Software Solution Report

Kyer Potts 30003389

Table of Contents

[Introduction 1](#_Toc4058427)

[Software Development Life Cycle 2](#_Toc4058428)

[Planning and Requirement Analysis 2](#_Toc4058429)

[Defining Requirements 2](#_Toc4058430)

[Designing the Product Architecture 2](#_Toc4058431)

[Build and Development 2](#_Toc4058432)

[Testing 2](#_Toc4058433)

[Deployment and Maintenance 2](#_Toc4058434)

[Option One 3](#_Toc4058435)

[Waterfall Model 3](#_Toc4058436)

[Waterfall Model Pros 3](#_Toc4058437)

[Waterfall Model Cons 3](#_Toc4058438)

[Option two 4](#_Toc4058439)

[Agile Model 4](#_Toc4058440)

[Agile Model Pros 4](#_Toc4058441)

[Agile Model Cons 4](#_Toc4058442)

[Summary 5](#_Toc4058443)

[Recommendation 5](#_Toc4058444)

# Introduction

The SDLC is a necessary component of the software design process. It is a broad high-level plan that outlines the stages of product development based upon client and stakeholder requirements, and functionality requested by the scope. There are several SDLC methodologies that have been defined and each have benefits and risks associated with each methodology. Therefor it is important to decide on which model to use based upon the nature of the project.

# Software Development Life Cycle

The software development life cycle are predefined processes that lay the framework for the progress and development of a software project. These processes are implemented to ensure that the development of the project runs efficiently and succinctly. SDLC processes provide direction for the team when developing, changing or upgrading software. SDLC methodology emerged in the 1960’s in response to the development of large-scale business systems. These systems often took a great deal of development time and resources to construct and implement and required processes to coordinate team efforts. The typical development life cycle consists of six stages:

### Planning and Requirement Analysis

Planning and requirement analysis is integral to the development life cycle. Requirement information is taken from a variety of sources, which is then analysed and utilised to create a comprehensive plan that considers quality assurance, and risks associated with the project.

### Defining Requirements

Requirements for the project are then defined by laying out expectations of the product in terms of functionality and performance. These requirements will usually require approval from the customer. This is compiled into an SRS – Software Requirement Specification.

### Designing the Product Architecture

After the requirements have been analysed and defined, the architecture for the project is designed by product architects. The architecture is based of the requirements for the project and will provide a modular blueprint for the development of the software. This will include communication and data flow representations and charts. This information is documented within a DDS – Design Document Specification. This document will include definitions of all of the modules used within the specification.

### Build and Development

This stage marks the beginning of the construction of the product. The DDS is used during this stage to develop the code that follows the correct infrastructure and processes outlined within the product architecture documentation. Code guidelines should be followed during this stage to ensure uniformity of code, and readability for other members of the team.

### Testing

The testing stage is used to identify and problems within the product, in the form of bugs or functionality errors. These errors are identified and reported, and returned to the development team to implement fixes, which are retested to ensure that the product works as intended.

### Deployment and Maintenance

Once the development of the product has been completed and tested, it is released for use. Post release use might identify further errors or additional required functionality. These problems are addressed, or, if they fall outside of the scope, maybe incur additional cost to the customer.

# Option One

## Waterfall Model

Waterfall methodology consists of sequential steps or phases of development that follow a linear progression. Theses steps will usually consist of logical sequences for planning, design, construction, testing and implementation, deployment and maintenance if required.

### Waterfall Model Pros

* Ease of use
* Clearly defined stages
* Clearly defined goals
* Organisationally simple
* Processes and results are well documented throughout stages
* Structure of the model ensures simple management of deliverables

### Waterfall Model Cons

* Testable or usable software is not available until late during the life cycle
* High risk and uncertainty factor due the rigid nature of the model
* Simplicity of the model does not mesh well with complex object orientated projects
* Structure of the model makes changing requirements difficult to implement
* Measuring progress of the product development is difficult
* Integration is done at the end of the project and doesn’t allow identification of technology or business bottleneck or other issues earlier within the development life cycle.

# Option two

## Agile Model

Agile methods segment the project into incremental builds which are provided in iterations. These iterations usually last several weeks, allowing deliverable versions of the software to be produced to the client for continual sign off until all software requirements are met. These iterations are produced by cross functional teams working simultaneously on various areas of the life cycle of the project.

### Agile Model Pros

* Promotes teamwork and cross training
* Rapid development of functionality and features
* Low resource requirements
* Adaptable to changing or added requirements
* Delivers basic working products incrementally
* Documentation is easily employed
* Little to no planning required before development begins
* Allows the developers to be flexible with the project
* Minimises scope creep
* Development progress is easily measured

### Agile Model Cons

* Difficult to handle complex dependencies
* Risk of sustainability and maintainability of the project
* A high-level plan and appropriate team leader is required to implement agile methods
* Deadlines for deliverables require strict adherence
* Depends on customer involvement with the project. Deliverables require continuous sign off during the development of the project
* Individual responsibility of each team member is high as there is little documentation for them to follow.
* Transfer of technology to additional team members can be difficult as little documentation is produced
* Requires strict adherence to code practices to ensure that code is uniform and readable across all areas of the project

# Summary

SDLC methodology is integral to the development of marketable software. The processes outlined allow technical leads to define the requirements and deliverables of the project, as well as provide the architecture necessary to develop software that effectively meets those requirements. This ensures that the product deployed to the market is of a high quality and falls within the project constraints.

### Recommendation

Agile methodologies are the most appropriate for a project of this size and scope. It is also appropriate for the implementation of changes to the scope, such as additional desired functionality, while minimising gold plating during development due to the short development cycles. It will also allow the product to be deployed in increments, minimising the amount of training needed while also familiarising them with the systems.