

# Final Report for Team 5

## Project Team Team 5

Latest update on: 2017-06-08

### Team Information

---

컴퓨터공학부 201111383 전 훈

컴퓨터공학부 201311259 권오승

컴퓨터공학부 201311292 유효상

컴퓨터공학부 201511266 배윤희

# Table of Contents

---

## **I. Static Analysis Feedbacks**

- i. PMD
- ii. Intelli J
- iii. FindBug

## **II. System Review**

- i. Class Diagram

## **III. Realization**

# I. Static Analysis Feedbacks

- T5 의 정적 분석에 사용된 4 개의 분석 결과에 대한 문제 별 대응을 기술한다. T7 이 PMD, T6 이 IntelliJ 를 사용하였으며, 두 팀 모두 Eclipse Metrics Plugin 과 Find Bugs 를 사용하였고 그에 대한 결과도 동일하게 나왔다.

## PMD

문제	중괄호 문제 - ForLoopsMustUseBraces - IfElseStmtsMustUseBraces,
대응	특별한 문제가 생기지 않으므로 그대로 둠

문제	비어 있거나 필요 없는 코드 - EmptyIfStmt - RedundantFieldInitializer - EmptyCatchBlock - UselessParentheses - UnusedPrivateField
대응	지적된 부분을 삭제하여 해결

문제	- AvoidFieldNameMatching
대응	변수와 메소드 이름이 같아서 생기는 문제이나, 변수는 private 으로 클래스 내부에서만 사용되고, 메소드는 클래스 외부에서만 사용되는 데다가, 메소드는 사실상 해당 private 변수를 반환하는 역할이기에 그대로 두어도 문제가 없다고 판단함

문제	- AvoidReassigningParameter
대응	<pre> public int calculate(Category category) {     HashSet&lt;String&gt; property_group = new HashSet&lt;String&gt;();     ArrayList&lt;Category&gt; category_list = new ArrayList&lt;Category&gt;();      int single_and_error = 0;     int external_count = 1;      category = category.first(); </pre> <pre> public int calculate(Category c) {     HashSet&lt;String&gt; property_group = new HashSet&lt;String&gt;();     ArrayList&lt;Category&gt; category_list = new ArrayList&lt;Category&gt;();      int single_and_error = 0;     int external_count = 1;      Category category = c.first(); </pre> <p>수신 매개변수를 수정할 우려가 있는 문제를 위와 같이 수정하여 해결</p>

## IntelliJ

문제	비어 있거나 필요 없는 코드 - Statement with empty body - Unused assignment - Unused import - Empty method
대응	지적된 부분을 삭제하여 해결

문제	- Fields can be local - Parameter can be local
대응	지역 변수로 수정하는게 낫다고 지적된 부분이나, 문제가 생겨 코드를 수정하거나 기능을 확장할 때에 쓰일 수 있는 변수들이기에 그대로 둬

문제	- Unused declaration
대응	클래스 다이어그램을 토대로 만들었거나, GUI 를 구현하기 위해 여러 시도를 하는 과정에서 만들어진 필요 없는 부분들은 제거하고, 코드를 수정하거나 확장할 때에 필요할 수 있다고 판단된 부분은 남겨둬

문제	축약 가능한 코드 - Pointless Boolean expression - Redundant 'if' statement - Redundant local variable - Duplicated code
대응	그대로 놔두는 것이 알아보고 이해하기 쉽다고 판단하여 그대로 둬

문제	- Method can be void
대응	해당 메소드의 반환 값을 void 로 수정함

문제	- Result of method call ignored
대응	메소드의 반환 값을 변수에 할당하는 과정이 생략되어 있었기에 이를 추가함

## Eclipse Metrics Plugin

문제	Calculator.java 의 internal_calculate 메소드의 복잡도가 12
대응	주요 계산 과정이 복잡하여 지적된 문제이나, 이를 더 간단히 할 방법을 찾지 못했기에 그대로 둠

문제	Value.java 의 Value 메소드의 복잡도가 12
대응	반복문 내부에 여러 if 문이 들어 있어서 생각보다 복잡도가 높게 측정된 것으로 보임

문제	Value.java 의 Value 메소드가 6 개로 많은 매개변수를 가짐
대응	이를 수정하려면 다른 클래스에서도 많은 수정이 필요하기에 이대로 둠

## Find Bugs

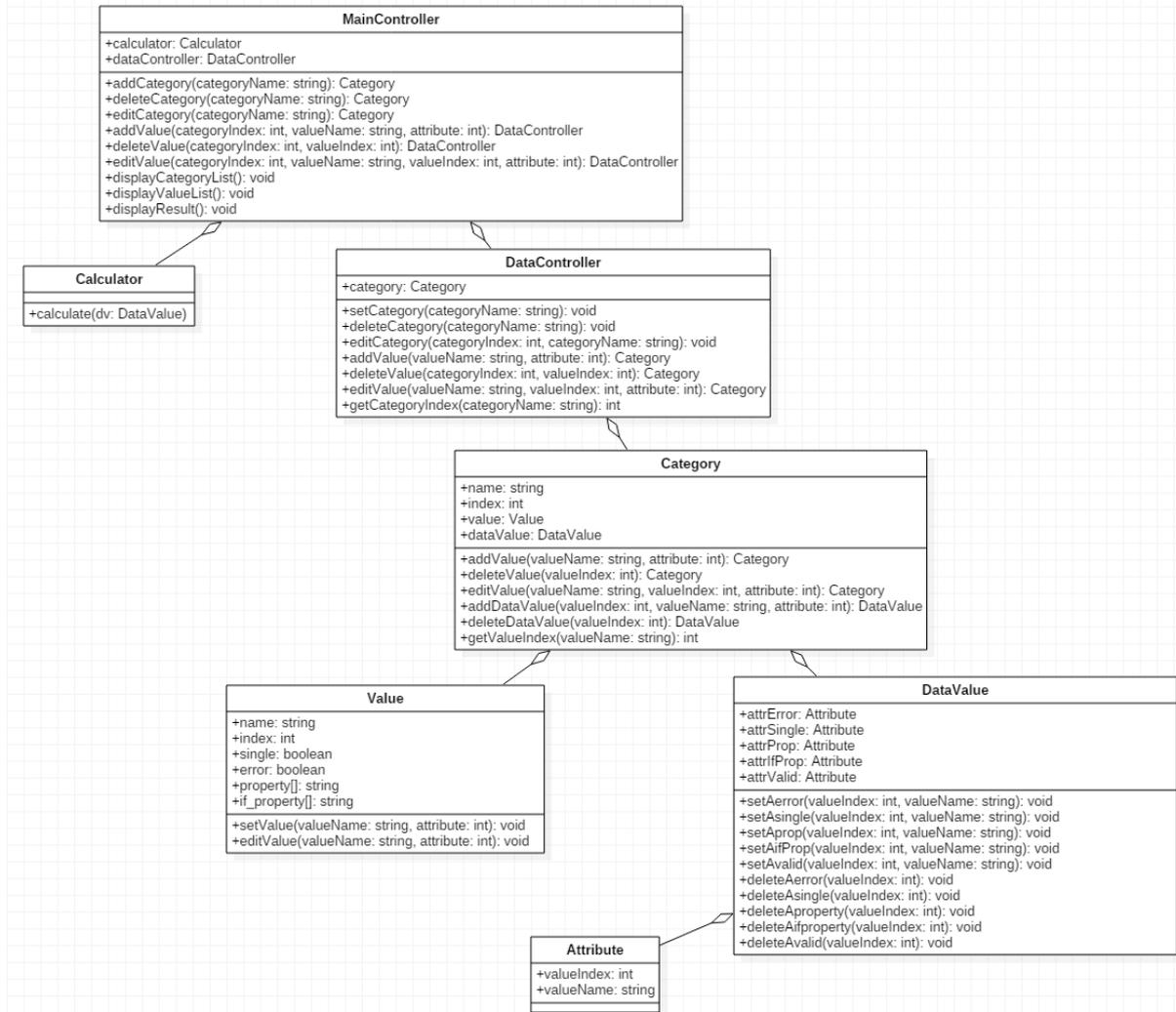
문제	- Comparison of String parameter using == or !=
대응	해당 부분을 .equals()로 수정

문제	- Return value of Model.Value.getName() ignored
대응	IntelliJ 의 'Result of method call ignored'와 동일한 문제

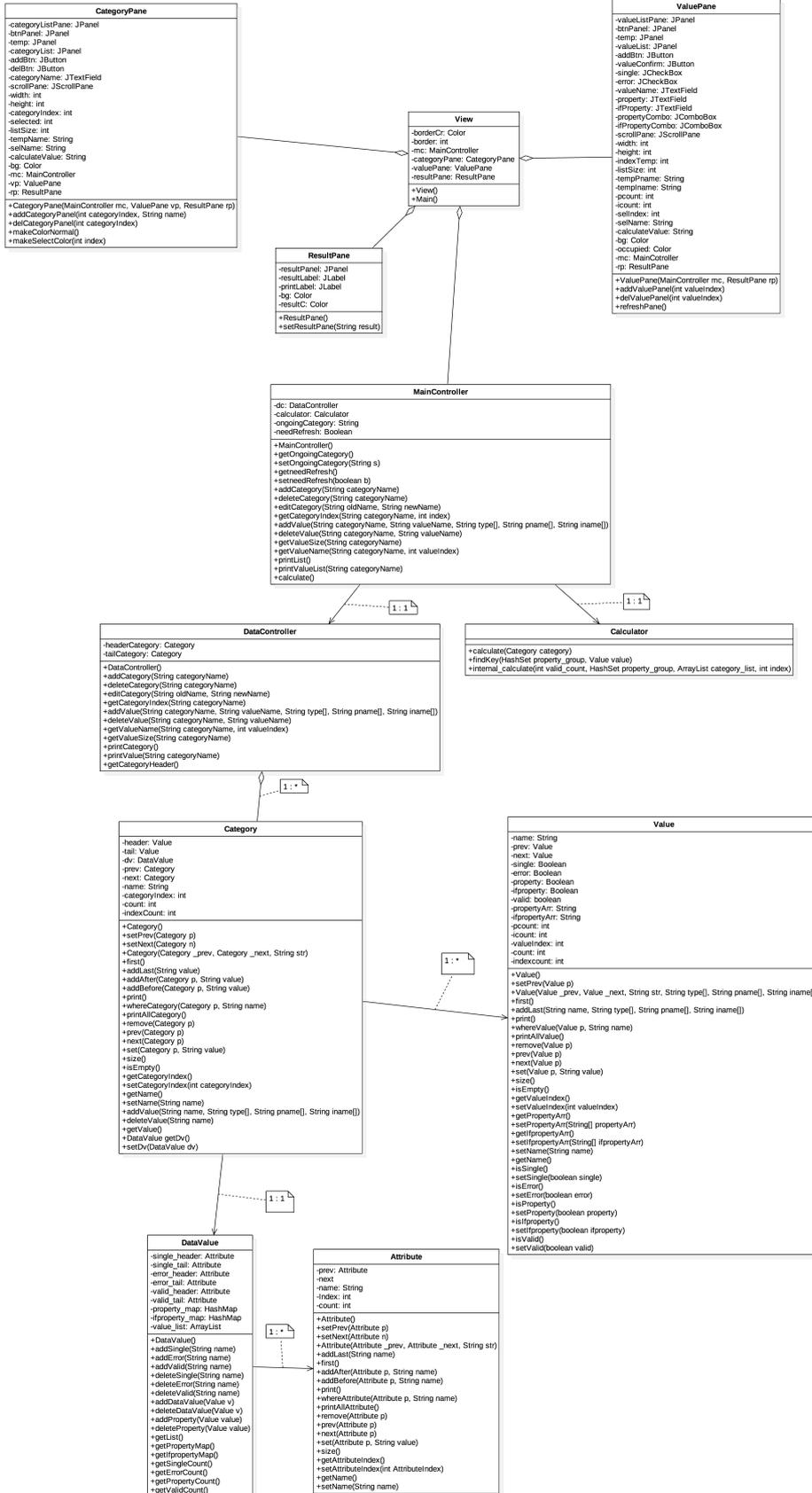
## II. System Review

- 새로운 시스템에 맞추어 새로운 클래스 다이어그램으로 변경 하였다.

### 구현 전 ClassDiagram (stage 2040)



# 구현후



### III. Realization

-이번 프로젝트를 통하여 좋은 설계를 한다는 것은 힘든 것 이며, 정확한 설계가 이루어졌을 때, 구현에 큰 영향을 끼칠 수 있다라는 것을 알게 되었습니다. 또한 팀 단위의 프로젝트를 진행 할 시에는 서로의 협력을 통한 구현을 위해서라도 프로그램의 설계는 필수적인 것을 알게 되었습니다.

-좋은 설계란 충분한 요구사항을 만들어 내는 것부터 시작되며, 설계는 요구사항을 기초로 두어 발전해나 가야 하는 것이라 생각합니다. 그렇게 진행하게 된다면, 설계 도중 엇나간 설계를 하는 것을 방지 할 수 있게 되며, 짜임새 있는 프로그램으로 발전해 나갈 수 있게 됩니다.

-좋은 설계에 가장 큰 장점으로서는 빠른 구현이 가능하게 한다는 점이며, 유지보수에도 용이 하게 된다는 것을 이번 프로젝트를 통해 깨닫게 되었습니다.

-또한 설계를 하는 동안은, 설계의 수정을 두려워 하면 안되며, 수많은 수정을 통해 설계가 진행 된다는 점을 프로젝트 진행중 알 수 있게 되었습니다.