One and a half hours

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

UNIVERSITY OF MANCHESTER
SCHOOL OF COMPUTER SCIENCE

Agile Software Engineering

Date: Friday 20th January 2012
Time: 09:45 - 11:15

Please answer any TWO 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) In a pure model-based approach, a platform independent model (PIM) should contain no implementation details. During the transformation of the PIM to a platform specific model (PSM) implementation decisions must be made. By using examples of the sort of implementation detail that must be added, critically evaluate alternative approaches for adding this implementation detail. (5 marks)

b) An on-line shopping application is to be developed using a model-driven approach. This application will use relational database technology for its backend, a web-based client for its customer facing front-end and Java clients for internal maintenance use. For the development of this application, outline, with reasons, the development flow that would be required, and the models, meta-models and transformations that would be used by this flow. (8 marks)

c) A model-based tool is going to be developed that allows end-application developers to define a model of an application that is used to run a business and generate the application from this model. As part of the model-based tool, a meta-model of business operations is required. It can be assumed that these business applications will have to deal with things like the type of staff, customers, accounts and products that business uses, and that these objects will have attributes and relationships between themselves. Given these business application requirements, describe, using Emfatic, a meta-model suitable for use in the model-based tool. (7 marks)
2. a) By considering the Emfatic model of a bank in Appendix A and the Emfatic model of part of a website in Appendix B, outline an Atlas Transformation Language (ATL) transformation that will generate a website interface that supports the display and maintenance of the information for a bank. (7 marks)

b) A model-to-text (m2t) transformation will need to be applied to the model that results from the transformation in part a) to produce the actual implementation. Describe the capabilities that must be present in any model-to-text transformation and evaluate how successfully different model-to-text systems achieve these. (8 marks)

c) An application is to be developed that parses a domain specific language (DSL) into an internal model representation. In terms of a model-based development environment, evaluate the options that are available for the development of tooling to process this language. (5 marks)
3. a) State what is wrong which each of the following poorly written stories:

i) As a user, I want to see a list of up-coming concerts so that I can quickly check whether there are any I would like to attend.

ii) As a product owner, I want to be able to demo the system to our investors in July, so that we can retain their confidence in the project.

iii) As a fan, I want to subscribe to a mailing list of info on my favourite bands’ concerts.

iv) As a fan, I want information on all top events, so that I can follow my favourites.

v) As a ticket purchaser, I want to say how many tickets I want to buy, so I can buy them all in one interaction with the system.

vi) As a band manager, I want to create an account for our band, so that I can enter our free dates for bookings.

(6 marks)

b) Refine the following epic story using the Split-by-Operation (CRUD) strategy. In your refined stories, you should take care to use vocabulary appropriate to the domain, so that the words “create”, “read”, “update” and “delete” do not appear.

As an undergraduate, I want to manage my course choices during the course selection period, so that my final choices best reflect my interests and learning goals.

(4 marks)

c) The following epic story is created by a team charged with developing the next generation of learning environments, that will succeed tools like Blackboard and Moodle. Refine the story using the Split-By-User strategy into 4 smaller stories. The 4 stories you produce do not have to be a full cover of the original epic. (Hint: you will find it useful to begin by listing as many different classes of user of tools such as Moodle as you can think of. You can then choose which of these users you will refine the story for.)

As a course participant, I want to make use of the team project wikis provided by the learning environment, so that I can keep track of progress throughout the lifetime of the project.

(6 marks)

Question continues on next page
3) (Continued from previous page)

d) You are part of an agile team (with 6 members) building the learning environment introduced in part c). The goal of your client is to use the learning environment to provide a consistent base line of learning support to all students, regardless of lecturer preferences and teaching styles.

An initial story gathering workshop has created many stories, describing the kind of functionality that you are used to from Moodle and Blackboard. You are now entering the first (two week) iteration, and must plan which functionality you will aim to deliver. The existing infrastructure you have to build on is:

- A relational database of information about course units (mainly syllabus information, but with some details of dependencies between course units, and relationships to programmes).
- A relational database containing basic student details, and information about which course units they are enrolled on.
- An XML file with information on teaching staff and their research interests, which is updated weekly.
- Some learning resources (e.g. handouts, exercise sheets, self-test results) are made available in Moodle, some in Blackboard and yet others on lecturer’s individual web pages using their own directory structures and file naming conventions.
- APIs for programmatic access to both Moodle and Blackboard exist.

What functionality would you choose to deliver by the end of this first iteration to give maximum value to the customer? Justify why you believe this could be achieved in the small amount of time available, and state clearly what the value is to the client in having this functionality implemented and deployed. (4 marks)
4.

a) Consider the following user story:

As a member of marketing staff, I want to offer discounts on orders from repeat customers, so that we can encourage customer loyalty.

i) What is the earliest point at which tests/examples for this story can be written? (1 mark)

ii) What is the earliest point at which tests for this story can be executed? (1 mark)

iii) In conversation with the customer about this user story, you discover that the marketing team wish to offer the following discounts to customers who are members of the company’s loyalty scheme:

- Free shipping on orders over £30 (not including shipping costs)
- Cheapest item free when 3 or more items are ordered

Design an acceptance test table that has the necessary columns to allow specification-by-example to be used to specify this behaviour. In designing the table, you should discover an ambiguity in the above description. Say what this ambiguity is, and explain the precise behaviour you will describe in your acceptance test table. (4 marks)

iv) Using the acceptance test table you designed in your answer to question a) iii), give 4 contrasting examples that specify the behaviour elicited from the conversation with the customer. (2 marks)

v) Write a JUnit test case (or, for almost full marks, a close pseudocode equivalent) for any one of the examples given in your answer to question a) iv), from which you can elicit some of the classes and methods needed to implement this application in a test-first manner. (Note: you should not give the implementation of any of the application classes/methods in your answer – only the signatures are required. Candidates will not be penalised for minor syntactic errors in Java code.) (5 marks)
b) Takeshi Kakeda has proposed a “Big Visible Chart” (BVC) approach to defect tracking that uses Lego. When a defect is discovered, a Lego figure is created which represents visually the estimated severity and complexity of the defect. For example, a big scary Lego monster would be used to represent a serious major bug, while a small cute Lego figure would represent a relative harmless defect. A post-it note is attached to give a basic description of the defect, and defects which are dependent on others are represented by vertically stacking the Lego bugs. When setting out to fix a defect, a team member takes the Lego figure for that defect to his/her desk, and disassembles it when the fix is verified, returning the Lego bricks to the box.

i) Discuss whether this approach to defect tracking delivers the benefits expected from an application of the BVC principle. Point out any benefits of BVC which this approach does not deliver. (3 marks)

ii) Is this approach to defect tracking one that can be applied successfully on a wide range of projects? Justify your answer, by describing the characteristics of a project for which it would be suitable and one for which it would not be appropriate. (4 marks)
Appendix A – Emfatic Model of a Bank

```java
package EBank;

enum MaritalStatus {
    Single; Married; Divorced; Widowed;
}

abstract class Person {
    attr String[1] name;
    attr int[1] age;
    attr MaritalStatus[1] maritalStatus;
}

class Female extends Person {
    ref Male[0..1]#wife husband;
}

class Male extends Person {
    ref Female[0..1]#husband wife;
}

class Bank {
    attr String[1] name;
    attr String[1] phoneNumber;
    val Branch[+]*#partOf branches;
}

class Branch {
    attr String[1] sortCode;
    ref Manager[1]#manages manager;
    ref Teller[1..20]#worksAt tellers;
    ref Customer[*]#branch customers;
    val Account[*]#heldAt accounts;
    ref Bank[1]#branches partOf;
}

enum AccountCurrency {
    GBP; EUR; USD;
}

abstract class Account {
    attr String[1] accountNumber;
    attr Date[1] openingDate;
    attr double[1] interestRate;
    attr double[0..1] overdraftLimit;
    attr AccountCurrency[1] accountCurrency;
    val Transaction[*] transactions;
    ref Branch[1]#accounts heldAt;
    ref Customer[1]#accounts ownedBy;
}

class CurrentAccount extends Account {
}

class SavingsAccount extends Account {
}
```
enum TransactionType {
    Withdraw; Deposit;
}

class Transaction {
    attr TransactionType[1] transactionType;
    attr Date[1] date;
    attr double[1] amount;
}

class Customer extends Person {
    ref Account[+]#ownedBy accounts;
    ref Branch[1]#customers branch;
}

abstract class Employee extends Person {
    attr int[1] employeeId;
}

class Manager extends Employee {
    ref Branch[1]#manager manages;
}

class Teller extends Employee {
    ref Branch[1]#tellers worksAt;
}
Appendix B – Emfatic model of a web interface

```scala
package Web;

class Website {
  val Page[+] pages;
}

class Page {
  val PageElement[+] elements;
}

abstract class PageElement {
}

class Table extends PageElement {
  val TableRow[+] rows;
}

class TableRow {
  attr String[1] label;
  attr String[0..1] value;
}

class Form extends PageElement {
  val FormField[+] fields;
}

class FormField {
  attr String[1] name;
  attr String[0..1] currentValue;
}
```

END OF EXAMINATION