Hanna-Maija Ilmola

TAVALLISEN KODIN MUUTTUMINEN ÄLYKODIKSI JA AUTOMAATIOTESTAUS OHJELMISTOTESTAUKSESSA

TAVALLISEN KODIN MUUTTUMINEN ÄLYKODIKSI JA AUTOMAATIOTESTAUS OHJELMISTOTESTAUKSESSA

Hanna-Maija Ilmola
Opinnäytetyö
Syksy 2017
Tietotekniikan tutkinto-ohjelma
Oulun ammattikorkeakoulu

TIIVISTELMÄ

Oulun ammattikorkeakoulu
Tietotekniikan tutkinto-ohjelma, laite- ja tuotesuunnittelu

Tekijä(t): Hanna-Maija Ilmola
Opinnäytetyön nimi: Tavallisen kodin muuttuminen älykodiksi ja automaatiotestaus ohjelmistotestauksessa
Työn ohjaaja(t): Kari Jyrkkä ja Jyrki Eskola
Työn valmistumislukukausi ja -vuosi: Syksy 2017 Sivumäärä: 61

Työ toteutettiin koosteopinnäytetyönä kahdessa osassa. Työn molemmat osat löytyvät liitteistä.

Ensimmäisen osan aiheena oli tutustua uudenlaisen kodin, älykodin, kehitykseen ja siihen, miten tavallinen käyttäjä voi itse rakentaa älykotia. Työn tarkoituksena oli selvittää, millaisia vaihtoehtoja kuluttajilla on toteuttaa älykodin ominaisuuksia omaan tarkoituksiin sopiviksi.

Ensimmäisen osan tarkoitus oli tutkia itseä kiinnostavaa aihetta ja kirjoittaa siitä tutkimusmielessä ns. tietopaketti. Valitsin aiheeksi älykodin, koska se on kasvava talotrendi ja älykkäät kodinkoneet ovat valtaamassa itselleen isoa osaa kodinkonemarkkinoista, mikä teki aiheesta myös hyvin ajankohtaisen.

Toisen osan aiheena oli automaatiotestaus ohjelmistotestauksessa. Työn tarkoituksena oli dokumentoida automaatiotestauksen käyttöönotto sekä myös testikäytänteitä. Työ toteutettiin Keysight Technologies Finland Oy:lle aikaisempien yritysprojektien jatkeena. Työssä keskityttiin automaatiotestauksen määritelmään, käyttöönottoon sekä testien käytäntöihin ja toteutukseen.

Työ antoi tulevaisuuden varalle paljon tietoa uusista älykodinkoneista ja mahdollisuuksista kehittää omaa kotia. Saatiin myös laaja kuva automaatiotestauksesta ohjelmistotestauksessa ja sen tärkeydestä testauksen tai testaajien kannalta. Nykyaikaiset sovellukset ja internetsivustot saattavat olla vaiheittain vaikeita testattavia manuaalisesti, joten automaatiotestaus on enemmän kuin tervetullut apu jokapäiväiseen testaukseen.

Asiasanat: älytalo, automaatiotestaus, koosteopinnäyte

SISÄLLYS

1 JOHDANTO

Opinnäytetyö toteutettiin koosteopinnäytetyönä. Opiskelija sai valita, haluaako tehdä työn kokonaisena (15 op) vai osissa (5 op ja 10 op). Ensimmäisen osan toteutin keväällä 2017 ja toisen osan heti perään syksyllä 2017. Olisi ollut myös mahdollisuus toteuttaa työ kolmessa osassa.

Ensimmäinen osa toteutettiin oman aiheen valinnan mukaan, joka oli pienempi kokonaisuus työstä. Toinen osa tehtiin Keysight Technologies Finland Oy:lle jatkona aikaisemmin tehdyille yritysprojekteille kyseisessä yrityksessä.

Molemmat osat löytyvät tämän dokumentin liitteistä.

2 OPINNÄYTETYÖN OSAN 1 AIHE, TAVOITE JA TULOKSET

Ensimmäisen osan (liite 1) aiheena oli tavallisen kodin muuttuminen älykodiksi ja tavallisen kuluttajan mahdollisuudet toteuttaa se itse. Työssä tarkasteltiin älykodissa käytettäviä tekniikoita ja vaihtoehtoja, millä esimerkiksi kodinkonelaitteilla tavallinen kuluttaja voi tehdä omasta kodistaan älykkään ilman, että joutuu rakentamaan itselleen uuden kodin tai asennuttamaan kalliit järjestelmät. Näkökulma oli niin sanotusti ”DIY” eli tee-se-itse.

Älykodin suosio on nousussa ja ihmiset haluavat laitteidensa olevan enemmän älykkäitä ja helppokäyttöisiä. Laitteita löytyy aina kahvinkeittimestä pyykinkoneeseen ja kattolampusta lämmitykseen. Työn tavoitteena oli tutkia mahdollisuuksia tavallisen kodin älyllistämiseen erilaisten kodinkoneiden avulla (ottaen huomioon niiden kehityksen vuosien varrella) ja myös tutkia, minkälaisia järjestelmiä on olemassa ja mitä mahdollisuuksia niissä on.

Valitsin aiheen mielenkiinnon ja ajankohdan takia. Itseä on älykoti kiinnostanut kaiken sen mahdollisuuksien vuoksi ja mielenkiinto nimenomaan on keskittynyt siihen, mitä kaikkea siihen voi tehdä ja missä se luovuuden raja tulee vastaan. Vielä ei ainakaan rajaa ole tullut vastaan. Nykypäivänä kun nuorempi sukupolvi alkaa ostaa itselleen taloja ja rakentamaan, haluavat he tehdä asumisesta helppoa ja älylaitteet mahdollistavat sen ainakin osittain. Sitä mukaa, kun laitteita kehitetään, kehittyy myös kodin tekniikka.

3 OPINNÄYTETYÖN OSAN 2 AIHE, TAVOITE JA TULOKSET

Toisen osan (liite 2) aiheena oli automaatiotestaus ohjelmistotestauksessa. Työssä tarkasteltiin automaatiotestauksen käsitettä ja sen käytänteitä. Tavoitteena oli dokumentoida automaatiotestauksen käyttöönotto, testikäytänteet sekä myös muutamia esimerkkitapauksia. Tarkoituksena oli tehdä dokumentti, jonka avulla testaaja pystyisi aloittamaan automaatiotestauksen tai jatkamaan toisen testaajan automaatiotestausta.

Työ toteutettiin Keysight Technologies Finland Oy:lle yritysprojektien jatkeena. Aihe liittyy yritysprojekteihin ja taustana dokumentille käytettiinkin yrityksen omaa tuotetta Nemo Cloud. Itse tuotetta ei käytetty esimerkkinä dokumentissa. Aihe oli myös jatkumoa yliopisto-opiskelijan gradulle, jossa valittiin kyseinen työkalu, jolla testit toteutettiin.

Keysight Technologies Finland Oy kehittää, valmistaa ja myy Nemo-tuotemerkillä testaus- ja mittausratkaisuja langattomien matkapuhelinverkkojen peiton ja laadun mittaamiseen, optimointiin, vianetsintään ja palvelutason vertailuun.

Työn aihe päätettiin nopeasti, sillä yrityksellä oli tarve automaatiotestauksen dokumentoinnille, koska se on iso lisä manuaaliseen testaukseen ja ei ollut olemassa olevia ohjeita testaajille kyseisestä asiasta. Automaatiotestauksen hyöty on suuri yritykselle ja sen avulla testaajat saavat nopeampaa tietoa sivustojen virheistä ja saavat tiedon myös eteenpäin kehittäjille.

4 YHTEENVETO

Opinnäytetyössä oli paljon opittavaa, sillä ensimmäinen ja toinen osa eivät tukeneet toisiaan ollenkaan. Tietenkin oli hyvä, että sai laajasti tietoa eri aiheista ja pystyy sitten tulevaisuudessa hyödyntämään tietoa työelämässä.

Automaatiotestaus on tällä hetkellä testauksen se niin sanottu ”kuuma aihe” ja olenkin tovin aikaa siihen tutustunut. Suurin osa tiedoista on tullut oppimisen kautta ja ihan konkreettisesti tekemällä testejä ja kohtaamalla epäonnistumisia. Tutkimalla kyseistä aihetta saatiin paljon tietoa testikäytänteistä ja ylläpidosta. Näiden tietojen avulla pystyy toinenkin testaaja jatkamaan tai aloittamaan testauksen. Dokumenttiin koottiin yleisimmät asiat automaatiotestauksesta Robot Framework -ohjelmalla toteutettuna. Tietyt seikat pätevät myös muihin työkaluihin.

Älykoti taas on uusi alue tekniikan saralla ottaen huomioon sen, että niitä on vasta pari vuotta rakennettu järjestelmineen. Kuluttajat voivat joko ostaa kokonaan uuden järjestelmän tai koota haluamansa kokoonpanon. Älylaitteiden suosio on ollut nousussuhdanteista ja kodinkoneet täytyy saada samalle tasolle. Aihe antoi paljon tietoa eri mahdollisuuksista oman älykodin kehittämiseen ja siitä, mitä tulevaisuuden rakennuksilta voidaan odottaa.

Molemmat aiheet liittyvät tulevaisuuteen, sillä älykoti on tavallisen kodin tulevaisuuden muoto ja automaatiotestaus on tulossa vahvasti manuaalisen testauksen rinnalle pysyvästi. Molempien aiheiden antama tietämys tulee olemaan hyödyksi työelämässä.

Liite 1

Hanna-Maija Ilmola

TAVALLISEN KODIN MUUTTUMINEN ÄLYKODIKSI

Liite 1

TAVALLISEN KODIN MUUTTUMINEN ÄLYKODIKSI

Hanna-Maija Ilmola
Opinnäytetyö, osa 1
Kevät 2017
Tietotekniikan tutkinto-ohjelma
Oulun ammattikorkeakoulu

SISÄLLYS

1 JOHDANTO

Tämä on kolmiosaisen opinnäytetyön ensimmäinen osa. Tässä osassa perehdytään nykypäivän älykoteihin ja niiden kehitykseen sekä suosion nousuun. Näkökulmana työhön on käytetty tavallisen kuluttajan mahdollisuuksia tehdä omasta kodistaan älykoti.

Älykoti-sanaa käytetään uudesta kodista, joka sisältää erilaisia laitteita, joita voidaan ohjata esimerkiksi ajastuksilla ja puhelimen sekä tabletin sovelluksilla. Ohjattavia laitteita voivat olla valaistus, viihde-elektroniikka, hälytysjärjestelmä, lämmitys ja ilmastointi. (1.) Kotien digitalisoituminen ja siihen liittyvä teknologia ovat alkaneet yleistyä. Asumisesta halutaan tehdä mahdollisimman helppoa ja vaivatonta, joka vastaa nykyihmisen vaatimuksia.

Tämän aiheen valitsemiseen vaikuttivat mielenkiinto asiaa kohtaan sekä teknologian kehittyminen. Taustalla on myös halu saada tietää älykodeista enemmän ja ehkä tulevaisuudessa rakennuttaa sellainen kaikkine laitteineen.

2 ÄLYKOTI

2000-luvun ihmiset ovat omaksuneet jatkuvasti uudistuvan teknologian omakseen ja soveltavat sitä eri asioihin. Se on laajentunut jo ihmisten asumiseen ja sen ympäristöön. Nykypäivänä on ihan normaalia, että kodissa on älylaitteita ja kaikenlaisia muitakin elektronisia laitteita. Ihmisten asumisesta halutaan tehdä mahdollisimman helppoa ja vaivatonta. Sen takia on päädytty kehittämään älykoteja, jotka takaavat elämisen helppouden.

Arjesta saadaan hyvin rentoa ja huoletonta automatisoimalla melkein kaikki kodissa. Enää ei tarvitse itse edes laittaa verhoja kiinni, vaan esimerkiksi ajastimella toimivat verhot tekevät sen itse. Nykypäivän ihmiset ovat niin kiireisiä ja paljon työtä tekeviä, että etäältä toimivat tai automatisoituneet laitteet ovat tervetulleita apureita arkeen.

Älykoti siis on ihan tavallinen koti, joka sisältää älykkäitä laitteita, kuten esimerkiksi ajastuksilla toimivat verhot, kodin hälytysjärjestelmä, ilmastointi ja valot. Suurimman osan kodin elektroniikasta pystyy automatisoimaan ja ohjaamaan puhelin sovelluksen avulla. Älylaitteiden asennuksella annetaan kodille ja sen asukkaille erilaisia hyötyjä: samat hyödyt mitä teknologia ja oma tietojenkäsittely ovat tuoneet meille 30 vuoden aikana, mukavuutta ja säästöä ajassa, rahassa ja energiassa (1). Älykoti yhdistää kaikki laitteet ja sovellukset niin, että ne voivat kommunikoida toistensa ja kuluttajan kanssa. Kaikki laitteet, jotka käyttävät sähköä, voidaan yhdistää kotiverkkoon ja ohjaukseen. Käskyn voi antaa äänellä, kaukosäätimellä tai tietokoneella ja koti reagoi siihen. Tekniikka on yksinkertainen, sillä systeemit, joita käytetään, ovat vain vastaanottimia ja lähettimiä. Vastaanottimet havaitsevat tietyn signaalin lähettimestä, joka myönnetään käskyksi. (2.)

2.1 Käytetyt tekniikat

Älykoti hyödyntää viimeisintä langatonta teknologiaa hienostuneiden sensoreiden kanssa säästämällä rahaa energialaskuista ja keskittää ohjauksen älykodin laitteista suoraan älypuhelimeesi tai tablettiisi (3). Kaksi tekniikkaa, joita

voi itse asentaa ja hyödyntää, ovat esimerkiksi Z-Wave ja UniPi. Näiden kahden lisäksi on olemassa paljon muitakin älylaitteiden tarjoajia.

2.1.1 Z-Wave

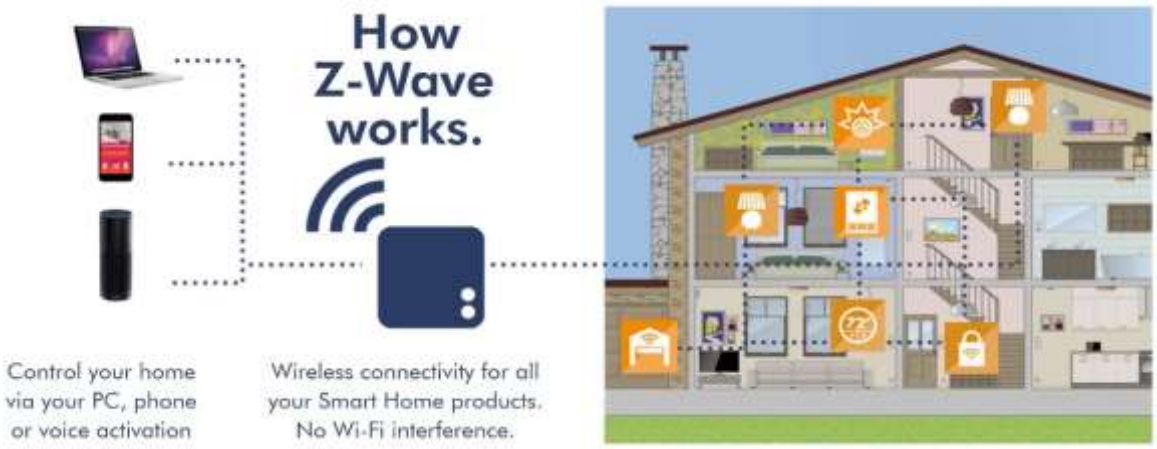
Z-Wave on langaton teknologia, joka antaa älylaitteiden kommunikoida toisilleen. Kotilaitteet, kuten valot, ovilukot ja termostaatit, on tehty älykkäiksi. Kun Z-Wave-yhdistettävyys on lisätty laitteeseen, antaa se niille kapasiteetin kommunikoida ja suorittaa haluttuja toimintoja. Z-Wave toimii langattomasti ja turvallisesti ja laitteita voidaan helposti ohjata puhelimessa, tabletilla jne. Tämä on langaton tekniikka, joka ei häiritse WiFi-signaalia. (4.)

Z-Wave on todella tehokas, vähäenerginen teknologia. Moni Z-Wave-laite toimii pattereilla yksistään, useimmiten vuoden ajan tai kauemmin, ennen kuin tarvitsee uudet patterit. Toiset kytketään seinään ja on olemassa myös Z-Wavella ohjattavia AC-pistorasioita, mikä antaa tehdä koko kodin elektronisesta järjestelmästä älykkään ja energiaystävällisen ohjaamalla ja tarkkailemalla energian käyttöä. (4.)

Ilman esteitä kuten seiniä tai huonekaluja, kantavuus kahden Z-Wave-tuotteen kanssa on noin 40 metriä. Kun esteet kodissa pienentävät kantavuutta, Z-Waven mesh-verkko antaa signaalin ”hypätä” muiden tuotteiden läpi ja saavuttaa päämääräisen laitteen ohjatakseen sitä. Z-Wave tukee neljää hyppyä, joten koko kodin kattavuus kasvaa riippuen tuotteiden määrästä verkossa. Maksimi kantama neljällä hypyllä on karkeasti 200 metriä.

Tällä voi ohjata 1−232 laitetta yhdellä verkolla. Z-Waven avulla voi rakentaa oman älykodin yksi laite kerrallaan. (4.)

Kuvassa 1 havainnollistetaan, miten Z-Wave toimii omakotitalon sisällä.



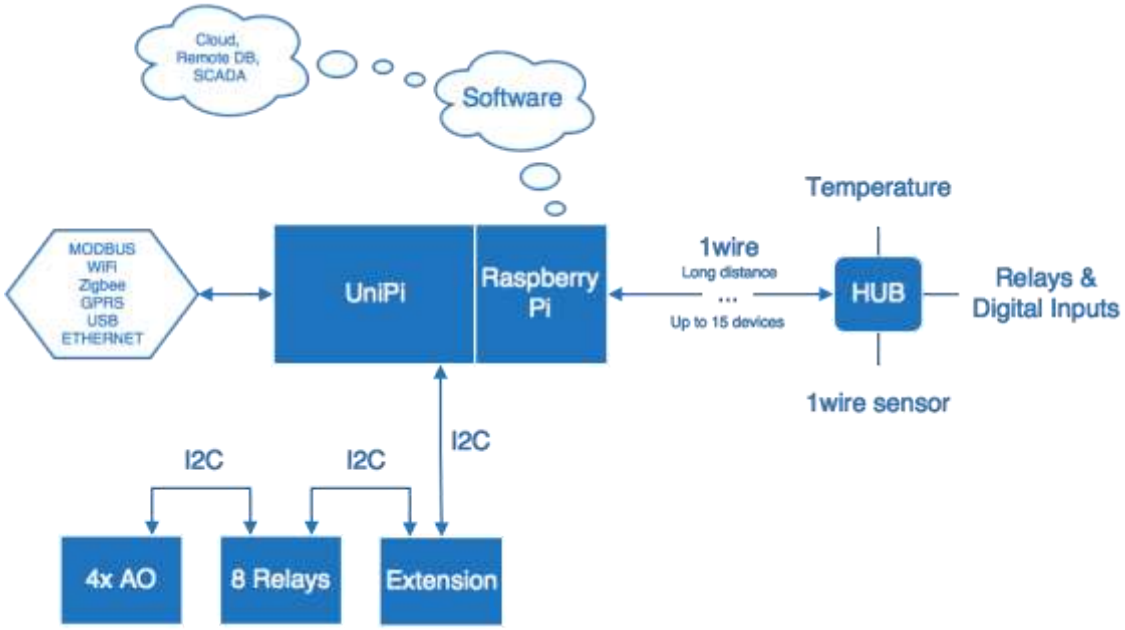
KUVA 1. Miten Z-Wave toimii (5)

2.1.2 UniPi

UniPi-alusta on suunniteltu sopimaan halvan ja helppokäyttöisen Raspberry Pi -minitietokoneen kanssa. Asennettu UniPi on kokonainen systeemi, joka sisältää asennetun ohjelmiston ja laitteiston.

Raspberry Pi on minitietokone, joka tarjoaa kaiken prosessointivirran. Kaikki ohjelmisto automatisointiin on asennettu sen sisälle. Raspberry Pi on koko järjestelmän aivot. Raspberry tarvitsee keinon, millä päästä yhteyteen sensoreiden, laajennusten ja muiden kanssa. UniPi on lisäalusta, jossa on analogisia ja digitaalisia sisääntuloja, ulostuloja, releitä ja kanavia. Se on silta Raspberryn ja muiden laitteiden välillä. Yhdessä nämä tekevät tehokkaan PLC:n (programmable logic controller). Laitteeseen on saatavilla laajennuksia esimerkiksi I2C ja 1wire. Viimeinen osa systeemistä saa kaikki laitteet yhdistettyä UniPihin kuten laajennuksen, magneettiset kontaktit tai 1wire-lämpömittarin. (7.)

Kuvassa 2 kuvataan näiden laitteiden yhteyttä.



KUVA 2. Raspberry Pi ja UniPi yhteyskaavio (6)

Tavoitteena on ollut tuottaa tuote, joka on yhteensopiva jokaisen suositun ohjelmistoalustan kanssa. UniPi tarjoaa sekä avoimen lähdekoodin (ilmainen ja voi muokata itse haluamallaan tavalla) että kaupallisen ratkaisun (sitä kehittää ja päivittää yksityiset yritykset). (7.)

Asennukseen on valmisteltu nopea ja helppo asennuspaketti. Monet ratkaisut tarjoavat visuaalisen ohjelmoinnin, joten asennus hoituu muutamassa tunnissa.

Monella tuetulla ohjelmistolla voi tehdä nerokkaita ja omaperäisiä ratkaisuja tai uusia projekteja. Kaikki voidaan automatisoida, esimerkiksi lämmityksen hallinta, lähteen kulutuksen monitorointi, ympäristön kunnon monitorointi, turvallisuus järjestelmä, pääsyn hallinta, automatisoidut valot ja lukot sekä siivouslaitteet.

Kaikki data voidaan ladata heti pilveen, josta pystyy tarkistamaan ja hallitsemaan, milloin tahansa, missä tahansa puhelimella, tabletilla tai tietokoneella. Sen pystyy jakamaan myös muiden kanssa tarvittaessa. (7.)

3 ÄLYKODIN KEHITYS JA SUOSIO

Voidaan katsoa, että älykodin kehitys on alkanut jo 1800-luvulla. Nikola Tesla kehitti 1898 langattoman kauko-ohjaimen, jolla hän ohjasi miniatyyristä venettä lähettämällä radioaaltoja (8). Tällä todistettiin langaton ohjaus, jota tarvitaan nykypäivänä moneen laitteeseen. Tämän jälkeen teknologia alkoi kehittyä ja mullistua entisestään. Laitteita on kehitetty niin paljon eteenpäin, että niistä pystyy tulemaan osa älykotia. Kaikkien laitteiden ei tarvitse olla langattomia toimiakseen osana älykotia.

On olemassa myös yrityksiä, jotka rakentavat kodin älykodiksi eli suunnittelevat ja asentavat taloon tulevat laitteet. Suomessakin on rakennettu näitä taloja, mutta niitä ei ole kovin paljon vielä. Helsingin Kalasatama on ensimmäinen älykkäiden energiajärjestelmien mallialue, jossa yrityksillä on mahdollisuus yhdistellä uusinta energia-, informaatio- ja viestintäteknologiaa (9).

3.1 Yleisten kodin laitteiden kehitys kohti älykodin maailmaa

Yleisien laitteiden kehitys on alkanut kauan aikaa sitten. Jotta niiden kehitys on päässyt siihen pisteeseen, missä ne ovat nykypäivänä, on tarvittu esimerkiksi sähköä ja erilaisia valmistusmateriaaleja. Aurinkopaneelien avulla kodin sähkö tuotetaan itse ja näin säästetään paljon energiaa. Uudet laitteet suunnitellaan energia- ja ympäristöystävällisiksi, joten aurinkopaneelit ovat hyvä energianlähde.

Laitteiden kehitys on pitkällä, mutta silti niiden toimintaan yritetään keksiä arkea helpottavia ratkaisuja ja aina vain enemmän energiaystävällisempiä vaihtoehtoja. Halutaan maksimoida säästöt ja elää vähemmällä kulutuksella. Laitteet alkavat automatisoida itseään ja ne myös tietävät enemmän ja osaavat käyttää tietoa hyväksi myöhemmin, eli ne ovat älylaitteita.

3.1.1 Imuri

Ensimmäinen kodinkone, joka tuli koteihin, oli imuri. Hubert Cecil Booth kehitti ensimmäisen version imurista 1901 ja patentoi sen (10). Booth rakensi koneen,

joka toimi sisäisellä polttomoottorilla. Se käytti mäntäpumppua saadakseen ilman läpi joustavista putkista ja suodattimesta, joka oli tehty kankaasta. Sitä täytyi vetää hevosilla, sillä se oli hyvin iso. Boothin seuraava imuri oli sähköllä toimiva, mutta silti vielä liian iso talojen sisälle (kuva 3). (11.)

KUVA 3. Hubert Cecil Boothin ensimmäinen imuri (12)

Nykypäivänä käytettävä imuri on huomattavasti pienempi ja helppokäyttöisempi. Siitä on tullut joka kodin peruskodinkone, jolla saa nopeasti siivottua. Omakotitaloihin on myös asennettu keskuspölynimureita, jolloin ei tarvita kuin imurin letkulle paikka seinässä. Edistyksellisin imuri lienee robotti-imuri (kuva 4), joka liikkuu itsestään rullien avulla ja sensoreilla se tunnistaa esteet edessään, pysähtyy ja vaihtaa suuntaa. Ihmisen ei tarvitse kuin käynnistää imuri ja tämä hoitaa itse loput.

KUVA 4. Robotti-imuri (13)

3.1.2 Kahvinkeitin

Kahvikeittimen kehitys on lähtenyt liikkeelle vanhasta tavasta tehdä kahvia eli pannukahvista. Kuumaan veteen annosteltiin kahvijauhetta ja kiehautettiin uudestaan, jonka jälkeen sen annettiin olla hetken ja kaadettiin sitten erilliseen tarjoilukannuun. Tämä tapa on edelleenkin yleinen valmistustapa, varsinkin mökeillä. Nykypäivänä vain käytetään suodatinta, jonka läpi lasketaan kuuma vesi. Kun kahvinkeitin tuli sähkökäyttöiseksi, siihen tuli lisää osia. Nykyään keitin koostuu vesisäiliöstä, suodatinpidikkeestä, kahvipannusta ja lämpölevystä pannun alla. Lämpölevy pitää huolen siitä, että kahvi pysyy lämpimänä niin kauan kuin keitin on päällä. Keittimissä on myös aikakatkaisu. Jos keitin on ollut esimerkiksi 40 minuuttia päällä, keitin menee itsestään kiinni. Halvemmissa kahvinkeittimissä on myös tämä ominaisuus ja sitä melkein vaaditaan jo nykypäivänä. Teknologia kuitenkin kehittyy ja perinteiset kahvinkeittimet tekevät tilaa uudenlaisille keittimille – mutta klassinen keitin on aina suosittu. Kuvassa 5 on yksi suosituimmista nykyaikaisista kahvinkeittimistä.

KUVA 5. Moccamaster kahvinkeitin (14)

Esimerkki uudenlaisesta kahvinkeittimestä:

Smarter Coffee WiFi -kahvinkeitin (kuva 6) on WiFi-yhteydellä toimiva sovelluksella ohjattava kahvinkeitin. Keittimessä käytetään kahvipapuja suodatinkahvin sijasta. Halutessa voi määrittää vahvuuden ja jauhatuskarkeuden. Ensin valitaan kuppien määrä ja vahvuus, sitten vasta pavut jauhetaan välttäen hukkapavut. Vesisäiliöön menee 1,5 litraa ja siitä saa 12

kupillista. Lämpölevy pitää kahvin lämpimänä 40 minuuttia, ennen kuin se sammuu. Sisäisen anturin avulla saa tiedon, kuinka moneen kuppiin vesi riittää ja milloin täyttää vesisäiliö. Ohjaus tapahtuu joko kahvinkeittimen painikkeilla tai Smarter Appliances -sovelluksella. Sovellusta pystyy käyttämään sekä iOS- että Android-käyttöjärjestelmällä. Laitteessa on langaton yhteyden muodostus WiFi-verkkoon. Lisäksi kahvinkeittimessä on 3,5 tuuman LDC-näyttö (liquid crystal display = nestekidenäyttö). (15.)

KUVA 6. Smarter Coffee WiFi -kahvinkeitin (15)

3.1.3 Kodin valaistus

Kodin valaistukseen olennaisena osana kuuluvat lamput. Yleisimmät lamput ovat hehkulamppuja. Ennen vanhaan käytettiin paljon öljylamppuja, joiden toimivuus perustui poltettavaan öljyyn, sekä myös kaasulla toimivia lamppuja. Näitä vanhoja lamppuja käytetään vielä jonkin verran mökeillä, missä ne ovat varsin käteviä, sillä kaikissa mökeissä ei ole sähköä tai aggregaattia. Nykypäivänä on myös käytössä LED-lamppuja, jotka kestävät pidempään kuin tavalliset

hehkulamput ja valolla on eri sävyjä. Monet vaihtavat lamppunsa LEDeihin, koska ne ovat pitkäkäyttöisempiä ja ympäristöystävällisempiä vaihtoehtoja – ne myös kuluttavat vähemmän energiaa. Valoja myös hallinnoidaan erilaisilla valokatkaisimilla. On ihan tavallisia katkaisimia, sekä myös pyöreitä, jota pyörittämällä saa valoa joko kirkkaammaksi tai himmeämmäksi oman tarpeen mukaan. Yleisin valaisin vaihtoehto jälkimmäiseen katkaisijaan on upotettu valo katossa.

Valaistukseen on jo nykyään erilaisia ratkaisuja: on äänentunnistus taputuksella, himmentävä ja kirkastava valokatkaisin, aikakatkaisu jne. Tulevaisuudessa valotkin pystytään automatisoimaan ja vähentämään kulutusta. Erilaisten sensoreiden avulla pystytään mittaamaan ikkunoista tulevan valon määrä ja suhteuttaa se sisällä tarvittavaan valaistukseen.

Esimerkki valaistuksesta:

VOCCA: Voice Activated Light Bulb Adapter (kuva 7) on äänitunnistuksella toimiva lamppuadapteri. Vocca on säädetty kuuntelemaan käskyjä vain englanniksi puhuttuna. Äänikäynnistys toimii maksimissaan 4,57 metrin etäisyydeltä. Laitteessa saa käyttää vain 30 W:n lamppua tai pienempää. Lamppu toimii tavallisesti, kun käyttää valokatkaisinta. Siinä on oletusasetuksena äänikäynnistys ja se toimii heti paketista otettuna. Lamppuun ei tarvita erikseen asennusta. Siinä on tavallinen Edison-kanta E27. Vocca Pro -versiossa on mahdollista käyttää myös muita kieliä – vaikka suomea. Tähän versioon on myös sovellus, joka toimii iOS- ja Android- puhelimissa. Sovelluksessa voi äänittää omat käynnistyskäskyt ja sen jälkeen ei enää tarvitse sovellusta. Sovelluksella voi asettaa ajastukset, milloin valo menee päälle ja pois päältä. (16.)

KUVA 7. Vocca ja Vocca Pro -sovellus (17)

3.1.4 Ilmastointi

Kodin ilmastointilaitteet eivät ole joka talon yleinen laite, mutta se on yleistymään päin. Uudemmissa taloissa on ilmastointilaite, mikä maksimoi ilmanvaihdon ja jättää sisälle hyvin vähän kosteaa ilmaa. Vanhemmissa taloissa ilmastoinnin virkaa toimittavat ilmanvaihtokanavat. Koneellinen ilmanvaihto on nykyään melkein välttämätön. Ilmanvaihdon merkitys nousee, kun huomioidaan asunnossa tai talossa nouseva ilman kosteuden määrä esimerkiksi pesuhuoneessa tai oleskelutiloissa. Kostea ilma kerääntyy sisälle ja saattaa luoda rakenteisiin homeelle erinomaisen kasvualustan. Ilmastointilaite pitää huolen siitä, että ilma vaihtuu tarpeeksi useasti hävittäen kosteaa ilmaa.

Ilmastointi kuulostaa todella kalliilta ratkaisulta hävittää kosteaa ilmaa sisätiloista, mutta oikeasti se on hyvin energiatehokas ja lämmin ilma ei karkaa sisältä minnekään. Laitetta voidaan säätää manuaalisesti sopivaksi tai sitten automatisoida kokonaan, jolloin laite seuraa anturien avulla ilman kosteuspitoisuutta. (18.)

Sensoreiden avulla ilmastoinnin voi automaattisesti saada puhaltamaan viileämpää tai lämpimämpää ilmaa. Tämän ominaisuuden voi toteuttaa lämpösensoreilla, jotka havaitsevat lämpötilan muutokset ja mittaavat sitä myös asetetun ajan välein. Tavoite on pitää huoneilma mahdollisimman optimaalisessa tilassa.

3.1.5 Hälytysjärjestelmä

Rikosilmoitinjärjestelmä eli usein hälytysjärjestelmänä tunnettu käsite ilmaisee luvattoman tunkeutumisen lisäksi haluttaessa myös vesivuodot, tulipalon tai muun vastaavan poikkeaman, josta kiinteistölle tai henkilölle voi aiheutua haittaa (19).

Yleisin hälytysjärjestelmä, joka löytyy joka kodista, on palovaroitin. Suomessa palovaroitin on pakollinen varuste ja niitä täytyy olla yksi alkavaa 60 m^2 :ä kohden. Palovaroitin reagoi ilmassa olevaan savuun, lämpöön, infrapunasäteilyyn tai palokaasuihin. Varoittimet toimivat kotitalouksissa paristoilla ja niiden toiminta tulisi tarkistaa vuosittain.

Rikosilmoitinlaitteistolla valvotaan kohteeseen tunkeutumista ja siellä tapahtuvaa liikkumista. Ilmoitinkeskus lähettää ilmoituksen hälytyskeskukseen ilmoituksensiirtojärjestelmän avulla ja/tai tekee paikallishälytyksen. Hälytyskeskus rekisteröi saapuneen ilmoituksen ja ryhtyy sen johdosta ennalta sovittuihin toimenpiteisiin. (20.) Tiettyihin aikoihin on olemassa omat koodinsa kytkemiseen ja poiskytkemiseen. Hälytys menee silloin päälle, jos tietyn ajan sisään ei ole poiskytkenyt hälytystä. Tällä tavoin varmistetaan henkilöiden turvallisuus. Hälytys on yleensä myös äänetön.

Älykotiin on mahdollista saada kaikki mahdolliset toiminnot, mikäli ne ovat automatisoitavissa. Nykyaikaisissa kotijärjestelmissä on mahdollista sovelluksen

avulla nähdä, esimerkiksi ketkä ovat kotona, mitkä valot ovat päällä ja ovatko kaikki ovet lukossa. Tätä ominaisuutta tarjoavat muutamat yritykset, esimerkiksi Verisure, joka on erikoistunut kotihälytysjärjestelmiin.

3.1.6 Älypistorasia

Tavalliset pistorasiat eivät ole lähdössä minnekään taloista vaan niihin kehitellään koko ajan erilaisia lisälaitteita kuten älypistorasiat. Älypistorasiat ovat tavalliseen pistorasiaan laitettava hieman isompi pistorasia, jossa on WiFi-yhteys esim. puhelimeen. Se on puhelimella tai tabletilla ohjattava.

Esimerkki älypistorasiasta:

S20 Smart Socket (kuva 8): Tässä älypistorasiassa on yhteensopivuus EU, US ja UK pistokkeisiin. Se on langaton älypistorasia, jonka voi yhdistää minkä tahansa kodinlaitteen ja elektronisen laitteen kanssa, jossa on WiFi-yhteys. Yhteys antaa käyttäjän kaukosäätää iOS tai Android -APP eWeLinkillä (älykodin hallinnointi sovellus).

KUVA 8. Smart Socket (21)

Sen jälkeen, kun älypistorasia on lisätty sovellukseen, sillä voi ohjata päälle tai pois yhdistettyjä laitteita mistä tahansa, milloin tahansa. Sovelluksella voi myös asettaa ajastinaikatauluja, jakaa laitteen ohjauksen muille, hallinnoida ryhmää ja paikkaa. Asennus on helppo ja sen voi aktivoida saman tien. Tukee

maksimissaan 10:tä ajastustehtävää jokaisella laitteella sekä maksimissaan 150:tä WiFi-älypistorasiaa yhdestä älypuhelimesta. (21.)

Kuvassa 9 on esitelty älypistorasian teknisiä tietoja.

.Power Supply:	AC 110-240V	.WLAN Consumption:	≤0.3W
.MAX Electricity:	10A	.Sensitivity:	802.11b;-93dBm(@11mbps.CCK)
.Material :	Resistant ABS		802.11g;85dBm(@54Mbps.OFDM)
.Type:	EU Standard		802.11n;-82dBm(@HT20.MCS7)
.WLAN Standard:	WiFi2.4GHz b/g/n	.Working Temps:	-20~70℃
.WLAN Frequency:	2.412-2.484GHz	.Working Humidity:	≤80%
.Encryption Type:	WEP/TKIP/AES		

KUVA 9. Smart Socketin tietoja (21)

3.1.7 Muita toimintoja

Pikkuhiljaa on myös alettu viemään eteenpäin esimerkiksi astianpesukoneen automatisointia. Samsung on kehittänyt astianpesukoneen, joka optimoi energian kulutuksen minimiin ja puhdistaa myös itsensä. Se pystyy mittaamaan astioiden määrän ja säätää pesut sillä perusteella. Kun pesu on valmis kuivauksen jälkeen, se aukeaa automaattisesti ja päästää höyryt ulos koneesta. Samsung on myös kehitellyt uudenlaisia muita kodinkoneita ja ennen pitkää nekin varmasti saavat olla osa kotia älykodinkoneina.

Älykotiin voi kehitellä paljon erilaisia ominaisuuksia, esimerkiksi elektroninen kotiavain, mielikuvitus on vain rajana! Eri yritykset myyvät erilaisiin tarpeisiin tuotteita, joita voi mahdollisesti yhdistää muihinkin tuotteisiin oman mielen mukaan.

3.2 Kodin tulevaisuus

Älykodin ja älylaitteiden suosio perustuu niiden energiaystävällisyyteen. Suurin osa näistä laitteista kuuluu energialuokkaan A, joka kuluttaa kaikista vähiten. Ihmiset haluavat koko ajan olla ajan tasalla, mitä heidän kotonaan tapahtuu, ja näiden laitteiden avulla se on entistä helpompaa. Vanhemmat haluavat tietää,

milloin lapset tulevat koulusta kotiin tai teini-ikäinen viikonloppuna kavereiltaan, ja nykyajan sovelluksilla se onnistuu. Jos tuntuu, että kahvinkeitin jäi päälle, voi puhelimella vilkaista ja tarvittaessa katkaista virran sovelluksen avulla pistorasiasta. Näillä laitteilla on mahdollisuus ennaltaehkäistä esimerkiksi laitteista aiheutuvia tulipaloja.

On myös mahdollista, että kun lähtee pois kotoa, talo katkaisee veden ja sähkön tietyistä paikoista ja aukaisee ne taas, kun tulee takaisin kotiin. Sen voi toteuttaa helpon kytkimen avulla. Mikäli ei pidä puhelinsovelluksista, mutta haluaa varmistaa, että sähköt eivät jää päälle kotoa lähtiessään, tämä on hyvä ratkaisu.

Vielä ei ole suurta ryntäystä älykoti-markkinoille, mutta ihmiset alkavat pikkuhiljaa lisätä kotiinsa toimintoja, jotka ovat hyödyllisiä. Yksi hyödyllisistä on autotallin oven aukaisu ja kiinni laittaminen. Liikesensorit ja aikakatkaisu varmistavat helpon autolla poistumisen nousematta autosta ulos.

Yksittäisten älykkäiden laitteiden hinnat tulevat laskemaan ja sen myötä on arvioitu, että vuoteen 2018 mennessä suomalaiskodista voisi löytyä noin 40 erilaista älylaitetta. Älykkäät ratkaisut huomioidaan jo kodin suunnitteluvaiheessa. Tällä hetkellä esimerkiksi 150-neliöisen omakotitalon automatisointi voisi maksaa noin 7000−10000 euroa – kerrostalokaksion automatisointi noin puolet vähemmän. (22.)

4 YHTEENVETO

Älykodin tulevaisuus kulkee rinta rinnan teknologian kehityksen kanssa. Erilaisia tekniikoita laitteiden yhdistämiseen keksitään ja kehitetään.

Erilaiset kodinlaitteet jatkavat kehitystään koko ajan ja tavallisimmistakin laitteista tulee älykkäitä. Käyttäjän on mahdollista itse koota älylaitteita kotiinsa ja asentaa ne haluamallaan tavalla. Laitteet ovat kaikkien saatavilla esimerkiksi Verkkokauppa.comista.

Vaikka kodit ovat tällä hetkellä vielä kalliita päivittää älykodiksi, on niitten suosio silti kasvamassa. Älykodit tulevat olemaan tulevaisuuden kodin muoto.

LÄHTEET

1. What is a smart home. 2017. SmartHomeUSA. Saatavissa: http://www.smarthomeusa.com/smarthome/. Hakupäivä 2.2.2017.

2. Edwards, Jordan – Fitzpatrick, Brad 2010. Smart Homes. Saatavissa: http://www.slideshare.net/BradFitzpatrick/smart-homes-609360. Hakupäivä 2.2.2017.

3. Adam 2015. Evolution of Smart Home and IoT Through History. Appcessories. Saatavissa: http://www.appcessories.co.uk/evolution-of-smart-home-and-iot-through-history/. Hakupäivä 2.2.2017.

4. Why is Z-Wave the preferred choice for smart home iot?. 2017. Z-Wave. Saatavissa: http://z-wave.sigmadesigns.com/why-z-wave/. Hakupäivä 2.2.2017.

5. How Z-Wave Works. 2017. Z-Wave. Saatavissa: http://www.z-wave.com/about. Hakupäivä 2.2.2017.

6. How does it work. 2016. UniPi.technology. Saatavissa: https://www.unipi.technology/products/unipi-1-1-19. Hakupäivä 2.2.2017.

7. UniPi Technology. 2016. Saatavissa: https://www.unipi.technology/?gclid=CLS6i9bww84CFafacgodQssFCA. Hakupäivä 2.2.2017.

8. Buckingham, Alan 2015. The history of the smart home[Infographic]. ItProPortal. Saatavissa: http://www.itproportal.com/2015/08/25/the-history-of-the-smart-home-infographic/. Hakupäivä 2.2.2017.

9. Salmela, Marja 2016. Kalasatamassa asutaan älykodeissa – ”jos olen keskustassa, voin napsaista kahvinkeittimen päältä pois”. Helsingin Sanomat. Saatavissa: http://www.hs.fi/kaupunki/art-2000002883756.html. Hakupäivä 2.2.2017.

10. National Academy of Engineering. 2017. Household appliances timeline. Great Engineering Achievements. Saatavissa: http://www.greatachievements.org/?id=3768. Hakupäivä 2.2.2017.

11. Vacuum Cleaner History. 2017. Hubert Cecil Booth – Biography and Facts. Saatavissa: http://www.vacuumcleanerhistory.com/vacuum-cleaner-inventors/hubert-cecil-booth/. Hakupäivä 2.2.2017.

12. Look And Learn. 2012. Hubert Cecil Booth vacuumed the red carpet for Edward VII's coronation. Saatavissa: http://www.lookandlearn.com/blog/17751/hubert-cecil-booth-vacuumed-the-carpets-for-edward-viis-coronation/. Hakupäivä 2.2.2017.

13. We Are Top 10. 2017. Top 10 Best Robot Vacuum Cleaner Reviews. Saatavissa: https://wearetop10.com/best-robot-vacuum-cleaners/. Hakupäivä 2.2.2017.

14.. Moccamaster K942 AO polished silver kahvinkeitin. Huvilaite.fi Saatavissa: https://www.huvilaite.fi/Moccamaster-K942-AO-Polished-Silver-Kahvinkeitin. Hakupäivä 2.2.2017.

15. Smarter Coffee WiFi Kahvinkeitin. 2017. Coolstuff. Saatavissa: https://www.coolstuff.fi/Smarter-Coffee-WiFi-Kahvinkeitin. Hakupäivä 2.2.2017.

16. Vocca Light. Activocal. Saatavissa: http://activocal.com/product/vocca-light/. Hakupäivä 2.2.2017.

17. Vocca Light. Activocal. Saatavissa: http://activocal.com/vocca/. Hakupäivä 2.2.2017.

18. Yleistä ilmanvaihdosta. 2014. 0. Rakentaja.fi. Saatavissa: https://www.rakentaja.fi/artikkelit/6708/yleista_ilmanvaihdosta_vallox.htm. Hakupäivä 2.2.2017.

19. Rikosilmoitinjärjestelmä. Auto-alarm, Saatavissa: http://autoalarm.fi/Yritysken-turvallisuus/Rikosilmoitinjaerjestelmae. Hakupäivä 2.2.2017.

20. Koskenranta, Harri 2007. T-110. 5610 Toimitilaturvallisuus – Rikosilmoitinjärjestelmä. Sivu 7. Saatavissa: http://www.tml.tkk.fi/Opinnot/T-110.5610/2007/kalvot/rikosilmoitus-1.pdf. Hakupäivä 2.2.2017.

21. S20 Smart Socket - WiFi Smart Socket EU Plug. 2017. Itead.cc. Saatavissa: https://www.itead.cc/smart-socket-eu.html. Hakupäivä 2.2.2017.

22. “Nykyiset kolmekymppiset ovat jo valmiita” – asumisen mullistuminen tekee älykkäistä kodeista tavallisia. 2017. Mtv.fi. Saatavissa: https://www.mtv.fi/lifestyle/koti/artikkeli/nykyiset-kolmekymppiset-ovat-jo-valmiita-asumisen-mullistuminen-tekee-alykkaista-kodeista-tavallisia/6479712#gs.h2g4l9c. Hakupäivä 4.9.2017.

Hanna-Maija Ilmola

AUTOMAATIOTESTAUS OHJELMISTOTESTAUKSESSA

Liite 2

AUTOMAATIOTESTAUS OHJELMISTOTESTAUKSESSA

Hanna-Maija Ilmola
Opinnäytetyö, osa 2
Syksy 2017
Tietotekniikan tutkinto-ohjelma
Oulun ammattikorkeakoulu

SISÄLLYS

1 JOHDANTO

Tämä on kaksiosaisen opinnäytetyön toinen osa. Tässä osassa keskitytään ohjelmistotestauksessa käytettävään automaatiotestaukseen ja sen käyttöönottoon sekä eri käytäntöihin ja testien ylläpitoon.

Työn tavoitteina on dokumentoida automaatiotyökalun käyttöönotto, jonka avulla toinen työntekijä voi aloittaa tai jatkaa automatisointia sekä myös dokumentoida testikäytänteistä pohjautuen yrityksen oman tuotteen testaukseen, joka on internetsivusto. Työssä ei käytetä esimerkkinä kyseistä tuotetta.

Automaatiotestaus tarkoittaa, että käytetään automaatiotyökalua toteuttamaan erilaisia testitilanteita. Sen tavoite on vähentää manuaalisen testauksen määrää eikä eliminoida sitä kokonaan. Automaatiotestauksella voi toteuttaa samat testit kuin manuaalisestikin ja mahdollisesti myös toteuttaa testejä, joita ei välttämättä pysty toteuttamaan manuaalisesti.

Tässä opinnäytetyössä käytetään esimerkkinä automaatiotyökalua Robot Framework, joka on yleiskäyttöinen testausautomaatiokehys hyväksymistestaukselle ja hyväksymisvetoiselle ohjelmistokehitykselle (engl. Acceptance Test-Driven Development, ATDD) (1). Ohjelmalla toteutetaan tässä työssä käyttöliittymään kohdistuvaa automaatiotestausta, minkä avulla varmistetaan sivuston toimivuus.

Työ toteutetaan Keysight Technologies Finland Oy:lle jatkona yliopisto-opiskelijan pro gradu -työlle, jossa käsiteltiin automaatiotyökalun valintaa (2). Gradu ei ole julkisesti luettavissa, vaan ainoastaan Oulun yliopiston E-thesiksessä. Työ tehdään yrityksen tuotteelle Nemo Cloud, minkä testauksen perusteella dokumentti toteutetaan. Keysight Technologies Finland Oy kehittää, valmistaa ja myy Nemo-tuotemerkillä testaus- ja mittausratkaisuja langattomien matkapuhelinverkkojen peiton ja laadun mittaamiseen, optimointiin, vianetsintään ja palvelutason vertailuun.

2 AUTOMAATIOTESTAUS

Tässä luvussa käydään läpi automaatiotestauksen määritelmä ja sen merkittävyys testaukseen sekä testaustyökalu, jolla testejä voidaan tehdä.

2.1 Määritelmä

Automaatiotestaus tarkoittaa, että käytetään automaatiotestaukseen soveltuvaa ohjelmaa suorittamaan testit testattavassa ympäristössä esim. web-sivustossa tai Android-sovelluksessa. Automaatiotestauksella toteutetaan testejä, jotka saattavat olla liian vaikeita suoritettavaksi manuaalisesti, esimerkiksi testit, jotka vaativat paljon toistoja ja erilaisia ajoituksia. Manuaalista testausta tehdään kuitenkin vielä paljon, eikä automaatio ole syrjäyttänyt sitä. Koska kaikkea ei kannata automatisoida, manuaalista testausta tarvitaan, esimerkiksi virheiden korjausten tarkistamiseen tai kun sivustolle tulee paljon muutoksia.

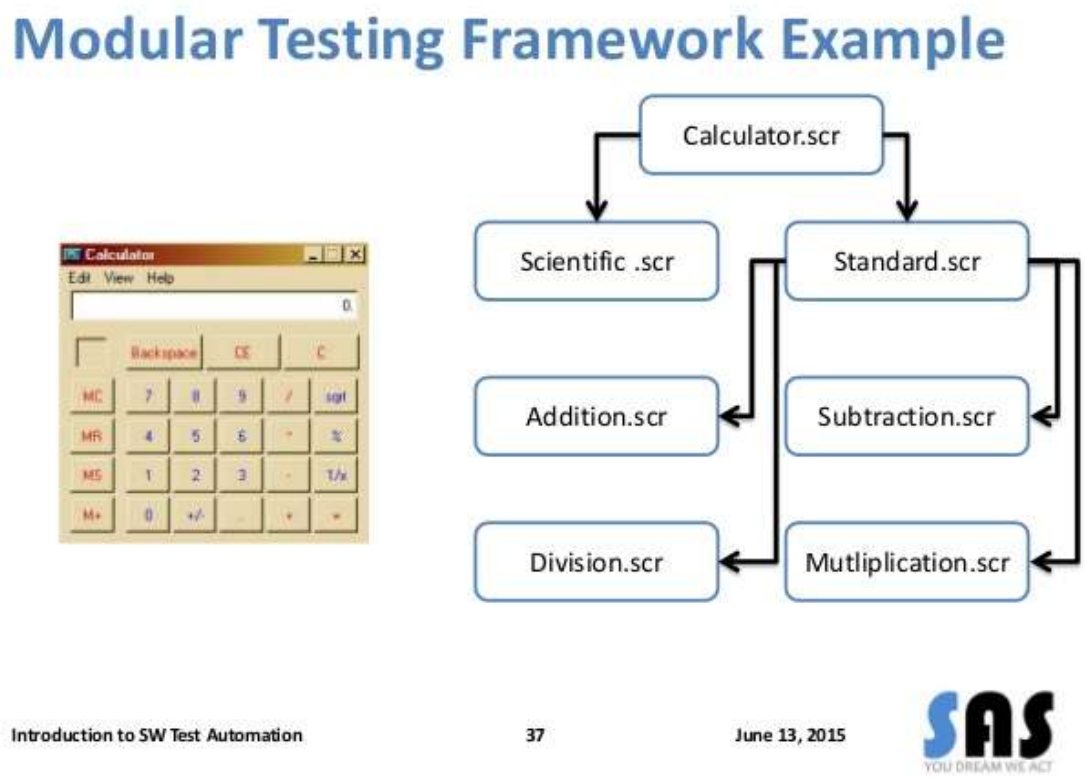
Automaatilla on erilaisia kehyksiä eli frameworkejä erilaisiin testauksiin. Frameworkin katsotaan olevan yhdistelmä protokollista, säännöistä, standardeista ja ohjesäännöistä, jotka voidaan sisällyttää tai seurata kokonaisuutena hyödyntämällä frameworkin tarjoamia telineitä. (3.) Yleinen käsitys järjestelmien kehittämisessä on koodin, datan tai koko ohjelmien käyttö, jotka on rakennettu virheenkorjaus- tai jäljitystarkoituksiin, mutta jotka eivät ole koskaan lopullisessa tuotteessa, ja tätä tekniikkaa kutsutaan telineeksi. Teline antaa ohjelmoijalle pääsyn osiin, minne he eivät muuten pääsisi. (4.)

Testiautomaatiossa framework on teline, joka on sijoitettu tarjoamaan toteutusympäristön automaation testeille. Kehys tarjoaa käyttäjälle monia etuja, jotka auttavat heitä kehittämään, toteuttamaan ja raportoimaan automaation testiskriptit (komentosarjat) tehokkaasti. Se on enemmänkin järjestelmä, joka on luotu erityisesti automatisoimaan testejä. (3.) Omin sanoin frameworkin avulla päästään käsiksi asioihin, joiden avulla pystytään toteuttamaan testejä erilaisille ympäristöille hyödyntämällä erilaisia kirjastoja. Kirjastot sisältävät eri argumentteja, joita noudattamalla testit tehdään toimiviksi. Framework on

järjestelmä, joka ei ole lopullisessa tuotteessa, mutta jolla tehdään tuotteelle testejä ja etsitään tuotteessa esiintyviä virheitä.

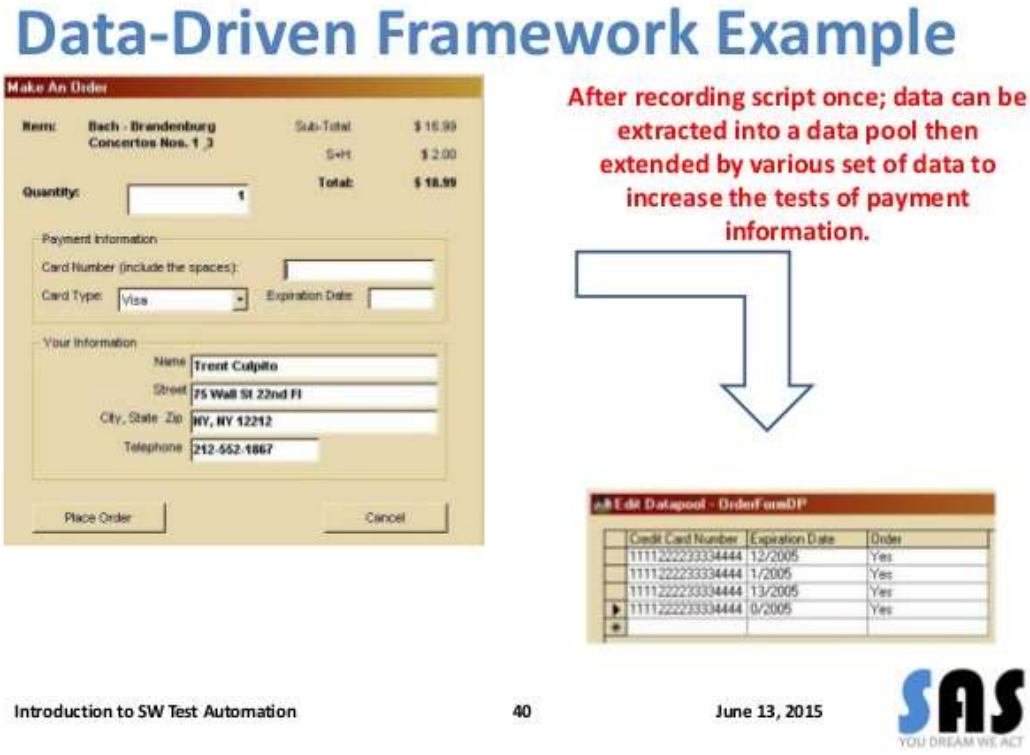
Testiautomaatio frameworkejä ovat esimerkiksi

- **Module Based Testing Framework**: jokainen moduuli tai toiminnallisuus on erotettu ja käsitelty itsenäisesti. Tämä vaatii pienien omatoimisten skriptien luomista, jotka edustavat moduuleja, osia ja funktioita (kuva 1). (3.)



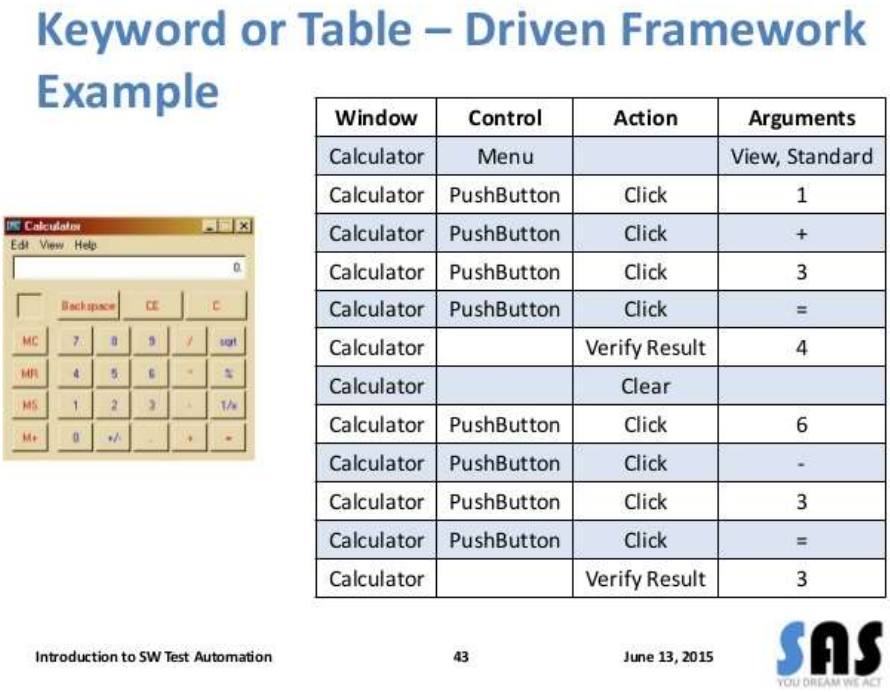
KUVA 1. Esimerkki Modular Testing Frameworkistä (5)

- **Data Driven Testing Framework**: auttaa käyttäjää erottamaan testiskriptin ja testitiedot toisistaan. Se antaa käyttäjän tallentaa testitiedot ulkoiseen tietokantaan (kuva 2). (3.)



KUVA 2. Esimerkki Data-Driven Frameworkistä (5)

- **Keyword Driven Testing Framework:** sopii manuaaliseen sekä automaatiotestaukseen. Se käyttää avainsanoja symboloimaan toiminnallisuutta, jota testataan (kuva 3). (3.)



KUVA 3. Esimerkki Keyword Driven Frameworkistä (5)

Keysightin tuotteen testaukseen soveltuu parhaiten Keyword Driven Framework. Frameworkillä pystyy tekemään testejä, jotka simuloivat oikean käyttäjän toimintatapoja sivustolla. Avainsanojen avulla pystytään toteuttamaan käyttäjäläheisiä testejä.

2.2 Merkitys

Testaus on tärkeä vaihe kehitysprosessissa. Sillä varmistetaan, että onko kaikki virheet löydetty ja korjattu ja että tuote, ohjelma tai laite, toimii kuten sen pitäisi tai se on lähellä oikeaa lopputulosta. (6.)

Automaatiotestaus säästää aikaa ja rahaa, kun testit tehdään tehokkaammiksi. Se myös parantaa testien tarkkuutta verrattuna ihmisten suorittamiin testeihin. Kattavuus kasvaa, koska monet testaustyökalut voidaan ottaa käyttöön kerralla, ja se myös auttaa kehittäjiä löytämään virheitä nopeampaa. (6.) Testejä on myös mahdollista ajaa yöllä ja katsoa myöhemmin raporteista läpi menneet ja epäonnistuneet testit.

Automaatiotestaus on hyvä vaihtoehto, jos tehdään esimerkiksi

- **regressiotestausta**: uudelleen testaus olemassa olevalle sovellukselle, jos siirretään uudempiin versioihin.
- **smoke-testausta**: nopeaan high level -arvioimiseen koontiversion (ohjelman versio ns. esi-julkaisu, johon sovelluksen komponentit kerätään ja toistuvasti kootaan testaustarkoituksiin (7)) laadusta.
- **staattista ja toistuvaa testausta**: testaus toistuvaa ja suhteellisen muuttumattomia yhdestä testikierrosta seuraavaan.
- **data-driven testausta**: applikaation funktioiden testaukseen, missä samat funktiot tarvitsevat vahvistuksen eri inputeilla ja suurilla data määrillä.
- **load- ja performance-testausta**: ei ole olemassa mahdollista manuaalista vaihtoehtoa. (8.)

Yleisintä testausta automaatiolla on regressio testaus. Testit tehdään tuotteelle, jota päivitetään tietyn ajan välein ja automaatiotestejä ajetaan varmistamaan tuotteen toimivuus päivityksen jälkeen. (8.)

2.3 Robot Framework

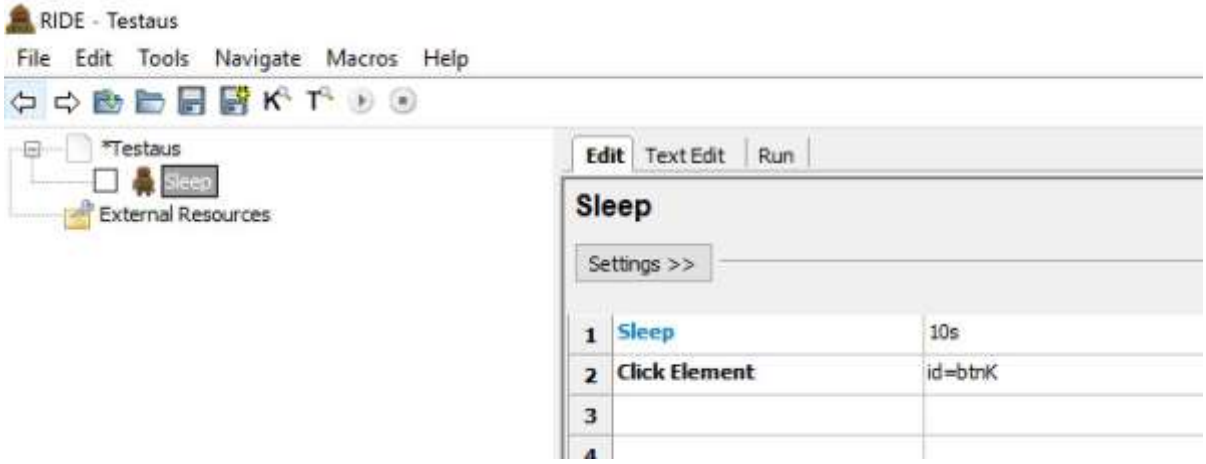
Robot Framework itse on avoimen lähdekoodin ohjelma (käyttäjällä mahdollisuus tarkastella ja muokata lähdekoodia omien tarpeiden mukaisesti), joka on julkaistu Apache Licence 2.0:n alla ja suurin osa sen kirjastoista ja työkaluista ekosysteemissä ovat myös avoimia lähdekoodeja. Kehyksen oli alun perin kehittänyt Nokia Networks ja nykypäivänä Robot Framework Foundation sponsoroi sitä. (1.)

Robot Frameworkillä on helppokäyttöinen taulukkomainen testaustietojen syntaksi ja se käyttää avainsanalähtöistä testausmenetelmää. Sen testauskapasiteettiä voi laajentaa testikirjastoilla, jotka on toteutettu Pythonilla tai Javalla, ja käyttäjät voivat luoda uuden korkeamman tason avainsanoja olemassa olevista ja käyttää samaa syntaksia, jota on käytetty testitapauksien luomiseen. (1.) Kuvassa 4 näkyy valmiiksi sisältyvät standardikirjastot ja näiden lisäksi on myös erikseen asennettavia kirjastoja, joihin kuuluu

mm. Selenium2Library, jota käytetään internetsivustoihin liittyvissä testeissä. Kyseinen kirjasto löytyy external-kirjastoista.

KUVA 4. Robot Frameworkin kirjastoja (1)

Testejä voidaan luoda esimerkiksi RIDE:llä (9) (kuva 5), joka on kehitysympäristö Robot Frameworkin testitapauksille. Ohjelma sisältää valmiiksi kirjaston BuiltIn, joka sisältää tavallisimpia avainsanoja, esimerkiksi **Sleep**, joka pysäyttää testin määritetyksi ajaksi. Kuvassa näkyy myös avainsanojen väreistä, onko avainsanan kirjastoa liitetty testiin vai ei: sinisellä oleva on ja mustalla oleva ei.



KUVA 5. RIDE-kehitysympäristö (10)

Robot Framework on keyword driven framework (ks. luku 2.1). Yksinkertaisuudessaan käytetään avainsanoja, jotka kuvaavat toimintaa, esimerkiksi **click element**, joka tarkoittaa, että painetaan tiettyä elementtiä, mikä löydetään avainsanaan liitetyn paikannustavan avulla, esimerkiksi xpath (XML Path Language) tai id (ks. kuva 5).

3 TESTIOHJELMAN KÄYTTÖÖNOTTO, KÄYTÄNTEET JA YLLÄPITO

Tässä luvussa käsitellään testauksessa käytettävän ohjelmiston asennus, testikäytänteitä sekä niiden ylläpitoa.

3.1 Käyttöönotto

Robot Frameworkin pystyy lataamaan ilmaiseksi tietokoneelle. Siihen lisäksi kuuluvat osat ladataan kaikki 32-bittisenä. Tiedostot voidaan ladata myös 64-bittisenä, mutta jotkut kirjastot kuten AutoItLibrary tarvitsee 32-bittisen Pythonin toimiakseen. Täytyy valita, kumpaa haluaa käyttää ja ladata kaikki tiedostot sama bittisenä. Näin varmistetaan osien yhteensopivuus testien ajon kannalta. Asennusohje suomeksi ja englanniksi löytyy liitteestä 1. Asennusohjeessa kerrotaan, miten ladataan Python 2.7, Robot Framework, wxpython, RIDE ja Selenium2Library eli kaikki tarvittavat osat testien toteutukseen.

Kun on saanut asennettua Robot Frameworkin kirjastoineen, voi RIDE:n aukaista command promptin kautta käskyllä ride.py. Ennen kuin aloittaa elementtien etsimisen sivustolta, määritetään testin asetuksiin käytettävä kirjasto. Kun puhutaan internetsivustosta, käytetään kirjastoa Selenium2Library, ja jos kyseessä olisi puhelinsovellus, voitaisiin käyttää kirjastoa AppiumLibrary. Selenium2Libraryn avainsanat löytyvät Robot Framework-sivustolta, joka ohjaa oikeaan paikkaan. Kun kirjasto on liitetty testiin, voi aloittaa testin rakentamisen. Rakentamisen voi aloittaa ilman kirjastoa, mutta oikeita avainsanoja ei löydy ja testi ei toteudu ilman sitä.

3.2 Testikäytänteet ja ylläpito

3.2.1 Elementtien locatorit

Tärkeä osa testien toteutumisessa on se, että kaikki painikkeet ja tekstit sekä muut tarvittavat elementit löytyvät sivustolta helposti. Testien teko perustuu käyttäjän liikkeisiin sivustolla, joten tarvitaan tiedot, mihin halutaan syöttää tekstiä

tai mitä halutaan painaa ja mistä kohti. Sen takia olisi tärkeää, että niiden paikannus, joka tapahtuu locatorin (paikannin) avulla, on suurimmaksi osaksi muuttumaton. Kuvassa 6 on esimerkkejä eri locatoreista, joilla voi löytää oikean elementin sivustolta. Id on yleensä sellainen locator, joka ei muutu useasti, ja vaikka painikkeen paikka muuttuisi, id pysyisi samana ja sen pystyisi silti löytämään sivustolta.

Toinen vaihtoehto on käyttää elementin nimeä, mikä on melkein sama asia kuin id. Kaikki tämä riippuu kuitenkin siitä, minkälaisesta elementistä on kyse ja mitä locatoria kannattaa käyttää, koska kaikissa ei ole mahdollista käyttää id:tä tai nimeä. Joissain tapauksissa elementissä olevan tekstin kautta elementti löytyy helpoiten. Tekstiä voidaan käyttää xpathin avulla.

Strategy	Example	Description
identifier	Click Element | identifier=my_element	Matches by @id or @name attribute
id	Click Element | id=my_element	Matches by @id attribute
name	Click Element | name=my_element	Matches by @name attribute
xpath	Click Element | xpath=//div[@id='my_element']	Matches with arbitrary XPath expression
dom	Click Element | dom=document.images[56]	Matches with arbitrary DOM express
link	Click Element | link=My Link	Matches anchor elements by their link text
partial link	Click Element | partial link=y Lin	Matches anchor elements by their partial link text
css	Click Element | css=div.my_class	Matches by CSS selector
class	Click Element | class=my_class	Matches by class name selector
jquery	Click Element | jquery=div.my_class	Matches by jQuery/sizzle selector
sizzle	Click Element | sizzle=div.my_class	Matches by jQuery/sizzle selector
tag	Click Element | tag=div	Matches by HTML tag name
default*	Click Link | default=page?a=b	Matches key attributes with value after first '='

KUVA 6. Selenium2Libraryn locatorit elementtien löytämiseen (11)

Huonoin vaihtoehto (silti joskus ainoa) on käyttää indeksejä suoraan sellaisinaan xpathin kautta, esimerkiksi

xpath=.//html/body/div[4]/div[3]/form/div[2]/div[3]/center (löytyy Googlen etusivulta).

Joskus elementeille ei löydy nimeä, id:tä tai tekstiä ja silloin täytyy käyttää suoraan sivulta saatua xpathia. Huono puoli tässä on se, että jos yksi asia sivustolla vaihtuu, niin silloin paikannus ei välttämättä toimi. Jos jotain muuttuu testattavan sivuston käyttöliittymässä (UI), niin silloin xpath-indeksi muuttuu. Tällaisia locatoreita kannattaa vältellä parhaan mukaan testien ylläpitävyyden

varmistamiseksi. Jos kaikki locatorit ovat luotettavia, ovat testitkin jatkossa helpompia ylläpitää ja niiden muokkaamiseen ei kulu niin paljon aikaa.

3.2.2 Setup ja avainsanat

Jokaiseen testiin pystyy määrittämään suite setupin (kokoonpanoasetus) ja suite teardownin (purku). Suite setup tapahtuu aina ensin, ennen kuin aloitetaan tekemään itse testiä. Esimerkiksi jos testissä kirjaudutaan sivustolle sisään ja toteutetaan loppu testi siellä, voi suite setupiksi määrittää esimerkiksi selaimen aukaisemisen ja sivustolle kirjautumisen,ja tämä tapahtuisi joka kerta, kun testi ajettaisiin. Suite teardown puolestaan tapahtuu testin loputtua, esimerkiksi sivustolta uloskirjautuminen ja selaimen sulkeminen. Molemmat määritellään avainsanojen avulla. Kuvassa 7 näkyy asetuksiin määritetyt suite setup ja teardown.

KUVA 7. Suite setup ja teardown asetuksissa (10)

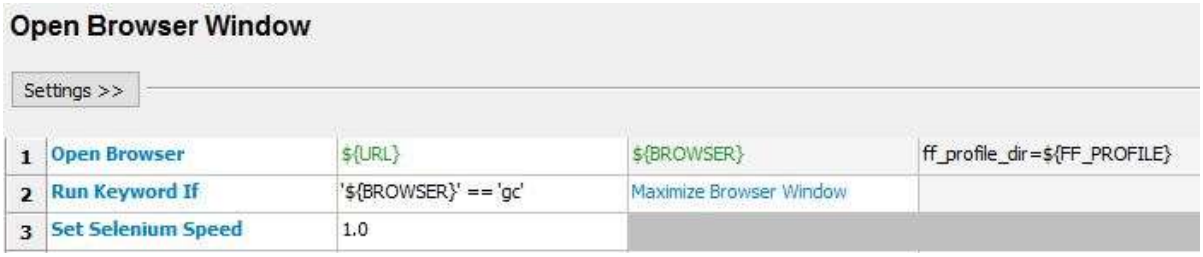
Kuvassa 8 näkyy Open Window And Login suite setupin sisältö, joka koostuu kahdesta eri avainsanasta, jotka ovat Open Browser Window ja Login.

KUVA 8. Suite setupin sisältö (10)

Mikäli kansion alla olevat testit liittyvät samaan sisältöön, riittää kun vain kansiolle määritellään suite setup, ja se tapahtuu ennen kuin aloittaa kansioon sisältyviä testitapauksia. Sillä ei ole väliä, kuinka monta testitapausta kansion alla on. Testitapauksia voi olla yli 10, mutta suositeltava määrä on alle 10. Jos kansio

sisältää viisi testitapausta, riittää yksi suite setup ja teardown, jotka suoritetaan vain kerran eikä jokaiselle testitapaukselle erikseen. Jokaiselle testille ei siis tarvitse tehdä samoja toimintoja testin alussa ja lopussa. Aina kun testi loppuu, siirrytään seuraavaan ilman ylimääräisiä sisäänkirjautumisia tai selaimen aukaisemista eli seuraava testi jatkaa siitä, mihin edellinen jäi. Suite setup ja teardown voidaan myös määrittää jokaiseen testiin erikseen, mikäli tarve vaatii.

Avainsanojen avulla testeistä tulee selkeämpiä. Kun testin osat ovat jaettu järkeviin osiin, on virheetkin helpompi löytää raporteista ja itse testistä. Avainsanat voivat olla yksittäisiä toimintoja tai koostua monesta yksittäisestä toiminnosta eli avainsana voi sisältää toisia avainsanoja. Kuvassa 9 on Open Window And Login (ks. kuva 8) avainsanan Open Browser Window sisältö. Tämä avainsana sisältää monta yksittäistä toimintoa.



KUVA 9. Avainsanan sisältö, joka koostuu yksittäisistä toiminnoista (10)

Kuvassa näkyvät ${URL} ja ${BROWSER} ovat muuttujia, jotka ovat määritelty testin asetuksiin tai testin sisältävän kansion asetuksiin. Samoja muuttujia voidaan käyttää monessa testissä ja riittää, että ne määritellään kerran. Kun arvoa muutetaan, päivittyy se joka paikkaan, missä lukee ${URL} tai ${BROWSER}. Ylläolevan avainsanan kaltaiset helpottavat ja nopeuttavat testien tekoa, sillä samoja avainsanoja voidaan käyttää vaikka jokaisessa testitapauksessa ja se on myös suositeltavaa.

3.2.3 Ylläpito

Ylläpidossa kannattaa keskittyä automatisoimaan oikeita asioita. Jos haluaa pitää ylläpidon kustannukset alhaisina, täytyy harkita, mille ominaisuuksille testit tehdään. Kun kirjoittaa testit oikeille asioille, se vähentää ylläpidettävyyden kustannuksia. Ei myöskään kannata automatisoida vakaita tai vähäarvoisia

ominaisuuksia. Täytyy ymmärtää, mille ominaisuuksille ei kannata tehdä automaatiotestejä, ja se auttaa vähentämään ylläpidon kustannuksia. (12.)

Taulukon avulla pystyy seuraamaan testien kulkua ja sitä pystyy myös päivittämään helposti. Taulukoiden kannattaa kuitenkin olla pieniä ja tehokkaita (kuva 10). (12.) Testaaja luo itse taulukon tukemaan testausta, jos se on tarpeellinen. Ohjelmat eivät luo automaattisesti testien aikana taulukoita. Taulukon avulla pystyy seuraamaan tilannetta ja pysymään aikataulussa.

OPTIMIZED CONFIGURATION MATRIX

	IE9	IE10	IE11	FF	Chrome
Win7	X		X		
Win8		X			
Win8.1			X	X	
Win10			X		X
Server2012			X		

KUVA 10. Esimerkki taulukosta (12)

Jos kyseessä olisi esimerkiksi internetsivuston testaus, taulukko voisi olla sellainen kuin kuvassa 11. Merkitään sivuston eri versiot ja selaimet, joilla sitä halutaan testata, sekä myös niiden versiot. Yksinkertaisella taulukolla saadaan käsitys siitä, mitä on testattu ja mitä ei, sekä myös mikä selaimen versio on toiminut sivuston versioilla, mikäli jatkossa tietyllä selaimen versiolla ei sivusto toimi kunnolla. Taulukkoon voi myös lisätä esimerkiksi päivämäärän ja tehtävät, mikäli se tuntuu tarpeelliselta.

Sivuston versio	Internet Explorer 11.0	Google Chrome 62.0	Firefox 56.0
1.2	X	X	
1.5			X
2.0		X	

KUVA 11. Esimerkki yksinkertaisesta taulukosta

Monimutkaisia testiskenaarioita kannattaa välttää. Ylläpidettävät testikokoonpanot keskittyvät yksinkertaisuuteen ja joustavuuteen. Sisäisten ehtojen (IF-THEN-ELSE) välttäminen on tärkeää, sillä ne tekevät testeistä vaikeampia testattavia ja luettavia. Hyvät automaatiotestit ovat erityisiä. Ne tarkistavat yhden asian ja se tarkistetaan hyvin. Testitapauksien sekoitus pidemmäksi skenaarioksi voi olla usein riskialtis. (12.)

Helposti muuttuvat locatorit ovat yleisin syy testien epäonnistumiseen. Testattavaan ympäristöön kannattaa tutustua kunnolla ja oppia, mitkä locatorit toimivat parhaiten ympäristössä ja mitkä eivät ole hyviä vaihtoehtoja. On myös tärkeää tutustua, miten kaikki tekijät toimivat ympäristössä ja miten niitä voi hyödyntää. Kehittäjät ovat iso apu, kun selvitetään locatoreita sivustolta. He voivat tehdä hyviä id-arvoja, mihin niitä on mahdollista tehdä, ja auttaa myös muissa locatoreissa, kun id ei ole mahdollinen. Kunnon locatorit säästävät paljon aikaa ylläpidossa. (12.)

Epäsynkroniset toiminnot (esimerkiksi elementin lataus) sivustolla tai sovelluksessa voivat aiheuttaa suuria ylläpito vaikutuksia (12). Esimerkiksi jos sivustolla on video, joka normaalisti latautuu alle viiden sekunnin, mutta se lataus kestääkin normaalia kauemmin, testit eivät osaa varautua tähän ja testi silloin epäonnistuu. Testit eivät osaa poikkeavuuden tullen reagoida siihen vaan yrittävät jatkaa seuraavaan vaiheeseen, ja kun se ei onnistu, epäonnistuu testi. Näiden ongelmien tehokas ratkaiseminen tarkoittaa, että täytyy oppia, miten järjestelmä ja sovellus käsittelee UI-päivitykset (12). Täytyy tehdä paljon

yhteistyötä kehittäjien kanssa ja selvittää, miten pystyy ratkaisemaan ongelmat ja miten he voivat tehdä auttavia toimintoja (12).

Automatisoi siis toimintoja, jotka ovat 80-prosenttisesti vakaita ja muuttumattomia. Kannattaa työskennellä kehittäjien kanssa eri elementtien nimeämisistä, että ne pysyisivät jatkossa muuttumattomina tai tieto kulkisi testaajan ja kehittäjän välillä muutoksista. Testeistä kannattaa myös tehdä mahdollisimman yleiskäyttöisiä. (8.)

4 ESIMERKKITAPAUKSIA

Tässä luvussa esitellään muutama esimerkki yksinkertaisista testeistä, jotka voidaan toteuttaa käyttämällä Robot Frameworkiä. Testisivustoina käytetään Googlea, Ebaytä ja Wikipediaa.

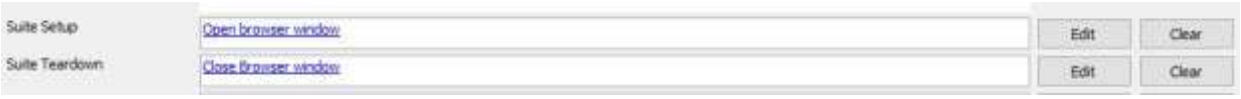
4.1 Alkumääritykset

Heti alkuun testille määritellään käytettävä kirjasto ja jos tarvitsee niin myös lähteitä, esimerkiksi tekstitiedosto, missä määritellään eri muuttujia tai avainsanoja (ks. kuva 12).

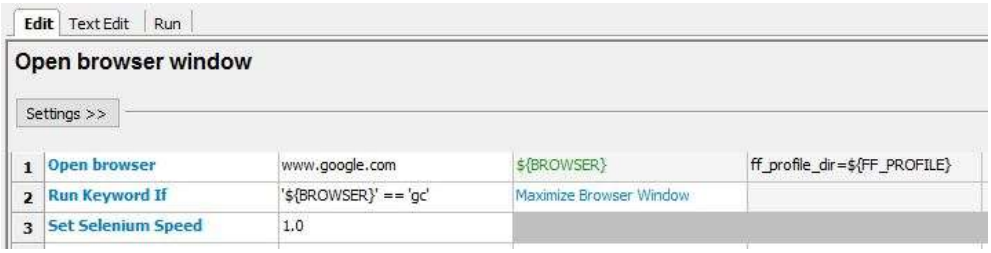
KUVA 12. Testiin määritetty kirjasto sekä kaksi muuttujaa (10)

Kuvassa näkyvä FF_PROFILE on Mozilla Firefoxin selaimen profiili, jota käytetään ja jonne tallentuu kaikki historia. Profiilin tarvitsee siihen, että testit lähtevät käyntiin. Toisena muuttujana kuvassa näkyy ${BROWSER}, jolla määritetään käytettävä selain testien toteutukseen ja nyt siihen on määritetty Mozilla Firefox. Tämä muuttuja on myös mahdollista määritellä suoraan testiin.

Heti alkuun voi määrittää testin asetuksissa testille suite setupin sekä suite teardownin (ks. kappale 3.2.2). Kuvassa 13, 14 ja 16 näkyy suite setup sekä teardown ja niiden sisällöt.

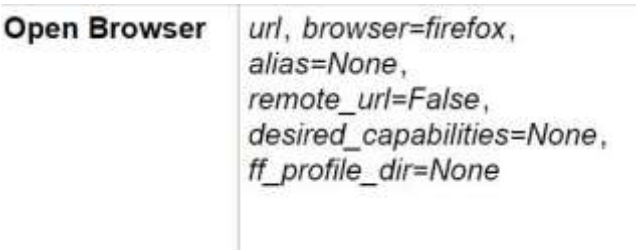


KUVA 13. Suite setup ja suite teardown (10)



KUVA 14. Suite setup avainsanan sisältö (10)

Open browser window -avainsanan sisältä löytyy yksittäisiä toimintoja, joilla määritellään ensin, mille sivulle selain aukaistaan, millä selaimella ja millä profiililla. Avainsanoihin on määritetty, missä järjestyksessä määritellään argumentit (esimerkiksi locator) avainsanalle. Kuvassa 15 näkyy, miten argumentit näkyvät kirjastossa. Jos argumentit eivät ole oikeassa järjestyksessä testi epäonnistuu.



KUVA 15. Open Browser avainsanan määritys (11)

Sen jälkeen siirrytään seuraavaan vaiheeseen, jossa tarkistetaan, onko selain Google Chrome ja jos on, selain suurennetaan maksimaaliseen kokoon. Lopuksi määritetään testin toteutuksen nopeus.

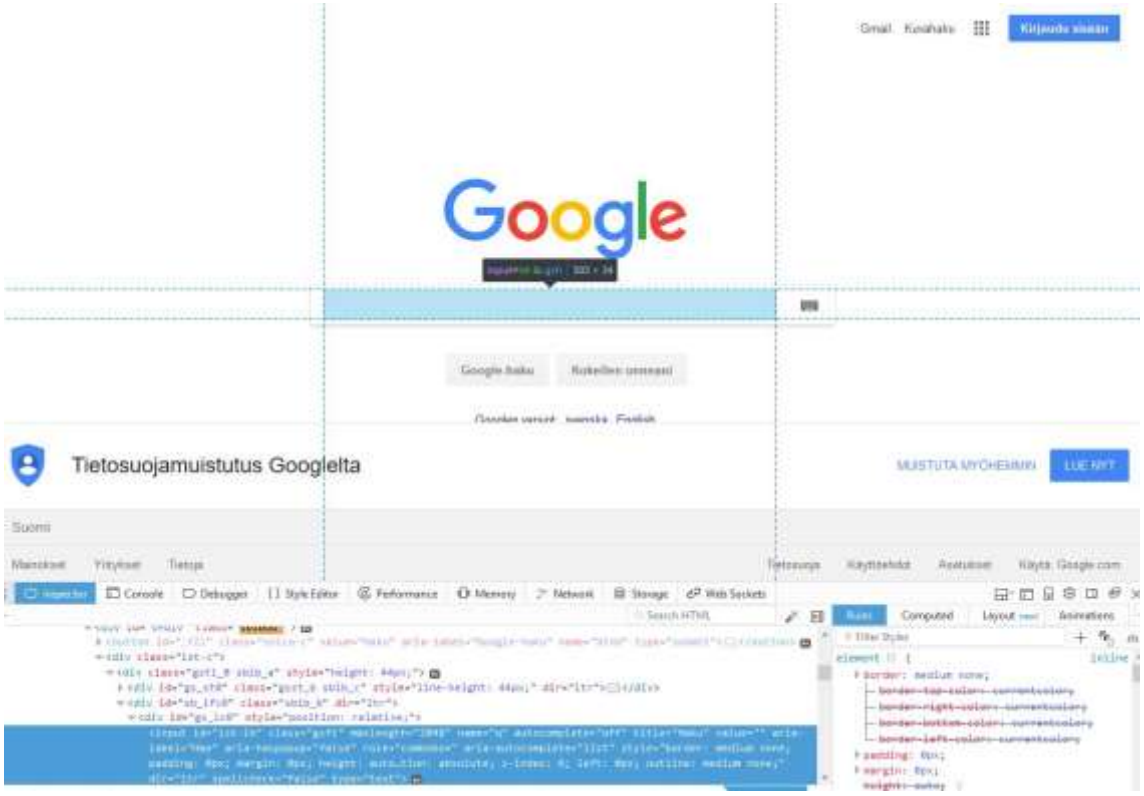
KUVA 16. Suite teardown avainsanan sisältö (10)

Suite teardownissa tehdään yksinkertainen vaihe, joka on selaimen sulkeminen.

4.2 Esimerkki 1

Ensimmäisessä esimerkissä tehdään Googlessa haku, ja siirrytään ensimmäiselle sivustolle luettelossa.

Xpath ja id löytyy internetsivustolta painamalla oikeaa hiiren painiketta elementin kohdalla ja valitsee ”Inspect element”. Kuvassa 17 on näkymä, kun on painanut kyseistä nappia. Elementti etsittiin Googlen hakukentästä, joka on kuvassa korostettuna sivustolla sekä koodissa.



KUVA 17. Inspect elementin takaa löytyvä tieto (13)

Koodista löytyvällä id:llä pystytään määrittämään, mitä hiirellä sivustolla painetaan tai minne kirjoitetaan tekstiä. Googlen hakukentässä id on ”lst-ib”. Kuvassa 18 näkyy Googlessa toteutettu valmis testi.

Edit | Text Edit | Run

Google

Settings >>

1	Input Text	id=lst-ib	kukka
2	Click Element	name=btnK	
3	Click Link	https://fi.wikipedia.org/wiki/Kukka	#href
4	Wait Until Page Contains	Kukka	

KUVA 18. Googleen tehty testiskripti (10)

Kun suite setup on toteutettu, alkaa testi Googlen etusivun näkymästä (ks. kuva 14, suite setup). Ensimmäisessä kohdassa kirjoitetaan Googlen hakukenttään sana kukka. Locatorina toimii hakukentän id. Toisessa kohdassa klikataan elementtiä nimeltä btnK, joka on Google-haku -painike. Kuvassa 19 näkyy Click element -avainsanan kuvaus. Samalla tavalla on muutkin avainsanat määritelty kirjastoille. Kuvauksessa kerrotaan käytettävät argumentit, toiminnon kuvaus, sekä millä locatorilla elementti löytyy.

Click Element	*locator*	Click element identified by *locator*. Key attributes for arbitrary elements are *id* and *name*. See *introduction* for details about locating elements.

KUVA 19. Click element -avainsanan kuvaus (11)

Kun siirrytään seuraavalle sivulle, klikataan linkkiä, joka on määritetty hrefin (hypertext reference, linkki toiseen sivustoon) avulla. Lopuksi odotetaan niin kauan, kunnes sivulle ilmestyy teksti kukka, minkä jälkeen tapahtuu suite teardown ja selain suljetaan. Tämän jälkeen RIDE:lle ilmestyy tiedot testistä ja sen onnistumisesta (kuva 20).

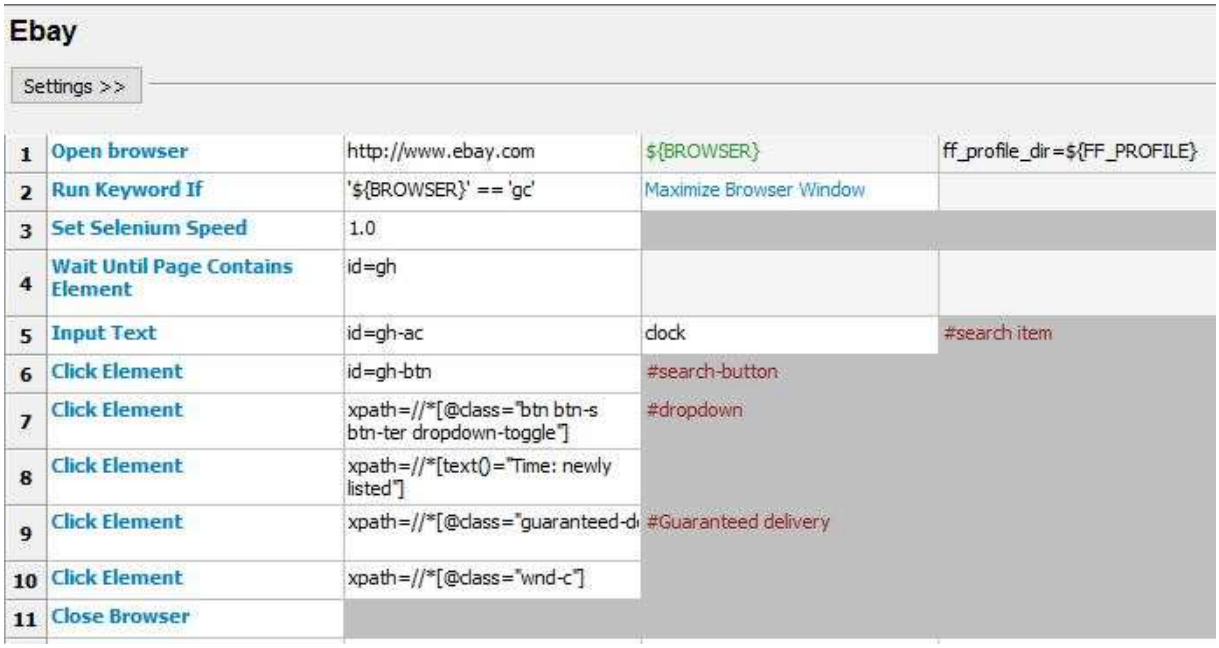
```
elapsed time: 0:00:30   pass: 1   fail: 0
command: pybot.bat --argumentfile c:\users\hanilmol\appdata\local\temp\RIDEtexs3w.d\argfile.txt --listener C:\Python27\
==============================================================================
Test
==============================================================================
Google                                                                | PASS |
------------------------------------------------------------------------------
Test                                                                  | PASS |
1 critical test, 1 passed, 0 failed
1 test total, 1 passed, 0 failed
==============================================================================
Output:  c:\users\hanilmol\appdata\local\temp\RIDEtexs3w.d\output.xml
Log:     c:\users\hanilmol\appdata\local\temp\RIDEtexs3w.d\log.html
Report:  c:\users\hanilmol\appdata\local\temp\RIDEtexs3w.d\report.html
```

KUVA 20. Testin tiedot (10)

4.3 Esimerkki 2

Toisessa esimerkissä tehdään tuotehaku Ebayssä. Tässä esimerkissä jätetään toteuttamatta suite setup ja suite teardown.

Elementit etsitään sivustolta samalla tavalla kuin esimerkissä 1, eli käyttämällä ”inspect element” -toimintoa. Tästä testistä huomaa, kuinka työlästä olisi määrittää jokaisen testin alkuun suite setup ja suite teardown erikseen avainsanoilla, kuten kuvassa 21.



Ebay

Settings >>

1	Open browser	http://www.ebay.com	${BROWSER}	ff_profile_dir=${FF_PROFILE}
2	Run Keyword If	'${BROWSER}' == 'gc'	Maximize Browser Window	
3	Set Selenium Speed	1.0		
4	Wait Until Page Contains Element	id=gh		
5	Input Text	id=gh-ac	clock	#search item
6	Click Element	id=gh-btn	#search-button	
7	Click Element	xpath=//*[@class="btn btn-s btn-ter dropdown-toggle"]	#dropdown	
8	Click Element	xpath=//*[text()="Time: newly listed"]		
9	Click Element	xpath=//*[@class="guaranteed-d	#Guaranteed delivery	
10	Click Element	xpath=//*[@class="wnd-c"]		
11	Close Browser			

KUVA 21. Esimerkki 2 testiskripti Ebay-sivustolle (10)

Aluksi tehdään samat toiminnot kuin esimerkin 1 suite setupissa. Tässä testissä vain siirrytään suoraan Ebayn sivustolle. Ensin sivustolla odotetaan, että sivuston header (ylätunniste), missä Ebay-logo sijaitsee, on näkyvissä ennen kuin siirrytään seuraavaan vaiheeseen. Hakukenttään kirjoitetaan sana ”clock” (kello) ja painetaan ”search”-painiketta.

Kohdassa 7 (ks. kuva 21) locatorina toimii xpath, ja sen avulla etsitään luokka, jonka nimi on ”btn btn-s btn-ter dropdown-toggle”. Tämä on vahva vaihtoehto id:lle, mikäli id:tä ei löydy kyseiseltä elementiltä. Tämä elementti on lista, josta valitaan tuotteiden lajittelutyyli. Lajittelutyyliksi valitaan sitten ”Time: newly listed”, eli listataan lisäysajan perusteella. Lopuksi vielä kohdassa 9 klikataan painiketta,

jossa lukee ”Guaranteed 3 day delivery” (luvataan kolmen päivän toimitus), ja painetaan aukeava ikkuna pois, mihin määriteltäisiin maa ja postinumero. Lopuksi suljetaan selain.

Ilman suite setupia ja suite teardownia joudutaan jokaiseen testiin kirjoittamaan samat toiminnot, jotka käyvät muihinkin testeihin. Tekemällä toiminnoille avainsanat säästetään paljon aikaa ylläpidossa ja testien teossa.

4.4 Esimerkki 3

Kolmannessa esimerkissä etsitään ja aukaistaan Wikipedia-artikkeli, siirrytään toiselle sivulle tekstistä ja sitten vielä siirrytään satunnaiseen artikkeliin.
Kuvassa 22 on Wikipediaan tehty testi vaiheineen.

Wikipedia

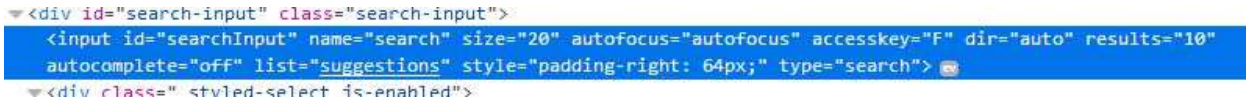
Settings >>

1	Open browser	http://www.wikipedia.org	${BROWSER}	ff_profile_dir=${FF_PROFILE}
2	Run Keyword If	'${BROWSER}' == 'gc'	Maximize Browser Window	
3	Set Selenium Speed	1.0		
4	Comment	Click Button	id=js-lang-list-button	
5	Select From List	id=searchLanguage	Suomi	
6	Input Text	name=search	Evoluutio	
7	Click Button	xpath=//*[@class="pure-button pure-button-primary-progressive"]		
8	Page Should Contain	Evoluutio viittaa sukupolvien myötä tapahtuviin		
9	Click Link	xpath=//a[@title="Sukupolvi"]		
10	Page Should Contain	Sukupolvi tarkoittaa		
11	Click Element	id=n-randompage		
12	Sleep	3s		
13	Close Browser			

KUVA 22. Wikipediassa toteutettu testi (10)

Tähän testiin ei myöskään tehty suite setupia tai suite teardownia. Esimerkkinä testistä löytyy kohdasta 4 vaihe, joka on kommentoitu. Kommentoitu vaihe näkyy testissä ja myös testiraportissa, mutta sitä ei toteuteta testissä, vaan sen yli hypätään seuraavaan.

Kun testissä siirrytään sivulle, ensin tehdään kielenvaihdos englannista suomen kieleen. Hakukenttään kirjoitetaan sitten evoluutio. Kuvassa 23 näkyy, millainen tieto hakukentän takaa löytyy ja mitä kaikkea se sisältää.



KUVA 23. Elementin sisältö (14)

Koodiin on kirjoitettu elementin tyyppi, id, nimi, koko ja muita tietoja. Sopiva locator on määritetty avainsanaan, ja sen perusteella löytää toimivan keinon löytää elementti. Tällä kertaa käytettiin nimeä, mikä näkyy kuvassa tekstinä *name=”search”*.

Seuraavaksi painettiin haku-painiketta ja siirryttiin seuraavalle sivulle, joka oli artikkeli evoluutiosta. Robot tarkistaa sivustolta, että se sisältää tekstin ”Evoluutio viittaa sukupolvien myötä tapahtuviin”, ja jos tekstiä ei löydy, testi epäonnistuu. Tällä tavalla voidaan varmistaa, että oikea sivu on näkyvissä.

Kuvassa 24 näkyy vaiheessa 9 (ks. kuva 22) käytetyn *xpath=//a[@title=”Sukupolvi”]* takaa löytyvä tieto.

KUVA 24. Kohta 9 locatorin takana (14)

Linkillä ei ole id:tä, nimeä eikä luokkaa, joten täytyy käyttää erilaista lähestymistä. Kuvassa määritellään href-linkki ja sen otsikko, minkä perusteella pystytään löytämään kyseinen linkki tekstin seasta. Xpath locatorilla siis etsitään href-linkkiä, jonka otsikko on sukupolvi ja klikataan sitä.

Kun linkkiä on klikattu, ja siirrytty seuraavalle sivulle, sivuston pitäisi sisältää teksti ”Sukupolvi tarkoittaa”. Erilaisilla odottavilla tai tarkistavilla avainsanoilla varmistetaan se, että ollaan oikealla sivustolla ja myös, että sivusto toimii testin mukaisella tavalla.

Lopuksi vielä painetaan satunnaiseen artikkeliin, pidetään kolmen sekunnin tauko ja suljetaan selain.

4.5 Yhteenveto esimerkeistä

Kuten esimerkeistä näkyi, testeistä pystyy tehdä hyvin yksinkertaisia. Kommentoimalla vaiheita testaaja pysyy perässä mitä testissä tapahtuu, mikäli

testistä ei suoraan saa sitä selville. Kommentoimalla kokonaan vaiheita pystyy esimerkiksi kokeilemaan erilaisia tapoja löytää elementti ilman, että joutuu poistamaan rivejä ja myöhemmin lisäämään uudestaan.

Kokeilemalla erilaisia locatoreita löytää sen sopivimman testattavalle sivustolle tai sovellukselle. Id on yleensä se luotettavin ja sitä suositellaan käyttämään, jos se vain on mahdollista, ja sivulta tai näkymästä ei löydy toista samaa id:tä. Välillä täytyy käyttää sellaista locatoria, joka ei ole suositeltava, mutta joidenkin elementtien kohdalla pakollinen.

Suite setup ja suite teardown kannattaa tehdä, jos se on mahdollista, ja testit alkavat samasta aloituspisteestä. Testit ovat sitten siistimpiä ja jokaista testiä ei tarvitse erikseen alkaa muokata. Se säästää huomattavasti ylläpidolta aikaa.

5 YHTEENVETO

Automaatiotestaus on iso apu manuaalisen testauksen rinnalla. Automaatiotestauksella voidaan testata esimerkiksi sivustojen isoja kokonaisuuksia, jotka suurimmaksi osaksi ovat muuttumattomia. Automaatiotestaus on tehokas lisä testaukseen. Se vähentää aikaa manuaalitestauksesta, ja saattaa myös olla tarkempi kuin ihmissilmillä testattuna. Manuaalista testausta ei kuitenkaan jätetä pois.

Tässä työssä käytetty työkalu Robot Framework on helppokäyttöinen automaatiotyökalu, jonka käyttö perustuu suurimmaksi osaksi avainsanoihin. Työkalulle on erilaisia kirjastoja, jotka sisältävät testeissä käytettäviä avainsanoja. Ohjelmiston käyttöönotto on nopeaa, koska se on ilmainen testaustyökalu. Se ei vaadi mitään tiettyjä ominaisuuksia käyttäjän tietokoneelta.

Testauksessa ei tarvitse koodata Pythonia tai Javaa, vaikka työkalu niihin perustuukin. Avainsanojen avulla testit saa toimimaan sillä tavalla, miten käyttäjä käyttäisi testattavaa tuotetta. Testaus tapahtuu siis käyttöliittymätasolla. Locatoreiden löytämiseen ei tarvitse erikseen työkaluja, vaan selaimesta itsessään löytyy työkalu siihen ”Inspect element”. Erilaisilla avainsanoilla saa tehtyä kokonaisuuksia, jotka helpottavat ylläpitoa ja selkeyttävät testejä jakamalla esimerkiksi testit avainsanoilla osiin toimintojen perusteella.

LÄHTEET

1. Introduction. Robot Framework. Saatavissa: http://robotframework.org/. Hakupäivä 8.10.2017.

2. Lehto, Topias 2017. Testiautomaatiotyökalun valinta web-sovellukselle. University of Oulu. Jultika University of Oulu repository. Saatavissa: http://jultika.oulu.fi/Record/nbnfioulu-201706022480. Hakupäivä 29.11.2017.

3. Most popular test automation frameworks with pros and cons of each – Selenium Tutorial #20. 2017. Software Testing Help. Saatavissa: http://www.softwaretestinghelp.com/test-automation-frameworks-selenium-tutorial-20/. Hakupäivä 10.10.2017.

4. Sellink, Alex – Verhoef, Chris 1999. Scaffolding for software renovation. Saatavissa: http://www.cs.vu.nl/~x/scaf/scaf.html. Hakupäivä 29.11.2017.

5. Ali, Amir 2015. Introduction to software test automation. SlideShare. Saatavissa: https://www.slideshare.net/amraldo/introduction-to-software-test-automation. Hakupäivä 29.11.2017.

6. Automated testing. 2017. Technopedia. Saatavissa: https://www.techopedia.com/definition/17785/automated-testing. Hakupäivä 10.10.2017.

7. Build. TechTarget. Saatavissa: http://searchsoftwarequality.techtarget.com/definition/build. Hakupäivä 29.11.2017.

8. Fernandes, Joe – Di Fonzo, Alex. When to automate your testing (and when not to). Oracle. Saatavissa: http://www.oracle.com/technetwork/topics/qa-testing/whatsnew/when-to-automate-testing-1-130330.pdf. Hakupäivä 10.10.2017.

9. Klärck, Pekka 2016. Home. Github. Saatavissa: https://github.com/robotframework/RIDE/wiki. Hakupäivä 29.11.2017.

10. Kehitysympäristö RIDE. Robot Framework. Saatavissa: Robot Frameworkin asennuksen yhteydessä. Hakupäivä 29.11.2017.

11. Selenium2Library. 2017. Libdoc. Saatavissa: http://robotframework.org/Selenium2Library/Selenium2Library.html. Hakupäivä 16.10.2017.

12. 10 Tips on how to dramatically reduce test maintenance. Telerik. Saatavissa: https://www.telerik.com/docs/default-source/Test-Studio/10-tips_maintainable_tests.pdf?sfvrsn=2. Hakupäivä 16.11.2017.

13. Google. Saatavissa: https://www.google.com. Hakupäivä 23.11.2017.

14. Evoluutio. 2017. Wikipedia. Saatavissa: https://fi.wikipedia.org/wiki/Evoluutio. Hakupäivä 23.11.2017.

LIITTEET

Liite 1.

ROBOT FRAMEWORKIN ASENNUS:
Asenna kaikki 32-bittisenä!

1. Asenna python v.2.7: https://www.python.org/downloads/windows/
2. Lisää python 2.7 kansio environment variables (system → advanced system settings → environment variables…). Lisää PATH (jos ei olemassa) ja sen täytyy sisältää C:\Python27 ja C:\Python27\Scripts. Yhteyden voit tarkistaa kirjoittamalla cmd:lle python -V ja sen pitäisi tulostaa käyttämäsi pythonin versio
3. Päivitä pip, jos tarvitsee (scripts-kansiossa): python -m pip install --upgrade pip
4. Asenna Robot Framework käyttämällä pip:ä: pip install robotframework
   Jos cmd ei löydä pip:ä, mene kansion sisälle. Luultavasti kansio tällä hetkellä on C:\Users\käyttäjä → kaksi kertaa cd.. (näkyy vain C tämän jälkeen) → cd Python27\Scripts → yritä uudestaan asentaa
5. Asenna RIDE (Robot IDE)
   Ensin asenna wxPython: https://sourceforge.net/projects/wxpython/files/wxPython/2.8.12.1/
   Asenna RIDE pip:ä käyttäen: pip install robotframework-ride
   RIDE aukeaa kirjoittamalla ride.py cmd:lle
6. Asenna Selenium2Library pip:ä käyttäen: pip install selenium2library
7. Browsereille tarvitaan omat ajurit: GC → chromedriver, FF → geckodriver, IE → iedriver
   Liitä tiedostot Python27\Scripts. FF tarvitsee elementtien löytämiseen lisäosia: firebug, firefinder ja firepath.

ROBOT FRAMEWORK INSTALLATION:
Install everything 32-bit!

1. Install python v.2.7: https://www.python.org/downloads/windows/
2. Add python 2.7 folder to environment variables (system → advanced system settings → environment variables…). Add PATH (if it doesn't exist) and it must contain C:\Python27 and C:\Python27\Scripts. You can check the connection by writing python -V to cmd and it should print the version of the python you are using.
3. Upgrade pip if needed (it's in scripts-folder): python -m pip install --upgrade pip
4. Install Robot Framework by using pip: pip install robotframework
   If cmd doesn't find pip, go to the folder. At this point the folder probably is C:\Users\useraccount → write two times cd.. (should show only C)→ go to folder cd Python27\Scripts → try again installing
5. Install RIDE (Robot IDE)
   First you need to install wxpython: https://sourceforge.net/projects/wxpython/files/wxPython/2.8.12.1/
   Install RIDE with pip: pip install robotframework-ride
   Ride opens by writing to cmd ride.py
6. Install Selenium2Library by using pip: pip install selenium2library

7. For all the browsers, you need their own drivers: GC → chromedriver, FF → geckodriver, IE → iedriver. Add downloaded files to folder Python27\Scripts. FF needs extensions for finding elements from websites: firebug, firefinder and firepath.