Comprehensive Exam for Requirements Engineering
Fall 2010

Student ID_____

Answer any 4 of the following: (25 points each)

Circle the number of the questions that you wish to have marked. *If you leave this blank, questions 1-4 will be graded.*

1 2 3 4 5 6

Use only the space provided. Do not attach additional sheets.

All questions of this examination use the following requirement specification:

Consider a scenario: A person, Jill, is at the grocery store checkout counter with a cart-full of items. While the cashier is ringing these up, Jill thinks about how to pay for her purchases. Should she use the Visa card with a cash-back bonus or the AMEX card with frequent-flyer points? Which card will give her the best deal? Jill also remembers that she can get additional discounts if she pairs particular payment cards with the store loyalty card. However, she can't remember which cards the store accepts for this extra discount. Unfortunately, the cashier only knows about two cards, neither of which Jill possesses.

This scenario reflects a typical pervasive computing problem— namely, users possess large amounts of information of which they're only vaguely aware, and they have no idea what is useful in different contexts. For example, the choice Jill makes in one store may not be applicable in another that offers separate discounts.

The complexity in this scenario arises from the fact that users typically have many payment, discount, and loyalty cards while retailers have their own preferred payment cards and discount, loyalty, and reward schemes that are time, store, product, and even usage dependent. Even with a complete list of retailer options, finding the optimal set of cards for a particular situation can be cognitively demanding for users.

Providing users with such information is nontrivial, as many combinations can provide similar overall deals but in different ways— for example, one may provide cash rebates and another frequent-flyer miles. Users need to be able to quickly find the cards that give them the deals they want.

To address this problem, we have propose a cell phone application, *pFerio*. At checkout time, users click on *pFerio*, authenticate themselves, and start the point-of-sale application. They place their phone on a reader provided at the checkout counter. The system finds and suggests matches. The user selects the cards that give the deal they want and makes the payment, completing the checkout procedure.

1. What is a *non-functional requirement*? Identify 4 non-functional requirements of *pFerio* using any requirements specification technique you wish; then specify how the requirement is to be tested, i.e., what is the acceptance criterion or criteria for each requirement that you have identified?

2. S*tructured Analysis* and *Object-Oriented* analysis are the two most common methods used to analyze requirements. Provide a brief description of each and discuss the applicability of each to the *pFerio* application. Which should be used for *pFerio*? Why (include rationale for not choosing the other)?

3. Describe the characteristics of a good requirement. Identify the major activities of requirements analysis and discuss whether or not the above requirement specification supports the activity.

4. Many studies have shown that faulty Requirements Engineering is a major contributor to software failures. Identify the most common RE problems and discuss whether or not the *pFerio* requirement specification addresses these problems. Where the specification does not address a particular common problem, show how the specification should be modified to correct the weakness.

5. Select a life-cycle model for the development of *pFerio*. Identify a possible evolution of the software requirements of pFerio and its affect on the selected software life-cycle model. Discuss the general principles for determining the appropriate software life-cycle based on the expected evolution of the software requirements.

6. What is a *requirements baseline*? Which of the *pFerio* requirements should be placed in the baseline? Why? What is the purpose of placing requirements under configuration control? Describe a typical requirements configuration control procedure, i.e., what are the steps in modifying a set of requirements that are under configuration management?