FACULTY OF SCIENCE
UNIVERSITY OF COPENHAGEN

MSc thesis

Søren Lund Hess-Petersen - pws412

Using Property-Based testing to test MCP Instance-Specifications

Academic advisor: Michael Kirkedal Thomsen
<m.kirkedal@di.ku.dk>

Submitted: October 1, 2019

Using Property-Based testing to test MCP Instance-Specifications

Søren Lund Hess-Petersen - pws412
DIKU, Department of Computer Science,
University of Copenhagen, Denmark

October 1, 2019

MSc thesis

Author: Søren Lund Hess-Petersen - pws412

Affiliation: DIKU, Department of Computer Science,
University of Copenhagen, Denmark

Title: Using Property-Based testing to test MCP Instance-Specifications /

Academic advisor: Michael Kirkedal Thomsen
<m.kirkedal@di.ku.dk>

Submitted: October 1, 2019

Abstract

Dansk Resumé

Preface

This Master's thesis is submitted in fulfilment of the master programme in Computer Science at the University of Copenhagen for Søren Lund Hess-Petersen.

Contents

Introduction

1

Shortly describe the project; about 1-2 pages

- Set the environment/story of the project,
- build that into a problem leading up to the "hypothesis"
- give an overview of how you have investigated this hypothesis and point to important work you have done; it can be an advantage to use references to exact sections

This chapter is intended to give the reader the information about your report and build up and expectation of what to come. You want to make the reader aware of the good things you have made, so he has something to look forward to.

1.1 test

Before starting with a section, a chapter should include a short description of what is to come and possible what assumptions you have made. Again we want to build up the expectation of the reader.

Background 2

Everything that is needed to understand you project that you have not made your self. Remember that the report should be written to a student on your level. Thus, material from the first couple of years can be general knowledge. Though if you some part extensively (e.g. further develop) it can be a good idea to recap. For example, if your project is to develop a better version of Dijkstra's algorithm , it can be a good recap it. However, if you are developing a domain specific languages, there is no need to recap lexer/parser generators , as these are used as tools.

2.1 Analysis

2.2 Parser Grammars

2.2.1 ServiceSpecificationScema

2.2.2 ServiceDesignSchema

2.2.3 ServiceInstanceSchema

Work

3

3.1 Implementation

3.2 Testing

3.3 Experiments

Results

4

Discussion

5

Conclusion

6

6.1 Future work