**SAMPLE QUESTIONS WITH ANSWERS FOR CS2212B FINAL – FOR REVIEW**

**Question 1 [Process Models]**

Consider that we have to implement a completely new information system for which requirements are particularly complex and requires thorough research to fully understand and prescribe functional and non-functional requirements. In addition to this, there is a need to assess at any level the risk of budget overruns and the inability to finalize the system due to technical difficulties and complexity of the system. This calls for several prototypes to be built during the early stages of the project, and also a thorough risk assessment to be conducted at various times (especially at the planning, designing and early implementation phases)

1. Discuss on what is an appropriate model for building this system and why.

2. Draw the organizational structure of the model you have selected as the most appropriate.

***Sample Answers:***

1. An appropriate process model to use is the Spiral model because this process model is suitable for systems for which the requirements need to be extracted from the beginning (it is a completely new system), are complex (that is the system entails risk) and there is a need to have frequent risk assessment stages in the overall analysis design and development phases. The model that allows for such risk assessment is Spiral.
2. Draw the diagram of the Spiral model schematic here.

**Question 2 [Architecture]**

Suppose we would like to implement the run time environment for a new programming language called *Mocha*. Our goal is to design a flexible program execution environment written in this language so that we can run Mocha programs on different computers regardless of their operating system or architecture. The *Mocha* *Runtime Environment* (MRE) will compile the Mocha source code in an intermediate language called *AuLait*. *Aulait* programs will run on a virtual machine - interpreter called the *Cup*. We also want to be able to implement and integrate easily into architecture a compiler that can compile when *AuLait* programs if needed to binary (i.e. machine language) for the more efficient execution of Mocha programs directly on a target computer processor.

1. Which architectural style is the most suitable for this application? Give a list of the main system components and briefly comment on the role of each component.

2. Draw a) the component diagram of the above system and

3. Comment which pattern would be appropriate to design the system so it is flexible enough to be able to replace the interpreter that runs Aulait programs in the Cup virtual machine with a compiler translator that translates AuLait programs into machine language without changing the rest of the design / implementation of the system.

***Sample Answers:***

1. The architectural style which is most appropriate is the layered architectural style and most specifically the Interpreter style. The list of possible components for this system are:

User Interface Module: provided the look and feel of an Integrated Development Environment (e.g. Eclipse).

AuLait Dispatcher: is responsible of accepting AuLait program files in order to parse them, syntactically analyze them, link them, and produce a amalgamated AuLait program that can be executed

Mocha Runtime Environment Engine (MRE-E): is responsible of executing amalgamated AuLait programs. The MRE-E is responsible of using low level OS/Kernel commands in order to allocate/deallocate memory, execute AuLait statements in order, synchronise threads, and facilitate I/O operations

OS/Kernel Services: Provide low level commands and functionality to interface with the hardware and the computer’s native OS system.

1. A sample component diagram is depicted below. An outline of its components and services are:

User Interface Module:

User Interface: allows for Mocha projects to be defined and generally provide a UI for an Integrated Development Environment (IDE) (e.g. Eclipse)

Editor: allows for editing and writing Mocha Programs (can be also part of the IDE)

User Interface Module:

Parser: parses Mocha programs and generates AuLait programs (e.g. bytecode in Java)

Linker: takes many AuLait programs and “links” them into one unit that can be executed (interpreted or compiled to native binary code – see part 3)

Interpreter: takes linked AuLait programs and interprets then using the MRE-E services below

Compiler: in case AuLait programs need be compiled to binary code for faster execution the Compiler module compiles linked AuLait programs directly to binary code for the computer’s native processor

Dispatcher: Decides based on user commands to either use the MRE-E or use the OS/Kernel Module to dispatch a compiled program instead

MRE-E Module:

Command Execution Module: Creates a stack of AuLait statements and executes them in order. It uses the OS/Kernel module services below

Memory Management and I/O: Allocates and de-allocates memory using the OS/Kernel module services below

Thread Synchronization Module: Synchronizes threads and applies locks using the OS/Kernel module services below

MRE-E Module: It provides an interface to the low level OS services offered by the computer’s native Operating System. It provides an abstraction so that different OSs’ can be used in this archiutecture.



1. The most appropriate design pattern to use is the Bridge design pattern as it allowsto decouple an abstraction (the execution AuLait program in this case) from its implementation (whether it will be interpreted or compiled) so that the two can vary independently. You could also show here an example of how the bridge design pattern can be used to change the implementation between an Interpreter and a Compiler.

**Question 3 [Architecture]**

Let us consider a corporate warehouse management application consisting of a database that stores information about stock availability of products in the warehouse, application programs that monitor from different terminals the volume of remaining stock of different items, application programs that change inventory data as they decrease (with sales) or increase (with new arrivals), and application programs that send vendors notifications of new orders as soon as the available inventory of a product falls below a threshold.

Using the appropriate architectural styles, design the architecture of such a system. Provide your architecture in the form of a Component diagram and explain its style and your design choices. Make sure you include the services offered by each component and an explanation of each such service.

***Sample Answer***:

The application calls for a Model View Controller (MVC) architecture style combined with a tiered architecture style. More specifically the top level architecture is a three-tier architecture. The first tier is the client tier consisting of services offered by the terminals and UIs interfacing with external entities (i.e. users, or other third party applications). The second (middle) tier implements the business logic as this is specified in the question (event processing, dispatching of requests etc.). The third tier is the back end data base tier that hosts services for data storage and persistence.

The middle tier is the one which is the focal point of this question and conforms to an MVC style.

A sample component diagram for the middle tier is given below:

Client Tier**:** contains the terminal services and the modules responsible for the generation of system events (e.g. that the stock for an item is below a certain threshold, or a shipment has arrived). It also contains the UI modules to interface with users or other systems

Middle Tier: contains a Model component, a Viewer component, and a Controller component, collectively implementing the MVC style. The Model Component has a Data Model Module which provide abstractions of the items in the data base (Tier 3), and a Notification Module to notify the Viewers when the Data Model changes. The Controller component gets events form event channels and process them through the Event Handler Module. The Model Change Initiator Module initiates changes to the Data Model through the Change Initiation Service. The Viewer Module contains components that appropriately react to changes in Data Model. For example, a “data change bringing a stock below a threshold” will trigger a “procurement and ordering service (to be sent to suppliers). Also payment services are offered so that payments can be made. New viewer components can be added to the Viewer Module and subscribe to the Notification Module.

Back-end Tier: It interfaces with the underlying data bases so that the application becomes independent of the DBMS technology used (Oracle, DB2, Postgress, MongoDB), and also with the Data Model Module at the Middle tier, offering the necessary information so that changes in the underlying data (e.g. stock values) can be appropriately abstracted and represented.



**Tier 3 – Back-end Tier**

**Tier 2 – Middle Tier**

**Tier 1 – Client Tier**

**Question 4 [Specification and UML]**

Let us consider a central library application with the following operations:

**Borrower Operations**:

* User can enter the system via a secure server his/her credentials (e.g. library card code, PIN, or user-name and password)
* User can retrieve information about books through an inventory server and the corresponding database. Information can be retrieved done by providing the title of the book, or its ISBN number
* User can reserve a book for a certain number of hours using the library card code so that another user can not borrow the same book. The book will be available for borrowing once returned from its current borrower
* User can borrow a book that is available for a max period of 2 months.
* User can borrow and check out a book that is specified as designated textbook for a course and for a specific limited time of four hours.
* User can return a book he/she has borrowed

**System Operations**:

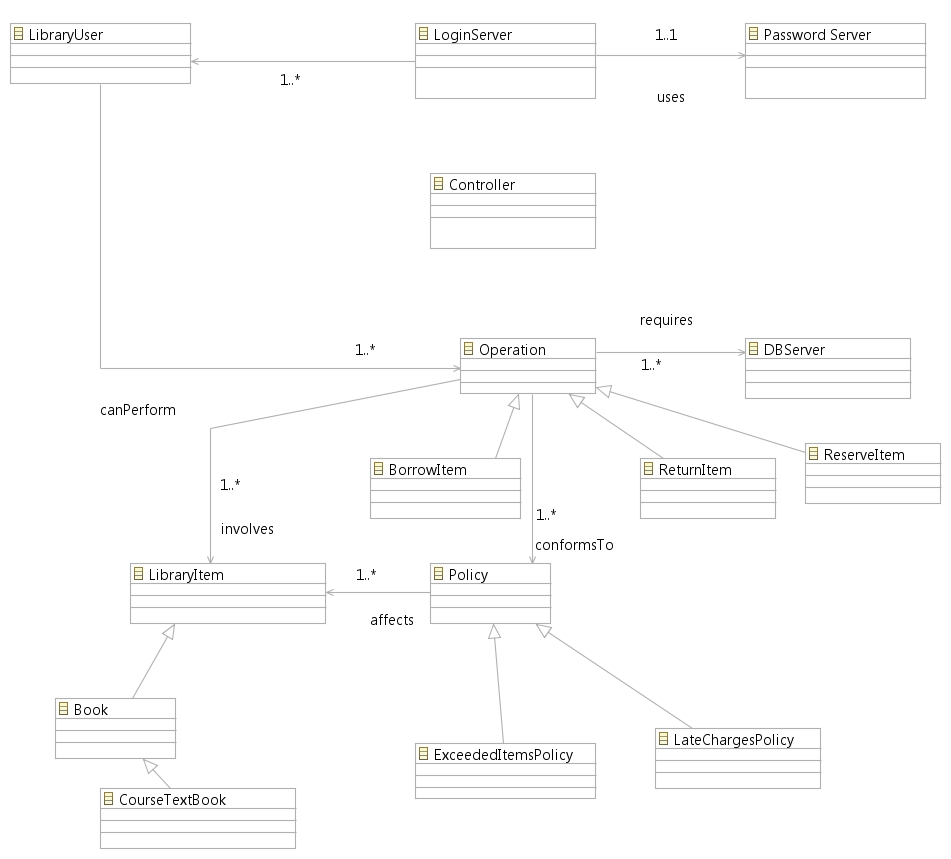
* If the user is late on more than 4 pending books to return, then the system does not allow lending to this user.
* Upon a user returning a book, the system checks for possible late return fines, charges the account of the user with the fine, and then releases the subscription from the user and provides the book for future lending to other users.
* The system notifies the user if he / she has exceeded the borrowing limit.

1. Design the domain model related to the system and scenario described above.

2. Design a sequence diagrams for the system operations.

***Sample Answer***:

1. The domain model is:



2. The Sequence diagrams are:

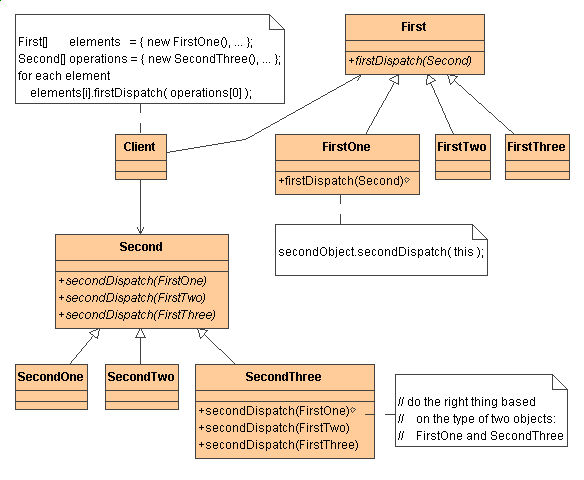


**Question 5 [Design Patterns]**

Consider the following class diagram:

1. Which design pattern is presented in this class diagram? Explain your choice [5 points]

2. Give a short example of code for this pattern using the classes given. [15 units]



**Note:** The purpose of this pattern is to model a function that should be applied to all elements of a structure as it is traversed. The pattern allows us to change this function with another without having to change the classes - objects of structure.

1. This is the Visitor Design Pattern. The reason is that we observe that there is a container (elements) composed of objects of type *First* (or objects of subclasses of First) and a set of operations of type *Second*. The “accept” method which in this case is the firstDispatch(operation) method, calls the secondDispatch (i.e. the visitor). The secondDispatch methods on the appropriate operation object are overloaded based on the type of the element First that dispatches them.

2.

Here I provide a sample of methods

class SecondOne public Second {

public void secondDispatch(FirstOne fo) {

print(“+++ sd in so dispatched by fo”);

}

public void secondDispatch(FirstTwo ft) {

print(“+++ sd in so dispatched by ft”);

}

class SecondTwo public Second {

public void secondDispatch(FirstOne fo) {

print(“+++ sd in st dispatched by fo”);

}

public void secondDispatch(FirstTwo ft) {

print(“+++ sd in st dispatched by ft”);

}

Client code

FirstOne objfo = new FirstOne(); // Create element

FirstTwo objft = new FirstTwo(); // Create element

SecondOne objso = new SecondOne(); // Create operation

SecondTwo objst = new SecondTwo(); // Create operation

Then the call

objfo.firstDispatch(objso); // will print “+++ sd in so dispatched by fo”

and the call

objfo.firstDispatch(objst); // will print “+++ sd in st dispatched by fo”

and the call

objft.firstDispatch(objst); // will print “+++ sd in st dispatched by ft”

**Question 6 [Reliability]**

We consider the total number of faults μ as a function of time (CPU usage hours) for a software system is given in the diagram below. We would like to give the system for use after the testing phase, when the failure intensity λ is equal to or less than the value of 0.005 failures / CPU hour. From an earlier experience of such systems, we assume that the total number of failures that can be observed in an infinite system operating time is 300. Using the Musa linear model, calculate the time in CPU hours that the system will need to be tested so that the system can available for use with the failure intensity sought. The graph below provides the values for μ(τ). For example on τ = 600 the value of μ(600) = 260.

***Answer***:

You use the diagram. You pick a point (say 600, 260). You use equation 1 given below and you solve for λ0. You know already that the total expected number of bugs v0 which produce failures is equal to 300, and you also know form the diagram that μ(600) = 260, (for τ = 600). More specifically, μ(600) = 260 =

300(1 – exp(-λ0\*600/300) ⬄ 260 = 300 – 300 exp(-2λ0) ⬄ 260 -300 = 300 exp(-2λ0) ⬄ -40/300 = exp(-2λ0) ⬄ -2λ0 = LN(-1.3) ⬄ -2λ0 = -2.04 that is λ0 is approx. equal to 1.

Once now you have λ0 = 1 you use the second equation and you find for which time τ, λ(τ) = 0.005.

More specifically, 0.005 = 1\*exp(-τ/300) ⬄ -τ/300 = LN(0.005) ⬄ -τ = -5.3 \* 300, that is τ is approx. 1590 CPU hours.

**NOTE:**

**V0** stands for the total expected number of failures over the system’s operational life (it is equal to the expected total number of bugs that can produce a failure). We can not know a-priori this number, we can only estimate it by looking at past similar projects or by error seeding. This number will be given or easily estimated by the data of the question.

**λ0** stands for failure intensity (i.e. number of failures observed per CPU hour). This number should be reduced over time as we find and correct bugs through testing. Ideally would drop to 0.

**μ(τ)** stands for the total number of failures observed (and corrected) up to a point of time τ.

**λ(τ)** stands for the failure intensity measured at time τ.

**λ(μ)** stands for the failure intensity measured as a function of the total number of failures observed

**Useful Equations**

**Equation 1.** μ(τ) = v0 (1 – exp(- λ0τ/v0))

**Equation 2.** λ(τ) = λ0 exp(-λ0τ/v0)

**Useful Tables**

