Final Presentation

2013 spring software verification

Team 1

200711460 이상열

200711470 정재호

201111344 김재엽

201211350 박주광

Contents

Chapter 1 - CTIP

Chapter 2 - System Testing

Chapter 3 - Static Analysis

Chapter 1

CTIP

CTIP

SVN Manage

- SVN 최종 현황
  - 최종 revision에 맞게 tags폴더로 버전 관리
  - Revision 91: 1.0 – 맨 처음 제출된 프로젝트
  - Revision 120: 1.1 – System Testing 이 후 제출된 프로젝트
  - Revision 124: 1.2 – Static Analysis 프로젝트

```
papimomi@dongbanghost:~$ cd team1
papimomi@dongbanghost:~/team1$ ls
PaintGame  Project  Test  branches  src  tags  trunk
papimomi@dongbanghost:~/team1$ cd tags
papimomi@dongbanghost:~/team1/tags$ ls
1.0  1.1  1.2
papimomi@dongbanghost:~/team1/tags$
```

CTIP

Mantis

- Manti와 SVN 연동완료.
- 실질적으로 거의 사용하지 않았다는 것을 확인할 수 있다.

이슈 보기 (1 - 10 / 10) [보고서 출력] [CSV 내보내기] [Excel 내보내기]

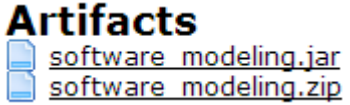
		P	ID	#		분류	중요도(심각성)	상태	업데이트됨	요약
☐			0000022			General	보통	신규	2013-06-08	각 Unit에 대한 개별적인 Coverage 비율 결과 알려주시면 감사하겠습니다
☐			0000012			General	보통	해결된 이슈 (jjh2088)	2013-06-06	화면 출력에 관한 테스팅 실패에 대한 보고입니다.
☐			0000011			General	보통	해결된 이슈 (jjh2088)	2013-06-06	게임 시작에 관련된 테스팅 실패 결과 입니다.
☐			0000021		1	General	보통	신규	2013-05-30	v2.0에 대한 Unit Test 결과입니다.
☐			0000014			General	보통	이슈 검토 (papimomi)	2013-05-24	보완사항 관련
☐			0000013			General	보통	이슈 검토 (papimomi)	2013-05-24	텍스트 파일 불러오기 오류 관련 확인부탁드려요~
☐			0000009			General	중요함	해결된 이슈 (papimomi)	2013-05-13	jdk 1.7 말고 1.6으로 개발해주세요.
☐			0000010			General	충돌	해결된 이슈 (papimomi)	2013-05-13	잘못 인코딩된 주석이 있습니다.
☐			0000008	1	1	General	보통	확인된 이슈 (papimomi)	2013-05-12	v1.0 Unit Test 결과입니다
☐			0000007			General	보통	신규	2013-05-07	확인

☐ 모두 선택 이동 확인

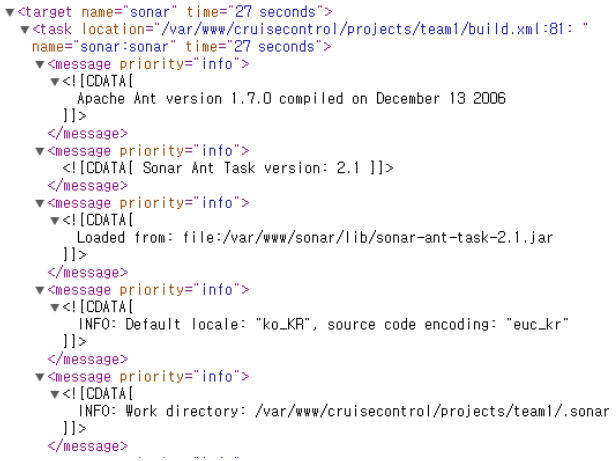
CTIP

Cruise Control

- Cruise Control과 Ant, Sonar연동 완료.
- 원활한 testing을 위해 다음과 같이 SVN에 올라온 파일을 압축파일로 관리 하였다.
- Ant와 Sonar를 연동하여 Sonar에서 적용된 것을 build log를 통해 확인 가능하다.




```
<target name="zip" depends="jar">
    <zip destfile="target/software_modeling.zip">
        <fileset dir="PaintGame">
            <exclude name="**/src/**" />
            <exclude name="**/.classpath" />
            <exclude name="**/.project" />
            <exclude name="**/.settings/**" />
        </fileset>
        <fileset dir="target" includes="software_modeling.jar" />
    </zip>
</target>
```



CTIP

Cruise Control(cont.)

- Cruise Control을 사용하여 CI의 장점을 체험할 수 있었다.
- Build.xml 조작으로 상대적으로 쉽게 여러 tool들을 연동 가능하다.
- 관리하는 프로젝트가 커지고 사용하는 tool이 많아지면 xml파일 수정이 어려워질 수 있다.
- 해당 tool에 대한 이해가 가장 중요하다.
- 각 tool마다 JDK버전 지원 범위 파악해야 한다.

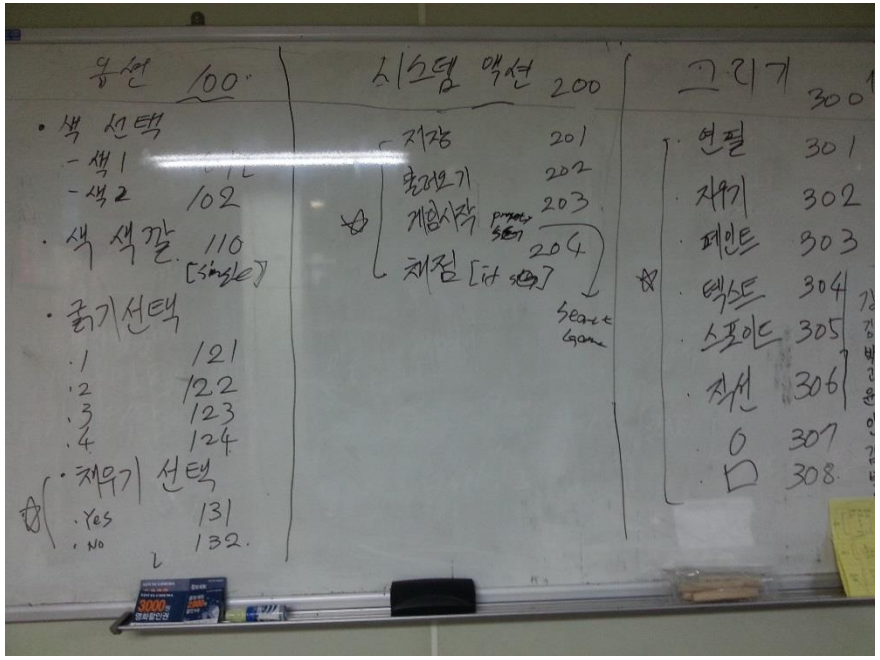
Chapter 2

System testing

System Testing

Category Partition(cont.)

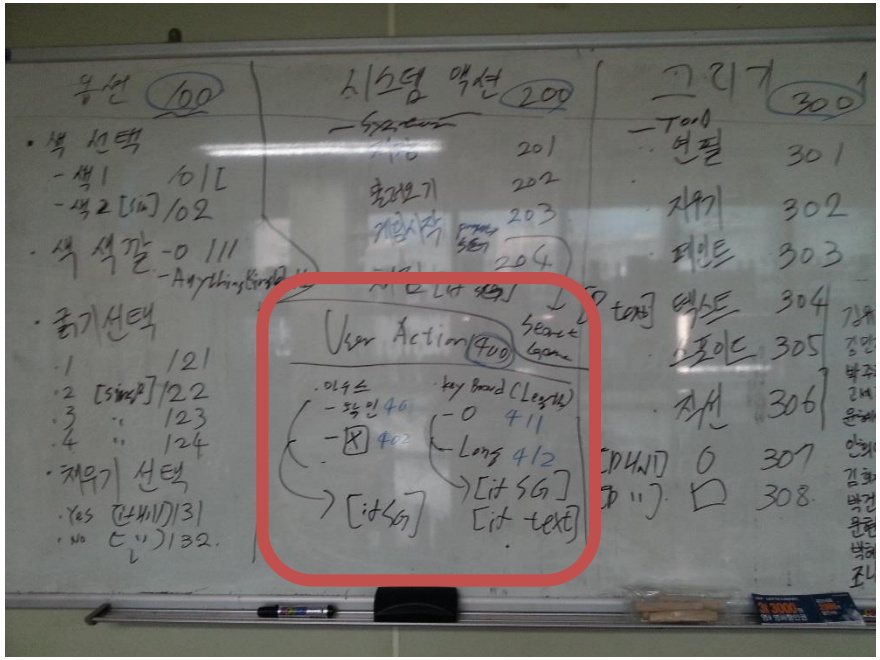
- 다음과 같이 화이트보드를 활용하여 Category를 정리하였다.
- 맨 처음 정의한 Category로 다음 category에 따라 test를 진행하였다.
- Test를 거치지 않는 부분이 발생하여 category를 수정하였다.
- User Action이라는 새로운 Category를 정의하여 삽입하였다.



System Testing

Category Partition(cont.)

- Mouse Action과 Keyboard액션 등 사용자의 선택에 따라 프로그램의 변화를 반영하였다.
- [Single]과 [Property] 재정의하였다.
- Category를 정의하기 위해서 어디에 초점을 맞추느냐에 따라 좋은 Category와 나쁜 Category가 나올 수 있다.
- Test Case를 개수를 구할 때는 100개 이하라면 손으로 하는 것이 더 빠르다.



System Testing

Category Partition(cont.)

Date	Test Case#	Revision
05/24	203.411.401.*	Yes
05/24	203.412.401.*.204.*	Yes
05/24	203.412.401.*.204.402	Yes
05/24	203.412.401.*.204.401.304.412	Yes
05/24	304.412.203.412.401.*	Yes
06/07	Brute Force test #1	No
06/07	Brute Force test #2	No
06/07	Brute Force test #3	Yes

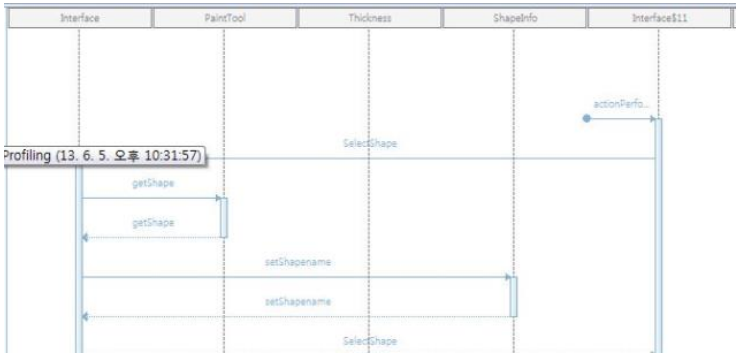
Chapter 2

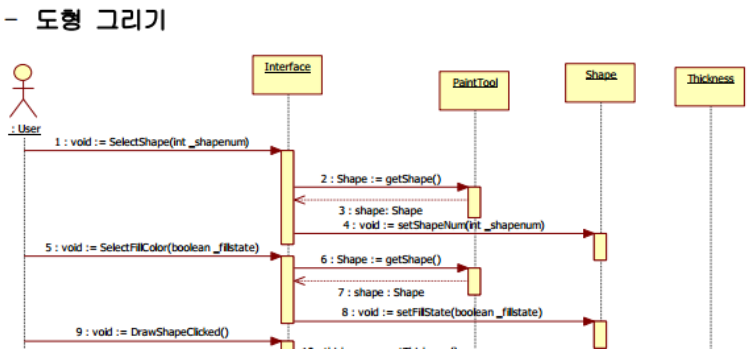
Static Analysis

Static Analysis

Eclipse TPTP

- 프로젝트 결과물의 Sequence Diagram 표현
- 디자인상에서 만들어진 Sequence Diagram 과 비교가 가능하다.
- Eclipse의 모든 버전과 호환되지 않고 최신 버전이 heilos SR2에서만 사용이 가능하였다.

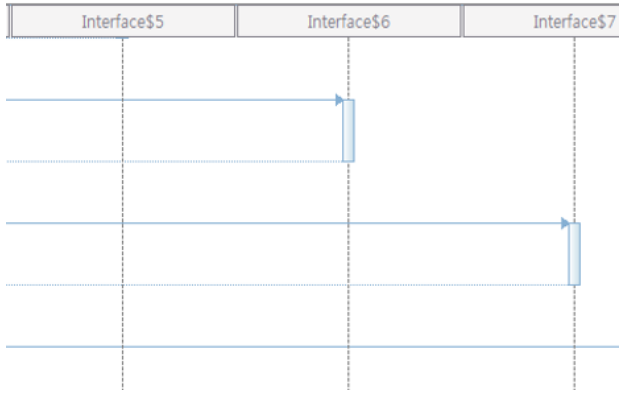


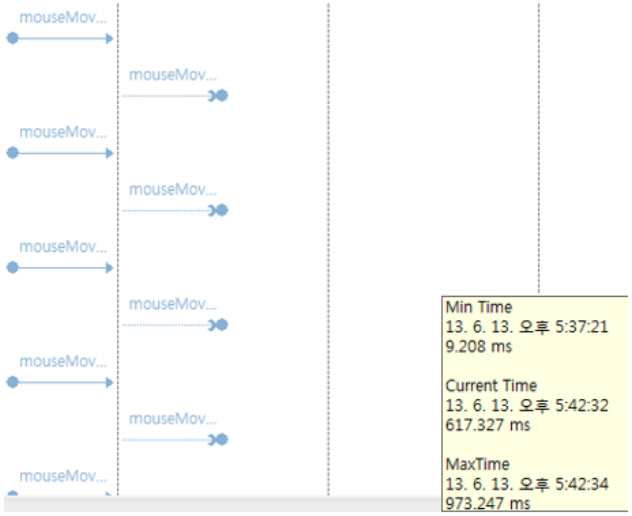


Static Analysis

Eclipse TPTP(cont.)

- 마우스 입력이나 키보드 입력과 같은 프로그램 외적인 부분까지 잡아주고 그 부분들을 Interface #로 표현한다.
- 또한, 시간 단위를 ms단위로 표시한다.
- 따라서 결과물을 보고 원하는 결과물을 찾아주기 위해서 해석이 필요하다.

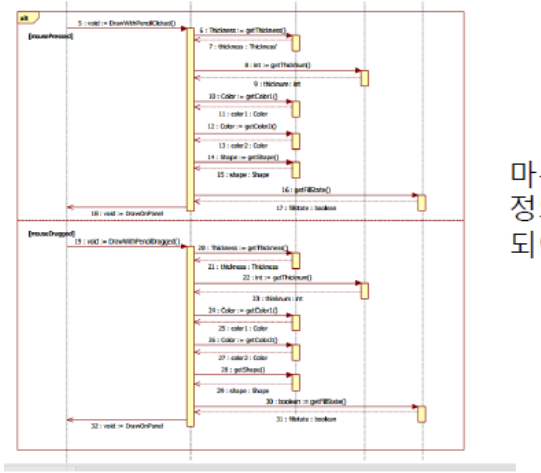




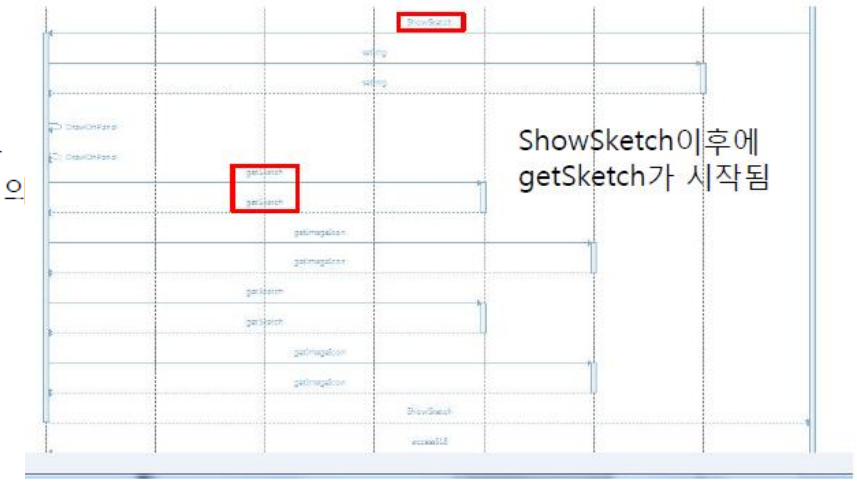
Static Analysis

Eclipse TPTP(cont.)

- 다음과 같이 TPTP결과 분석을 토대로 문서와 일치하지 않는 부분에 대해서 report하였다.



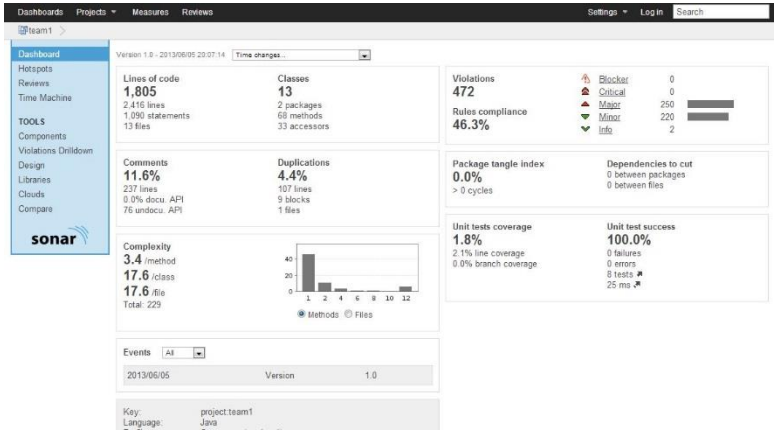
마우스 press와 dragged 부분만 정의되어 있고 release부분은 정의되어 있지 않음



Static Analysis

SONAR

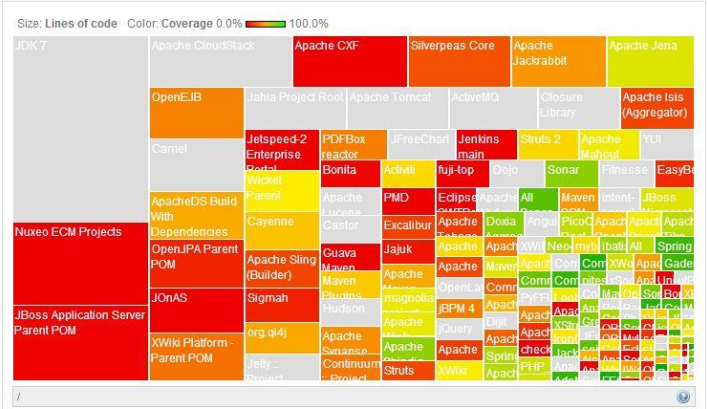
- 다음과 같이 다양한 정보를 한 눈에 파악이 가능하다.
- Coverage나 Dependency도 다른 tool과 연동하여 보여줄 수 있다.
- 원하는 정보를 선택하여 좀 더 상세한 정보들을 볼 수 있다.

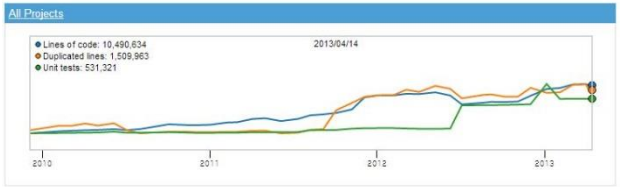


Static Analysis

SONAR(cont.)

- 다음과 같이 다른 프로젝트와 비교가 가능하다.
- 시간 순서에 따른 프로젝트의 변화를 확인할 수 있다.
- Analysis에 필요한 요소들은 아니지만, 개발자의 사기를 북돋아 주거나 프로젝트 개발 진행상황을 기록으로 남길 수 있다.





Static Analysis

SONAR(cont.)

- 다음과 같이 다양한 정보를 한 눈에 파악이 가능하다.
- Coverage나 Dependency도 다른 tool과 연동하여 보여줄 수 있다.
- 원하는 정보를 선택하여 좀 더 상세한 정보들을 볼 수 있다.

II. Duplication

i. D001

이 Duplication의 경우 주로 도형과 관련된 class의 변수를 설정하거나 출력하는 부분으로 보인다. 따라서 각 도형 클래스 내부에 해당 속성들의 설정해주는 Method를 만들어주거나, 또는 interface 클래스에 해당 도형의 속성을 설정해주는 Method를 만들어 주는 방법들이 있을 것이다. 다만, Method를 만들어 줌으로써 가독성이 떨어지거나 복잡도가 올라갈 수 있으므로, 개발자에 재량에 따라 적절한 방법을 취할 것을 권유한다.

ii. D002

D001의 Duplication과 마찬가지로 비슷한 동작을 하는 부분이 잡혀있다. 위와 같은 방법으로 해결할 수 있다.

iii. D003

D001의 Duplication과 마찬가지로 비슷한 동작을 하는 부분이 잡혀있다. 위와 같은 방법으로 해결할 수 있다.

III. Coverage

i. Coverage가 1.8%밖에 안되어 coverage가 상당히 낮아 보이지만, 실질적으로 accessor(setter, getter)는 굳이 Unit test를 거치지 않아도 괜찮으며, 주석이나 변수 선언부 또한 coverage비율에 포함되므로 coverage가 낮게 나오는 것은 큰 문제가 아니라고 볼 수 있다. 다만, PaintGame.java의 public void CalculateScore(BufferedImage d_img, String filename)의 경우 System testing에서도 정확한 수치를 확인할 방법이 없었으므로 unit test를 한 번 거쳐보는 것이 좋을 것 같다.

IV. Violation

i. Violation의 경우 프로젝트가 실행되는 logic을 고려하지 않고, 단순히 code standard를 지키기 위해 제시된다. 따라서, 실제로 프로그램이 어떻게 돌아가는 지에 따라 violation rule어기게 되더라도 큰 문제는 되지 않는다고 볼 수 있다. 다만, 유지, 보수 및 source code공유를 고려 한다면, V001이나 V004와 같은 부분들도 고려해보아야 한다. 밑에 나열하는 violation은 다음과 같은 취지에서 선정해본 것들이다.

ii. V006, V011, V023: 다음 violation들은 보안과 관계된 부분이므로 수정해볼 수 있다.

iii. V012, V018, V021: 다음 violation들을 지킴으로써 프로그램의 효율성을 올릴 수 있으므로 수정해볼 수 있다.

Q & A

Thank You!!