Midterm Exam (Take Home)

CS 414 Object-Oriented Design, Fall 2012

100 points

Due via RamCT: 11:59 PM MST, Saturday, October 13, 2012

Honor Pledge

"I have not given, received, or used any unauthorized assistance."

(TYPE YOUR NAME HERE AS A PROXY FOR YOUR SIGNATURE)

Instructions

- Copy and paste the Honor Pledge above into your document and type your name as a proxy for your signature. Note that not signing the honor pledge will not be considered as evidence that a student has committed academic misconduct. Students whose religious tenets prohibit taking oaths will not be expected to sign the pledge. More information on the honor pledge can be found here (`http://tilt.colostate.edu/integrity/honorpledge/howDoCSU.cfm`).
- Include your name on each page of your answers.
- Questions must be answered in numerical order.
- Answers must be typed. Code must be typed. Diagrams may be hand drawn and scanned as long as they are legible.
- Answers should be succinct, coherent English prose. Make only the main points with necessary support.
- Keep your answers as specific as possible. Avoid generalities.
- The exam must be turned in as a single document (PDF only). Code and diagrams must be included in the same document. Separate code or diagram files will not be accepted.
- This exam is take-home, due at 11:59 PM MST, Saturday, October 13, 2012. Late submissions are not allowed.
- Do not post comments and questions about the exam to the course discussion board. Send questions directly to the instructor via email (`cs414@cs.colostate.edu`).
- You may use your notes, books, and available articles, but may not consult with other people, except the instructor.
- You must cite your sources properly. Any verbatim quotations must be enclosed in quotation marks, with page numbers indicated. You will receive severe point deductions for using material from the text or other sources without proper citation.
- See the CS department student information guide for guidelines on legitimate and illegitimate consultation.
- Submit your document via RamCT.

Answer all parts of all nine (9) questions.

1. (8 points) What is meant by the term "low representational gap" in Larman's text? What are the advantages of having low representational gap? What is the role of domain modeling in achieving low representational gap?

2. (8 points) Explain with an example how the controller GRASP pattern from Larman's text achieves model-view separation. The example may use UML class and sequence diagrams, and even pseudo-code or Java code snippets. Do not use the example given in Larman's text.

3. (8 points) Explain the difference between the concept of aggregation and composition in UML class modeling. Give two examples, one for aggregation and one for composition, and provide a class diagram for each example. Do not use the example given in Larman's text.

4. (8 points) Why should you express system events "at the level of intent rather than in terms of the physical input medium or interface widget"? Your answer should include an example of a good and bad description, but don't use the example in Larman's text.

5. (5 points) Consider the three sequence diagrams in Figure 1. They represent three different implementations of the same operation. From each of these diagrams what can you infer about the cohesion of class `A` and its coupling with other classes? Among the three, which implementation would be ideal, and why?

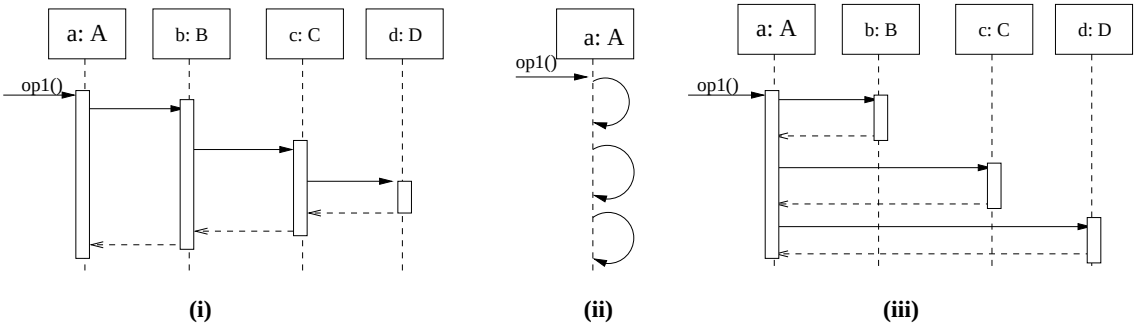


Figure 1: Sequence Diagrams Showing Different Designs for the Same Operation op1 (other operations not labeled)

6. (8 points) Figure 2 shows a class diagram and Figures 3(a)–(d) show four object diagrams. Your job is to state for each object diagram, whether or not any constraint described in the class diagram is violated. If you think a constraint is violated, specifically state the constraint. If you think the object diagram is consistent with respect to the class diagram, state so.

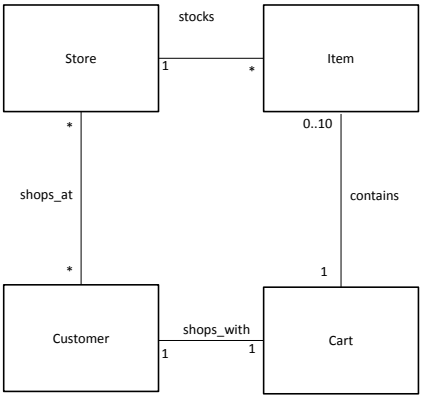


Figure 2: Class Diagram

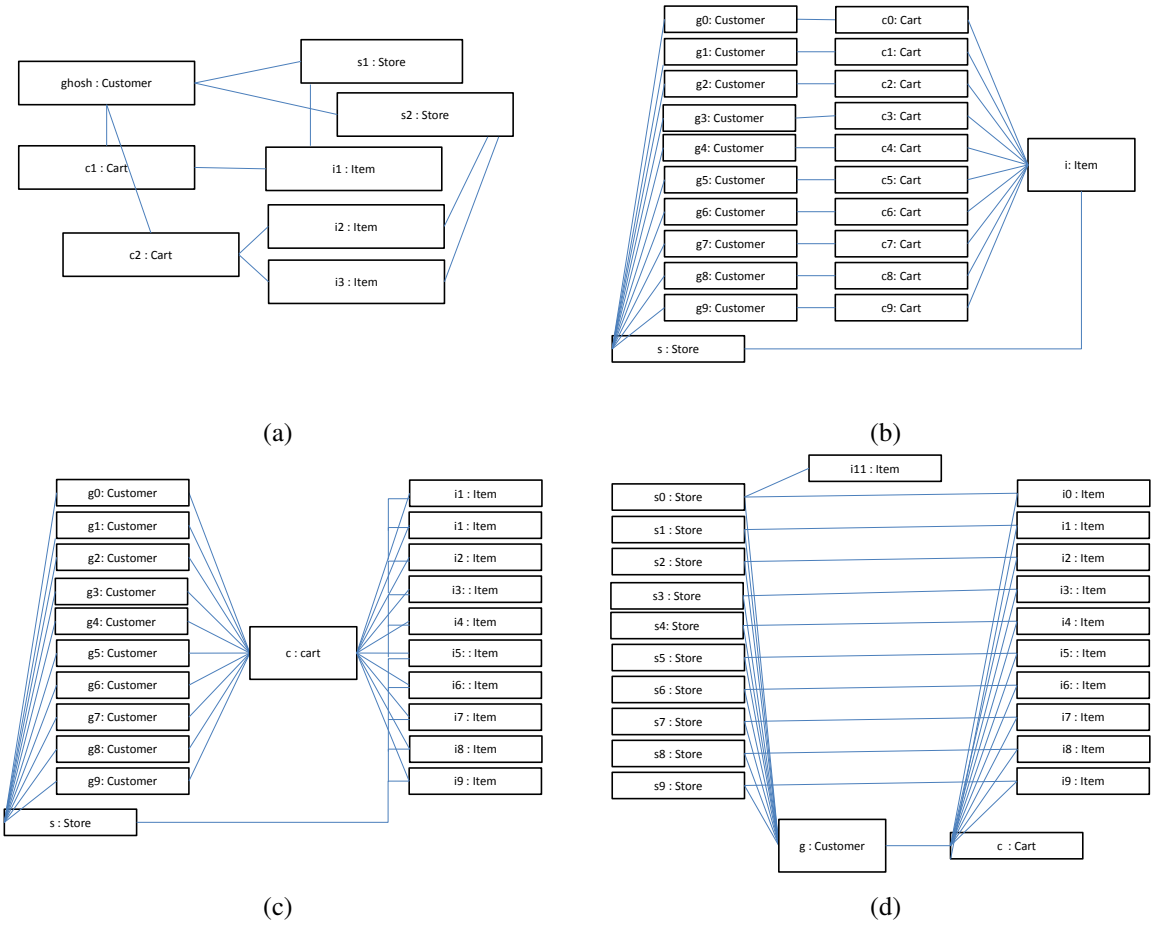


Figure 3: Object Diagrams

7. (15 points) Choose any one of the following two applications, each with an overall user goal and a use case. Note that several use cases may be needed to achieve the overall user goals.

   - *Hotel reservation system for a hotel chain*

     Overall goal: Enable users to search for hotel availability, book hotels room(s), modify or cancel bookings, join frequent traveler program, earn frequent traveler points, redeem points, and add a frequent traveler number to an existing booking.

     Example use case: Modify Booking. This requires an existing booking, which can be changed in various ways (e.g., new dates, new room type, new rates, such as prepaid, AAA, government, and regular). The frequent traveler number (if already entered in the booking) and name cannot be changed. Any other traveler information can be changed (e.g., mailing address, phone number, email address). A frequent traveler number can be added to an existing booking.

   - *Social networking site:*

     Overall goal: The overall goal is to network with friends.

     Example use case: Create a Profile. This requires providing personal information, some of which is required, and some optional.

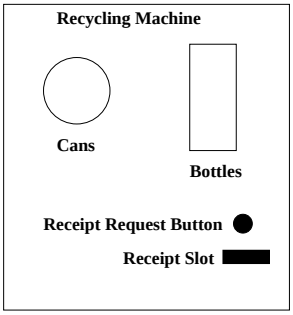
   (a) (5 points) Draw a use case diagram showing five use cases for the application you selected, including the **already provided use case** as an example.

   (b) (10 points) Describe in detail the **already provided use case** for the application you selected. Mention the pre-condition for this use case. Do not include the steps for logging in as part of the description of the scenario of the use case. Be specific about the types of information used in the description. For example, saying *Name*, *Address*, and *Gender* is more specific than just saying *Personal information*. Include one main scenario and one alternative scenario.

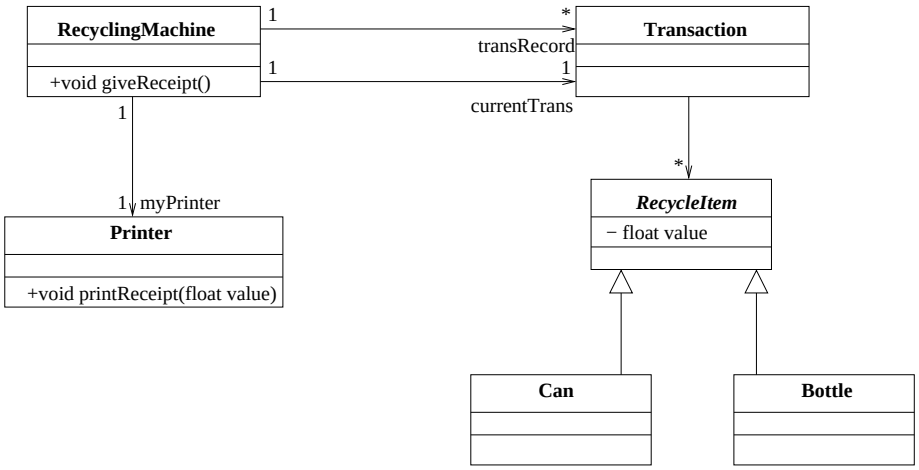
8. (15 points) The following paragraph is part of a statement for a course scheduling system to be developed for a university. Draw a UML requirements class diagram (i.e., create a domain model) from the description. Show all concepts that can be identified from the problem statement. Include associations and attributes. Label associations and show multiplicity. Provide a glossary defining all the concepts, attributes, and associations.

   > The university has several departments and buildings. Each building has several rooms and can house more than one department. Each department has a campus address that contains the building name. A department offers several courses each semester. A course can be crosslisted by at most one more department. Courses have between 1 and 5 sections. Each section is scheduled to meet at an assigned room on campus. Each department has a number of faculty members who teach the courses. These courses appear on the catalog for the semester. Each section of a course is taught by only one faculty member. The scheduling system is used to assign rooms and meeting times to sections. Faculty and students can use the system to view the complete schedule for the semester. Schedule administrators enter information regarding the courses that are offered, how many sections are required and the faculty that are teaching the course.

9. (25 points) Consider a software system that supports a Recycling Machine for recycling cans and bottles with the physical design shown below.



The following UML Class Diagram represents the design of a portion of a software system that controls the Recycling Machine. The system registers the items that each customer returns. When the customer has finished depositing items, the system prints a receipt describing the total return sum to be paid to the customer. The system also maintains a record of all transactions; there is one transaction for each generated receipt.



Consider the `giveReceipt()` operation in the Recycling Machine software system design above. When the Customer presses the *Request Receipt* button, the system operation, `RecyclingMachine.giveReceipt()` is invoked; `giveReceipt()` does the following:

(a) Computes the total value of the customer's deposit.

(b) Prints a receipt showing the total value of the deposit.

(c) Adds the current transaction to the list of `Transactions` in the `RecyclingMachine`.

(d) Clears the current transaction to prepare for the next customer.

The questions are:

(a) (15 points) Draw a UML sequence diagram that models the `giveReceipt()` operation.

(b) (10 points) For the messages involved in the interactions initiated by the `giveReceipt()` operation:

i. List all operation calls (and operations for any abstract classes) involved in the interactions.

ii. Assign each operation to appropriate classes and draw a class diagram.

iii. Specify the GRASP pattern(s) to justify your choices.