Tampereen ammattikorkeakoulu

Ohjelmistorobotiikan perusteet ja prosessin toteutus

Antte Hursti

OPINNÄYTETYÖ
Joulukuu 2022

Tieto- ja viestintätekniikka
Tietoliikennetekniikka ja tietoverkot

TIIVISTELMÄ

Tampereen ammattikorkeakoulu
Tieto- ja viestintätekniikka
Tietoliikennetekniikka ja tietoverkot

HURSTI, ANTTE:
Ohjelmistorobotiikan perusteet ja prosessin toteutus

Opinnäytetyö 30 sivua, joista liitteitä 1 sivu
Joulukuu 2022

Opinnäytetyössä esiteltiin ohjelmistorobotiikan eli RPA:n käyttötarkoitusta ja sen ylätason toimintaperiaatetta. Työssä tutustutaan lyhyesti myös RPA:n historiaan sekä sen lähivuosien taloudelliseen kasvuun.

Opinnäytetyössä parannetaan ohjelmistorobotiikasta kiinnostuneiden ymmärrystä sen hyödyistä, ominaisuuksista ja mahdollisuuksista. Työssä esitellään millaisia vaiheita ohjelmistorobotiikkaprosessin toteuttaminen voi sisältää ja miten prosessin automatisoiminen käytännössä tapahtuu.

Projektissa syvennyttiin myös ohjelmistorobotiikkaprosessin eri toteuttamistapoihin, näiden hyötyihin ja haittoihin sekä mihin tarkoitukseen ne sopivat. Tämän lisäksi esitellään myös dokumentaatiota, mitä RPA-prosessin toteuttamisen yhteydessä usein luodaan.

Jatkotutkimuksena opinnäytetyöhön voisi selvittää, miten kolmannen osapuolten raportointi- ja hallintatyökaluja, kuten Microsoft Power BI:tä tai RPA Supervisor:ia, voisi hyödyntää itse ohjelmistorobotiikkaohjelmiston rinnalla. Näiden sovelluksien hyödyntäminen voi tuoda huomattavia hyötyjä sekä lisätä läpinäkyvyyttä digitaalisten ja ihmistyöntekijöiden välillä.

Asiasanat: ohjelmistorobotiikka, automatisointi, liiketoiminta, kehitys

ABSTRACT

Tampereen ammattikorkeakoulu
Tampere University of Applied Sciences
Degree Programme in ICT Engineering
Telecommunications and Networks

HURSTI, ANTTE:
The basics of RPA and the implementation of a process

Bachelor's thesis 30 pages, appendices 1 page
December 2022

In this thesis, the intended use and the high-level operating principle of robotic process automation or RPA were explained. The history and recent financial growth were also touched upon.

The goal of this thesis is to educate parties interested in robotic process automation about its benefits, features and possibilities. On top of that, the aim is to provide an example on what kind of phases the development process includes and how it would be executed in practice.

In the project, more in-depth information was given about different methods as to how an RPA process could be implemented and what are their pros and cons. Some insight about the documentation that is often created during the implementation of a process is also offered.

As an additional study to this thesis, it would be interesting to explore the possibilities of third-party reporting and management software, such as Microsoft Power BI or RPA Supervisor alongside the actual RPA software that is being used. The use of these software could bring notable benefits and increase transparency between the traditional and the digital workers.

Key words: robotic process automation, automation, business, development

SISÄLLYS

LYHENTEET JA TERMIT

API	Application Programming Interface eli ohjelmointirajapinta
Application Modeller	Ohjelmiston mallintaja
Attach-vaihe	Vaihe, joka yhdistää ohjelmistorobotiikkaohjelmiston automatisoitavaan sovellukseen
Blue Prism	Ohjelmistorobotiikkaohjelmisto
Business Exception	Liiketoimintapoikkeus
Debug	Virheenjäljitys
End-vaihe	Lopetusvaihe, johon prosessi, prosessin alisivu tai objektin toiminto päättyy
FTE	Full-Time Equivalent
HIL	Human-in-the-loop, ihmisen syötettä tarvitaan kesken automaatioprosessin ajon
Objekti	Loogisesti jaoteltu toimintoja sisältävä kokonaisuus, jota hyödynnetään prosessitason logiikan kokoamisessa
PDD	Process Definition Document eli prosessin määrittelydokumentti
RPA	Robotic Process Automation eli ohjelmistorobotiikka
SME	Subject Matter Expert, automatisoitavan manuaalisen prosessin hyvin tunteva henkilö
System Exception	Järjestelmäpoikkeus
Start-vaihe	Aloitusvaihe, josta prosessi, prosessin alisivu tai objektin toiminto aloittaa ajamisen
UI	User Interface eli käyttöliittymä
UAT	User Acceptance Testing eli hyväksymistestaus
Wait-vaihe	Odotusvaihe, ohjelmistorobotti odottaa sille määritellyn ajan tai tietyn elementin havaintoa
Wildcard	Villikortti tunnistusehto elementin attribuutin arvolle

1 JOHDANTO

Tämän opinnäytetyön aiheeseen tutustuminen ja aiheen rajaus on tehty yhteistyössä Eetu Lahtisen kanssa. Alun perin opinnäytetyömme oli tarkoitus toteuttaa kokonaisuudessaan parityönä, mutta päädyimme tekemään erilliset kirjalliset opinnäytetyöt aikataulullisten hankaluuksien seurauksena. Idea opinnäytetyölle syntyi, kun pääsimme työskentelemään täysipäiväisesti ohjelmistorobotiikan parissa.

Ohjelmistorobotiikka eli RPA on suhteellisen yksinkertainen automatisoimistapa, jolla pystytään jäljittelemään ihmisen syötettä digitaalisissa järjestelmissä. Lähivuosina RPA on kuitenkin kehittynyt käyttämään monimutkaisempiakin teknologioita, kuten koneoppimista ja muita tekoälyn osa-alueita. Ohjelmistorobotiikka on hyvin tehokas tapa automatisoida suuria tapausvolyymeja sisältäviä yksitoikkoisia prosesseja.

Ohjelmistorobotiikka on yhä tuore teknologia – kehitystä on 2010-luvun jälkimmäisellä puoliskolla tapahtunut huomattavasti ja se näkyy myös ohjelmistorobotiikan suosion nousuna. Uusia RPA-ohjelmistoja on myös kehitetty ja kehitetään edelleen, kun yhä useampi yritys päättää automatisoida rutiininomaisia työtehtäviään ja tämän seurauksena RPA työllistää yhä suurempia määriä asiantuntijoita.

Tässä opinnäytetyössä tarkastellaan yleisesti ohjelmistorobotiikan mahdollisuuksia, hyötyjä ja mahdollisia haittoja lähinnä liiketoiminnan näkökulmasta. Työssä selvitetään miten ohjelmistorobotiikkaprosessit eli niin sanotut digitaaliset työntekijät käytännössä toimivat, millä tavalla ne kehitetään alusta loppuun ja minkälaista työpanosta se vaatii RPA-ratkaisun tarjoajalta kuin myös automatisoinnin vastaanottavalta osapuolelta.

2 OHJELMISTOROBOTIIKKA

Lyhenne RPA tulee englannin kielen sanoista Robotic Process Automation, joka käännetään suomeksi yleensä ohjelmistorobotiikaksi. RPA:ta käytetään rutiininomaisien ja toistuvien digitaalisten prosessien automatisointiin, joita tehdään suuria määriä manuaalisesti.

2.1 Mitä on RPA?

Ohjelmistorobotiikkaa voidaan hyödyntää monella eri tapaa. Manuaalisia prosesseja voidaan automatisoida täysin tai osittain – täysin automatisoitavat prosessit eivät kuitenkaan saa sisältää ihmisen harkintaa vaativaa päätöksentekoa. Kaikelle täytyy olla hyvin selvästi määritellyt säännöt, jotta prosessi voidaan automatisoida käyttäen ainoastaan RPA:ta. On kuitenkin mahdollista tehdä prosesseja, joissa ihminenkin on manuaalisesti mukana työskennellen yhdessä robotin kanssa. Tällaista automaatiotapaa kutsutaan nimellä Human in the Loop (HIL).

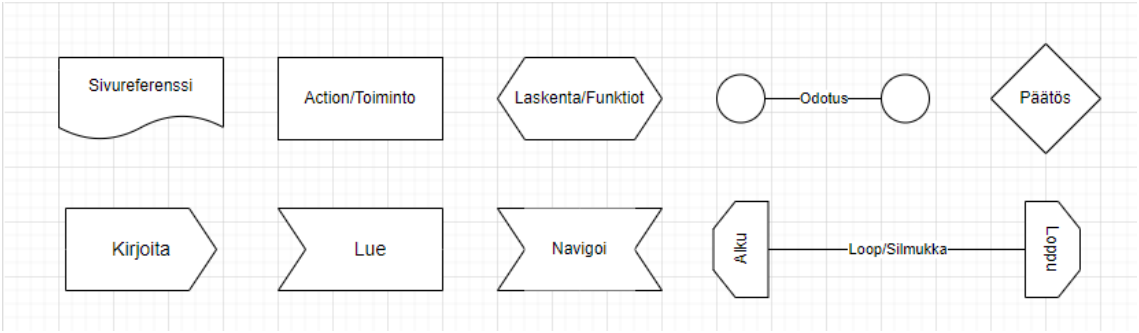
RPA:n valttikortti varsinkin sen alkutaipaleilla oli sen tapa hyödyntää automatisointiin samaa käyttöliittymää, mitä ihmistyöntekijäkin käyttää. Tällöin se olisi asiakkaalle mahdollisimman yksinkertaista, eikä heidän tarvitsisi muuttaa mitään robotin suorittaakseen tehtävänsä. Näin ollen myös kohdejärjestelmät voivat olla mitä vain, eikä automatisoitavat prosessit rajoitu vain sellaisiin ohjelmiin tai järjestelmiin, jotka ovat tehty automatisointi mielessä pitäen. Nykypäivänä kuitenkin ohjelmistorobotiikkaa toteutetaan myös muilla tavoilla, kuten ohjelmointirajapintojen kautta.

Ohjelmistorobotiikan yhteydessä puhutaan yleensä roboteista ja tämä voi olla hämmentävää henkilölle, jolle RPA on konseptina uusi. Kun tässä yhteydessä puhutaan robotista, tarkoitetaan nimenomaan automaatiota, joka toimii jonkin ohjelmistorobotiikkatyökalun kautta. Robotti tässä asiayhteydessä tarkoittaa vain ns. ”digitaalista työvoimaa”.

2.1.1 Selitys RPA:n toiminnasta

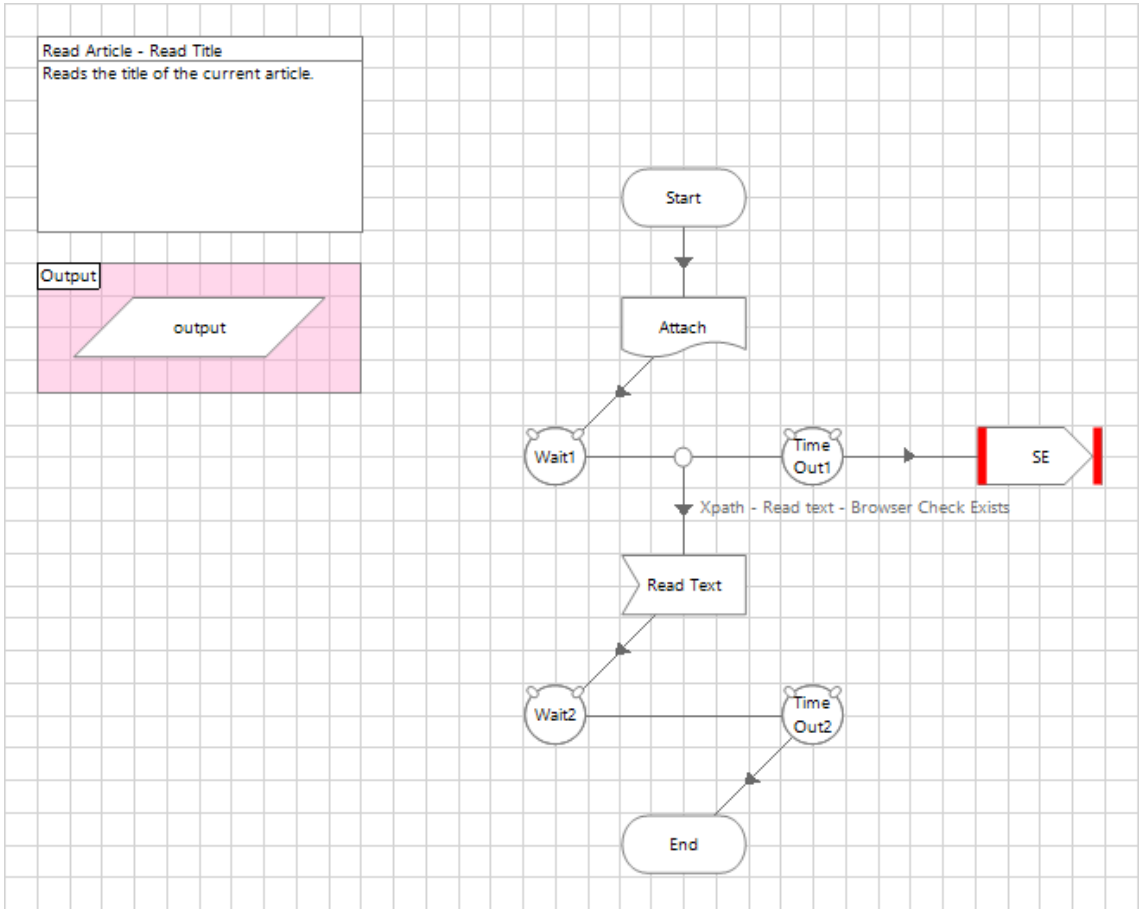
Ohjelmistorobotiikka-automaatiot koostuvat prosessi- ja objektitasoista. Objekteissa rakennetaan yksittäisiä toimintoja, joita hyödynnetään prosessitasolla halutun lopputuloksen saavuttamiseksi. Esimerkki objektitason toiminnosta voisi olla jonkin elementin, kuten napin, klikkaaminen. Syyt prosessi- ja objektitasoon jakoon ovat yksinkertaistaminen ja uudelleenkäytön mahdollistaminen. Tässä opinnäytetyössä kaikki esitellyt esimerkit ovat Blue Prism -ohjelmistosta.

Kaaviossa näkyy esimerkki prosessitason osasta (ks. liite 1. Esimerkki prosessitason osasta Blue Prism -ohjelmistossa). Kyseisessä prosessissa robotti lukee uutissivun artikkeleista niiden otsikon sekä lyhyen kuvauksen. Jos kuvaus on liian pitkä ja sitä ei pysty lukemaan kokonaan avaamatta artikkelia, robotti avaa artikkelin, lukee koko kuvauksen ja palaa etusivulle siirtyen lukemaan seuraavaa artikkelia. Tämä on myös mainio esimerkki, minkälaisia päätöksiä ohjelmistorobotti pystyy itsenäisesti tekemään ilman ihmisen osallistumista. Kuva 1 auttaa tulkitsemaan liitteen 1 diagrammissa näkyvien kuvakkeiden tarkoitusta.



KUVA 1: Blue Prism -ohjelmiston studion yleisimpien vaiheiden muodot ja niiden merkitykset.

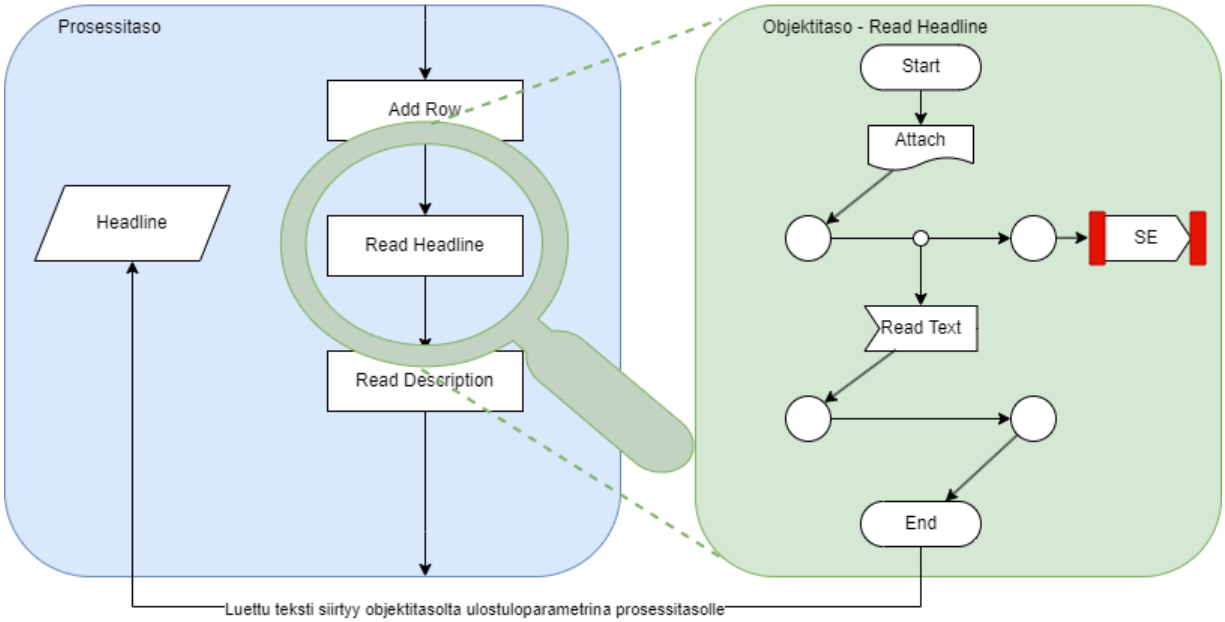
Kuvan 2 objektitason toiminto saa applikaation mallintajalta attribuutit haluttuun elementtiin, joka kertoo Blue Prism:in lukuvaiheelle, mitä elementtiä tämän tulisi lukea. Tämä toiminto kutsutaan prosessitasolla *Read Headline* -vaiheen kohdalla osasta (ks. liite 1. Esimerkki prosessitason osasta Blue Prism -ohjelmistossa).



KUVA 2: Toiminto objektitasolla.

Kun toimintoa kutsutaan prosessitasolla, aletaan sitä suorittamaan *Start*-vaiheesta. Toiminto voi tässä vaiheessa saada myös sisääntuloparametreja prosessilta, jotka siirrettäisiin *Start*-vaiheen kohdalla objektitasolle. *Attach*-vaihe kiinnittää objektitason automatisoitavaan sovellukseen, esimerkiksi selaimeen. Jos samasta ohjelmasta on useampi ilmentymä samaan aikaan auki, kiinnitysvaiheella voidaan myös määritellä mihin näistä ohjelmistorobotti yhdistetään. *Wait*- eli odotusvaihe voi olla joko staattinen tai ehdollinen; staattinen odotusvaihe odottaa aina sille määritellyn ajan ennen seuraavaan vaiheeseen siirtymistä, kun taas ehdollinen odotusvaihe odottaa sille määritellyn elementin löytymistä ennen jatkamista. Ehdollista odotusvaihetta hyödyntäen voidaan siis aina varmistua, että esimerkiksi haluttu selainsivu on latautunut ennen kuin automaatio jatkaa eteenpäin. Jos haluttua elementtiä ei löydy annetun aikaikkunan sisällä, toiminto menee kuvan 2 SE-vaiheeseen (System Exception, suom. järjestelmäpoikkeus). *Read Text*-vaihe lukee sille määritellyn elementin sisältävän tekstin ja palauttaa sen kuvan 2 mukaisesti *output*-nimiseen datamuuttujaan, joka siirretään *End*-vaiheessa ulostulona prosessitasolle.

Alla on eräästä liitteen 1 prosessi- ja objektitason kohdasta kuvio, joka kuvaa näiden kahden tason vuorovaikutusta keskenään. Kaikki päätöksenteko ja logiikka pyritään rakentamaan prosessitasolle, kun taas objektitasolla vaikutetaan kohdejärjestelmiin usein yksinkertaisilla toiminnoilla.



KUVIO 1: Havainnollistava kuva prosessi- ja objektitason dynamiikasta. Luettu otsikko palautuu prosessitasolle *Headline*-nimiseen datansäilytysmuuttujaan.

2.2 Milloin ja miksi RPA on kehitetty?

Ensimmäiset ohjelmistorobotiikkatyökalut julkaistiin 2000-luvun alussa tarvittavien teknologioiden kehityttyä 90-luvun aikana. Yksi tärkeimpiä RPA:n mahdollistavia tämän ajan uusia teknologioita oli ns. tiedonharavointi (engl. Data scraping) Kyseinen teknologia mahdollisti kaiken ihmiselle luettavissa olevan tiedon lukemisen myös ohjelmistoille. Näin ollen saatiin rakennettua jo hyvin alkukantaisia ohjelmistorobotteja, jotka suorittivat mitä yksinkertaisimpia tehtäviä, kuten kopiointia ja liittämistä. (History of Robotic Process Automation (RPA), 2021. Robomotion).

Ohjelmistorobotiikka ei saanut yritysmaailmassa kunnolla tuulta alleen ennen vuotta 2015. Tämän jälkeen RPA:n suosio on ollut huimassa nousussa – Gartnerin arvion mukaan vuonna 2018 maailmanlaajuinen kulutus RPA-työkaluihin arvioitiin olevan 680 miljoonaa dollaria, mikä oli 57 % kasvu vuoteen 2017 verrattuna. Vuonna 2018 arvioitiin saman luvun olevan 2,4 miljardia dollaria vuonna 2022 (Shetty, 2018). Syksyllä 2022 Gartnerin mukaan kyseinen luku on kuitenkin 2,9 miljardin dollarin vauhdissa, joka olisi vieläkin 19,5 % nousu edelliseen vuoteen verrattuna (Stamford, 2022).

RPA on kehitetty yksinkertaisten ja suurissa määrissä toistuvien prosessien automatisointiin. Hyvin suuri osa digitaalisista töistä sisältää huomattavan määrän toistuvaa työtä, kuten lomakkeiden täyttämistä, Excel-tiedostojen manipulointia ja eri järjestelmissä yksinkertaista työskentelyä. Kyselyjen mukaan yli 40 % kaikista työntekijöistä uskoo, että he voisivat säästää vähintään neljänneksen viikon työmäärästä, jos heidän työnsä toistuvat tehtävät automatisoitaisiin (Beloof, Smartsheet).

2.2.1 RPA:n hyödyt

Ohjelmistorobotiikan suurin hyöty on ihmisresurssien vapauttaminen paljon aikaa vievistä tehtävistä, jotka ovat helposti automatisoitavissa. Kun tällaiset prosessit automatisoidaan, saadaan sitä aikaisemmin suorittanut ihminen tekemään työtä, joka on parempaa vastinetta ihmistyöntekijän ajalle. Yleisesti ottaen tällainen työ on myös mielekkäämpää kuin saman toistuvan tehtävän tekeminen päivästä toiseen. (Savaram, mindmajix)

Resurssien vapauttamisen lisäksi RPA:n hyötyihin kuuluu hyvin usein myös työn nopeutuminen. Voi olla hyvinkin yksinkertaisia kopioi-ja-liitä-prosesseja, joita tekee päivittäin monta henkilöä, jotta kaikki saataisiin tehtyä ajallaan. Robotin luomat säästöt mitataan usein Full Time Equivalent -säästöinä, eli kuinka montaa täysipäiväistä ihmistyöntekijää yksi robotti vastaisi. Oletetaan, että ihminen tekisi yhden tehtävän minuutissa ja ohjelmistorobotti tekisi saman tehtävän 20 sekunnissa. Tällöin robotin FTE-säästö olisi 3, eli robotti tekisi käytännössä kolmen henkilön työmäärän olettaen molempien työskentelevän samat työtunnit. Vaikka FTE olisikin hieman alle yksi ja robotti toimisi hitaammin kuin ihminen, RPA:n muut hyödyt voisivat silti tehdä automatisoinnista kannattavan. Resurssien säästämisen lisäksi RPA auttaa myös työn varmuudessa – täydellisesti kehitetty robotti ei tee koskaan virheitä, lukuun ottamatta tietenkin kohdejärjestelmistä tai muista ulkoisista tekijöistä johtuvia.

Usein on myös mahdollista ajaa prosessia useammalla virtuaalikoneella samaan aikaan huomattavasti pienemmällä kustannuksella kuin palkkaamalla uusi ihmistyöntekijä. RPA-prosessit voivat myös olosuhteiden salliessa pyöriä vuorokauden ympäri, jolloin ne ovat myös huomattavasti tehokkaampia kuin ihmiset. Edellä mainitut hyödyt ovat kuitenkin hyvin prosessikohtaisia ja vaihtelevat suuresti prosessin luonteen ja kohdejärjestelmien mukaan. Myös automatisointiin käyteyllä RPA-ohjelmistolla voi olla näihin merkitystä.

RPA-ohjelmistoissa on myös mahdollisuus aikatauluttaa prosesseja. Tämä mahdollistaa useiden eri automaatioprosessien ajamisen yhdellä tai useammalla resurssilla ilman ihmisten syötettä prosessien välissä. Näin ollen useamman prosessin ajaminen yhdelläkin resurssilla on mahdollista, kunhan aikataulutus on

tehty niin, että jokainen prosessi ehtii työskennellä tarvitun ajan ja ne ovat lomitettu sopivasti.

Yksi suurimmista tekijöistä, mikä tekee ohjelmistorobotiikasta varsinkin yrityksille houkuttelevan ja sopivan vaihtoehdon automatisointiin on sen skaalautuvuus. Skaalautuvuudella tarkoitetaan automaattiprosessien ajavien tietokoneiden resurssien lisäämistä tai vähentämistä työmäärän mukaan. Voisi olla esimerkiksi ohjelmistorobotti, joka käsittelee asiakkaiden tilauksia nettikaupasta, jossa tilausten määrä vaihtelee huomattavasti vuodenajan mukaan. Tällöin voitaisiin ruuhka-aikana ottaa useampi resurssi, eli virtuaalikone tai ”digitaalinen työntekijä” käyttöön prosessien ajamiseen, jotta tapaukset saataisiin käsiteltyä halutun ajan sisällä. Ruuhka-ajan jälkeen ylimääräiset resurssit voidaan poistaa käytöstä, jolloin lisenssimaksut ja resurssien käyttö voidaan minimoida. Skaalautuvuus ja sen hyödyt vaihtelevat kuitenkin suuresti käytetyn RPA-ohjelmiston tarjoajan mukaan.

3 RPA:N KEHITYSTAVAT

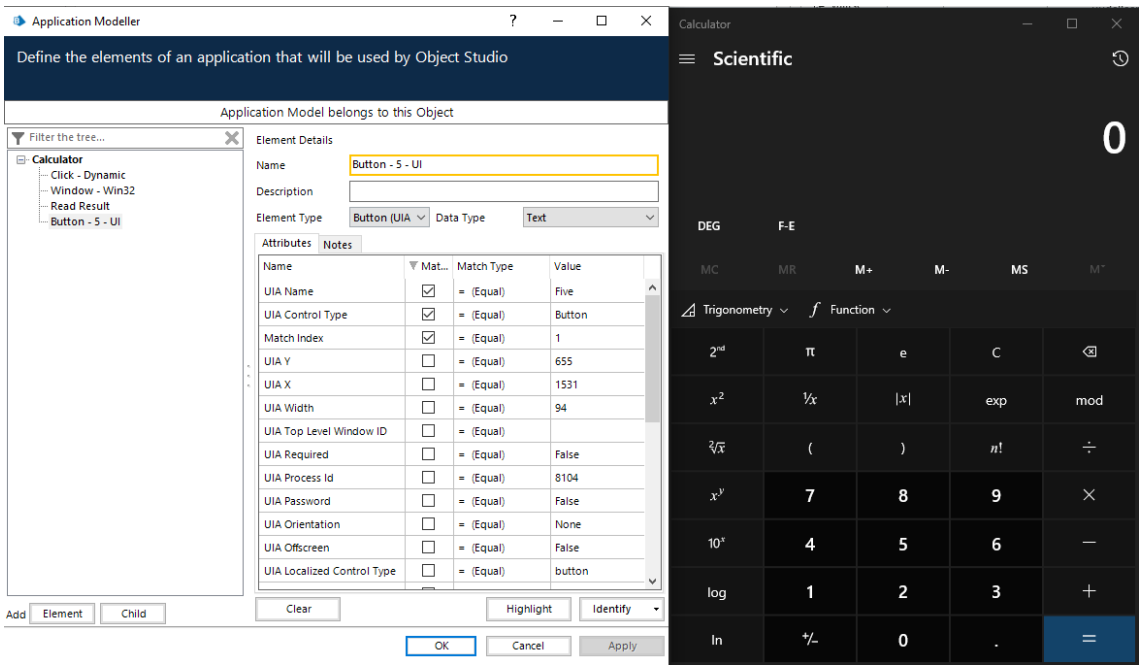
RPA on ajan myötä kehittynyt myös ominaisuuksiltaan. Alun perin ohjelmistorobotiikka kehitettiin vain suoraan käyttöliittymän kautta automatisointiin, mutta nykypäivänä se on kehitetty hyödyntämään muitakin ohjelmistojen osa-alueita. Tässä kappaleessa käydään läpi eri lähestymistapoja prosessien automatisointiin sekä muita RPA:n yleisiä ominaisuuksia.

3.1 UI-automaatio

UI-automaatio eli käyttöliittymän kautta automatisointi on edelleen ohjelmistorobotiikan maailmassa hyvin yleinen tapa automatisoida prosesseja. Tämä pohjautuu sen monikäyttöisyyteen ja helppouteen kehittäjän, niin kuin asiakkaankin näkökulmasta.

Kaikki sovellukset, jossa käyttäjä operoi ohjelmaa käyttöliittymän kautta, koostuu elementeistä. Joka ikinen nappi, teksti, taulukko tai valintaruutu ovat elementtejä, jotka koostuvat attribuuteista, kuten nimestä, sijainnista ja koosta. Näitä attribuutteja hyväksikäyttäen käytettävä RPA-työkalu tunnistaa elementin, jota halutaan tavalla tai toisella automatisoida. Tämä mahdollistaa esimerkiksi napin painamisen, tekstiruutuun kirjoittamisen tai sen sisällön lukemisen.

Kuvassa 3 Blue Prism ohjelmiston mallintajassa (Application Modeller) nähdään luotu elementti laskimen napille 5 ja listattuja attribuutteja, joilla kyseinen elementti tunnistetaan. Kyseisessä tapauksessa Blue Prism tunnistaa elementin sen nimen (kuvassa UIA Name) ja sen ohjaustyypin (UIA Control Type) perusteella. Vastaavuusindeksin (Match Index) ollessa 1, BP valitsee aina ensimmäisen attribuutteja vastaavan löydetyn elementin. Tämä yleisesti nopeuttaa automatisointia.



KUVA 3: Windows laskimen ’5’-nappi Blue Prism:in ohjelmiston mallintajassa.

Tämän toimintaperiaatteen ansiosta UI-automaatiolla voidaan yleensä automatisoida ohjelman näkökulmasta kaikki, mitä ihminenkin voisi sitä käyttäessään tehdä. Automatisoitavaan ohjelmaankaan ei siis tarvitse tehdä mitään muutoksia, jotta sitä hyödyntävä prosessi voidaan automatisoida käyttöliittymää käyttäen.

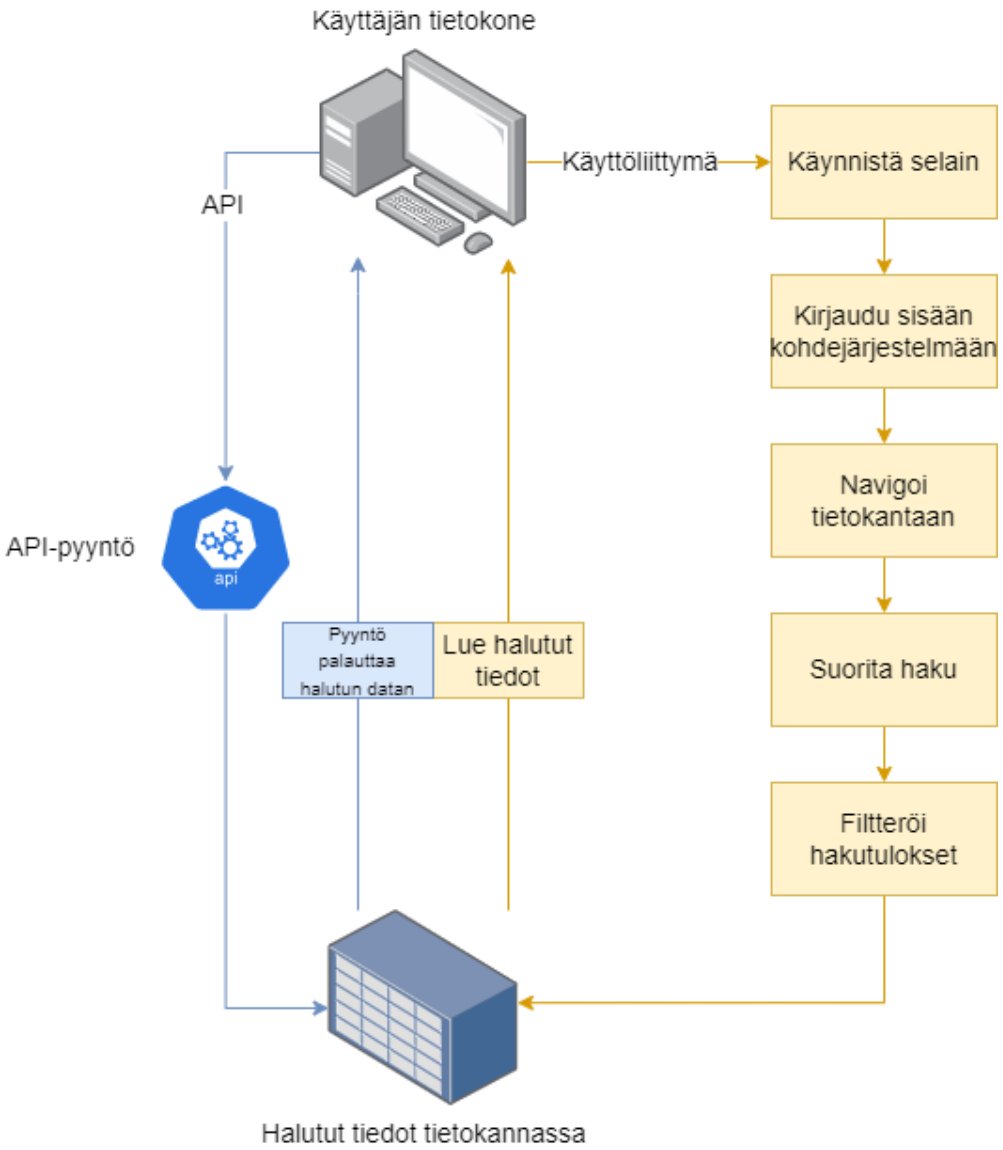
Kyseisessä automaatiotavassa on kuitenkin myös huonoja puolia. Automaation sujuvuus ja varmuus ovat hyvin pitkälti riippuvaisia kohdesovelluksesta tai -järjestelmästä. Varsinkin internet-selaimia automatisoidessa robotin toiminta riippuu suuresti kohdesivun rakenteesta ja tavasta, jolla se on toteutettu. On mahdollista, että monella elementillä on hyvinkin samanlaiset attribuutit, jolloin RPA-ohjelmisto saattaa tunnistaa useamman elementin, vaikka haluttaisiin vain yksi. Tällaisessa tapauksessa automaatio aina keskeytyy, sillä ohjelmisto ei osaa itse päättää, mitä elementtiä oikeasti halutaan. Kyseisiä tapauksia voi havaita usein esimerkiksi pudotusvalikoissa tai erilaisissa taulukoissa. Näihin ongelmiin löytää kuitenkin usein ratkaisun joko kokeilemalla eri elementin attribuutteja tai käyttämällä wildcardia eli ns. villikortti-tunnistustyyppiä. Esimerkiksi, jos attribuutissa olisi päivämäärä seuraavanlaisesti: ”*google_04122022”*, voitaisiin villikorttia hyödyntäen tehdä siitä ”*google_*”*, jolloin päivämäärää ei tarkasteta ollenkaan. Jos ratkaisua ei löydy, joudutaan turvautumaan johonkin toiseen automaatiotapaan.

3.2 API-automaatio

API:n eli ohjelmointirajapinnan (engl. Application Programming Interface) kautta automatisointi on yleisesti ottaen paras tapa toteuttaa automatisointia. Se on suorin mahdollinen tapa ohjelmistorobotille kertoa automatisoitavalle kohdejärjestelmälle, mitä sen tulisi tehdä. Kyseistä automatisointitapaa käytetään kuitenkin harvoin, sillä sen käyttämismahdollisuudet ovat hyvin rajalliset, sekä API-kutsujen kehittäminen vie usein huomattavasti enemmän aikaa, kuin suoraan käyttöliittymän kautta automatisoiminen. Jos kuitenkin halutaan prosessin olevan mahdollisimman vakaa, on ohjelmointirajapinnan käytön tutkiminen kannattavaa.

Ohjelmointirajapinta tarkoittaa ikään kuin väylää, jonka kautta tässä tapauksessa RPA-työkalu ja kohdejärjestelmä kommunikoi suoraan keskenään, eli käyttöliittymää ei tarvitse ollenkaan näiden kahden ohjelman välille. Näin ollen voidaan prosessien suorittamisesta saada huomattavasti nopeampaa ja sujuvampaa.

Rajapintaa käytettäessä myös säästetään huomattavasti aikaa sekä tietokoneen resursseja, kun voidaan unohtaa kaikkien sivujen, välilehtien ja grafiikoiden lataamisajat. Tällöin resursseja käytetään vain ja ainoastaan prosessin oleelliseen suorittamiseen. Kuvion 2 esimerkissä olevassa API-pyynnössä kestäisi maksimissaan muutama sekunti, kun taas käyttöliittymän kautta tehdyssä haussa voisi kestää jopa minuutti tai mahdollisen hitaan käyttöliittymän takia jopa kauemmin.



KUVIO 2: Tarvittavien vaiheiden vertailu haettaessa tietoja selainpohjaisesta tietokannasta käyttöliittymää ja ohjelmointirajapintaa käyttäen.

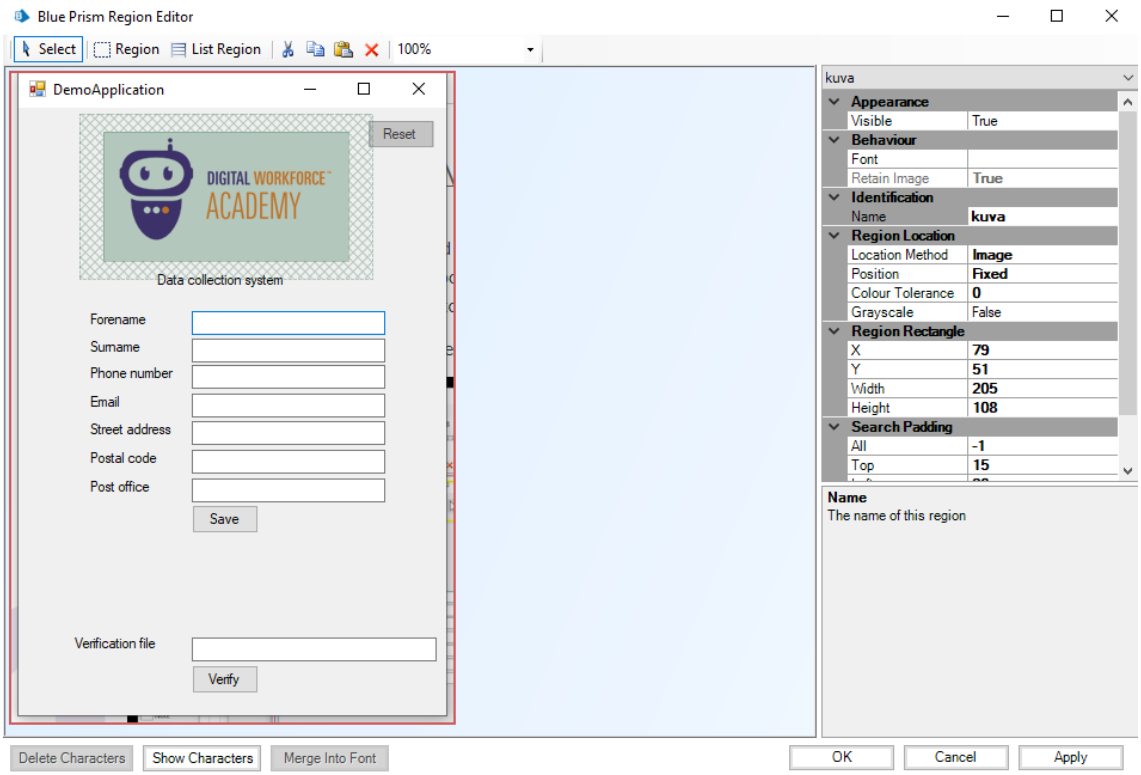
Ohjelmointirajapinnan kautta automatisointi ei ole kuitenkaan yhtä yksiselitteistä kuin käyttöliittymän kautta – jotta automatisointi API:a hyödyntäen on mahdollista, täytyy kohdejärjestelmässä olla valmiiksi käytettävä rajapinta saatavilla. Tämä tarkoittaa sitä, että RPA-työkalusta voidaan lähettää järjestelmälle API-pyyntö, jolloin kohdejärjestelmä tietää mitä sen tulee pyynnön lähettäjälle palauttaa.

3.3 Pinta-automaatio

Pinta-automaatio on nimensä mukaan hyvin pinnallista ja sen hyödyntämistä pyritään usein välttämään. Pinta-automaatio perustuu täysin tietokoneen ruudun eli resoluution koordinaatteihin tai kuvantunnistamiseen. On helppo ymmärtää, että on paljon asioita, jotka voivat mennä pieleen kyseisiä menetelmiä käyttäen.

Pinta-automaatiota käytettäessä RPA-työkalu ottaa usein kuvankaappauksen ruudusta, josta automatisoija voi valita alueen ja määrittää tälle jonkin toiminnon, kuten siihen hiirellä klikkaamisen. Tämä voi olla hyvin hyödyllistä, jos automatisoitavaa järjestelmää ei voida ohjata ohjelmointirajapinnan tai elementtien kautta.

Automaatiossa RPA-ohjelmisto tunnistaa valitun alueen usein resoluution koordinaateilla, jossa vasen ylänurkka on (0, 0) ja FHD (full HD 1080p) -näytöllä oikea alanurkka olisi (1920, 1080). Ongelmia siis ilmenisi heti, jos prosessia ajettaisiin jollain muulla resoluutiolla, kun millä se on alun perin konfiguroitu toimimaan, sillä valittu alue ei olisi tällöin halutussa paikassa. Kuvassa 4 nähdään esimerkki, miten kuvantunnistus toimii. RPA-ohjelmisto etsii vihreällä maalatulta alueelta vastaavaa kuvaa, mikä kuvankaappauksessakin alueella näkyy.



KUVA 4: Esimerkki kuvantunnistuksesta pinta-automaatiota hyödyntäen.

Ongelmia pinta-automaation käytössä ilmenee helposti myös ajoituksessa. Koska ohjelmistorobotti ei pinta-automaatiota käyttäessä käytännössä kommunikoi käytettävän järjestelmän kanssa lainkaan, vaan raa'asti painaa tiettyihin koordinaatteihin sen saatua käskyn, ei se voi ikinä olla varma, että järjestelmä on valmis seuraavalle komennolle. Esimerkiksi, jonkin internet-sivun latausajat voivat vaihdella hyvinkin paljon sen hetkisen verkkoliikenteen mukaan ja pinta-automaatiolla tehty robotti ei osaa ottaa näitä vaihtelevia latausaikoja huomioon. Näin ollen robotti voi koittaa painaa jotain nappia sivun vielä ladattaessa.

Pinta-automaatiollakin on kuitenkin tapoja ”odottaa” kohdejärjestelmiä. On mahdollista hyödyntää kuvantunnistusta prosessin edistymisen seuraamiseen, eli esimerkiksi tarkastaa jonkin internetsivun logo, jolloin voitaisiin varmistua siitä, että sivun on ladannut. Tämä kuitenkin olisi altis myös resoluutiomuutoksille, kuten myös kohdejärjestelmään tehtäville pienillekin muutoksille, jotka muuttavat sen ulkoasua. Jos käytetään internetselainta, myös tämän asetukset voivat vaikuttaa pinta-automaatioon; työkalupalkit, kirjanmerkit ja selaimen sisäinen suurennus tai pienennys voivat kaikki vaikuttaa kriittisesti pinta-automaation toimintaan.

On järjestelmiä, joiden automatisointiin ainoa vaihtoehto on nimenomaan pinta-automaatio. Nämä prosessit ovat lähes poikkeuksetta tehottomia ja epävakaita juuri edellä mainituiden syiden takia. Useimmiten pinta-automaatiota hyödyntävissä prosesseissa joudutaan käyttämään liioiteltuja staattisia odotuksia eri toimintojen välillä, eli odottamaan esimerkiksi 20 sekuntia jokaisen klikkauksen välissä, joka tekee prosesseista myös hitaampia. Kokenut kehittäjä voi kuitenkin tehdä pinta-automaatiollakin toimivan prosessin. Tämä kuitenkin riippuu erittäin paljon kohdejärjestelmästä, jota automatisoidaan. Pinta-automaatiota voidaan kuitenkin käyttää myös tehokkaasti käyttöliittymän ja ohjelmointirajapinnan rinnalla.

4 RPA-PROSESSIN TOTEUTTAMINEN

Ohjelmistorobotiikkaprosessin toteuttamiseen kuuluu prosessin monimutkaisuudesta tai koosta huolimatta monta eri vaihetta. Yleisesti ottaen kaikki RPA:n tarjoajat noudattavat samantyylistä toimitussuunnitelmaa, vaikkakin yksityiskohdat vaiheissa ja dokumenteissa ovat tietenkin erilaisia tekijästä riippuen.

4.1 Prosessin toteuttamisen vaiheet

Toteutus alkaa automatisoitavan manuaalisen prosessin tunnistamisesta ja päättyy prosessin luovuttamiseen ns. tuotantoon, jossa prosessi toteuttaa sille määritellyt tehtävät varsinaisessa ympäristössä. Tässä kappaleessa esitellään vaiheet, mitkä usein tähän prosessiin sisältyvät.

4.1.1 1. Vaihe: Prosessin tunnistus

Prosessin automatisoinnin ensimmäinen vaihe on luonnollisesti sopivan manuaalisen prosessin löytäminen, jota on kannattavaa automatisoida. Yleensä sopivat prosessit ovat hyvinkin helppoja tunnistaa, kun tietää RPA:n vahvuudet, mahdollisuudet ja rajallisuudet. Tähän on olemassa myös tiettyjä työkaluja, joilla voidaan määritellä, onko prosessi kannattavaa automatisoida liiketoiminnan kannalta. Tekijöitä päätöksen tekemisessä on esimerkiksi FTE, mahdollinen automatisointiaste, prosessin yksinkertaisuus, sijoitetun pääoman tuottoaste jne. Jotkut prosessit voivat olla

Tässä vaiheessa voidaan yleensä jo päättää, millä RPA-ohjelmistolla prosessi kannattaa toteuttaa, jos kehityksen tarjoajalla on valinnanvaraa. Eri RPA-ohjelmistot soveltuvat joskus tietynlaisiin prosesseihin paremmin kuin toiset – joskus sillä on taas puolestaan hyvin pieni merkitys, jolloin lisenssimaksut ovat usein määrittelevä tekijä.

4.1.2 2. Vaihe: Manuaalisen prosessin läpikäynti

Prosessin varsinaisen automatisoinnin ensimmäinen vaihe on varmistaa RPA-prosessin kehittäjän täysi ymmärrys automatisoitavasta prosessista. Tämä on hyvin tärkeää tehdä heti aluksi, jotta kehittäjä voi aloittaa kokonaisvaltaisen kehityksen halutulla ohjelmistorobotiikkaohjelmistolla.

Tämä voidaan toteuttaa esimerkiksi visuaalisella läpikäynnillä, eli manuaalisen prosessin hyvin tunteva henkilö (SME, Subject Matter Expert) voi näyttää prosessin kehittäjälle tai muulle analyytikolle alusta loppuun kaikki yksityiskohdat mukaan lukien. On tärkeää, että asiat, mitkä voivat mennä pieleen käydään myös tarkasti läpi, jotta nämä voidaan huomioida kehityksessä ja automaatiosta saadaan mahdollisimman luotettava ja itsenäinen.

4.1.3 4. Vaihe: Prosessin kehittäminen ja testaaminen

Automaation toteuttamisen pisin vaihe on lähes poikkeuksetta itse prosessin kehittäminen. Tämä tarkoittaa automatisoinnin rakentamista jollain ohjelmistorobotiikkatyökalulla, kuten UiPath:lla, Blue Prism:lla tai Automation Anywhere:lla. Prosessin kehittämiseen kuluva aika riippuu hyvin paljon prosessin monimutkaisuudesta, kehittäjän kokemuksesta sekä tämän osaamisen tasosta. Useimmiten tähän vaiheeseen kuluu kuitenkin vähintään viikko täysipäiväisellä työpanoksella, mutta hyvin monimutkaisiin prosesseihin voi kulua jopa kuukausia.

Kehitysvaiheessa kehittäjä tarvitsee pääsyn mahdollisiin järjestelmiin, jossa ohjelmistorobotti tulee työskentelemään. Ilman pääsyä, kehittäjä ei voi tunnistaa tarvittavia elementtejä ja suorittaa testausta. Useimmiten järjestelmistä on olemassa ns. testiympäristö, jossa voidaan testata robotin toimintaa huolettomasti.

4.1.4 5. Vaihe: Laadunvarmistus ja UAT

Prosessin kehityksen jälkeen useimmiten varmistetaan sen laatu. Laadunvarmistuksen tarkoituksena on pitää heikosti kehitetyt ja epävakaat prosessit poissa tuotannosta, jossa niistä voisi olla enemmän haittaa kuin hyötyä. Laadunvarmistuksen kanssa samoihin aikoihin pidetään myös usein UAT (engl. User Acceptance Testing) eli hyväksymistestaus.

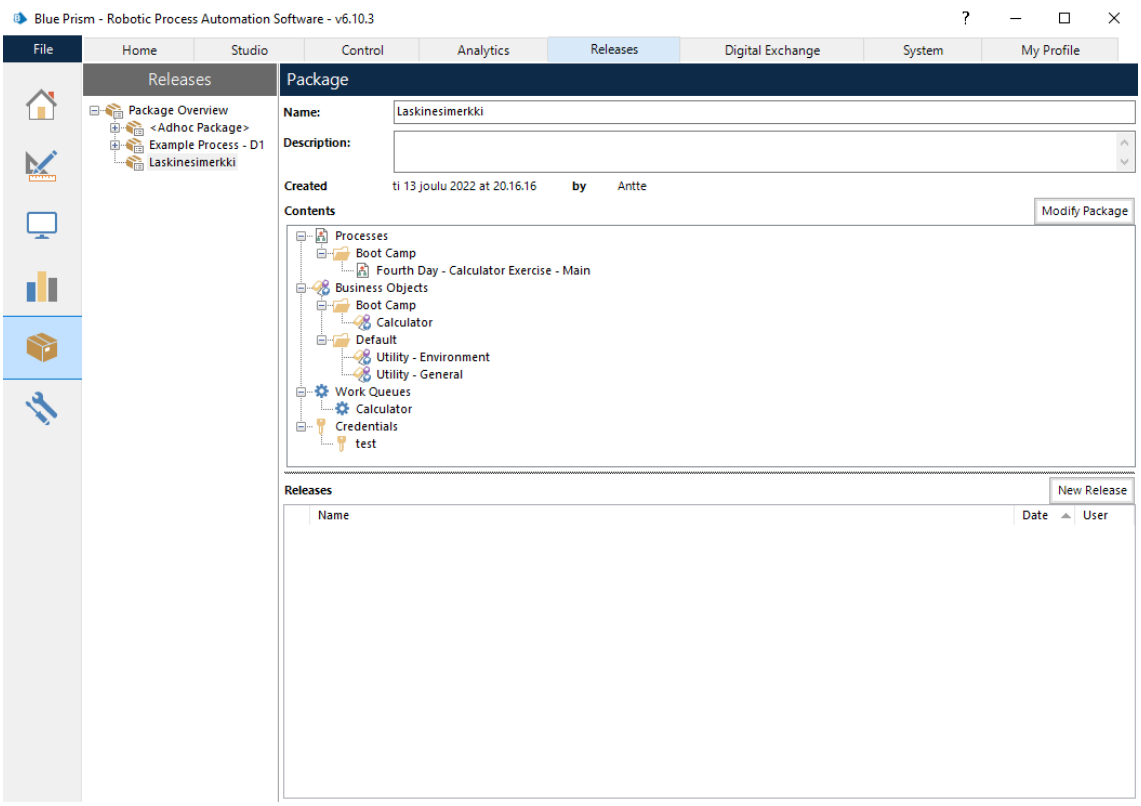
Laadunvarmistus voidaan suorittaa monella eri tapaa; useampi kokenut kehittäjä voi käydä prosessin ratkaisun läpi ja tuomita, täyttääkö se kaikki parhaat käytännöt tai yksi laadunvarmistukseen erikoistunut henkilö voi tehdä sen yksin. Tarkistettavia asioita voi olla esimerkiksi, miten robotti toimii, jos kohdejärjestelmässä tulee jokin odottamaton virhe tai mahdollinen kohdenettisivu ei vastaa. Ideana on siis saada robotista mahdollisimman vankka, ennen kuin sitä aletaan käytännössä hyödyntämään.

Kun prosessin laatu on varmistettu, pidetään hyväksymistestaus. Tämä pidetään prosessin ekspertin kanssa, eli henkilön, joka on kyseistä automatisoitavaa prosessia suorittanut aikaisemmin manuaalisesti tai sen muuten perin pohjin tuntevan henkilön kanssa. Tämä vaihe on hyvin tärkeä, sillä vasta tässä vaiheessa voidaan täysin varmistua siitä, että robotti tekee täysin oikeat asiat. Hyväksymistestauksessa kehittäjä ajaa robottia ns. debug -tilassa, eli askel kerrallaan, jolloin prosessin ekspertin on helppo seurata, mitä robotti tekee.

4.1.5 6. Vaihe: Prosessin tuotantoon vienti

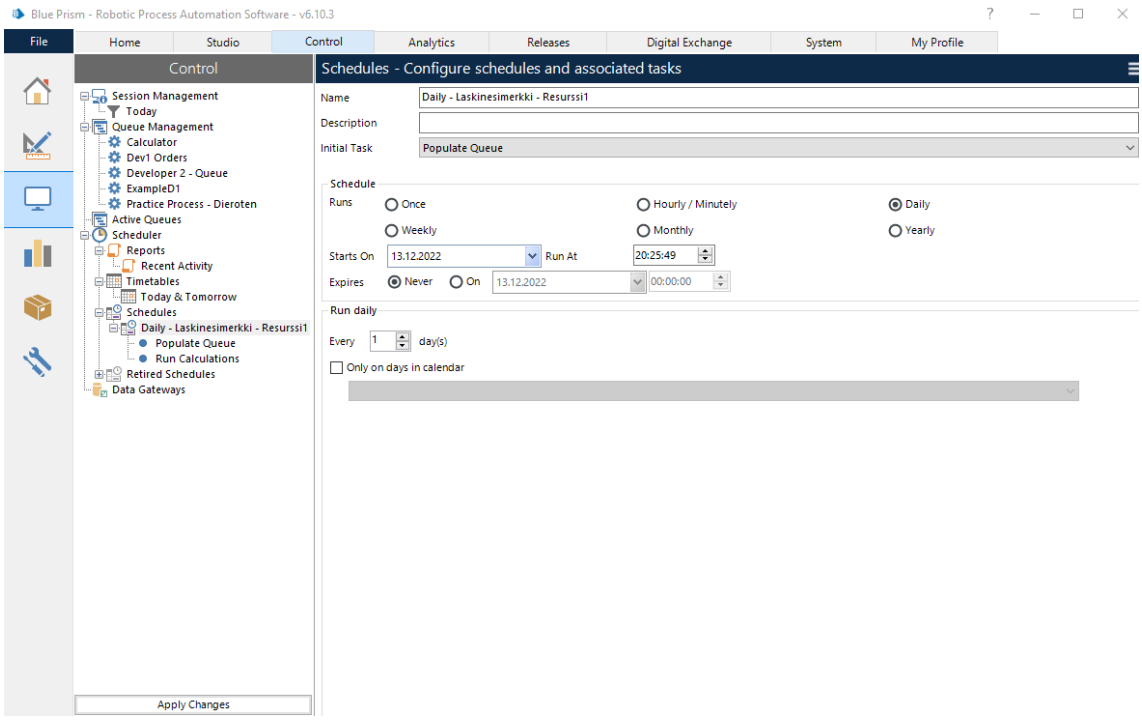
Prosessin toteuttamisen viimeinen vaihe on itse prosessin vieminen tuotantoon, eli prosessi laitetaan ajamaan ympäristössä, jossa se pystyy suorittamaan sille määritellyt tehtävät. Vasta kun robotti ajaa tuotannossa, on siitä liiketoiminnalle varsinaista hyötyä. Tuotantoon vientiin kuuluu käytetyn RPA-ohjelmiston mukaan erilaisia vaiheita. Esimerkin vuoksi käyn vaiheet läpi siten, miten se tehtäisiin Blue Prism -ohjelmistossa ilman kolmannen osapuolen sovelluksia.

Blue Prism:ssä on kehitys- ja tuotantopuoli erikseen. Kehitys tapahtuu luonnollisesti kehityspuolella ja vasta tuotantoon viennin yhteydessä viedään prosessi, objektit, työjonot, tunnukset ja muut muuttujat, joita prosessi tarvitsee ajaakseen tuotantopuolelle. Tämä onnistuu Blue Prism:ssä julkaisemistyökalulla, jolla saa kaikki prosessin tarvitsevat riippuvuudet yhteen tiedostoon, joka on helppo tuoda Blue Prism:n tuotantopuolelle. Kuvassa 5 näkyy laskinprosessin itse prosessi ja kaikki sen tarvittavat riippuvuudet.



KUVA 5: Prosessipaketin tekeminen Blue Prism:ssä.

Kun itse prosessi on tuotannossa, voidaan tälle luoda aikataulu, jossa määritellään, milloin prosessi ajaa milläkin virtuaalikoneella. Kun prosessi on aikataulutettu, sitä ei tarvitse manuaalisesti aloittaa aina kun robotin halutaan työskentelevän. Kuvassa 6 nähdään, miten aikatauluun voi määritellä kuinka usein ja minä kellonaikana robotti aloittaa ajamisen sekä missä järjestyksessä prosessit ajetaan, jos prosessi on jaettu useampaan osaan. Kalenterominaisuutta hyödyntäen voidaan myös määritellä tarkemmin, minä päivinä robotin halutaan tai ei haluta työskentelevän – näin voitaisiin esimerkiksi sulkea pois pyhäpäivät ja viikonloput robotin ajoaikataulusta.



KUVA 6: Aikataulun määritys Blue Prism:ssä.

Viimeisenä täytyy ohjelmistorobotin ympäristömuuttujat muuttaa siten, että ne vastaavat varsinaisen tuotantoympäristön arvoja. Näihin ympäristömuuttujiin voikuulua esimerkiksi URL-osoite, joka on eri testi- ja tuotantosivuille. Jos automatisoitava järjestelmä tarvitsee sisäänkirjautumista, myös tunnukset ovat usein eri testi- ja kehitysympäristöissä.

4.2 Dokumentointi

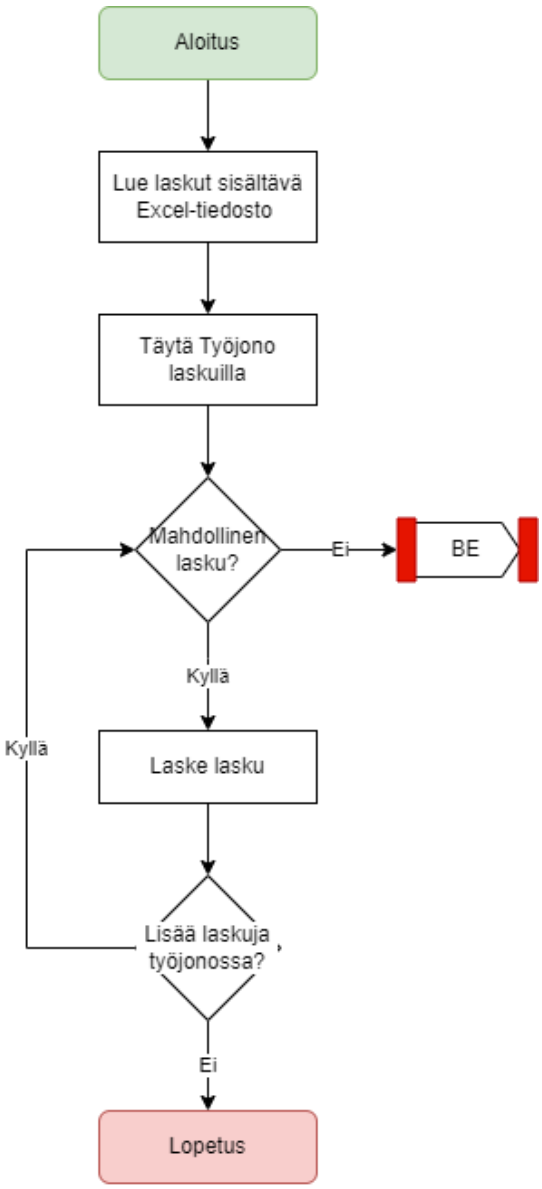
RPA-prosessin toteuttamisen hyvin oleellinen osa on asianmukainen dokumentointi. Dokumentointi monessa tapauksessa varmistaa, että kaikki osapuolet ovat samalla aaltopituudella, mitä ollaan automatisoimassa ja miten. Tarkat dokumentit myös mahdollistavat jatkossakin ohjelmistorobotiikkaprosessiin muutosten tekemisen, vaikka alkuperäinen kehittäjä ei olisikaan enää saatavilla. Kaikista yleisin ja kokonaisuutena hyödyllisin dokumentti on prosessin määrittelydokumentti.

4.2.1 PDD

PDD (Process Definition Document) eli prosessin määrittelydokumentti on ensimmäinen ja joskus ainoa dokumentti, joka tehdään prosessin automatisoinnin yhteydessä. PDD:n tarkoitus on olla pääasiallinen ohje kehittäjälle, joka prosessin automatisointiratkaisun tekee sekä toimia ikään kuin varmistuksena, että määrittely on tehty oikein, sillä tämän yleensä myös lukee ja hyväksyy manuaalisen prosessin ekspertti. Dokumentti siis kuvaa hyvin yksityiskohtaisesti, millainen prosessin tulisi olla.

Määrittelydokumentissa on usein useita osioita, mutta tärkeimpänä on yksityiskohtainen työohje. Tämä osio kuvaa automatisoitavan prosessin hyvin yksityiskohtaisesti askel askeleelta – optimaalisessa tilanteessa jokainen klikkaus, napin painallus tai muu toiminto, minkä robotti joutuu prosessin aikana tekemään, olisi kuvattu työohjeessa. Näin kehittäjälle ei jäisi yhtään tulkinnan varaa, mikä minimoisi liiketoiminnalliset virheet kehitysvaiheessa.

Kuviossa 3 nähdään korkean tason kuvaus laskinprosessista, joka esitellään tarkemmin Eetu Lahtisen osuudessa (Ohjelmistorobotiikan hyödyt ja vaatimukset. Lahtinen, E. 2022, 20–33). Samantyylisen vuokaavion löytää yleensä jokaisesta määrittelydokumentista yksityiskohtaisen työohjeen lisäksi. Tämänlainen diagrammi antaa lukijalle jo nopealla perehtymisellä hyvän kuvan prosessin pääasiallisesta ideasta ja toimintaperiaatteesta.



KUVIO 3: Korkean tason kuvaus laskinprosessista.

Vuokaaviossa käydään ylätasolla kaikki prosessin vaiheet läpi. Tässä tapauksessa vaiheita on hyvin vähän, sillä esimerkkiprosessi on todella yksinkertainen ja lyhyt. Kaavioon on kuitenkin merkitty oleelliset vaiheet, kuten työjonon täyttäminen ja laskujen laskeminen. Kaavioon on myös laitettu tärkeät päätöskohdat, kuten onko lasku mahdollinen laskea. Jos ei, niin prosessi merkitsee kyseisen työjonon tapauksen liiketoimintapoikkeukseksi (engl. Business Exception). Prosessi myös tarkistaa jokaisen laskun jälkeen, onko työjonossa lisää laskuja. Tämä myös on merkitty diagrammiin päätöskohtana.

Taulukossa 1 esitellään esimerkki yhdestä yksityiskohtaisen työohjeen vaiheesta. Määrittelydokumentissa on tarkoituksena esitellä koko prosessin kulku vastaavanlaisilla hyvin yksityiskohtaisilla taulukoilla, jotta kehittäjälle jäisi mahdollisimman vähän tulkinnanvaraa.

Esimerkkilaskuna 1 + 9

1. Klikkaa **1**
2. Klikkaa **+**
3. Klikkaa **9**
4. Klikkaa **Yhtä suuri kuin** (**=**) näppäintä
5. Lue tulos tuloskentältä
6. Tyhjennä laskin klikkaamalla **CE** nappia

Kommentti: On mahdollista myös lähettää toiminnot laskimelle käyttäen näppäimistön näppäimiä, jos klikkaus tuottaa ongelmia.

HUOM: Jos laskutoimitus on mahdoton (0:lla jakaminen), merkkaa tapaus poikkeukseksi ja siirry seuraavaan laskuun.

TAULUKKO 1: Esimerkki yksityiskohtaisen työohjeen vaiheesta.

5 POHDINTA

Olen työskennellyt täysipäiväisesti ohjelmistorobotiikan parissa opinnäytetyön valmistumishetkellä noin 10 kuukautta. Tähän työskentelyyn on kuulunut paljon tämän opinnäytetyön sisällössä olevia asioita, joten suuri osa kirjoitetusta tiedosta on omien kokemuksieni ja oppimiseni tulosta. RPA:n tarjoajia on kuitenkin hyvin monenlaisia ja olen pyrkinyt kirjoittamaan aiheista todella yleisesti, enkä vain, miten olen itse asioita työssäni tehnyt. Opinnäytetyössäni tarjotut tiedot ovat sovellettavissa yleisesti ottaen kaikkien RPA:n tarjoajien toimintatapoihin.

Ohjelmistorobotiikka aiheena oli hyvin mielenkiintoinen ja siitä kirjoittaminen tuntui luonnolliselta työskennellessäni näiden aiheiden parissa hyvin aktiivisesti. Minun ja Eetun osuudet yhdessä luovat hyvin kokonaisvaltaisen ylätason kuvan ohjelmistorobotiikasta ja sen mahdollisuuksista tahoille, jotka aiheesta ovat kiinnostuneita.

Työtä kirjoittaessa opin ohjelmistorobotiikasta itsekin lisää, vaikka suuri osa aiheista oli jo tuttuja. Näiden asioiden miettiminen ja kirjoittaminen kuitenkin paransi omaa ymmärrystäni aiheesta ja auttoi yhdistämään asioita toisiinsa. Myös RPA:n historiasta ja sen suosion räjähtävästä kasvusta oppiminen on itselleni hyvin tärkeää ja hyödyllistä.

Opinnäytetyössä olisin voinut kertoa vielä enemmän mitä prosesseille tapahtuu kehitysvaiheen jälkeen, kun ne ovat siirtyneet tuotantoon. Esimerkiksi miten meneteltäisiin, jos prosessien kanssa tulee teknisiä ongelmia tai jos kohdejärjestelmät muuttuvat päivitysten seurauksena siten, ettei robotti enää osaa niissä operoida.

LÄHTEET

K. Beloof. How Much Time Are You Wasting on Manual, Repetitive Tasks? Viitattu 21.5.2022. https://www.smartsheet.com/content-center/product-news/automation/workers-waste-quarter-work-week-manual-repetitive-tasks

History of Robotic Process Automation (RPA), 2021. Viitattu 23.5.2022 https://www.robomotion.io/blog/history-of-rpa/

Savaram, R. RPA Statistics, MindMajix. Viitattu 16.10.2022 https://mindmajix.com/rpa-statistics

Shetty, S. Gartner Says Worldwide Spending on Robotic Process Automation Software to Reach $680 Million in 2018, 2018, Gartner. Viitattu 16.10.2022 https://www.gartner.com/en/newsroom/press-releases/2018-11-13-gartner-says-worldwide-spending-on-robotic-process-automation-software-to-reach-680-million-in-2018

Stamford, C. Gartner Says Worldwide RPA Software Spending to Reach $2.9 Billion in 2022, 2022, Gartner. Viitattu 16.10.2022 https://www.gartner.com/en/newsroom/press-releases/2022-08-1-rpa-forecast-2022-2q22-press-release

Choudhuri, A. What are PDD and SDD In RPA?, 2021, Probe CX. Viitattu 27.11.2022 https://www.probegroup.com.au/blog/what-are-pdd-and-sdd-in-rpa

Lahtinen, E. 2022. Ohjelmistorobotiikan hyödyt ja vaatimukset. Tieto- ja viestintätekniikan tutkinto-ohjelma. Tampereen ammattikorkeakoulu. Opinnäytetyö. Viitattu 14.12.2022. https://www.theseus.fi/handle/10024/755053

LIITTEET

Liite 1. Esimerkki prosessitason osasta Blue Prism -ohjelmistossa