# PRT 452 Project Report

Project: Improving SNP-based microbial genotyping software

Written by: Kian Soon Hoon (s304689)

Email: ldwgkshoon@gmail.com

Table of Contents

[PRT 452 Project Report 1](#_Toc493244590)

[Introduction 2](#_Toc493244591)

[Requirements 3](#_Toc493244592)

[UML design 3](#_Toc493244593)

[Testing 3](#_Toc493244594)

[TDD 3](#_Toc493244595)

[Change Management 4](#_Toc493244596)

[Configuration Management 4](#_Toc493244597)

[Continuous Integration 4](#_Toc493244598)

[System Building 4](#_Toc493244599)

[Design Pattern 4](#_Toc493244600)

[Pair Programming 5](#_Toc493244601)

[Story Card 5](#_Toc493244602)

[Artefacts 5](#_Toc493244603)

[Agile Process 5](#_Toc493244604)

[Security 5](#_Toc493244605)

[Code 5](#_Toc493244606)

[Documentation 6](#_Toc493244607)

# Introduction

In clinical, public health, and research microbiology, there is frequently a need to divide bacterial species into “types”. A bacterial type may be thought of as a classification unit at a finer scale than “species”, and roughly synonymous with the term “strain”. Bacterial typing is used to trace instances and patterns of transmission and dissemination, and may be used at all scales from global to within individual buildings (e.g. hospitals), to indicate virulence or resistance properties of isolates, and in basic research to assist understanding of evolutionary history. While early typing methods were based upon reactions with standard sets of antisera (serotyping), or susceptibility to standard sets of viruses (phage typing), for decades, the great majority of bacterial typing has been performed using genetic methods. As a result, the term “genotyping” will be used in this document. Over the years, considerable ingenuity has been applied to the development of bacterial genotyping methods. Many methods have been described, with most being variations on the theme of electrophoresis of complex mixes of DNA fragments so as to yield banding pattern “fingerprints”, or sequencing of standardised gene fragment(s).

In the early 2000s, Phil Giffard led a project within the QUT node of the Cooperative Research Centre for Diagnostics. This was essentially a re-think of bacterial genotyping methodology. The innovative concept was that the volume of known comparative gene sequence data from within bacterial species was exploding. This constitutes a suburb resource to mine for sets of variable genetic locations (single nucleotide polymorphisms (SNPs)) that are optimised for use in bacterial genotyping methods with predefined performance specifications. In this way, genotyping methods that yield the needed resolving power or other information as efficiently as possible could be easily designed. An added impetus for this area of research was rapid development in technology for interrogating SNPs. The first publication outlining this approach was published in 2004. This reported the construction of the SNP-mining “Minimum SNPs” software package, and the demonstration of two bacterial genotyping methods, using allele specific PCR on a real time PCR platform.

A major development in very recent years has been the advent of low cost whole genome sequencing of bacteria. The work described above made extensive use of multilocus sequence typing (MLST) databases as the input data for SNP mining. MLST was developed in the UK (e.g. see http://saureus.mlst.net/), as a major and very successful initiative to standardize bacterial genotyping methods and terminology. For each bacterial species for which it has been implemented, seven standardised gene fragments are defined. These are sequenced to generate the “sequence type” (ST). The MLST database curators maintain comprehensive internet accessible information regarding sequencing variants at the loci (alleles) and the alleles found together in bacterial isolates (STs). Often several thousand STs are known. Our SNP based typing methods have been based on mining alignments of these STs. The MLST loci only cover ~0.1% of the bacterial genome. Lost cost whole genome sequence makes it possible to genotype a bacteria by whole genome sequencing and causes MLST obsolete. However, despite the decreasing cost and time needed, sequencing an entire genome still requires a substantive amount of money and time. On the other hand, the increasing data whole genome sequences available openly online provide the opportunity of identifying useful Single Nucleotide Polymorphism (SNP). Accordingly Giffard, work is shifting to SNP mining from alignments of entire genomes rather than alignments of small fragments of genomes such as MLST databases. This means that ~1000 more SNPs are available for selecting optimal combinations, resulting in much higher performance of the genotyping methods and hence called for the improvement of the existing software.

This project will scrap the old system and reverse engineer the original system in R programming environment. The project will also implement some extra features as described by Philip Giffard. An example of the extra features is the implementation of error checking for the FASTA file accepted by the program prior computation of the minimum SNP, this will ensure that the program doesn’t crash in case it has receive files with allelic profile(s) that contain deletion(s). Besides, it will also create a table to present the SNPs and the allelic profiles that they detect.

# Requirements

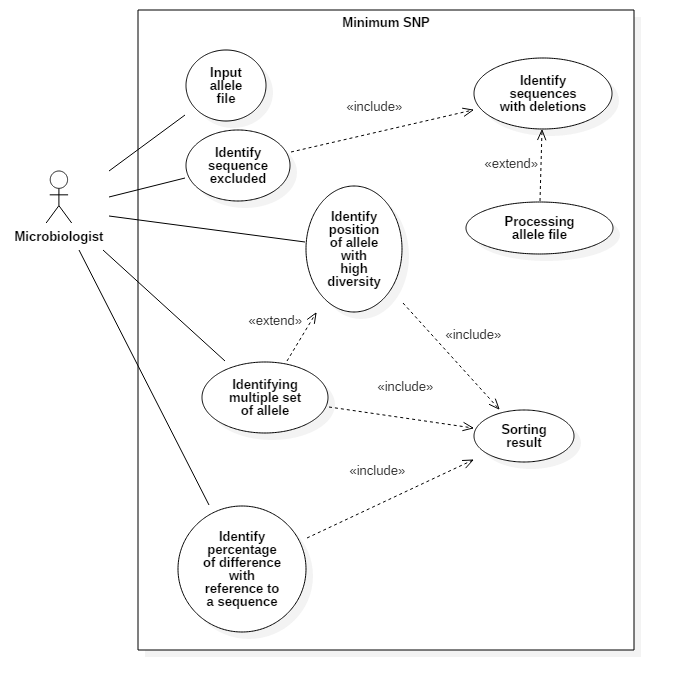
Due to time and resource constraints, the scope of this project has been limited to reverse engineering some functionalities from the Minimum SNPs software as described in Price E.P., Inman-Bamber, J., Thiruvenkataswamy, V., Huygens, F and Giffard, P.M. 2007, Computer-aided identification of polymorphism sets diagnostic for groups of bacterial and viral genetic variants. BMC Bioinformatics 8:278. The original functionalities can be found in the Minimum SNPs software user manual which is attached together in the manual folders. According to Phillip Giffard, the followings are the requirements of the project:

1. Reverse engineer the ‘Minimum SNPs’ software into R programming environment.
2. Improve the error checking of the new ‘Minimum SNPs’.
3. Integrate additional features into the new ‘Minimum SNPs’.

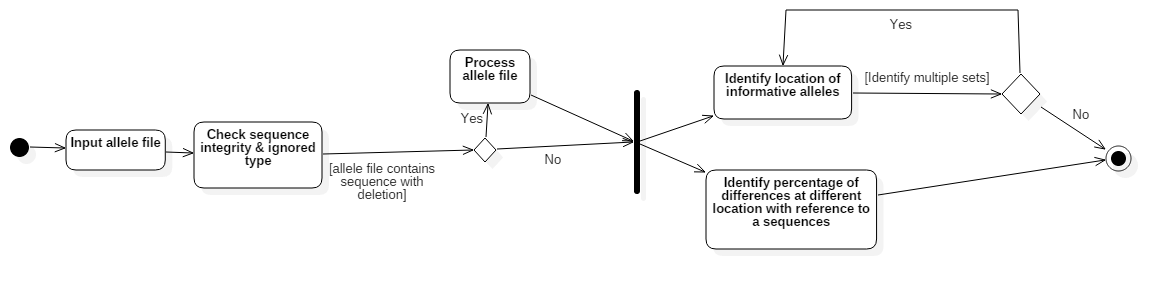
\*\*Talk about the architecture from the project plan & how the current project fits into the architecture.

# UML design

The use case diagram below outlines how the users will interact with the new ‘Minimum SNPS’.



The activity diagram below outlines how the user will use the new ‘Minimum SNPs’ to get result(s).



# Testing

Testing is very much the essence of this project. Testing is used to ensure that the program will runs smoothly without error. Using Test Driven Development (TDD), it is ensured that the testing covers 100% of the code. The program is tested in the development platform before being committed to the GitHub repository.

# TDD

Unit-testings have been designed prior to the coding. The results expected are from the descriptions of the clients and the ‘Minimum SNP’ software written in Java. The software is run using the dataset and the result is used in unit testing to ensure that the calculation is correct. In situation where the result is unattainable (i.e. when it’s a new features or modification requests to the new software), description of the client is used and expected result are calculated or generated manually.

Some sample of the tests are shown below.

\*\*

# Change Management

## Backlog

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Iteration | Date | Title | Descriptions | Priority | Status |
| 1 | 28August 2017 | Detect abnormal | Detecting the allelic profiles which are ignored during computation of the minimum SNPs and informing the users about the ignored profiles. | Low | Completed |
|  | 28 August 2017 | Percentage calculation | Implementing percentage of differences at different loci with reference to a particular allelic profile | Medium | Completed |
|  | 31 August 2017 | Simpson’s index | Implementing the Simpson’s index of diversity. | Medium | Completed |
| 2 | 5 August 2017 | Table for Simpson’s | Focus on creating table for Simpson’s index and corresponding allelic profile | High | Completed |
|  | 5 August 2017 | Simpson’s search | Looking for multiple SNP that will define a particular allelic profile from the rest. Implementing multi-levels Simpson’s index. | Medium | Completed |
| 3 | 13 September 2017 | Simpson’s inclusion & exclusion | Including and/or excluding certain SNPs while calculating Simpson’s index. | High | Completed |
|  | 13 September 2017 | Simpson’s search (Multiple pairs) | Giving the option to generate multiple pairs of SNPs. This can be done by running the calculation again with the exclusion of the 1st SNP found. The rationale is that the 1st SNP with highest Simpson’s index may not return the highest index when combined with the other SNPs. | High | Completed |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

## Story Card

Story cards are used to help with the generation of requirements. It shows how the users will normally use the system and help with the coding process.

|  |  |
| --- | --- |
| Story name (ID) | Detect abnormal (1) |
| As a | Microbiology researcher |
| I want to | Input FASTA file for analysis and be informed if there are allelic profiles that contains deletion and are ignored from computation |
| So that | I know what are the allelic profiles that have been excluded and make my work easier |
| Importance | Low |
| Estimated time | 2 days |

|  |  |
| --- | --- |
| Story name (ID) | Percentage Calculation (2) |
| As a | Microbiology researcher |
| I want to | Calculate the percentage of differences at different location for an allelic profile |
| So that | I can find the location(s) of the SNPs with the highest percentage of difference |
| Importance | Medium |
| Estimated time | 2 days |

|  |  |
| --- | --- |
| Story name (ID) | Simpson’s Index (3) |
| As a | Microbiology researcher |
| I want to | Calculate the Simpson’s index of the SNP at a location |
| So that | I know how useful the SNP at that location is |
| Importance | Medium |
| Estimated time | 2 days |

|  |  |
| --- | --- |
| Story name (ID) | Table for Simpson’s index (4) |
| As a | Microbiology researcher |
| I want to | To know see the sequence type in a table |
| So that | I can know how the sequence types are categorised given any number of SNPs |
| Importance | High |
| Estimated time | 3 days |

|  |  |
| --- | --- |
| Story name (ID) | Simpson’s search (5) |
| As a | Microbiology researcher |
| I want to | Find the SNP that returns the highest Simpson’s index at each level |
| So that | I can find the most useful SNPs to categorise the sequence types |
| Importance | High |
| Estimated time | 4 days |

|  |  |
| --- | --- |
| Story name (ID) | Simpson’s search inclusion & exclusion (6) |
| As a | Microbiology researcher |
| I want to | Include and exclude certain SNPs in the Simpson’s search |
| So that | I am able to see how the SNP performs under certain situation |
| Importance | High |
| Estimated time | 3 days |

|  |  |
| --- | --- |
| Story name (ID) | Simpson’s multiple search (7) |
| As a | Microbiology researcher |
| I want to | Have multiple SNP returned at different levels |
| So that | I am convinced that the SNPs I found is really better than the rest, as different SNP perform differently when combined with another SNP. |
| Importance | High |
| Estimated time | 5 days |

|  |  |
| --- | --- |
| Story name (ID) |  |
| As a | Microbiology researcher |
| I want to |  |
| So that |  |
| Importance |  |
| Estimated time |  |

\*\*

# Configuration Management

How the software can be maintained after being release?

How the version control for the software is done?

# Continuous Integration

Travis CI is used for continuous integration. Travis CI is a free services run on top of Github and is completely integrated with Github. In order to support R programming language, 2 R packages are used, i.e. devtools and r-travis.

# System Building

The output of the project is a

# Design Pattern

# Pair Programming

Pair programming was originally planned to be used in order to reduce errors in code and to make it easier when different people needs to modify the code. However, it is not used as a technique in the end because the number of member in the group has been reduced to one.

# Artefacts

# Agile Process99

As part of the agile process, there have been frequent releases. Moreover, the client was met at least once a week and client also gets to decide what features are prioritised first. Inputs from the client is put as the top priority. Moreover, different team members also work on different part of the project together and different tools that support the agile process are used. Some of the examples are shown below.

\*\*

# Security

Security is not a major part of the project as it is designed to be used by individuals and are not connected to the Internet or contain any sensitive information. However, it is important that the data imported by the users maintains its integrity throughout the usage of the program. This is because the integrity of the data will affects the result from the computation. Hence system testing and integration testing are focused on checking that the data passed between functions are in their right format and are integrated. Some examples of the efforts to ensure that is shown below.

# Code

The code is available in the GitHub repository. Please refer to [https://github.com/ludwigHoon/PRT452-BINFO-](https://github.com/ludwigHoon/PRT452-BINFO-%20) for the codes. Some snippets are presented here for some discussion.

# Documentation

The user documentation can be found attached together with the software.

The technical documentation are generated with roxygen2. It helps the users with the usage of the software. The technical documentation can be found in the manual folders in the software package. Alternatively, it can be accessed in the R programming environment by typing help(<function>). Please refer to the github repository for the documentations.