Ohjelmistojen laadun parantaminen refaktoroinnilla

Simo Mäkinen
Tietojenkäsittelytieteen laitos
Helsingin yliopisto

Refaktorointi

- Refaktoroinnilla tarkoitetaan olemassaolevien ohjelmiston koodirakenteiden muokkaamista ilman ulkoisen toiminnallisuuden muutosta
- Refaktorointi tähtää koodin sisäisen laadun parantamiseen rakennemuutoksin

Tutkimuskysymyksiä

Tunniste	Kuvaus
RQ1	Millä tavoin ohjelmistoyritykset Suomessa kokevat refaktoroinnin?
RQ2	Miten refaktorointi nivoutuu kehitystyöhön?
RQ3	Mihin refaktorointipäätökset yrityksissä perustuvat, käytetäänkö laatumittareita?
RQ4	Onko kehityssyklin pituudella vaikutusta refaktorointiin?

Need for Speed (N4S)

- Valtakunnallisessa tutkimushankkeessa pyritään
  - parantamaan suomalaisten ohjelmistoyritysten reaktiivisuutta
  - kykyä toimittaa ohjelmistoja nopeammin loppukäyttäjälle
- Hankkeessa n. 30 yritystä ja n. 10 tutkimuslaitosta

DIGILE | N4S

Tutkimusasetelma

- Tapaustutkimus suomalaisista ohjelmistoalan yrityksistä ja niiden refaktorointikäytännöistä
- Haastateltiin keväällä 2015 yritysten kokeneita ohjelmistokehittäjiä ja arkkitehtejä
- 10 haastattelua, 12 haastateltavaa, 9 yritystä
- Semi-strukturoidut haastattelut kestoltaan tunnista kahteen
- Refaktorointiteemoja: määritelmä, menetelmät, työkalut, metriikat, riskit, hyödyt, haasteet, päätöksenteko, kehityssyklin vaikutukset
- Pääyhteistyökumppani: TAMPEREEN TEKNILLINEN YLIOPISTO

Mikä on refaktorointia?

- Päivittäin tehtäviä pieniä rakennemuutoksia ei usein koettu refaktoroinniksi
- Refaktorointi miellettiin enemmän useita työpäiviä kestäväksi koodin uudellenorganisoimiseksi
- Haastateltavien käsitys refaktoroinnista vastaa paremmin restrukturoinnin (restructuring, reengineering) käsitettä
- Kehittäjät arvioivat käyttävänsä n. viidenneksen työajastaan refaktorointiin tai restrukturointiin

Ohjelmiston laadun määrittäminen

- Kuvailkaa mitkä laatutekijät saattaisivat olla keskeisiä *refaktoroinnin* kannalta
- Miten ja mistä niitä voisi mitata, millaisia laatumittareita on olemassa?

Laatumetriikoiden käyttö yrityksissä

- Haastateltavat eivät pääsääntöisesti pitäneet laatumetriikoita oleellisina
- Laatumetriikan parantamisesta voi tulla tavoite itsessään, huomio keskittyy vääriin asioihin
- Staattista koodianalyysia kuitenkin hyödynnettiin jonkin verran turvallisuuskriittisissä tapauksissa

Refaktorointikokemukset edellisistä tutkimuksista

- Stroggylos ja Spinellis tutkivat avoimen lähdekoodin projektien rakennetta ennen ja jälkeen refaktoroinnin
- Koodin staattisen analyysin perusteella laatumetriikat eivät parantuneeet, päinvastoin
- Kytkentä- ja koheesiometriikat huonontuivat refaktoroinnin jälkeen

→ koodirakenteiden laadun parantamista ja refaktoroinnin onnistumista voi olla vaikea nähdä koodimetriikoista

Mistä refaktorointitarpeet johtuvat?

- Kiireen takia tehdään tietoisesti rakenteeltaan heikompia koodiratkaisuja
- Kehitystyötä aloitettaessa ei ole täydellistä tietoa nykyisistä ja tulevista tarpeista
- Kehittäjät oppivat koko ajan uutta ja kokonaisuus nähdään myöhemmin uudessa valossa

Näkemys refaktorointiin johtavista syistä

"*…vähän niinku omakotitaloo rakentaa, niihän sitä sanotaan että ensimmäinen pitäis polttaa, toinen myyä ja kolmannessa asuu ite. Koodin kanssa vähä sama juttu.*" –pääsuunnittelija suomalaisesta ohjelmistoalan yrityksestä

Refaktoroinnin hyödyt

- Refaktoroinnilla pyritään haastatteluiden mukaan ensisijaisesti parantamaan koodin
  - Ylläpidettävyyttä
  - Luettavuutta
  - Ymmärrettävyyttä
- Muita mainittuja suorituskyvyn parantaminen, modulaarisuuden tuomat edut, kehittäjämotivaatio

- Ensisijaisesti tavoitellut hyödyt samansuuntaisia kuin Microsoftin laajassa kehittäjäkyselyssä

Refaktoroinnin riskit ja haitat

- Kattavista testeistä huolimatta koodi voi hajota ja virheiden määrä lisääntyä
- Muutokset ulkoisiin rajapintoihin saattavat vaikuttaa kolmannen osapuolen palveluihin
- Ei takeita koodin muutoksesta parempaan suuntaan, laatu voi huonontua
- Vaivalla tehty isompi rakenteellinen muutos saatetaan joutua hylkäämään

Refaktoroinnin tekniset vaatimukset ja työkalut

- Versionhallintajärjestelmät
- Automatisoidut testit eri tasoilla antavat regressiovarmuutta
- Jatkuvan integraation palvelimet

- Kehitysympäristöjen refaktorointiapuvälineet hyvä lisä, mutteivät täysin välttämättömiä kehitystyölle

Refaktorointi ja päätöksenteko

- Refaktorointipäätökset perustuvat kehittäjien intuitioon
- Hyvien laatumittareiden puutteen vuoksi tarpeita vaikea kvantifioida
- Pienemmät refaktoroinnin hoituvat kehitystyön osana
- Laajempiin rakennemuutoksiin hyväksyntä ylempää
- Alihankintana tehdyssä asiakastyössä laajaa restrukturointia vaikeampi perustella
- Sisäisissä projekteissa refaktorointipäätökset helpompia

Refaktoroinnin dilemma

- Pyrittävä näkemään millaisia tarpeita ohjelmiston tulee täyttää tulevaisuudessa
- Kehittäjän tulee tehdä paras arvaus mitkä osat koodista pysyvät staattisina, mitkä mahdollisesti muuttuvat
- Lyhyen elinkaaren omaavilla ohjelmistotuotteilla refaktorointi ei ehkä kannata

Lähteet

- Fowler, M., Beck, K., Brant, J., Opdyke, W., Roberts, D.: Refactoring - Improving the Design of Existing Code, Addison-Wesley, 1999.
- Stroggylos, K., and Spinellis, D.: "Refactoring--Does It Improve Software Quality?." Proceedings of the 5th International Workshop on Software Quality. IEEE Computer Society, 2007.
- Kim, M., Zimmermann, T., Nagappan, N., A field study on refactoring challenges and benefits., In: SIGSOFT'12/FSE-20, ACM, pp. 1-11, 2012

Kiitos!