Mallipohjainen testiautomaatio: perusteet ja huomioita käyttöönotosta

Rasa Siegberg, Conformiq Software Oy

Mallipohjaiseksi testaukseksi kutsutaan menetelmää, jossa kohdejärjestelmää koestavat testit ja niiden parametrit luodaan kohdejärjestelmän toimintaa ja sen ympäristöä kuvaavaan malliin perustuen. Todellista testiautomaatiota menetelmästä saadaan lisäämällä testien generoinnin jatkoksi niiden varsinainen suoritus, testitulosten tallentaminen, sekä tulosten arviointi.

Mallipohjaisen testiautomaation potentiaali testauksen ja laadunvarmistuksen tuottavuuden tehostajana on merkittävä, sillä automatisoimalla rutiininomaisia osia testauksen suoritus- ja raportointitöistä se tarjoaa mahdollisuuden vapauttaa testaushenkilöstöä korkeamman tason suunnittelu- ja analyysitehtäviin. Tätä tehostamispotentiaalia selkeyttänee analogia kääntäjiin, ohjelmistokehityksen perustyökaluihin - mallipohjainen testiautomaatio on testaukselle sitä mitä kääntäjäteknologia ohjelmistokehitykselle.

Ohjelmoinnin esihistoriassa ohjelmakoodi tuotettiin matalan tason konekielirutiineina. Tämä menetelmä oli hidas, kankea, ja erittäin virhealtis. Valtava harppaus tuottavuudessa saavutettiin kääntäjien kautta. Kääntäjät mahdollistivat ohjelmistosuunnittelun (eli varsinaisen ohjelman mallinnuksen) korkean tason ohjelmointikielellä (C, Pascal, Fortran). Korkean tason kielen esityksestä (mallista) kääntäjä sitten tuotti varsinaisen konekielisen toteutuksen eli itse ohjelman. Mallipohjaisessa testauksessa mallin voidaan katsoa olevan analogisessa asemassa esimerkin korkean tason ohjelmakoodin kanssa, kun taas itse testit vastaavat esimerkin konekielistä koodia. Mallipohjainen testiautomaatiosovellus on luonnollisesti analoginen vastine kääntäjälle.

Mallipohjainen testiautomaatio on testauksen välinekentässä verrattain uusi tulokas joten tältä kannalta on perusteltua puhua menetelmästä uuden sukupolven testausautomaationa tai uutena automatioparadigmana. Yksi menetelmän edelläkävijöitä on espoolainen Conformiq Software, jonka tuote Conformiq

> ”Mallipohjainen testiautomaatio on testauksen välinekentässä verrattaen uusi tulokas”

Test Generator on ensimmäinen kaupallinen mallipohjainen testiautomaatioratkaisu.

Seuraava kappale esittelee ne peruskomponentit, joista mallipohjainen testiautomaatiosovellus koostuu. Esittely liikkuu yleisellä tasolla eikä ole sinänsä sidoksissa Conformiq Softwaren ratkaisuun.

”Testimalli pyrkii kuvaamaan tuotteen ja ympäristön välistä rajapintaa tai tuotteen liittymää todellisuuteen”

Perusteita

Mallipohjaisella testiautomaatioratkaisulla voidaan ajatella olevan kolme geneeristä osakomponenttia:

- mallinnusformalismi
- analysaatio- ja suorituslogiikka
- liitynnät testattavaan järjestelmään

Seuraavat kappaleet esittelevät nämä komponentit ja joitakin huomioita niiden käytännön toteutuksesta.

Mallinnusformalismi

Testimallin tehtävä mallipohjaisessa testiautomaatiossa on kuvata testattava kohde, ja sen käyttäytyminen siltä osin kuin se on testaamisen kannalta olennaista. Mallin rakentamiseen voidaan käyttää erilaisia tapoja - se voidaan esittää tekstuaalisessa muodossa jonkinlaisella kuvauskielellä tai se voi olla graafinen esitys testattavan järjestelmän eri tiloista ja siirtymistä niiden välillä. Tärkeää on kuitenkin, että testimalli sisältää tiedon järjestelmän erilaisista mahdollisista tiloista ja niistä ehdoista tai lainalaisuuksista, jotka säätelevät tilasta toiseen siirtymistä.

Mallipohjaista testausta tarkastellessa huomio usein kiinnittyy testimallin ja jo olemassaolevien tuotemääritysten samankaltaisuuteen. Käytännön toteutusten käytettävyyden ja mallipohjaisen testiautomaation käyttöönoton helpottamiseksi olisikin edullista, että mallinnusformalismi olisi joku jo entuudestaan toimialalla tunnettu menetelmä. Lisäksi olisi edullista, että jo olemassaolevia tuotemäärityksiä ja -spesifikaatiota voitaisiin hyväksikäyttää testimalleja luotaessa. Conformiq Test Generatorissa käytettävyyttä ja käyttöönottoa on pyritty helpottamaan valitsemalla testimallien mallinnusformalismiksi UML (Unified Modeling Language) -tilakaaviot laajennettuna yksinkertaisella proseduraalisella ohjelmointikielellä.

Vaikka testimallien ja tuotteen määrittelyiden (arkkitehtuurimallien) välillä onkin helppo huomata samankaltaisuuksia on myös tärkeää tunnistaa niiden välinen merkittävä ero. Siinä missä arkkitehtuurimalli pyrkii kuvaamaan tuotteen sisäistä rakennetta ja sen jakoa erilaisiin rakenteellisiin osakokonaisuuksiin, testimalli pyrkii kuvaamaan tuotteen ja sen ympäristön välistä rajapintaa tai tuotteen liittymää todellisuuteen. Tästä syystä usein ehdotettu lähestyminen jonka mukaan testiaineisto ja -tapaukset tulisi voida generoida suoraan testattavan järjestelmän määrittelytiedoista ei tarkemmalla tarkastelulla vaikuta hedelmälliseltä. Testimallien olennaisimpana ominaisuutena voikin pitää niiden sisältämää tietoutta siitä kuinka, ja millaisilla lainalaisuuksilla, testattavan järjestelmän tulisi toimia (ulkoapäin katsottuna), kun taas saman järjestelmän määrittely kertovat lähinnä sen kuinka ko. järjestelmä tuon käytöksen saa aikaiseksi. Pelkistetysti testimallin voidaan ajatella kuvaavan testattavaa järjestelmää ns. ”black-box”-testauksen kannalta, kun taas tuotespesifikaatiot kuvaavat järjestelmää ns. ”glass-box”-lähestymisellä.

Analysaatio- ja suorituslogiikka

Mallipohjaisen testiautomaation suurin työtä säästävä potentiaali syntyy menetelmän tarjoamasta mahdollisuudesta tuottaa suuria määriä testiaineistoa automaattisesti testimalliin perustuen. Testiaineiston ja itse testien generointi asettaa mallipohjaisille testiautomaatioratkaisuille vaatimuksia niin mallien analysoinnin logiikan ja ”älykkyyden” kuin testien suorittamisenkin suhteen, ja sen perusteena oleva teknologia on laskentateoreettisen ja tietojenkäsittely-

tieteellisen tutkimuksen kohteena saapunut todellisen läpimurron kynnykselle 90-luvun aikana.

Käytännön mallipohjaisessa testiautomaatioratkaisussa analysaatio- ja suorituslogiikan tehtävä on:

- analysoida testimalli ja todeta sen mahdollistamat erilaiset testitapahtumat ja -aineisto, sekä huomioida millainen testijärjestelmän käytös tulkitaan läpäistyksi testiksi ja mikä hylätyksi testiksi
- tuottaa, mallin asettamissa rajoissa, "testistimulusta", jota testien suorituksen aikana syötetään testattavalle järjestelmälle
- pitää kirjaa toteutetuista suoritetuista testeistä ja niiden lopputuloksista sekä "testistimuluksina" käytetyistä arvoista
- pitää kirjaa ja seurata testikattavuutta, ja verrata saavutettua kattavuutta testimallin mahdollistamaan "testiavaruuteen"

kattavan, laajuudeltaan valtavan, ja tarkkuudeltaan yksityiskohtaisen testiaineiston luomisen varsin korkealla käsitetasolla liikkuvan testimallin pohjalta.

Liitynnät testattavaan järjestelmään

Mallipohjaisen testiautomaatioratkaisun kytkentä reaalimaailman testiympäristöön mahdollistaa testauksen joko reaaliajassa tai eräajona jonkin ohjelmointikielen kautta. Testijärjestelyn liityntää testattavaan järjestelmään (SUT - System Under Test) kutsutaan testausrajapinnaksi (Test API). Rajapinta on luonnollisesti järjestelmäkohtainen, mutta yleisenä lainalaisuutena voidaan pitää sitä, että rajapinnan tulisi olla testaamisen kannalta merkityksellinen, eli sellainen, että sen kautta voidaan tarkkailla ja vaikuttaa juuri testauksen kohteena oleviin toiminnallisuuksiin. Kun ”perinteisessä” käyttöliittymäpohjaisessa testiautomaatiossa rajapinta rajoittuu useimmiten testattavan sovelluksen käyttöliittymään (ja sen suoranaisesti kontrolloimiin toiminnallisuuksiin), on mallipohjaisen testiautomaation kannalta tehokasta etsiä ”lähempänä” testattavan järjestelmän logiikkaa olevia rajapintoja jolloin testausta eivät rajoita käyttöliittymän ominaisuudet ja konventiot.

Käytännön sovelluksissa mallipohjaisen testiautomaation kytkentä testattavaan järjestelmään toteuttaa seuraavat toiminnat:

- muuntaa testityökalun tuottamat testiaineistot testattavan järjestelmän ymmärtämään muotoon
- välittää testityökalulta testattavalle järjestelmälle testien vaativaa syötettä
- muuntaa testattavan järjestelmän vastaukset testityökalun ymmärtämään muotoon
- välittää testattavan järjestelmän vastauksia testityökalulle
- havainnoi testattavan järjestelmän käytöstä testien aikana

Conformiq Test Generator:in tapauksessa liityntä testattavan järjestelmän ja testityökalun välillä toteutetaan järjestelmäkohtaisella adapterikomponentilla, joka siis kootaan jokaista testiympäristöä ja sen erityispiirteitä vastaavaksi. Vaikka kyseisen kaltainen liityntä olisi mahdollista myös kutakuinkin vakiintuneita teknologioita (CORBA, XML) käyttäen, on valitun menetelmän etuna sen joustavuus. Järjestelmäkohtaisella adaptaatiolla voidaan kytkeytyä myös esimerkiksi sulautettuihin järjestelmiin, joissa suorituskyky on usein rajoitettu ja vaatimukset esim. muistinkulutukselle ovat tiukat.

”Mallipohjaisen testiautomaation käyttöönottoa edelsi menetelmän arviointi- ja pilotointiprojekti””

Conformiq Test Generator:in tapauksessa ratkaisun analyysi- ja suorituslogiikka on rakennettu kykeneväksi niin reaaliaikaiseen "on-the-fly"-testaukseen kuin erilaisten eräajoina suoritettavien testikirjastojen (esimerkiksi Java- tai TTCN-3 kielillä) luomiseenkin. Analyysi- ja suorituslogiikan sisältämä älykkyys mahdollistaa

Havaintoja käyttöönoton tiimoilta

Conformiq Software:n mallipohjaisen testiautomaation ratkaisun asiakaskunta tarjoaa menetelmän ja työkalun käyttöönotto- ja käyttökokemuksia usean vuoden ajalta. Asiakaskunnan ja sen toimialojen heterogeenisyys tarjoaa laaja-alaisen aineiston uuden sukupolven testaustyökalun käyttöönoton tarkastelulle. Seuraavat kappaleet tiivistävät käytännön testiautomaatioprojektien aikana tehtyjä havaintoja.

Valmistautuminen ja suunnittelu

Lähes kaikissa tapauksissa mallipohjaisen testiautomaation käyttöönottoa edelsi (usein varsin perusteellinen) menetelmän arviointi- ja pilotointiprojekti. Tätä lähestymistä on lähes poikkeuksetta pidetty hyvänä sen tarjotessa taloudellisesti riskittömän ja organisaation/testattavan järjestelmän omia erityispiirteitä huomioivan käynnistysvaiheen.

Teknisen tuote-evaluaation ohella on myös havaittu, että paras tulos uuden testausparadigman soveltamisessa saavutetaan, kun uusi menetelmä otetaan huomioon jo varsin aikaisessa vaiheessa testausta suunniteltaessa, jolloin on syytä kriittisesti tarkastella niitä tavoitteita, joita testauksen automatisoinnilla tavoitellaan. Mitä tarkemmin tavoitteiden asettelu on ennen automaatioprojektin käynnistystä tehty, sen paremmin on tavoitteita kyetty saavuttamaan ja prosessia seuraamaan.

Prosessi ja kompetenssit

Testauksen toteuttamiseen liittyvät työkalujen ja suunnittelun ohella olennaisesti se organisaatio ja henkilöstö, joka itse testausprojektin toteuttaa, ja ne prosessit, joiden mukaan tämä organisaatio toimii. Testauksen automatisoinnin onnistumisen todennäköisyyttä kasvattaa olennaisesti se, että lähtötilanteena on ainakin jollakin tasolla formalisoitu testauksen ja laadunvarmennuksen prosessi.

”Testausautomaatiotyökalut vaikuttavat vaativan henkilöstöltä eri tyyppisiä kompetensseja kuin “perinteinen” manuaalinen testaus”

Yleisesti testausautomaatiotyökalut automaatioparadigmasta riippumatta vaikuttavat vaativan henkilöstöltä eri tyyppisiä kompetensseja kuin “perinteinen” manuaalinen testaus. Tämä pätee myös mallipohjaisen testiautomaation kohdalla. Käyttöönottoprojektien alussa on usein ohjelmointi-intensiivinen vaihe, kun testityökalua sovitetaan ja kytketään testattavaan järjestelmään. Projektin alkuvaiheen jälkeen testausjärjestelyn tuntemuksen sijaan korostuu testattavan järjestelmän tuntemus, kun varsinaisia testimalleja rakennetaan ja ylläpidetään. Kokemusten mukaan projektin alkuvaiheissa on usein tilausta työkalutoimittajan projektiin osallistumiselle. Tämän tarve kuitenkin nopeasti vähenee projektin edetessä ja testijärjestelyn kypsyessä.

Johtopäätelmiä

Mallipohjainen testiautomaatio lupaa merkittävää parannusta testauksen laadun ja kattavuuden saralla. Se myös vaatii käyttöönottajaltaan merkittävää sitoumusta. Mallipohjaisen testiautomaation käyttöönotto merkitseen organisaatiolle panostusta teknologiaan, osaamisen kehittämiseen ja kurinalaiseen ja hyvin organisoituun laadunvarmennukseen. Sillä saavutettavat hyödyt toteutuvat sitä todennäköisemmin mitä tarkemmin automaatioprojektin tavoitteet ja resurssointi on määritelty ja organisaatio on niihin sitoutettu.

Kirjoittajalla toimii testauskonsulttina Conformiq Software Oy:ssä, ja osallistuu aktiivisesti asiakasprojekteihin, joissa mallipohjaista testiautomaatiolähestymistä ja Conformiq Test Generator:ia sovelletaan käytännön ympäristöissä.

Espoolainen Conformiq Software työllistää noin 40 testauksen asiantuntijaa, ja on maan johtava mallipohjaisen testiautomaation kehittäjä ja Conformiq Test Generator –testiautomaatioratkaisun kehittäjä.