

**National University of Singapore**

**School of Computing**

**CS3202: Software Engineering Project II**

**TEAM 05: Flying Cockroach**

Semester 1, AY2014/2015

|  |  |  |  |
| --- | --- | --- | --- |
| **Matriculation Number** | **HP Number** | **Student Name** | **Email** |
| **Group-PQL:** | | | |
| A0099214B | 8518 2707 | Adinda Ayu Savitri | savitri.adinda@gmail.com |
| A0098139R | 9082 0864 | Hisyam Nursaid Indrakesuma | indrakesuma.hisyam@gmail.com |
| A0103494J | 9620 7018 | Lacie Fan Yuxin | lacie.jolene.fan@gmail.com |
| **Group-PKB:** | | | |
| A0101286N | 9833 2474 | Ipsita Mohapatra | ipsita@nus.edu.sg |
| A0080415N | 9148 6248 | Steven Kester Yuwono | a0080415@nus.edu.sg |
| A0099768Y | 9178 6540 | Yohanes Lim | yohaneslim93@gmail.com |

Consultation Day/Hour: Monday 6-6.30pm

Contents

[1. SPA 3](#_Toc400895570)

[1.1. Architecture 3](#_Toc400895571)

[1.2. Development Plan 4](#_Toc400895572)

[1.2.1. For Whole Project 4](#_Toc400895573)

[1.2.2. For Iteration 3 5](#_Toc400895574)

[2. Components 6](#_Toc400895575)

[2.1. Code Parser 6](#_Toc400895576)

[2.2. PKB 7](#_Toc400895577)

[2.3. Design Extractor 8](#_Toc400895578)

[2.4. Query Processor 9](#_Toc400895579)

[2.4.1. Query Processor 9](#_Toc400895580)

[2.4.2. Query Parser 9](#_Toc400895581)

[2.4.3. Query Evaluator 9](#_Toc400895582)

[3. Testing 10](#_Toc400895583)

[3.1. Testing Plan For Iteration 3 11](#_Toc400895584)

[3.2. System Testing 11](#_Toc400895585)

[4. API 11](#_Toc400895586)

# 1. SPA

Static Program Analyser (SPA) is a program to answer queries about an input SIMPLE program. In this report, we will be describing the design and implementation decisions made during the development of the basic SPA and any bonus features.

## Architecture

The architecture consists of 3 main components: the Code Parser, the PKB and the Query Processor. Code Parser parses the code and stores design abstractions in each of the 8 tables in the PKB. After Query Parser has parsed the query, the Query Evaluator consults the PKB API to answer queries.

Both the Code Parser and the Query Processor are dependent on PKB but not dependent on each other. This ensures low coupling between components within SPA. At the same time there is high cohesion within each component, as described in the subsequent parts.



Figure 1

## 1.2. Development Plan

### 1.2.1. For Whole Project

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Iteration 1** | | | | | **Iteration 2** | | | |
| Team member | **Implement Calls in PKB** | **Implement Modifies and Uses in PKB** | **Extend Query Parser to support with clause** | **Extend Query Evaluator to support with clause** | **Write system test cases for the enhancement** | **Implement Next in PKB** | **Extend Query Parser to support multiple clauses** | **Extend Query Evaluator to support multiple clauses** | **Write system test cases for the enhancement** |
| Adinda |  |  |  |  |  |  |  |  |  |
| Lacie |  |  |  |  |  |  |  |  |  |
| Hisyam |  |  |  |  |  |  |  |  |  |
| Steven |  |  |  |  |  |  |  |  |  |
| Ipsita |  |  |  |  |  |  |  |  |  |
| Yohanes |  |  |  |  |  |  |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Iteration 3** | | | |
| Team member | **Implement Affects in PKB** | **Extend Query Parser to support Affects and tuple results** | **Extend Query Evaluator to support Affects and tuple results** | **Write system test cases for the enhancement** |
| Adinda |  |  |  |  |
| Lacie |  |  |  |  |
| Hisyam |  |  |  |  |
| Steven |  |  |  |  |
| Ipsita |  |  |  |  |
| Yohanes |  |  |  |  |

### 1.2.2. For Iteration 3

# 2. Components

## 2.1. Code Parser

## 2.2. PKB

## 2.3. Design Extractor

The main role of the design extractor is to extract relationships about the SIMPLE program that could not be extracted in the one-time parsing done by the Code Parser. This includes:

* + Extracting information about Modifies and Uses for procedures and for program lines that are calls statements
  + Building the Control Flow Graph (CFG) from the AST and subsequently storing it in the PKB and storing the Next relationship in the PKB.

## 2.4. Query Processor

Query Processor consists of three parts: Query Processor (controller), Query Parser, and Query Evaluator.

### 2.4.1. Query Processor

Query Processor is a façade class for the whole component. Its responsibilities include:

1. Query Processor calls QueryParser to create a Query object from the given query string.
2. Query Processor then passes the Query object to the QueryEvaluator.
3. Query Evaluator will compute all necessary relations and return the results in the form of a list of integers.
4. Query Processor transforms the result into the correct display format and returns the answer to the user.

### 2.4.2. Query Parser

### 2.4.3. Query Evaluator

# 3. Testing

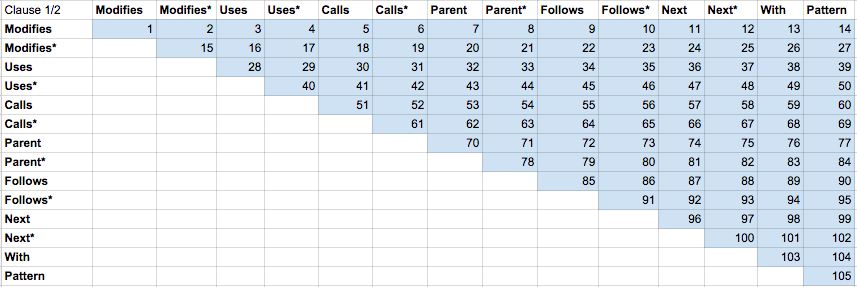
We did testing on 3 different levels, namely unit testing (using CPPUnit), integration testing (using CPPUnit) and system testing (using AutoTester). Unit Testing was done while coding the components, while integration testing was done between SIMPLE program parser and PKB and between PKB and Query component.

From the testing experience in this project, we realised the need for timely and consistent unit, integration and system testing. By testing individual components early, we detect bugs earlier in the project’s lifetime, thus, saving us time towards the end of the project. We also did regression testing by reusing our unit tests and system tests. This helped us to quickly identify bugs that could have been introduced while we were trying to solve other bugs.

## 

## Testing Plan For Iteration 3

## System Testing



# 4. API

Please view our Doxygen at:

[www.comp.nus.edu.sg/~kester/CS3202](http://www.comp.nus.edu.sg/~kester/CS3202)