Testaustyökalut

Luento 11

Antti-Pekka Tuovinen

Tavoitteet

- Työkalutyyppejä
  - Testauksen hallinta
  - Testien määrittely
  - Staattinen analyysi
  - Dynaaminen testaus

Työkalut ja testaus

- Työkalujen käytön tavoitteet
  - Käsin tehtävän, samanlaisena mekaanisesti toistuvan ja/tai aikaa vievän työn automatisointi (ihmiset ovat huonoja automaatteja)
  - Sellaisten testien tekeminen, joita on vaikeaa tai mahdotonta tehdä manuaalisesti (rasitus- ja suorituskykytestit)
  - Inhimillisten virheiden riskin vähentäminen esimerkiksi suurten testi- ja tulosdatamäärien käsittelyssä

Työkalutyyppejä

- Testien suorituksen automatisointi, testidatan generointi, testitapausten generointi, testausprosessin hallinta, testin kohteen monitorointi, monikäyttötyökalut, jne.
- Yleiskäyttöisiä ja hyvin erikoistuneita
- Kaupallisia ja avoimia
  - Esimerkiksi OSS työkaluja löytyy osoitteesta http://www.opensourcetesting.org/
  - Muita http://www.imbus.de/english/test-tool-list/

Testauksen hallintatyökalut

- Testitapausten määrittely, luokittelu, priorisointi ja muu hallinta
- Testien suorituksen tilan seuranta
- Mahdollisesti linkitys vaatimushallintaan
- Vikaraporttien luonti- ja seurantatyökalut ovat välttämättömiä (incident management, bug tracking)
- Hallintatyökalujen integrointi muihin
- Testiraporttien ja –dokumenttien generointi

Testauksen hallintatyökalut

- Testia Tarantula –demo http://www.youtube.com/watch?v=E6DXwpJxlJY

Testien määrittelytyökalut

- Testitapauksen määrittelyyn kuuluvat esi- ja jälkiehdot, syöte ja odotettu tulos
- Testitapausten ja –datan generointityökalut helpottavat testien laatijan työtä tuottamalla testisyötteitä tai kokonaisia testitapauksia automaattisesti

Testitapausten generoinnin peruste

- Tietokantarakenteen määrittely (data base schema)
  - Tuotetaan skeeman mukainen testitietokanta
- Testikohteen koodi (white box)
  - Testioraakkeli tarvitaan
  - Pelkkä koodi (toteutus) on huono lähtökohta testaukselle

Testitapausten generoinnin peruste

- Testikohteen rajapinta (black box)
  - Automaattinen ekvivalenssiluokkiin jako ja raja-arvo analyysi rajapinnan määrittelyn perusteella (API, GUI)
  - Negatiivisten testitapausten automaattinen generointi
  - Testioraakkeli tarvitaan
- Testikohteen (formaali) spesifikaatio
  - *Model based testing*
  - Testisyötteiden ja tulosten generointi UML mallin avulla
  - Malli toimii oraakkelina

Esimerkki

- AgitarOne Junit –yksikkötestien automaattinen generointi
  - Yhdistää black box ja white box analyysejä
  - Kaupallinen työkalu
  - Demo http://www.agitar.com/downloads/demos/agitarOneVs_legacy/agitarOneVsLegacy.html
  - Mentelmän yksityiskohdat ja analyysialgoritmit liikesalaisuuksia, mutta tutkimustietoa toki aihepiiristä löytyy

Staattisen testauksen työkalut

- Katselmoinnit
- Spesifikaatioiden ja mallien staattinen analyysi
  - *Model checking*
  - UML-mallin täydellisyys ja ristiriidattomuus
- Koodin staattinen analyysi
  - Datavuoanomaliat, turvallisuusongelmat
  - …ja kymmeniä muita, katso. esimerkiksi http://findbugs.sourceforge.net/bugDescriptions.html

Dynaamisen testauksen työkalut

- Testien suorittamisen automatisointi
  - Testitapaukset syötteineen ja odotettuine tuloksineen koodataan testiskripteiksi tai –ohjelmiksi (JUnit -testitapausluokat)
  - Työkalu ajaa testit automaattisesti ja kirjaa tulokset (JUnit –laajennokset NetBeans ja Eclipse -ympäristöissä)
- Debuggerit
  - Kehittäjän työkaluja pääasiassa, mutta voivat olla tarpeellisia tiettyjen virhetilanteiden luomiseksi

Dynaamisen testauksen työkalut

- Testikehikot ja –ajurit (test framework, test harness)
  - Testiympäristön automaattinen generoiminen testikohteen analysoinnin perusteella
  - Tuki ajurien kirjoittamista varten
  - Riippuvuuksien analysointi ja tynkien (stubs) tai jäljittelijöiden (mock object) automaattinen (osittainen) generointi
    - http://en.wikipedia.org/wiki/Mock_object
  - Google Test C++ testing framework http://code.google.com/p/googletest/wiki/Primer

Dynaamisen testauksen työkalut

- Simulaattorit, emulaattorit
- Testausrobotit (capture and replay)
  - Käyttöliittymätapahtumien nauhoitus ja toisto testitapausten määrittelyn avuksi
  - Selenium on suosittu työkalu selaimessa toimivien sovellusten testitapausten muodostamiseen ja suorittamiseen http://www.youtube.com/watch?v=OeyeY6gaDVg

Dynaamisen testauksen työkalut

- Datalähtöinen testaus (data driven testing)
  - Periaatteessa sama testitapaus suoritetaan monta kertaa peräkkäin vain syötteiden vaihtuessa
  - Syötedata ja odotettu tulos talletetaan tiedostoon, tietokantaan tai taulukkon, josta testiskripti lukee datarivit ja ajaa testitapauksen
- Komento- tai avainsanatestaus (command/keyword driven testing)
  - Testitapaukset ja -skenaariot koostuvat abstrakteista toiminnoista (”enter login credentials”) ja niiden syötteistä
  - Testiä suoritettaessa abstrakti toiminto korvataan konkreettisella testikohteen toiminteella/dialogilla

Dynaamisen testauksen työkalut

- Vertailijat (comparator)
  - Suurien tietomassojen vertailuun (testin tuottama tulosdata vs. oraakkelin tuottama data)
- Dynaamiset analysaattorit testikohteen suorituksen aikana
  - Muistin käyttö jne.
- Kattavuusanalyysi testitapausten suorituksesta
  - Lause- ja haarautumakattavuus jne.
  - Työkaluissa paljon eroja
  - Esimerkiksi EclEmma coverage plugin for Eclipse

Suorituskyvyn testaus ja monitorointi

- Kuormitustestaus, vasteajat
  - Datan ja tapahtumien generointi
    - Määrä, frekvenssi
- Monitorointi
  - Testin aikaisen käyttäytymisen mittaaminen ja mittausdatan kerääminen
  - Profilointi
  - Vaatii erikoistuneita ja joskus kalliita välineitä
- Turvauhkien testaus
  - http://www.metasploit.com/index.jsp