UNIVERSITY OF AARHUS

Faculty of Science

Department of Engineering

Eksamensdispositioner Softwaretest

Bjørn Nørgaard
IKT
201370248
bjornnorgaard@post.au.dk

Joachim Andersen
IKT
20137032
joachimdam@post.au.dk

Sidste ændring: June 17, 2016, kl. 16:10

LaTeX-koden kan findes her:
https://github.com/BjornNorgaard/I4SWT/tree/master/Eksamen/Disposition

Todo list

Indholdsfortegnelse

List of Figures

1 Software Quality Metrics

Citat fra Troels: *"Hvad kan man finde ud af om sit program, uden at køre det? Denne disciplin kaldes Static Analysis, og kan give feedback til programmøren, der er ligeså værdifuld som en faktisk test."*

Statisk analyse kan bruges til:

1. Give *quality measures*.
2. Gennemtvinge *code style and decipline*.
3. Finde *possible errors*.

1.1 Værktøjer

til formålet har vi nogle værktøjer.

1.1.1 FxCop

Statisk analyseværktøj, udviklet af Microsoft til Visual Studio. Lavet for at øge performance, sikkerhed og design.

1.1.2 Compilers

Compileren er et eksempel på et Statisk analyseværktøj. Alle moderne compilere viser advarseler og lignende for koden, nogle vigtige, andre not so much.

1.2 Quantitative code analysis

Aka: Software Metrics. Bruges til at beskrive:

- Lines of code per function/module.
- Cyclomatic complexity (se afsnit 1.2.2).
- Microsoft Maintainability index.

1.2.1 Maintainability

Et index som beskriver hvor holdbar og effektiv ens kode er. Samt hvor let den er at vedligeholde. Indexet beregnes på følgende måde.

1.2.2 Cyclomatic Complexity

Bruges til at beskrive kompleksiteten i et program. Det er et kvantitativt mål for antallet af lineære uafhængige vejen gemmet programmets source code.

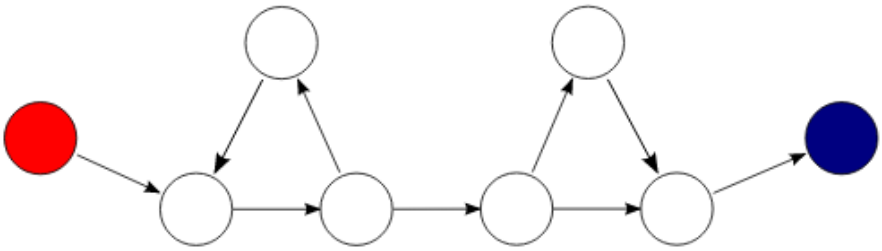

Figure 1: Grafisk fremstiling af den cyclomatiske complexity i code listing 1.

```
1 public void Method() {
2       while(Condition1)
3             Action1();
4       if(Condition2)
5             Action2();
6       Action3();
7       return;
8 }
```

Code listing 1: Kode for eksempel vist på figur 1.

Når man har en graf som vist på figur 1, er følgende udtryk gældende:

$$E = NumberOfEdges \tag{1}$$

$$N = NumberOfNodes \tag{2}$$

$$P = NumberOfConnectionComponents \tag{3}$$

Med disse termer kan M (antallet af *decision points*) findes sådan her:

$$M = E - N + 2P \tag{4}$$

$$M = 9 - 8 + 2 * 1 = 3 \tag{5}$$

2 Test-Driven Development (TDD)

3 Fakes

4 Unit-tests

5 Testkvalitet, Coverage og BVA

6 Integrationstest

7 Continuous Integration