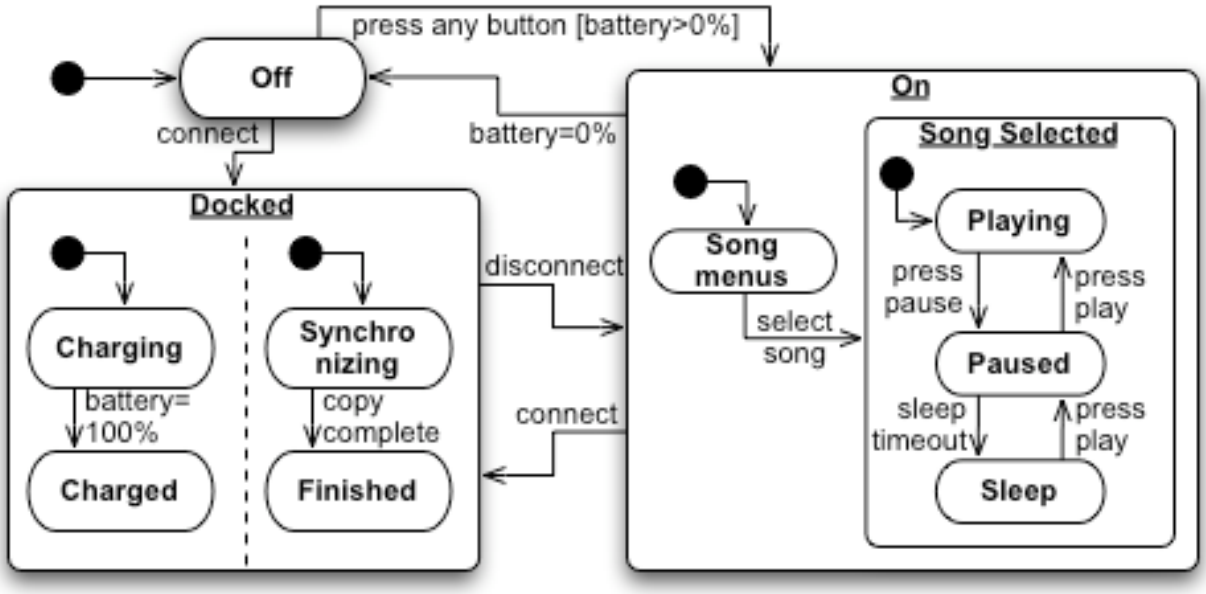
(d) [Requirements vs. Specifications – 5 marks] Michael Jackson has proposed a conception of requirements engineering that distinguishes machine domain phenomena from application domain phenomena, as illustrated in the following diagram:

Explain the distinction Jackson makes between Requirements, R, and Specifications, S. What steps are needed to turn requirements into a specifications?

Requirements are any properties of phenomena in the application domain that a stakeholder would like to be made true by some new system. Requirements may refer to any phenomena, whether accessible to the machine or not. Specifications are a subset of requirements, covering only phenomena that are shared between the application domain and the machine.

A specification must be expressed only in terms of things the machine can accomplish by reference to its inputs and outputs. Hence, for each requirement, it is necessary to identify how the phenomena the stakeholders actually care about can be mapped onto inputs to outputs that the machine can manipulate.

2. **[State Modelling – 20 marks]** The following statechart diagram is a sketch of some of the behaviours of an iPod.



a) Write down a *sequence of events* that will play a song (from the initial state).

press any button; select song (this assumes that initially battery has power)

b) What is the shortest *sequence of events* needed to get back to the playing state if the battery runs out

connect; disconnect; select song

c) What is the shortest *sequence of events* needed to select a different song while playing?

connect; disconnect; select song (note: the model isn't very accurate!)

d) For each of these properties, state whether the property is true or false, and explain your reasoning:
"the iPod remembers what song was playing when it is docked"

False – the only path from the docked state requires you to select a song

"the iPod remains in the sleep state if the battery dies"

False – the "battery=0%" returns the iPod to the Off state from the sleep state.

"sometimes you need to press play more than once to get it to play"

True – from the sleep state, you need to press play twice.

"there is no way to turn the iPod off other than running the battery down"

True – You need to generate a "battery=0%" event to get back to the Off state

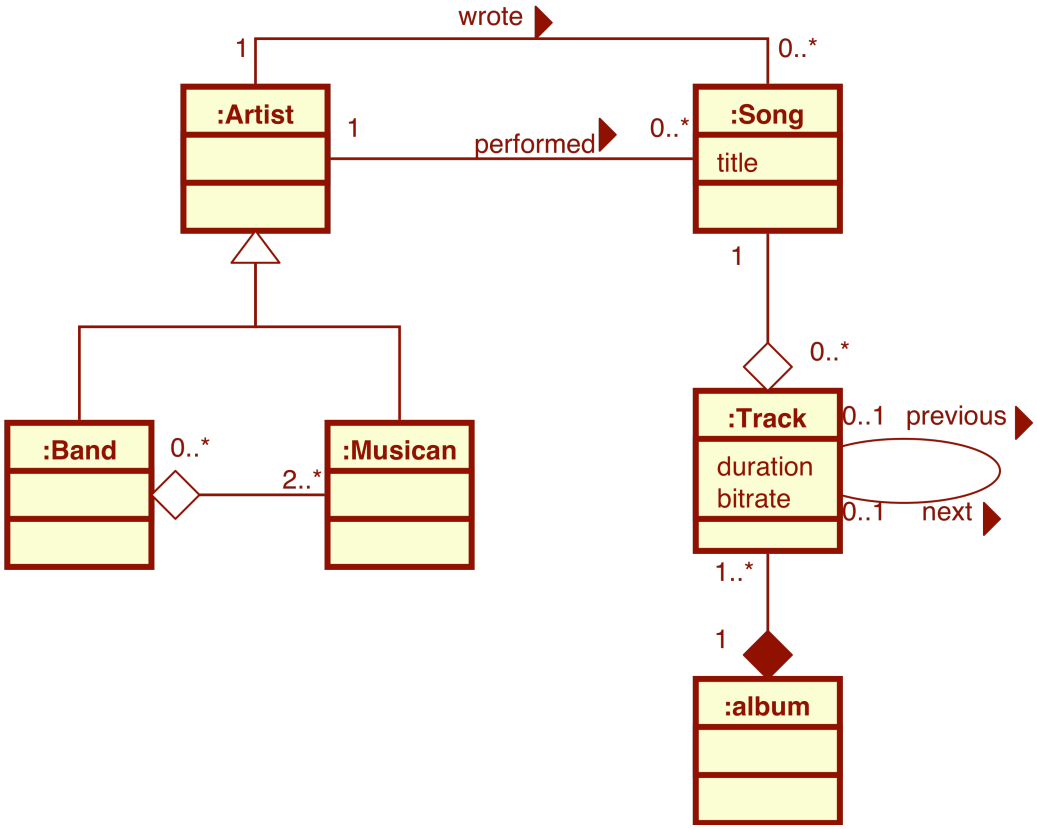
e) What does the model say about what would happen if the battery dies while the iPod is docked?

It ignores this case. If the battery is dead when you disconnect, the model says you go to the menus state, and then to the Off state immediately, implying the menu shows, if briefly. This might be okay if docking always provides a minimal charge.

f) When you first buy an iPod it has no songs on it. How would you modify the model to include this situation?

Add another state distinct from "off" to indicate "empty". The iPod would start in this state, but would probably never return to it except perhaps for a factory reset. There would be no transition from the empty state to the menus state.

3. **[Class Diagrams – 20 marks]** Draw a UML Class Diagram representing the following elements from the problem domain for digital music players: An artist is either a band or a musician, where a band consists of two or more musicians. Each song has an artist who wrote it, and an artist who performed it, and a title. Assume a "song" means a *recording* of a piece of music, so that if a piece of music is recorded more than once (say, by different artists), we treat them as different songs. Therefore, each song is performed by exactly one artist, and written by exactly one artist. An album is composed of a number of tracks, each of which contains exactly one song. A song can be used in any number of tracks, because it could appear on more than one album (or even more than once on the same album!). A track has a bitrate and a duration. Because the order of the tracks on an album is important, the system will need to know, for any given track, what the next track is, and what the previous track is (if there is one). Draw a class diagram for this information, and be sure to label all the associations with appropriate multiplicities.



4. **[Modelling Stakeholders' Goals – 20 marks total]**

(a) [5 marks] Who are the stakeholders for a new system? Give examples of at least three different types of stakeholder. Suggest two things you could do to check that you have found the relevant stakeholders.

A stakeholder is anyone who is affected in any way by the development of a new software system. Examples include: users, customers (who pay for the system), developers, senior management, testers, marketing, etc…

To find the relevant stakeholders, you could examine the organization chart and find those roles in the organization who's job will be affected; you could use a checklist of types of stakeholder, to see if you missed any roles. You could ask any stakeholders you have already identified to see if they can think of anyone else who will be affected by the system.

(b) [5 marks] A major online store manager has told you that her primary goal in developing a new digital music downloading service is to make the service significantly easier to use than their existing service. Give two different *types* of question that could be used to develop a goal tree from this starting point, and suggest an example goal that each type of question might elicit.

"How" questions, to elicit lower level goals,
e.g. "How would you make it easier to use?"
might elicit subgoals such as "fewer steps involved in making a purchase"; "easier to browse recommended music"; "simplify special offers"; etc

Why questions, to elicit higher level goals,
e.g "Why do you need to make it easier to user?"
might elicit higher level goals such as "increase market share"; "attract more customers to other parts of the store"; etc

(c) [5 marks] Explain the difference between a softgoal and a hard goal. Give an example of each, and explain what makes each hard or soft.

A hard goal is one that can be satisfied completely, by some function provided by the new system. For example, the goal "provide one-click shopping" is hard, because if the new system allows customers to purchase an item with one click, the goal is fully met.
A softgoal cannot be satified fully, but it is possible to improve the level of satisfaction. For example, "make the system easier to use" is a softgoal. Various features included in the new system might improve satisfaction of this goal, but it's always possible to improve it further.

(d) [5 marks] Goals may be related to one another in various ways. For example, one goal may be a subgoal of another. Describe three other relationships that may hold between a pair of goals, and give examples.

One goal may *hurt* another, meaning that satisfaction of the first goal reduces the satisfaction of the second. E.g. the goal "make the products cheaper" will hurt the goal "increase profits"

One goal may *make* another, meaning satisfaction of the first goal guarantees satisfaction of the second. E.g. the goal "eliminate server downtime" makes the goal "provide 24/7 web presence"

One goal may precede another, meaning they have to be satified in a particular order. E.g. the goal "collect customers' purchase histories" has to be satisfied before the goal "make recommendations for additional purchases".

[scratch paper]