PAPER CODE NO.
COMP 319

EXAMINER : Mr Sebastian Coope
DEPARTMENT: Computer Science

UNIVERSITY OF LIVERPOOL

JANUARY 2015 EXAMINATIONS

ANSWERS
ANSWERS
ANSWERS

COMP319 : SOFTWARE ENGINEERING II

TIME ALLOWED : Two and a Half Hours

INSTRUCTIONS TO CANDIDATES

Answer ALL questions in Section A

(Section A is worth 70%)

Answer One question from Section B

(Section B is worth 30%)

Section A

Answer ALL questions in Section A. Section A is worth 70% of the marks available.

A1 **(a)** Describe the open/closed principle for OO development, its benefits and illustrate how it can be implemented in code written in Java? **[10 Marks]**

This principle explains that an object class must be closed for modification but open for extension. [2 Marks] This has the benefit of keeping the current code base stable and bug free but allowing it to be used as a basis for future code.[2 Marks]

One way to close the Java class is to define its interface in a Java interface definition. If the class is only used via this abstract interface, changes to the class definition itself will not change its interface. The second way a class can be closed for modification is to use the word final on its method definitions, this way the method cannot be overridden. Finally the use of a protected overridable method can be used as an extension point to actually implement code that can be added to the base class. [3 Marks]

```
public interface ICardProvider {
        public boolean makePayment(String name);
}
public class CardProviderBase implement ICardProvider {
        public final boolean makePayment() {// This is closed for modificatin
                        // Default code to test for stolen cards etc.
                return(onMakePayment());
        }
        protected boolean onMakePayment() {     // This is open for extension
                return(false);          // default implementation just fails
        }
}
public class NatwestCardProvider extends CardProviderBase {
        protected boolean onMakePayment() {
        }
}
```

(b) Explain with the aid of examples and class diagrams the Model/View/Controller architecture OO design pattern? **[20 Marks]**

MVC Model/View/Controller

This is an architectural pattern which splits the code into 3 parts, model, view and controller.

Model

This stores application data as well as handle all the business logic, rules (e.g. who can access a student's transcript) for the system. It is also responsible for validation (can also be in controller), data persistence, application state (for example keeping a reference to a user's shopping cart in an online session). **[2 Marks]**

View

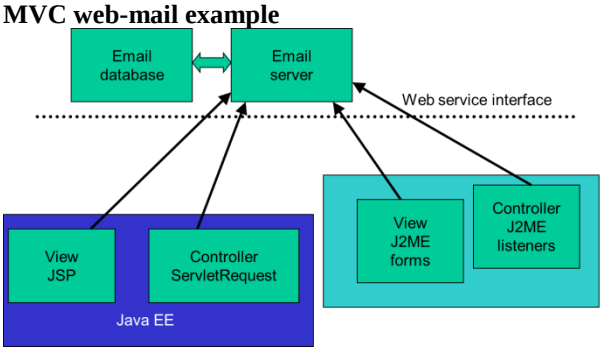
This part of the code renders the data to a format that can be presented to the user, for example for a web application the view code would generate HTML code that was sent to the user's browser. **[2 Mark]**

Controller This part of the code interprets user input (such as mouse clicks or keyboard input) and sends it to the model. For GUI interfaces, each on screen widget capable of input, has typically an associated piece of controller code. **[2 Mark]**

MVC benefits of

Makes it easier to change the user interface, (the VC) code without having the modify the model code. This is because the VC code can be changed without exposing the critical business logic to change. The MVC architecture makes it simpler to support multiple interfaces to the application. The module can present a standard public interface that the different VCs can be connected into. Using MVC the software can be developed by two separate teams, one's with skills in GUI and others in skills in working with database technologies and the application area. **[4 Marks]**

MVC web-mail example



[4 Marks for diagram]

The user goes to the home URL for the mail service and the VIEW sends a user a login page in HTML to the user's browser. The view code could render different versions of the HTML or use CSS to work with different sizes screens etc. The user types in there user/name and password and presses the submit button. The controller software intercepts the submitted Form page, constructs a Login command object and sends it to the Model command handling code. The Model checks the user's credentials against the database and then makes a request to the view If the user's credentials fail, the VIEW is requested to generate a failed login page. If the login is successful, the model code retrieves the user's current email inbox from the database and makes a request to the View to display this data to the user. The VIEW generates the HTML which is sent back to the user's browser.

[6 Marks for explanation]

A2 (a) Describe and explain the benefits of the poker planning technique for project time estimation. **[10 Marks]**

Each member of planning team given pack of cards with numbers on, (sometimes playing cards are used). Then project manager introduces project and the team asks questions and clarifies assumptions with the project and discusses any possible risks. Each member picks a card from their pack as their estimate and puts it face down, then all members show their cards at the same time. The members with the lowest and highest estimation members given chance to justify their decision to the whole group. The whole process continues until a consensus is reached, typically the developer(s) who will be responsible for the project, will get a larger share of the vote than others.

[6 Marks]

Benefits

Planning poker has been found to help avoid a phenomena called anchoring. This a where in open discussion members of the team who strongly advocate a particular time estimate and this persuades other team members to follow their lead. Low anchors i.e. low estimations will generally come from sales, marketing or other product owners. High anchors (sometimes overly pessimistic) will generally come from development team members. A study on planning poker [K. Molokken-Ostvold and N.C. Haugen] has found that estimates were less optimistic and more accurate than ones obtained through the simple combination of individual estimates for the same tasks.

[4 Marks]

(b) Describe the SCRUM development lifecycle. **[10 Marks]**

SCRUM is an iterative development technique which splits the lifecycle into a series of development efforts called sprints. Each sprint lasts around 1 to 4 weeks and is preceded by a sprint planning meeting. [2 Marks]

At the start of the project and as the project moves forward, the total work needed to complete the project is contained within a document called the product backlog, as items are completed they are removed from the product backlog. This can include anything from product features to debugging tasks and technical tasks. Each item in the product backlog has a description/specification as well as a score in terms of its business value. Each item in the backlog is also given an estimate by the development team in terms of the effort required to complete it. [4 Marks]

In the sprint planning meeting it is decided the scope of work to be done, how long it is going to take, this work is added to the sprint backlog. The amount of work is carefully selected so that the it is enough to fill the sprint, this is done by looking at the workload delivered by previous sprints and what is called the projects velocity. Any work that is not complete at the end of a sprint is returned to the product backlog.

[2 Marks]

Daily SCRUM This is a short meeting in which all developers answer the following questions.. What have you done since yesterday? What are you planning to do today? Any impediments/stumbling blocks? [1 Mark]

As the project progresses its progress is measured using a burn down chart which gives you a clear indication of how much has been completed. [1 Mark]

A3 Explain and critique the XP pair-programming approach, in your answer refer to relevant research results. **[10 Marks]**

Two programmers work together to complete the programming task, one programmer writes the code while the other programmer, reviews the code, makes suggestions and comments on the approach. The two programmers swap places on a regular basis.

Pair programming provides the benefit of reducing risk, because the change of a design error or serious flaw passing 2 programmers working on the task is a lot lower than 1 programmer. A programmer working in a pair is a lot less likely to implement a "hack" or shortcut which is structurally unsound. The observer has the time to review the code while it is being produced, therefore ensuring higher levels of quality. The concept of collective code ownership is important here, it means that responsibility for code that is spread between developers. [5 Marks]

Looking at the research does not show conclusive benefits for pair programming, we can compare studies of Arisholm, Gallis and Sjøberg and Laurie Williams of the University of Utah.

The Williams et al. 2000 study showed that using pair programming there was a decrease in time to develop between 15% and 30% but this did require an increase in effort (programmer hours) between 15 to 60% with an increased in correctness of 15%.

Arisholm, Gallis and Sjøberg carried out research to determine how effective pair programming was in different contexts, for example with complex and relatively simple problems and with different combinations of staff skilling. They used a fixed set of problems and

split the developers into a pairing group and a single programming group.

They found that for most tasks the time taken was not significantly different when using a pair programming on average 84%, the amount of time was reduced but not by a large amount on average a reduction of 8%, however if the pairs were junior the increase in effort was much larger and the 111% and the time taken was larger as well.

There was however a positive outcome in terms of correct solutions found overall

of 7% and 73% for juniors.

One of the most interesting studies was a meta-analysis of many studies carried out by Jo E. Hannay et al. This showed small reductions in time (depending on the task complexity, less complex tasks have an improved time greater than complex tasks. They also showed that pair programming produces higher quality results for more complex problems. They also discovered a research bias in favour of pair-programming by some of the leading researchers in the area.

In general pair-programming results are somewhat mixed it seems to be able to deliver code slightly faster, of higher quality but at considerable cost. However the overall cost of the product for its whole life-cycle should be taken into account so quality improvements earlier on in development could result in savings later on, these are hard to quantify with short term studies and a more longer term study may produce more conclusive results.

[5 Marks]

A3 Look at the code segment show in Appendix A. For this program construct the following slices:

Forward slices for int sum=0 and int product=1 **[5 Marks]**

Sum=0
Int sum=0
Sum=sum+1
Write(sum)
product=0
product=product*i
write(product)

Backward slides for write(sum) and write(product) **[5 Marks]**

```
write(SUM) slice
int N=5;

int sum = 0;
for(i = 1; i < N; ++i) {

  sum = sum + i;


}

write(sum);

write(product)
int N=5;

int product = 1;

for(i = 1; i < N; ++i) {

  product = product * i;

}
write(product);
```

Section B

Answer ONE question from Section B.

Section B is worth 30% of the marks available.

B1 The 1996 Chaos report is widely cited as a showing clear evidence for a crisis in the software industry. However there have been a number of papers which criticise the Chaos reports findings.

(a) Summarize the criticisms given in the "The Rise and Fall of the Chaos report figures" and explain the authors own approach in evaluating the project performance of organisations. **[20 Marks]**

This report found the following problems with the Chaos report.

The Standish report classified project outcomes in the following categories:

Type 1, project success. The project is completed on time and on budget, offering all features and functions as initially specified.

Type 2, project challenged. The project is completed and operational but over budget and over the time estimate, and offers fewer features and functions than originally specified.

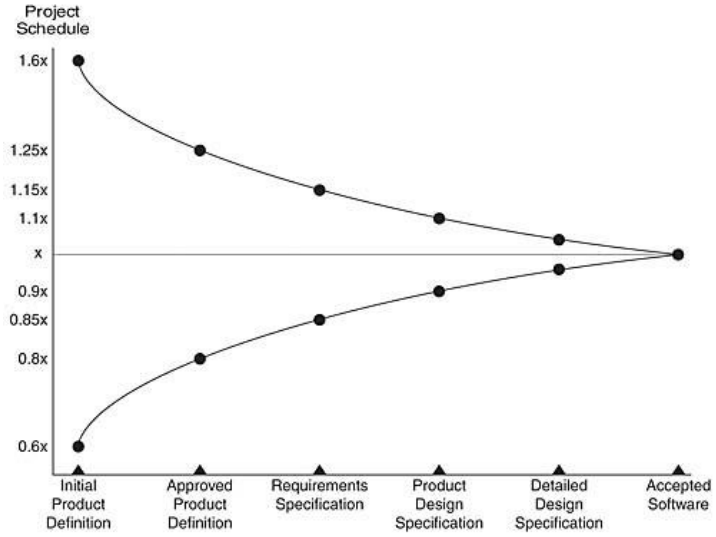
Type 3, project impaired. The project is cancelled at some point during the development cycle.

The first problem with this classification is that it is incomplete, For instance, a project that's within budget and time but that has less functionality doesn't fit any category, this was pointed out in a Chaos review paper by Robert Glass and Magne Jørgensen.

The other more serious issue is looking at what the Chaos report was measuring, in summary projects classified as successes has a forecast/actual ratio>=1 for time (no overrun) and a forecast/actual ratio<=1 for functionality (functionality complete)

However these ratios depend on two variables the actual project value and the forecast value, so unless an organisation can either overestimate or get the forecast time correct, the project will overrun based on an incorrect forecast (not because of a failure to deliver). This problem of project forecasting was examined by Barry Boem and described by a concept which he referred to as his cone of uncertainty.

This is illustrated below:



The points on the line show the typical maximum and minimum values for forecasts for project completion times, as the project proceeds. When the project is at the initial phase, typically the forecast is a lot more uncertain and there is much scope for getting the estimation wrong, the better we are at estimating the narrower the cone will be. Therefore for a typical organisation which doesn't have any bias in its estimation technique, the forecast will underestimate time about 50% of the time, leading to (according to the Chaos report) of 50% of the projects failing due to overrun.
We can see that we can only judge overruns in the context of the validity of the original forecast.
The report looked at a number of different organisations and found the following interesting points.
The first organisation they looked at has a very high accuracy of forecast yet only managed (according to the Chaos criteria) to managed a 59% success rate.
The second organisation they looked at admitted they used the Standish criteria to determine if a project was a success, this led them to increase the forecast time for their projects (to achieve a greater success rate), many projects ran within budget and within time, yet the accuracy of the forecasting for this organisation was less than the first organisation, due to the bias introduced by the Chaos reports targets.
The also looked at a third organisation which despite having a good performance in terms of forecasting, (small deviation from the actual), has an institutional bias in producing low forecasts for time. This led to them having a low success rate according to Chaos criteria based on this bias.
So the criticisms are that the raw f/a ratio is a poor judge of project success and organisations following that criteria will often bias their forecasting to achieve apparent project success.

(b) Explain the significance of the EQF (Estimating Quality Factor) as proposed by DeMarco, as it applies to project management.

[5 Marks]

Estimating Quality Factor is a metric which measures how accurate a forecast or project estimation is. It is calculated by dividing the actual value by the deviation from this value for your forecast. So for example, if you estimate a project will take 10 days and it takes 14 days, the deviation is 4 and the EQF will be 10/4 or 2.5. The higher the EQF the better the project estimation is. The issue arises when EQF values are very low, this may cause projects to be highly over resourced or overrun causing planning problems. Monitoring and improving EQF is therefore a useful process for an organisations approach to planning. **[5 Marks]**

(c) Discuss the importance of THREE major factors which contribute to successful management of a software project.

[5 Marks]

Any Three reasonable factors, properly justified.

High levels of customer involvement throughout the project. This allows validation issues to be handled at early stages and reduce late re-working of code.
Close monitoring of the project progress. It is important to identify overruns as soon as possible so as to be able to put in contingency action. For example each week, the actual project progress can be compared with the project baseline to determine if extra resources or time is needed.
Constant project testing and regression testing
There help to determine the current build is stable, they also deal with bugs that creep into established features by the addition of new code. The regression testing ensures the whole of the project is constantly under test, not only new features.
[5 Marks]

B2 **(a)** Discuss the issues of lost update, deadlock and thread starvation in the context of a concurrent system. **[10 Marks]**

Proper description of lost update by multiple threads with simple example.
[4 Marks]
Description of locking and deadlock with example. [4 Marks]
Description of problem of thread starvation with issue of the queuing of threads on monitors in Java as example. [2 Marks]

(b) Describe in detail the use of the Actor model and how it is can be applied to an application running over distributed system architecture.
[20 Marks]

Clear description of actor model, including description of Actors, messaging, mailboxes and clear explanation of how this approach avoids the problems encountered with the shared memory model.
[10 Marks]
Description of types of mobility (weak and string) and how it is applied to actor concurrency. Application of mobility to the implementation of highly scalable architecture. [10 Marks]

Appendix A

```
int i;
int N=5;
int sum = 0;
int product = 1;
for(i = 1; i < N; ++i) {
  sum = sum + i;
  product = product * i;
}
write(sum);
write(product);
```