# Timed Assessment

|  |  |  |  |
| --- | --- | --- | --- |
| **Title** | **Models and Software Metrics (Timed Online Test)** | **% of final grade** | 50% |
| **Format** | *This paper consists of* ***SIX*** *questions you are required to answer* ***TWO*** *questions from Section A and* ***TWO*** *questions from Section B* ***ONLY****.*  ***Rename*** *the test paper with your* ***student id*** *and include this at the bottom of each page along with the page number.*    *Enter your answer using* ***your own word****s directly under each of the questions/sub questions in the space provided.* ***Do not include references, bullet points, diagrams, images, or graphics****.*  *Submit this* ***ONE*** *Word file via Canvas.* | **Duration** | The duration of this test is 150 minutes as follows:   * 120 minutes for answering questions * 30 minutes for preparing and uploading your submission |
| The **submission deadline** is:  (see **Due** time on Studynet/Canvas) | 14th January  15:00 hours | This assignment will be available until:  (see **Available Until** time on Studynet/Canvas) | 15:00 hours |
| **This timed assignment assesses the following module learning outcomes:** | | | |
| 1. the range of individual products and processes that exist within a complex software engineering project. 2. the selection of appropriate models and measures that can be applied to software engineering products and processes. 3. measurement based techniques to typical software engineering problems. | | | |
| **Additional information for students** | | | |
| **No late work is acceptable.**  If you miss the deadline:   * You may be eligible to apply for Serious Adverse Circumstances: <https://ask.herts.ac.uk/serious-adverse-circumstances-sac> **or** * UPR AS14 D52.2.2 and D5.2.2.3 will apply (“*Where a module numeric grade of 19 or less has been achieved through unintended non-submission of coursework or non-attendance at an examination or in-class test, Module Boards and Short Course Boards have the discretion to award a FREFE/FREFC/FREFB status code*”).   This is an individual assignment. Regulations governing academic integrity and academic misconduct apply, see: <https://www.herts.ac.uk/__data/assets/pdf_file/0007/237625/AS14-Apx3-Academic-Misconduct.pdf>  For postgraduate modules, a score of 50% or above represents a pass mark. | | | |

**Section A Process Models and Quality**

1. **Software Process [25 marks]**

***Case Study***

You are a software project manager in a large motor car manufacturing company. The company’s software department is about to start developing a new highly sophisticated and leading edge on-board navigation system for the next generation of the company’s cars. The new navigation system will provide significantly enhanced features and functions compared to the previous version that is currently used in the company’s vehicles

You will be responsible for overseeing a development team of 6 staff. Rapid development of the software is required so that the product can be released quickly to gain competitive advantage, but high quality in the delivered software is also vital since product recalls after the cars have been launched are expensive and embarrassing. Six months ago, the software department gained CMMi level 4 having been a level 3 company for the previous 2 years. However, the procedures that the department has used to date have proved cumbersome for the rapid development needed for this project and so the organisation has decided to review their development environment.

The software development department has until now used a paper-based top-down conventional approach to the specification and design of software. It also used an unsophisticated software development support environment that was developed in-house. This environment has proved to be very unpopular with developers who do not find it either easy to use or useful. Senior management has therefore decided to investigate the use of a new set of software development tools for the design, coding and testing of the system to be developed in this development project. The department hopes that the new toolset will address the persistent software problems they have been experiencing despite the high rating of their software development process. These include: significant schedule slippage (late delivery of software releases), product quality problems and problems integrating the software with the hardware of the motor car.

1. Referring to the case study above *in no more than 600 words* describe why you would recommend the Rapid Application process model for adoption by the software development department for the new development. Give the general characteristics and features of this process model, the stages/phases and any other differentiating characteristics of this model. State any assumptions made [17 marks]

|  |
| --- |
|  |

1. Briefly *(250 words)* outline one different process model from the one suggested in 1a) above that you consider **not** to be suitable. Explain why this approach would not be recommended.

[8 marks]

|  |
| --- |
|  |

1. **Agile and Prototyping [25 marks]**

1. Explain what you understand by the term "software process model". State how the choice of process model affects the management of a project. In no more than *250 words* describe the Evolutionary Development process model and the circumstances in which it would be appropriate to use it. Comment on the manageability of projects employing this process model.

[15 marks]

|  |
| --- |
| The process of designing, specifying and implementing the set of coherent activities is called as software process model. In addition to that it is also stated as the demonstrating a specific process in a particular perspective. Description of a process is done in a way by overlooking based on specification, Designing and implementation, validation and evolution.  Specification is purely based on the service required for the system. It is important that one should manage the requirement engineering techniques. Requirement engineering techniques are as follows: Requirement elicitation, Requirement specification, Requirement validation.  **Requirement elicitation:**  It is that one or more prototype is developed by observing the available system.  **Requirement Specification:**  At the same time the information from the requirement analysis is gathered and this information is provided to the end user as abstract statements. The detailed description and functionality are provided in addition.  **Requirement validation:**  Validation helps in checking the realism, consistency and completeness.  Design and implementation help in designing and implementing the system so that the executable system is provided to the customer. This design process includes architectural design, database design, interface design and component design.  **Software validation:**  The developed software is now validated and checks if the requirements of the customer are met. This process takes place first by meeting the exceptions given by the customer and then validation and inspection takes place. The process of validation undergoes component testing, system testing and customer testing.  The choice of choosing the process model is so important that it has good and adverse effects on specific projects. It is essential to choose the software process model according to requirement by considering the cost and time constraints. The basic requirement involve selection on the basis of project type, basis of requirements of the project and finally on the basis of user.  **Evolutionary process model:**  This process model is considered as the combination of iterative and incremental model where time bounded iterations are needed for the development. So as to support the iterations this process must allow small increments. On the other hand, the design abstraction gives the specification for the technical review which in turn inspects the defects. These defects are corrected and later as the time allows modelling, designing and testing can be improved up to the deadline.  This model is uses in large projects where the model is designed on the basis of steps and user can start using it even before the complete software is released. This model is mostly used in object-oriented software development because in this case the software will be divided.  In this model all the process are done in development phase where complete software is divided into chunks one after the completion of a one stage it goes down for the customer feedback and moves to next stage at this time the change in requirements is made. |

1. Referring to the case study, in no more than *250 words* discuss the issues that need to be addressed when deciding which prototyping method to adopt.

[10 marks]

|  |
| --- |
| As the case study shows that new navigation system should to installed so as to provide the enhanced version of the software. At the same time the cost of the software is also getting raised so that the user will not get satisfied. Moreover, there will be some specification that will not be useful for the some specific customers. Thus results in organization’s failure. To overcome this situation prototyping methods has to used which at the beginning demonstrates the complete process to the customers which helps to identify a problem and its respective solution.  Prototypes tells us how the system supports their work when releasing it creates errors and omissions so that the specifications can be modified. For the prototype to be successful it has to be explicit. On the other hand, the prototype must not be unstated because if left unstated it leads to misunderstanding.  For the prototype to be successful the it must be quick and cheap to check the requirements.  Prototyping goes through Developing, designing, revising, usability test and evaluation.  To resolve the issue in the case study and to satisfy the process of prototyping the organization must first get involved in low fidelity prototyping and paper prototyping. Where in low fidelity prototyping the software only passes the resemblance of final product with limited functionality and it is cheap which additionally involves many users and flexible to use.  Later after passing this section the wizard oz prototyping has to be followed where the users are unaware that they are being observed as they use wit high fidelity. Here the designers are involved with users |

1. **Software Quality [25 marks]**
2. Referring to the case study discuss ***three***attributes of software product quality that are relevant to the software development process in 1a. State why each aspect is important.

[10 marks]

|  |
| --- |
|  |

1. Referring to the case study, in no more than *250 words* discuss how badly developed software might affect the work performance of the employees and productivity at this large motor car manufacturing company?

[15 marks]

|  |
| --- |
|  |

**Section B: Software Measurement and Metrics**

1. *The Goal-Question-Metric (GQM) model of quality and* Cyclomatic complexity metric.

**[25 marks]**

1. An organization goal is: *to reduce the time to effect changes to program code.*

Identify four suitable Questions and associated Metrics. For each question, justify each Question and Metric.

[16 marks]

|  |
| --- |
| Fully maintainable software is that in which the software codes are developed based only on the current state. If the codes are future oriented, it has a big impact on understanding. So, if you create a code only considering the current requirements, you will save time. Limiting the code and breaking it down into targeted units also saves time. The type of code used to save time must follow the design metrics. The design metrics are calculated only from the requirements before the system is implemented. The type of code also includes size-oriented metrics where LOC is the normalized value because the size-oriented metrics depend on the programming language. If the code is futuristic, the user cannot understand the code, and in this case the size-oriented metrics cannot be used. Moreover, the data structure metric comes into play which divides the amount of data by assigning the number of variables and operands and calculates these variables as the usage of data in the module. On the other hand, the weaknesses of the program are revealed and fixed as they affect the effort and time metrics.  Entity- Size and requirements of the code.  Attribute- Length and age of the code  Measure- Line of code and data structure  What coding language is used with their required feature?  The Object-Oriented programming language is determined by the scope and complexity of the technology required for booking and research system. Typically, complex applications are broken down into smaller components called iterations. Each iteration has different technologies for which different conditions are used, allowing sufficient time. However, at some point, the finished code needs to be handed off to another team. So, we need to make sure that we adopt the code that meets the current requirements. The related metric for selecting a size of programming language with the desired function is the size-oriented metric, which compares programmer productivity. The metric is derived by normalizing the quality and productivity by considering the size of the product as a metric. Following the LOC, the Halstead's Software Metrics, which initialize the program volume where the actual size of the program is measured and the difficulty of the program is known by the number of operators, while a well-structured program is a function that contains unique operators and operands. Overcoming the Halstead's Software Metrics, Cyclomatic Complexity is used to measure the independent paths by making sure every path is tested and risk associated is evaluated which in turn reduces the nested condition which saves time.  Entity- Type of Source Code.  Attribute- Complexity, Scope, Time, Productivity  Measure- Number of operators, operands in Line of Code, reduction of conditions.  How many iterations of code must be developed?  As stated earlier, dividing the application into simple components called iterations allows for easier programming and saves time. On the other hand, creating more iterations leads to using more resources, which is not effective. So, if you create a scrum board for each iteration and follow a backlog diagram, you will save time by knowing which iteration the programmer needs to work on. This process follows component metrics, cohesion metrics, and cohesion metrics in conjunction with information flow metrics. Since each iteration must be connected to the system components, the sum of complexities is considered. The effects of the first iteration are processed by size-oriented metrics, where the number of lines of code (LOC) depends on the programming language. Thus, LOC chooses the number of lines of code in each iteration, while the component metric decomposes a system into iterations and the cohesion metric executes a function in each iteration up to the degree in which elements are related where an elements output is treated as other iteration’s input and at the same time the same input is used for other iterations and coupling represents the connection between one component to another. This metric executes a function in each iteration by passing only the data and allows reuse functionality.  Entity- Application to small tasks and respective links.  Attribute- Sum of Complexities and their Connections.  Measure- Iterations depend on LOC and worked by decomposing and creating link between iteration.  What type of testing must be done to reduce time and defects?  Begin the test with the path of the code. It is known that it is not possible to cover 100% of the code, you need maintenance to correct the code and a reduction of the code is necessary. Testing the behaviour unit and not the implementation unit saves time. Testing better supports the code by telling us how it should behave. Continuous testing needs to be provided for each iteration so that bugs in the iterations can be fixed on the spot by the programmers and developers. From the case presented by the University of Dublin, Function Point Analysis (FPA) must be used to choose the type of testing because FPA estimates the software projects and performs the tests depending on the functional size of the code, which results in the functional size of the software being presented to the client, customer, and stakeholders. At the same time, in project metrics the project manager collects the data from the past projects these data are compared with the original effort. This saves time and cost of development.  Entity- Tester and testing software.  Attribute- Function Point Analysis, Behavioural testing, functional size.  Measure- Comparing with the past project and estimating with the original project.  What are the resources needed for the code to save time?  From the above case, the university needs a software system for booking and research. So, the resources needed are the quality workstation, a quality team and a project manager and owner. With these resources and people who have more experience in programming, the software system needed will be successful and save time. Resources are handled to match both complexity and effort. By considering both effort and complexity, we opt for function point analysis as it analyses the character complexity and represents the manpower requirement. These FPAs are mostly used in information systems to estimate the software project. Team velocity measures the size of the previous projects and size of the team needed to complete the project at a specific time. It helps to understand how much value the team is providing in the given time.  Entity- Team, Workstation.  Attribute- Complexity and manpower requirements.  Measure- Increased resources save time and effort with increased cost. |

1. Using your own words state ***two***claimed uses and ***two*** criticisms for the Cyclomatic complexity metric.

[9 marks]

|  |
| --- |
| **Uses of cyclomatic complexity:**  Helps to determine the linearly independent paths and proven to be helpful for developers and testers. In addition to that It maskes sure at least every path has atleast tested once and which helps to focus on uncovered paths. By using this code coverage is improved. By using it earlier it helps to reduce the risk. It breaks the program into smaller blocks due to decision making constructs. It creates nodes representing the modules associated with it.  **Criticism of cyclomatic complexity:**  When coming to counting of independent paths it creates objection. In addition to that it is not at all clear that the points are clear. No engineering approximation from empirical evidence for large class software usually for lines of code because inmany cases it does not work well. |

**5. Software Metrics [25 marks]**

1. Software Metrics is a standard of measure that contains different activities which involve a degree of measurement.

Discuss ***five*** of these activities and include a brief description of ***five*** characteristics that you think demonstrates a good software metric. Justify your answer.

[15 marks]

|  |
| --- |
| **Activities:**  Cost and effort estimation models and measure  Data Collection  Quality models and measures  Reliability models  Security models  Structural and complexity metrics  Capability maturity assessment  Evaluation of tools and methods  Performance and evaluation models  Productivity measures and models  **Time Trackable and economical:**  When we measure code or software, we know exactly how much time we need to see the control flow in order to track it. It is very cost-effective for us.  **Aligned with the business:**  By choosing the metric that is pertinent to the business we are working on, we can assess the quality and the amount of testing effort we must devote to developing the product.  **Actionable and predictable:**  The fact that metrics have a quantitative nature enables us to analyse them in simple numbers, and it enables us to predict whether any changes are necessary.  **Language Independent:**  We can use the same metric in different languages, if necessary, and it will be independent of the programming language.  Consistent:  It is very consistent in nature since the metric values will always be the same no matter how many times we measure.  **Comparable with external benchmarks:**  Since the metrics are simple to understand, it can be compared with other software applications. |

1. Briefly outline ***two*** metrics which relate to software product quality, discuss why you think each can be used to assist in measuring the quality of a software product

[10 marks]

|  |
| --- |
| Software metrics are divided into:  **Product metrics:**  It is the description of characteristics of size, complexity design features, performance and quality level.  **Process metrics:**  This metrics leads to the development and maintenance and acts effective in removal of defects by predicting it before it’s arrival and correcting it.  **Project metrics:**  This maintains the project characteristics and execution which includes the resources, staffing pattern and life cycle and cost.  We have wide range of metrics like  Internal metrics includes Readability, Reliabilty, Amount of documentation, LOC  External metrics includes Cohesion, Coupling  Usability  Complexity includes Cyclometric complexity, algorithmic complexity.  **Internal metrics:**  LOC: Lines of Code  LOC includes SLOC, KLOC, NLOC, KSDI.  SLOC predicts the amount of effort needed for the development of the program.  Physical SLOC counts the line and the comment lines and it is inefficient.  Logical SLOC measures the line with respect to specific computer language.  KLOC is known as ELOC (Effective lines of code), it includes non-comment OLOC and includes partial lines.  **Reliability:**  With this we can calculate the Mean time to failure (MTTF), It the MTTF is 200 then we can  expect an error in every 200-time units. This can help us how reliable the code that is  developed. The lower the MTTF, the higher the reliability.  **Readability**  Influences how good the code can be read and understood and affects the maintainability and quality if the automated tools are built, they can be integrated.  **Reliability:**  MTTF (Mean time to failure): It is the time interval between two successive failures in large transactions. This can help us how reliable the code that is developed. The lower the MTTF, the higher the reliability.  **Cohesion:**  Cohesion is the degree of relationship between the elements of the modules in the code. If we want to focus on a particular task in the project, we may use High cohesion. We may use cohesion as a measure of the strength of the module.  **Coupling:**  The degree of independence between modules. It can be used to determine the intermediate approach. It indicated the relative independence between modules. Hence, low coupling was always preferred.  **Good software design should have high cohesion and low coupling** |

1. **Reliability [25 marks]**
2. Only one of the two models can have produced the following graph.

Which is it, and why? Which is the best distribution to model the graph?

[3 marks]



|  |
| --- |
|  |

1. Explain the difference between hardware reliability and software reliability. Illustrate your answer by using ***two*** examples that you studied on the module.

[12 marks]

|  |
| --- |
|  |

1. A key feature of Rapid Application Development is code generation and reuse for rapid delivery of software. What product methods can be used to identify faults and failures when developing software in this way to try to ensure reliability? [10 marks]

|  |
| --- |
|  |

**\*\* END OF TEST \*\***