**Objectives:**

* Define and understand the software development lifecycle in the context of the 3D TicTacToe programming project (Module B.5).
* Apply the lifecycle to anticipate and plan future activities that will be required to bring the 3D TicTacToe programming project to conclusion.

**Level 1: Defining The Software Development Lifecycle Phases**

The software development life cycle involves the following distinct phases.

1. problem definition
2. analysis & design
3. writing code
4. testing
5. implementation & delivery
6. maintenance

For each of the above phases:

1. Research the definition in the context of software development

problem definition:

**Problem:** The programmer uses research and investigation to define the problem that must be addressed. Defining the problem is the most important part of the software development cycle. If the requirements and the parameters of the problem are clearly understood then the actual output of the development program (process) is more likely to meet the expected output. Defining a problem is to fully understand the issue which needs to be solved.

There can be a lack of Flexibility which is based on the idea that detailed requirements and a plan can be laid out at the beginning of the project without any change as the project is in progress. Which can force the project team to make a lot of assumptions and many times those can be wrong

There can be a lack user feedback and inputs in which is where the user doesn’t see the solution until the very end of the project and by that time, it may be either too late or very expensive to make any corrections.

**Analysis & Design**: Once the requirements are understood, programmers can begin to design the software. Referred to as the architecture which is concerned with making sure the software will meet the requirements provided by your client/leader as well as ensuring that future requirements can be addressed.

**Writing Code:** Development of the programs source code to be to be written. Programmers must follow the coding guidelines defined by their organization and programming tools like compilers, interpreters, debuggers, etc. are used to generate the code.

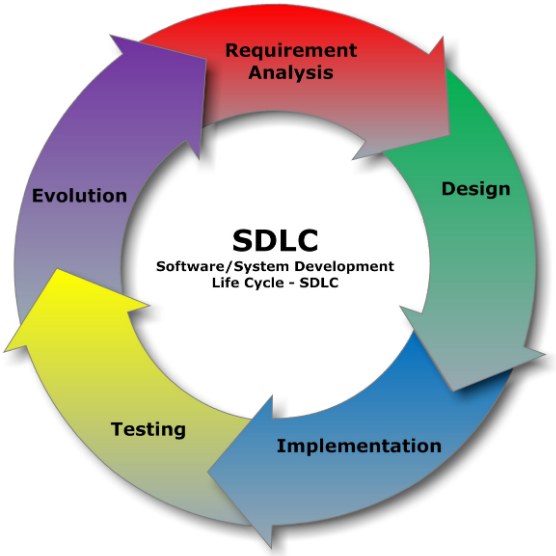
**Testing**: This is the process of finding any defects or bugs in the created program. Testing can also be known as application testing. which refers to the testing only of the program where defects are reported, tracked, fixed and is then retested, until the program reaches the quality standards. This is a stage where the program is almost complete to qualify for the next stage. By which the program goes through testing to discover any errors or incompatibilities that may exist. This may include users who provide feedback when a beta or prototype version is released publicly to the market or privately under business ownership to have users discover any bug or glitches within the software.

**Implementation & Delivery**: The process of getting the software when it is in a official state to the market or your client.

**Maintenance**: To maintain the software and manage it with newly discovered problems or new requirements. These additional modifications are written to be understood and to update applications that were already well-developed. Software must be monitored constantly to ensure proper operation. Bugs and defects discovered in programs are reported and responded to. Which nothing is perfect so bugs may appear after a while from the software release date, so the team can then identify the problem and address it. Some maintenance is extending systems to do new things, which in many ways can be considered new work. While as some can be minor like in simple patches for various bugs/glitches found in video games during a new release.

1. Explain how it applies (or will apply) to the 3D TicTacToe project

The Software Development Life Cycle steps are in relation to our 3D Tic-Tac Toe project because Mr. Nestor is basically our client who provided us with the general concept of this program by giving a brief of the requirements and expectations for when it is officially completed. After knowing what the problem is we would then get into the stage of analysis and design where we split each other up giving each 2 out of 6 group members a component to develop and work on. As when starting any form of a project or assignment you always need to layout and ensure that it is done in correlation to meet those requirements and expectations. When developing 3D TicTacToe we had to turn to the drawing board to get an understanding of how the overall interface of the 3-dimensional game would look like in comparison to its 2d counterpart. This refers throughout the development process of building our TicTacToe program methods and classes in order to get the full functionalities of the game board, the game’s player and AI control and the interface of the game to make the UI to the user accessible. Which in testing, we had to ensure that there were no errors such as syntax, runtime or logic within our program classes. An example of this was when our TicTacToe program was ready to be tested. So we had what we would call a beta version of this program running on all the Grade 11’s computers to set up a TicTacToe tournament and get some feedback on any errors and bugs found during the process of running the program. We implemented and delivered the program because after our program was finished beta testing with the feedback from the Grade 11 ’s. This helped us improve and debug our program to a stable state. Due to the phase of maintenance, the 3D TicTacToe program is still not free of any errors such as runtime or logic that we will then need to modify and update our TicTacToe program to ensure that all of our code is in proper use of the Java language and that it gives a sense for what it is fully supposed to do.



**Level 2: Understanding The Software Development Lifecycle Phases**

Explain how the problem definition phase could have been improved to produce a clearer specification that would result in a richer set of application features.

Consider the use of the following techniques: dialogue, questionnaires, surveys, and research

As a team, develop a tool/technique that will help to improve your 3D TicTacToe specification.

Think about what you did regarding analysis and design for the 3D TicTacToe project.

What was good about the process you followed?

The general idea behind the game board methods as those are simply defined on how the moves and functionalities of the game are inputted and work. Simply put it is a lot more easier to call and define as they are methods that contain conditional based code such as if statements which cover the many possibilities from the 3D TicTacToe game in perspective of the players capabilities based on the game’s 3D structure which are then returned true/false to tell if the move is valid or invalid.

What was bad about the process and what could be done to improve the process?

Think about what will be required for testing your 3D TicTacToe project code.

List some major things that you will have to test?

We will have to test for any necessary runtime or even logic errors as most of our classes seems to be free of syntax currently but there is always a possibility of failure while the program is running and some functions with methods may need to be debugged or re-coded because of compatibility issues and lack of functionality when called and then runned.

.

Research about what a software test plan looks like

Create a test plan for your 3D TicTacToe project code

This test case is based on the overall plan for our current working version of the TicTacToe program.

Action; Open up a new game

Result; Clear/refresh game board

Action; Click on empty space for first move

Result; Places an X

Action: Click on empty space for second move

Result; Places an O

Action; Click on an already selected space

Result; Message printed that you don’t lose your move.

Action; Player X clicks on an empty square

Action; Player X clicks on an X square

Action; Player X clicks on an O square

Write similar tests cases for Level 2 and 3.

Action; All moves are made with no winner

Result; Tie game

Action; Test case for each of the 49 winning combinations.

**Missing features:**

- Restart Button

- Score System/Tracker

**To-Do List:**

Assign names and delivery dates for the To Do List items.

- Finish programming win conditions

- Finish implementing Restart button

- Finish implementing Score button

- Finish documenting test cases

- Run application on Grade 11’s Computer

- Rules for Grade 11 TicTacToe competition

- Collect Feedback from Grade 11s

**Test Plan:**

Introduction:

Provide an overview of the test plan.

Specify the goals/objectives.

Specify any constraints.

This will be used to give an overview of the various factors such as any unknown issues we have with our program in relation our level of organization with developing this project to help cope and maintain the 3D TicTacToe game’s working status.

References:

List the related documents, with links to them if available, including the following:

Project Plan

Configuration Management Plan

**The Game board section of the project is outlined with the creation of :**

Public Constants & Variables = Constant palyerFree, Constant playerX, Constant playerO.

Public Methods= boolean makeMove( playerXorO, Level, Row, Column)Returns true/false depending if move is valid, boolean deleteMove(Level, Row, Column)

Returns true/false depending if board space was occupied playerXorO checkMove(Level, Row, Column), Returns Free, X, or O depending how the board space is occupied

**The GameAI section of the project is outlined with the creation of :**

Public Methods= boolean isWinForX (GameBoard) Returns true if player X has won,

boolean isWinForO (GameBoard) Returns true if player O has won, boolean isGameOver (GameBoard) Returns true if all moves have been made,

[level,row,column] bestMoveForX (GameBoard) Returns the best next move for player X. [level,row,column] bestMoveForO (GameBoard) Returns the best next move for player O.

**The Interface section of the project is outlined with the creation of :**

Application Requirements = Controls Game Flow / Play, Creates Game Interface Using SWT Widgets, Listens for Actions on Widget buttons / content changes, Calls Included Classes / Objects to verify and implement game moves.

Classes = Game Board Class - Implement & Manage Game Board and Moves

GameIA – To Check for WIN / Recommend next move

Test Items:

List the test items (software/products) and their versions.

Eclipse Oxygen running on a Windows 8 Computer.

Features to be Tested:

List the features of the software/product to be tested.

Provide references to the Requirements and/or Design specifications of the features to be tested

The game board’s interface of the 3 dimensional structure and its interaction with basic functionalities and rules that both the player and the AI need to comply with.

Approach:

Mention the overall approach to testing.

Specify the testing levels [if it’s a Master Test Plan], the testing types, and the testing methods [Manual/Automated; White Box/Black Box/Gray Box]

A host of Grade 11’s will test and run the 3D TicTacToe program. They would download and open the file which will be posted on Github that can be accessed from a computer which will run our 3D TicTacToe program. We will convert our java program into one executable file type that is capable of automatically running upon clicked without having to have access to the source code and use a compiler within eclipse.

Item Pass/Fail Criteria:

Specify the criteria that will be used to determine whether each test item (software/product) has passed or failed testing.

If the game can perform the basic functions of moving by selection and the conditional possibilities of player validation can be applied such as checking if a row/column moved is valid or invalid then it is quite stable meaning a pass if less than these expectations we would probably consider it a fail.

Test Environment:

Specify the properties of test environment: hardware, software, network etc.

List any testing or related tools.

Windows 7 Computers connected to a LAN network that are capable of accessing a browser to download and run files.

Estimate:

Provide a summary of test estimates (cost or effort) and/or provide a link to the detailed estimation.

A few days as it will take a day to possibly run the testing process with the Grade 11s and when that is over we will need a day or two to try and debug the program methods that need to be addressed.

Responsibilities:

List the responsibilities of each team/role/individual.

Me and Antonio are assigned the role to create the Game board (Check Possibilities and Player moves).

Rahul and Kiran have the part to create the Game board interface ( Overall UI containing an interactive SWT based screen with the general 3D game structure and buttons).

Robin and Kiran have the part for the Games AI (Check Player/AI moves and winnings)

Assumptions and Dependencies:

List the assumptions that have been made during the preparation of this plan.

List the dependencies.

The game board methods related to moves may need to be debugged as there is a possibility of runtime/logic errors.

Approvals:

Specify the names and roles of all persons who must approve the plan.

Provide space for signatures and dates. (If the document is to be printed.)

Mr. Nestor who is our teacher (client) that provided us with this concept will validate and give feedback.

For the delivery phase, think about what will be required to host a small 3D TicTacToe tournament for the Grade 11 Students.

How will the tournament be organized?

We will convert our java program into one executable file type that is capable of automatically running upon clicked without having to have access to the source code and use a compiler within eclipse.

How will they load and run your TicTacToe application?

They would download and open the executable file which will be posted on Github on their computer which will run our 3D TicTacToe program.

How will you deal with problems and bug reports?

We will ensure to modify any of our classes that the bug is responsible for and if necessary even re-code some parts from scratch because as mentioned before our programs are free of possible syntax but there is a possibility of errors and lack of functionality while the program is running in which a group of testers can do a better job of pinpointing.

**Level 3: Applying The Software Development Lifecycle Phases**

t.b.d

**Curriculum Notes**

B4.1 describe the phases (i.e., problem definition,

analysis, design, writing code, testing, implementation,

maintenance), milestones (e.g., date

of completion of program specification), and products

(e.g., specification, flowchart, program,

documentation, bug reports) of a software development

life cycle;

B4.3 use project management tools (e.g., Gantt

chart, critical path diagram, PERT chart) to show

tasks and milestones in a teacher-led project;

B4.6 communicate information about the status

of a project (e.g., milestones, work completed,

work outstanding) effectively in writing

throughout the project.

Grade