jamk

Automaatiosuunnittelun ohjelmointiympäristöjen virtualisointi

Leevi Ruotsalainen

Opinnäytetyö, AMK
Maaliskuu 2023
Sähkö- ja automaatiotekniikan tutkinto-ohjelma, Insinööri, AMK

jamk | Jyväskylän ammattikorkeakoulu
University of Applied Sciences

jamk

Kuvailulehti

Ruotsalainen, Leevi

Automaatiosuunnittelun ohjelmointiympäristöjen virtualisointi

Jyväskylä: Jyväskylän ammattikorkeakoulu. Maaliskuu 2023, 43 sivua.

Sähkö- ja automaatiotekniikan tutkinto-ohjelma. Opinnäytetyö AMK.

Julkaisun kieli: suomi

Julkaisulupa avoimessa verkossa: kyllä

Tiivistelmä

Rejlers Finland Oy:n Jyväskylän toimistolla automaatiosuunnittelussa käytetään sovellussuunnittelun tehtävissä laajalti virtuaalikoneita eri automaatiojärjestelmien suunnitteluympäristöjen eristämiseen. Virtuaalikoneiden avulla saadaan ennaltaehkäistyä eri suunnitteluohjelmistojen aiheuttamat mahdolliset ristiriidat eristämällä ne omiin ympäristöihinsä. Virtuaalikoneilla on usein suunnitteluohjelmiston lisäksi asiakkaiden käyttämiä työkaluja etäyhteyksien muodostamiseen ja tietokantojen käsittelyyn. Virtuaalikoneiden suuren koon johdosta niitä joudutaan usein siirtämään erillisille kiintolevyille, jolloin niiden olinpaikan seurannassa voi ilmetä haasteita. Opinnäytetyössä tavoitteena oli helpottaa näiden haasteiden käsittelyä ja selvittää mahdollisia vaihtoehtoja nykyiselle käytännölle.

Työn tietoperustassa käsitellään virtualisointia virtuaalikoneiden ja nousua tekevän konttiteknologian muodossa, automaatiosuunnittelun vaiheita niin yleisellä tasolla, kuin myös automaation sovellussuunnittelua käytännönläheisestä näkökulmasta. Tietoperustassa käsitellään myös logiikkaohjaimia, niiden ohjelmointia ja siihen käytettäviä työkaluja.

Työhön sisältyy teemahaastatteluja, joissa kerättiin tietoa sovellussuunnitteluprojektin käytännön vaiheista, sovellussuunnittelussa käytettävistä työkaluista ja kehitysehdotuksia nykyistä käytäntöä koskien. Näiden ehdotusten avulla saatiin työssä aikaiseksi uusi toimintatapa helpottamaan suunnitteluohjelmistojen lisenssienhallintaa. Työhön sisältyy myös selvitys Siemensin tarjoamien lisenssien käyttöehdoista, jossa tarkasteltiin käyttöehtoja väärinkäytösten tapahtumisen minimoimisen näkökannalta.

Työssä myös tarkasteltiin pilvipohjaisia vaihtoehtoja sovellussuunnittelun työkaluina sekä konttien käyttömahdollisuuksia virtuaalikoneiden korvaajana. Konttien nykyisiä käyttökohteita automaatiojärjestelmissä käsiteltiin myös.

Avainsanat (asiasanat)

Automaatiosuunnittelu, Virtualisointi, Konttiteknologia, Ohjelmoitava logiikkaohjain, Lisenssienhallinta

Muut tiedot (salassa pidettävät liitteet)

Liitteet 2 ja 3 ovat salassa pidettäviä, ja ne on poistettu julkisesta työstä. Salassapidon peruste on Julkisuuslain 621/1999 24§, kohdat 7 ja 21. Salassapitoaika on viisi (5) vuotta, salassapito päättyy 18.3.2028.

jamk

Description

Ruotsalainen, Leevi

Virtualization of automation design programming environments

Jyväskylä: JAMK University of Applied Sciences, March 2023, 43 pages.

Degree Programme in Electrical and Automation Technology. Bachelor´s thesis.

Permission for open access publication: Yes

Language of publication: Finnish

Abstract

In the automation design department of Rejlers Finland's Jyväskylä office virtual machines are widely used for isolating the design environments of different automation systems. With virtual machines it's possible to prevent conflicts between different designing programs by isolating them into their own environments. Besides the designing programs the virtual machines often include tools used by the client for forming remote connections and accessing databases. Because of the large sizes of the virtual machines, they often must be moved to separate hard disks making it sometimes difficult to keep track of the virtual machines. The target of the thesis was to make it easier to handle these challenges and to research possible options for the current method.

The thesis' knowledge base covers virtualization in the forms of virtual machines and container technology. It also covers programmable logical controllers and different tools used with them as well as automation designing in a general manner and application development with a practical viewpoint.

The thesis includes thematic interviews which were used to collect information about the practical steps of an automation designing project, the tools used in application development and suggestions for improving the current method for handling the virtual machines. With the suggestions received from these interviews there was a solution implemented to ease license management. The thesis also includes a report made from the licensing terms and conditions of Siemens' PLC programming tools for minimizing the cases of wrongful usage of these licenses.

Also cloud-based options for application development as well as possibilities for the usage of containers as a replacement for virtual machines were observed in the thesis. Current usages for containers in automation technology were covered as well.

Keywords/tags (subjects)

Automation design, Virtualization, Container technology, Programmable logical controllers,
License management

Miscellaneous (Confidential information)

Attachments 2 and 3 are confidential and have been removed from the public release. Grounds for confidentiality are defined in the act on the openness of government activities 621/1999 24§, parts 7 and 21. Confidentiality period is five (5) years, confidentiality will end on 18.3.2023.

Sisältö

Kuviot

1 Johdanto

1.1 Opinnäytetyön lähtökohdat

Tietokoneen resurssien virtualisointia on käytetty vuosikymmeniä hyväksi virtuaalikoneiden muodossa palvelimissa sekä PC:issä sellaisissa kohteissa, joissa halutaan joustavuutta resursseilta, ja joissa halutaan eristää sovelluksia niiden omaan ympäristöön. Lähivuosina konttiteknologia on alkanut korvaamaan perinteisiä virtuaalikoneita palvelintoimintojen saralla, mutta virtuaalikoneet ovat edelleen laajassa käytössä virtualisoinnin maailmassa.

Automaatiosuunnittelussa virtuaalikoneita hyödynnetään, kun halutaan estää sovellussuunnittelussa käytettävien työkalujen keskenään aiheuttamia ristiriitoja. Automaatiosuunnittelutoimistossa toimitaan monien eri valmistajien laitteistojen sekä ohjelmistojen kanssa, josta syntyy tarve usean virtuaalikoneen luontiin. Tässä opinnäytetyössä tullaan muun muassa selvittämään syitä, miksi automaation sovellussuunnittelussa käytetään niin laajalti virtuaalikoneita sen sijaan, että kyseiset virtuaalikoneet korvattaisiin konteilla, sekä mitä hyötyjä konttien käyttämisestä virtuaalikoneiden seuraajana olisi.

Opinnäytetyön toimeksiantajana toimii Rejlers Finland Oy. Toimeksiantaja halusi selkeyttää virtuaalikoneiden ja lisenssien hallintaa työskentely-ympäristössä, jossa käytetään useita eri suunnitteluohjelmistoja päivittäin. Tästä voi syntyä hankaluuksia eri ohjelmien lisenssien olinpaikan seuraamisesta. Valmistajat ovat määritelleet lisensseidensä käyttöön ehtoja, joiden kanssa on pysyttävä linjassa, kun otetaan käyttöön uusia virtuaalikoneita suunnittelukäyttöön ja tämän takia on tärkeää, ettei väärinkäyttöä tapahdu.

1.2 Opinnäytetyön tavoitteet ja rajaus

Opinnäytetyön aihe syntyi alun perin tarpeesta saada käytössä oleville virtuaalikoneille ja niiden sisältämille lisensseille keskitetty hallintamenetelmä korvaamaan edellinen menetelmä, jossa virtuaalikoneita oli tallessa vain henkilökohtaisilla tietokoneilla ja kovalevyillä, ja lisenssejä irrallisilla lisenssitikuilla. Tätä aihepiiriä lähdettiin laajentamaan ja päädyttiin tekemään myös selvitykset vir-

tuaalikoneiden suuren määrän aiheuttamien ongelmien mahdollisesta vähentymisestä tulevaisuudessa, sekä lisenssien käyttöehtoihin liittyvistä kysymyksistä, joita esiintyy virtuaalikoneiden siirtelyn seurauksena.

Opinnäytetyön tavoitteena on saada toimeksiantajalle toimiva ympäristö helpottamaan edellisessä kappaleessa mainittuja haasteita. Tämä uusi ympäristö tulee sujuvoittamaan uuden projektin aloitusta, kun otetaan käyttöön tarvittavat ohjelmat sisältävää virtuaalikonetta. Ympäristön valinta ja toteutus on toimeksiantajan vaatimuksesta salassa pidettävää materiaalia.
Kehitysehdotusten ja mahdollisten vaihtoehtojen kartoittamista varten tullaan opinnäytetyössä suorittamaan kaksi teemahaastattelua.

Tavoitteena on myös saada aikaiseksi kattava selvitys lisenssien käyttöön liittyvistä ehdoista, sillä suuressa yrityksessä on tärkeää, että lisenssien käytön ehdot ja rajoitteet ovat selkeät, jotta väärinkäytöksiltä vältytään. Tämän lisäksi opinnäytetyössä tullaan tekemään selvitys, mitä hyötyjä konttien käytöstä virtuaalikoneiden sijaan olisi, miksi kontteja ei käytetä automaatiosuunnittelussa, ja olisiko niiden käyttö mahdollista tulevaisuudessa. Konttien käyttö on yleistymässä nopeaa vauhtia, ja uskon että niitä tullaan pian näkemään enemmän palvelinten ulkopuolellakin.

Tutkimus- ja kehittämiskysymyksiksi muodostui siis:

- Kuinka voidaan suoraviivaistaa virtuaalikoneiden ja lisenssien hallinnoimista?
- Kuinka voidaan ennaltaehkäistä ohjelmistolisenssien väärinkäytöksiä?
- Kuinka konttiteknologiaa hyödynnetään automaatiossa ja mitä käyttömahdollisuuksia tulevaisuudessa voisi olla?

1.3 Aikaisemmat tutkimukset

Virtuaalikoneiden ollessa melko vanhaa tekniikkaa, on niistä paljon materiaalia kirjoissa, opinnäytetöissä ja erilaisten toteutusten dokumentoinneissa. Käytännön elämässä on myös paljon erilaisia käyttökohteita, joista ei yhtä paljon ole materiaalia kirjoitettu, kuten automaatiosuunnittelun työkaluna. Yhtenä hyvänä ja luotettavana tiedonlähteenä käytin Matthew Portnoyn vuoden 2016 kir-

jaa ”Virtualization essentials”. Kyseisessä kirjassa on käyty kattavasti läpi virtualisoinnin määritelmää, mistä tarpeesta virtualisointi on saanut alkunsa sekä missä ja miksi sitä nykyaikana käytetään.

Konttiteknologia taas on huomattavasti uudempi käsite. Kontit hyödyntävät virtualisointia kuten virtuaalikoneetkin, mutta huomattavasti eri tavalla. Konttiteknologiastakin on paljon julkaisuja ja aihealueen ollessa niin tuore, ovat julkaisutkin uusia. Toisaalta, koska aihepiiri on niin uusi, keskittyvät julkaisut enimmäkseen siihen, kuinka konttiteknologiaa voidaan hyödyntää palvelintekniikassa virtuaalikoneiden korvaajana. Perinteisempien työpöytäsovellusten virtualisointiin kohdistuvia julkaisuja ei täten ole yhtä paljon.

Automaation sovellussuunnittelusta on tehty paljon opinnäytetöitä, sekä joitakin julkaisuja, mutta opinnäytetöissä harvoin käsitellään projektin tekemiseen liittyviä eri käytännön vaiheita. Useimmiten julkaisuissa esitellään lähinnä itse toteutusta. Laajemmissa suunnittelua yleisemmällä tasolla käsittelevissä julkaisuissa taas ei käytäntöön liittyviä asioita käydä itse sovellussuunnittelijan näkökulmasta. Tämän takia koettiin, että paras mahdollinen lähde sovellussuunnittelusta on itse sovellussuunnittelija.

1.4 Tutkimusote

Opinnäytetyöhön sisältyy sekä konkreettista toteutusta, että selvitystyötyyppistä aineistoa, joten opinnäytetyössä ei tulla pysymään tiukasti yhdessä tietyssä tutkimusmenetelmässä. Opinnäytetyön konkreettinen osuus, jossa luodaan uusi ympäristö helpottamaan uuden ohjelmointiympäristön käyttöönottoa, tulee muistuttamaan kehittämistutkimusta. Kehittämistutkimuksen tavoin alussa on epäoptimaalinen ratkaisu, jota halutaan muuttaa selventämällä ensiksi itse ongelmaa ja tullaan selvittämään vaihtoehtoja sen korjaamiseksi. Kun päädytään parhaaksi katsottuun ratkaisuun, toteutetaan tämä ja arvioidaan lopputulos. Opinnäytetyön aikataulun vuoksi aika riittää vain yhteen tutkimus- ja muutossykliin. Tulen käyttämään opinnäytetyössäni enimmäkseen kvalitatiivista tutkimusmenetelmää teemahaastatteluiden muodossa. Teemahaastatteluissa pääteemoina tulevat olemaan sovellussuunnittelun eri vaiheet ja työkalut automaatioprojektissa sekä nykyiset toimintatavat projektin alkaessa ja kehitysehdotukset sen suhteen.

1.5 Rejlers Finland Oy

Rejlers on vuonna 1942 perustettu insinööriosaamiseen perustuva konsultointia, suunnittelua ja projektiratkaisuja tarjoava ruotsalainen Tukholman pörssiin listattu yhtiö. Rejlers toimii Suomessa, Ruotsissa, Norjassa sekä Yhdistyneissä Arabiemiirikunnissa, ja on jaettu teollisuuden, rakentamisen sekä energian ja infran toimialoihin. Suomessa työntekijöitä on noin 1000 yli kahdellakymmenellä paikkakunnalla. (Meistä n.d.)

Teollisuuden suunnittelupalvelut Rejlersillä on jaettu prosessi-, tehdas-, rakenne-, kone-, sähkö- ja automaatiosuunnitteluun (Suunnittelupalvelut n.d.). Teollisuuden alan projekteja Rejlersillä on esimerkiksi Borealis Polymersillä HAZOP-tarkastelujen parissa sekä Aurora Kilpilahdelle EPC-toimituksena (engineering, procurement and construction) tehty muuntamorakennus, jossa Rejlers toimi päätoteuttajana ja -suunnittelijana (Projektimme n.d.).

Rejlersin konsernitason tavoite on vuoteen 2025 mennessä saavuttaa 10 % vuosittainen kasvu sekä 10 % EBITA marginaali (Tavoite ja strategia n.d.). Vuoden 2022 kolmannella neljänneksellä Rejlersin EBITA marginaali oli 7.0 % ja orgaaninen myynnin kasvu 13.1 % (Investors n.d.).

2 Kehittämistutkimus

Kun halutaan parantaa jotain olemassa olevaa asiaa tai toimintaa, puhutaan kehittämisestä. Kun kehittämistyö halutaan tuoda julki, voidaan tehdä kehittämistutkimus (Kananen 2015, 33). Kanasen (2015, 33) mukaan Edelson (2002) määrittelee kehittämistutkimuksen sykliseksi prosessiksi, jossa yhdistyvät tutkimus sekä kehittäminen. Kehittämistutkimus lukeutuu yhdistelmätutkimuksiin, koska siinä voidaan hyödyntää sekä laadullista, että määrällistä tutkimusta (Kananen 2015, 39).

Kehittämistutkimuksen sykli perustuu tutkimussykliin sekä muutossykliin. Tutkimussyklissä pääpiirteiltään kartoitetaan ongelmaa, sen syitä ja vertaillaan mahdollisia ratkaisuja, joista vertailun perusteella valitaan yksi. Tämä ratkaisu puolestaan toteutetaan muutossyklissä ja toteutuksen jälkeen ratkaisun onnistuneisuus arvioidaan ja sitä jäädään seuraamaan. Mikäli ratkaisu ei tuottanut haluttuja tuloksia palataan kehittämistutkimuksen syklissä takaisin alkuun. Tätä prosessia toistetaan niin kauan, että lopputulos on onnistunut. (Kananen 2015, 40-42.)

2.1 Kvalitatiivinen ja kvantitatiivinen tutkimus

Kvalitatiivisessa eli laadullisessa tutkimuksessa käsitellään aineistonhankintaa ja -tulkintaa yksittäisen asian tai ilmiön ominaisuuksien ymmärtämisen kannalta. Kvalitatiivisen tutkimuksen aineistonkeruulle ominaisia menetelmiä ovat henkilöiden teemahaastattelut sekä havainnointi. (Kananen 2015, 34-35.)

Kvantitatiivisessa tutkimuksessa taas luotetaan suureen otantaan siten, että annetaan tutkittavalle ryhmälle ennalta määritellyt vaihtoehdot. Kvantitatiivisen tutkimuksen aineistosta käsitellään lukuja sekä vastausten jakaumia sen sijaan, että kysyttäisiin epämääräisiä kysymyksiä, kuten kvalitatiivisessa tutkimuksessa. (Kananen 2015, 38)

3 Virtualisointi

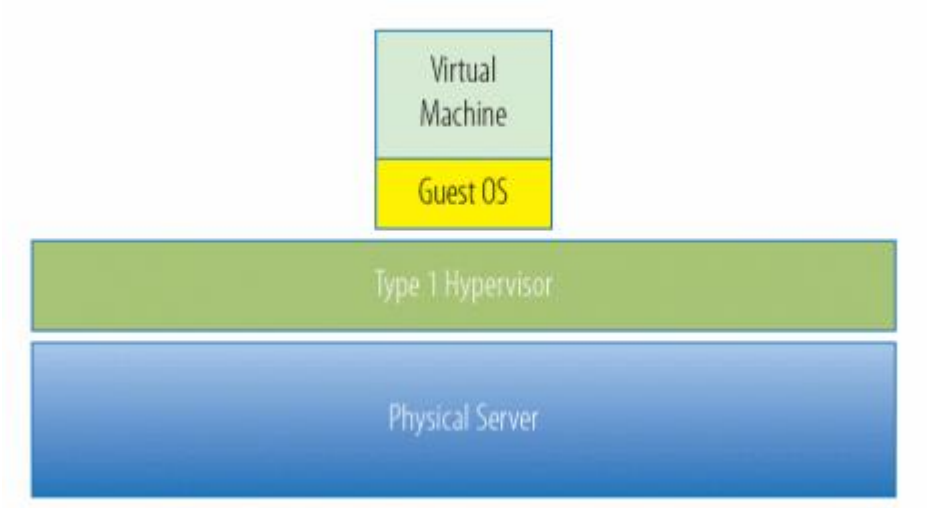
Virtualisoinnilla tarkoitetaan sitä, kun jokin fyysinen asia tai kokemus pelkistetään tai ”abstrahoidaan” todellisuudesta ja muutetaan muotoon, joka ei todellisuudessa ole olemassa, mutta asian tai kokemuksen hyödyntäjä kokee sen todellisena. Esimerkkinä ”virtuaalinen todellisuus”, jossa hyödyntäjä tai käyttäjä todellisuudessa on fyysisesti yhteydessä esimerkiksi vain ohjaimiin, ja visuaalisen ja fyysisen palautteen kautta kokee olevansa jossain muussa tilanteessa, kuin oikeasti on. Arkipäiväisempänä esimerkkinä mainittakoon videopelit ja verkkokaupat. Videopeleissä käyttäjä kokee saavuttavansa tai luovansa asioita, joita ei fyysisessä todellisuudessa ole olemassa. Verkkokaupoissa taas pystytään tekemään ostoksia, kuten normaalissa fyysisessäkin kaupassa. (Portnoy 2016, 1-2.)

Tietotekniikassa virtualisoinnilla tarkoitetaan yleensä sitä, kun jokin fyysinen komponentti abstrahoidaan loogiseksi asiaksi digitaaliseen muotoon (Portnoy 2016, 2). Esimerkiksi kokonainen tietokone pystytään muuttamaan virtuaaliseksi, jolloin virtuaalitietokoneen näkökulmasta sillä on esim. oma prosessori, tallennustila ja verkkorajapinnat, vaikka todellisuudessa virtuaalitietokone käyttää isäntätietokoneen resursseja ja rajapintoja.

3.1 Virtuaalitietokone

Virtuaalitietokone on tietokone, jolla on oma käyttöjärjestelmänsä ja fyysiseltä laitteelta määritellyt resurssit eli prosessorin ytimet, välimuisti, tallennustila sekä verkkorajapinnat ja kaikki muu, mitä normaalissakin tietokoneessa on. Normaalista tietokoneesta poiketen virtuaalitietokone on siis olemassa vain fyysisen tietokoneen sisällä, jolta se lainaa resursseja tarpeen mukaan. Virtuaalitietokoneen luontiin ja ajamiseen käytettävää ohjelmistokerrosta kutsutaan nimellä Virtual Machine Monitor (VMM) tai Hypervisor. Hypervisorin tehtävänä on jakaa isäntätietokoneen resursseja sen alaisuudessa toimiville virtuaalitietokoneille ja hoitaa virtuaalitietokoneiden kommunikointia muiden laitteiden ja isäntätietokoneen kanssa. (Portnoy 2016, 2.)

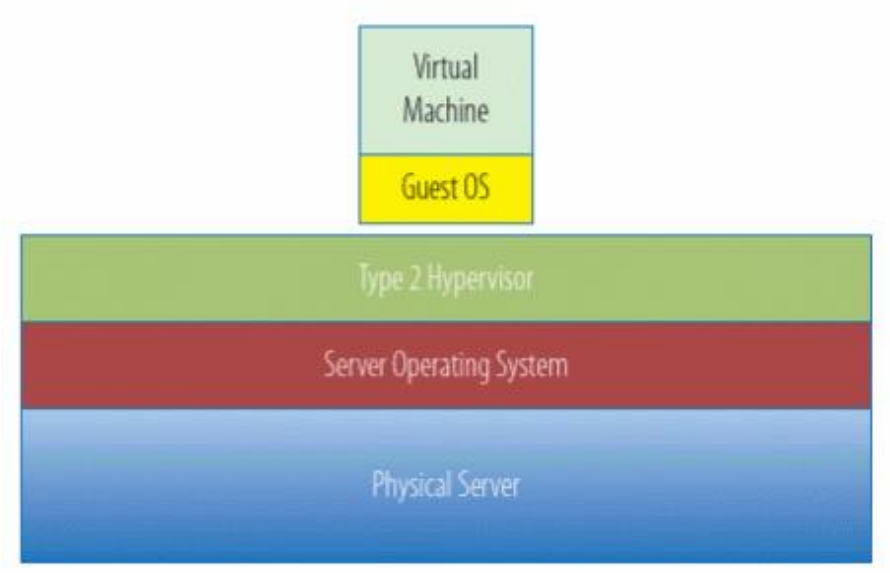
Hypervisorit on jaettu kahteen eri luokkaan; tyypin 1 Hypervisorit ja tyypin 2 Hypervisorit. Erona on se, missä topologian kohdassa Hypervisor suorittaa tehtäviään. Tyypin 1 Hypervisor sijoittuu suoraan fyysisen laitteiston ja virtuaalitietokoneiden väliin, kuten kuviossa 1 on kuvattu. Koska tyypin 1 Hypervisor ohjaa virtuaalitietokoneen pyynnöt suoraan isäntälaitteistolle ilman, että välissä on toista käyttöjärjestelmää, on se huomattavasti tehokkaampi kuin tyypin 2 Hypervisor. (Portnoy 2016, 23-24.)



Kuvio 1. Tyypin 1 Hypervisor (Portnoy 2016, 24.)

Tyypin 2 Hypervisor taas sijoittuu isäntäkäyttöjärjestelmän ja virtuaalitietokoneiden välille, eli isäntälaitteelle on asennettu oma käyttöjärjestelmä, jonka päällä Hypervisor toimii. Tämä tarkoittaa

myös sitä, että vieraslaitteen pyynnöt isännälle menevät virtuaalikoneen oman käyttöjärjestelmän ja Hypervisorin lisäksi myös isäntäkäyttöjärjestelmän läpi, kuten kuviossa 2 on kuvattu. (Portnoy 2016, 24-25.)



Kuvio 2. Tyypin 2 Hypervisor (Portnoy 2016, 25.)

Paremman suorituskyvyn lisäksi tyypin 1 Hypervisorit ovat turvallisempia ja luotettavampia kuin tyypin 2. Koska tyypissä 1 pyynnöt menevät suoraan isännälle, virtuaalikoneet eivät pysty aiheuttamaan haittaa muualle, kuin itseensä, jolloin muut virtuaalikoneet jatkavat normaalia toimintaansa vain haitallisen virtuaalikoneen kaatuessa. Tyypissä 2 isännän käyttöjärjestelmään kohdistuvat ongelmat vaikuttavat kaikkiin käytössä oleviin virtuaalikoneisiin. Esimerkiksi, kun isäntätietokoneen käyttöjärjestelmä vaatii päivityksen, ovat kaikki virtuaalikoneet alhaalla sen ajan. (Portnoy 2016, 24-25.)

Esimerkkejä tyypin 1 Hypervisoreista on VMware ESX, Microsoft Hyper-V ja Citrix Xen -variantit. Tyypin 2 Hypervisoreita ovat VMWare Player, VMware Workstation ja Microsoft Virtual Server. (Portnoy 2016 24-25.)

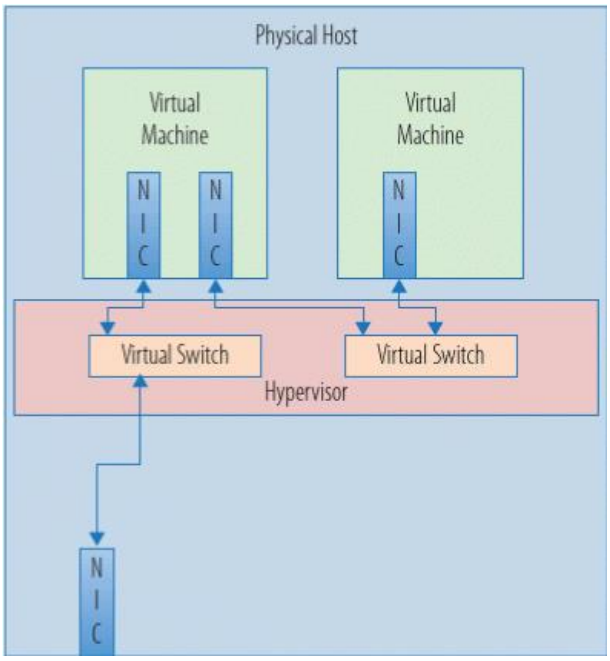
3.1.1 Virtuaalitietokoneen resurssit

Virtuaalikone koostuu pääosin kahdesta tiedostosta: konfiguraatiotiedostosta ja virtuaalilevystä. Konfiguraatiotiedostossa määritellään virtuaalikoneen käytössä olevat resurssit ja tämä antaa virtuaalikoneelle kuvan, minkälaista laitteistoa se kuvittelee käyttävänsä. Virtuaalilevy taas toimii virtuaalikoneelle tallennustilana samalla tavalla, kuten normaalillakin tietokoneella on tallennustila, joka sisältää käyttöjärjestelmän, ohjelmat, tiedostot jne. Virtuaalikoneella siis ei ole olemassa omaa fyysistä tallennustilaansa, vaan konfiguraatiossa on varattu isäntälaitteistolta tila, jota virtuaalikone saa käyttää. (Portnoy 2016, 37.)

Prosessorin osalta virtuaalikoneelle määritellään, kuinka monelle prosessorin ytimelle virtuaalikone saa enintään ajoittaa ytimen toimintasyklejä. Isäntäkone ei siis varaa virtuaalikoneelle ytimiä kokonaan, vaan virtuaalikoneen pyynnöstä antaa sen käyttää määriteltyjä ytimiä. Jos virtuaalikone ei tee pyyntöjä Hypervisorin kautta, pysyy ydin kokonaan isännän käytössä. Keskusmuistin (RAM, Random Access Memory) osalta virtuaalikoneelle annetaan megatavuissa määrä, jota virtuaalikone saa enimmillään hyödyntää isäntäkoneelta. Välimuistissakin virtuaalikoneelle annetaan välimuistia käyttöön vain tarvittaessa. (Portnoy 2016, 40-41.)

3.1.2 Virtuaalitietokoneen verkkorajapinnat

Jotta virtuaalikone pystyisi kommunikoimaan ulkomaailman kanssa, täytyy sille luoda virtuaalinen verkkokortti, joka kuvastaa yhteyttä verkkoon. Tämä verkkokortti ei ole suorassa yhteydessä fyysiseen laitteistoon, vaan Hypervisorilla täytyy olla virtuaalinen kytkin, joka taas on yhteydessä isäntälaitteiston fyysiseen verkkokorttiin. Näiden virtuaalisten verkkokorttien kautta virtuaalikone pystyy kommunikoimaan myös toisen samalla isäntälaitteella olevan virtuaalikoneen kanssa ilman fyysistä yhteyttä, kuten kuviossa 3 on kuvattu. (Portnoy 2016, 42.)



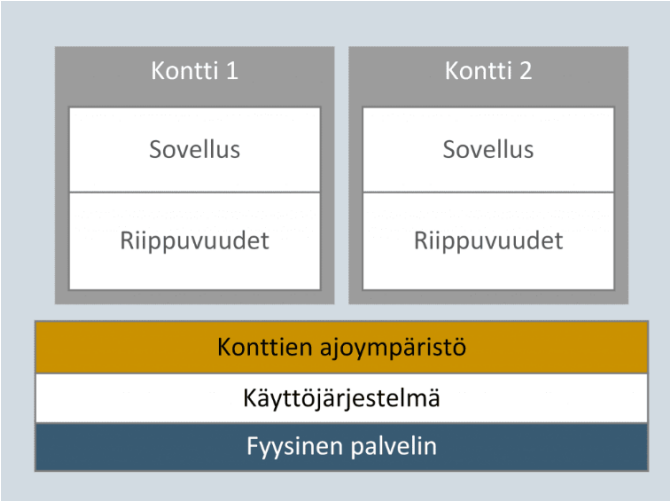
Kuvio 3. Palvelimen esimerkkitopologia (Portnoy 2016, 42.)

Virtuaaliverkko on mahdollista toteuttaa myös siten, että yksi virtuaalikone on yhteydessä ulkomaailmaan vain toisen virtuaalikoneen kautta. Näin on mahdollista saada yksi kerros lisää verkon tietoturvallisuuteen. Kuviossa 3 on kuvattu verkkoa, jossa isäntälaitteella sijaitsee 2 virtuaalikonetta ja Hypervisorilla on kaksi virtuaalista kytkintä. Näistä kytkimistä vasemmanpuoleinen kommunikoi yhden virtuaalikoneen välillä ja oikeanpuoleinen välittää virtuaalikoneiden välistä kommunikointia. Kuviossa verkkokortteja kuvastavat NIC:t (Network Interface Card), joista virtuaalikoneiden sisällä sijaitsevat ovat virtuaalisia ja alimpana sijaitseva on fyysinen verkkokortti. Kuviosta huomataan, että oikeanpuoleisen virtuaalikoneen kommunikointi ulkomaailmaan kulkee vasemmanpuoleisen virtuaalikoneen kautta. (Portnoy 2016, 42.)

3.2 Konttiteknologia

Kuten edellisessä osassa mainittiin, virtuaalitietokone vaatii oman kokonaisen käyttöjärjestelmänsä, jotta sillä pystytään ajamaan haluttuja ohjelmia. Käyttöjärjestelmä sisältää ohjelmien tarvitsemia välttämättömiä riippuvuuksia, mutta käyttöjärjestelmässä on myös paljon tavaraa ja taustalla toimivia prosesseja, joita käytettävä ohjelma ei tarvitse. Kontit (eng. containers) sen sijaan voivat sisältää vain hyvin kevyen käyttöjärjestelmän, joka sisältää vain ohjelman tarvitsemat riippuvuudet, jolloin kontti-imagen koko voi Walleniuksen (2022) mukaan olla sadasosan vastaavan

virtuaalipalvelimen koosta. Virtuaalikoneessa oleva käyttöjärjestelmä myös kuluttaa isäntäkoneen resursseja tarpeettomasti, eli mikäli isäntäkoneella käytetään useaa virtualisointia hyödyntävää sovellusta, kertautuvat nämä hyödyt suhteessa sovellusten määrään. (Wallenius 2022.)



Kuvio 4 Konttiteknologiaa hyödyntävän palvelimen rakenne. (Wallenius 2022.)

Samalla tavalla, kuin virtuaalikoneesta voidaan luoda kopioita, jotka sisältävät samat ominaisuudet, kuin alkuperäinenkin ja joita voidaan käyttää samaan aikaan, voidaan kontti-imagesta luoda useita samaan aikaan toimivia versioita. Kontti-image sisältää tiedot, joiden pohjalta käynnistetään itse kontit, jotka ovat kopioita alkuperäisestä imagesta ja tämän jälkeen elävät omaa elämäänsä. Kun kontti on tehnyt tehtävänsä, voidaan kontti sulkea ja poistaa isäntäkoneelta, jolloin jäljelle jää vain alkuperäinen image ja mahdollisesti muut käynnissä olevat kontit. Virtuaalikoneiden kopioillakin on oma virtuaalinen laitteistonsa ja käyttöjärjestelmänsä, jolloin ne tarvitsevat isäntäkoneelta sekä enemmän tilaa ja resursseja, että enemmän aikaa niiden luomiseen. (Wallenius 2022; Stoneman 2021)

Jotta kontteja pystytään rakentamaan ja käyttämään, tarvitaan niille ympäristö, jossa ne toimivat, kuten kuviossa 4 on kuvattu. Näihin tarkoituksiin on useita tarjolla olevia työkaluja, joista osa on kokonaisuuksia, jotka sisältävät useita kontteihin liittyviä toimintoja ja toiset työkalut hoitavat vain tietyn osa-alueen. Näistä tunnetuin on Docker (Kisller 2023). Dockerilla pystyy sekä luomaan konteja, että käyttämään niitä. Dockerilla on myös laaja rekisteri, joka sisältää sekä virallisia kontti-imageja ohjelmavalmistajilta ja avoimen lähdekoodin projekteilta, että yksityisiltä käyttäjiltä. Näitä

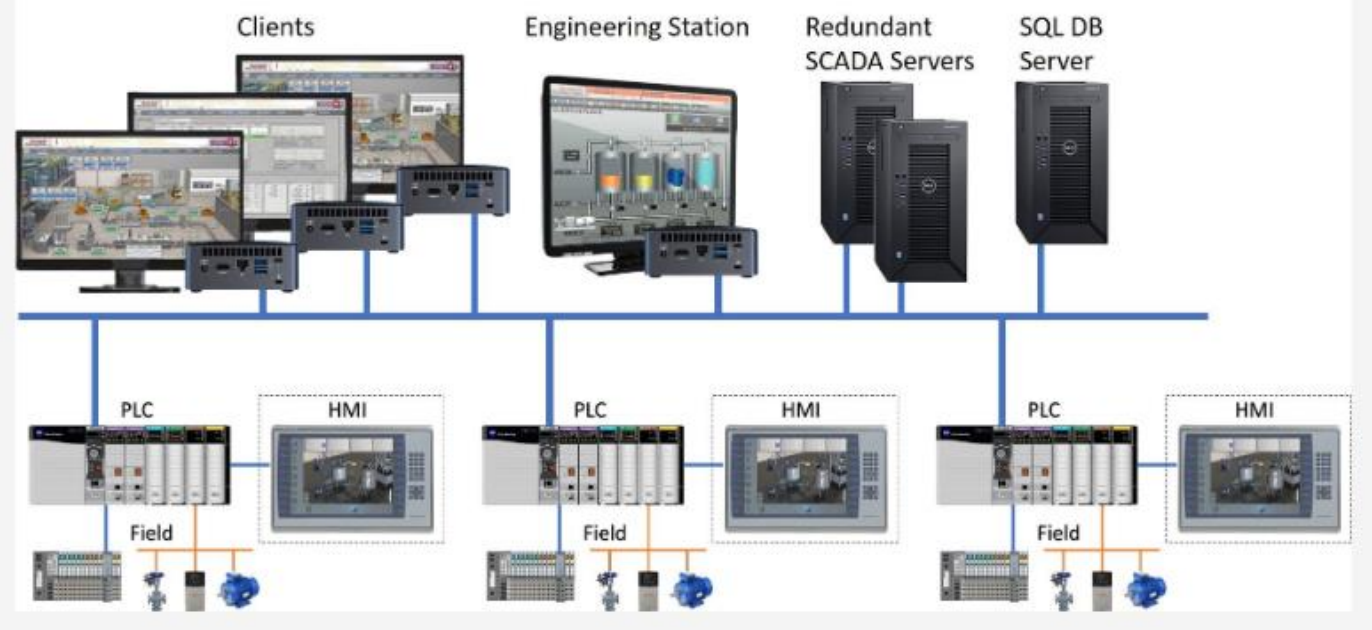
imageja voi käyttää pohjina, joilla esimerkiksi pystyy ylläpitämään SQL- tai verkkopalvelinta. Pohjana voi käyttää myös esimerkiksi pelkkää käyttöjärjestelmää, johon käyttäjä asentaa tarvitsemansa ohjelmat. Dockerilla pystyy käyttämään muitakin rekistereitä, jotka voivat olla samanlaisia, kuin Docker Hub tai vaikka yrityksen oma rekisteri. (Stoneman 2021.)

4 Ohjelmoitava logiikkaohjain

Ohjelmoitava logiikkaohjain tai PLC (Programmable Logic Controller) on yksinkertainen tietokone, jolla pystytään automatisoimaan laitteiden toimintoja, prosessin ohjausta, tai tuotantolinjaa tai sen tiettyä osaa. PLC pystyy tyypistä, ominaisuuksista ja järjestelmän muista laitteista riippuen esimerkiksi lukemaan ja kirjoittamaan digitaalisia ja analogisia tuloja ja lähtöjä. PLC tulkitsee kenttälaitteiden signaaleja tulokorttien kautta, ja nämä luettuaan se käy käyttäjän tekemän ohjelman läpi ja lähettää kentälle signaalin toimilaitteelle lähtökortin kautta. Toimilaite vaikuttaa käytössä olevaan prosessiin ja erilaisten antureiden kautta PLC saa taas uutta tietoa prosessista takaisinkytkentänä. (Bolton 2009, 3-5.)

4.1 HMI ja SCADA

Jotta käyttäjät pystyisivät vuorovaikuttamaan PLC:n ohjelman kanssa, tarvitaan yksinkertaisimmillaan fyysisiä painonappeja, kytkimiä tai potentiometrejä, jotka yhdistyvät PLC:n tulokortteihin. Jos taas halutaan tarkempaa tietoa prosessin kulusta, tarvitaan HMI (Human-Machine Interface), suomeksi käyttöliittymäsovellus tai ihmisen ja koneen välinen rajapinta. Tämä voi olla kentällä sijaitseva näyttö fyysisillä napeilla, kosketusnäyttö, tai tietokoneella oleva ikkuna. HMI keskustelee PLC:n kanssa jonkin käytössä olevan väyläteknologian kautta. PLC:n laitteistomäärittelyssä on määritelty käytettävä väylä ja toisessa päässä oleva laite sekä näiden ominaisuudet, jonka avulla PLC osaa tulkita väylän kautta tulevia viestejä. (What is HMI? 2018.)



Kuvio 5 Esimerkkitopologia SCADA-järjestelmästä. (What is SCADA? 2021.)

HMI:n läheinen sukulainen on SCADA (Supervisory Control And Data Acquisition). Suomeksi SCADA:sta voidaan puhua nimellä valvomo-ohjelmisto tai PC-valvomo. Tärkein ero HMI:n ja SCADA:n välillä tulee luetun datan tallentamisessa. HMI:n kyky tallentaa dataa on hyvin rajallinen, kun taas SCADA pystyy koordinoimaan erilaisen laitteiston kanssa siten, että PLC:n keräämää dataa pystytään tallentamaan tietokantoihin ja sieltä käyttämään esimerkiksi jonkin mittauksen historiakäyrän piirtämiseen. Useimmat SCADA-järjestelmät pystyvät kommunikoimaan OPC-standardin välityksellä, jolloin kommunikointi lähes kaikkien PLC-valmistajien logiikkaohjainten kanssa on mahdollista. (What is SCADA? 2021.)

4.2 Ohjelmointiympäristöt

Ohjelmoitavien logiikoiden ohjelmointi suoritetaan useimmiten valmistajakohtaisella ohjelmalla, jonka avulla käyttäjän luoma ohjelma voidaan myös ladata logiikan muistiin sen suoritettavaksi (Bolton 2009, 15). Nykyisin ohjelmointiympäristöissä pystyy tekemään muitakin asioita, kuin luomaan logiikkaohjelmia, kuten tekemään HMI-paneelin määrittelyt ja käyttöliittymät. Esimerkiksi Siemensin TIA Portalissa on mahdollista konfiguroida SINAMICS-taajuusmuuttajakäyttöjen parametrit samassa ohjelmointiympäristössä. (TIA Portal drive integration with SINAMICS Startdrive n.d.)

Vaikka eri valmistajien ohjelmien käyttöliittymät eroavat toisistaan jonkin verran, käyttävät ne laajalti IEC 61131-3-standardin määrittelemiä ohjelmointikieliä (Bolton 2009, 14-15.). IEC 61131-3 määrittelee viisi ohjelmoitavissa logiikoissa käytettävää kieltä; LAD (ladder diagram), SFC (sequential function chart), FBD (function block diagram), ST (structured text) ja IL (instruction list) (Bolton 2009, 15.). Näistä ensimmäisenä kehitettiin LAD, koska logiikoiden ohjelmoinnin suorittivat sähköasentajat, joilla ei aikaisempaa kokemusta ohjelmoinnista ollut. LAD luotiin muistuttamaan normaaleja kytkentäkuvia, sillä aikaisemmin loogiset ohjelmoinnit toteutettiin fyysisillä relekytkennöillä, joita sähköasentajat olivat tottuneet lukemaan. (Bolton 2009, 14.) Toinen graafinen ohjelmointikieli on FBD. FBD:ssä käytetään ”kytkinten” sijaan funktiolohkoja, jotka suorittavat erilaisia loogisia tehtäviä. (Bolton 2009, 126.)

IL on hyvin pelkistetty tekstipohjainen ohjelmointikieli, jota voisi kuvailla tekstipohjaista LAD:tä vastaavaksi. IL:ssä annetaan erilaisilla operaattoreilla ohjeita, joiden mukaan ohjelman muuttujia käsitellään. (Bolton 2009, 147-149.) IL on määritelty IEC 61131-3-standardissa, mutta sitä ei enää ylläpidetä (Status IEC 61131-3 standard n.d.). Esimerkiksi Siemens kuitenkin vielä sisällyttää sen kielivaihtoehtona TIA Portalissa. Valmistajien välillä kielen syntaksissa on pieniä eroja, mutta kyseessä on toiminnallisuudeltaan sama kieli. Kuten IL, ST eli Structured Text on myös tekstipohjainen ohjelmointikieli. Siinä missä IL on helpompi ajatella LAD:tä muistuttavana kielenä, ST muistuttaa enemmän tavallista ohjelmointikieltä, kuten Pascalia. (Bolton 2009, 160.)

SFC (Sequential Function Chart) on graafinen ohjelmointikieli, joka muistuttaa ulkomuodoltaan askeleittain etenevää prosessia. SFC:ssä lähdetään liikkeelle aloitusaskeleesta, jossa ympäristö on normaalitilassa, ja josta siirrytään seuraavaan askeleeseen yhden tai useamman ehdon täytyttyä, tämä ehto voi olla esimerkiksi painonapin painallus tai ajastimen täyttyminen. Seuraavassa askeleessa taas ohjelma tekee, mitä sen halutaan siinä askeleessa tekevän, jonka jälkeen odotetaan taas seuraavan ehdon täyttymistä ja siirrytään seuraavaan askeleeseen, lopetetaan sekvenssi tai siirrytään takaisin määriteltyyn kohtaan ohjelmassa. (Bolton 2009, 156-157.)

4.3 Lisenssit

Ohjelmistolisenssi on asiakirja, joka määrittelee lailliset ohjeet ohjelmiston käytölle ja levittämiselle. Tyypillisesti ohjelmalisenssi antaa sen käyttäjälle oikeuden yhteen tai useampaan ohjelmakopioon siten, että se ei riko tekijänoikeuksia. Lisenssissä myös määritellään sopimuksen osapuolien vastuut ja se voi myös asettaa rajoituksia ohjelmiston käytölle. Lisensointiehdot yleensä määrittelevät kohtuullisen käytön, takuut, vastuuvapauslausunnot sekä vastuunrajoitukset. Ohjelmistolisenssit voivat esimerkiksi sisältää määrittelyt ohjelmiston latauskerroista, ohjelmiston hinnasta sekä siitä, minkä tasoinen pääsy ohjelmiston lähdekoodiin käyttäjillä tulee olemaan. Lisenssit voidaan tyypillisesti määritellä joko sopimuksella loppukäyttäjän kanssa tai sopimuksella yrityksen kanssa ja käyttäjän on hyväksyttävä ohjelmiston käyttöehdot ohjelmistoa hankittaessa. Ohjelmisto sisältää lisenssiavaimen, jota käytetään ohjelmiston version tunnistamiseen, varmentamiseen ja sen aktivointiin. (Lutkevich 2021.)

4.3.1 Lisenssityypit

Ohjelmistolisenssit voidaan pääpiirteiltään jakaa kahteen lajiin; ilmaisiin avoimen lähdekoodin ohjelmistoihin (eng. free and open source software eli FOSS) ja omistusoikeudellisiin lisensseihin (eng. proprietary licenses). FOSS-ohjelmistossa tuotteen lähdekoodi on asiakkaan saatavissa ja asiakkaalla on yleensä lupa myös muokata ohjelmistoa. Omistusoikeudellisissa lisensseissä asiakkaalle toimitetaan käyttökelpoinen koodi, mutta asiakas ei pysty vapaasti muokkaamaan tätä. Omistusoikeudellisissa lisensseissä useimmiten myös estetään asiakkaan pääsy ohjelmiston lähdekoodiin. (Lutkevich 2021.)

5 Automaatiosuunnittelu

Teollisuudessa automaatio on suuressa osassa ja sen toteutukseen tarvitaan automaatiosuunnittelua. Automaatioprojektin aluksi määrittelyvaiheessa määritellään käyttäjän vaatimat ominaisuudet automaatiolta, tehdään toiminnallinen kuvaus sekä tilaajan ja toimittajan välinen sopimus. Suunnitteluvaiheessa tehdään järjestelmä- ja toteutussuunnittelu. Kun järjestelmä- ja toteutussuunnittelun jälkeen saadaan toteutuslupa, siirrytään toteutusvaiheeseen. Toteutusvaiheessa automaatiojärjestelmä toteutetaan, ja tähän sisältyy myös sovelluksen toteutus. Tämän jälkeen asennusvaiheessa automaatiojärjestelmä toimitetaan ja asennetaan. Toiminnallisessa testauk-

sessa järjestelmä otetaan käyttöön ja luovutetaan asiakkaalle, jonka jälkeen automaatiojärjestelmä kelpuutetaan asiakkaalla ja tuotantovaihe alkaa. (Automaatiosuunnittelun prosessimalli 2007, 16-17.)

5.1 Sovellussuunnittelun vaiheet

Sovellussuunnittelu toteutetaan automaatiosuunnittelun projektissa tyypillisesti laitteistosuunnittelun jälkeen, joskin joskus aikataulullisista syistä niitä toteutetaan osittain samanaikaisesti. Sovellussuunnittelu aloitetaan useimmiten toimintaselostuksen laadinnalla yhdessä prosessisuunnittelun kanssa. Usein suunniteltavalle kohteelle on jo olemassa esimerkkikohteita, joista saadaan malli uutta kohdetta varten ja vanhoja dokumentteja sovelletaan uuteen sopivaksi. (Vanhempi automaatiosuunnittelija 2023.)

Itse toteutus on selkeintä aloittaa tekemällä IO:hon liittyvät käsittelyt ensiksi ja tämän jälkeen aikaisemmin määritellyn toiminnankuvauksen mukaiset toimilohkot. Kun ohjaukset on saatu valmiiksi, tehdään valvomonäytöt, joiden kautta prosessia tullaan ohjaamaan käyttötilanteessa. Joissain tapauksissa toimintaselostus voi olla ohjauksen toteuttamisen kannalta liian epämääräisiä, jotta toimilohkot saataisiin sen perusteella tehtyä, jolloin voidaan turvautua juuri esimerkiksi aikaisempiin projekteihin. (Vanhempi automaatiosuunnittelija 2023.)

FAT-vaiheessa (Factory Acceptance Testing) sovellussuunnittelun osalta päätarkoituksena on testata luodun sovelluksen sisältämien piirien toimintaa samanlaisessa ympäristössä, jossa ohjelma tulee käyttökohteessakin toimimaan. Nykyaikana sovelluksen FAT-testauksessa käytetään harvoin oikeata fyysistä IO-laitteistoa. Sen sijaan fyysinen laitteisto voidaan testata hyödyntämällä simulointipiirejä sekä virtuaalista IO:ta. Fyysinen laitteisto voidaan simuloida erilaisilla laskennoilla tai juuri virtuaalisella IO:lla. FAT-vaiheessa operointinäytöt tulevat olemaan lopulliset näytöt, samoin kuin oikeankin laitteiston kanssa ja simulointipiirejä voidaan hyödyntää tuottamaan oikeaa elämää vastaavaa dataa, jolloin optimaalisessa tilanteessa operointinäytöltä ei ilmene onko kyseessä oikea fyysinen laitteisto vai FAT-testaus. Näin saadaan testattua kaikkien hälytysten haluttu toiminta sekä laitteiston funktiot ohjelmallisesti. Oleellinen osuus FAT-vaiheessa on varmistaa toimivatko binääritiedot oikeinpäin (esim. varmistutaan, onko signaalilla 1 kyseinen laite auki vai kiinni) ja näkyvätkö mittaukset halutulla tavalla. Nämä tiedot ovat ohjelman toimintakuvauksessa ja on erityisen tärkeää, että näihin tietoihin voidaan luottaa. (Johtava automaatiosuunnittelija 2023.)

Kun SAT-vaihe (Site Acceptance Testing) alkaa, aloitetaan työskentely perehtymällä paikan päällä sijaitseviin laitteisiin, tarkistetaan, onko kaikki tarvittava paikalla ja otetaan kuvat esimerkiksi tarvittavista moottorikilvistä. Myös mahdolliset etäyhteydet toteutetaan tässä vaiheessa. Kun laitteisto on asennettu ja niille tarvittavat sähköt saatu kytkettyä tehdään IO-testaus, jossa tarkastetaan kenttälaitteiden toiminta sisältäen kenttälaitteiden toiminnan testaus sekä moottoreiden pyörimissuunnat. IO-testauksen jälkeen aloitetaan laitteiston toiminnallinen testaus, jossa testataan laitteiston toimintaa kokonaisuutena. Jotkin toiminnalliset testaukset voidaan suorittaa ilman, että prosessi on käynnissä, mutta jotkin vaativat prosessin käynnissä olon. IO-testauksen ja toiminnallisen testauksen yhteydessä päivitetään näihin liittyviin dokumentteihin mahdollisia muutoksia, esimerkiksi punakynien muodossa. (Vanhempi automaatiosuunnittelija 2023.)

Laitteiston testauksen saavuttua päätökseen, luovutetaan As Built -versiot ohjelmista asiakkaalle ja laitteistoon tehdyt muutokset instrumentointisuunnittelijoille, jotka piirtävät punakynistä puhtaat versiot asiakkaalle luovuttamista varten. As Built -ohjelmat tallennetaan myös yrityksen omalle verkkolevylle mahdollista myöhempää käsittelyä varten. Vanhemmalla automaatiosuunnittelijalla (2023) on myös tapana tehdä raporttia tehdystä työstä työn yhteydessä sekä jälkeenpäin. Raportista voi olla hyötyä jälkeenpäin tehtävissä muutoksissa sekä siitä näkee työssä käytetyt ohjelmistoversiot ja salasanat. Myös laitteiston operointiohje on hyvä tehdä ja helpottaa loppukäyttäjän koulutusta. (Vanhempi automaatiosuunnittelija 2023.)

5.1.1 Sovellussunnittelun työkalut

Automaation sovellussuunnittelussa käytetään erittäin laajalti virtuaalikoneita, sillä niillä saadaan sekä eri projektit eristettyä omiin ympäristöihinsä, että estettyä suunnitteluohjelmistojen mahdolliset ristiriidat. Vanhemmalla automaatiosuunnittelijalla (2023) on virtuaalikoneita usealle eri suunnitteluympäristölle. Myös etäyhteyksiä paikan päälle käytetään laajalti sovellussuunnittelussa. Etäyhteyden hyvä puoli on, että kaikki tarvittavat ohjelmat ovat palvelimella paikan päällä, jolloin ei henkilökohtaiselle tietokoneelle tarvitse ohjelmia erikseen. Useimmat suunnitteluohjelmistot myös tarvitsevat tietokoneelle lisenssit käyttöä varten. (Vanhempi automaatiosuunnittelija 2023.)

Oleellisia dokumentteja sovellussuunnittelussa ovat mm. PI-kaaviot, moottorilistat, taajuusmuuttajien parametrilistat, IO-listat sekä pneumatiikka- ja hydrauliikkakaaviot. IO-testauksessa merk-

kaukset tehdään usein pienemmissä projekteissa piirikaavioihin ja suuremmissa projekteissa erilliseen testauslistaan. PI-kaavioita vanhempi automaatiosuunnittelija (2023) on tarvinnut usein prosessiteollisuuden järjestelmien kanssa. IO-listat usein syötetään kerralla ohjelmaan, jonka jälkeen niitä ei juuri tarvitse. Nopeusohjattujen moottoreiden taajuusmuuttajien parametrilistat ovat erityisen tärkeät, sillä moottoreiden onnistunutta testaamista varten täytyy parametrien olla kunnossa. (Vanhempi automaatiosuunnittelija 2023.)

6 Logiikkaohjelmien lisenssien vertailu

Tähän kappaleeseen valittiin vertailtavaksi kolmen eri valmistajan lisenssien aktivointimenetelmää. Valinnat tehtiin hallintamenetelmien erilaisuuden perusteella. Jotta opinnäytetyö saadaan pysymään tiiviinä, tullaan käyttöehtojen selvityksessä keskittymään Siemensin toimittamiin ohjelmistolisensseihin, sillä niitä toimeksiantajalla käytetään eniten.

6.1 Eri valmistajien lisenssienhallintamenetelmät

6.1.1 Siemens

Siemens hyödyntää ohjelmistolisenssiensä käsittelyssä omaa Automation License Manager-ohjelmaansa. Lisenssit toimitetaan joko fyysisellä USB-tikulla tai verkosta ladattavana. Lisenssi siirretään Automation License Managerilla kiintolevylle, jolta ohjelmistot pystyvät niitä käyttämään. Automation License Managerin voi määritellä myös lisenssipalvelimeksi, jolloin lisenssejä pystyy siirtämään verkon yli tietokoneelta toiselle. (Software and licenses n.d.)

6.1.2 Schneider Electric

Schneider Electricin lisenssinhallintamenetelmällä voidaan lisenssit aktivoida verkon yli tai ilman internet-yhteyttä. Verkon yli lisenssit saa aktivoitua syöttämällä aktivointi-ID Schneiderin verkkoportaaliin ja tämän jälkeen lisenssi on aktivoitu. Offline-menetelmällä aluksi luodaan ohjelmalla lisenssin aktivointipyyntötiedosto, joka internet-yhteyden omaavalla laitteella syötetään aktivointiverkkosivulle, josta saadaan lisenssivastaustiedosto. Lisenssivastaustiedosto voidaan kopioida offline-laitteelle ja aktivoida sieltä. (Activating a license 2022.)

6.1.3 Beckhoff

Beckhoff tarjoaa järjestelmilleen kahdenlaisia lisenssejä; TwinCAT Standard License ja TwinCAT Volume License. Standard Licensellä on jokaisella lisenssimoduulilla oma yksilöllinen lisenssi-ID ja Volume Licensessä on useampi lisenssimoduuli, jotka jakavat yhteisen lisenssi-ID:n. Beckhoffin lisenssien tapauksessa luodaan lisenssin aktivointipyyntötiedosto, joka muodostuu kyseisen laitteen ID:stä tai Volume ID:stä, tilausnumerosta ja TwinCAT Platform Levelistä. TwinCAT Platform Level muodostuu laitteiston suorituskyvyn perusteella ja tämän perusteella muodostuu myös lisenssin hinta. Schneiderin tapaan tämä lähetetään Beckhoffin lisensointipalvelimelle, joka palauttaa vastaustiedoston, joka syötetään ohjelmistolle ja aktivoi lisenssin. (EL6070 | EtherCAT Terminal, license key for TwinCAT 3.1 n.d.)

6.2 Selvitys Siemensin lisenssien käyttöehdoista

Siemens tarjoaa automaatio-ohjelmistoilleen neljänlaisia lisenssejä; kokeilulisenssejä, vuokralisenssejä, yksittäisiä lisenssejä sekä kelluvia lisenssejä (eng. floating license). Lisenssityypit erottavat niiden kesto, tuotantokäytön salliminen ja se, kuinka monella laitteella lisenssiä voidaan käyttää. (Softwares and licenses n.d.)

Kokeilulisenssit ovat ilmaisia, ja niitä voi käyttää yhdellä laitteella rajoitetun ajan. Kokeilulisenseitä ei saa käyttää tuotantokäytössä eikä Siemens ota vastuuta niiden käytöstä. Vuokralisenssit puolestaan sallivat tuotantokäytön ja sisältävät kaikki käyttöön tarvittavat ominaisuudet. Vuokralisenssit tulevat yleensä aloituspaketin mukana, eikä niitä pysty tilaamaan erikseen. Vuokralisenssit on suunnattu pidempää kokeilujaksoa varten, ja voivat olla hyviä testatessa yhteensopivuutta laitteiston kanssa. (Softwares and licenses n.d.)

Yksittäisellä lisenssillä ei ole mitään rajoitteita ohjelmiston käytön keston tai ominaisuuksien suhteen, mutta yksittäinen lisenssi sallii lisenssin käytön ja ohjelmiston asennuksen vain yhdellä tietokoneella, ja se tallennetaan kyseisen koneen kiintolevylle. Yksittäinen lisenssi soveltuu hyvin esimerkiksi valvomotietokoneille, jotka sijaitsevat aina tietyssä paikassa ja joilta ohjataan logiikkaohjaimen hallitsemaa prosessia. Kelluva lisenssi eroaa yksittäisestä lisenssistä siten, että se sallii ohjelmiston asentamisen usealle koneelle, sekä lisenssin käyttämisen usealta tietokoneelta.

Kelluvassa lisenssissä tosin on määritelty, kuinka monella tietokoneella lisenssiä pystyy yhtäaikaisesti käyttämään. Lisenssin voi säilöä lisenssipalvelimelle, johon käyttäjän tietokoneella voidaan muodostaa yhteyden ja näin ollen käyttää lisenssiä verkon yli. Kelluvat lisenssit on suunnattu ohjelmointikäyttöön, jossa yrityksessä voi olla useampi sovellussuunnittelija, jotka tarvitsevat lisenssiä käyttöönsä eriaikaisesti. (Softwares and licenses n.d.)

Siemens jaottelee ohjelmistonsa ajonaikaisiin ohjelmistoihin (eng. runtime software) ja suunnitteluohjelmistoihin. Siemensin mukaan suunnitteluohjelmiston haltija saa ilman lisenssimaksun maksamista käyttää luomiaan suunnitteluohjelmiston osia sisältäviä ohjelmia osana omia ohjelmiaan sekä luovuttaa niitä kolmansille osapuolille. Tämä on ainoa tapaus, jossa on sallittua luovuttaa suunnitteluohjelmiston osia eteenpäin, muutoin suunnitteluohjelmiston osien erottaminen on kiellettyä. Ajonaikaisista ohjelmistoista mainitaan, että joka kerta, kun ajonaikaista ohjelmistoa tai sen osia sisällytetään ohjelmiston haltijan ohjelmiin, on jokaista kopiota kohden ostettava oma lisenssinsä. Tämän lisenssin on vastattava kyseistä käyttötarkoitusta ja oltava voimassa olevan Siemensin katalogin mukainen. Mikäli ajonaikainen ohjelmisto sisältää parametrointiin tai konfigurointiin liittyviä työkaluja ja laajennettuja oikeuksia, mainitaan niistä ohjelmiston Readme-tiedostossa. (Software Licensing Conditions 2018.)

Jos ohjelmistoa päivitetään päivityslisenssillä, loppuu edellisen lisenssin voimassaolo, mutta sitä voi kuitenkin käyttää eriaikaisesti päivitetyn version kanssa. Eli vanhemman version, joka on päivitetty, käyttöön on oikeus päivitetyllä lisenssillä, mutta ei yhtäaikaisesti päivitetyn version kanssa. Ohjelmiston haltija voi siis käyttää vanhentunutta versiota samassa määrässä instansseja, kuin on hankkinut päivityslisenssejä. (Software Licensing Conditions 2018.)

Ohjelmiston haltijan on sallittua luovuttaa hankkimiaan ohjelmistoja ja lisenssejä eteenpäin kolmansille osapuolille siten, että alkuperäinen haltija lopettaa kaikkien ohjelmistojen kopioiden käytön, tai Siemensin pyynnöstä toimittaa ne Siemensille, mikäli haltijan ei lain mukaan tarvitse säilyttää niitä pidempään. Jos ohjelmistoja täytyy säilyttää käytöstä poiston jälkeen, ei niitä saa enää käyttää. Ohjelmiston toimituksen yhteydessä on myös luovutettava ohjelmistojen vaatimat lisenssiavaimet, tilausvahvistus sekä lisenssisertifikaatit. Mikäli kyseessä on päivityslisenssi, on myös aikaisemman version lisenssi toimitettava samassa yhteydessä. Ohjelmiston haltija ei saa vuokrata

tai lainata ohjelmistoa kolmannelle osapuolelle käytettäväksi. Siemens voi vaatia kirjallista vahvistusta käytöstä poistosta tai säilytyksen syystä. Luovuttajan on myös velvoitettava kolmas osapuoli noudattamaan Siemensin käyttöehtoja. (Software Licensing Conditions 2018.)

Käyttöehtojen pääpiirteet ovat seuraavassa listassa tiivistettynä:

- Yksittäinen lisenssi (Single license) sallii ohjelmiston asennuksen ja käytön vain yhdelle laitteelle ja lisenssin täytyy sijaita kyseisellä laitteella.
- Kelluva lisenssi (Floating license) sallii ohjelmiston yhtäaikaisen käytön lisenssin määräämällä määrällä laitteita ja ohjelmiston saa asentaa kymmenkertaiselle määrälle laitteita. Kelluvaa lisenssiä voi käyttää etänä lisenssipalvelimelta.
- Kokeilulisenssiä ei saa käyttää tuotantoon ja ohjelma täytyy poistaa laitteelta jakson päätyttyä.
- Suunnitteluohjelmiston asianmukaisella käytöllä luotuja ohjelmistoja saa ohjelmiston haltija käyttää ja luovuttaa kolmannelle osapuolelle ilman erillistä lisenssiä.
- Ohjelmaversion päivityksen yhteydessä vanhan version käyttöoikeus poistuu, mutta päivityslisenssillä on oikeus käyttää vanhaa versiota.
- Jos ohjelmistot luovutetaan eteenpäin, on lisenssit, tilausvahvistus, mahdolliset vanhan version lisenssit ja lisenssisertifikaatit toimitettava samassa yhteydessä.

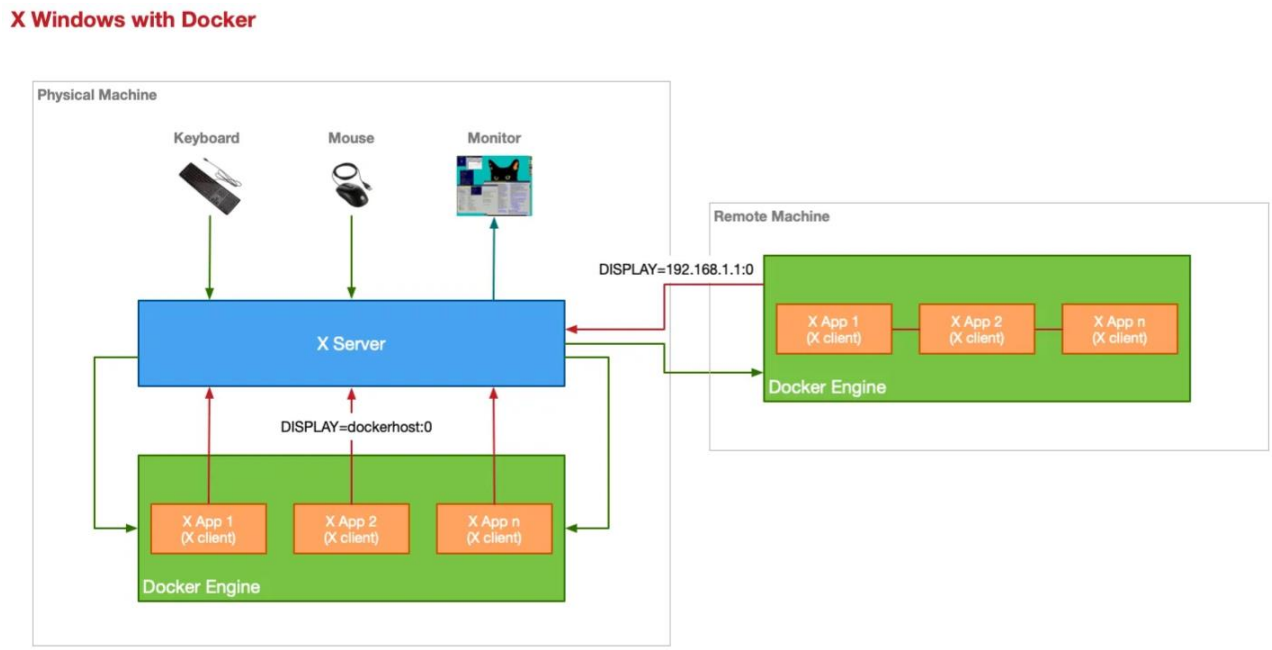
7 Virtualisoinnin tulevaisuus automaatiosuunnittelussa

7.1 Virtuaalitietokoneiden ongelmat

Kuten tietoperustassa mainittiin virtuaalikoneiden ja konttien välisistä kokoeroista, virtuaalikoneissa voi muodostua ongelmaksi niiden suuri säilytyskoko kontteihin verrattuna. Sovellussuunnittelua tämä haittaa siltä osin, että tietokoneen omalle kovalevylle ei välttämättä mahdu kaikki tarpeelliset virtuaalikoneet siten, että muullekin tarpeelliselle jäisi tilaa kovalevylle. Tällöin virtuaalikoneita joudutaan siirtämään erillisille kiintolevyille ja niiden seuranta voi olla hankalaa. Esimerkiksi Siemensin lisenssit tallentuvat koneen kiintolevylle ja virtuaalikoneiden kohdalla ne siirtyvät virtuaalikoneen mukana, sillä ne tallentuvat virtuaalikoneen virtuaaliselle levylle. Näissä tapauksissa tulee siten olla myös tarkkana, ettei lisenssejä sisältävää virtuaalikonetta kopioida, koska silloin sillä koneella sijaitsevat lisenssitkin kopioituvat ja tällöin rikottaisiin valmistajan määrittämiä ehtoja. Tästä voi seurata tilanne, jossa ei tiedetä missä virtuaalikoneilla sijaitsevat lisenssit ovat.

7.2 Konttiteknologian mahdollinen käyttö

Samaan tarkoitukseen, mihin nykyisin sovellussuunnittelussa käytetään virtuaalikoneita, voisi tulevaisuudessa olla mahdollista käyttää kontteja virtuaalikoneiden sijaan. Ennen kuin se on mahdollista, on olemassa muutamia kysymyksiä, joihin olisi vastattava. Yksi kysymys on graafisten käyttöliittymien toiminta konteissa. Docker ei itsessään osaa näyttää graafista käyttöliittymää samalla tavalla, kuin virtuaalikone luo virtuaalisen näytön ja ohjaa graafiset elementit siihen näkyville. Tämä voidaan konteissa ratkaista asentamalla isäntäkoneelle X-palvelin, johon kontilla toimiva ohjelma yhdistyy X-clientin kautta ja ohjaa ohjelman tuottamat graafiset elementit näytölle. Näin kontilla toimivan ohjelman kanssa voidaan kommunikoida hiirellä ja näppäimistöllä siten, että palvelin lähettää clientille saamansa ohjeet ja client lähettää graafisen tiedon takaisin palvelimelle, joka ohjaa sen käytettävälle näytölle. X voi keskustella paikallisesti isäntäkoneen kanssa, tai IP:n kautta etäyhteydellä, kuten kuviossa 6 on kuvattu. (Michas 2019.)



Kuvio 6 Havainnollistus graafisen käyttöliittymän käytöstä (Michas 2019.)

Windows-ohjelmien yhteydessä edellinen ratkaisu on haastavampi, sillä toistaiseksi selvitystyön perusteella ei ole ratkaisua graafisen käyttöliittymän tuontiin Windows-pohjaiselta kontilta, vaikka Windows-pohjaiset kontitkin ovat mahdollisia. Tätä voidaan kiertää Linux-pohjaisella kontilla, jossa on asennettuna Wine. Wine toimii välikätenä, jonka avulla Windows-pohjaisia sovelluksia voidaan

suorittaa Linuxilla. Tällöin voidaan Linux-pohjaisella kontilla suorittaa Windows-pohjaisia ohjelmia ja aikaisemman kappaleen mukaisella ratkaisulla saada graafinen käyttöliittymä isäntäkoneelle näkyviin.

Siemens tukee Docker-konttien käyttöä jo WinCC OA -SCADA-järjestelmänsä palvelinosissa, mutta WinCC OA:nkaan graafisen käyttöliittymän sisältämät toiminnot eivät vielä tue Docker-käyttöä (Docker support n.d.). Myös Siemensin Industrial Edge-järjestelmässä, joka on IoT-analyyseihin perustuva järjestelmä, on sovelluksia, jotka käyttävät Dockeria. Industrial Edgessä on myös mahdollista luoda omia Docker-pohjaisia sovelluksia (Industrial Edge for IT-specialists n.d.).

Siemensin edustajan mukaan TIA Portalia ei ole systeemitestattu konttiympäristössä eikä hänen tiedossaan ole, että olisi lähitulevaisuuden suunnitelmissa. (Siemensin teknisen tuen asiantuntija 2023.)

7.2.1 X Window System-ikkunointijärjestelmä

X Window System on avoimeen lähdekoodiin perustuva ikkunointijärjestelmä, joka mahdollistaa ohjelman kommunikoinnin käyttäjän kanssa esimerkiksi näppäimistön, hiiren ja näytön välityksellä, eli graafisen käyttöliittymän käytön. X toimii kehyksenä valitulle ikkunanhallintajärjestelmälle. Esimerkiksi Linux-pohjaiset käyttöjärjestelmät käyttävät X:ää käytännössä standardina käyttöjärjestelmissä, joissa halutaan graafinen käyttöliittymä. X:n käyttämiseen tarvitaan palvelin sekä client, jotka kommunikoivat keskenään välittäen tietoa näppäimistöltä ja hiireltä sovellukselle ja sovellukselta näytölle. X:n käyttämä palvelimen ja clientin kommunikointi poikkeaa normaalista siinä, että sen sijaan, että monet clientit pyytäisivät palvelimelta tietoja, pyydetään tässä tapauksessa palvelimella tietoja monelta sovellukselta eli clientiltä. (The X Window System: A Brief Introduction 2006.)

X:lle on useita vaihtoehtoja palvelinalustaksi, joista yksi on Windowsilla toimiva VcXsrv. Tätä voidaan käyttää kommunikointiin kontilla toimivan Linux-sovelluksen ja VcXsrv:n Windowsin työpöydälle luoman ikkunan välillä. Tässä tapauksessa voidaan kontilla käynnistää graafisella käyttöliittymällä toimiva sovellus, joka tällöin toimii X-clientinä. (Michas 2019.)

7.2.2 Wine-yhteensopivuuskerros

Wine on yhteensopivuuskerros, joka mahdollistaa Windows-pohjaisten sovellusten käyttämisen esimerkiksi Linuxilla. Wine ei simuloi Windowsia kuten virtuaalikone tai emulaattori, vaan kääntää Windows-kutsut käytössä olevan käyttöjärjestelmän ymmärrettäväksi, jolloin ei tule muiden ratkaisujen aiheuttamia suorituskykyhäviöitä. Wine on ilmainen ja avoimeen lähdekoodiin perustuva ohjelma. (About Wine n.d.)

7.3 Siemensin pilvipohjaiset ratkaisut

Siemens tarjoaa TIA Portalin uusimpia versioita myös pilvipohjaisena palveluna, jossa itse käyttäjän tietokoneelle ei tarvitse asentaa ohjelmia, vaan selaimella päästään internet-yhteyden avulla etänä käyttämään TIA Portalia. TIA Portal Cloudin avulla myös projektitiedostot voidaan tallentaa pilveen. TIA Portal Cloudissa saatavilla on versiot V15.1, V16, V17 ja V18. TIA Portal Cloudista on saatavilla kuukausittainen tilaus tai tuntipohjainen laskutus. (TIA Portal Cloud n.d.)

Yksi varteenotettava vaihtoehto on myös TIA Portal Cloud Connector. Tässä ratkaisussa käyttäjän omalle palvelimelle asennetaan TIA Portalin haluttu versio sekä TIA Portal Cloud Connector, joka määritellään toimimaan palvelimena. Tällöin käyttäjän omalle tietokoneelle tarvitsee asentaa vain TIA Portal Cloud Connector, joka määritellään toimimaan clientinä. Tietokoneella voidaan näin ollen ottaa etäyhteys palvelimeen, jolla voidaan siten käyttää TIA Portalia ilman, että se on käyttäjän tietokoneella asennettuna. Cloud Connectorin avulla TIA Portal tunnistaa tietokoneen verkkorajapinnat ja näin ollen voidaan yhdistää logiikkaohjaimeen ohjelman latausta tai monitorointia varten. Tässä ratkaisussa haittapuolena on se, että verkkoyhteyden yli tulee viivettä, joka voi hankaloittaa työskentelyä ja mikäli halutaan reaaliaikaista tietoa prosessista voi viive haitata huomattavasti. (Working with the TIA Portal Cloud Connector 2017.)

Molemmissa näissä vaihtoehdoissa tarvitaan internet-yhteys, joka voi olla karsiva tekijä kentällä oltaessa, mikäli internet-yhteyttä ei ole tai se on liian hidas.

8 Teemahaastatteluiden toteutus

Saadakseni lähtötilanteesta ja sen ongelmista hyvän yleiskuvan päätin toteuttaa kaksi teemahaastattelua hieman eri taustoista olevien automaatiosuunnittelijoiden kanssa. Samassa yhteydessä näin hyvän tilaisuuden yleiskuvan muodostamiseen automaation sovellussuunnittelusta tietopohjaa sekä omaa ammatillista kehittymistä varten. Teemahaastatteluissa pääteemoina olivat kuvan muodostus haastateltavien taustoista, sovellussuunnittelun vaiheet ja niiden sijoittuminen projektikokonaisuuteen, sovellussuunnittelussa käytettävät työvälineet sekä nykyinen toimintatapa projektin aloituksessa ja sen kehityskohteet. Haastateltaviksi valitsin vanhemman automaatiosuunnittelijan sekä johtavan automaatiosuunnittelijan, jotka molemmat toimivat Rejlersin Jyväskylän toimistolla automaatiosuunnittelussa. Vanhemmalla automaatiosuunnittelijalla on enemmän kokemusta pienemmissä projektikokonaisuuksissa toimimisesta ja on työssään käyttänyt laajasti Siemensin logiikkaohjaimia ja TIA Portalia. Johtava automaatiosuunnittelija puolestaan on toiminut suuremmissa projekteissa ja Damatic XD (myöhemmin Valmet DNA), STEP 7 Classic sekä PCS7-järjestelmien parissa.

Vanhemman automaatiosuunnittelijan koulutustaustaan sisältyy lukio, tietotekniikan insinöörin AMK-tutkinto, automaatiotekniikan YAMK-tutkinto sekä ammatillisen opettajan pätevyys. Hänellä on vajaan 20 vuoden työkokemus automaatiosuunnittelijana Jyvästekillä samalla toimistolla, joka nykyisin tunnetaan yrityskauppojen kautta Rejlersinä. Hän on työskennellyt paperi- ja kartonkikoneiden, paperimassalinjojen, kaukolämpöjärjestelmien parissa sekä tehnyt projektin sairaalan pesukoneiden ohjauksiin liittyen. Suurimmaksi osaksi hän on työskennellyt Siemensin logiikkaohjainten parissa ja on perehtynyt myös Mitsubishi Electricin järjestelmällä ja Valmet DNA:lla työskentelyyn. (Vanhempi automaatiosuunnittelija 2023.)

Johtavalla automaatiosuunnittelijallakin on koulutustaustassaan lukio ja hän on toiminut automaatiosuunnittelijana noin 30 vuotta Jyvästekillä ja yrityskauppojen myötä nykyisellä Rejlersillä. Hänen työkokemuksestaan valtaosa on paperitehtaan eri osuuksien sovellussuunnittelussa. (Johtava automaatiosuunnittelija 2023.)

8.1 Saadut kehitysehdotukset

Vanhempi automaatiosuunnittelija ehdotti, että eri ohjelmaversioiden ohjelmointiympäristöille olisi hyvä olla olemassa pohjat, joista pystyisi tarvittaessa ottamaan kopion ja lisäämään itse tarvittavat lisenssit. Nykyisellä toimintatavalla ennen ohjelmointiympäristön luovutusta toiselle sitä tarvitsevalle henkilölle on sieltä poistettava salassa pidettävät projektitiedostot. Myös virtuaalikoneella olevat lisenssit on siirrettävä pois kyseisestä ympäristöstä niiden kopioitumisen välttämiseksi ja joudutaan käyttämään esimerkiksi lisenssimuistitikkua välikappaleena. (Vanhempi automaatiosuunnittelija 2023.)

Johtava automaatiosuunnittelija näki hyvänä ideana, että pelkän käyttöjärjestelmän sisältävän virtuaalikonepohjan yhteydessä olisi saatavilla eri ohjelmien tarvittavat asennuspaketit, joka olisi helppo verkkolevyltä noutaa. Toisaalta taas ohjelmien asennuksetkin vievät aikaa samalla tavalla kuin suuren virtuaalikoneen noutaminen. (Johtava automaatiosuunnittelija 2023.)

9 Opinnäytetyön eettisyys ja tutkimuksen toteutus

Teemahaastattelut tehtiin kasvotusten toimistolla käyttäen kynää ja paperia muistiinpanojen kirjaamiseen ja toimeksiantajalta saatiin tutkimuslupa suullisena. Haastattelujen tuloksena saatu sisältö käytiin läpi haastateltujen kanssa. Siemensin teknisen tuen antaman vastauksen referointiin saatiin lupa.

10 Pohdinta

Tuloksena opinnäytetyössä toimeksiantajalle saatiin toimiva ratkaisu, joka tulee tulevaisuudessa suoraviivaistamaan uuden ohjelmointiympäristön käyttöönottoa projektin alkaessa ja helpottamaan lisenssienhallintaa. Ratkaisussa hyödynnettiin Siemensin Automation License Managerin ominaisuuksia ohjelmistolisenssien hallinnan keskittämiseksi. Työn aikana ilmi tulleiden ajatusten myötä myös jatkokehitysmahdollisuuksia ilmaantui. Samalla saatiin myös helppolukuinen selvitys Siemensin ohjelmistolisensseihin liittyvistä käyttöehdoista ja tämän selvitys tulee auttamaan ohjelmistolisenssien väärinkäytön ennaltaehkäisyssä.

Opinnäytetyössä käsiteltiin myös laajasti virtualisointia virtuaalikoneiden ja konttiteknologian muodossa. Työssä todettiin, että kontit ovat jo nopeaa vauhtia korvaamassa virtuaalikoneita palvelinpuolella, mutta työpöytäsovelluksissa niitä ei vielä juurikaan käytetä. Automaatiotekniikankin puolella kontteja käytetään jo SCADA-järjestelmissä sekä pilvianalytiikassa, mutta sovellussuunnittelussa virtuaalikoneet näyttävät lähitulevaisuudessakin olevan pääasiallinen työkalu suunnitteluympäristöjen eristämiseen.

Teemahaastatteluiden myötä saatiin ensi käden lähteistä hyvää tietoa automaation sovellussuunnittelusta ja siihen liittyvistä työkaluista ja näihin liittyvistä haasteista. Teemahaastatteluista saatiin myös kehitysehdotuksia, joiden perusteella pystyttiin kartoittamaan vaihtoehtoja toteutusmenetelmille sekä hyviä ajatuksia tulevaisuutta koskien.

10.1 Tulosten luotettavuus

Työn tietoperustassa lähteinä käytettiin automaatiosuunnitteluun liittyvää kirjallisuutta niin Jamkin kirjastoista fyysisinä kirjoina lainattuna, kuin Janetin e-kirjoina. Lähteenä käytettiin myös Suomen automaatioseuran julkaisua automaatiosuunnittelun vaiheisiin liittyen, sekä ensisijaista tietoa automaatiosuunnittelijoiden haastatteluista. Työssä käytettiin näiden lisäksi myös luotettaviksi koettuja verkkojulkaisuja useasta paikasta lähteinä.

Kehittämistutkimus aloitettiin selvittämällä lähtötilanteet vapaamuotoisella keskustelulla ja tämän jälkeen teemahaastatteluiden yhteydessä kehitysehdotuksia kysymällä. Valmiista toteutuksesta mielipiteet olivat positiivisia. Toimeksiantajan tietoturvallisuuteen liittyvien syiden vuoksi lisenssienhallinnan toteutus on salassa pidettävässä liitteessä. Siemensin ohjelmistolisenssien käyttöehdoista tehtiin oleellisimmista kohdista tiivistelmä käyttäen käyttöehtodokumenttia ensisijaisena lähteenä.

Konttitekniikan käyttömahdollisuuksia tutkittaessa käytettiin monipuolisesti verkkojulkaisuja aihepiiriin liittyen sekä tarkasteltiin aikaisempia käyttökohteita. Myös Siemensin teknisestä tuesta kysyttiin, onko käyttömahdollisuuksia TIA Portaliin liittyen tarkasteltu. Pilvipohjaiset vaihtoehdot tuotiin myös selvityksessä esille.

Lähteet

About Wine. N.d. Wine HQ. Artikkeli. Viitattu 13.2.2023. https://www.winehq.org/about.

Activating a license. 2022. Artikkeli. Viitattu 15.2.2023. https://digital-energy-help.se.com/pme/content/4_configuring/licensing/activating_a_license.htm.

Automaatiosuunnittelun prosessimalli. 2007. Helsinki: Suomen automaatioseura ry. Kirja. Viitattu 3.1.2023. https://www.automaatioseura.fi/site/assets/files/1426/automaatiosuunnittelun_prosessimalli.pdf.

Bolton, W. 2009. Programmable Logic Controllers. Burlington: Elsevier Ltd. Viitattu 1.2.2023. https://janet.finna.fi/, Ebook Central Academic Complete.

Docker support. N.d. Siemens. Ohjeistus WinCC OA:n käyttöön. Viitattu 12.2.2023. https://www.winccoa.com/documentation/WinCCOA/3.18/en_US/WCCOA_Voraussetzungen/docker_support.html.

EL6070 | EtherCAT Terminal, license key for TwinCAT 3.1. N.d. Beckhoff. Tuote-esittely. Viitattu 15.2.2023. https://www.beckhoff.com/en-en/products/i-o/ethercat-terminals/el6xxx-communication/el6070.html.

Industrial Edge for IT-specialists. N.d. Siemens. Industrial Edge-palvelun esittely. Viitattu 12.2.2023. https://www.siemens.com/global/en/products/automation/topic-areas/industrial-edge/it-specialists.html

Investors. N.d. Vuoden 2022 Rejlersin taloustietoja yrityksen www- sivuilla. Viitattu 16.1.2023. https://www.rejlers.com/Investors/.

Johtava automaatiosuunnittelija. 2023. Rejlers Finland Oy. Teemahaastattelu 15.2.2023.

Kananen, J. 2015. Kehittämistutkimuksen kirjoittamisen käytännön opas. Jyväskylä: Jyväskylän ammattikorkeakoulu.

Kisller, E. 2023. THE BASICS: 7 Alternatives to Docker: All-in-One Solutions and Standalone Container Tools. Viitattu 24.1.2023. https://jfrog.com/knowledge-base/the-basics-7-alternatives-to-docker-all-in-one-solutions-and-standalone-container-tools/.

Lutkevich, B. 2021. What is a software license? Everything you need to know. TechTarget. Viitattu 10.2.2023. https://www.techtarget.com/searchcio/definition/software-license.

Meistä. N.d. Tietoa Rejlersistä yrityksen www- sivuilla. Viitattu 16.1.2023. https://www.rejlers.fi/Meista/.

Michas. N. 2019. Running Desktop Apps in Docker. Medium. Artikkeli. Viitattu 12.2.2023. https://betterprogramming.pub/running-desktop-apps-in-docker-43a70a5265c4.

Portnoy, M. 2016. Virtualization essentials. Indianapolis: John Wiley & Sons Inc.

Projektimme. N.d. Tietoa Rejlersin teollisuuden alan projekteista yrityksen www- sivuilla. Viitattu 16.1.2023. https://www.rejlersindustry.fi/projektimme/.

Siemensin teknisen tuen asiantuntija. 2023. TIA Portalin käyttö Docker-kontissa. Sähköpostiviesti 15.2.2023. Vastaanottaja L. Ruotsalainen. Siemensin teknisen tuen kanssa käyty keskustelu TIA Portalin käytön mahdollisuuksista ja tulevaisuudensuunnitelmista Docker-konttien suhteen.

Software Licensing Conditions. 2018. Siemens. Ohjelmistolisenssien käyttöehdot. Viitattu 15.2.2023. https://mall.industry.siemens.com/legal/DE-EN/Software_Licensing_Conditions_en_LKB10710.pdf.

Softwares and licenses. N.d. Siemens. Tietoa yrityksen tarjoamista tuotteista. Viitattu 11.2.2023. https://new.siemens.com/global/en/products/automation/topic-areas/simatic/licenses.html.

Status IEC 61131-3 standard. N.d. Tietoa standardin nykytilasta organisaation www- sivuilla. Viitattu 31.1.2023. https://plcopen.org/status-iec-61131-3-standard.

Stoneman, E. 2021. Learn Docker. J. Ross Publishing. Viitattu 24.1.2023. https://janet.finna.fi/, Skillsoft Books ITPro.

Suunnittelupalvelut. N.d. Tietoa Rejlersin teollisuuden alan tarjonnasta yrityksen www- sivuilla. Viitattu 16.1.2023. https://www.rejlersindustry.fi/teollisuuden-palvelut/suunnittelupalvelut/.

Tavoite ja strategia. N.d. Tietoa Rejlersin tavoitteista yrityksen www- sivuilla. Viitattu 16.1.2023. https://www.rejlers.fi/Meista/Tavoite_ja_strategia/.

The X Window System: A Brief Introduction. 2006. The Linux Information Project. Artikkeli. Viitattu 13.2.2023. http://www.linfo.org/x.html.

TIA Portal Cloud. N.d. Siemens. Artikkeli. https://new.siemens.com/global/en/products/automation/industry-software/automation-software/tia-portal/highlights/tia-portal-cloud.html.

TIA Portal drive integration with SINAMICS Startdrive. N.d. Palveluiden esittelyä yrityksen www-sivuilla. https://www.siemens.com/global/en/products/drives/selection-and-engineering-tools/sinamics-startdrive-commissioning-software.html.

Vanhempi automaatiosuunnittelija. 2023. Rejlers Finland Oy. Teemahaastattelu 14.2.2023.

Wallenius, N. 2022. Konttiteknologia – mitä kontit ovat ja mitä hyötyä niistä on? Wallenius consulting. Artikkeli. Viitattu 24.1.2023. https://niklaswallenius.fi/konttiteknologia-mita-hyotya/.

What is HMI? 2018. Inductive Automation. Artikkeli. Viitattu 17.1.2023. https://www.inductiveautomation.com/resources/article/what-is-hmi.

What is SCADA? 2021. PLCynergy. Artikkeli. Viitattu 17.1.2023. https://plcynergy.com/what-is-scada/.

Working with the TIA Portal Cloud Connector. 2017. Siemens. Artikkeli. https://support.industry.siemens.com/cs/document/109747305/working-with-the-tia-portal-cloud-connector?dti=0&lc=en-BO.

Liitteet

Liite 1. Teemahaastattelurungot

Teemahaastattelun runko (mukaillen Kananen 2015)	
Yrityksen taustatiedot	
Yrityksen nimi	Rejlers Finland Oy
Toimiala	Tekninen suunnittelu ja konsultointi
Liikevaihto	95,1 milj. euroa (2021)
Henkilöstö	945 (2021)
Teemahaastattelun toteutus	
Haastattelija	Leevi Ruotsalainen
Ajankohta	14.2.2023
Haastattelun kesto	1h
Haastateltava henkilö	Vanhempi automaatiosuunnittelija
Teemat	
Teema 1	Tausta
Teema 2	Sovellussuunnittelun vaiheet
Teema 3	Sovellussuunnittelun työvälineet
Teema 4	Nykyinen toimintatapa

Teemahaastattelun runko (mukaillen Kananen 2015)	
Yrityksen taustatiedot	
Yrityksen nimi	Rejlers Finland Oy
Toimiala	Tekninen suunnittelu ja konsultointi
Liikevaihto	95,1 milj. euroa (2021)
Henkilöstö	945 (2021)
Teemahaastattelun toteutus	
Haastattelija	Leevi Ruotsalainen
Ajankohta	15.2.2023
Haastattelun kesto	1h
Haastateltava henkilö	Johtava automaatiosuunnittelija
Teemat	
Teema 1	Tausta
Teema 2	Sovellussuunnittelun vaiheet
Teema 3	Sovellussuunnittelun työvälineet
Teema 4	Nykyinen toimintatapa

Liite 2. Lisenssienhallinnan toteutus

Liite 3. Lisenssienhallinnan ohje