Work Load Participation Index Form

for

Assignments & Project(s)

(To be Submitted/Attached with Every Assignment and Phases of Project)

(Date of Submission): April 2nd 2020

Artifact/Document Type (e.g. Exam/Assignment #1/#2/#3, Project Phase 1/2): Term Test #2
Participation in his/her allocated task's completion

5= Full (as allocated) 4= Partial (slightly less) 3=Half (as allocated) 2/1=Little 0= No Participation

Student Number	*Student Name*	*Participation Index Value e.g. 5/4/3/2/1*	*Signature*
040946430	*Thang Nguyen*	*5*	*Thang Nguyen*
040953846	*Diep Pham*	*5*	*Diep Pham*
040950904	*Mukta Debnath*	*5*	*Mukta Debnath*
040951042	*Ningxin Zhao*	*5*	*Ningxin Zhao*
040930563	*Min Li*	*5*	*Min Li*

Due to the current social distancing problem, we cannot get everyone's signature. All participation proof can be obtained from group leader.

*The **Project Leader** for this Group is: Thang Nguyen*

Signatures of Team/Project Leader: Thang Nguyen
Team Number/Name (if any) _____

***Note:** Team leader please briefly describe below if any group member(s) is/are not participating properly to justify learning and workload participation in the located task(s).*

_____

_____

_____

_____

_____

_____

CST2234 - 303 – Term Test #2

Algonquin College

1. White Box Testing and Black Box Testing are two approaches or techniques involved in testing software being developed.

- Explain how these techniques affect the quality of the software being developed?

a) White Box Testing

Explain:

White box testing is testing that is conducted to test a software's internal **"structure, design, and coding"** [1]. This type of testing focuses on verifying **the flow of inputs and output of the application** as well as the overall technical design such as database design. It is the exact opposite to the black box testing concept.

How it affects the quality of the software:

This type of testing emphasizes on making sure that all the internal engineering process are properly conducted and produce the expected output based on the input. In the software world, if the internal code structure of a product has any signs of bad performance or bad code design, white box testing would expose the signs of defect to the people who develop that software and allow them to rectify those problems as soon as possible while the code is still in the development environment. White Box Testing makes sure that **the inner quality or backend side of the software meets the standard software requirements**.

b) Black Box Testing

Explain:

Black box testing is testing that involves user interaction with the whole application. All the testing is conducted on the external user side without any regards to the inner technical system.

How it affects the quality of the software:

Black box testing exposes how a software application can break from the user's perspective. This is where black box testing differs from white box testing, it does not care about the internal structure or anatomy of a system, or how it was implemented. It only cares if all the desired actions of a user can be performed fluently without any error. Black Box Testing ensures the **software quality from the user's perspective is top-notch and error-free**.

- Identify who would be involved in the implementation of these techniques and why

a) White Box Testing

Who are involved: Developers, System Architects, or engineers who are active in the software development process.

Why: Due to the fact that white box testing is about testing the internal code structure and design, only developers or the people who were active in the development process can comprehend

potential errors with the application code input and output. If non-technical people are involved in the white box testing process, they many not understand the code structure or design which leads to their inability to properly test the internal system. Hence, white box testing is suitable for people with technical skills such as developers or system architects.

b) Black Box Testing

Who are involved: Product Manager, Volunteer Users, Software Testers, CEO, Developers (Rarely)

Why: Since black box testing is all about testing software application on the user end, almost all people from any background can be involved in the black box testing process. Black box testing requires almost zero technical knowledge on how the internal system works, and that is why even a regular user can be involved in this process. In the traditional Software Development Cycle, black box testing is usually carried out by professional software testers who are dedicated to make detailed reports about which test cases does not meet the requirement from the user side. Product Manager or CEO are also people that do not possess any technical knowledge, but they can still participate in the black box testing process due to its non-technical nature.

- **Explain also how these techniques would be implemented within the concept of Validation and Verification?**

Explain concept of Validation and Verification:

Validation and Verification are both procedures that are used together to check whether a software product has met its requirements and specifications or not.

- **Validation** is the concept of assuring the software system meets **the operational needs of the customers or product managers**. Validation usually involves seeking for acceptance requirements from the customer perspective.

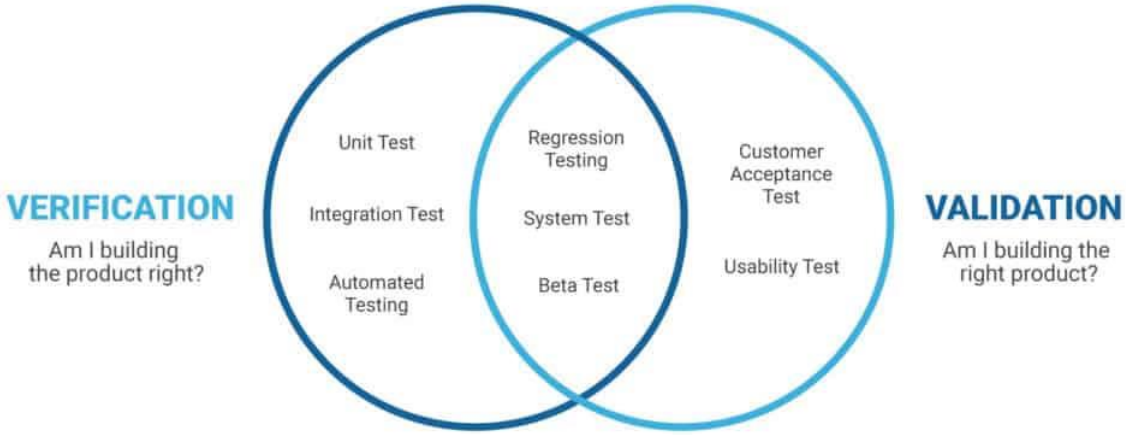
- **Verification** is the concept of evaluation whether the software system **complies with the initial technical regulation or requirements of the software system design**. Verification is usually an internal process within the development team.

How black box testing would be implemented within the concept of Validation:

Since validation is all about making sure all the requirements are met from the customer **external side**, **black box testing** is the most suitable testing methodology for the **verification concept** because it involves testing to make sure that the software system meets all the requirement of actions from the **user perspective**. Black box testing is also sufficient within the validation concept due to the fact that it can make sure that the validation requirements are met by simply testing the product from outside without knowing too much about the internal system.

How white box testing would be implemented within the concept of Verification:

We have all known that verification usually involves the process of checking for initial **technical** regulation and requirements of software system. Due to its **low level and backend nature**, the concept of verification is best associated with the concept of **white box testing** which ensures that the internal code structure and low-level system meets the **technical requirements**. White box testing can be used in the verification process by making sure that it can catches errors that the validation process cannot catch. Verification also incorporates many low-level methods from white box testing such as inspections, reviews, walkthroughs, and desk-checking etc.



Verification vs Validation [2]

2. The Context Diagram is one of the first diagrams to be generated in an SDLC. If the External Entities are not correctly identified when developing the Context Diagram

a) Explain the effect on the 'Event Table' and on the 'Use Case Diagram'

Overview:

Context Diagram: The context diagram is a diagram that "shows the system under consideration as a single high-level process and then shows the relationship that the system has with other external entities (systems, organizational groups, external data stores, etc.)" [3].

Event Table: The event table is a table that models all the events that happens between users and system entities along with expected source and destination of event flow. [4]

Use Case Diagram: Use case diagrams are usually referred to as behavior diagrams used to describe a set of actions that some system or systems should or can perform in collaboration with one or more external users of the system

Explain the effect on the 'Event Table' and on the 'Use Case Diagram': Since event table needs to have the detailed information of the external **entities** that each **user action** needs to interact, incorrect or missing external entities in the Context Diagram can lead to **missing action case** in the table or declaring the wrong information for the **source entity column** and **destination entity column**. Incorrect source entity or destination entity can possibly **create unnecessary or unwanted events** inside the event table. Similar to the event table, the use case diagram also needs all the external entity information correctly to brainstorm all the possible **use cases** inside the system between entities. If any of the external entity is missing in the context diagram, it means that the use case diagram will also **miss one or more possible use cases** due to the missing link required between use cases and entities in the diagram. Incorrect external entity information can also be critical because it may lead the system architect to **have the wrong expected behavior** of an actor, and he or she mays add in use cases that **are not relevant** to the expected requirements of the system.

If the system architect detects the errors in the context diagram, **it takes a lot of effort and time** to change the affected use cases in the use case diagram because of its heavy linked nature between use cases and actors. It will also **be very costly to change the affected event cases inside the event table** due to the changing expected behavior between the source event entity and the destination event entity

b) Explain also how the 'Context Diagram' and 'Event Table' could be tested to ensure accuracy of design

Context diagram can be tested by unifying agreement and ideas between key members of the team who will participate in the development of the project. It would be more beneficial if team members can do cross-checking to make sure all the **required entities** are there inside the context diagram. All the names of the required entities and data flow meanings should be double-checked between every team member to ensure their **clarity in meaning** and their **conciseness in naming convention**. The content of the context diagram also has to pass the rationale test, which means that every dataflow between external entities **have to make sense logically** before moving on to further steps in SDLC.

Event table can also be tested by making good use of **cross-checking** between team members to ensures that **all events are properly listed for every use case**. Team members also have to come up with one **standard format for use case description** to ensure that every event case is readable and comprehensive. All events have to be double-checked to have at least **one source entity, one destination entity, one trigger description, one use case description, and one response.**

Context Diagram and event table can also be further tested by drafting a simple UI/UX design to check for **edge cases** where team members can **find missing or incomplete external entities and events**. It is easier to test the accuracy of the design if the team has a **primitive sketch design** that reaches the consensus of every team member. The design can serve as the hinge point for everyone in the team to **imagine all the possible scenarios that a user may potential do**, and from that imagination, team members can automatically find **the missing external entities or the missing events** that are needed to ensure the accuracy of design in both the **context diagram** and **event table**.

References

[1] “What Is WHITE Box Testing? Techniques, Example, Types & Tools.” *Guru99*, www.guru99.com/white-box-testing.html.

[2] “Verification vs Validation: Do You Know the Difference?” *Plutora*, 7 Apr. 2019, www.plutora.com/blog/verification-vs-validation.

[3] Business Analyst Community & Resources | Modern Analyst. “Interview Questions for Business Analysts and Systems Analysts.” *What Is a Context Diagram and What Are the Benefits of Creating One?*, www.modernanalyst.com/Careers/InterviewQuestions/tabid/128/ID/1433/What-is-a-Context-Diagram-and-what-are-the-benefits-of-creating-one.aspx.

[4] Fakhroutdinov, Kirill. “UML Use Case Diagrams.” *UML Graphical Notation Overview, Examples, and Reference.*, 4 Jan. 2014, www.uml-diagrams.org/use-case-diagrams.html.