SAVONIA

OPINNÄYTETYÖ - **AMMATTIKORKEAKOULUTUTKINTO**
TEKNIIKAN JA LIIKENTEEN ALA

ROBOTIIKKA JA TYÖN AUTOMATISOINTI

TEKIJÄ: Toni Salminen

SAVONIA-AMMATTIKORKEAKOULU

OPINNÄYTETYÖ
Tiivistelmä

Koulutusala Tekniikan ja liikenteen ala	
Koulutusohjelma/Tutkinto-ohjelma Tietotekniikan tutkinto-ohjelma	
Työn tekijä(t) Toni Salminen	
Työn nimi Robotiikka ja työn automatisointi	
Päiväys 12. Joulukuuta 2018	Sivumäärä/Liitteet 28
Ohjaaja(t) Jussi Koistinen ja Pekka Granroth	
Toimeksiantaja/Yhteistyökumppani(t) DNA Oyj	

Tiivistelmä

Kysyntä manuaalisen tietoteknisen työn muuttamisesta automaattisesti suoritetuksi työksi on kovempi kuin koskaan. Yritykset haluavat automatisoida työtä säästääkseen rahaa ja resursseja, jolloin varsinaisille työntekijöille jää enemmän aikaa järjestelmiensä kehittämiseen ja hallinnointiin. Tämän opinnäytetyön aiheena on ohjelmistorobotiikan hyödyntäminen erilaisissa työtehtävissä.

Tämän opinnäytetyön päätavoitteina on luoda vaatimusmäärittelyt ensimmäiselle testirobotille, tehdä esimerkkirobotti sekä tutustua ohjelmistorobotiikkaan yleisesti. Esiselvityksinä vaatimusmäärittelyihin tarkasteltiin ja analysoitiin työtehtäviä, joista valittiin paras ehdokas ensimäiseksi työksi. Tämän jälkeen selvitettiin UIPath -ympäristön tarjoamat mahdollisuudet sekä oikea formaatti vaatimusmäärittelyille. Varsinaisten vaatimusmäärittelyiden suunnittelu aloitettiin työtehtävien aineiston keruulla, joista valittiin lopulta paras ehdokas jatkoon erilaisten ominaisuuksien kuten työtehtävässä kuluvan ajan käytön suhteen. Lisäksi vaatimusmäärittelyitä varten selvitettiin virheenhallinta ja automatisaatiosta saatavat hyödyt. Tämän jälkeen määriteltiin robotin toimintaa työtehtävää varten.

Lopputuloksena syntyivät robotin vaatimusmäärittelyt sekä esimerkkirobotti UIPath -ympäristöön. Esimerkkirobottin luonnin aikana ohjelmistorobotiikasta opitut asiat korostuivat ja osoittivat, että automatisaatio on järkevää. Tätä esimerkkirobottia tullaan hyödyntämään testirobotin suunnittelussa. Opinnäytetyötä tehdessä opittiin paljon ohjelmistorobotiikasta, vaatimusmäärittelyiden tekemisestä sekä UIPath -ympäristön toiminnasta.

Avainsanat

SAVONIA UNIVERSITY OF APPLIED SCIENCES

THESIS
Abstract

Field of Study Technology, Communication and Transport			
Degree Programme Degree Programme in Information Technology			
Author(s) Toni Salminen			
Title of Thesis Robotics and the Automation of Jobs			
Date	12 December 2018	Pages/Appendices	28
Supervisor(s) Mr Jussi Koistinen, Senior Lecturer and Mr Pekka Granroth, Senior Lecturer			
Client Organisation /Partners DNA Oyj			

Abstract

The demand for manual IT work to be automatically executed is harder than ever. Businesses want to automate work to save money and resources, giving the actual employees more time to develop and manage their systems. That move towards automation was what happened in the DNA Oyj company.

The main objectives of this thesis were to create requirements for the first test robot, make an example RPA robot by using the UIPath environment and to learn more about the Robotic Process Automation (RPA) in general. The methods used in this thesis involved analyzing and using the UIPath environment. The prerequisites for the test robot requirements were to analyze the current job assignments, choose one of them as the best candidate to be the first automated job, study the possibilities offered by the UIPath environment and the correct format for specification definitions.

The planning of the actual requirements was started by collecting job assignments data from the earlier made analyzes, from which one was ultimately chosen as the best candidate for various reasons, such as work-time consumption. In addition, error management and automation benefits were investigated for the automation process. After that, the robot's work was defined for the job. The creation of the example robot in UIPath environment was documented in this thesis with a step-by-step guide that included pictures for easier understanding of the current step. After the robot was created, it was then executed to see the results of this creation.

The results were the robot specification requirements and the example robot for the UIPath environment. An example robot was created in the UIPath environment and during the building it was shown that the RPA is the way to go and it also demonstrated that automation is sensible. This example robot will be utilized in designing the test robot later on.

Keywords

SISÄLTÖ

1 JOHDANTO

1.1 Opinnäytetyön tausta, tavoitteet ja rajaukset

Maailmalla on jo pitkään puhuttu robotiikasta, eli työn automatisoinnista. Automatisointia varten luotu robotti hoitaisi työtä, jonka tekemiseen on ennen vaadittu ihminen. Tänä päivänä tätä ajatusta on hyödynnetty varsinkin tehdastyössä, jossa aikaa ja tarkkuutta vaativat työt hoitavatkin ihmisen ohjaama työrobotti tai tietokonesovellus. Viime aikoina tämä kehitys on vahvemmin siirtynyt tietotekniikan puolelle ja käsin tehtyä työtä pyritään automatisoimaan ja näin vähentämään aina vain enemmän (Talouselämä).

Ohjelmistorobotti ja työn automatisointi ovat tärkeä osa tietoteknistä kehitystä. Toistuvat ja työläät työtehtävät suorittaa jatkossa automaattisesti toimiva robotti. Työntekijöillä on enemmän aikaa suunnitella ja kehittää työympäristöjään, puhumattakaan työnlaadun parantumisesta. Kukapa haluaisi esimerkiksi painaa Enter -näppäintä toistuvasti kahdeksan tuntia päivässä, kun automatisointi hoitaisi sen helposti.

DNA Oyj:llä on ollut keskustelua jo jonkin aikaa tietynlaisen työn automatisaatiosta. Yritykselle tulee päivittäin aikaa ja vaivaa vaativaa, toistuvaa virheenkorjaustyötä, jonka suorittaminen ihmistyövoimalla kuluttaa resursseja ja ve aikaa pois itse järjestelmien tarkkailusta ja kehityksestä. Nämä aiemmin mainitut virheenkorjaustyötehtävät ovat virhetilanteita asiakastilauksissa, jotka tulevat lopulta työntekijöiden korjattaviksi. Vaikka järjestelmät ovat automaattisia, kaikki ei aina mene suunnitellun mukaisesti, ja näin syntyy virhetilanne. Automatisoinnin eli tässä yhteydessä työrobotin suunnittelu ja käyttöönotto vähentäisi merkittävästi ajankäyttöä, rahaa ja ihmistyöstä johtuvaa virhemarginaalia.

Tämän opinnäytetyön tavoitteena on osoittaa automatisoinnin hyödyt ja selvittää vaatimusmäärittelyt ensimmäiselle testirobotille, josta itse varsinainen työrobotti tullaan rakentamaan. Tämä tarkoittaa käytännössä seuraavien asioiden läpikäyntiä:

- Tutustutaan erilaisiin työtä vaativiin toimenpiteisiin huolella ja kirjoitetaan työvaiheet talteen eli dokumentoidaan ne. Etsitään paljon aikaa ja vaivaa aiheuttavat työtoimenpiteet, joiden automatisoinnista olisi suurin hyöty.
- Vaatimusmäärittelyiden teko, joiden avulla testirobotin lopullinen suunnittelu ja käyttöönotto tulee olemaan merkittävästi helpompaa.
- Hyödyn osoittaminen; Kuinka paljon ajan, rahan ja työvoiman käyttöä ohjelmistorobotilla olisi mahdollista säästää.

Opinnäytetyö on rajattu käsittelemään robotiikkaa yleisesti ja vaatimusmäärittelyjen luontia testirobotille. Testirobotin rakentamisessa hyödynnetään tässä opinnäytetyössä syntyneitä vaatimusmäärittelyitä.

Yhteistyökumppanina tässä opinnäytetyössä toimii DNA Oyj. Tausta-aineistot saadaan UIPath -alustalta ja tutkimusalustana tässä opinnäytetyössä käytetään yrityksen omaa sisäistä, salassapidettävää dataa.

1.2 DNA Oyj

DNA Oyj on yksi Suomen suurimmista tietoliikennekonserneista ja teleoperaatoreista, jolla on liiketoimintaa niin yritys – kuin kuluttajapuolellakin matkapuhelinliittymissä, internetyhteyksissä ja myös televisioyhteyksissä. Yritys on perustettu vuonna 2000 ja sillä on yli 2,8 miljoonaa puhelinliittymäasiakasta ja lähes 1,5 miljoonaa internetasiakasta. Yrityksen liikevaihto vuonna 2017 oli 886 miljoonaa euroa. (Wikipedia: DNA)

Tämän lisäksi yritys myy myös laitteita ja palveluita kuluttajille ja yrityksille, kuten esimerkiksi puhelimia ja DNA TV- palvelua yli 60:ssä DNA Kauppa liikkeessä ympäri Suomea sekä internetissä. (Wikipedia: DNA)

1.3 Lyhenteet ja määritelmät

RPA = Robotic process automation
Robotiikka = Työn automatisaatioon keskittyvä ala
Automatisointi = Manuaalisen työn muuttaminen automaattisesti suoritettavaksi, esimerkiksi koneen avulla.
Alusta = Työympäristöstä koostuva kokonaisuus

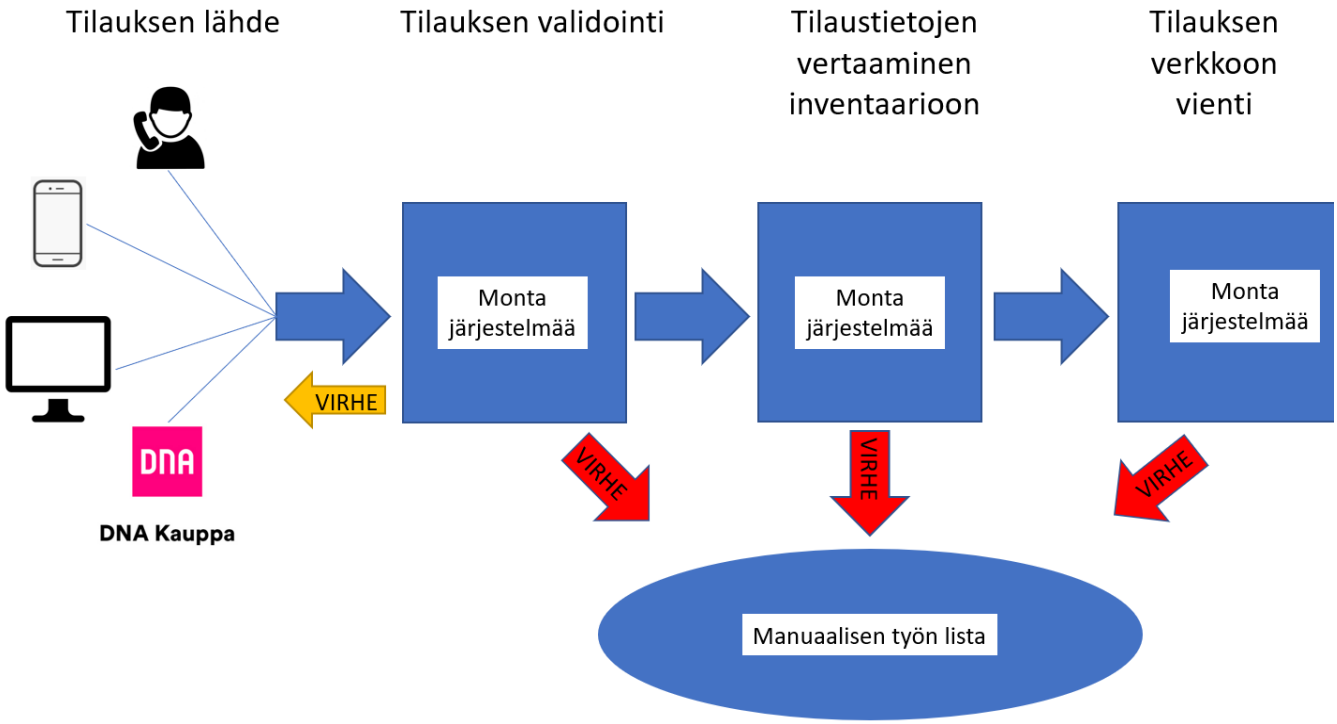
2 TAUSTATIETOA

2.1 Virheen synty ja tilauksen yleinen rakenne

Virheen syntymiseen on monta syytä ja paikkaa, joten asian parempaan hahmottamiseen on parempi aloittaa tilauksen alusta esimerkin avulla. Esimerkiksi uusi puhelinliittymätilaus voi syntyä monesta eri lähteestä, joista yleisimmät ovat:

- Asiakas tilaa internetselaimella puhelimen tai tietokoneen kautta uuden palvelun, esimerkiksi uuden puhelinliittymän.
- Asiakas tilaa uuden puhelinliittymän DNA Kaupassa
- Asiakas soittaa DNA:n asiakaspalveluun ja tilaa uuden puhelinliittymän
- Yritysasiakas soittaa omaan yrityksensä asiakaspalveluun, käyttää omaa portaalia tai tietokone luo automaattisesti uusia puhelinliittymätilauksia.

Kuten huomataan, tilauksen lähteitä on monia. Yksinkertaisimmillaan tilaus etenee kolmessa vaiheessa: tilauksen validointi, vertaaminen inventaarioon ja verkkoon vienti. Tällöin varmistetaan tiedon oikeellisuus ja vähennetään virhetilanteita asiakkaalle (kuva1.).



Kuva 1. Tilauksen kulku

Ensimmäiseksi validoidaan tilauksen sisältämät tiedot. Jos tilaus ei ole standardien mukainen, vaan sisältää esimerkiksi vääriä parametrejä, jotka ovat syntyneet esimerkiksi tietojärjestelmän virheen takia, se hylätään suoraan. Jos tilaus on oikein muodostettu, siirtyy se eteenpäin (Kuva 1.).

Seuraavassa vaiheessa tutkitaan ja verrataan tietokannassa olevia tietoja. Jos esimerkiksi yritetään luoda uusi puhelinliittymä numerolle, joka on jo käytössä tai ei kuulu DNA:lle, tilaus on virheellinen ja vaatii toimenpiteitä. Tässä vaiheessa tämä tilaus päätyy manuaaliseen käsittelyyn. Jos tilauksen suorittamiseen ei ole tietokannassa esteitä, siirrytään eteenpäin (Kuva 1.).

Viimeisessä vaiheessa tilauksessa tulleet tiedot siirtyvät verkkoon eli käytettäväksi. Jos tässä vaiheessa tulee virhe, nämäkin virheet päätyvät manuaaliseen käsittelyyn. Virheen korjauksen jälkeen tilaus voi jatkaa kulkuaan eteenpäin. Kun tilaus on suoritettu onnistuneesti, asiakas saa vahvistuksen siitä ketjun alkuun ja näin tilaus on saatu loppuun. (Kuva 1.)

2.2 Ohjelmistorobotiikka

Ohjelmistorobotiikka tarkoittaa toistuvien työtehtävien ja prosessien automatisointia ohjelmisto -pohjaisen ”työrobotin” avulla. Kyseinen robotti osaa käyttää erilaisia tietojärjestelmiä ja sovelluksia kuin normaali ihminen, ja se osaa toimia itsenäisesti. Robotti osaa esimerkiksi lukea tietoa näyttökuvasta, syöttää tietoa erilaisiin informaatiokenttiin ja liikkua eri sovellusten välillä (Wikipedia: Ohjelmistorobotiikka). Nykyiset järjestelmät riittävät: robotille ei ole väliä, mitä järjestelmiä se käyttää (Affecto).

Ohjelmistorobotiikan alkuna voidaan pitää yleisiä teollisuusrobotteja, joissa alkuperäisenä tarkoituksena oli rakentaa fyysisiä, toistoa vaativan työvaiheen suorittavia robotteja. Ensimmäisiä teollisuusrobotteja suunniteltiin jo 1960-luvulla, ja ensimmäinen Unimate -teollisuusrobotti asennettiinkin Fordin tehtaalle jo vuonna 1961. (Wikipedia: Ohjelmistorobotiikka)

Ohjelmistorobotti ei ole fyysinen robotti, vaan kyse on ohjelmasta, joka toimii ihmisen kaltaisesti. Sovelluksessa suoritetaan hiirenklikkauksia ja kirjoitetaan, aivan kuten normaali työntekijäkin kirjoittaisi. Ohjelmistorobottia ei siis näin ollen varsinaisesti ohjelmoida perinteisin menetelmin, vaan sille opastetaan eri työvaiheet ohjelmansa avulla ja neuvotaan, mitä milloinkin tulee tehdä. Periaatteessa voitaisiin jopa puhua virtuaalisesta työntekijästä! Esimerkiksi UIPath Orchestrator -sovelluksessa käytetään vuokaaviota, jossa käyttäjä voi halutessaan lisätä robotille työvaiheita haluamaansa kohtaan tai muokata annetun ”johdattelun” avulla syntyneitä työvaiheita. (Wikipedia: Robotic Process Automation).

Ohjelmistorobotille voidaan opettaa työn suoritukseen logiikkaa, eli tapaa millä se toimii missäkin tilanteessa. Ohjelmistorobotille voi myös luoda virheenkäsittelymalleja ja voidaan jopa puhua tekoälystä. Kaikki opetettavat ja ohjelmoitavat asiat tähtäävät lopulta täydelliseen työtehtävän automatisaatioon. Dokumentointi nähdään usein työläänä työvaiheena, mutta suoritettujen työtehtävien dokumentointi ja ajantasaisten tietojen ylläpito ei ole ongelma robotille. (Wikipedia: Robotic Process Automation).

Ohjelmistorobotiikkaa voi harjoittaa kahdella eri tavalla, joista toinen on ohjelmointipainotteinen ja toinen graafisten käyttöliittymien hyödyntäminen. Näistä ensimmäistä on käytetty laajalti jo eri yrityksissä esimerkiksi skriptien muodossa. Nämä skriptit suorittavat asioita taustalla, mutta ne eivät osaa hyödyntää tai käyttää graafisia käyttöliittymiä. Nämä skriptit on usein myös sijoitettava järjestelmien ytimiin ja siinä on oma tietoturvariskinsä, kun skriptien ajajat tarvisevat oikeuksia niiden suorituksiin. Graafisiin käyttöliittymiin perustuvat ohjelmistorobotit sisältävät yleensä oman alustan ja robotin saa käyttäytymään hyvin ihmismäisesti. Näin pääsyä tietoturvariskeihin paikkoihin ei ole tarvetta ja yksi robotti pystyy tekemään ja käyttämään monen eri järjestelmän välineitä. (Wikipedia: Robotic Process Automation).

Suurin hyöty ohjelmistorobotiikalla saavutetaan, kun käsittelyyn otetaan eniten aikaa ja määrällisesti paljon toistuvia työtehtäviä. Työtä ei tarvitse optimoida robottia varten, vaan työ voi pysyä juuri sellaisena kuin se on (Wikipedia: Robotic Process Automation). Ja koska robotti osaa toimia itsenäisesti, se voi työskennellä jopa 24 tuntia vuorokaudessa; robotti ei vaadi lepoa, ja se on pois käytöstä ainoastaan huoltokatkojen aikaan. Näin saavutetaan huomattavia säästöjä henkilöstön tarpeessa. Ohjelmistorobotiikan hyödyistä puhutaan lisää omassa kappaleessaan.

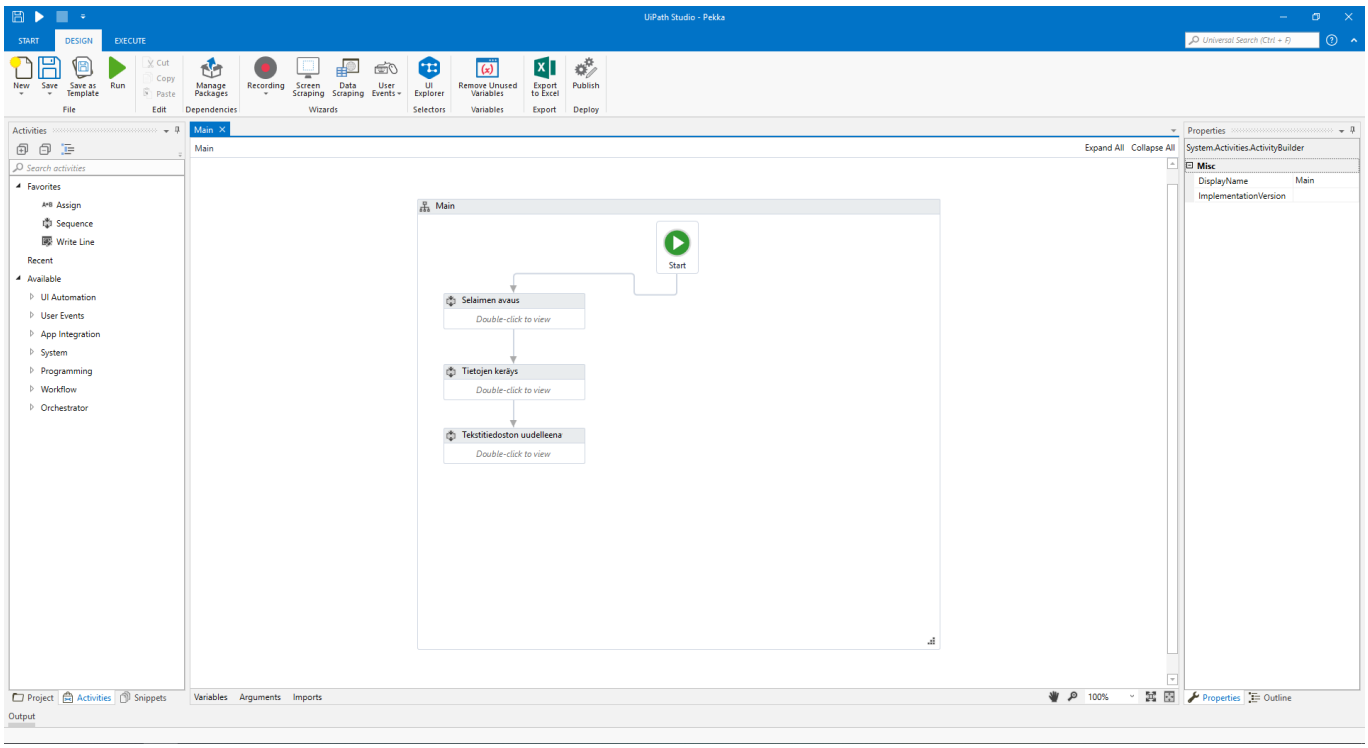
2.3 UIPath Orchestrator

UIPath Orchestrator on ohjelmistorobotiikan hyödyntämiseen perustuva sovelluskokonaisuus, jonka avulla käyttäjä voi luoda oman ohjelmistorobotin. Sen on kehittänyt UIPath -niminen monikansallinen yhtiö, joka on perustettu vuonna 2005 (Wikipedia: UIPath). Yrityksen tulot viimevuonna olivat 140 miljoonaa dollaria (UIPath). Yrityksen pääliiketoimintaa ovat tekoäly ja sen ympärille rakennetut ohjelmistorobotiikan sovellukset. 18. päivä syyskuuta 2018 UIPath sai 225 miljoonan dollaria viimeisimmällä rahoituskierroksellaan. Maaliskuussa 2018 Kleiner Perkins Caufield & Byers oli arvioinut yrityksen arvoksi noin 1,1 miljardia dollaria. (UIPath)

UIPath Orchestrator on alusta, josta löytyvät niin UIPath Studio kuin UIPath Robot -osiot. UIPath Studio on graafinen käyttöliittymä, joka mahdollistaa ohjelmistorobotin luonnin kaavioiden avulla (Kuva 2.). Lisäksi robottia voi ”opastaa” näyttämällä itse, mitä ihminen tekee missäkin tilanteessa.

Kehittyneen konenäön ansiosta robotti osaa tunnistaa erilaisia tekstikenttiä, sovelluksia ja kuvia sekä toimia niiden mukaan. Vaikeampia työtehtäviä varten sovelluksessa voi luoda käsin erilaisia työvaiheita, kuten arvojen käsittelyä ja asioiden ymmärrystä. Myös ehtolauseiden käyttö if, if else, ja else -lauseilla onnistuu. Silmukkarakenteet ovat myös tuettuja, näin robotti pystyy toimimaan itsenäisemmin.

UIPath Robot on robottien suorittamiseen, tilanteen seuraamiseen ja valvontaan perustuva sovellus. UIPath Robot tarjoaa graafisen käyttöliittymän, jossa kaikkien robottien hallinta on tehty mahdollisimman helpoksi.



Kuva 2. Esimerkkikuva UIPath Studio -sovelluksesta

3 AINEISTON KERUU

Ennen kuin työrobotin vaatimusmäärittelyjä voidaan lähteä rakentamaan, on selvitettävä yrityksen nykyinen tilanne. Tilanteen saa parhaiten selville keräämällä aineistoa ja dataa monesta lähteestä. Työntekijät, tietokannat ja omat kokemukset auttavat kaikki osaltaan varsinaisten vaatimusmäärittelyiden tekoa. Onko työ hidasta suorittaa? Minkälainen virheenkorjaustyö esiintyy useasti työpäivän aikana? Huolellinen taustatyö on tärkeää.

3.1 Työntekijät ja omat kokemukset

Virheenkorjaustyötä suorittaa yrityksessä tilaushallinnan osastolla tällä hetkellä varsinaisesti kolme ihmistä, joista yksi on tämän opinnäytetyön kirjoittaja. Tilaushallinnan työntekijät tietävät, kuinka paljon aikaa kuluu keskimäärin tietyntyyppisen tilaushallinnassa olevan virhetilanteen korjaukseen ja miten virhetilanne tulisi hoitaa. Työntekijöiltä kysyttäessä esiin nousi työtehtäviä, jotka ovat paljon aikaa vieviä, arvaamattomia ja tarkkuuta vaativia. Tiettyjen järjestelmien käyttö komentorivitasolta tekee tilaushallinnan työstä usein hitaampaa ja tarkkuuta vaaditaan entistä enemmän. Huolimattomuudella voi saada paljon tuhoa aikaan.

Työläimpien virhetilanteiden korjaus noudattaa usein samaa perusajatusta. Näissä virhetilanteessa tehdään yleensä seuraavat asiat, jotka on yksinkertaistettu työvaiheiden selkeyttämiseksi:

- Selvitetään oikeat halutut tiedot saadusta tilauksesta.
- Verrataan haluttuja arvoja järjestelmistä tai komentorivitasolta löytyneisiin arvoihin.
- Muokataan tai luodaan muutostiedostoja oikeilla arvoilla, jotka lähetetään eteenpäin serverille. Muutostiedostojen avulla muokataan tai poistetaan tietoa järjestelmissä.
- Jatketaan virheeseen pysähtyneen tilauksen ajoa uudelleen.

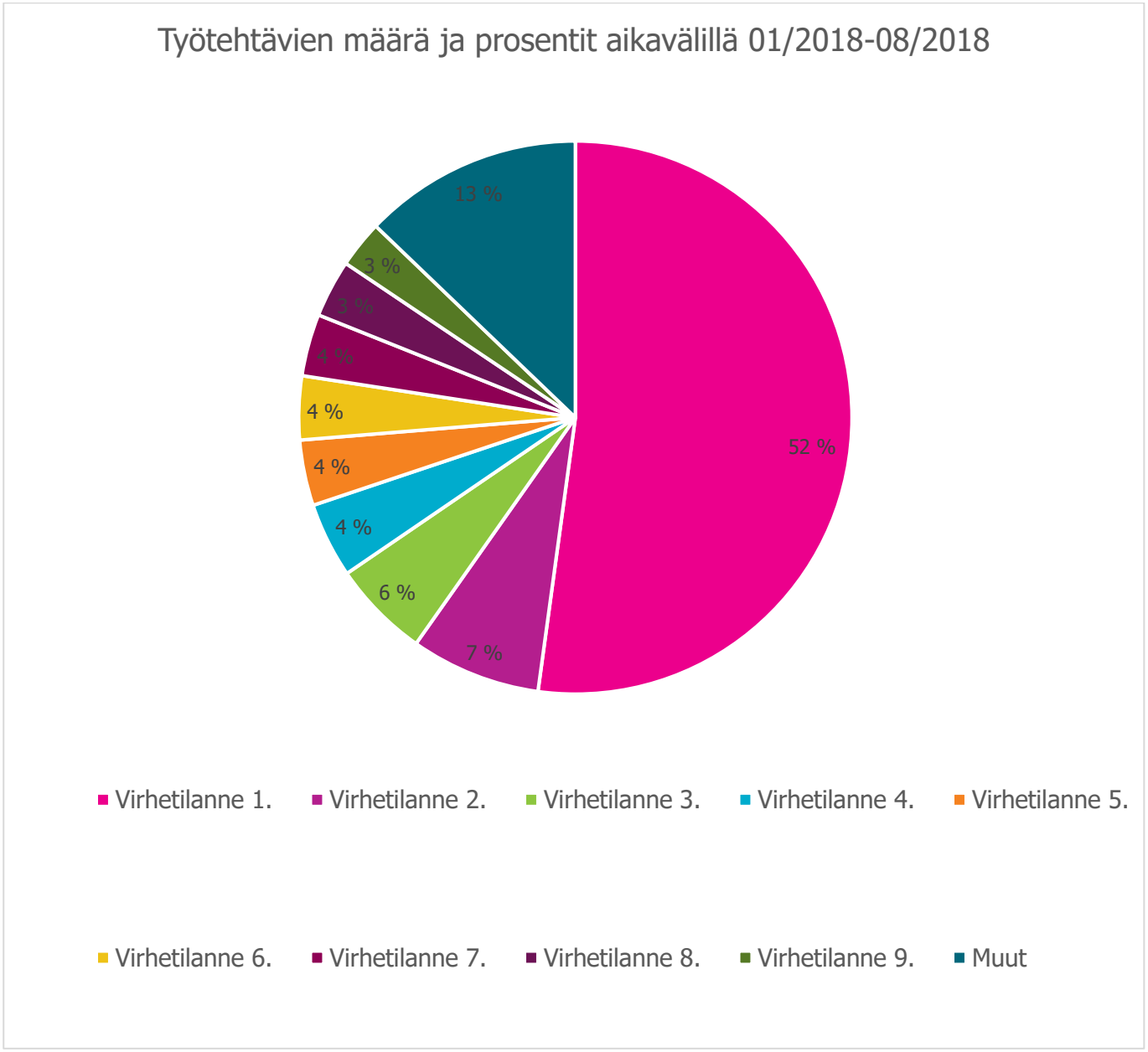
Joissain tapauksissa virhetilanteen ratkaiseminen voi muuttua hankalammaksi, ja se lisää ajankäyttöä. Vaikeampien työtehtävien lisäksi esiin nousi helppoja, mutta usein toistuvia virhetapauksia. Näistä virhetilanteista selviää usein komennon uudelleenlähetyksellä, mutta asiasta pitää olla täysi varmuus. Siksi näissäkin tapauksissa on verrattava järjestelmistä saatuja tietoja tilauksen tietoihin. Näissä virhetilanteissa aikaa kuluu runsaimmin järjestelmästä toiseen siirtymisessä, koska kaikkiin järjestelmiin ei aina pääse nopeasti sisään tarkastelemaan tietoja.

Omat kokemukset nojaavat pitkälti muiden työntekijöiden esiintuomiin hyviin faktoihin, mutta lisäksi aikaa kuluttavia virhetilanteita ovat tapaukset, joiden tietoja työntekijällä ei ole mahdollisuutta nähdä, vaan näissä tapauksissa työntekijä joutuu kysymään tietoja sähköpostin välityksellä tai odottamaan lisätietoa virhetiketin muodossa. Näitä tapauksia onneksi on vähemmän.

3.2 Tietokannat

Tietokannoista on mahdollista nähdä erilaisten työtehtävien lukumääriä. Tästä on hyötyä varsinkin, kun suunnitellaan ensimmäistä työtehtävää, jonka voisi testirobotin rakentaa suorittamaan. Tietokannoista ei tosin näe kulunutta aikaa työn suorittamiseen eikä työn vaikeustasoa, mitkä ovat osaltaan myös tärkeitä asioita automatisoinnissa. Ei ole järkevää suunnitella ja rakentaa robottia työtehtävälle, joka tapahtuu vain kerran.

Työtehtävien tarkat lukumäärät saatiin selville yrityksen tilaushallintajärjestelmän omista tietokannoista. Saadut tiedot sain Excel -tiedostona, jonka käsittelyyn meni hieman aikaa. Virhetilanteita oli excel -tiedostossa 9900 kappaletta. Alla olevassa kuviossa näkyvät jakaumat eri virheenkorjaustyötehtävien välillä. Virhetilanteiden oikeat nimet on piilotettu, koska kyseessä on yrityksen salassapidettävää sisäistä tietoa.



KUVIO 1. Työtehtävien määrä ja prosenttijakauma aikavälillä 01/2018-08/2018

Virhetilanteista aiheutuvia työtehtäviä on monia, yhteensä jopa 55 kappaletta, mutta kuvion selkeyttämiseksi siihen on valittu yleisimmät virheenkorjaustyötehtävät ja loput ovat laskettu yhteen ”Muut” -kohtaan. Oleellisinta olisi löytää eniten automatisoinnin kannalta hyödyttävät virhetilanteet, jolloin saavutetaan parempaa tuottavuutta ja ajan säästöä. Nämä taas lopulta kääntyvät rahansäästöksi.

Tässä tapauksessa suurin prosenttimäärä ei automaattisesti tarkoita sitä, että se valittaisiin automatisoitavaksi. Myös muut asiat vaikuttavat. Näitä ovat muun muassa kaavamaisuus virheenkorjauksessa, automatisoinnin helppous sekä uudelleenkäytettävyys muihin tulevaisuudessa automatisoitaviin virheenkorjaustöihin.

Paras lähestymistapa työtehtävien automatisointiin onkin aloittaa pienistä asioista ja järkevästi edetä kohti vaikeampia malleja. Ensimmäistä testirobottia silmällä pitäen, turha kompleksisuus hidastaisi merkittävästi testirobotin suunnittelua ja rakennusta.

4 VAATIMUSMÄÄRITTELYIDEN SUUNNITTELU JA TOTEUTUS

Vaatimusmäärittelyt ovat tärkeä osa uuden tietoteknisen asian suunnittelussa. Oikein tehtynä ne tarjoavat hyvän pohjan konkreettiselle luonnille. Vaatimusmäärittelyt voi nähdä eräänlaisena suunnitelmana ja johdatteluna. Uuden asian ”speksaaminen” helpottaa luotavan asian ja sen luonnin hahmottamista. Tässä opinnäytetyössä puhutusta ohjelmistorobotista on tarpeen tehdä huolelliset vaatimusmääritelyt; näin robotin hyödyllisyysaste nousee merkittävästi.

4.1 Vaatimusmäärittelyiden suunnittelu

4.1.1 Perustiedot

Ohjelmistorobotin vaatimusmäärittelyiden suunnitteluvaiheessa hyödynnetään kerättyjä tietoja monesta eri lähteestä. Pääasiat suunnittelussa olivat:

- Mikä on nykyinen tilanne automaation osalta?
- Miten työtehtävät tällä hetkellä hoidetaan?
- Työtehtävien yleisyys
- Työtehtävien suorittamiseen kuluva aika
- UIPath- alustan tarjoamat mahdollisuudet

4.1.2 Esiselvittelyt

Vaatimusmäärittelyiden esiselvitysvaiheessa on tärkeää varmistaa mikä on vaatimusmäärittelyiden oikea rakenne ja varmistaa päämäärä. Vaatimusmäärittelyistä syntyvät dokumentit erovat merkittävästi toisistaan eri yritysten välillä. On tärkeää selvittää vaatimusmäärittelyille oikea rakenne ja pohja, jota yrityksen sisällä käytetään. Tällöin dokumentin vastaanottajan on helppo lukea ja ymmärtää, mitä robotin halutaan suorittavan, ja millä tavalla tähän päästään. Päämäärän tärkeyttä ei saa unohtaa; Päämäärä ei saa muuttua matkalla, vaan siitä on pidettävä kiinni. Selkeä päämäärä myös ohjaa dokumentin tekoa ja kirjoitusta.

DNA:lla on käytössään omanlaisensa pohja vaatimusmäärittelyille, jota työssäni käytin. Rakenteessa oleellisinta oli pohjatiedon keruu, tiedon keruusta syntyneet ajatukset siitä, mikä virheenkorjaustyö toimisi hyvänä alustana myös muiden virheenkorjaustyön automatisoinnille ja luoda robotin määrittely eli ”työohje”robotille, jonka avulla robotti voidaan rakentaa.

DNA:n oman projektin päämääränä, jonka osana opinnäytetyössä syntyvät vaaimusmäärittelyt ovat, on saada aikaan ohjelmistorobotti, joka osaa tehdä sille opetetun virheenkorjaustyön. Tähän luotavat vaatimusmäärittelyt, jotka tässä opinnäytetyössä tehdään, on tähdättävä antamaan apuvälineet siihen.

4.2 Vaatimusmäärittelyiden toteutus

Vaatimusmäärittelyiden toteutus aloitettiin ensin keskustelemalla ohjelmistobotin tarpeellisuudesta tekemässäni työssä. Yhdessä esimieheni kanssa päätimme aloittaa ohjelmistorobotin suunnittelun ja koska minulla oli jo kerääntynyt tietämystä virheiden korjauksesta ja opinnäytetyö oli vielä tekemättä, minut valittiin tekemään vaatimusmäärittelyt ja valitsemaan ensimmäinen automatisoitava virheenkorjaustilanne.

Vaatimusmäärittelyn teko aloitettiin keräämällä tietoa monesta eri lähteestä, jotta tarkasteluun saadaan mahdollisimman monipuolinen otanta nykyisestä tilanteesta. Tässä vaiheessa hankin tietoa DNA:n omista tilaushallintajärjestelmistä. Sieltä sain erityisesti virheiden lukumääriä Excel -muodossa. Datan muuntaminen ymmärrettävämpään muotoon kesti jonkin aikaa, mutta lopulta yksittäisten virheiden lukumäärät olivat selvillä. Materiaalia tutkiessa huomasin virheenkorjaustyötehtävien ison määrän, ja olin entistä vakuuttuneempi automatisoinnin hyödyistä.

Seuraavaksi vuorossa oli työntekijöiden haastattelu. Kysyin työntekijöiltä, mitkä virheenkorjaustyötehtävät he kokevat paljon työskentelyaikaa vieviksi, jotta sain selville muun muassa virheenkorjaustöihin kuluvaa aikaa sekä töiden vaikeustasoa. Hyödynsin myös omaa tietämystäni, mitä olin muutaman kuukauden aikana hankkinut työskellessäni virheenkorjaustöiden parissa.

Tämän jälkeen saadusta informaatiosta saatiin koostamalla, vertailemalla ja arvioimalla irti potentiaalisimmat virheenkorjaustilanteet, joista olisi hyvä valita paras ehdokas jatkokäsittelyyn. Näistä ehdokkaista lopulta valittiin järkevin virheenkorjaustyö robotille suoritettavaksi. Ratkaisevina etuina pidettiin virheenkorjaustyön monipuolisuutta, kaavojen uudelleenkäytettävyyttä sekä työtehtävien määrää.

Seuraavaksi vuorossa oli robotille suunniteltavan virheenkorjaustyön työohje, eli yksityiskohtainen työohje siitä, miten kyseinen työtehtävä suoritetaan. Ohjelmistorobotille pitää luoda logiikka, jolla se toimii. Koska robotilla ei ole kykyä sisäistää asioita, on määrittelyn oltava tarkka. Tässä vaiheessa olin yhteydessä DNA:n omaan robotiikasta vastaavaan tiimiin, josta sain paremman kuvan siitä, miten robotin määrittely olisi loogisinta.

Ohjelmistorobotti ei lähtökohtaisesti tunnista, mihin sen täytyy syöttää esimerkiksi salasana päästäkseen eteenpäin, vaan kaikki työvaiheet on opastettava sille. Joten virheenkorjauksen työohjeen täytyy olla erittäin yksityiskohtainen. Työvaiheiden dokumentoinnissa aikaa saa kulumaan merkittävästi, varsinkin jos dokumentointia pitää ajatella robotin määrittelyä silmällä pitäen. Vaatimusmäärittelyt saatiin valmiiksi muutaman korjauskäsittelyn jälkeen

5 AUTOMATISAATION HYÖDYT

Automatisaation hyödyt on helppo määritellä. Lähtökohtaisesti syitä automatisoinnille ovat:

- Ajallinen hyöty
- Rahallinen hyöty
- Virhemarginaalin pienentyminen
- Työmotivaation nousu
- Asiakastyytyväisyyden kasvu

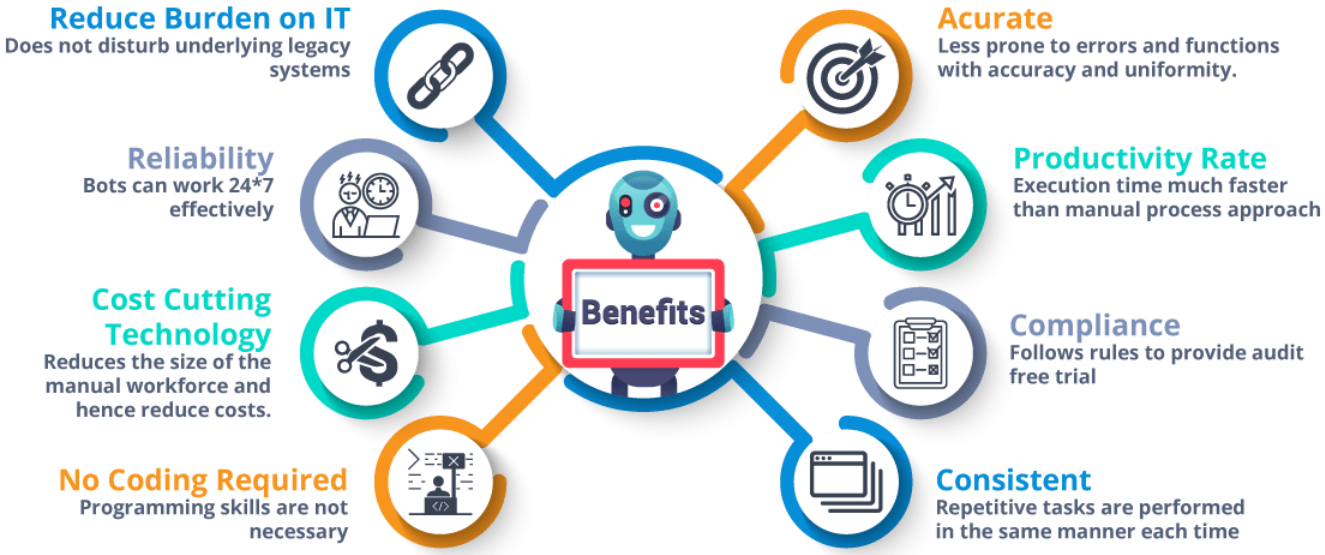
Ajallinen hyöty saavutetaan pääasiassa jättämällä pois ihmisen aiheuttama hidastelu, kuten esimerkiksi kirjoitusvaiheet. Vaikka ihminen hallitsikin kymmensormijärjestelmän, jää henkilö silti tietokonetta hitaammaksi. Vaikka järjestelmiin siirtymisestä kuluvaa aikaa ei saakkaan pois, sen suorittaa kone, ei ihminen. Ajallista hyötyä on myös se, että robotti pystyy toimimaan myös toimistoaikojen ulkopuolella, viikonloppuisin ja jopa juhlapyhinä. (Edureka)

Rahallinen hyöty nähdään yleensä suoraan ajankäytön vähentymisestä. Ne ovat hyvin paljon liitoksissa keskenään. Mitä vähemmän työtunteja tarvitaan toistuvien töiden tekoon, sitä enemmän aikaa jää vaikeampien ongelmien ratkaisemiseen, järjestelmien kehitykseen ja ylläpitoon. (Edureka)

Virhemarginaalin pienentyminen johtuu suurimmalta osin siitä, että robotti kirjoittaa täsmällisesti halutut parametrit ja tekstit kuten se on määritelty. Ihminen saattaa tehdä kirjoittaessaan ja toimiessaan virheitä, jotka sitten myöhemmin joudutaan korjaamaan uudelleen. Robotin tuottamat asiat ovat aina samanlaisia. Riskialttiissa ympäristössä, jossa virheet ovat iso ongelma, riskiä sisältävien työtehtävien automatisaatio on järkevää. (Edureka)

Työmotivaation nousulla tarkoitetaan tässä yhteydessä töiden monipuolistumista. Jos työntekijä joutuu tekemään paljon toistuvaa, työlästä työtä työpäivän aikana, laskee työntekijällä motivaatio työtä kohtaan. Motivaation lasku taas heijastuu suoraan työn laatuun ja -tahtiin. Tekemällä mielekästä työtä työntekijän työtahti, -motivaatio ja kehittyminen ovat nähtävissä (Wikipedia: Robotic Process Automation).

Asiakastyytyväisyys paranee, koska virhetilanteet ratkeavat nopeasti robotin avulla, näin asiakas ei edes ehdi huomata, että tilauksessa on ollut virhe ja sitä kautta hidasteita. Manuaalisessa ihmisten tekemässä työssä voi olla välillä ruuhkaa, jolloin virheen lopulliseen käsittelyyn voi kulua paljon aikaa. Vaikka virheellisiä tilauksia onkin melko vähän suhteessa kaikkiin tilauksiin, on tämä silti hyvä keino vaikuttaa asiakastyytyväisyyteen (Provintl). Tämä heijastuu omalta osaltaan myös myynnin lisäykseen, koska kaikki toimii, kuten pitääkin.



Kuva 3. Ohjelmistorobotiikan edut ja hyödyt (Edureka).

6 VIRHEENHALLINTA

Ohjelmistorobotin määrittelyjä tehdessä ei välttämättä osaa kuvitella tilannetta, jossa robotti tekisi virheen: Ohjelmistorobotin yksi perushyödyistä on nimenomaan vähentää virheitä. Todellisuudessa ohjelmistorobotit ovat virhealttiita, minkä takia oikeanlainen virhetilanteiden hallinta on hyvä suunnitella jo robottia suunnitellessa ja rakentaessa. Robotti kannattaa rakentaa mahdollisimman riskittömäksi muuttamalla epävakaat tilanteet mahdollisimman vakaiksi esimerkiksi ylimääräisillä askelilla. Jos silti tulee virhe, mitä tulee tehdä?

Virheenkorjauslogiikkaa on monenlaista, mutta yleisimmät ovat:

- Toiminnan pysäytys.
- Toiminnan jatko virheen yli.
- Toiminnan seurauksena syntyneen virheen korjaus ja toiminnan jatkaminen.

Tominnan pysäytys tarkoittaa, että ohjelmistorobotti ja sen käyttämät laitteet laitetaan tilaan, jossa kaikki toiminta loppuu. Tämän jälkeen tilannetta tutkitaan ihmistyövoiman avulla. Toimntaa jatketaan heti, kun vika on löydetty ja virhe on korjattu robotin logiikkaan. Tämä on useimmiten huonon suunnittelun tulosta, mutta joskus virhetilanteet tulevat yllättäen, jolloin niihin ei voi varautua. (Symphony HQ)

Toiminnan jatko virheen yli tarkoittaa sitä, että robotti ohittaa virheeseen menneen työvaiheen ja jatkaa tästä eteenpäin kuin virhettä ei olisi tapahtunutkaan. Tämä on todella riskialtis logiikkamalli: jos virheeseen mennyt työvaihe ja sen jälkeinen työvaihe liittyvät toisiinsa, voi koko robotille suunniteltu työ mennä virheeseen ja näin aiheuttaa ihmistyövoiman käyttöä. (Symphony HQ)

Toiminnan seurauksena syntyneen virheen korjaus ja toiminnan jatkuminen on malleista paras. Jos robotille syntyy virhe ajon aikana, yrittää se korjata tilanteen itse. Esimerkiksi, jos robotti yrittää etsiä näytöltä internetselainta ja ei sitä löydä, osaa robotti avata uuden internetselaimen ikkunan ja näin toiminta voi jatkua (Symphony HQ). Tämä ns. tekoälyn luonti ohjelmistorobotille on erittäin suotavaa: Tällä tavoin vältetään koko järjestelmän alasajoa.

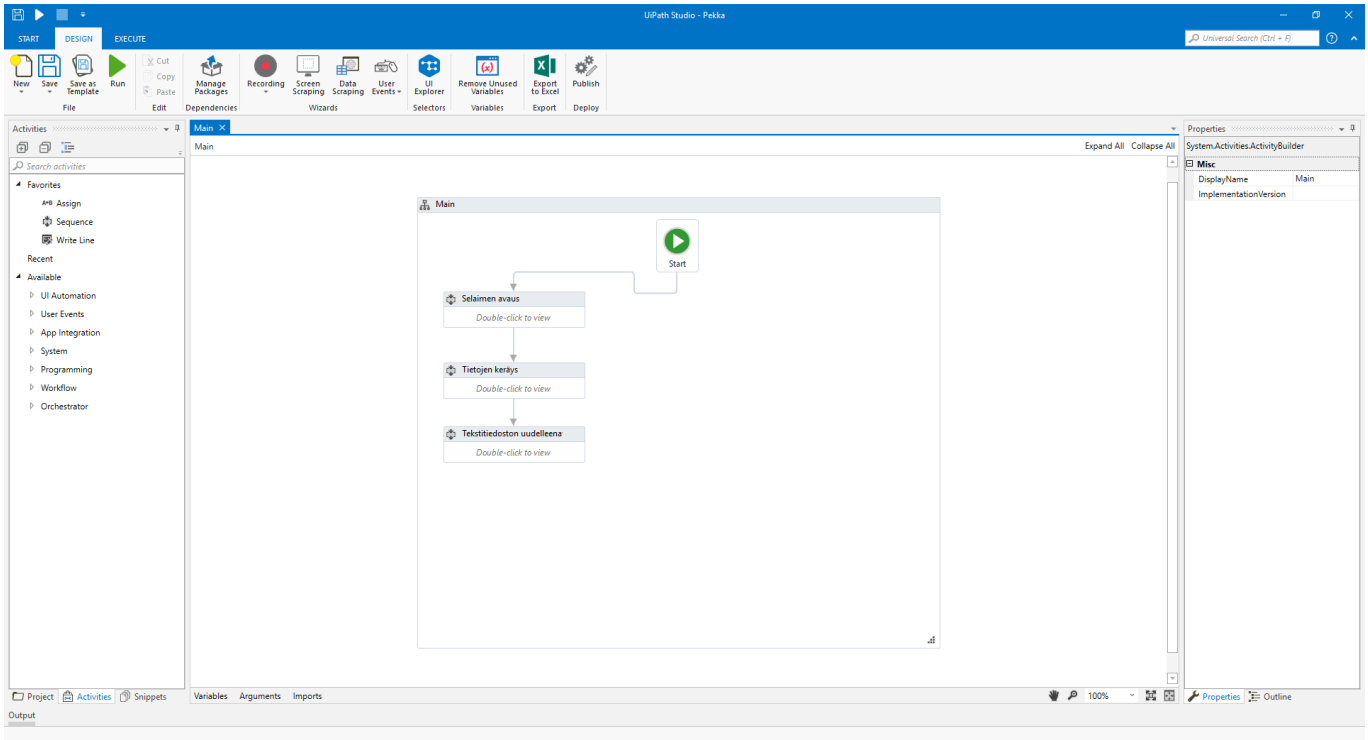
7 ESIMERKKIROBOTIN RAKENTAMINEN UIPATH -SOVELLUKSESSA

UIPath ja siihen liittyvät ohjelmistot ovat ohjelmistorobotin tuleva alusta. Vaikka varsinaista lopullista robottia ei tässä opinnäytetyön puitteissa rakenneta, voin esimerkkirobotin avulla näyttää, miten UIPath sovellus toimii. Tulen työskentelemään UIPath -sovellusten ja -alustan parissa opintojen jälkeen, joten tämä esimerkki toimii pohjana tulevalle lopulliselle ohjelmistorobotille.

Esimerkkirobotti Pekka, joka tässä kappaleessa luodaan, tekee yksinkertaistettuna seuraavat asiat:

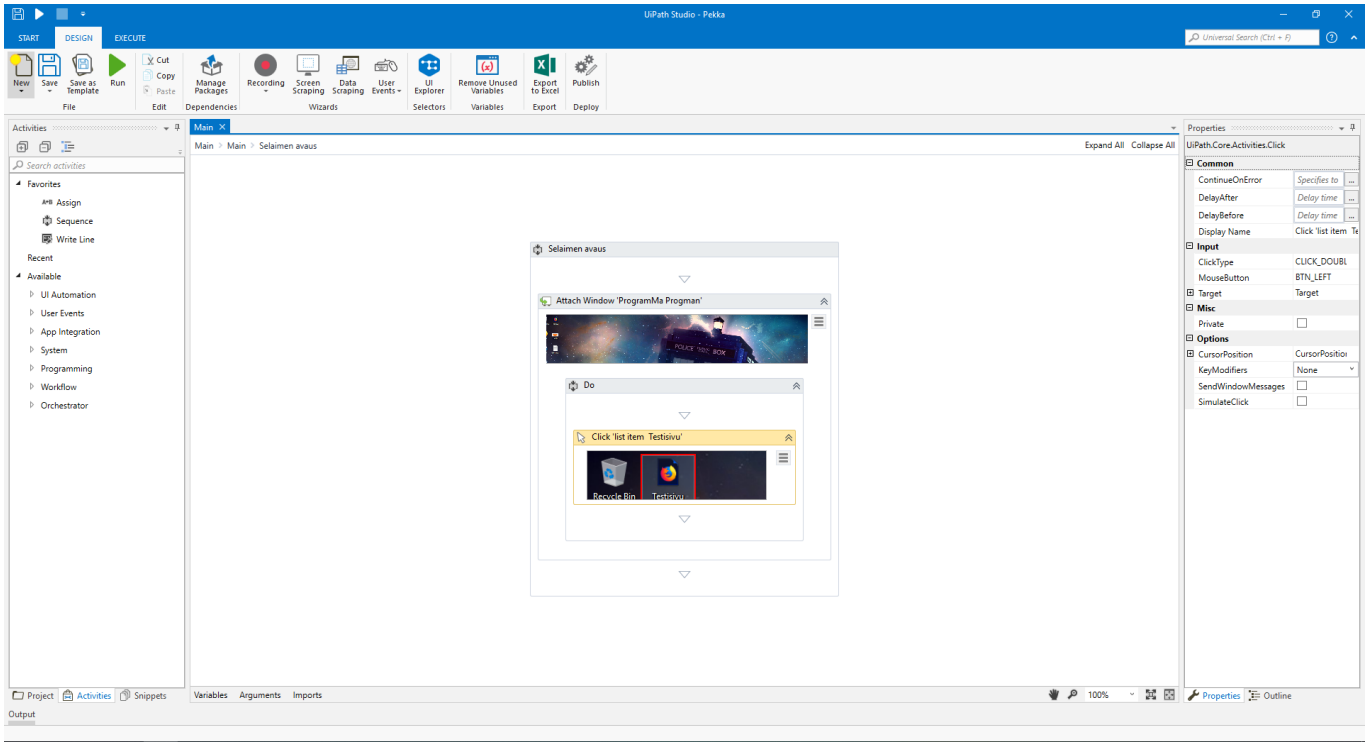
- Avaa internetsivun selaimeen.
- Etsii internetsivulta tietyltä alueelta tietoa.
- Ottaa löydetyt tiedot ylös muuttujaan.
- Sulkee internetselaimen.
- Avaa työpöydältä tekstitiedoston.
- Syöttää tekstitiedostoon aiemmin muuttujaan tallennetut tiedot.
- Tallentaa tekstitiedoston.
- Sulkee tekstitiedoston.
- Avaa tekstitiedoston lopuksi uudelleen esille, jotta tulokset ovat nähtävissä.

Esimerkkirobotti Pekka ei tee työtä silmukassa, eli kun Pekka saavuttaa lopun, suoritus loppuu. Työ olisi helppo ketjuttaa uudelleentoistuvaksi, mutta toistaiseksi kyseistä ominaisuutta ei tarvita. Pekan työ on suoraviivaista, kuten kuvasta neljä näkyy. Työskentelykaaviot ovat UIPathin yksi peruskulmakivistä, ja tämä ominaisuus auttaa selkeyttämään haluttuja prosesseja.



Kuva 4. Esimerkkirobotti Pekan työskentelykaavio UIPath -sovelluksessa

UIPath -sovelluksesta löytyy ”Recording” -ominaisuus, jolla ohjelmisto yrittää seurata, mitä käyttäjä tekee, ja kerää niistä työvaiheita ylös. Ja vaikka tämä voi ajatuksena kuulostaa yksinkertaiselta tavalta luoda robotti, liittyy siihen riskejä ja lisävaiheiden luontia. Ensinnäkin, se ei ole tarkka, eli kohdistuksia joutuu tekemään jälkeenpäin. Lisäksi, se ei ymmärrä tuplaklikkausten, tekstin kopioinnin, sekä muuttujaan lisäystä, vaan nämä on itse lisättävä robotin logiikkaan jälkikäteen.



Kuva 5. Internetsivun avaaminen selaimeen

Esimerkkirobotti Pekan rakentaminen aloitettiin transaktiopohjalla, eli robotti etenee yksi vaihe kerrallaan, odottaen, että senhetkinen työvaihe on tehty loppuun ennen uuden aloitusta. Internetselaimen avaamisprosessi aloitettin Pekalle ensin näyttämällä maalaamalla alue työpöydältä, mitä kohtaa Pekan tulee klikata. Lähtökohtaisesti klikkaus on vain yksittäinen klikkaus ja se tapahtuu keskelle määritettyä aluetta. Jotta internetselain aukeaisi, on klikkaus vaihdettava tuplaklikkaukseksi (Kuva 5.).

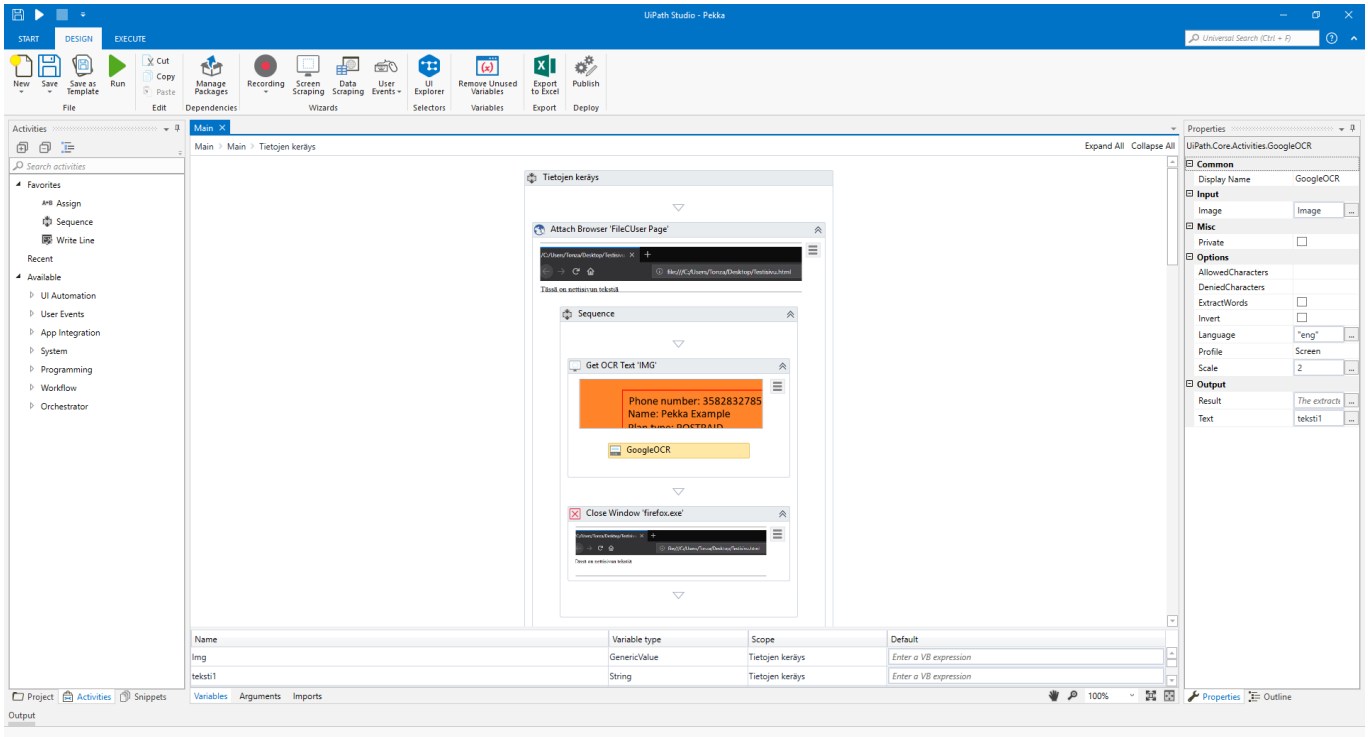
UIPath käsittelee jokaisen näkymän omana vaiheenaan, ja vaiheen sisällä olevat työvaiheet ovat ”Do” -rakenteen sisällä. Rakenne muistuttaa etäisesti graafista ohjelmointia, jossa erilaisilla palikoilla saadaan aikaan erilaisia asioita, kuten vaikkapa sekvenssejä (Kuva 5.).

Vasemmalla näkyvässä ”Activities” -valikosta voidaan lisätä haluttuja työvaiheita, kuten vaikkapa esimerkkinä sähköpostin lähetys Outlookin tms. kautta. Sovellus tarjoaa runkomalleja erilaisiin tilanteisiin ja niitä voi muokata laajalti oikealla olevassa ”Properties” -palkista.

Seuraavaksi on vuorossa tiedon etsintä internetsivulta. Ihminen osaa helposti erotella tekstin kuvasta, mutta tietokone ei. Onneksi maailmalla on kehitetty tietokonenäköä, jonka avulla tietokone ja sitä kautta ohjelmistorobotti osaa ”nähdä” kuin ihminen, ja tunnistaa asioita kuvasta. Tätä paljon

puhuttua ominaisuutta löytyy nykyään jo älypuhelimistakin, missä sitä hyödynnetään valokuvauksessa tunnistamaan suotuisimmat asetukset kohteesta riippuen.

OCR: ään eli tekstintunnistukseen (eng. Optical Character Recognition) löytyy UIPathistä työkalu, jonka avulla ohjelmistorobotti voi yrittää löytää halutulta alueelta tekstiä. Vaihtoehdot tekstintunnistukseen käytettäväksi ”moottoriksi” on valittavana Microsoftin ja Googlen konenäöt. Ne toimivat kumpikin omalla tavallaan, mutta lopputulos on useimmiten sama. Joissain tilanteissa toinen näistä saattaa löytää tekstiä paremmin kuin toinen.

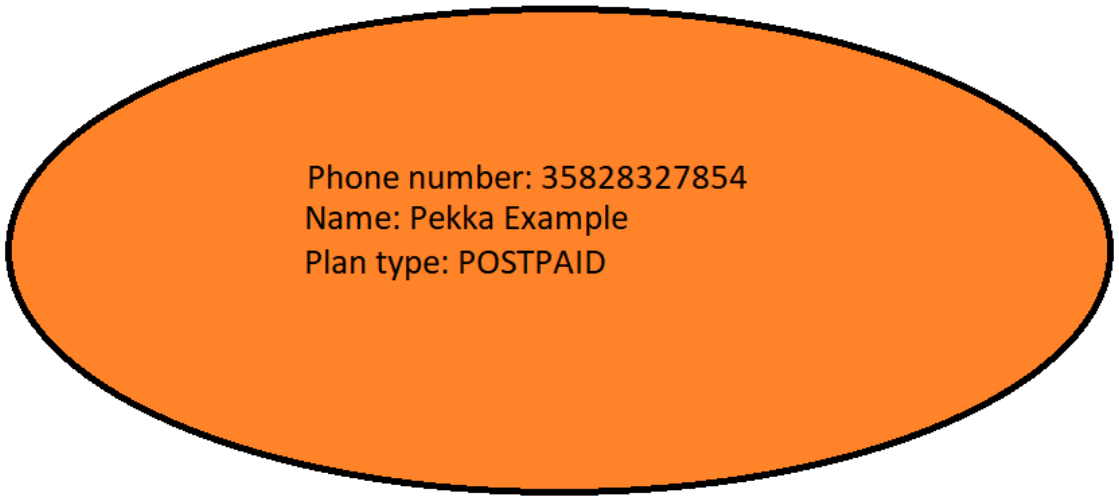


Kuva 6. Tiedon etsinnän ja sen muuttujaan tallentaminen

Tämän robotin käytössä hyödynnettiin Googlen konenäköä ja esimerkkinä toimineessa internetsivussa oli monta erilaista kohtaa, joilla pyrittiin hämäämään konenäköä. Haluttu tieto oli oranssiin ovaalin muotoiseen kuvaan laitettuna, joka oli .png kuva (Kuva 7.). Konenäkö tunnisti tekstit kuvasta ja tarjosi suoraan mahdollisuutta tallentaa ne leikepöydälle. Internetsivulta saadun tiedon tallennus osoitettiin ”teksti1” -nimiseen muuttujaan, jolloin tieto pysyy tallessa myöhempää käyttöä varten (Kuva 6.).

Tämän jälkeen oli vuorossa internetselaimen sulkeminen. UIPath tarjoaa tähän kaksi erilaista vaihtoehtoa, klikkaus ja sovelluksen sulku. Klikkaus tarkoittaa tietystä kohdasta näyttöä klikkaamista, ja sen suurin heikkous piileekin juuri siinä: jos klikattavan alueen ympäristö muuttuu, robotti ei välttämättä enää löydä referenssikuvaan perustuvaa aluetta, vaan se lopettaa toimintansa siihen. Sovelluksen sulku taas osaa sulkea sovelluksen ikkunan automaattisesti (Kuva 6.). Tämä ei aina toimi suunnitellusti, varsinkin jos ikkunassa ei ole näkyvää sulkemisnappia. Tällöin on käytettävä klikkausta.

Tässä on nettisivun tekstiä



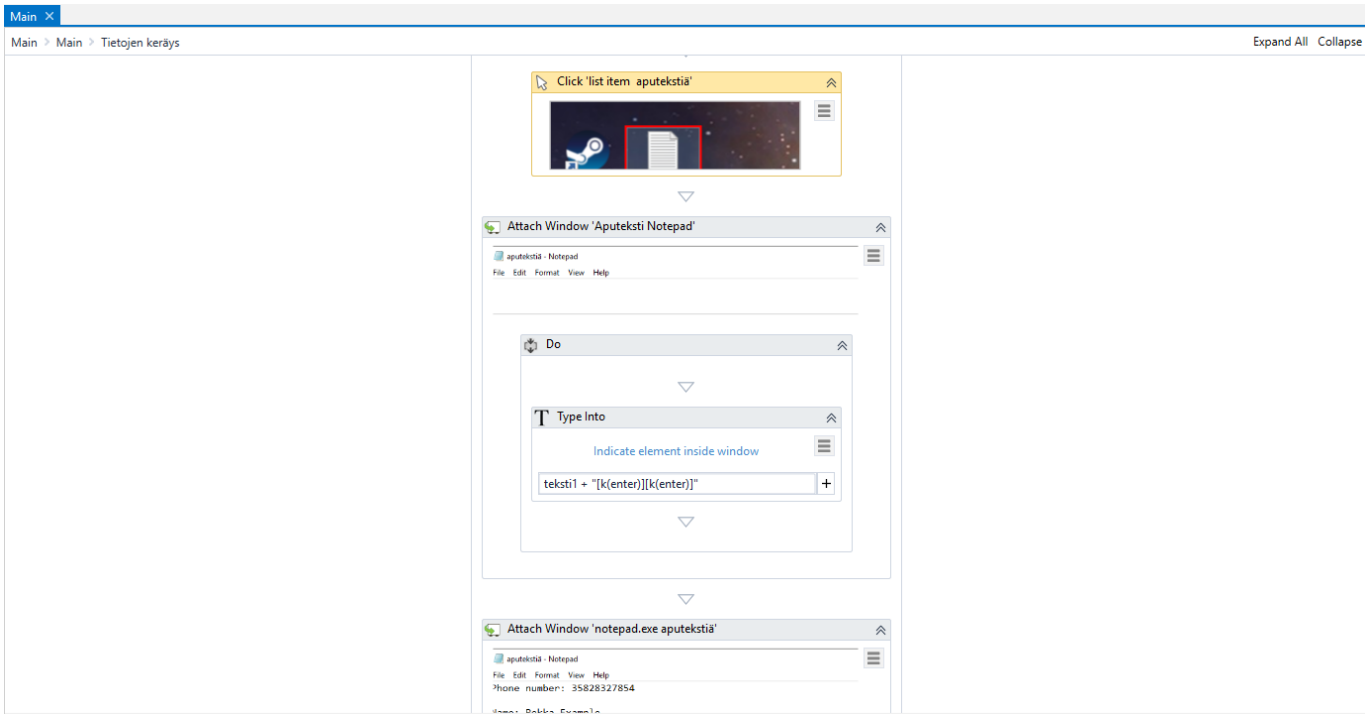
Account holder	Phone Number	TYPE	Error Type
Pekka Example	3589564534	POSTPAID	500
Liisa Simple	3584354324	PREPAID	389

Kuva 7. Esimerkkinä toiminut internetsivu

Robotin suljettua internetselaimen ikkunan, on vuorossa tekstitedoston avaus. ”aputekstiä” -niminen tekstitiedosto löytyy työpöydältä, joten robotti pääsee suoraan klikkaamaan sitä aikaisemman työvaiheen jälkeen. Työympäristö kannattaakin pitää yksinkertaisena: näin vältytään ongelmista myöhemmässä vaiheessa. Jälleen kerran, klikkaus on muutettava tuplaklikkaukseksi, jotta kyseinen tekstitiedosto aukeaisi.

Tämän jälkeen on seuraavana työvaiheena tekstin kirjoittaminen avoinna olevaan tekstitiedostoon. Tämä aloitetaan osoittamalla alue, mihin tekstin haluaa kirjoitettavan. Vaikka tämä voi tuntua lähinnä triviaalilta, on tärkeää osoittaa robotille oikeat työskentelykohteet. Robotti saattaa toimia täysin oikein, vaikka paikkoja ei tarkemmin määrittelisi, mutta määrittelemällä kohteet oikein, robotti voi todennäköisemmin toimia oikein, vaikka ympäristö hieman muuttuisikin.

Tekstin kirjoitukseen on valittavana kaksi vaihtoehtoa, ”Type Into” ja ”Copy and Paste”. ”Type Into” toimii samalla tavalla kuin ihminen kirjoittaisi tekstin todella nopeasti, kun taas ”Copy and Paste” liittää tekstin suoraan esimerkiksi leikepöydältä tekstikenttään. Tässä robotin määrittelyssä käytettiin ”Type Into” -tekstinsyöttöä, vaikka eroa näiden tekstinsyöttötapojen välillä ei käytännön toteutuksen kannalta juurikaan ole. Lisäksi robotti määriteltiin tekemään kaksi erikoismerkkiä, tässä tapauksessa Enter -näppäimen painallusta, tekstin loppuun, jotta robottia uudelleenajaessa lopputulokset ovat selkeämmin nähtävissä (Kuva 8.).



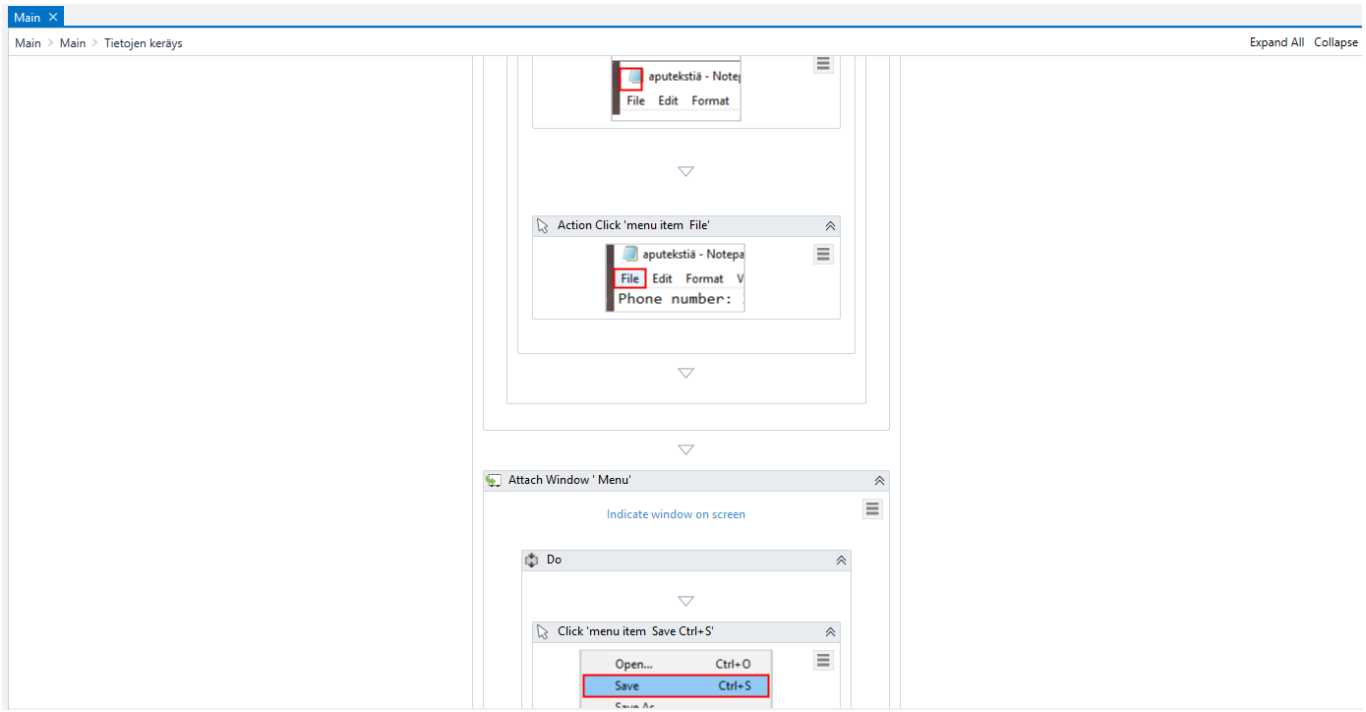
Kuva 8. Tekstitiedoston avaus ja tiedon syöttö

Tässä vaiheessa robotti on nyt syöttänyt internetsivulta löytyneen tekstin tekstitiedostoon, mutta ei ole vielä tallentanut sitä. Tekstitiedoston tallentamisessa on kolme työvaihetta:

1. Klikkaa ”File” -kohtaa tekstieditorin navigaatiopalkista
2. Määritä ympäristöstä luotettava ankkuri
3. Klikkaa avautuvasta valikosta kohtaa ”Save”

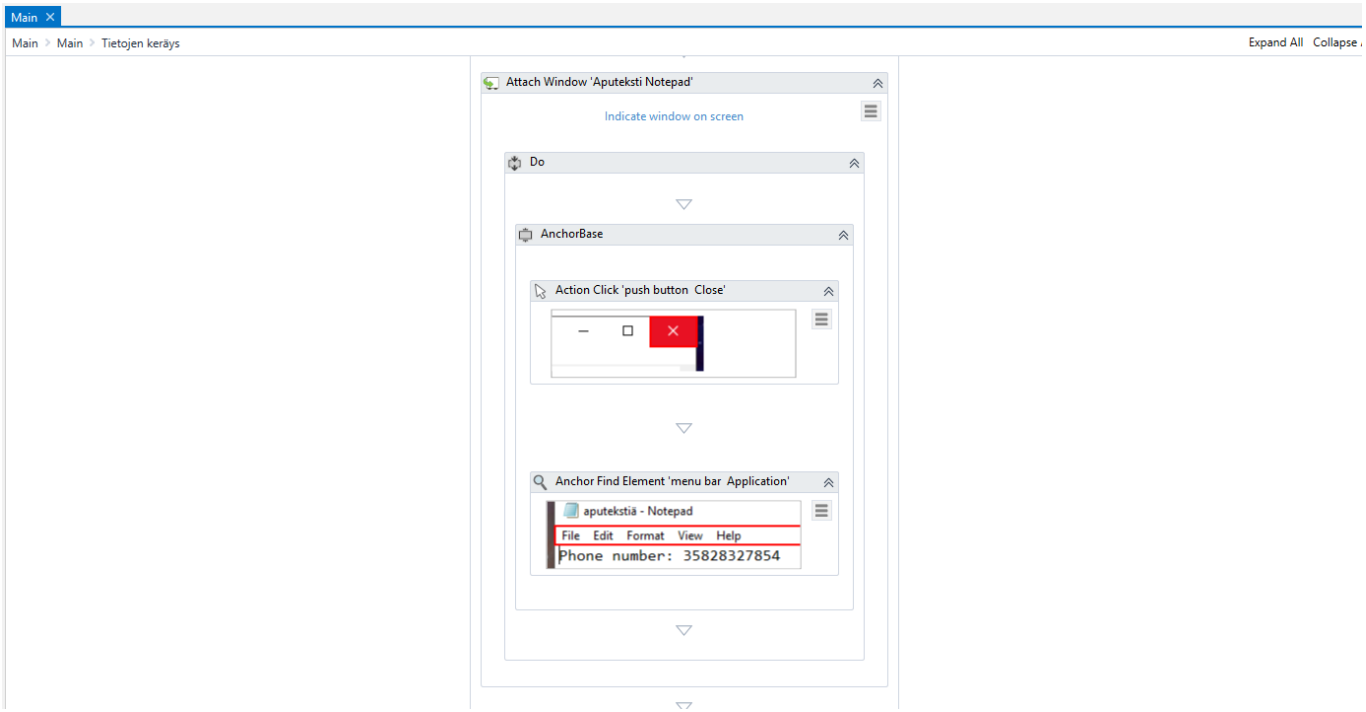
Ensimmäisessä vaiheessa määritellään maalaamalla alue näytöstä, mistä kohtaa halutaan klikkauksen tapahtuvan. Sovellus tunnistaa älykkäästi halutun ”File” -kohdan osaksi menua, joten seuraava vaihe on määritellä sille ankkuri, joka on luotettava eikä muutu. Ankkuri on tässä yhteydessä kiintopiste näytöllä, jonka sijainnin avulla robotti tietää, mistä oikean kohdan löytää. Halu käyttää ankkuria liittyy menun yleisyyteen, se on oletusmenu monessa Windows -pohjaisessa sovelluksessa. Mutta koska käytämme vain yhtä tekstieditoria, tämän voi määritellä osoittamaan tekstieditorin ikoniin (Kuva 9.).

Ankkurin määrittelyn jälkeen on vuorossa ”Save” -napin klikkaaminen. On huomionarvoista, että UIPath käsittelee syntyneen valikon uutena näkymänä, koska sitä ei ollut aiemmin nähtävissä näytöllä. Tämä työvaihe toimii ilman ankkuria, koska se on määritelty aiempaan työvaiheeseen (Kuva 9.).

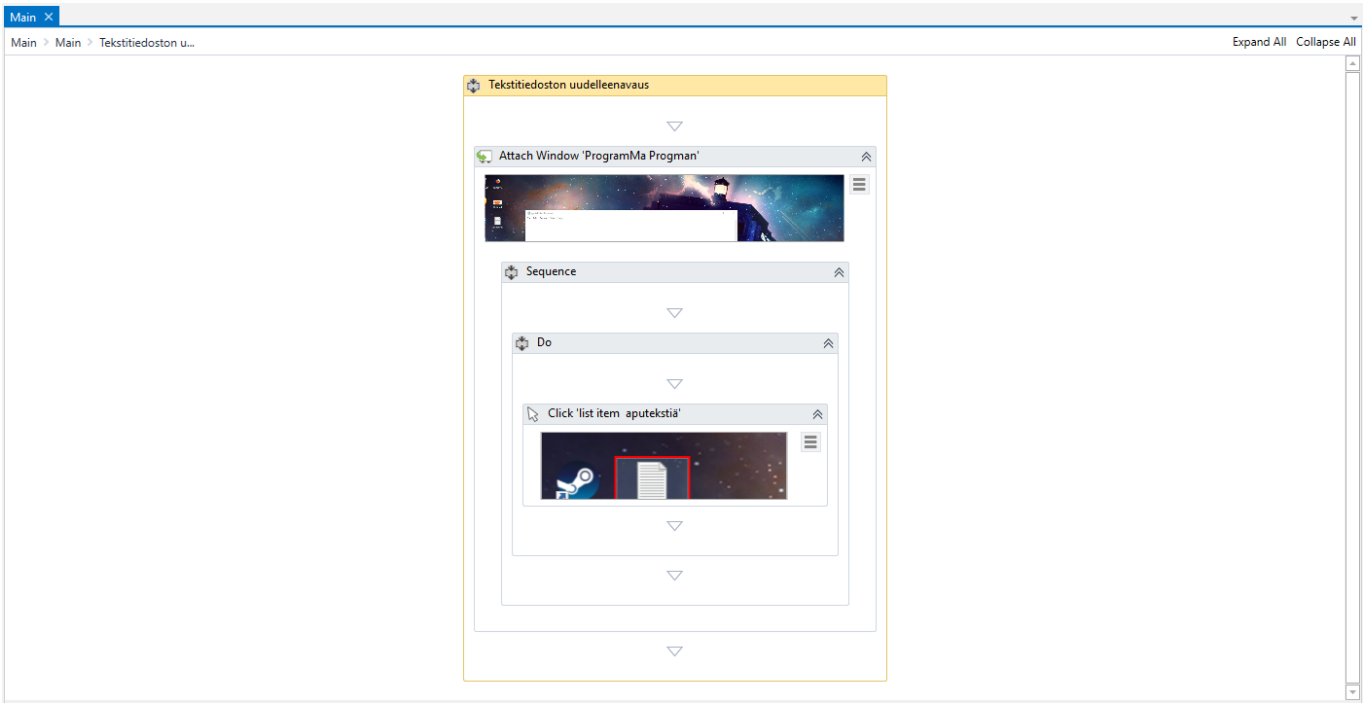


Kuva 9. Tekstitiedoston tallennus

Tekstitiedosto on nyt tallennettu. Seuraavaksi vuorossa tekstitiedoston sulkeminen. Tässä esimerkissä se on toteutettu klikkauksella, koska ympäristössä ei ole käytössä kuin yksi tekstieditorin ikkuna, löytää robotti kohteen ankkurin avulla. Ankkuriksi laitetaan tekstieditorin menu, koska se ei toistu näytöllä missään, joten virhemarginaali on todella pieni. Vaikka sovellus käsittää tämänkin työvaiheen uutena ikkunana, ei sille kuitenkaan tarvitse määritellä tarkkaa ikkunaa johtuen klikkauksen ankkurista. Vaikeammissa työympäristöissä se olisi väistämättä lähes pakollista (Kuva 10.).



Kuva 10. Testitiedoston sulkeminen



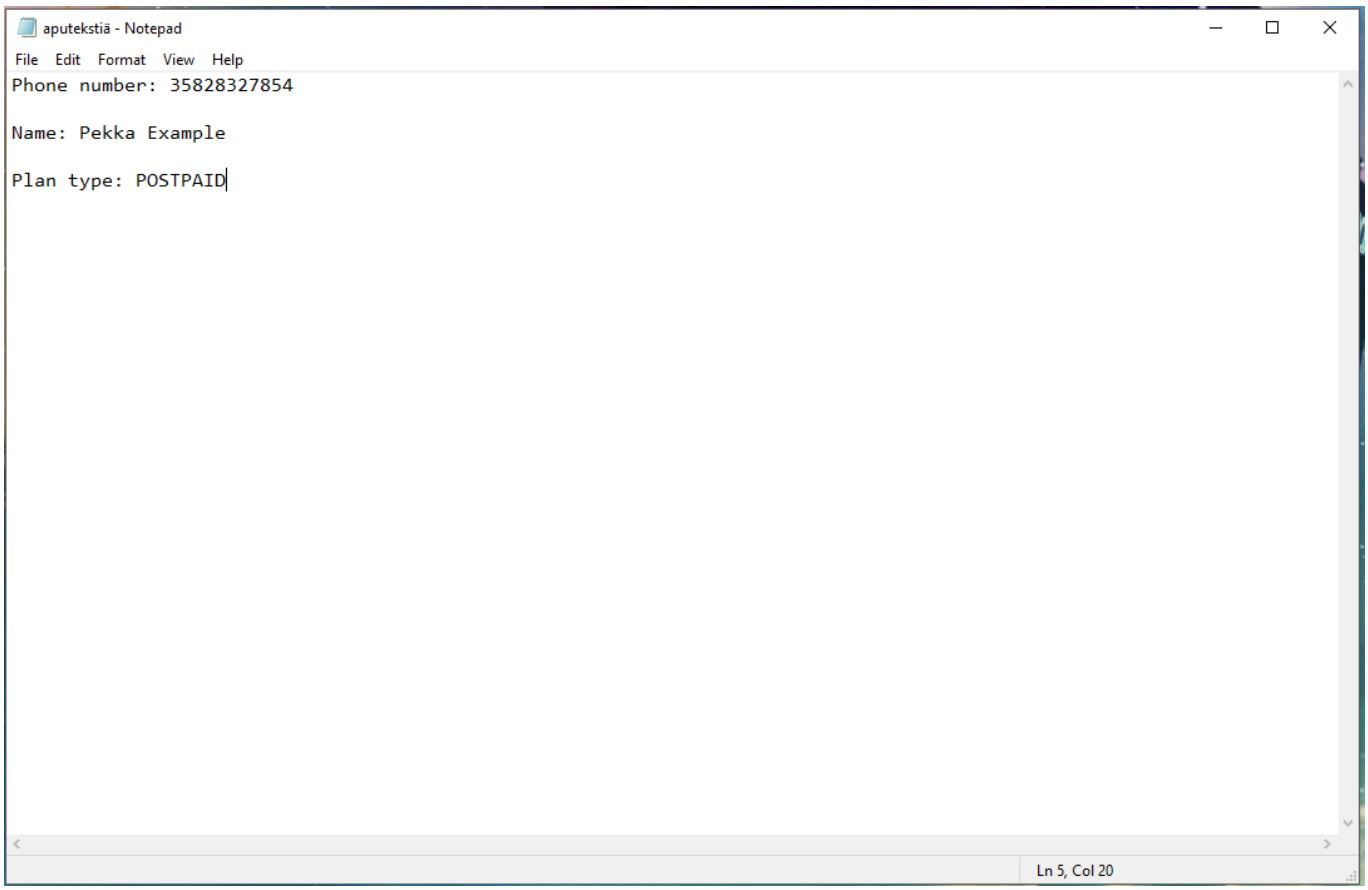
Kuva 11. Tekstitiedoston uudelleenavaus

Seuraava työvaihe on vain omaa tarkastelua varten. Robotti avaa tekstitiedoston ”aputekstiä ”uudelleen ja jättää sen tällä kertaa auki. Tämä työvaihe on toteutettu samalla tavalla kuin aiemmin: klikattava kohta on ”maalattu” eli robotille on näytetty mitä aluetta robotin tulee klikata ruudulla, ja tämän jälkeen klikkaus on muutettu tuplaklikkaukseksi, jotta tekstitiedosto aukeaa (Kuva 11.). Robotin suoritus loppuu tähän.

Kuten aiemmin on todettu, tämä esimerkki ei aja itseään loputtomasti, vaan tehtyään kaikki nämä työvaiheet, robotin suoritus katkeaa. Robotin työvaiheiden luonti ei ollut vaikeaa, mutta alussa vaiheiden luontia esti Nvidian Shadowplay -ominaisuus. Shadowplay -ominaisuus on Nvidian näytönohjaimien näytöntallennussovellus. UIPathin ohjeissa asiasta ei ollut mitään puhetta, vaan asiaa joutui itse selvittämään. Ottamalla Shadowplayn pois käytöstä, UIPathin kaikki toiminnallisuus palasi.

Jos robotin ajaa, tekee se kuten se on määritelty. Robotti avaa internetsivun selaimeen, kopioi tiedot sieltä, sulkee selaimen, avaa tekstitiedoston, tallentaa tiedot sinne, ja sulkee tekstieditorin. Lopuksi robotti avaa tekstitiedoston uudelleen. Lopputilanne on, että työpöydällä on auki vain tekstieditori (Kuva 12.). Mikäli robottia ajaa toistuvasti, lisääntyy tekstin määrä, eli robotti ei tyhjennä vanhoja tekstejä pois ”aputeksti” -tiedostosta. Mikäli näin haluttaisiin tehdä, on sitä varten tehty sovellukseen yksinkertainen valintaruutu.

Robotilla kuluu aikaa tämän työn suorittamiseen vain noin viisi sekuntia, kun taas ihmisen tekemänä työn tekemiseen kuluisi noin minuutti. Nopeuserot työn suorittamiseen ovat valtavat.



Kuva 12. Robotin työvaiheista syntynyt ”aputekstiä” -tekstitiedoston sisältö.

8 POHDINTA

Ohjelmistorobotiikka on yrityksissä paljon puhuttu asia. Vanhoista fyysisistä tuotantoroboteista 60-luvulta katse nykypäivään ja robotiikka on siirtynyt vahvasti tieteknisten ratkaisujen piiriin. Työn automatisaatio varsinkin rutiininomaisissa työtehtävissä on järkevää ja ohjelmistorobotiikka tarjoaa mahdollisuudet siihen.

Tämän opinnäytetyön tavoitteena oli tutustua ohjelmistorobotiikkaan ja sen tarjoamiin mahdollisuuksiin, paneutua manuaalisesti tehtyihin virheenkorjaustyötehtäviin ja luoda vaatimusmäärittelyt ensimmäiselle testirobotille. Näiden lisäksi tässä opinnäytetyössä ja vaatimusmäärittelyissä näytettiin automatisaation hyödyt, jotka näissä työtehtävissä olisi saavutettavissa. Tutustuminen ohjelmistorobotikkaan eteni opiskelemalla UIPath -akatemiassa, jossa opin käyttämään UIPath Orchestrator -ohjelmistorobotiikan alustaa. Osana tutustumista aiheeseen tein esimerkkirobotin, jonka avulla automatisaation hyötyjä, kuten ajankäyttöä, pystyy näkemään konkreettisesti. Luodun esimerkkirobotin pohjaa tullaan hyödyntämään testirobotin suunnittelussa.

Vaatimusmäärittelyiden suunnittelu ja toteutus oli uutta minulle. Tutustuin vaatimusmäärittelyiden erilaisiin määritelmiin internetin avulla ja lisäksi selvitin yrityksen sisäistä tapaa tehdä vaatimusmäärittelyitä. Vaatimusmäärittelyitä tehdessäni huomasin, että ohjelmistorobotin vaatimusmäärittelyihin kuuluu paljon muutakin kuin vain hyvin kuvailtu työohje tehtävän suorittamiseen. Esimerkiksi virheenhallinta ohjelmistorobotille jo suunnitteluvaiheessa on tärkeä asia, jotta vältetään jatkuvat ongelmatilanteet ja käyttökatkot. Vaatimusmäärittelyt viedään DNA:lle testirobotin suunnitteluun ja rakentamiseen.

Vaatimusmäärittelyiden päätöksentekoon vaikuttanut kerätty aineisto muutti merkittävästi alkuperäistä ajatustani parhaimmasta automatisoitavasta työtilanteesta. Vaikka olin manuaalisesti kyseisiä työtehtäviä tehnyt aiemmin, ei minulla ollut pidemmän ajan näkymää ja tietoa työtehtävien yleisyyteen. Tämän sain paremmin selville muilta työntekijöiltä sekä DNA:n tietokannoista.

Automatisointia ei kannata tehdä työtehtäville, jotka eivät noudata samaa kaavaa tai ovat todella harvinaisia. Vertailemalla virheenkorjaustyötehtävien eri tilastoja, kuten esimerkiksi virheenkorjaustyötehtävän yleisyyttä ja sen korjaukseen kuluvaa aikaa, päästiin lopulta yhteen automatisoitavaan työtehtävään. Tämän lisäksi saatiin ”roadmap” tuleviin automatisoitaviin työtehtäviin.

Kokonaisuutena voidaan sanoa, että opinnäytetyön toteutus onnistui mallikkaasti. Aluksi oli pelko ajan riittävyydestä työmäärään nähden, mutta säännöllisellä työskentelyllä opinnäytetyöraportti saatiin tehtyä juuri sellaiseksi, kuin oli alun perin suunniteltu. Opinnäytetyön sivutuotteena syntyneet vaatimusmäärittelyt saatiin myös tehtyä aikataulun mukaisesti. Esimies oli näihin erittäin tyytyväinen.

LÄHTEET JA TUOTETUT AINEISTOT

KAPPAGANTULA, Sahiti 2018-10-23 What is Robotic Process Automation? – An Introduction to RPA [Verkkoaineisto] [Viitattu 2018-11-01] Saatavissa: https://www.edureka.co/blog/what-is-robotic-process-automation/

Provintl 2018. Top-5 Benefits of Robotics Process Automation (RPA) Adoption for Your Company [Verkkoaineisto] [Viitattu 2018-11-13] Saatavissa: https://www.provintl.com/blog/top-5-benefits-of-robotics-process-automation-rpa-software

Symphony HQ 2017. RPA Technical Insights, Part 14: How Exception Handling Can Save Your Automation From Failure [Verkkoaineisto] [Viitattu 2018-11-15] Saatavissa: http://blog.symphonyhq.com/rpa-technical-insights-part-14-exception-handling-overview

Wikipedia 2018. Robotic Process Automation [Verkkoaineisto] [Viitattu 2018-11-01] Saatavissa: https://en.wikipedia.org/wiki/Robotic_process_automation

Wikipedia 2017. Ohjelmistorobotiikka [Verkkoaineisto] [Viitattu 2018-11-01] Saatavissa: https://fi.wikipedia.org/wiki/Ohjelmistorobotiikka

Wikipedia 2018. UIPath [Verkkoaineisto] [Viitattu 2018-11-03] Saatavissa: https://en.wikipedia.org/wiki/UiPath

Wikipedia 2018. DNA (Yritys) [Verkkoaineisto] [Viitattu 2018-11-10] Saatavissa: https://fi.wikipedia.org/wiki/DNA_(yritys)

UIPath 2018. UIPath [Verkkoaineisto] [Viitattu 2018-11-03] Saatavissa: https://www.uipath.com/hubfs/Valentin/Brand%20Kit/Company%20Overview.pdf

Talouselämä 2017. Karolinan työkaveri on ohjelmistorobotti – Ohjelmistorobotiikan aika on nyt [Verkkoaineisto] [Viitattu 2018-11-18] Saatavissa: https://www.talouselama.fi/uutiset/karolinan-tyokaveri-on-ohjelmistorobotti-ohjelmistorobotiikan-aika-on-nyt/0b2b57b3-d25e-33fe-98f2-96ae919be0a5

Affecto 2018. Ohjelmistorobotiikka [Verkkoaineisto] [Viitattu 2018-11-09] Saatavissa: http://www.affecto.com/fi-fi/palvelut/sovellusratkaisut/ohjelmistorobotiikka/