

Assessment Feedback Sheet

|  |  |
| --- | --- |
| **Overall Assessment Grade Achieved** |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Learner Name** | | Husnain Ahmed | | | |
| **Assessor Name** | | C. Livesey | | | |
| **Qualification Title** | | Pearson BTEC Level 3 National Extended Diploma in Computing | | | |
| **Unit/Module No./Title** | | Unit 23: System Methodology | | | |
| **Assignment No./Title** | | **23.1 Principles of Systems Methodology** | | | |
| **Learning Aim(s)** | | A: Investigate the principles of systems methodology and systems techniques used to solve computing problems | | | |
| **Issue Date** |  | **Planned Submission Date** |  | **Actual Submission Date** |  |

|  |  |  |
| --- | --- | --- |
| First/Second Submission | | |
| Criteria | Criteria Achieved | Assessor’s Feedback *Your feedback should include:*   * *What the learner has done well. (Knowledge, skills, etc.)* * *What the learner has not achieved and what was missing.* * *Information or guidance available to the learner they could have drawn on (e.g. class notes; handouts; resources in assignment brief etc.)* |
| P1  Explain the stages of the software development lifecycle. |  |  |
| P2  Explain the principles of systems methodologies used in the problem-solving process |  |  |
| M1  Compare the use of the applied software development life cycle model and systems methodology in the problem-solving process against alternative options. |  |  |
| D1  Evaluate the use of the applied software development life cycle model and systems methodology in the problem-solving process. |  |  |
| **BTEC Rules**  All resubmissions must be authorised by the **Lead Internal Verifier**. Only **one** resubmission is possible per assignment, providing:   * The learner has met initial deadlines set in the assignment, or has met an agreed deadline extension. * The tutor considers that the learner will be able to provide improved evidence without further guidance. * Evidence submitted for assessment has been authenticated and accompanied by a signed and dated declaration of authenticity by the learner.   Any resubmission evidence **must** be submitted within 10 working days of receipt of results of assessment (BTEC only) | | |
| **Wider Skills (Linked to Positive Futures)** *Comment on the quality of the learner work, the learner’s process and practice during assessment, research skills, presentation, general behaviour and conduct, meeting deadlines, etc.* | | |
|  | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Assessor Declaration | *I certify that, to the best of my knowledge, the evidence submitted for this assignment/assessment is the learner’s own. I understand that false declaration is a form of malpractice.* | | |
| Assessor Signature: |  | Date: |  |
| Learner Declaration | *I certify that the evidence submitted for this assignment/assessment is my own. I have clearly referenced any sources used in the work. I understand that false declaration is a form of malpractice.* | | |
| Learner Signature: | H.Ahmed | Date: | 16/01/2020 |

**Learner Actions**

|  |  |
| --- | --- |
| Learner’s Targets/Actions  *What needs to be developed to improve future work?* | *Actions linked to the following Positive Futures outcomes* |
|  | Confidence  Commitment  Collaboration  Resilience |

# Task 1: The Software Development Life Cycle

Analysis:

* Customer will meet with business manager to discuss and provide the requirements for it.
* You can get more requirements for the specification through interviews, questionnaires, observations and through reviewing existing documents.
* The software brief is broken down into segments to make the little jobs easier and more efficient to carry out.
* Each segment is assigned to specialists in their own fields and the methods and requirements they will need are decided.
* They undergo a feasibility study. Operational, financial, legal, technical, political.

Design:

* Software engineers work on designing the software.
* Plan out how the separate segment will look and work.
* Create DFD and other diagrams to help with mapping out the steps that are needed to be taken.
* You need to draw some designs and prototypes for the project.
* You can make different types of diagrams like flowcharts, hierarchy graphs, UML, class diagrams, ERD to see the relationship in-between different entities within the project and IPO tables.

Implementation:

* Choose programming language that will be used to code the program.
* Choose whether you will use procedural programming, event driven programming or object-oriented programming.
* Decide whether your project needs a database.

Testing and installation:

* Choose which method of installation to use Direct (taking out the old and putting in the new), Parallel (using old and new simultaneously until it is judged to be adequate) or pilot (letting a small group of testers evaluate the new system and implement changes based on their feedback).
* Code is tested for errors, stability issues, performance issues, integration and security.
* Black box testing where the tester doesn’t have prior knowledge or access to the code, this looks for incorrect functions, errors in data, external database access and interface errors (making sure that no one can hack into the system).
* White box testing where the tester has prior knowledge about the code and its design/structure (opposite of black box testing).
* User testing is when you get feedback from users of your product and implement ideas based on their feedback and fix bugs or things that they have found to be not to their liking, this is also similar to beta testing which is done in a pre-release by a selected group of testers and uses their feedback to find things to improve on.
* Acceptance testing is when a system is tested for acceptability through meeting a set of criteria.
* Equivalence Partitioning: splits input values into valid and invalid parts and selects some values from each one as test data.
* Boundary Value Analysis:  determines bounds for input values and selecting values that are within the bounds or just inside/outside them as test data.
* Cause-Effect Graphing:  identifies the causes and effects and uses them to generate test cases.

Evaluation and maintenance:

* The evaluation stage looks at the overall project and considers (evaluates) how things went.
* Identify what could have been done better and what should be done in the future.
* Identify which requirements have and haven’t been met.
* Fix issues
* Go through the different types of maintenance: Corrective (fixing errors), adaptive (changing the project to add new features), perfective (implementing new or changed user requirements), and preventative (regular maintenance to prevent any new errors from occurring).

Waterfall – used to create a system with a linear and sequential approach. It is called waterfall because the model develops systematically from one phase to another in a downward fashion and can’t go back like a waterfall.

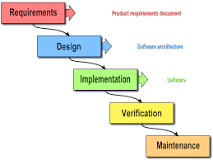
This can be used in a game such as fifa to add new updates as you can go back to fix things but can fix with extra updates

Pros – Before the next phase of development, each phase must be completed

* Suited for smaller projects where requirements are well defined
* Any changes in software is made during the process of the development

Cons – Error can be fixed only during the phase

* It is not desirable for complex project where requirement changes frequently
* Clients valuable feedback cannot be included with ongoing development phase



Spiral – is based on the unique risk patterns of a given project, it guides a team to adopt elements of one or more process models, such as incremental or waterfall and make them its own by using separate parts where they are most suited.

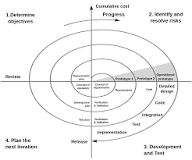
This can be used in a game such as Minecraft to add new updates as you can add more content.

Pros – Continuous or repeated development helps in risk management

* Development is fast and features are added in a systematic way
* Additional functionality or changes can be done at a later stage

Cons – Risk of not meeting the schedule or budget

* For its smooth operation spiral model protocol needs to be followed strictly
* Documentation is more as it has intermediate phases



Prototype –a prototype is built, tested and then reworked as necessary until an acceptable outcome is achieved from which the complete system or product can be developed.

This can be used in a game such as SpongeBob where it all comes out at once.

Pros – Reduced time

* Improved and increased user involvement and feedback
* Reduced costs
* Users can have a better understanding of the system from prototype models

Cons – insufficient analysis

- Excessive development time

* + Users can confuse it for the finished product

C:\Users\B1905311\AppData\Local\Microsoft\Windows\INetCache\Content.MSO\67650A1F.tmp

Incremental - requirements are broken down into multiple standalone modules of the software development cycle and are done in steps from analysis design, implementation, testing/verification, maintenance.

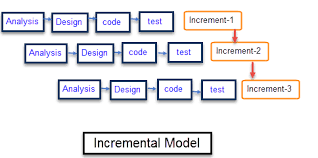
This can be used in a game such as cod where the game is standalone and there is DLC.

Pros – Potential defects are spotted and dealt with early

* Progress is easily measured

Cons – More resources may be required

* Each successive phase is rigid with no overlaps



# Task 2: Principles of Systems Methodology

Yourdon systems method

The method has 2 design phases: analysis and design. It includes 3 steps. The feasibility study (looks at system and environment), essential modelling (describes essence of system and how it must behave) and implementation modelling (incorporates features found in customer statement).

It offers a series of models.

* behavioural model: states that system behaviour can be described in three ways: functions, dynamics and relationships.
* processor environment model (PEM): describes the allocation of computing functions in processor hardware.

* software environment model (SEM): defines the software architecture and its effects from each processor.
* code organizational model (COM): shows the modular structure of each task.
* Pros – creates detailed plans and diagram, very structured and easy to use.
* Cons – not very flexible, time consuming.

structured systems analysis and design method

a set of standards for systems analysis and application design. It uses a formal methodical approach to the analysis and design of information systems.

The techniques used in SSADM are logical data modelling, data flow modelling and entity behaviour modelling.

* Logical Data Modelling: identifying, modelling and documenting data as a part of system requirements gathering. The data is classified further into different entities.
* Data Flow Modelling: involves tracking the flow of data in an information system.
* Entity Behaviour Modelling: identifying / documenting the events influencing each entity and when / how events happen.
* Pros – it had multiple angles of analysis, a very controlled method.
* Cons – Time consuming, rigid control.

Jackson’s System Structured Development

Three basic principles of operation of JSD is that:

* Development - describe and model the real world.
* An adequate model of a time-ordered world must itself be time-ordered. Main aim is to map progress in the real world on progress in the system that models it.
* The way of implementing the system is based on transformation of specification into efficient set of processes. These processes should be designed in such a manner that it would be possible to run them on available software and hardware.

The method originally consisted of six steps: Entity/action step, Initial model step, Interactive function step, Information function step, System timing step, System implementation step.

Later, some steps were combined to create a method with only three steps.

* Modelling stage (analysis): with the entity/action step and entity structures step.
* Network stage (design): with the initial model step, function step, and system timing step.
* Implementation stage (realisation): the implementation step.
* Pros – It is designed to solve real time problem, it considers simultaneous processing and timing.
* Cons – is a complex methodology due to pseudo code representation, it is difficult to understand.

Rapid Applications Development

* Rapid application development emphasizes rapid prototyping and iterative delivery.
* Designers and developers can aggressively utilize knowledge and discoveries found in the development process to shape the design.
* environment, where any necessary full-scale testing or team training can take place.
* Pros - Measurable Progress, Quickly Generate Productive Code.
* Cons - Requires Modular Systems, Difficulty Within Large-Scale Projects.

Scrum Methodology

* used primarily for software development projects with the goal of delivering new software capability every 2-4 weeks.
* Pros – Higher productivity, Better-quality products.
* Cons – short time frames can cause projects to be rushed and not done properly, requires a high level of commitment.

Dynamic Systems Development Model

* The project lifecycle and its overall impact on the company will be considered before it is carried out so that it is going to be worth doing.
* Pros – completed on time, early feedback can be given as the basic functions are made first.
* Cons – requires continuous user involvement, requires high commitment levels to finish project.

Extreme Programming

* Designed to improve the quality of software and its ability to properly adapt to the changing needs of the customer or client.
* Pros – frequent feedback from customer allows requirements to be more easily met and quality of program is better due to more testing and reflections.
* Cons – not a very good structure as changes to be made a lot, unpredictable and undefinable timescale.

# Task 3: The Application of Systems Methodology

Scenario – Re-vamping retro games into modern formats, like board games into computer games. Snakes and ladders, it requires a working game and a GUI. And it should allow 2 players and a winner.

This project is Re-vamping retro games into a video game. The project is targeted at people who enjoy retro horse. The requirements are the game must work and have a GUI, it should allow 2 players and a final winner.

By using the Yourdon’s systems method to make this game you would have to analyse it first to know about it and what you will have to do, then the feasibility study where you will have to decide based on the data whether it is worth it to go through worth making the product. Then you will have to design the models for it based on the info you have for the game or program, in this case I would design the snakes and ladders game to have 100 squares and some snakes and ladders.

Then the behavioural model where it dictates what would happen when you step on the snakes or ladders or the   
end post. It would be structured and easy to use but it would also not be very flexible and would take a long time to complete.

If you instead choose to use the structured systems analysis and design method you would first gather data and plan out your steps to do, then you will break the steps up into separate models and give them to people who specialise on doing them. It is a very structured and clear method with multiple angles of analysis but it has rigid control and is very time consuming.

The scrum methodology would be used by quickly fragmenting what you need to do into steps and complete them in a timeboxed iteration in what is called a “sprint”. It has higher productivity and better-quality products but short time frames can cause projects to be rushed and not done properly and it requires a high level of commitment.

You could use extreme programming where you converse with customers to get lots of feedback to improve your program. You can get frequent feedback from customer allows requirements to be more easily met and the quality of program is better due to more testing and reflections. But it’s not a very good structure as changes to be made a lot, it is unpredictable and has an undefinable timescale.

In conclusion I believe that extreme programming is the best option as you get a lot of feedback and know what your customers want and can cater to them. This way you can ensure that all of the changes are needed and wanted by the target audience and will be accepted by them.