Software Maintenance

Impact and Feature Location

Jeppe Stenstrup Lauridsen

September 22, 2021

1 Concept Location

1.1 Prefactoring

Refactor sourcecode without change the bahavior of the program

If codesmell is discovered, refactoring will be performed.

It is also good practice to split of large classes with lots of responsibility (God Classes), into smaller classes to seperate the responsibility.

1.2 Postfactoring

When going over the newly written code, it is important to go though it and eleminate anti-patterns, and lastly clean the code for any mistakes or redundancies.

1.3 Verification

Verifying the newly written code with tests guarantees the correctness of the code.

1.3.1 Testing

There are three main tests to be performed:

- Functional
- Unit
- Structual

A concept of development is **Test First development**. This is a technique where a test-specification is written, and the program is coded to match the output/rules of the specification.

2 Conclusion

When programming, commit often. When changes has been made that works, commit. This allows for much clearer version control.
When everything is commited and pushed, it's time to lay out the plan for the next change / sprint.

3 Partial code comprehension

Partial code comprehension often appears in larger systems where you have no prior knowledge.

3.1 Concept Location

Concept location is a method where you decide which tool/class you need to edit, and then manually goes through all it's dependencies, mapping out what future changes will affect throughout the system.

3.2 Concept Triangle

The concept triangle is to take map out concepts in part, ie. $<< name >>$, $<< intention >>$ and $<< extension >>$

3.3 Concept Location Methods

- Human knowledge
  - Understandin of the programs domain
- Traceability Tools (Featureous, etc)
- Dynamic Search (Execution traces)
  - Run and go through all features and let the tool map accordingly
- GREP Search
  - Global Regular Expression Print