SOFTWARE VERIFICATION

2016-06-09

2nd TESTING – SYSTEM TEST

유 준 범 교수님

TEMA 5

201214262 라 가 영
201313250 서 지 혁

Contents

1. Specification Review

- **OOPT Stage 1000 ver.6**
  - Performance Requirements

    “A. 소스 코드 파일 분석 작업은 5초 이내로 수행되어야 한다.”

    “B. 기준 소스코드 파일 변경 시 2초 이내로 비교 결과가 변경되어야 한다.”

    대응서에는 100개 이하의 파일이 들어있는 폴더를 선택해야 한다고 기술되어 있지만

    문서에는 적지 않음.

1. Specification Review

- **OOPT Stage 2030 ver.4**

  - 2131 Start
    대응서에서는 “소스 코드 파일의 크기는 무관하다”
    문서에서는 “0바이트인 파일은 안된다”

  - 2136 Change Center
    프로그램 시행 시 선택되는 첫 번째 파일이 파일명의 오름차순 인것 인지, 표현이 모호함.

1. Specification Review

- **OOPT Stage 2040 ver.3**

  - 2141

    프로그램 실행화면 이미지가 1005와 서로 다른 버전의 프로그램 이미지로 삽입되어 있어 일치하지 않는다.

  - 2141 Change Annotation

    C 스타일 주석의 /*은 //로 치환한다고 기술되어 있지만,

    */은 어떻게 처리 할 것인지 기술되어 있지 않음.

2. Category-Partition Testing

- **Test Specification Language**
  - Ostrand와 Balcers의 1988 CACM 논문에서 Category-Partition Testing을 위해 정의한 도메인 특정언어 TSL(Test Specification Language)를 사용
  - 테스트 카테고리와 선택지를 명세

```
Line:
    Lines:
        Zero. [single]
        Positive.

Annotation:
    Comments:
        Zero.
        Positive. [property ContainsComment]
    Style:
        Single-line. [if ContainsComment]
        Multi-line. [if ContainsComment] [property MultiLineComment]
    Matching:
        True. [if MultiLineComment]
        False. [if MultiLineComment]

Function:
    Functions:
        Zero.
        Positive.

Variable:
    Variables:
        Zero.
        Positive.

Preprocessor:
    Includes:
        Zero.
```

2. Category-Partition Testing

- **Test Frame Generation**
  - Category-Partition 테스트 명세를 TSL로 작성
  - tslgenerator를 사용하여 테스트 프레임 생성을 자동화

```
Test Case 1                    <single>
   Lines :   Zero


Test Case 2                    (Key = 2.1.0.0.1.1.1.1.)
   Lines          :  Positive
   Comments       :  Zero
   Style          :  <n/a>
   Matching       :  <n/a>
   Functions      :  Zero
   Variables      :  Zero
   Includes       :  Zero
   Defines        :  Zero


Test Case 3                    (Key = 2.1.0.0.1.1.1.2.)
   Lines          :  Positive
   Comments       :  Zero
   Style          :  <n/a>
   Matching       :  <n/a>
   Functions      :  Zero
   Variables      :  Zero
   Includes       :  Zero
   Defines        :  Positive


Test Case 4                    (Key = 2.1.0.0.1.1.2.1.)
   Lines          :  Positive
   Comments       :  Zero
```

3. Pairwise Testing

- **Test Modelling**
  - PICT는 Miscrosoft에서 개발한 Pairwise 테스트 케이스 생성 도구
  - PICT가 정의 하는 모델 포맷으로 테스트 모델링

```
Lines: Zero, Positive
Comments: Zero, Positive
Style: Single-line, Multi-line
Matching: True, False
Functions: Zero, Positive
Variables: Zero, Positive
Includes: Zero, Positive
Defines: Zero, Positive
```

3. Pairwise Testing

- **Test Case Generation**
  - PICT모델로부터 테스트 케이스 생성

Lines	Comments	Style	Matching	Functions	Variables	Includes	Defines
Positive	Positive	Single-line	False	Positive	Zero	Zero	Zero
Zero	Zero	Multi-line	True	Zero	Positive	Positive	Zero
Zero	Zero	Single-line	False	Zero	Positive	Zero	Positive
Positive	Positive	Multi-line	True	Zero	Zero	Positive	Positive
Zero	Positive	Multi-line	False	Positive	Positive	Positive	Positive
Positive	Zero	Single-line	True	Positive	Positive	Zero	Zero
Zero	Zero	Single-line	True	Zero	Zero	Positive	Positive
Positive	Positive	Multi-line	True	Positive	Zero	Zero	Positive

3. Pairwise Testing

- **Test Results**
  - java.lang.NullPointerException 에러로 인하여 테스트가 진행되지 않음.

4. Static Analysis

- **Eradicate**

```
$ infer -analyzer eradicate  -gradle jar
Running and capturing gradle compilation...
Starting analysis (Infer version v0.8.1)
FFFFF.....................................................
Analyzed 5 files

Found 19 issues

src/main/java/Analyze.java:16: error:
   ERADICATE_FIELD_NOT_INITIALIZED
  Field 'Analyze.tempFolderPath' is not initialized in
     the constructor and is not declared '@Nullable'
  14.             ArrayList<String> listType = new
     ArrayList();
  15.
  16. >           public Analyze(ArrayList<String> source){
  17.                    this.source = source;
  18.                    this.listType.add("void");
  19.
```

4. Static Analysis

- **Infer**

```
$ infer -analyzer infer  -gradle jar
Running and capturing gradle compilation...
Starting analysis (Infer version v0.8.1)
FF..FF.F.....................................................
    ............................
Analyzed 5 files

Found 1 issue

src/main/java/Controller.java:48: error: NULL_DEREFERENCE
  object fileList last assigned on line 46 could be null
     and is dereferenced at line 48
  46.                     File [] fileList = dir.listFiles
     ();
  47.
  48. >                   for(File f : fileList)
  49.                     {
  50.                             if(!f.isHidden() && f.
     getName().endsWith(".c"))
  51.
```

4. Static Analysis

- **Tracing**

```
$ infer -analyzer tracing  -gradle jar
Running and capturing gradle compilation...
Starting analysis (Infer version v0.8.1)
FFFF..F.....................................................
    ............................
Analyzed 5 files

Found 3 issues

src/main/java/Calculate.java:24: warning:
    PRECONDITION_NOT_MET
  in call to calLine() at line 24
  22.                     for(fileNum=0;fileNum<this.
     numOfFile;fileNum++)
  23.                     {
  24. >                           calLine();
  25.                             calFunction();
  26.                             calVariable();
  27.
```

4. Static Analysis

- **PMD - Design**

```
$ pmd pmd -dir src/main/java/ -R java-design
Analyze.java:1: Possible God class (WMC=66, ATFD=19, TCC
    =0.0909090909090091)
Analyze.java:62:        Deeply nested if..then statements
     are hard to read
Calculate.java:10:      Perhaps 'numOfFile' could be
    replaced by a local variable.
Controller.java:1:      Possible God class (WMC=63, ATFD
    =204, TCC=0.05789473684210526)
Controller.java:21:     Private field 'files' could be
    made final; it is only initialized in the declaration
    or constructor.
Controller.java:40:     Overridable method 'DisplayMain'
    called during object construction
Controller.java:99:     Avoid unnecessary comparisons in
    boolean expressions
Controller.java:257:    Avoid unnecessary comparisons in
    boolean expressions
Controller.java:327:    New exception is thrown in catch
    block, original stack trace may be lost
```

4. Static Analysis

- **PMD - Optimization**

```
$ pmd pmd -dir src/main/java/ -R java-optimizations
Analyze.java:6: Avoid using redundant field initializer
    for 'numOfLine'
Analyze.java:7: Avoid using redundant field initializer
    for 'numOfFunction'
Analyze.java:8: Avoid using redundant field initializer
    for 'numOfVariable'
Analyze.java:9: Avoid using redundant field initializer
    for 'numOfPreprocessor'
Analyze.java:10:        Avoid using redundant field
    initializer for 'numOfAnnotation'
Analyze.java:16:        Parameter 'source' is not
    assigned and could be declared final
Analyze.java:28:        Parameter 'file' is not assigned
    and could be declared final
Calculate.java:8:       Avoid using redundant field
    initializer for 'centerNum'
Calculate.java:9:       Avoid using redundant field
    initializer for 'fileNum'
Calculate.java:10:      Avoid using redundant field
```

4. Static Analysis

- **PMD - Imports**

```
$ pmd pmd -dir src/main/java/ -R java-imports
Controller.java:7:      Avoid duplicate imports such as '
    java.io.BufferedReader'
Controller.java:8:      Avoid duplicate imports such as '
    java.io.File'
Controller.java:9:      Avoid duplicate imports such as '
    java.io.FileNotFoundException'
Controller.java:10:     Avoid duplicate imports such as '
    java.io.FileReader'
Controller.java:11:     Avoid duplicate imports such as '
    java.io.IOException'
Controller.java:12:     Avoid importing anything from the
     package java.lang
Controller.java:198:    Unnecessary use of fully
    qualified name 'java.io.File' due to existing import '
    java.io.*'
```

4. Static Analysis

- **PMD - Unnecessary**

```
$ pmd pmd -dir src/main/java/ -R java-unnecessary
Analyze.java:84:        Useless parentheses.
Calculate.java:100:     Useless parentheses.
Calculate.java:103:     Useless parentheses.
Calculate.java:170:     Useless parentheses.
Calculate.java:173:     Useless parentheses.
Calculate.java:236:     Useless parentheses.
Calculate.java:239:     Useless parentheses.
Controller.java:371:    Useless parentheses.
Controller.java:379:    Useless parentheses.
Controller.java:435:    Useless parentheses.
Controller.java:514:    Useless parentheses.
```

4. Static Analysis

- **PMD - String**

```
$ pmd pmd -dir src/main/java/ -R java-strings
Analyze.java:84:        String.indexOf(char) is faster
    than String.indexOf(String).
Analyze.java:85:        String.indexOf(char) is faster
    than String.indexOf(String).
Analyze.java:219:       String.indexOf(char) is faster
    than String.indexOf(String).
Analyze.java:222:       String.indexOf(char) is faster
    than String.indexOf(String).
Analyze.java:229:       String.indexOf(char) is faster
    than String.indexOf(String).
Analyze.java:256:       The String literal "struct"
    appears 5 times in this file; the first occurrence is
    on line 256
Controller.java:143:    The String literal "Malgun Gothic
    " appears 8 times in this file; the first occurrence
    is on line 143
Controller.java:238:    String.indexOf(char) is faster
    than String.indexOf(String).
```

4. Static Analysis

- **FindBugs - Results**

```
$ findbugs -textui build/libs/clone-checker.jar
M P Dm: Calculate.calFunction() invokes inefficient new
    String(String) constructor  At Calculate.java:[line
    85]
M P Dm: Calculate.calVariable() invokes inefficient new
    String(String) constructor  At Calculate.java:[line
    147]
M P Dm: Calculate.calPreprocessor() invokes inefficient
    new String(String) constructor  At Calculate.java:[
    line 221]
M D NP: Possible null pointer dereference in Controller.
    mkFileInstance() due to return value of called method
     Dereferenced at Controller.java:[line 48]
M B Nm: The method name Controller.DisplayMain() doesn't
    start with a lower case letter  At Controller.java:[
    lines 119-189]
M B Nm: The method name Controller.DisplayResult() doesn'
    t start with a lower case letter  At Controller.java:[
    lines 213-289]
H I Dm: Found reliance on default encoding in Controller.
```

5. References

[1] Junbeom Yoo, Eui-Sub Kim, *Software Verification – Introduction to Software Testing & Static Analysis*, http://dslab.konkuk.ac.kr/Class/2016/16SV/16SV.htm, 2016.

[2] Junbeom Yoo, Sejin Jung, *Software Modeling & Analysis*, http://dslab.konkuk.ac.kr/Class/2016/16SMA/16SMA.htm, 2016.

[3] Junbeom Yoo, Jaeyoeb Kim, *Programming Project*, http://dslab.konkuk.ac.kr/Class/2016/16PP/16PP.htm, 2016.

[4] T. J. Ostrand, M. J. Balcer, *The category-partition method for specifying and generating functional tests*, 1988.

[5] Alex Orso, *TSL Generator for the Category Partition Method*, https://github.com/alexorso/tslgenerator

[6] Jacek Czerwonka, *Pairwise Testing*, http://www.pairwise.org

[7] Microsoft Corporation, *Pairwise Independent Combinatorial Tool*, https://github.com/Microsoft/pict

[8] Jacek Czerwonka, *Pairwise Testing in the Real World: Practical Extensions to Test-Case Scenarios*, https://msdn.microsoft.com/en-us/library/cc150619.aspx, 2008.

5. References

[1] Junbeom Yoo, Eui-Sub Kim, *Software Verification – Introduction to Software Testing & Static Analysis*, `http://dslab.konkuk.ac.kr/Class/2016/16SV/16SV.htm`, 2016.

[2] Junbeom Yoo, Sejin Jung, *Software Modeling & Analysis*, `http://dslab.konkuk.ac.kr/Class/2016/16SMA/16SMA.htm`, 2016.

[3] Junbeom Yoo, Jaeyoeb Kim, *Programming Project*, `http://dslab.konkuk.ac.kr/Class/2016/16PP/16PP.htm`, 2016.

[4] Facebook, *Infer*, `http://fbinfer.com` 2016.

[5] PMD, *PMD*, `https://pmd.github.io`, 2016.

[6] Bill Pugh, Andrey Loskutov, *FindBugs*, `http://findbugs.sourceforge.net`, 2016.

감사합니다.