UNIVERSITY OF BOLTON

CREATIVE TECHNOLOGIES

BSc(HONS) COMPUTING

SEMESTER 2 EXAMINATION 2016/2017

SYSTEMS ANALYSIS AND DESIGN

MODULE NO: CPU5006

Date: Tuesday 16th May **Time:** TBA

INSTRUCTIONS TO CANDIDATES:

Answer 5 questions.

Marks for parts of questions are shown in bold.

Unless otherwise stated all symbols take their usual meaning.

Electronic calculators may be used provided that data and program storage memory is cleared prior to the examination.

Q1) Software Development Life Cycles

1. Describe the four core phases of the System Development Life Cycle and their phase deliverables **(8 Marks)**

Investigation: The existing system is evaluated in order to determine any deficiencies. This can be done via interviews, observations, inspection of records and questionnaires. (1) A feasibility report is usually produced in order to determine the likelihood of being able to create a new system. (1)
Design: This stage includes the design specification for new software. It includes design of input methods, processes and output methods. It also addresses any personnel issues such as staff training and re-organisation. (2)
Implementation: This is where the new system is implemented. There are different types of changeover. For example, a 'Direct Changeover' is where the old system is stopped one day and the new system is used the day after. A parallel changeover is where the new system is run alongside the old one for a period of time. A phased conversion is where one module at a time is converted from old to new until the whole system is transferred. (2) **PDP (Phased, Pilot, Direct)**
Maintenance & Evaluation: After the new system has been implemented, it needs to be maintained. For example, 'Perfective Maintenance' is making improvements to the performance of software based on user feedback. 'Corrective Maintenance' is fixing bugs that were not discovered during the initial testing. 'Adaptive Maintenance' is software may need to be changed if the business' needs change. (2) **APC (Adaptive, Perfective, Corrective)**

IDIME (Investigation, Design, Implementation, Maintenance, Evaluation)

2. Identify the circumstances that would cause you to recommend the following software development methodologies.
   a. Waterfall
   b. Scrum
   c. Throwaway prototype **(6 Marks)**

Waterfall: A waterfall methodology would be recommended when the user requirements are clear. This is because once the process has started, if any major changes need to be made, the whole process needs to be started again. (1) It would also be recommended if there is not a short time schedule as the process is lengthy. This means it is also good for complex systems. (1)

Scrum: Scrum is an agile methodology that runs all stages of the system development concurrently in short bursts. This allows for changes to made and does not need clear requirements from the outset as it is flexible. (2)

Throwaway Prototype: Throwaway prototyping allows a system to produce a prototype, test the implementation and decided whether or not to implement it into the final system. If not, they can keep revising the system and producing prototypes until the desired system is produced. (1) This is recommended for a situation where the user requirement are unclear as modifications can be made at the prototyping stage and then tested. (1)

3. Identify how the following artefacts are used in software development
   a. Work Breakdown Structure
   b. Gantt Chart
   c. Network diagram ***(L2 – S10-12)*** **(6 Marks)**

Work breakdown structures are a list of all the tasks broken down into sub-sections. They include a description of the task, the duration in which to complete it and any dependencies they have upon other tasks. Each task is numbered and has a status indicating whether it is Open, In Progress or Complete. (2)

Gantt Charts are similar to work breakdown structures, but they visualise the tasks and their duration and dependencies using horizontal bars and vertical arrows along a time axis. They list the tasks name, the duration and the predicted start and finish date. These charts produce an easy means of quickly seeing what tasks should have been done, are currently in progress and must be done, along with the people involved. (2)

Network diagrams visually display all the tasks in boxes which show the task names and duration. The flow and dependencies is shown by connecting the boxes together with arrows. (1) These diagrams are useful in a situation where you need to quickly see which activities are critical, carry the most dependencies and would require the most materials and personnel. (1)

Q2) System Representation

1. Identify the **five** main elements in an **activity diagram** and state their **purpose.** **(10 Marks)** – Reference Here

Activity: An activity is a sequence of behaviour. It is represented by a rounded rectangle and encloses all actions, control flows and other elements that make up the activity. (2)
Action: An action represents a single step within an activity, it is denoted a smaller rounded rectangle. (2)
Control Flow: A control flow shows the order of flow in the activity by connecting two actions together. It is represented by a line with an arrowhead. (2)
Initial Node: An initial node is a solid black circle that is connected to the first action in the activity via a control flow arrow. It depicts the first action in the sequence of activities. (2)
Final Node: There are two types of final nodes. The first is an activity final node which is depicted by a hollow circle with a smaller black circle in the center. This node represents the end of all controls flows within the activity. However, the flow final node, depicted by a hollow circle with a cross inside, represents the end of a single control flow. (2)

CAFAI (Control Flow, Activity, Final Node, Action, Initial Node)

2. Identify three different representations for the functional model of a system and describe a set of rules that ensure consistency between the three models.

(7 Marks)

Use Case Diagram: A use-case diagram is a graphical representation of all the major processes, interactions and associated actors within a chosen sub-system. (1)
Activity Diagram: An activity diagram is a graphical representation of the workflow between activities of a system. (1)
Communication Diagram: A communication diagram is a UML interaction diagram that shows the interaction between objects. (1)

It is important to ensure that consistency is maintained throughout the creation of all the diagrams. This means that all actors, actions, associations, dependencies and boundaries are consistent across all representations. All actions and processes should be included and there should not be any extra ones that weren't previously used. (4)

UAC (Use-Case, Activity, Communication) – User Account Control

3. Compare the types of **event flow** within a **use-case**. **(3 Marks)**

The three types of event flow are basic, alternative and sub-flow. The basic flow of events covers what should 'normally' happen when the use case is performed. (1) The alternative flow of events cover optional and exception behaviour in relation to the normal behaviour. (1) The alternative flow of events can branch off and return to the normal flow or branch off and end the execution of the use case. (1) A sub-flow is a separate path that branches off from the normal flow and loops back round to the position it branched off from. (1)

Q3) Application Architectures

1. Identify the advantages and disadvantages of the three principal application architectures in use today. **(9 Marks)**

2. Describe three major factors in selecting hardware and software. **(6 Marks)**

3. Compare and contrast cloud computing and ubiquitous computing **(5 Marks)**

Q4) Verification and Validation

1. Describe the steps required to evolve an analysis model into a design model. **(5 Marks)**

2. Discuss the relevance of UML package diagrams in the design of a software system. ***(L7 – S13)*** **(3 Marks)**

3. Compare and contrast custom development, packaged software and outsourcing as viable software acquisition strategies. ***(L7 – S17)*** **(12 Marks)**

Q5) Class and Method design

1. Describe the four main characteristics of object oriented design**.** **(4 Marks)**

Classes, Objects, Methods & Messages: Classes are like blueprints for objects. They contain fields and methods and can be instantiated to create objects. These objects pass messages between one another in order to communicate. (1)

Polymorphism & Dynamic Binding: Polymorphism is the ability of an object to take on many forms. For example, if you had a Student class that inherits from a Person

class, you could say that the Person is polymorphic as it can take on many forms such as Teacher, Nurse, Firefighter etc. (1)

Encapsulation & Information Hiding: This involves setting visibility modifiers for fields inside classes. Private fields are not directly accessible by other classes and have to be accessed via public methods. (1)

Inheritance: Inheritance is when one class inherits properties, such as fields and methods, from another class. Some OOP languages use single inheritance where one class can only inherit from one other. They can however be stacked in a hierarchy. (1)

2. **Compare and contrast** the class design concepts of **coupling**, **cohesion** and **connascence**. ***(L8 – S6-9)*** – Reference Here **(6 Marks)**

Cohesion refers the relationships inside a module, whereas coupling is the indication of relationship between modules. (2) When designing classes, you should aim for high cohesion, while coupling should be kept to a minimum. (1)

Connascence is ensuring consistency across the whole program. For example, ensuring that a database connection is made in the same way every time. (1) It doesn't matter whether the system is cohesive or coupled properly, as long as the methods used are consistent, meaning it has good connascence. (1)

3. Identify the major stages of the object design process **(6 Marks)**

Adding Specifications: One major stage of the object design process is adding specifications. This means setting the visibility modifiers for all attributes to ensure security is as tight as possible. (1) Also, adding constraints where necessary to ensure the validity of any information input. For example, setter methods having specific restrictions to the range of values and data types are correct. (1)

Restructuring: Another major design stage is restructuring. This involves things like re-factoring code as much as possible in order to refine it. This means that the code is minimised and is as efficient as possible such that the system load is kept to a minimum. (1) It also involves object normalisation which is used to increase the amount of cohesion and reduce the amount of coupling between classes. (1)

Optimising: A third object design stage is optimisation. This involves reviewing access paths and attributes in order to minimise the amount of computation involved.

(1) You can also review the execution order of the statements in often-used methods in order to achieve the same result. (1)

OAR (Optimising, Adding Specifications, Restructuring)

4. Discuss how class and method designs can be validated. **(4 Marks)**

Class and methods designs can be validated by comparing them to the functional representation diagrams. I.e. Use Case, Communication, Sequence, Activity. (1) Classes should match up with the actors used in the diagrams (1), methods should match with the actions (1) and messages should match the associations. Any dependencies should also be the same. (1)

Q6) Data Management

1. Discuss the major approaches to **object persistence** within **data storage**.
(L9 – S3-4) **(10 Marks)**

Sequential and Random Access Files: One major strength of sequential and RAM files is that they are usually part of an object-orientated language and can be designed for fast performance. (1) They are also good for short-term data storage. (1) However, one major weakness is that redundant data must be updated using programs, instead of using a query language. (1)
Relational Database Management System (RDBMS): Another approach is the use of an RDBMS which is currently the leader in the database market as it can handle diverse data needs. (1) However, a weakness is that it cannot handle complex data and has no support for object orientation. (1)
Object Relational Database Management System (ORDBMS): Another approach is the use of an ORDBMS. One major advantage of this method is that it is able to handle both simple and complex data. (1) However, one major dis-advantage is that has limited support for object-orientation. (1)
NoSQL Data Store: The NoSQL approach on the other hand has the ability to handle complex data. (1) However, this technology is still maturing and it is hard to find the skills to work with it. (1)

It is important to consider that depending on the approach you choose, the underlying hardware is still required in order to support that method. Scalability should also be taken into account from the outset to ensure the system is futureproofed. (2)

2. Describe strategies for **efficient data storage *(L9 – S9)*** **(5 Marks)**

One strategy for optimising data storage is by reducing redundancy. Data redundancy refers to the unnecessary duplication of data, so by reducing it, efficiency is increased. (2) Similarly, eliminating null values also increases efficiency as the value still has a size and takes up space on the disk. (1) Another strategy is normalisation of the data. Normalisation arranges that data into logical groupings, eliminating data redundancy and ultimately decreasing the amount of data stored. (2)

3. Identify appropriate strategies to **optimise data access speed** **(5 Marks)** ***(L9 – S9)***

One strategy that can be used to optimise data access speed is de-normalisation. By de-normalising a database it allows for quicker query response times and more efficient index usage, resulting in faster data-access speed. (2) Another strategy is to use hardware acceleration and caching. Some GPU architectures such as NVidia's CuDA technology allows GPU Hardware Acceleration. This utilises the GPU's cores as well as the CPU's in order to execute the data retrieval algorithms. Caching the most frequently used data will allow fast access speed as cache has very high read and write speeds compared to standard flash memory and hard disks. (2) Clustering is another strategy that can increase performance in database access speeds. A single cluster key is created that represents that value in every row that uses. (1)

Q7) Fundamental HCI Design principles

1. **Discuss 5** design principles for **User Interface design** **(10 Marks)**

Layout: The layout of the user interface should be separated into distinctly different areas with different purposes. For example, navigation at the top, middle for information input/output and bottom for status information. (2)
Aesthetics: The interface should be functional and inviting. It should have a good balance between colour and design and white space so it looks pleasing but leaves enough space for the important information and content. (2)
Consistency: User Interface design across a program or a suite of programs should be kept consistent. This means that users should be able to predict what happens before they perform a function. It aids ease of use and learning. (2)
Minimal User Effort: The interface should be simple to use making user interaction as easy as possible. It is common practice to ensure that no more than three mouse clicks from the starting menu until the user performs work. (2)

Content Awareness: Users should always be aware of where they are in the system and what information is being displayed. If the user is confused about where they are and how to get back, then changed should be made to the interface to make it clearer what is happening at any given time. (2)

Consistency
Content Awareness
Layout
Aesthetics
Minimal User Effort

CCLAM

2. Identify the **basic design principles** for
   a. Navigation
   b. Input
   c. Output – Reference **(6 Marks)**

One type of navigation control is menus. There are lots of different types of menus such as drop-down, pop-up, menu-bar etc. Menus are designed to quickly provide the user with several options which can navigate the website or program. (1) It is also important to consider language. Ensure that names and descriptions are short and to the point. I.e Home, About, News, Gallery etc… (1) Direct Manipulation can be mentioned for another mark if needs be (Quick access in minimal clicks). (1)

Data input should be designed in such a way that it minimises the number of keystrokes the user must perform, assuming the data capture in question is via a keyboard. (1) Only necessary, variable data should be captured. For example, there's no need to store date of birth and age, as the age can be calculated from the date of birth. (1)

One basic design principle for data output is report usage. A report is used to present information and can be historical in nature. They document the processing of data and serve as an audit trail. (1) It is also important to minimise bias by carefully choosing the way data output is sorted and setting data limits responsibly. (1)

3. **Describe** how **international** and **cultural** issues **influence User Interface design** - Reference **(4 Marks)**

One international issue that can influence user interface design is when considering multi-lingual requirements. For example, if a piece of software is intended to be used internationally, a locale file is required to ensure the software has all intended languages available. (1) It is also important to keep language short and simple while avoiding humour and jargon such to avoid confusion. (1) One cultural issue regarding the user interface is the use of certain colour schemes as they have different meaning in different cultures. For example, the colour red means “Death” in Egypt, but “Happiness” in China. (1) Another cultural issue is masculinity vs femininity. It is important to consider the distribution of emotional roles between genders as masculine cultures are competitive and assertive while feminine cultures place more value on relationships and the quality of life. (1)

Q8) Design Evolution

1. Discuss the how the following techniques are used in the transitioning of a system design into an appropriate solution.

   a. Factoring

   b. Partitioning

   c. Layers **(10 Marks)**

2. Identify the strengths and weaknesses of the three major design strategies to provide system solutions. **(10 Marks)**

END OF QUESTIONS