Problem Based Learning Report

SENG3160, The University of Newcastle

Atharv Naphade

c3386230

# Contents

[Contents 1](#_Toc207368822)

[1. Abstract 2](#_Toc207368823)

[2. Literature Review 3](#_Toc207368824)

[2.1. Introduction 3](#_Toc207368825)

[2.2. Areas of Software Testing 3](#_Toc207368826)

[2.2.1. Unit and Integration Testing 3](#_Toc207368827)

[2.2.2. System Testing 5](#_Toc207368828)

[2.2.3. Acceptance Testing 5](#_Toc207368829)

[2.2.4. Contract Testing 5](#_Toc207368830)

[2.2.5. Non-functional Testing 5](#_Toc207368831)

[3. Testing Plan Proposal 6](#_Toc207368832)

[3.1. Testing Strategy 6](#_Toc207368833)

[3.2. Justifications 6](#_Toc207368834)

[3.3. Tooling 6](#_Toc207368835)

[3.4. Integrating testing into development workflow 6](#_Toc207368836)

[4. Conclusion 7](#_Toc207368837)

[5. Bibliography 8](#_Toc207368838)

# Abstract

This report illustrates a comprehensive literature review of a few key software testing techniques, unit testing, integration testing, system testing, acceptance testing, contract testing, and non-functional testing.

# Literature Review

## Introduction

Testing is important component of software engineering as it ensures reliability, maintainability and performance. There are various types of testing techniques and processes that are undertaken at different points in the software development life cycle.

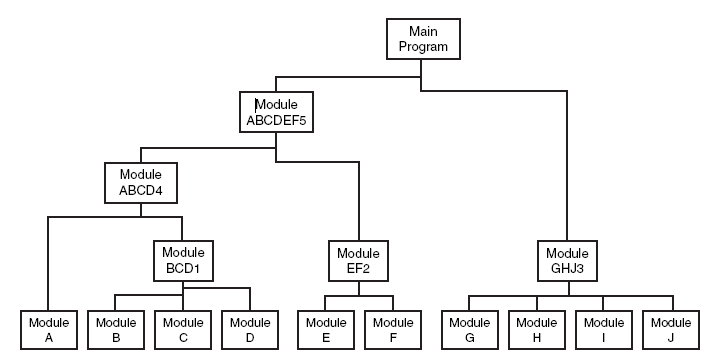
For any given software project, it is important to strategically identify areas that are testable and will potentially lead to improvements, mitigations, or provide insight into the health of the project at any given point.

The following sections provide details on some of the popular types of software testing approaches.

## Areas of Software Testing

### Unit and Integration Testing

Module testing (or unit testing) is a process of testing the individual sub-programs, subroutines, classes, or procedures in a program (Myers et al., 2012, p. 85). Unit testing is usually undertaken parallelly to writing code, this allows for quick feedback and concrete definitions for each component’s behaviour such as inputs, outputs and edge cases. Myers et al., 2012 presents three motivations for why unit testing is used (p. 85) –

* It aids in managing other kinds of testing that involve multiple units working together, as it allows for working on the knowledge that each of the individual units are functioning well.
* It simplifies debugging by allowing developers to immediately direct their attention to concise units of the software program, in case of errors.
* It is a time and cost-efficient form of testing as it allows for parallelly running tests on all units, as they are detached from each other’s state.

**Figure 2.1.**

Representation of modules within a software program, shown as being integrated for testing.

*Note.* From *Software Testing* (2nd ed., p. 109), by R. Patton, 2005, Sams Publishing.

Integration testing is very closely linked to unit testing, as it involves testing the identified units’ interactions and finding any interfacing issues. Figure 2.1. illustrates how several units/modules in a software program can be organized together in a hierarchical manner.

### System Testing

### Acceptance Testing

### Contract Testing

### Non-functional Testing

# Testing Plan Proposal

## Testing Strategy

## Justifications

## Tooling

## Integrating testing into development workflow

# Conclusion

# Bibliography

Myers, G. J. ., Sandler, Corey., & Badgett, Tom. (2012). Module (Unit) Testing. In *The Art of Software Testing* (3rd ed.). John Wiley & Sons.

Patton, Ron. (2013). Software Testing. In *Software Testing* (2nd ed.). Sams Publishing.