Com S 362 Name: ______________________
Spring 2003

Object-Oriented Analysis and Design

Answers to Exam 2 on Requirements Analysis

This test has 6 questions and pages numbered 1 through 6.

Reminders

This test is open book and notes. However, it is to be done individually and you are not to exchange or share materials with other students during the test.

If you need more space, use the back of a page. Note when you do that on the front.

This test is timed. We will not grade your test if you try to take more than the time allowed. Therefore, before you begin, please take a moment to look over the entire test so that you can budget your time.

For diagrams and programs, clarity is important; if your diagrams or programs are sloppy and hard to read, you will lose points. Correct syntax also makes some difference.

1. (5 points) You are working in a company that is doing time boxed, iterative development. The day before the end of the current iteration, you realize that your part of the project will not be completed on time. Should you:

   (a) Tell your manager that you need an extension of the deadline for your part of the project, to take into account the time needed to complete the functionality you were supposed to implement.

   (b) Tell your manager that you need to reduce the scope of your part of the project by eliminating some of the functionality you were supposed to implement.

   (c) Tell your manager that you need to go to New Orleans for a vacation.

   Which of these alternatives best fits with time boxed iterative development? Pick one (and only one) of these alternatives, and give a brief explanation of why this is the best alternative.

   Answer: Answer (b) is correct. You need to reduce the scope of the project, because it is important to finish to the iterative cycle on time to get feedback. Also, this is the essence of iterative development.

2. (12 points) Consider an e-mail client system (such as Pine, Eudora, or Outlook); that is, a system that can read and send e-mail for users, and manage their e-mail files and addresses. For each of the following descriptions, you are to decide whether it satisfies the elementary business process (EBP) guidelines. Write "yes" or "no" (but not both) and give a brief description of why this does or does not follow the EBP guidelines.

   (a) Exchange several e-mail messages in a conversation between the User and one or more other people.

   Answer: No, this does not satisfy the guidelines because it does not happen to one person and at one time. It's too large.

   (b) In the list of the e-mail messages sent to the User, change the selected e-mail message to the previous message.

   Answer: No, this is too low-level. It doesn't have measurable value by itself.

   (c) Read an e-mail message (i.e., present it to the User).

   Answer: Yes, this adds measurable value, because it is something that the User wants to do with e-mail, and it occurs at one time, and with one user, and leaves the system in a consistent state.

3. (10 points) Briefly describe one *non-functional* requirement for an e-mail client system.

   Answer: It should be reliable in that it should not lose e-mail messages.

4. (13 points) Consider a simple e-mail client system (such as Pine, Eudora, or Outlook); that is, a system that can read and send e-mail for users, and manage their e-mail files and addresses. Using the brief format, write a use case titled "Send Single Recipient E-mail," which would involve a User composing an e-mail message, with a subject and body, and sending it to one person. (That is, no one else is to receive the message.)

   You should write your use case for a straightforward e-mail client, without lots of embellishments or fancy features. Keep it simple.

   Be sure your use case is written in an essential, UI-free style. You only need to consider the "happy path."

   Answer:

   Send Single Recipient E-mail: The User tells the System the name of the intended recipient of the e-mail message. The System confirms that it knows this name by telling the User the e-mail address of the recipient, and the System remembers the recipient. The User tells the System of the subject of the e-mail. The System confirms that the subject is not empty, and remembers it. The User tells the System the body of the e-mail message. The System remembers the body and asks the User if they want to send the message. The User confirms that they want to send the message. The System sends the message to the intended recipient, confirming to the User that the recipient e-mail address is legitimate (i.e., it exists on the network) and that the message was sent.

5. (25 points)

   Consider again a simple e-mail client system. Using the casual format, write a use case titled "Reply to E-mail Message," in which the User answers an e-mail message. In the main success scenario, assume that the e-mail message has just one recipient, namely the User, and that the User wishes to reply only to the sender of the e-mail message (i.e., to no one else). Also, in the main success scenario, assume that the sender has a legitimate network address.

   To save time, in addition to the main success scenario, just write two alternate scenarios: (a) what happens if the User decides that, after composing the reply, they do not wish to send it after all, and (b) what happens if the e-mail address of the sender of the original message is not legitimate (i.e., it does not exist on the network).

   Be sure your use case is written in an essential, UI-free style.

   Answer: **Reply to E-mail Message**

   Main Success Scenario: The User tells the System about an e-mail message they wish to reply to. The System confirms that this e-mail message has only one recipient, and begins composition of a reply remembering the sender of the selected message as the reply's recipient, making the reply's subject be "Re: " followed by the selected message's subject, and making the reply's body being a quoted version of the original message (without attachments). The User tells the System the desired e-mail's subject. The System remembers this as the subject of the new message. The User tells the System the desired message body. The System remembers the message's body and asks the User to confirm that they want to send the e-mail reply. The User confirms that they want to send the reply. The System sends the reply, confirming to the User that the recipient e-mail address is legitimate (i.e., it exists on the network) and that the message was sent. The system also remembers that the selected message was answered.

   Alternate Scenarios:

   If the User does not wish to send the reply, then the System forgets the composed reply. The use case ends.

   If the e-mail address of the sender of the original message is not legitimate (i.e., it does not exist on the network), then the System informs the User of this problem and asks the User if they want to save the reply message. The User decides not to save the reply. The use case ends.

6. (35 points) Consider again a simple e-mail client system. Using the fully-dressed format, write a use case titled "Save Attachments," in which the User saves one or more, but not necessarily all, of the attachments in an e-mail message to files (on their computer). (An *attachment* is the contents of a file that has been transmitted inside an e-mail message in a format that allows an e-mail client system to extract its contents.) In the main success scenario, assume that the e-mail message has several attachments all of which can be successfully saved to files.

   To save time, in addition to the main success scenario, just write two alternate scenarios. These should describe (a) what happens if the file in which the User wishes to save an attachment is not writable, and (b) what happens if an input/output error occurs during the writing of an attachment to disk (e.g., the disk is full). Within each of these alternative scenarios you need only consider the "happy path" (i.e., you don't have to consider alternatives to the alternatives).

   Also, you can leave the "Technology and Data Variations List" section empty. You can consider failures and recovery to be an open issue.

   Be sure your use case is written in an essential, UI-free style.

   Answer: Use Case: Save Attachments

   Primary Actor: User

   Stakeholders and Interests:

   – User: wants fast, accurate treatment of e-mail attachments.

   Preconditions: The User has been authenticated.

   Success Guarantee (Postconditions): The attachments and the selected e-mail message have been saved into files in the User's computer.

   Main Success Scenario (or Basic Flow):

   1. The User tells the System about an e-mail message with attachments that they would like to save attachments from.
   2. The System confirms that the selected e-mail message has attachments.
   3. The User tells the System about an attachment in the selected e-mail messages that they want to save, and the name of a file to save it into.
   4. The System confirms that the named file is writable, and saves the contents of the attachment into the named file.

      The User repeats steps 3–4 until indicates done.

   Extensions (or Alternative Flows):

   4a. If the named file is not writable,
      1. The System tells the User that the file is not writable, and asks the User for a different file name.
      2. The User tells the System the name of a file that is writable.
      3. Use case continues from step 4 in the main success scenario.

   4b. If an input/output error occurs during the writing of an attachment to disk,
      1. The System tells the User about the input/output error.

2. The use case ends.

Special Requirements:

- The System must respond to the User, at least giving some progress indication, within 10 seconds, 90% of the time.
- Users are not experts, because they did not use the system continuously, so the interface must guide them through the process of saving attachments.

Technology and Data Variations List: (left empty)

Frequency of Occurrence: Sporadic.

Open Issues:

- How to interface with virus scanning software?
- How to deal with failures and recovery?
- Should the System be able to open attachments directly, without having the User need to save them to disk?
- How to recover from input/output errors more gracefully?