Faculty of Arts and Science
University of Toronto

Midterm Test

Department:	Computer Science
Instructor:	Steve Easterbrook
Date and Time:	10:10am, Thurs 28th Feb, 2008

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]

(a) [Conway's Law – 5 marks] Conway's law states that the structure of a large software system will reflect the structure of the organization that built it. Why does this happen, and what clues does it give us about how to organize large software development teams?

Conway's law works because software design is a knowledge-intensive activity. Each decision taken by a developer can potentially affect the tasks of any other member of the team. To control the dependencies, and hence minimize the amount of knowledge sharing across subteams, the developers inevitably create modules that contain the parts they are responsible for, and try to simplify the interfaces with other people's modules.

This suggests that the best way to organize the development team is to choose a stable architecture for the software, and divide up the team according to the main architectural units. A good architecture minimizes coupling between units, which then means that the need for communication between subteams is minimized.

(b) [Risk Management – 5 marks] Identify the three biggest risks you face on your course project this term? How do you know these are the biggest risks?

[Many possible choices for risks here – marks awarded for any reasonable list]
e.g.
1. team members dropping the course
2. failure to coordinate the tasks needed to complete the assignments
3. failure to understand what the assignments are asking for

[to answer the second part, some reasonable risk assessment process must be described]
e.g.
We know these are the biggest risks because we conducted a risk assessment exercise. We brainstormed a list of possible risks, using the sample risk lists given in the lectures as a starting point. We then estimated the likelihood of each risk occurring, and the impact if it did. Impact was assessed as the potential for lowering our course grades. We then used these to rank the risks. The above three risks had the highest probability of occurring and all would lead to significant reduction of our course grades.

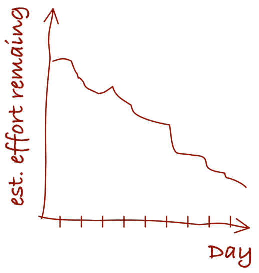
(c) [Software Aging – 5 marks] What are the *causes* and *symptoms* of software aging? What steps can be taken to reduce the problems associated with software aging?

Software ages because the world continually changes – technology improves and users demand more functionality. If the software is not updated continually, it becomes steadily less useful. Symptoms of software aging include growing size, growing complexity, and deteriorating structure, because it gets harder to accommodate changes within the original design.

To reduce the problems associated with software aging, we can attempt to restructure the software by re-factoring it, or we can re-engineer it, through a process of reverse engineering a design model, redesigning these to improve modularity, and then re-implementing the system.

(d) [Project Management – 5 marks] What techniques do agile development methods offer for keeping track of progress on a software project? How do these compare with traditional techniques?

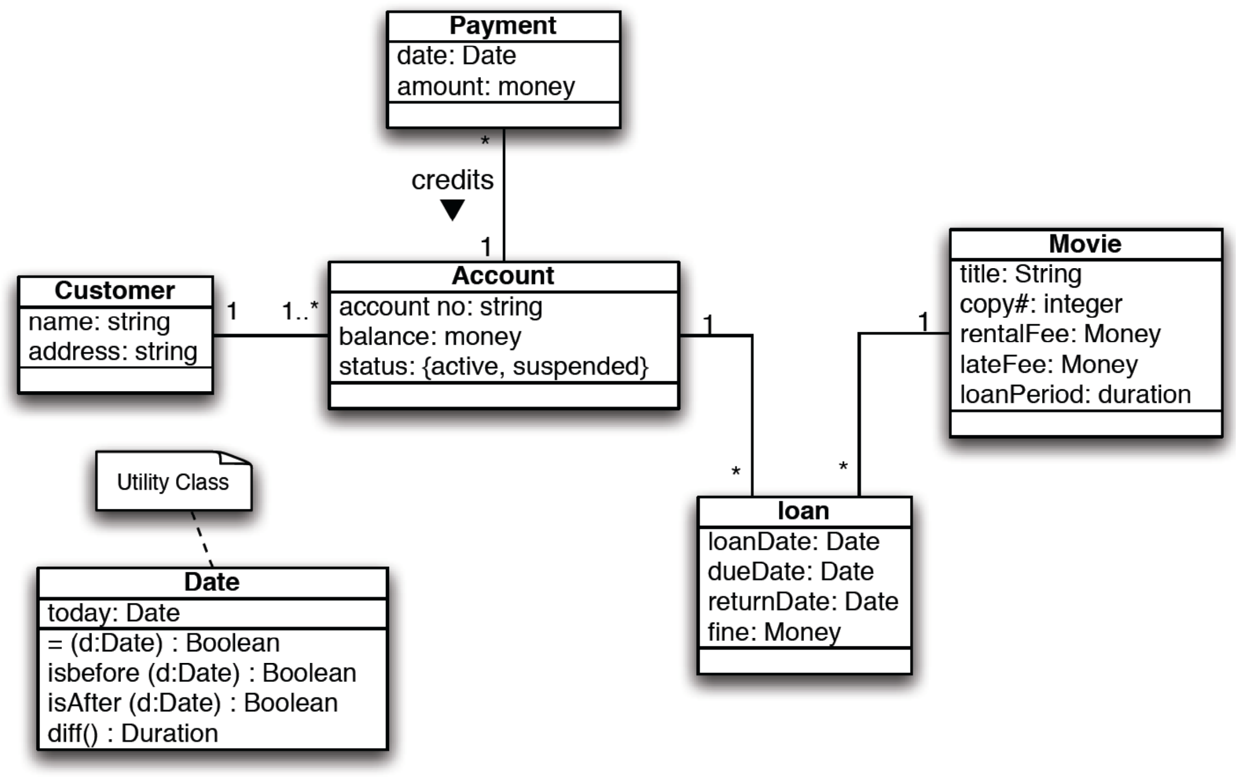
Agile methods emphasize small iterations, and regular delivery of working code. To keep track of progress, they compare completion of coding tasks with the estimates of effort needed for those tasks.



One way of doing this is with a burn down chart, which plots effort remaining (sum of effort estimated for all unfinished tasks) against time. This shows day-by-day how the project is doing, compared to the plans.

Traditional (non-agile) techniques focus on much larger-scale plans, using pert charts or gantt charts. They suffer from the problem that it's hard to tell whether tasks other than coding are really completed.

2. **[Class Diagrams – 20 marks]** The following class diagram shows part of the design of for software for EasyDVD, a web-based DVD rental agency. EasyDVD allows customers to browse its website to see the inventory of DVDs, and select DVDs for rent. DVDs are sent to the customer and returned by mail. Each DVD has a set rental fee and a set loan period.



(a) [5 marks] There are no associations shown for the Date class. How would you redraw the diagram to show such associations? What at the advantages and disadvantages of adding these explicitly?

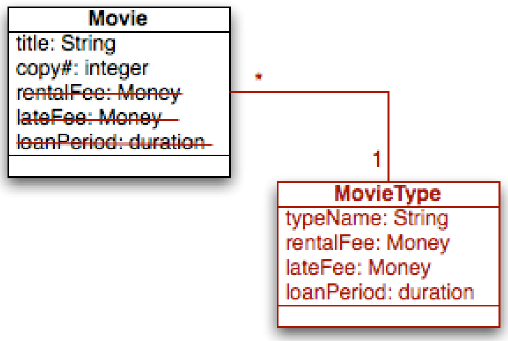
You would need to add three associations from the Loan class to the Date class, labeled loadDate, dueDate, and returnDate, plus another association from the Payment class to the Date class. You would then remove these date attributes from the Loan and Payment class. Adding these to the diagram makes it more obvious how the date class is used, but makes the diagram more cluttered and harder to read.

(b) [5 marks] EasyDVD wants to restrict customers to have no more than 3 DVDs on loan at a time. How would you modify the diagram to show this constraint? Note: EasyDVD still needs to keep track of (a potentially large number of) past loans for each customer's account.

First, you would need to distinguish between current loans and past loans. A simple way to do this is to create two subclasses of the Loan class. The existing 1-to-many association from Account to Loan would then be moved to the "pastLoans" subclass, and a new association from Account to currentLoans, with 0..3 at the loan end.

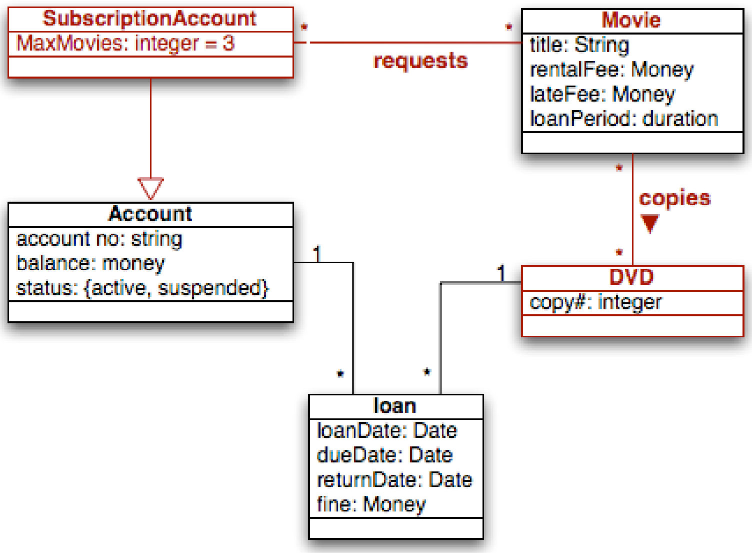
(c) [5 marks] EasyDVD wants to simplify its pricing structure, so that there are a small number of categories of movies, where each category has a set rental fee and loan period. How would you modify the diagram to allow this?

Add an additional class for movie types, which records the rental fee and loan period (and probably the late fee too?) and link this to the Movie class via a 1-to-many association. These attributes would need to be deleted from the Movie class:

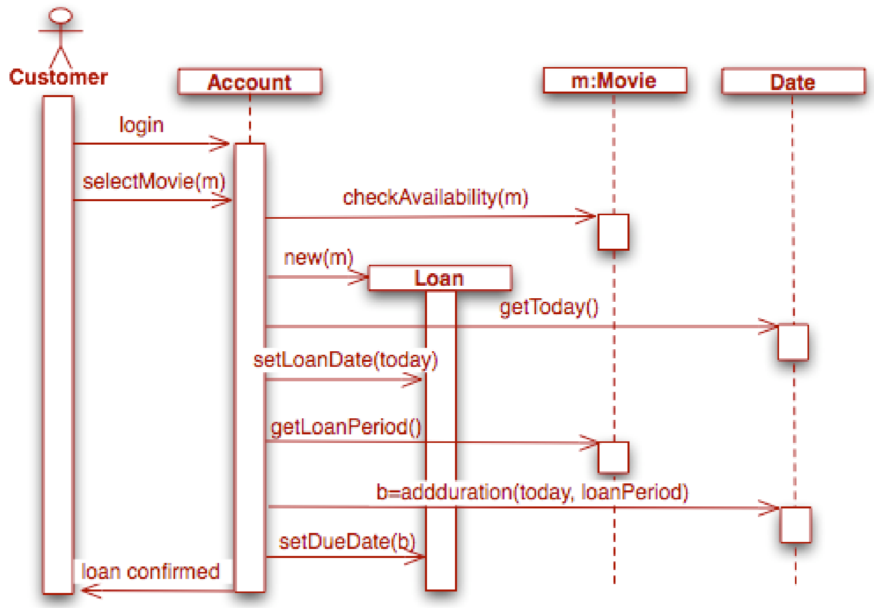


(d) [5 marks] EasyDVD wants to add a new *subscription service* in addition to the existing ad hoc rental service. In a subscription service, customers maintain an ordered lists of movies they would like to rent, and whenever they return a movie, the next movie on the list is sent out. For the subscription service, customers pay a fixed monthly fee rather than individual rental fees on each movie, and each rental is open-ended – i.e. there is no due date and no fine. How would you modify the class diagram to facilitate this service?

You would need to add a special type of account, which could be done by subclassing the Account class. Note that to keep a request list, we need to distinguish between movies (which are subject to requests) and the actual DVD, which is what gets loaned. Here's one possible solution:

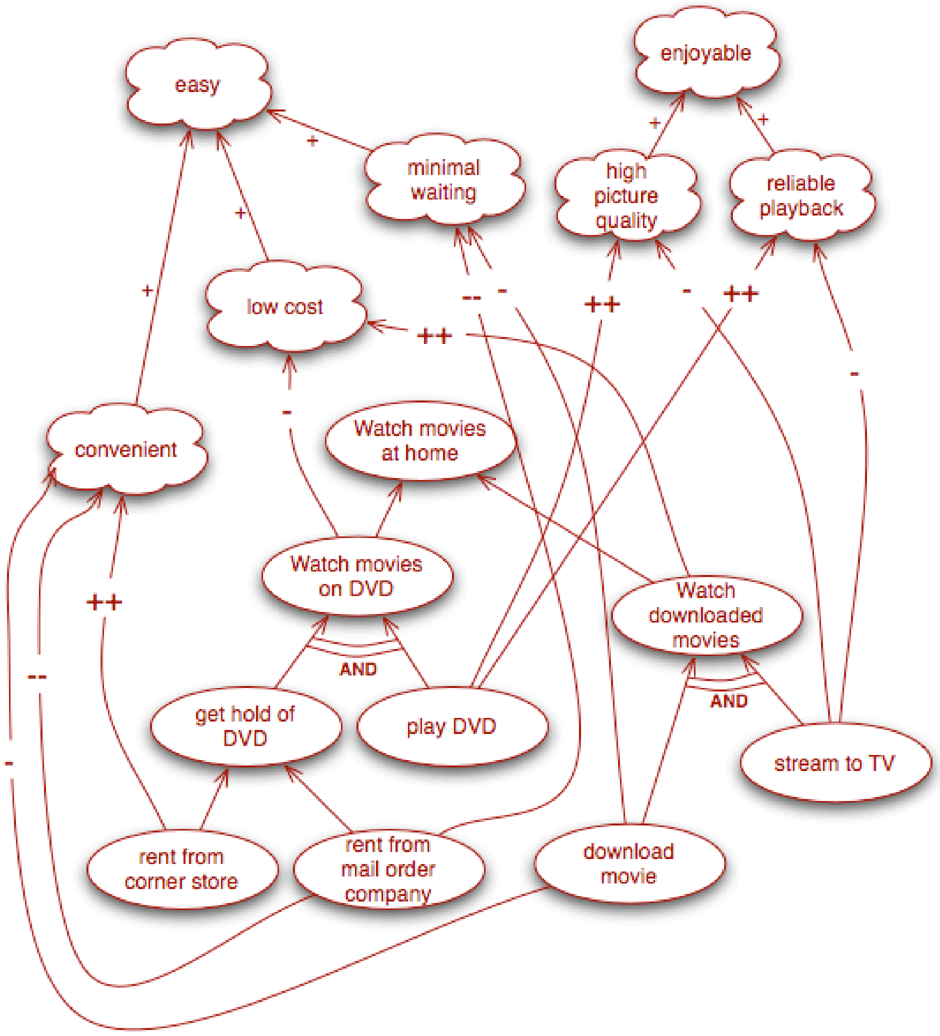


3. **[Sequence Diagrams – 20 marks]** Sketch a UML Sequence Diagram for the process of renting a DVD for the EasyDVD service described at the start of the previous question (i.e. the regular rental, ***not*** the subscription service). Be sure to show the customer logging into her account and selecting a DVD to rent, and show how the system sets up a new loan with the appropriate dates set, using the Date class to calculate these.



[Note: must state assumptions to get full marks]
This solution assumes the Account class acts as a controller to set up the loan.
Also assumes that the requested movie is available, and that there is some mechanism to lock the requested movie until the loan is confirmed, to prevent someone else renting it while the loan is set up.
Also assumes that the date class has a method to add a duration to a date (this was not shown on the original class diagram).
Also assumes returns from method calls are implicit.
Also assumes that when a new Loan object is created, you have to tell it which movie it refers to (shown by passing the parameter 'm' here)
This solution doesn't show payments, nor setting up delivery address, as we assume these are already taken care of somewhere (and the question didn't ask for them)

4. **[Goal Modeling – 20 marks]**. Sketch a goal model to represent the following information elicited from Fred about his preferences for watching movies at home. *Fred's main concerns are that watching movies should be easy and enjoyable. By easy, he means that it should be convenient, low cost, and involve minimal waiting. By enjoyable, he means the movie should have a high picture quality, and it should play reliably. He has experienced two ways of watching movies: DVDs and downloads. He finds downloads to be very cheap, while DVDs are not. For DVDs, he first has to rent the DVD, then play it. He can rent DVDs from his corner store, which is very convenient, or from a mail order service, which is very inconvenient, and involves a long wait. For downloads, he has to first download the movie, then stream it to his TV. Downloading is slightly inconvenient, as he has to wait a little while. He finds that playing DVDs gives him high quality picture and a reliable playback, while streaming downloads to his TV is neither high quality nor reliable.* Your goal model can use any suitable notation, but must distinguish softgoals from hardgoals, and different types of goal contribution link must be clearly labeled. State any assumptions.



Note: In this notation, clouds represent softgoals, ovals are hard goals. Goal contribution links are shown as ++ (makes the goal), + (helps the goal), - (hurts the goal), -- (breaks the goal).