Senior Design

Web Platform for Java Specifications

Design Document

Group May1620

Alex Dana, Deeksha Juneja, Cody Hanika
Advisor: Professor Rajan

Table of Contents

Figures

Figure 1: This is a diagram outlining the basic structural design and communication flow for our project.

Overview

Java specifications are precise definitions of methods of Java libraries. They are something that nearly all programmers could benefit from. However, formal specifications are hard to come by and often times difficult to produce. Formal specifications involve describing methods in mathematically precise language and verifying preconditions & postconditions.
Hence, the purpose of this project is to create a socio-technical solution to this problem. The problem will be solved by using pre-generated specifications (which may or may not be complete or correct) which will be automatically posted to a Q&A site (for example - stackoverflow) asking for further analysis on the specification. Then the program designed by May1620 will be used to analyze the answers and comments on the question to generate accurate and complete specifications.

Problem Statement and Background

Formal specifications provide a generalized, high-level, and easily extendable approach to verification. Through the use of formal specifications, it is easier to write correct code and to avoid the use of an API in a manner that was not intended. Currently, popular Java libraries either lack publicly available formal specifications or it is difficult to obtain these specifications. Previous attempts at creating specifications at scale have failed, due to the difficulty of generating correct formal specifications without human intervention.

System Level Design

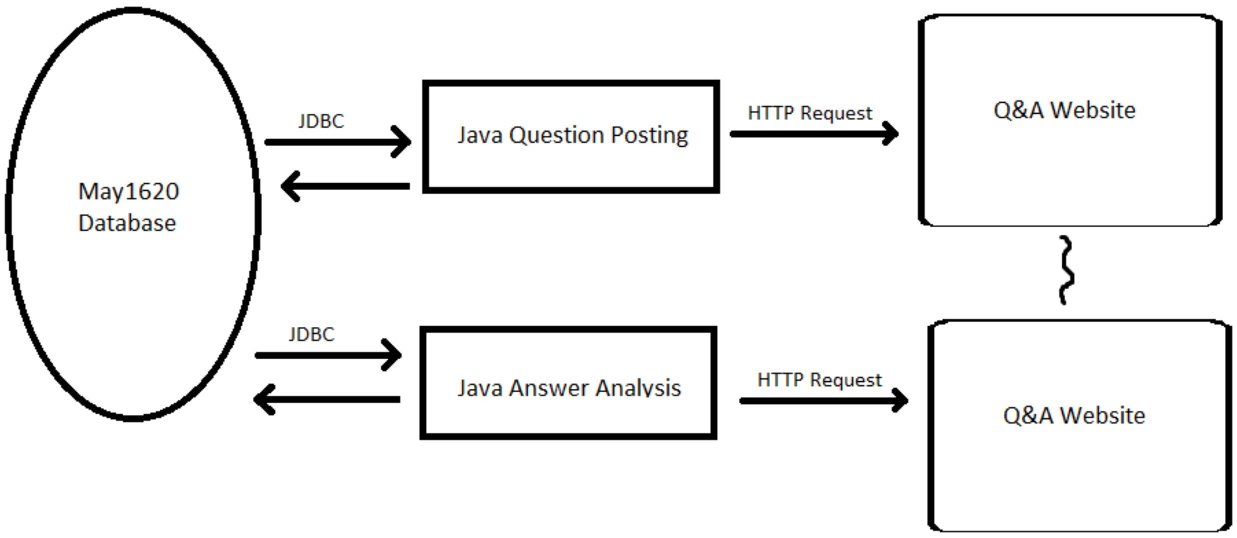
- The application must be able to communicate with the Q&A site via HTTP requests
- Specification questions must be posted to the Q&A site via HTTP requests
- Answers to the questions must also be retrieved with HTTP requests
- Specifications must be stored in a database to be retrieved and inserted
- Java will be used to analyze answers grabbed via the HTTP requests

There are 3 main parts of the application.

1. Database communication - This is the part of the project where the communication with the database is setup.

2. HTTP communication - In this part, the link between the project and the website is established.
3. Java analysis - Finally, using java, the responses are searched through to find meaningful specifications for a method.

Figure 1: This is a basic diagram showing the communication flow and design for the project



Detailed Description

Major Packages

analyze: The analyze package will contain all the classes related to the Stack Exchange answer analysis portion of our work.

common: The common package will contain classes that are used by many components of the system. For example, Specification is a class that will need to be used by all the components as the whole system will be dealing with a specification.

post: The post package will contain all the classes related to the posting side of the system. The Post class will contain the main method to run our class.

stackexchange: The stackexchange package will contain classes related to retrieving and posting data to Stack Exchange.

Each of these packages will have a corresponding test package that will contain JUnit tests for their functionality.

Database Tables

A MySQL database will be used to store all of the data.

Specification

Columns	**Type**
SpecificationId	Integer (Key)
PackageName	String
ClassName	String
MethodName	String
Finalized	Boolean

We will store both specifications that we have assumed to be correct based on community feedback and specifications that still need to be posted to be answered in this table. The Finalized column will specify whether this method has been posted and answered.

Precondition

Columns	Type
SpecificationId	Integer (Foreign Key)
Precondition	String

This table will have a row for each precondition that a certain method has.

Postcondition

Columns	Type
SpecificationId	Integer (Foreign Key)
Postcondition	String

This table will have a row for each postcondition that a certain method has.

OpenQuestion

Columns	Type
StackExchangeQuestionId	Integer
SpecificationId	Integer (Foreign Key)

This table will track open questions on StackExchange.

Major Classes

Post: This class will be responsible for posting a method to Stack Overflow to be specified. The high-level algorithm is this:

1) Query the Specification database table for a specification that is not yet finalized.
2) Create a Stack Overflow post title and body using this specification
3) Make an http request to Stack Exchange to post this question and store the question id returned
4) Store this question id in the OpenQuestion table

Analyze: This class will be responsible for analyzing the answers to open questions on Stack Overflow and inserting finalized specifications in the database.

1) Query the OpenQuestion table to get a list of open question ids.
2) For each open question, analyze the answers (this currently is not specified).
3) If an acceptable answer is found, accept the answer on Stack Overflow, update the specification in the Specification table as finalized, and remove the question from the OpenQuestion table.

Conclusion

Overall this project will reduce the problem of specifications being unavailable for popular Java libraries. The accessibility of these specifications will hopefully promote proper usage of the libraries and functions within; which will result in better and safer code. Although attempts have been made in the past to generate specifications, we feel that this project will be more successful because of the social side to of it. It allows for a larger level of analysis that cannot be achieved with just a technical solution.