Parking Assignment Problem

Software Engineering Lab Report 1

January 23rd 2018

Professor In-Charge
K. Chandrasekaran

Nihal Haneef 16CO128
Gurupungav Narayanan 16CO1145

Process Model

Incremental Approach to Software Development.

- The initial software concept, requirements analysis, and design of architecture and system core are defined using the Waterfall approach.
- This is followed by a mini-Waterfall development of individual elements of the software. Each completed waterfall results in a new build of the software. A new build is the result of an ***increment***.
- Increments are done until all the software requirements are fulfilled.

Justification

1. ***We can make use of what we learn at each incremental step to make our software better.***
   The first algorithm we develop for assigning parking spaces to incoming cars might not be the most efficient and may not work as effectively for some scenarios. Having an incremental model would allow us to improve our current algorithm and implement it in the next build.

2. ***There's always concrete evidence of the work being done. The project completion status is easily measured.***
   Our software implementation idea is modular in nature. After completion of the base module (that would assign parking spaces to an arbitrary layout), it needs to be expanded and generalized to add more features. An incremental model would allow us to clearly know and understand which feature was implemented in which increment.

3. ***We can maintain control over the project by documenting, reviewing and receiving feedback.***
   There would be no "partial" incomplete features, since after each increment the module would work completely well but for only limited scenarios. Each module would be tested and documented before moving on to implement the next feature. This would allow us to control how each module is to be implemented and improve on it in further builds.

4. ***We can easily identify and prevent problems while integrating the project or while creating the architecture for the project.***
   Since each module is thoroughly tested and reviewed before deployment, it would allow us to identify any problems that we may have missed and correct them.

Explanation

Stage 1 : Requirements

The first stage of an incremental approach is to find the overall requirements of the software. Proper implementation of the Software Requirement Engineering Process needs to be carried out using appropriate Requirement Management tools (in our case, rmtoo) to find and establish User, System, Functional, Non-Functional and Domain requirements.

Stage 2 : Design Stage

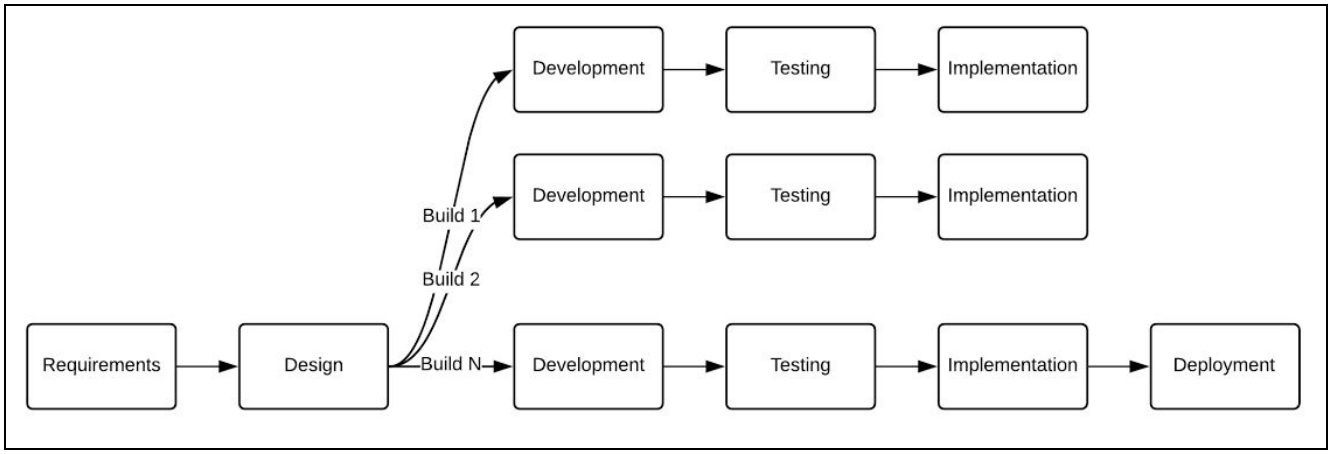
The overall structure of the software is designed in this stage. It includes empathising, definition, ideation and prototyping. The overall design structure of our software has been planned out and will be designed on the Star UML CASE tool.

Stage 3 : Incremental Stages (Mini-Waterfalls)

Each increment is a mini-waterfall that involves development, testing and implementation. Each incremental stage would attempt to fulfill a requirement and build a new version of the software that is more efficient and effective than any previous build.

Most of our software's requirements aren't explicitly dependant on any other requirement to be fulfilled first. This allows us to treat every requirement separately and fulfill all the requirements one build at a time.

For instance, say we want to add an OCR feature that would automatically detect number plates and return it to the system. An increment would involve designing and testing our OCR module and then integrating it with the rest of the software.



Tools

We're planning to implement the project using Java and the GUI using Swing.

S.No	Stage	Tool Name	About the Tool
1	Requirements	rmtoo	We plan on using rmtoo to help manage our requirements and their dependencies. Rmtoo also integrates nicely with git and with maven (which builds our Java project).
2	Design	Star UML	Star UML will be used to make UML use case diagrams for the software.
3	Mini Waterfall	IntelliJ	IntelliJ is going to be our primary IDE.
		MongoDB	We're using MongoDB for managing our data.
		Jenkins	Jenkins is a continuous development and deployment tool that integrates with git and works well with Java. We may not use this tool, since it isn't explicitly apparent to us if it's actually necessary.

Summary

The ultimate goal of our project is to build a software which would solve the parking slot assignment problem. The SDLC chosen to systematically build this software is the ***Incremental Life Cycle Model.*** It consists of first the *requirements stage*, where all the requirements of the software are mentioned and established. We then move onto the *design stage* where after careful ideation and prototyping we create the design structure of our software using appropriate tools. It then moves on to N number of *mini waterfalls*, each consisting of a development, testing and implementation stages (each waterfall is called a build). N builds culminate to the final software which will be deployed.
In this report we also mention the various CASE tools that will be utilized by us for the implementation of our SDLC and for the creation of our software along with their appropriate applications.