SMA:
Software Modeling and Analysis

Practical Session

Week 08

Assignment 07

Discussion

A07 - Exercise 01 | Code metrics in theory

General *knowledge*.

a) What is the cyclomatic complexity? Explain!

M = E - N + 2P

M is metric

E are the CFG edges (potential execution flows)

N are the nodes (instructions)

P is amount of connected components (1 for now)

Benefits: reports complexity of code, easy to apply

Drawbacks: simplifies real world

A07 - Exercise 01 | Code metrics in theory

General *knowledge*.

b) Which other metrics do you know?
LOC, TIME, BUGS, SIZE, ...

c) Do metrics always express problems?
No! They often lack context.

d) How and when are nowadays metrics integrated into development processes?
Metrics are used throughout the whole development life-cycle.
Development: IDE plug-in
Build process: automated verification during build
Release: evaluation of customer feedback

A07 - Exercise 02 | Simple code metrics in practice

Writing **code.**

Find all classes that have > 100 methods in `modelArgo`.

```
modelArgo allModelClasses select: [ :each |
    each numberOfMethods > 100 ].
```

A07 - Exercise 03 | Advanced code metrics in practice

***Writing* code and *interpretation* of the results.**

a) Find all methods in `modelArgo` that have:

1) > 150 lines of code, and

2) an acyclomatic complexity of < 4

```
modelArgo allModelMethods select: [ :each |
   (each numberOfLinesOfCode > 150) and:
      [each cyclomaticComplexity < 4 ] ]
```

A07 - Exercise 03 | Advanced code metrics in practice

***Writing* code and *interpretation* of the results.**

b) Apply your implementation to `modelSolr`.
Which differences can you see in the result?
ArgoUML: many factory methods
Solr: many complex test setups

c) Is it appropriate to use the same thresholds for any models?
Justify!

Yes, because thresholds are legitimate for most scenarios.
Exceptions: generated code, ...

A07 - Exercise 04 | Expert code metrics in practice

Writing **code.**

Add a method to `FAMIXType` to obtain the ATFD metric for its instances.

```
atfd
^ ( (self queryAllOutgoingInvocations opposites
    reject:  [ :each | each parentType = self ])
    select:  [ :each |
        (each name beginsWith:  'set') or:
        [ each name beginsWith:  'get'] ] ) size.
```

Assignment 08

Preview

A08 - Exercise 01 | Code smells

a) Choose two different code smells and explain them. (2 pts)

b) What is the fundamental problem in developers bad code smell perception? (1 pt)

c) What is “association rule mining” in the context of HIST? (1 pt)

A08 - Exercise 02 | Test code smells

a) Choose one test code smell and explain it. (1 pt)

b) Find and explain the test code smell in the test below. (2 pts)

```
public void testDataIsVariable() throws Throwable {
   JSTerm term = new JSTerm();
   term.makeVariable();
   term.add((Object) "");
   jSTerm0.matches(jSTerm0);
   assertEquals(false, term.isGround());
   assertEquals(true, term.isVariable());
}
```

A08 - Exercise 03 | Detection of eager tests

Extract all `JUnit3` tests from `modelWeka` that suffer from the “Eager Test” code smell.

➔ Find every method with `#isJUnit3Test` set to `true` that contains at least two assertion statements. (3 pts)