LAB University of Applied Sciences

RPA-ylläpidon seurantarobotin toteutus

LAB-ammattikorkeakoulu
Insinööri (AMK)
2022
Ossi Ruhanen

Tiivistelmä

Tekijä(t)	Julkaisun laji	Valmistumisaika
Ruhanen, Ossi	Opinnäytetyö, AMK	2022
	Sivumäärä	
	30	

RPA-ylläpidon seuranta robotin toteutus

Tutkinto ja koulutusala

Insinööri (AMK), Tieto- ja viestintätekniikan koulutus (LAB)

Telia Finland Oyj

Tiivistelmä

Opinnäytetyön tavoitteena oli kehittää Telian RPA-tuotantoympäristössä toimivien robottien töiden seurantaa. RPA-ylläpito tilasi Blue Prism-ohjelmistorobotin analysoimaan ja keräämään dataa muiden robottien töistä.

RPA-robotiikan kysyntä on kasvanut nopeasti viimevuosien aikana. Ohjelmistorobotiikan etuina ovat prosessin automatisoinnin nopeus ja helppous, koska robotiikan kehitystyökalut vaativat vähän ohjelmointiosaamista ja se tarjoaa valmiin ajo- ja kehitysympäristön roboteille. Markkinoilla on useita robotiikka työkaluja tarjoavia yrityksiä ja näistä työkaluksi valittiin Blue Prism-ohjelmisto koska se oli jo ennestään Telialla käytössä.

Kehitettävä robotti seuraa muiden ohjelmistorobottien toimintaa ja hälyttää sähköpostilla hälytysrajojen ylittyessä. Hälytykset tapahtuvat sähköpostilla ja hälytysrajat määritellään Excel-tiedostoon jota robotti lukee.

Seurantarobotin toteutus onnistui ja se otettiin hyvin vastaan. Robotti mahdollistaa nyt virheiden nopeamman havainnoinnin ja useasti ongelmat saadaan jo korjattua ennen kuin niistä tulee asiakkaalta virheilmoituksia.

Asiasanat

RPA, ohjelmistorobotiikka, Blue Prism

Abstract

Author(s)	Type of Publication	Published
Ruhanen, Ossi	Thesis, UAS	2022
	Number of Pages	
	30	

RPA maintenance surveillance robot development

Degree and field of study

Bachelor of Engineering, Information and communications technology

Telia Finland Plc

Abstract

The aim of this thesis was to develop production RPA robots work tracking. RPA maintenance team ordered developing of Blue Prism robotic process automation robot to analyze and save data from other robot's work.

RPA robotics demand has been fast growing in the last years. The benefits of robotic process automation are faster development times and the ease of developing robots. This is because RPA tools require less programming knowledge, and they offer ready made run and development environment for the robots. There are many companies offering RPA tools in the market, and from these tools Blue Prism was chosen because it was already in use in Telia robotics.

Robot tracks other robots work and alerts with email if the alert levels are reached. Alerts are done with email and alert levels are defined in Excel file. Robot also saves data from all the production robots work queues for the maintenance team.

Work tracking robot development was successful, and it was well received. Robot enables faster error recognition and fixing. Often the errors in production are already fixed before the customer notifies us about them.

Keywords

RPA, Software robotics, Blue Prism

Sisällys

Termit ja lyhenteet

.NET = Avoimen lähdekoodin ohjelmistokehys

API = Application Programming Interface

Application modeller = Blue Prism-objektin mallinnustyökalu

Blue Prism = Ohjelmistorobotiikkaohjelmisto

RPA = Robotic Process Automation eli ohjelmistorobotiikka

Objekti = Kokoelma Blue Prism-ympäristön funktioita

1 Johdanto

Ohjelmistorobotiikan hyödyntäminen yrityksen prosessien tehostamisessa on kasvattanut suosiotaan viime vuosien aikana, vaikka se ei vaadikaan perinteisen ohjelmoinnin verran ohjelmointitaitoja. Ohjelmistorobotiikan tekijöitä ei ole tarpeeksi yritysten tarpeisiin. Ohjelmistorobotiikka-alustojen alentunut kustannustaso on mahdollistanut robotisoinnin myös pienille ja keskisuurille yrityksille. Ohjelmistorobotiikan etuina ovat prosessin automatisoinnin nopeus ja helppous, koska robotiikan kehitystyökalut vaativat vähän ohjelmointiosaamista ja se tarjoaa valmiin ajo- ja kehitysympäristön roboteille.

Telia Company AB on monikansallinen teleoperaattori, joka muodostui alun perin ruotsalaisen Telian ja suomalaisen Soneran fuusion tuloksena vuonna 2002. Telia tarjoaa sekä kiinteän- että mobiilin verkon tietoliikennepalveluja yritys- ja kuluttaja-asiakkaille.

Tässä opinnäytetyössä on tavoitteena kehittää tuotantoympäristössä toimivien robottien töiden seurantaa. RPA-robottien ylläpitäjät tarvitsevat nopeamman tavan reagoida tuotantoympäristön robottien virhetilanteisiin. Ylläpidon tilauksesta kehitettävä RPA Blue Prism-robotti analysoi ja tallentaa muiden robottien jonojen datan. Hälytykset tulee tehdä muuttujien avulla, joita tarkkailija voi halutessaan muuttaa Excel-tiedostosta. Automaattiset hälytykset tulee lähettää tarkkailijalle sähköpostilla. Prosessia ajetaan kerran päivässä ja ajon aikana tulee tarkistaa edellisen päivän työt sekä tarkkailijan määräämän aikaikkunan työt ja hälyttää rajojen ylittyessä. Työssä tulee käyttää Blue Prism (versio 6.10.3) ohjelmistoa ja robotin ohjauksessa Exceliä.

2 RPA-robotiikka

2.1 Määritelmä

Ohjelmistorobotiikka, eli RPA, on ihmisen tekemien tietoteknisten toimenpiteiden tai prosessien automatisoimista, joka tehdään suoraan käyttöliittymään. Tarkoituksena on mallintaa robotteja, joilla suoritetaan toiminteita lähes samalla tavalla kuin ihminenkin tekisi käyttöliittymää käyttämällä. (Tripathi 2018, 10.)

Robotti toimii siis aivan kuten ihminenkin, kirjautumalla järjestelmään ja käyttämällä sitä ihmisen tavoin. Esimerkiksi työntekijän tehtävänä on syöttää tilaustietoja lomakkeen kenttiin yrityksen tilausjärjestelmään, suorittaa tilaaminen ja ottaa tilauksesta kuitti talteen. Nämä tehtävät ohjelmoidaan robotille, jotta ihmisen ei tarvitse tehdä toistuvaa työtä itse.

Ohjelmistorobotiikkaa käytetään usein automatisoimaan applikaatioita, joihin ei ole API rajapintaa. RPA tarjoaa API kehitystä halvemman ja nopeamman ratkaisun lyhyellä aikavälillä. (Data Matics 2020.)

Ohjelmistorobotiikka ja API ratkaisut eivät poissulje toisiaan, prosesseissa käytetäänkin usein myös API ratkaisuja jos siihen on mahdollisuus. Rajapintakutsut ovat robotteja luotettavampia ja tehostavat ohjelmistorobotiikan toimintaa.

Robotiikan kehitystyökaluja ei ole suunniteltu isoja prosesseja varten, joten robotilla automatisoitavien prosessien tulisi olla suhteellisen yksinkertaisia. Ohjelmistorobotiikan työkalujen valmiin ympäristön ja helpon ohjelmoinnin takia kustannukset pysyvät pienempänä kuin ohjelmoinnissa yleensä, sekä kehitys on nopeampaa. Tästä syystä, voidaan virheellisesti alkaa automatisoida prosesseja, jotka ovat liian monimutkaisia roboteille. Prosessin tarkka analyysi auttaa välttämään nämä ongelmat.

Globaalisti yritysten käyttämä määrä rahaa RPA-ohjelmistorobotiikkaan on ennakoitu olevan vuoden 2022 loppuun mennessä 2.9 miljardia dollaria (Taulukko 1). Vaikka robotiikan kasvu onkin viime vuosina hidastunut, se ylittää silti ohjelmistokehityksen 16 % kasvun. (Gartner 2022.)

	2021	2022	2023
End-User Spending	2,389	2,854	3,352
Growth (%)	30.9	19.5	17.5

Taulukko 1. RPA-loppukäyttäjien käyttämä raha (Gartner 2022)

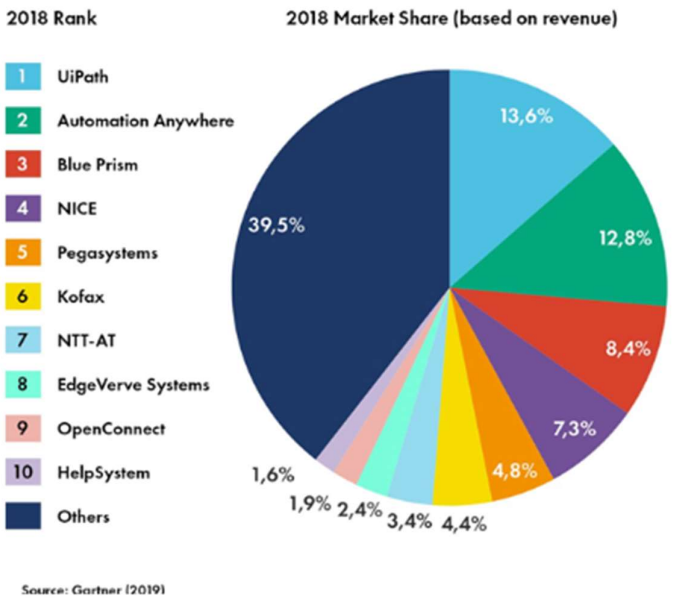
2.2 Ohjelmistorobotiikan työkalut

Ohjelmistorobotiikan tekemiseen on tarjolla useita eri työkaluja, ja suurin osa työkaluista perustuu vuokaavio kehittämiseen. Työkaluilla pyritään usein minimoimaan ohjelmoinnin tarve, jotta kehittäjän ei tarvitsisi osata erikseen ohjelmointia vaan kehittäminen on tehty mahdollisimman helpoksi. Osa työkaluista tarjoaa mahdollisuuden applikaatioiden käyttämisen nauhoittamiseen, ja tällä toiminteella pystytään helposti luomaan pohja robotille, mutta esimerkiksi virheenkäsittely on tehtävä erikseen kehittäjän toimesta.

Kuvassa 1 esitetään kymmenen markkinaosuudeltaan suurinta ohjelmistorobotiikkatyökalua tarjoavaa yritystä. Markkinoiden kolme suurinta toimijaa, UIPath, Automation Everywhere ja Blue Prism tarjoavat RPA-työkalua taustarobotiikan kehitykseen. Nice tarjoaa myös mahdollisuuden robottityöpöytäautomaatiolle (RDA). Työpöytäautomaatiolla tarkoitetaan käyttäjän omalla koneella suoritettavaa robotiikkaohjelmistoa, joka auttaa käyttäjää esimerkiksi asiakaspalvelussa.

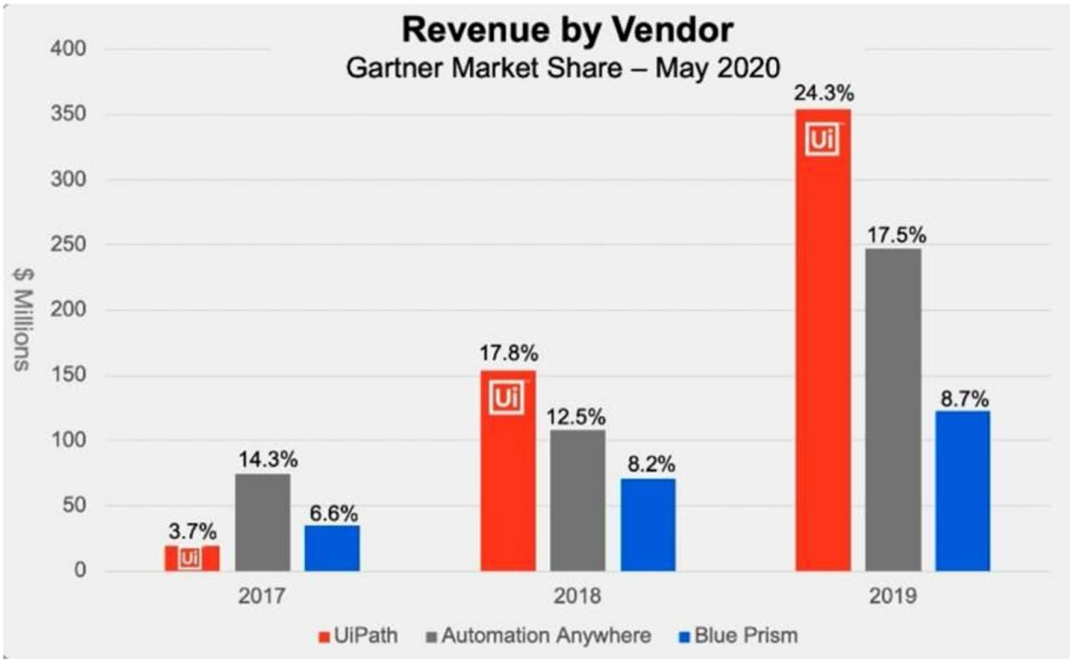
Yritysten tarjoamat työkalut ovat perinteisesti olleet kalliita ja niitä on myyty isoille yrityksille. Nyt markkinoille on alkanut tulla myös yrityksiä, jotka tarjoavat ohjelmistorobotiikkatyökaluja myös pienemmille yrityksille.

Markkinoilla on myös tarjolla ohjelmistorobotiikan kehitystä ja ylläpitoa palveluna myyviä yrityksiä. Robotiikan kehityksen ja ylläpidon ulkoistaminen voi olla pienelle yritykselle paras ratkaisu, jos robottia ei tarvitse useasti päivittää.



Kuva 1. 10 suurinta ohjelmistorobiikan toimijaa (Patil 2022)

UIPath ja Automation Everywhere ovat kasvattaneet tulostansa ja osuuttaan kokonaismarkkinoista tehokkaasti, Blue Prism taas on säilyttänyt markkinaosuutensa tasaisena (Kuva 2).



Kuva 2. Markkinaosuudet (Mullakaral 2022)

2.3 Älykäs automaatio

IA eli älykäs automaatio yhdistää ohjelmistorobotiikan älykkäisiin teknologioihin. Robotti voi käyttää prosessissa esimerkiksi koneoppimista ja tehdä valintoja siihen perustuen.

Seuraavassa on lista teknologioita joita on käytetty robottien kanssa (Cognizant 2022):

- Tekoäly
- Koneoppiminen
- Konenäkö
- Luonnollisen kielen käsittely (NLP)
- Prosessilouhinta

RPA-robotilla voidaan esimerkiksi hyödyntää puheentunnistusta. Puheentunnistuksella muutetaan asiakaspuhelut tekstimuotoon ja lähetetään data robotille käsiteltäväksi. Robotti etsii datasta vikatilanteita ja tekee hälytyksiä, jos sen hälytysrajat ylittyvät.

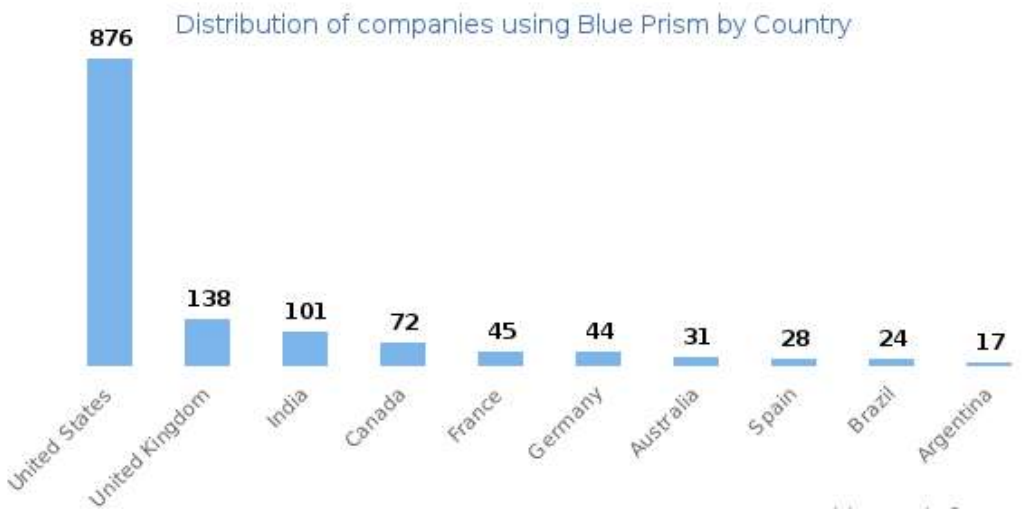
Ohjelmistorobotiikkaa voidaan myös hyödyntää chattiboteissa. Asiakas pystyy syöttämään chattibotin kautta suoraan töitä taustalla suoritettavalle ohjelmistorobotiikalle tehtäväksi. Tämä poistaa kokonaan välistä ihmistyön tarpeen.

Tulevaisuudessa ohjelmistorobotiikka yhdistyy entistä enemmän älykkäisiin teknologioihin ja tällä mahdollistetaan entistä haastavampien ongelmien ratkaisu robottien avulla. Robotilla toteutetaan applikaation käyttö mutta monimutkaista logiikkaa vaativat toiminteet suoritetaan robotin ulkopuolella.

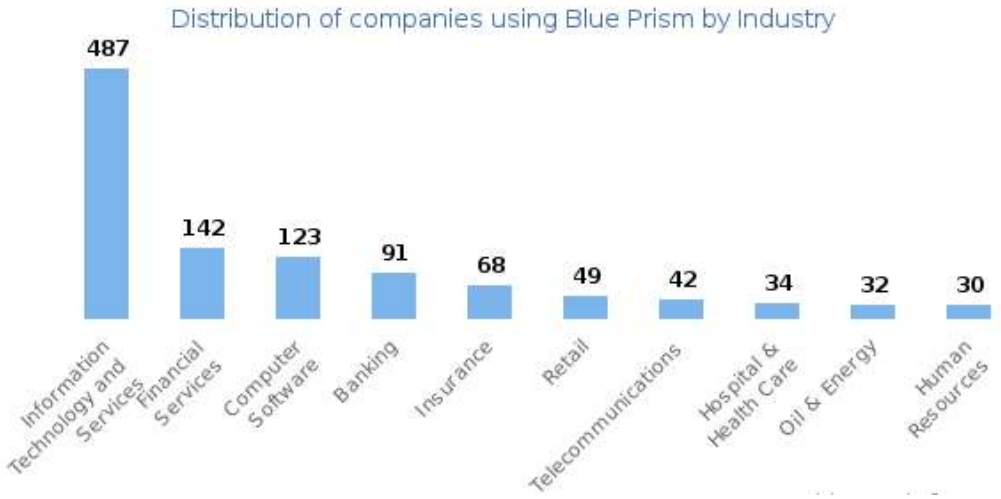
2.4 Blue Prism

Blue Prism on RPA-kehittämisessä käytetty ohjelmisto. Yritys on perustettu vuonna 2001 Alastair Bathgaten ja David Mossin toimesta. Yrityksen pääkonttori sijaitsee Warringtonissa, Englannissa. Yrityksellä on yli 2000 asiakasta eri toimialoilta ympäri maailmaa. Blue Prismin mukaan he itse kehittivät RPA-termin (Tech Crunch 2019). Yritys on isoimpia ohjelmistorobotiikan toimijoita. Blue Prismin yrityskumppaneihin kuuluu isoja yrityksiä, kuten Daimler, 3m, Ebay, Siemens, Jaguar sekä NHC. (Blue Prism 2021a.)

Kaaviossa 1 on esitetty Blue Prismiä käyttävät yritykset maittain. Suurin osa käyttäjistä on Yhdysvaltalaisia tieto- ja viestintäteknologia yrityksiä (kaavio 2) ja 54% yrityksistä on yli 1000 työntekijän yrityksiä. (Enlyft 2022)



Kaavio 1. Blue Prismiä käyttävien yritysten sijainti (Enlyft 2022)

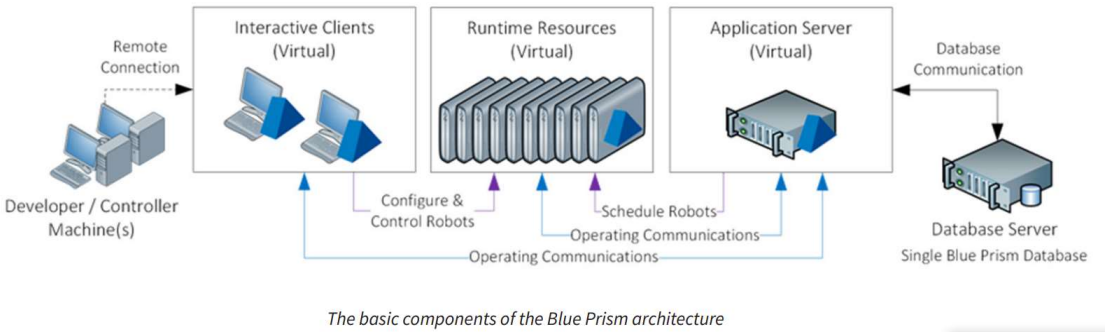


Kaavio 2. Blue Prismiä käyttävien yritysten teollisuusalat (Enlyft 2022)

Blue Prism on kehitetty Microsoftin .NET-teknologialla. Ohjelma tukee Visual basic ja C# ohjelmointikieliä, mutta koodin kirjoittaminen ei ole pakollista. Automatisoitavat kohteet ovat yleensä Windows-käyttöliittymiä, Microsoft Office, verkkoselaimia ja -palveluja Blue Prism myy lisenssejä asiakkaille ja he rakentavat, hallinnoivat ja omistavat omat robottinsa.

Blue Prism-arkkitehtuuri perustuu kolminaisuuteen. Ohjelmassa on resurssi, mallinnus- ja hallintatyökalut, joilla automatisointi toteutetaan. Hallintatyökalu toimii Windows Server -palvelimella ja se yhdistetään tietokantaan, esimerkiksi MS SQLServer – palvelimeen, johon data tallennetaan.

Kuvassa 3 on esitetty Blue Prismin kolminaisuus arkkitehtuuri mallinnus, resurssi ja hallinta. Nämä on yhdistetty kuvassa näkyvään tietokantaan Database Server. Mallintajaa kuvaa Developer/Controller.

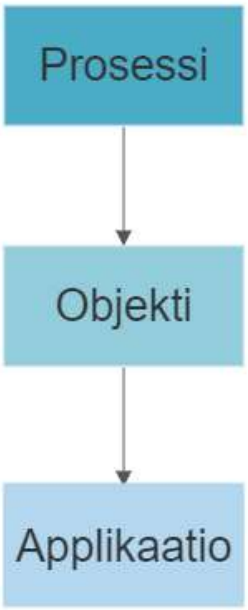


Kuva 3. Blue Prism-arkkitehtuuri (Blue Prism 2021b)

Käyttöliittymä on jaettu viiteen moduuliin:

- Home sisältää kojelaudan, joka koostuu dynaamisesti muokattavista prosessikuvaajista. Kuvaajat päivittyvät automaattisesti prosessien ajojen aikana.
- Studiossa tapahtuu uusien prosessien ja objektien luonti käytettävissä oleville resursseille.
- Control moduuli on prosessien ajojen toteutusta ja seurantaa varten.
- Analytics työkalulla tehdään home moduulin kuvaajat.
- Releases mahdollistaa toteutuksien siirron muihin Blue Prism-ympäristöihin paketteina ja yksittäisinä tiedostoina.
- Digital Exchange on internet ympäristö prosessien ja objektien jakoon muiden kehittäjien kanssa.
- System moduulista hallitaan tietokantoja, työjonoja, käyttäjiä ja rooleja.

Kuvassa 4 on esitetty Blue Prism-ohjelmointi jaettuna kolmeen osaan, prosessi-, objekti- ja applikaatiotasoille. Automatisoitavia applikaatioita ohjelmoidaan ylemmällä tasolla Process studiosta. Täällä tehdään robotin käyttölogiikka ja virheenhallinta. Objektitasolla tehdään pienempiä monikäyttöisiä funktioita, joilla robotti käyttää applikaatiota.



Kuva 4. Ohjelmointitasot

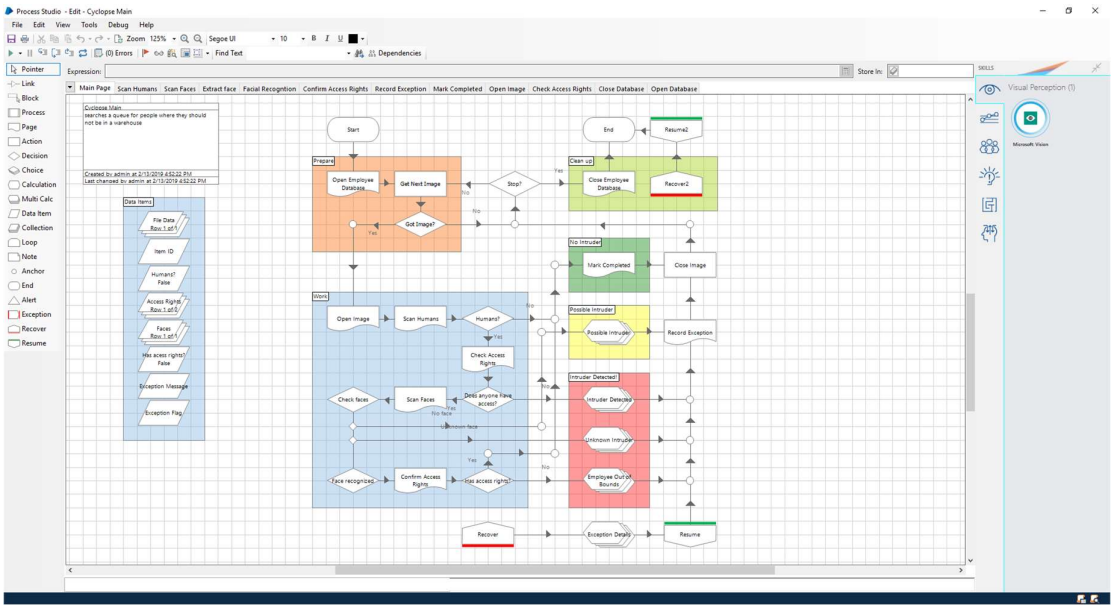
2.4.1 Process-studio

Ylimmän tason ohjelmointi tehdään editorissa vuokaavioiden avulla. Jokainen vuokaavio tehdään omalle sivullensa. Robotin prosessin logiikka luodaan ensimmäiselle sivulle, josta se jaetaan alisivuihin. Logiikka koostuu valinnoista, sivuista, laskentalogiikasta, virheenhallinnasta ja objektitason toiminteista.

Kuvassa 5 esitetty esimerkki prosessin pääsivusta, jossa logiikka on jaettu erillisiin osaprosesseihin sekä pääsivun käyttämiin data muuttujiin. Kuvassa osaprosessit on erotettu toisistaan

Virheenhallinta toimii rajoittamalla osia prosessista aliprosesseiksi, niistä virhe kuplii palautus osioon. Yleinen käytäntö on yrittää osaa prosessista kolmeen kertaan, ja jos osaprosessi ei onnistu, virhe kuplii pääsivulle aiheuttaen prosessin terminoinnin.

Prosessitasolta voidaan kutsua aliprosesseja, ja tässä tulee ottaa huomioon, että aliprosessin ajettua se vapautetaan muistista automaattisella .net roskien kerääjällä. On mahdollista että muistin tyhjentäminen ei aina tapahdu riittävän nopeasti, jolloin tästä aiheutuu muistivuotoja. Paras käytäntö on käyttää näissä tilanteissa objekteja.



Kuva 5. Process studio (Blue Prism)

2.4.2 Object-studio

Objektitasolla tehdään pienempiä monikäyttöisiä funktioita, jotta uudelleenkäytettävyys paranee. Jokaiselle applikaation toiminteelle tulisi luoda oma objekti sivu, jota voidaan kutsua prosessitasolta. Objekti voi sisältää useita sivuja, ja usein prosessitasolla käytetään monia eri objekteja. Objektit suositellaan jaettavaksi utility ja applikaatio objekteiksi. Utility objekteilla tarkoitetaan funktioita, jotka eivät ole applikaatiokohtaisia, vaan niitä voidaan käyttää useissa eri applikaatioissa. Objektitasolta voidaan myös kutsua muita objekteja. Tämä mahdollistaa isäntäobjektien teon, joilla voidaan kutsua aliobjekteja.

Objekteja pyritään rakentamaan mahdollisimman yksinkertaisiksi funktioiksi. Ongelmana isommissa funktioissa on uudelleenkäytettävyys. Jos objektiin on tehty isompi funktio, joka esimerkiksi kirjautuu www-sivulle sisään ja tekee kirjauksen, sen muuttaminen vaatii testausta myös muissa objektia käyttävissä prosesseissa. Oikea tapa tehdä tämä toteutus on jakaa objekti pienempiin funktioihin, käyttäjätietojen kirjoittaminen, sisäänkirjautuminen ja lopulta kirjauksen tekeminen. Kirjauksen tekeminenkin tulisi tässä tapauksessa rikkoa niin pieniin funktioihin, kuin se vain on mahdollista.

Objekteja tehdessä on hyvä muistaa, että koko objekti ladataan käyttömuistiin. Tämän takia objekteja ei kannata tehdä liian monisivuisiksi, vaan käyttää erillisiä objekteja eri osiin applikaatiosta.

2.4.3 Application modeller

Application modeller on työkalu objektitasolla applikaation mallinnukseen. Mallinnus tehdään suoraan applikaation mallinnettavia osia tunnistamalla, ja applikaation tulee olla käynnissä, kun mallinnusta tehdään.

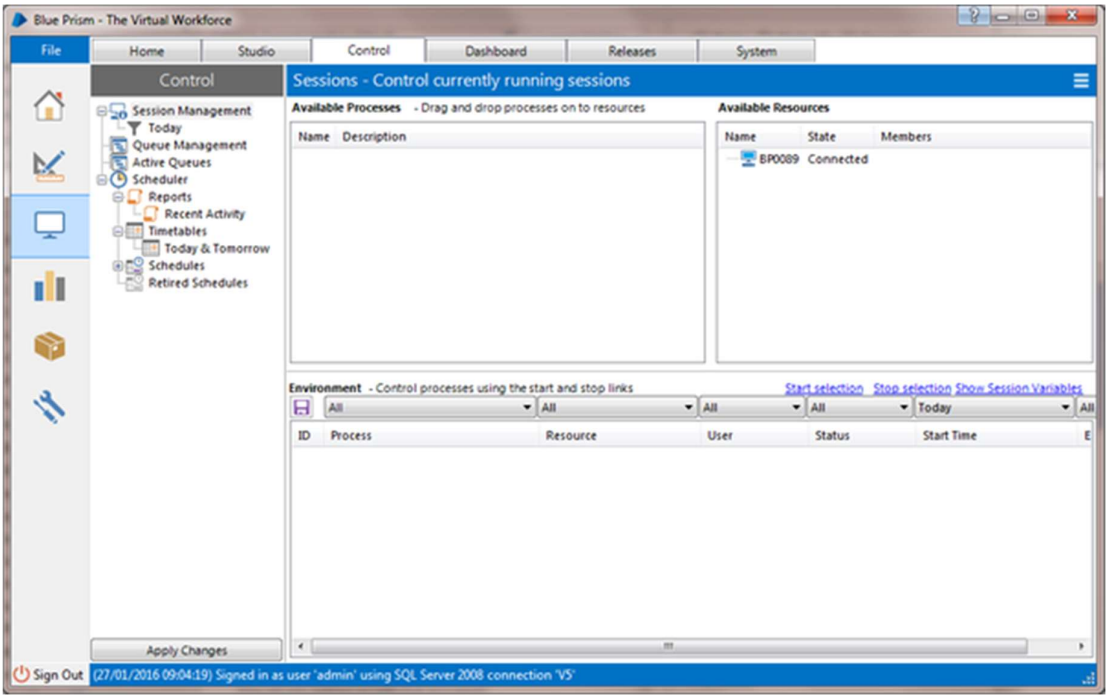
Elementti mallinnetaan spy modessa klikkaamalla haluttua elementtiä. Application modeller palauttaa elementin tunnistetiedot, näistä tiedoista valitaan käytettävät parametrit tunnistusta varten. Highlight toiminteella voidaan varmistaa, että tunnistus toimii halutusti.

Elementtien tunnistamisen työkalun toimintamoodit:

- Browser mode, tunnistaa selaimen elementit.
- Win32 mode, tunnistaa Windows applikaatiot.
- UI Automation mode, tunnistaa .Net tunnisteet.
- Region mode, tunnistaa elementit kuvantunnistuksella

2.4.4 Control room

Kuvassa 6 näkyvästä ohjauskeskuksesta (Control room) ohjataan prosessien käynnistymistä aikataulujen avulla ja seurataan jonotöitä jonojen kautta. Robotit käynnistyvät aikataulujen kautta, mutta robotti voi myös tarvittaessa käynnistää toisia robotteja. Jonoista pystyy poistamaan töitä, pakottamaan työn uudelleenkäsittely ja merkitsemään töitä virheellisiksi.



Kuva 6. Control room (Blue Prism)

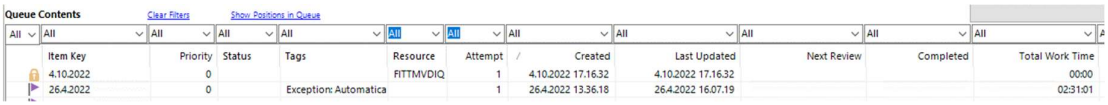
Kuvassa 7 näkyvä työjono on sisäinen lista jonotöistä, joka mahdollistaa toiminnallisuuden monitorointiin, jakamiseen, lokittamiseen ja datan käsittelyyn Blue Prism-prosessien toimesta. Työjono on oleellinen komponentti Blue Prism-ratkaisuissa. Kaikki työjonot löytyvät ohjauskeskuksesta listattuna.

Jonotyö rakenteeseen kuuluu:

- Status ikoni, sisältää tilat lukittu, valmis, virhe tai odottava
- Item Key, sisältää työn tunnisteen. Sitä ei voida muokata enää jonotyön luonnin jälkeen.
- Priority sisältää työn kiireellisyyden asteikolla 0-10. Korkein numero otetaan käsittelyyn ensin.

- Status, sisältää tekstikentän työn kulun merkitsemiseen jotta prosessia voidaan jatkaa tekemättä edellisiä prosessin toimintoja useaan kertaan.
- Tags, sisältää tekstikentän, jolla kehittäjä erottelee töitä ryhmiin tai merkitsee töihin haluttuja tunnisteita.
- Resource, sisältää resurssin, jolla jonotyö on käsitelty.
- Attempt, sisältää jonotyön käsittelyjen määrän. Ainoastaan virheet käsitellään uudestaan.
- Created, sisältää aikaleiman jonotyön luontihetkestä.
- Last updated, sisältää edellisen käsittelyn aikaleiman.
- Next review, sisältää seuraavan käsittelyn aikaleiman odottavissa töissä.
- Completed, sisältää työn valmistumisen aikaleiman.
- Total work time, sisältää työn tekemiseen kuluneen ajan.

Työjonoa voidaan suodattaa millä tahansa näistä muuttujista, ja tämä antaa ylläpitäjälle paremman kuvan jonosta.

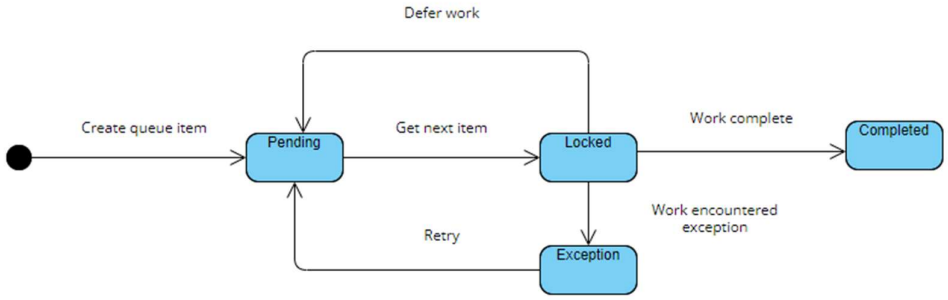


Kuva 7. Prosessin työjono

Jonotyön tilat:

- Odottava, sisältää jonotyöt jotka odottavat käsittelyä. Käsittelyyn ottamisen ajankohta määritetään päivämäärällä (deferral date/time).
- Lukittu, sisältää robotin käytössä olevat työt. Lukitseminen estää työn käsittelyn kahdesti. Lukon voi tarvittaessa manuaalisesti poistaa jonosta.
- Valmis, sisältää robotin valmiiksi saadut työt. Työt voidaan poistaa jonosta manuaalisesti tai prosessin toimesta.
- Virhe, sisältää epäonnistuneet työt. Virheet jaetaan kahteen eri kategoriaan, Järjestelmä- ja Business virhe. Jonoon määritellään järjestelmä managerin (System Manager) kautta kuinka monta kertaa järjestelmävirheitä yritetään uudestaan. Jokaisesta yrityksestä syntyy uusi työ samalla datalla, business virheitä ei yritetä uudestaan.

Kuvassa 8 on esitetty jonotyön tilat tilakoneena. Jonotyö luodaan aina odottavaan tilaan, jonka jälkeen se siirtyy aina lukossa tilaan, kun robotti sitä käsitelee. Robotti asettaa työn takaisin odottavaan tilaan, jos työtä tarvitsee vielä myöhemmin jatkaa. Virheen tapahtuessa tila asetetaan virhetilaan ja työ voidaan vielä asettaa takaisin odottavaan tilaan tai tehdä kokonaan uusi jonotyö. Työn valmistuessa se merkitään valmistunut tilaan ja työtä ei voi enää ottaa uudestaan käsittelyyn.



-

Kuva 8. Jonotyö tilakoneena

3 Tietotekniikan ylläpidosta

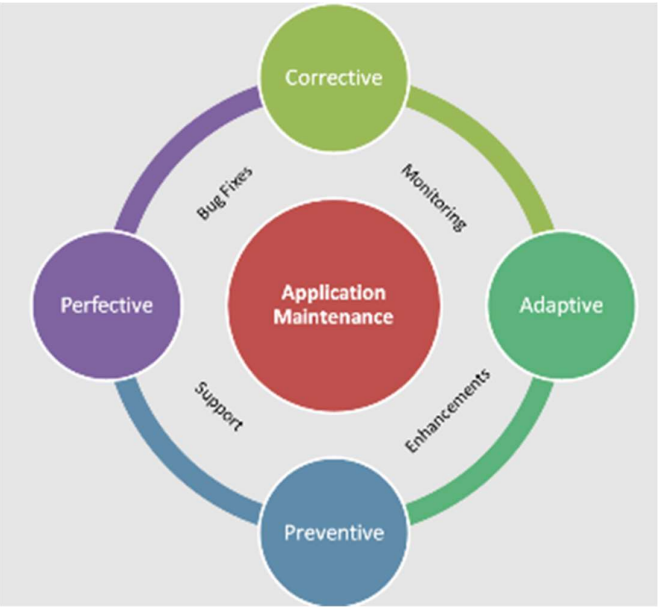
3.1 Ylläpidon kustannukset

Ohjelmistojen ylläpito on tunnustettu ohjelmistotuotannon avainalueeksi. Ylläpidon työn laajuus näkyy ohjelmiston muokkaamisen kustannuksissa. Tämä kustannus voi olla jopa 70 % elinkaarikustannuksista (Grubb & Armstrong 2003, vii).

Ohjelmistorobotiikan robottien ylläpito on yleisesti halvempaa ylläpitää, koska robotteja on helpompi muokata ja niissä on vähemmän riippuvuuksia muihin prosesseihin pienen kokonsa takia. Isompien robottien kohdalla ongelmaksi tuleekin usein objektitason jakaminen toimivasti, jotta robottia voi korjata useampi työntekijä samaan aikaan. Objektia voi käsitellä vain yksi henkilö kerrallaan, joten objekteja ei kannata tehdä liian isoiksi.

3.2 Ohjelmistoihin tehtävien muutoksien luokittelu

Tietotekniikan ylläpidossa korjaavat toimenpiteet on hyvä jakaa neljään eri luokkaan, koska toimintatapa virheen korjaamiseksi muuttuu luokan mukaan. Kuvassa 9 näkyvät toimenpiteet ovat ohjelmistoon tehtäviä muutoksia, niihin sisältyy korjaavia-, mukautuvia-, parantavia- ja ennaltaehkäiseviä muutoksia.



Kuva 9. Ohjelmistoon tehtävät muutokset (Lookfar Labs 2022)

3.2.1 Korjaava muutos

Korjaava muutos tarkoittaa ohjelmiston vioista johtuvaa muutosta. Vika voi johtua suunnitteluvirheistä, logiikkavirheistä tai koodausvirheistä. Suunnitteluvirheitä syntyy, kun esimerkiksi ohjelmistoon tehdyt muutokset ovat virheellisiä, epätäydellisiä, virheellisesti kommunikoituja tai muutospyyntö ymmärretään kehittäjän toimesta väärin. Logiikkavirheet johtuvat virheellisistä testeistä ja johtopäätöksistä, suunnitteluspesifikaatioiden virheellisestä toteutuksesta, virheellisestä logiikasta tai tietojen puutteellisesta testauksesta. Koodausvirheet johtuvat yksityiskohtaisen logiikkasuunnittelun virheellisestä toteutuksesta ja lähdekoodilogiikan virheellisestä käytöstä. Vikoja aiheuttavat myös tietojenkäsittelyvirheet ja järjestelmän suorituskykyvirheet. Kaikki nämä virheet, joita usein kutsutaan "jäännösvirheiksi" tai "bugeiksi", estävät ohjelmistoa noudattamasta sovittuja määrityksiä. Virheestä johtuvan järjestelmävian sattuessa ryhdytään toimiin ohjelmistojärjestelmän toiminnan palauttamiseksi (Grubb & Armstrong 2003, 35).

Johdon painostuksen alaisena ylläpitohenkilöstö turvautuu joskus hätäkorjauksiin. Tämän lähestymistavan tilapäinen luonne aiheuttaa useita ongelmia, joihin kuuluvat ohjelman lisääntynyt monimutkaisuus ja odottamattomat heijastusvaikutukset. Ohjelman lisääntynyt monimutkaisuus johtuu yleensä ohjelman rakenteen rappeutumisesta, mikä tekee ohjelman ymmärtämisestä entistä vaikeampaa, ellei mahdotonta. Tätä kutsutaan joskus "spagettioireyhtymäksi" tai "ohjelmiston väsymykseksi", mikä tarkoittaa, että ohjelman muutosvastus on maksimissaan (Grubb & Armstrong 2003, 35).

3.2.2 Mukautuva muutos

Vaikka jäännösvirheet eivät aiheuta ongelmia, ohjelmisto muuttuu, kun sitä yritetään mukauttaa sen jatkuvasti muuttuvaan ympäristöön. Mukautuva muutos on muutos, joka johtuu tarpeesta tehdä ohjelmistojärjestelmään muutoksia sen ympäristön muuttuessa. Termi ympäristö viittaa tässä yhteydessä kaikkien olosuhteiden ja vaikutteiden kokonaisuuteen, jotka vaikuttavat ulkopuolelta järjestelmään, esimerkiksi liiketoimintasäännöt, hallituksen politiikat, työtavat, ohjelmistojen ja laitteistojen käyttöympäristöt. Muutos koko tähän ympäristöön tai sen osaan edellyttää vastaavan muutoksen tekemistä ohjelmistoon.

Mukautuva ylläpito sisältää kaikki työt, jotka on aloitettu ohjelmiston siirtämisen seurauksena toiselle laitteistolle tai ohjelmistoalustalle, käyttöjärjestelmälle tai uudelle prosessorille (Grubb & Armstrong 2003, 36).

3.2.3 Parantava muutos

Tätä termiä käytetään kuvaamaan muutoksia, jotka on tehty nykyisten vaatimusten laajentamiseksi. Tämä perustuu olettamukseen, että kun ohjelmistosta tulee käyttökelpoinen, käyttäjät kokeilevat uusia tapauksia laajemmalle kuin mitä varten ohjelmisto alun perin kehitettiin. Vaatimusten laajentaminen voi tapahtua olemassa olevan järjestelmän toiminnallisuuden parantamisen tai laskentatehokkuuden parantamisen muodossa (Grubb & Armstrong 2003, 37).

3.2.4 Ennaltaehkäisevä muutos

Ohjelmistojärjestelmän parissa tehty työ rakenteen heikkenemiseen liittyvien ongelmien ratkaisemiseksi tunnetaan ennaltaehkäisevänä muutoksena. Ennaltaehkäisevä muutos tehdään toimintahäiriöiden estämiseksi tai ohjelmiston ylläpidettävyyden parantamiseksi. Muutos käynnistetään yleensä huolto-organisaation sisältä, jotta ohjelmia olisi helpompi ymmärtää ja siten helpottaa tulevaa kunnossapitotyötä. Ennaltaehkäisevä muutos ei yleensä johda perustoimintojen oleelliseen lisäykseen. Esimerkkejä ehkäisevistä muutoksista ovat koodin uudelleenjärjestely, koodin optimointi ja dokumentaation päivitys. Ohjelmistojärjestelmään tehtyjen pikakorjausten sarjan jälkeen sen lähdekoodin monimutkaisuus voi kasvaa hallitsemattomalle tasolle, mikä oikeuttaa koodin täydellisen uudelleenjärjestelyn. Koodioptimointi voidaan tehdä, jotta ohjelmat voivat toimia nopeammin tai hyödyntää tallennustilaa tehokkaammin. Käyttäjä- ja järjestelmädokumentaation päivittäminen, vaikka se jätetään usein huomiotta, on usein tarpeen, kun jotakin ohjelmiston osaa muutetaan. Asiakirjoja, joihin muutos vaikuttaa, tulee muokata vastaamaan järjestelmän nykyistä tilaa (Grubb & Armstrong 2003, 40).

3.3 Ylläpidon jatkuva tuki

Huoltotöiden luokka, joka tarjotaan ei-ohjelmointiin liittyvien työpyyntöjen tyydyttämiseksi. Jatkuva tuki, vaikka se ei sinänsä ole muutos, on olennaista toivottujen muutosten onnistuneelle viestimiselle. Jatkuvan tuen tavoitteita ovat tehokas viestintä ylläpidon ja loppukäyttäjähenkilöstön välillä, loppukäyttäjien koulutus ja liiketoimintatiedon tarjoaminen käyttäjille ja heidän organisaatioilleen päätöksenteon helpottamiseksi.

Tehokas viestintä on olennaista niiden osapuolten välillä, joihin ohjelmistojärjestelmän muutokset vaikuttavat. Ylläpito on ohjelmiston elinkaaren asiakasintensiivisin osa, koska suurempi osa ylläpidosta kuluu asiakkaiden pyytämien parannusten tekemiseen kuin muihin järjestelmän muutoksiin. Käytetyn ajan ja resurssien on perustuttava asiakastyytyväisyyteen. Tärkeää onkin osoittaa jatkuvasti pyrkivänsä tyydyttämään asiakkaiden tarpeita niin,

että molemmat osapuolet ymmärtävät toisiaan. Tämä suhde on äärimmäisen tärkeä. Se voi parantaa toimittajan ja käyttäjän välistä viestintää, mikä antaa vähemmän tilaa käyttäjien muutospyyntöjen väärin ymmärtämiselle. Se tarkoittaa, että kunnossapito-organisaatio pystyy paremmin ymmärtämään käyttäjien liiketoimintatarpeita ja voi siten välttää teknisiä päätöksiä, jotka ovat ristiriidassa liiketoimintapäätösten kanssa. Lisäksi on mahdollista lisätä käyttäjien osallistumista ylläpitoprosessiin, mikä on olennainen osa onnistunutta huoltoa ja käyttäjien tyytyväisyyttä. Epäonnistunut viestintätaso ylläpito-organisaation ja ohjelmistomuutosten kohteena olevien välillä voi johtaa ohjelmiston epäonnistumiseen (Grubb & Armstrong 2003, 42).

4 Työn toteutus

4.1 Käytetty data

Robotilla seurataan robotteja, joiden jonosta kerätään tiedot robotin tekemistä töistä. Työ voi olla kolmessa eri tilassa, valmis, virhe tai odottava. Virheet jaetaan kahteen eri kategoriaan, business tai järjestelmä. Odottava työ voi olla myös siirretty myöhemmin tehtäväksi ajastettu tilaan. Robotti kerää jonoista valmis, business, järjestelmä, odottava ja ajastettustatuksen työt ja kirjaa niiden määrät ylös. Päivittäin kerättävä data kerätään lukemalla seurattavien robottien jonot. Seurantarobotti laskee käynnistyessään tarkastushetkeksi 24 tuntia taaksepäin.

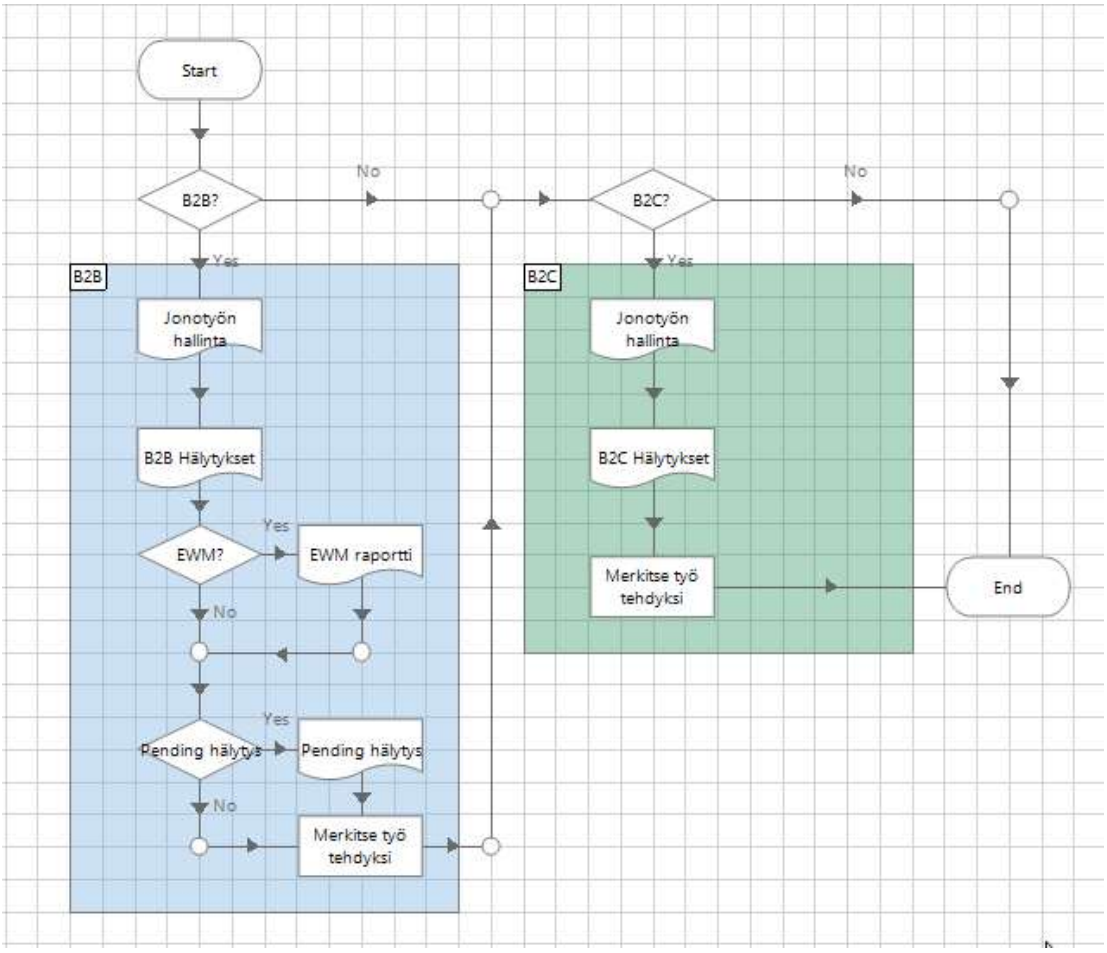
4.2 Hälytysrajat

Robotti tekee automaattisia hälytyksiä sähköpostitse, jos käyttäjä on kirjannut sähköposti osoitteen ohjaus Excel-tiedostoon. Hälytysrajoja ovat system exception, pending, maximum deferred ja oldest allowed work. Käyttäjä määrittää nämä arvot itse ja jos joku arvoista ylittyy, robotti kirjaa tiedon sähköpostiin ja lähettää hälytyksen. System exception kenttään syötetään arvona, kuinka monta prosenttia kaikista töistä saa olla järjestelmävirhe tilassa. Pending hälytysrajaan määritetään odottavien töiden maksimimäärä. Maximum deferred work etsii töitä, jotka on siirretty tehtäväksi liian kauaksi tulevaisuuteen. Oldest allowed work etsii liian vanhoja töitä, ja muuttuja on numero päivinä tästä päivästä taaksepäin.

Robottiin on tehty toiminto muokattavan aikaikkunan määrittämiselle, jos tähän kenttään on annettu arvo, robotti etsii muuttujan määrän päiviä taaksepäin ja hälyttää järjestelmävirheistä ja odottavista töistä tällä aikaikkunalla.

4.3 Blue Prism-prosessi

Kuten kuvassa 10 näkyy, prosessi jaettiin kahteen osaan, B2C ja B2B prosessit. Molemmissa prosesseissa on omat ohjaus Excel-tiedostot, joissa määritetään hälytysrajat sekä kenelle hälytykset lähetetään. B2B prosessissa on myös päivittäin kolme kertaa ajettava odottavien töiden hälytys, joka tarkkailee jonojen tekemättömien töiden määriä ja hälyttää, jos robotti ei pysty purkamaan työkuormaansa.



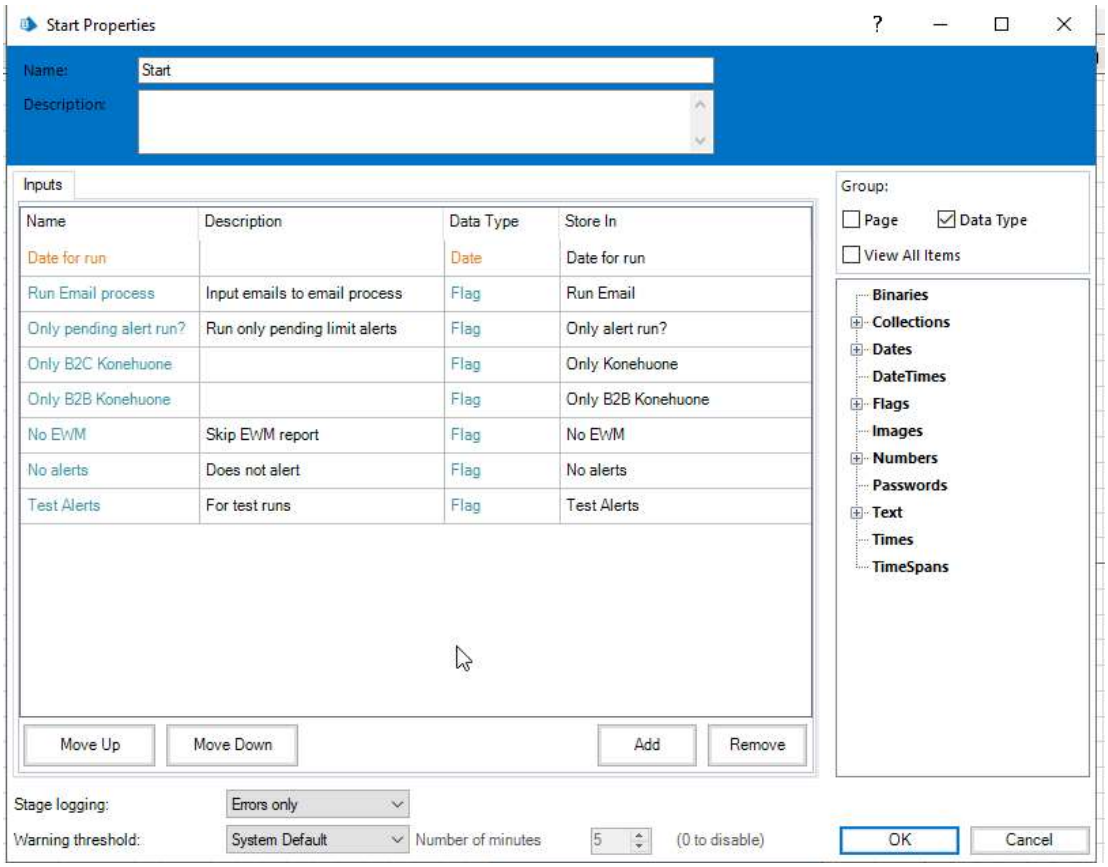
Kuva 10. Havainnekuva robotin logiikasta

Robottia käynnistäessä voidaan määrittää päivämäärä, jonka aikaisia töitä tarkastellaan. Robotissa on muuttujat, joilla määritetään mitä hälytyksiä halutaan ajaa, esimerkiksi voidaan tehdä ajo ilman B2C hälytyksiä, jos B2B puolen hälytykset on jo tehty, mutta B2C hälytyksien aikana tapahtui virhe.

Kuvan 11 aloitus parametreilla voidaan ohjata robotin ajoa. Näitä parametreja pystytään asettamaan, kun robotille tehdään ajastus ohjauskeskuksesta. Pakollisia parametreja ei ole, kaikki parametrit voidaan jättää tyhjiksi, jolloin ne asettuvat oletusarvoihin.

Prosessilla voidaan tarkastella minkä tahansa päivän dataa menneisyydestä. Annetulla päivämäärällä suodatetaan muiden robottien jonoista palautuneita jonotöitä. B2B ja B2C raportteja ja hälytyksiä voidaan generoida myös erikseen. Oletusarvona robotti ajaa molemmat osiot.

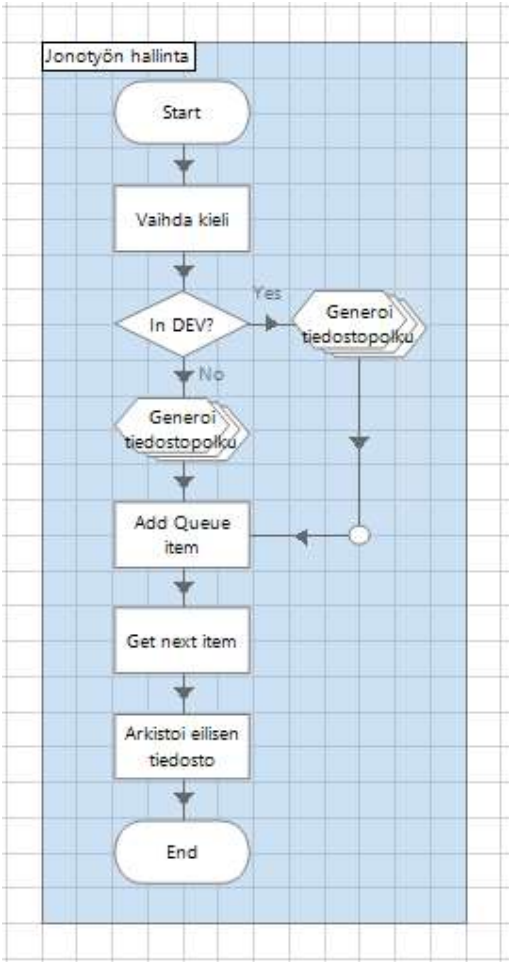
Jos ajo on jostain syystä epäonnistunut, mutta raportti on generoitunut, sen uudelleen generointi voidaan ohittaa. Testauksen ajaksi hälytykset voidaan ottaa pois päältä kokonaan, tai ajaa vain testihälytyksiä.



Kuva 11. Aloitusparametrit

Jonotyön hallinta vaihtaa ympäristön kulttuurin suomen kielelle, tämä vaikuttaa prosesseissa laskettaviin päivämääriin. Logiikassa on rakennettu kehitysympäristöön oma tiedostopolku, jotta tuotannon järjestelmiä ei käytetä kehittäessä tai testatessa prosessia.

Jonotyön lisäys toimii kuvassa 12 näkyvällä sisäisellä toiminteella Add Queue Item. Se lisää prosessin jonoon työn käyttäen jono avaimena ajon päivämäärää. Get next item ottaa työn heti käsittelyyn ja asettaa jonotyön lukitus tilaan. Verkkolevylle on tehty myös arkistointi, johon siirretään vanhojen ajojen tekemät tiedostot.



Kuva 12. Jonotyön hallinnan toimintaperiaate

B2B ja B2C hälytykset käyttävät samaa koodia. Sivulle syötetään parametreinä kumpaa osiota halutaan käyttää. Molempien hälytyksille on oma ohjaus Excel-tiedosto, jossa määritellään tarkasteltavat työjonot sekä hälytysmuuttujat.

Kuvassa 13 näkyvä prosessi lukee ohjaus Excel-tiedostosta jonon nimet, hälytysrajat ja sähköpostiosoitteet. Robotti käy läpi jokaisen rivin Excelistä ja tekee niistä tietoalkion. Jokaisessa rivissä on tarkasteltavan robotin jonon nimi sekä muut tarvittavat tiedot.

Exception hälytys tarkastaa jonon kaikkien virheiden määrän, jos prosentti ylittyy, tehdään hälytys.

System exception hälytykset toimivat seuraavalla kaavalla $\frac{SE\ limit}{(Completed\ work + Exceptions)} * 100$

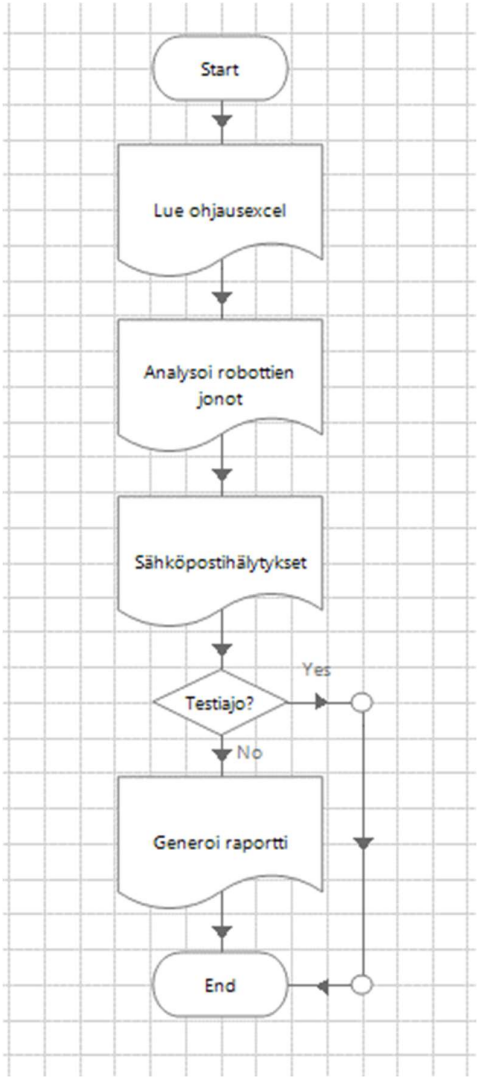
SE limit on määritetty excelissä ja se jaetaan kaikkien tehtyjen työn määrällä. Esimerkiksi jos halutaan hälyttää tilanteissa, joissa virheitä on 10% töistä, SE limit olisi 10.

Pending hälytykset toimivat odottavien töiden määrän ylittäessä hälytysrajan. Odottavia töitä seurataan kolme kertaa päivässä, jotta robottien ruuhkautuminen havaitaan aikaisin.

Oldest allowed work tarkastaa onko robotin jonoon jäänyt käsittelemättömiä vanhoja töitä. Kenttään kirjataan, kuinka monta päivää työ saa olla jonossa ennen hälytystä.

Järjestelmävirhe ja odottavan tilan hälytyksistä on myös versiot, jossa Custom time window (d) määrittää aikavälin, jolla töitä tarkastellaan. Muokattavan aikaikkunan hälytyksiin voidaan myös asettaa cooldown arvo, joka estää hälytyksien lähettämisen päivittäin. Esimerkkikuvassa muokattavan aikaikkunan hälytykset tehdään viikon välein.

Tuotannon robottien jonojen analysoinnissa käytetään Blue Prismin omaa work queues objektia. Objektilla haetaan kaikki jonon työt halutulta ajalta. Robotti kerää jonon työt talteen myöhempää raportointia varten, ja analysoi tietojen perusteella, tehdäänkö jonosta hälytyksiä. Hälytykset lähetetään sähköpostilla Excel-tiedostossa määriteltyihin sähköposteihin.



Kuva 13. Hälytys ja raportointilogiikka

Robotti luo kuvassa 14 näkyvän Excel-raportin jonon töistä. Raporttiin sisältyy Loaded count, Completed items, Business Exceptions, System Exceptions, Total Exceptions, Pending items, Deferred ja 7 days pending. Raporttia käytetään konehuoneen toimesta, joka päivän raporteista kerätään data RPA-seuranta Exceliin.

	Prosessin jonon nimi	Loaded Count	Completed Items	Business Exceptions	System Exceptions	Total Exceptions	Pending Items	Deferred	7 days pending
2	Jono 1	150	143	9	5	14	1	0	0
3	Jono 2	323	310	17	3	20	0	905	28

Kuva 14. Robotin generoima Excel-raportti

Virhetilanteissa robotti hälyttää sähköpostilla ylläpitäjälle, ja pyrkii jatkamaan seuraavaan osioon. B2B osion epäonnistuttua robotti yrittää silti ajaa B2C hälytykset. Koko robotin terminointi tapahtuu, jos sähköpostia ylläpitäjälle ei voida lähettää. Terminointi näkyy ohjauskeskuksessa ja siitä lähetetään erillinen hälytys ylläpitäjille.

Robotti kirjaa myös hälytyksien määrät ja edellisen hälytyksen päivämäärän Exceliin kuvan 15 mukaisesti. Hälytysten määrää käytetään hälytysrajojen säätämisessä, jotta robotti ei hälyttäisi liian useasti.

Prosessin jonon nimi	Notify emails	Daily SE limit	Daily pending limit	Custom time	Custom SE	Custom pending	Custom alert cooldown	Oldest allowed work	Minimum items	Rob Last alert	
RPA_343	ossi.ruhanen(	15	200	15	15	200	7	180	2	6	23.10.2021
RPA_500	ossi.ruhanen(	10	100	10	5	100	7	180	2	21	3.12.2021
RPA_246	ossi.ruhanen(	5	100	15	5	100	7	180	2	1	30.8.2021

Kuva 15. Ohjaus Excel

Robotti kirjaa hälytyksen syyn ja aikaikkunan kuvassa 16 näkyvään hälytys sähköpostiin Tämän jälkeen viestiin listataan kaikki järjestelmävirheet, töiden avain sekä virheen syy. Avain on tunniste, jolla työ löytyy robotin jonosta. Jos hälytyksistä halutaan poistaa tietyt työt, voi Exceliin kirjata avaimen arvon, jotta robotti ei hälytä tästä työstä.

System exceptionit aikavälillä: 1d ylittivät hälytysrajan!

Item Key: 1024102598
Exception reason: Customer has order in progress

Kuva 16 Sähköpostihälytys

5 Johtopäätökset ja jatkokehitysehdotukset

Työn tavoitteena oli kehittää tuotantoympäristössä toimivien robottien töiden seurantaa. Robottien ylläpito tilasi robotin, jonka tarkoitus oli parantaa robottien ylläpitoa automaattihälytyksillä ja Excel-tiedostoon kerättävällä datalla muiden robottien tekemistä töistä. Hälytyksillä pyrittiin löytämään nopeasti robottien virhetilanteet, jotta ylläpito saisi ne nopeammin korjattua. Työjonojen ajodata tallennettiin Excel-tiedostoon, jotta robottien toiminnasta saataisiin rakennettua historia, jonka avulla pystyttäisi kohdistamaan mahdollisia kehitys tai korjaus toimenpiteitä oikeisiin robotteihin tehokkaammin.

Seurantarobotin toteutus onnistui ja se otettiin hyvin vastaan. Robotti mahdollistaa nyt virheiden nopeamman havainnoinnin ja useasti ongelmat saadaan jo korjattua ennen kuin niistä tulee asiakkaalta virheilmoituksia. Tämä parantaa asiakastyytyväisyyttä ja tekee roboteista luotettavampia. Jokainen uusi robotti on lisätty tämän robotin seurantaan näiden hyötyjen takia.

Robottia on nyt käytetty noin vuoden ajan. Ongelmatilanteita robotin käynnistymisen kanssa on välillä ollut, mutta tästä ei käytännössä ole haittaa, koska robotti voidaan käynnistää ylläpidon toimesta uudelleen. Ensimmäisen kuukauden aikana löydettiin robotin logiikasta virheitä, mutta nämä saatiin nopeasti korjattua. Isoimpia ongelmia oli etenkin ohjelmointivirheet, jotka aiheuttivat virheellisiä hälytyksiä liian vanhoista töistä. Myös tietokannan eri aikavyöhyke aiheutti ongelman, jossa päivän ajodata ei vastannut samaa määrää kuin Blue Prism-ympäristön jonosta tarkastettuna. Robottia on laajennettu hälyttämään myös järjestelmävirheiden ylittäessä päivittäisen rajan ja uusia kehitysideoita on jo tilattu.

Robottia rakentaessa logiikka olisi ollut selkeämpi, jos b2b ja b2c puolet olisi rakennettu erillisiksi prosesseiksi. Prosessit jouduttiin eristämään toisistaan niin että logiikkamuutos ensimmäiseksi ajettavaan b2b puoleen ei vaikuttaisi jälkimmäiseen b2c puolen seurantaan. Prosessit eristettiin toisistaan virheenhallinnalla niin että toisen epäonnistuessa ajossa seuraava prosessi saadaan kuitenkin vielä käynnistettyä.

Jatkokehityksenä robottiin voisi lisätä jonotöiden työaika hälytykset. Jos työ jostain syystä kestää liian kauan, tästä tulisi tehdä hälytys. Sähköpostin lähetyksessä tapahtuvista virheistä voisi myös erikseen hälyttää. Näissä tapauksissa on mahdollista, että työ merkitään virheeksi ja sitä ei enää ajeta uudestaan eli työ jää tekemättä. Robottia kehitetään tulevaisuudessa ylläpidon tarpeiden mukaan. Robotille tulleet muutospyynnöt on saatu toteutettua nopeasti,

Lähteet

Blue Prism. 2021a. About our customers. Viitattu 29.10.2021. Saatavissa https://www.blueprism.com/customers/

Blue Prism. 2021b. Media center. Viitattu 29.10.2021. Saatavissa https://www.blueprism.com/resources/media-center/

Cognizant. 2022. Intelligent Automation. Viitattu 23.10.2022. Saatavissa https://www.cognizant.com/us/en/glossary/intelligent-automation

Data Matics. 2020. RPA vs. APIs – which type of integration your business needs? Viitattu 5.10.2022. Saatavissa RPA vs. APIs – which type of integration your business needs? (datamatics.com)

Enlyft. 2022. Companies using Blue Prism. Viitattu 5.10.2022. Saatavissa Companies using Blue Prism and its marketshare (enlyft.com)

Gartner. 2022. Press release. Viitattu 5.10.2022. Saatavissa Gartner Says Worldwide RPA Software Spending to Reach $2.9 Billion in 2022

Grubb, P., Armstrong, T. 2003. Software Maintenance Concepts and Practice. 2. uudistettu painos. River Edge, N.J: World Scientific

Lookfar Labs. 2022. Software Maintenance: Understanding and Estimating Costs. Viitattu 30.10.2022. Saatavissa https://www.lookfar.com/blog/2022/01/12/software-maintenance-understanding-and-estimating-costs/

Mullakaral, N. 2022. Top RPA Companies, their Industry Valuations, and Market Share. Viitattu 23.10.2022. Saatavissa https://nandan.info/top-rpa-companies-their-industry-valuations-and-market-share/

Patil, S. 2022. Top RPA automation Tools by the market shares. Viitattu 23.10.2022. Saatavissa https://www.linkedin.com/pulse/top-rpa-automation-tools-market-shares-sandeep-patil?trk=articles_directory

Tech Crunch. 2019. Blue Prism looks to partners to expand robotic process automation with AI. Viitattu 29.10.2021. Saatavissa Blue Prism looks to partners to expand robotic process automation with AI | TechCrunch

Tripathi, A. 2018. Learning Robotic Process Automation. Birmingham: Packt Publishing.