Two hours

Appendices A and B are located at the back of the exam

UNIVERSITY OF MANCHESTER
SCHOOL OF COMPUTER SCIENCE

Agile Software Engineering

Date: Monday 17th January 2011
Time: 09:45 - 11:45

Please answer any THREE questions from the FOUR questions provided

Please use a separate answerbook for each Section

For full marks your answers should be concise as well as accurate
Marks will be awarded for reasoning and method as well as being correct

This is a CLOSED book examination

The use of electronic calculators is NOT permitted
Section A

1. a) For an agile model-driven approach to the development of real-world software, outline the workflow that must be followed to transform a platform independent model (PIM) of an application into implementation code. (4 marks)

b) A question and answer application is going to be developed in which users can ask questions to which other users can provide replies. In this system, it must be possible to find all of the replies to a question, the question to which a reply applies, all of the questions and answers posted by an individual user, and assign a positive integer to replies that indicates the appropriateness of the reply.

By using Emfatic, capture a platform independent model (PIM) of this question and answer application. In your version, although it is necessary to identify the questions and answers posted by an individual user, no other information about a user is required. You should assume that EMF will be used to generate some of the implementation of the application and, therefore, that your model must capture a single containment hierarchy. (6 marks)

c) If the implementation for the application described in part b) is to be made on a web platform, what PSMs are required? (2 marks)

d) When generating an implementation from a model, there are two approaches (early or late bound) to the generation of Application Programming Interfaces (APIs) that can use used by other code. By using simple examples, discuss the advantages and disadvantages of these two approaches. (6 marks)

e) One of the key aspects to model-driven software development is the use of meta-models. Briefly describe the role of meta-models and how they differ from other models. (2 marks)
2. a) Appendix A gives a simplified platform independent meta-model that allows the modelling of the persistent elements of an application. In this simplified meta-model, entity features can only be of a primitive type or uni-directional references to other entities. Appendix B gives a very simplified platform specific meta-model of Java. In this, only the basic features of Java classes are captured.

Show the ATL model-to-model (m2m) transformation that would allow the transfer of a persistent platform independent model (PIM) conformant to the Appendix A meta-model to a Java platform specific model (PSM) conformant to the Appendix B meta-model. Your transformation should assume that the target Java has a bean style (private properties and public accessors). You do not need to include a body to any Java methods that you generate. (6 marks)

b) For a model-to-text (m2t) transformation, outline the elements that a tool must provide to allow a user to describe the transformation that they wish to be performed. (4 marks)

c) A Domain Specific Language (DSL) is required to support the input of information into a computer application of which only a small number of instances will exist. Three possible approaches to the development of the tooling that parses the language and delivers the information to the application have been identified:

- bespoke tooling based on a compiler toolkit
- the use of a UML profile
- the use text-to-model (t2m) toolkit.

Discuss the advantages and disadvantages of each of these approaches to the development of tooling to support the required DSL. (6 marks)

d) For the Domain Specific Language (DSL) outlined in part c), how would the advantages and disadvantages of the three approaches be altered if many instances of the application were to exist and implementation code was to be generated from the DSL? If you identify any differences, your answer should explain why they happen. (4 marks)
Section B

3. a) For each of the following scenarios, state whether you would advise a waterfall approach to requirements specification (i.e., the creation of a complete written document up front) or an agile approach (i.e., continuous requirements gathering using user stories). In each case, justify your decision by describing the pros and cons of each approach in relation to the features of the scenario.

i) A training system is required that simulates the interview process for people wishing to emigrate to the recently created country of Zembla. The structure and details of this process are laid down by law, and the task of the software is to mimic them exactly (in order to be an accurate teaching tool for Zemblan Embassy staff). While changes to the process are to be expected in the future, none are anticipated within the lifetime of the project. (3 marks)

ii) Dulwich & Sons started out as a small family company, but in recent years it has expanded rapidly as demand for its expert advice on the rapidly evolving investment market has grown. The company now has need of a software system to provide support for long-range/strategic management planning. This is new territory for the company, which until now has been managed based upon the instincts and experience of its two Dulwich family owners. Now, however, the increased size of the company and the radical changes foreseen in the market place and the regulatory environment mean that this *ad hoc* approach is no longer adequate. Instead, a software system is required to provide reports and scenario-based planning functions, to assist the managers in their more difficult tasks. (3 marks)

b) You are a member of an agile team developing a new Website to match final year students and recent graduates with graduate placements offered by employers. You begin by identifying the roles that might be involved in such a process. List three of the key roles you might discover for this Website.

(1 mark)

c) For each of the roles you identified in your answer to part b) of this question, write an epic story that is of value to the role.

(6 marks)

(Question 3 continues on the following page)
(Question 3 continues from the previous page)

d) Select any one of the epic stories you identified in your answer to part c) of this question, and refine it into a set of smaller stories. At least one of the refined stories should be implementable in a single iteration. (In your answer, you must state the length of iteration you are assuming. Anything from 1 week to 8 weeks is acceptable. You must also state clearly which of the refined stories you believe to be implementable in this iteration length.)

(5 marks)

e) For any one of the stories you write for your answer to part d) of this question, write two contrasting acceptance tests.

(2 marks)

4. a) Suppose that you are the manager of each of the agile teams described below. For each scenario, diagnose the problem that the team is having with testing, and suggest a simple process improvement action that can be taken to avoid the problem reoccurring in future. Make clear any assumptions you make about the team and their actions.

- Team A is writing lots of unit tests, which all pass at the end of the iteration. But, when user acceptance testing begins, based on the acceptance tests written on the user story cards, lots of different defects are found.
- Team B is writing some tests, but they are being run only at the end of the iteration. When questioned, the team say there is not enough time to run tests as well as write code.
- Team C have created very few test cases so far for the current iteration. This is odd as in previous iterations this team has always created excellent test suites, and ran them regularly. On investigation, you discover that one member of the team has been on sick leave since the end of the last iteration.
- Team D is writing a lot of tests, and they are being run often. Unfortunately, the defects they are uncovering are not being fixed. A large percentage of the test cases fail at each run, and the team test results chart is covered in large red areas.

(8 marks)
b) You are part of an agile team developing a Website using Java, to allow users of public libraries to manage their accounts online. You have chosen to implement the following story as your next task on the project:

As a library user, I want to be able to pay any fines on-line using my credit card, so that I can pay at a time and place of my own convenience.

This will be implemented by a Payment class, which it is your job to create using a test-driven-development approach. Another developer will write the GUI code and the controller code which invokes your Payment class. Write three JUnit test cases for the Payment class, showing contrasting aspects of the required behaviour. Annotate each test case with a brief description of the behaviour it tests. Make clear any assumptions you make about the other classes/methods that might exist and that you need to use (e.g. LibraryUser).

Notes: do not give the implementation of the Payment class, or even its interface. Candidates will not be penalised for small errors of syntax or typos in JUnit method names. If you are not confident with writing JUnit, you can instead write a detailed pseudocode version of the test case, describing what behaviour you would expect to be implemented. It is still possible (just) to get a first class mark for your answer, even if you don’t write in JUnit. (9 marks)

c) In their book on Agile Testing, Crispin and Gregory give 10 principles for agile testers to follow, such as “Deliver value to the customer” and “Provide continuous feedback.” The fourth principle, however, is “Have courage.” Explain why courage is necessary for agile software developers in general, and for agile testers in particular. (3 marks)
Appendix A – Emfatic Platform Independent Meta-Model of Persistent Elements

@namespace(prefix=ab, uri="http://x.y.z/ab")
@Ecore(
    invocationDelegates="http://www.eclipse.org/emf/2002/Ecore/OCL")
package Persistence;

class Persistence {
    val Primitive[*] primitives;
    val Entity[*] entities;
}

abstract class NamedElement {
    attr String name;
}

class Primitive extends NamedElement {
}

class Entity extends NamedElement {
    val Feature[*] features;
}

abstract class Feature extends NamedElement {
}

class Attribute extends Feature {
    ref Primitive[1] type;
}

class Reference extends Feature {
    ref Entity[1] type;
}
Appendix B – Emfatic Platform Specific Meta-Model for Java

@namespace(prefix=ab, uri="http://x.y.z/ab")
@Ecore(
    invocationDelegates="http://www.eclipse.org/emf/2002/Ecore/OCL")
package Java;

class Java {
    val JClass[*] classes;
}

abstract class NamedElement {
    attr String name;
}

class JClassifier extends NamedElement {
}

class JPrimitive extends JClassifier{
}

enum JVisibility {
    private; protected; public;
}

abstract class JClass extends JClassifier {
    val JProperty[*] properties;
    val JMethod[*] methods;
}

class JProperty extends NamedElement {
    attr JVisibility visibility;
    ref JClassifier[1] type;
}

class JMethod extends NamedElement {
    attr JVisibility visibility;
    ref JClassifier type;
    val JParameter[*] parameters;
}

class JParameter extends NamedElement {
    ref JClassifier[1] type;
}