Test Driven Developement-menetelmän tehokkuus ja laatu

Petri Pihlajaniemi

Kandidaatintutkielma
HELSINGIN YLIOPISTO
Tietojenkäsittelytieteen laitos

Helsinki, 11. helmikuuta 2015

HELSINGIN YLIOPISTO — HELSINGFORS UNIVERSITET — UNIVERSITY OF HELSINKI

Tiedekunta — Fakultet — Faculty	Laitos — Institution — Department
Matemaattis-luonnontieteellinen	Tietojenkäsittelytieteen laitos

Tekijä — Författare — Author
Petri Pihlajaniemi

Työn nimi — Arbetets titel — Title

Test Driven Developement-menetelmän tehokkuus ja laatu

Oppiaine — Läroämne — Subject
Tietojenkäsittelytiede

Työn laji — Arbetets art — Level	Aika — Datum — Month and year	Sivumäärä — Sidoantal — Number of pages
Kandidaatintutkielma	11. helmikuuta 2015	2

Tiivistelmä — Referat — Abstract

Tiivistelmä.

Avainsanat — Nyckelord — Keywords

Säilytyspaikka — Förvaringsställe — Where deposited

Muita tietoja — Övriga uppgifter — Additional information

Sisältö

1 Johdanto

Ohjelmistotestauksen tarkoituksena on parantaa ohjelman laatua havaitsemalla ja poistamalla virheitä ohjelmakoodissa. Testauksen avulla ei voida todistaa ohjelman olevan virheetön, mutta sillä voidaan todistaa virheiden olemassaolo. Ohjelmaa testataan erilaisilla syötteillä, jonka jälkeen ohjelman toimintaa verrataan odotettuun oikeaan lopputulokseen. Jos lopputuloksissa on eroa, on testi löytänyt virheen. [Muc08].

Testien kirjoittajan on tunnettava ohjelman rakenne ja toiminta pystyäkseen suunnittelemaan ja kirjoittamaan testejä. Testin kirjoittamiseni on vaikea ja aikaa vievä prosessi, ja se vaatii kehittäjältä hyviä taitoja. [Whi00]

Ohjelmiston testaus suoritetaan eri vaiheissa riippuen valitusta ohjelmistokehitysmenetelmästä. Vuonna 1970 Winston W. Roycen määrittelemä *vesiputousmalli* (Waterfall Model) kuvaa ohjelmistokehityksen viisivaiheisena prosessessina: ensin analysoidaan vaatimukset ja suunnitellaan koko ohjelma tarkasti, sitten kirjoitetaan varsinainen ohjelmakoodi ja lopuksi tuotettu ohjelmakoodi testataan. Vaiheittaisissa menetelmissä (Incremental Model) edellinen vesiputousmalli on jaettu useisiin pienempiin palasiin; koko ohjelmaa ei siis tarvitse suunnitella ja toteuttaa kerralla noudattaen vesiputousmallin järjestystä, vaan ohjelman rakentuu pienemmistä osista jotka on suunniteltu ja toteutettu erikseen. *Stoica et al* mukaan testaus on helppoa inkrementaalisissa menetelmissäi, kun taas vesiputousmallissa testauksessa havaittujen virheiden korjaaminen voi olla hankaa. Ketterät (Agile Model) ohjelmistokehityksen menetelmät perustuvat inkrementaaliseen malliin. [SMGM13]. Yksi erityisesti testaamiseen keskittyvät toimintapa on ketteriin malleihin perustuva *Test Driven Development*. [Cri06]

Aion tutkielmassani tarkastella TDD:n toimivuutta kahdella mittarilla: laadulla ja tehokkuudella. Nämä mittari ovat erittäin epämääräisiä, joten ensin on tutkittava mitä ne oikeastaan tarkoittavat.

2 Ohjelmistotestauksen mittareita

2.1 Laatu

2.2 Tehokkuus

3 Test Driven Development

3.1 Mitä se on? Miten eroaa muista?

3.2 Tutkimuksia

3.3 Tuloksia

3.3.1 Laatu

3.3.2 Tehokkuus

3.4 Meta-analyysit

4 Käyttö yritysmaailmassa

Lähteet

[Cri06] Crispin, L.: *Driving Software Quality: How Test-Driven Development Impacts Software Quality.* Software, IEEE, 23(6):70–71, Nov 2006, ISSN 0740-7459.

[Muc08] Muccini, Henry: *Software testing: Testing new software paradigms and new artifacts.* Wiley Encyclopedia of Computer Science and Engineering, 2008.

[SMGM13] Stoica, Marian, Mircea, Marinela ja Ghilic-Micu, Bogdan: *Software Development: Agile vs. Traditional.* Informatica Economica, 17(4):64–76, 2013.

[Whi00] Whittaker, J.A.: *What is software testing? And why is it so hard?* Software, IEEE, 17(1):70–79, Jan 2000, ISSN 0740-7459.