LAMK Lahden ammattikorkeakoulu
Lahti University of Applied Sciences

Vanerinlajittelulinjaston simulointi 3DCreatella

Case: Raute Oyj

LAHDEN
AMMATTIKORKEAKOULU
Tekniikan ala
Kone- ja tuotantotekniikan
koulutusohjelma
Suunnittelupainotteinen
mekatroniikka
Opinnäytetyö
Kevät 2016
Kalle Torniainen

Lahden ammattikorkeakoulu
Kone- ja tuotantotekniikan koulutusohjelma

TORNIAINEN, KALLE:

Vanerinlajittelulinjaston simulointi 3DCreatella
Case: Raute Oyj

Suunnittelupainotteisen mekatroniikan opinnäytetyö, 40 sivua

Kevät 2016

TIIVISTELMÄ

Opinnäytetyön tarkoituksena oli luoda joustava, parametrinen ja tarkka simulaatiomalli vanerinlajittelulinjastosta. Mallia tuli voida käyttää kyseisen linjaston esittelyyn ja läpimenoaikojen määrittämiseen. Työ tehtiin Raute Oyj:lle syksyn 2015 ja kevään 2016 aikana.

Simulaatiomalli tehtiin kaksivuotisena opiskelijaprojektityönä kolmen insinööriopiskelijan voimin. Projektin ensimmäinen vuosi käytettiin Visual Componentsin 3DCreate-ohjelmiston opiskeluun ja toinen vuosi itse simulaatiomallin tekoon. Simulaatiomallia varten tehtiin 3DCreatella 16 erilaista komponenttia, joista malli rakennettiin. Mallin teossa käytettiin resurssikeskeistä näkökulmaa. Opinnäytetyön tekijä toimi projektipäällikkönä ja vastasi kokonaisuuden ohella myös osasta komponentteja.

Työn tuloksena saatiin parametrinen simulaatiomalli, joka tallentaa simulaatioajon asetustiedot ja käsiteltyjen tuotteiden määrät tekstitiedostoon analysointia varten. Toimeksiantajan mukaan simulaatiomallin tulokset vaikuttavat oikeilta ja realistisilta ja itse simulaatiomalli oikeiden arvojen pohjalta laaditulta ja luotettavalta.

Asiasanat: Raute Oyj, simulointi, tehdassimulointi, 3DCreate, vanerinlajittelulinjasto

Lahti University of Applied Sciences
Degree Programme in Mechanical and Production Engineering

TORNIAINEN, KALLE:

Simulating a plywood grading line with 3DCreate
Case: Raute Corporation

Bachelor's Thesis in Mechatronics 40 pages

Spring 2016

ABSTRACT

The purpose of this Bachelor's Thesis was to create a flexible, parametric, and accurate simulation model of a plywood grading line. The model was intended to be used for demonstration of the line, and in defining the throughput of the line. The work was commissioned by Raute Corporation and conducted between autumn 2015 and spring 2016.

The simulation model was created as a two-year student project by three students of engineering. The first year was spent studying 3DCreate, a 3D simulation software developed by Visual Component, a Finnish software company. The simulation model was created during the second year of the project. The simulation model consists of 16 different components which were created with 3DCreate. The model was created from a resource-oriented perspective. The author of this Bachelor's Thesis worked as a project manager, and was responsible for the overall configuration and some of the components.

The resulting parametric simulation model can be used to measure the throughput of the products, and it saves the simulation settings and the results into a txt-file. According to the client, the results of the simulation model seem to be realistic and the simulation model itself seems to be reliable and created with correct values.

Key words: Raute, simulation, factory simulation, 3DCreate, plywood grading line

SISÄLLYS

1 JOHDANTO 1

2 RAUTE OYJ 3

3 SIMULOINTI 6
- 3.1 Simulointi osana opetusta ja opiskelua 6
- 3.2 Tehdassimulointi 7
- 3.2.1 Simuloinnin hyödyt 7
- 3.2.2 Simulaatiomallin suunnittelu ja rakentaminen 8
- 3.2.3 Simulaatiomallin käyttö 10

4 SIMULOINTI KÄYTÄNNÖSSÄ 11

5 3DCREATE 12
- 5.1 Visual Components 12
- 5.2 Kokemuksia 3DCreatesta 12
- 5.3 Ohjelman esittely 14
- 5.4 Create-välilehti 16
- 5.5 Simulaatiomallin luonti 3DCreatella 19

6 PYTHON 21

7 SIMULAATIOMALLIN TOTEUTUS 23
- 7.1 Projektin aloitus 23
- 7.2 Toteutusprosessi 24
- 7.2.1 Simulaatiomallin komponentit 25
- 7.2.2 Komponenttien mallinnus ja ohjelmointi 30
- 7.2.3 Simulaatiomallin kasaus ja testaus 33

8 YHTEENVETO 36

LÄHTEET 38

1 JOHDANTO

Tehdassimuloinnissa luodaan tehtaan laitteistosta tai sen osasta tietokonepohjainen simulaatiomalli. Tehdassimuloinnilla saavutetaan useita käytännön etuja. Simulaatiomalleja voidaan hyödyntää niin järjestelmien suunnittelussa kuin valmiiden järjestelmien pullonkaulojen selvittämisessä. Järjestelmien suunnitteluvaiheessa simulaatiomalleilla voidaan kustannustehokkaasti ja nopeasti tutkia uusien ideoiden toimivuutta sekä esitellä näitä asiakkaille.

Opinnäytetyössä toteutettiin simulaatiomalli vanerinlajittelulinjastosta. Työn tilaajana toimi Raute Oyj. Opinnäytetyö tehtiin välillä syyskuu 2015 – huhtikuu 2016. Tavoitteena oli tehdä tarkka simulaatiomalli vanerinlajittelulinjastosta, Rauten projektista 8150, joka työn alussa oli vasta suunnitteluvaiheessa. Simulaatiomallin tavoitteena oli, että sitä hyödyntämällä tilaaja voisi tutkia robotin liikenopeuksia ja lajittelulinjaston läpimenoaikoja ennen linjaston kokoonpanoa asiakkaan tiloihin. Jotta simulaatiomallista olisi suurin mahdollinen hyöty, siitä tuli tehdä parametrinen siten, että niin robotin, kuin kuljettimienkin nopeuksia voidaan muuttaa. Myös erikokoisia ja -laatuisia vanerilevyjä piti toteuttaa.

Simulaatiomalli toteutettiin Lahden ammattikorkeakoulun neljännen vuoden projektityönä, jossa opinnäytetyön tekijän lisäksi olivat mukana insinööriopiskelijat Eppu Huusko ja Ossi Ruotsalainen sekä projektityön ohjaajana laboratorioinsinööri Timo Lahtinen. Yhteyshenkilönä Rautella toimi suunnittelupäällikkö Janne Kousa. Opinnäytetyön tekijä toimi projektissa projektipäällikkönä ja vastasi täten kokonaisuudesta. Projektityö aloitettiin syyskuussa 2015 ja saatiin päätökseen huhtikuussa 2016.

Kehitysalustana simulaatiomallille toimi Visual Componentsin 3DCreate, joka on monipuolinen ohjelma 3D-simulaatiomallien luontiin. 3DCreaten simulaatiomalleissa voidaan hyödyntää valmiita kirjastokomponentteja, CAD-malleja tai toteuttaa geometriat itse. Simulaatiomallien ohjelmointikielenä on Python.

Valmis simulaatiomalli koostuu seuraavista komponenteista: viisi kuljetinta, vapaarullasto, Kuka KR500 -robotti, kaksi pinkkaria, kolme nostolavaa, aputarttuja, hoitotaso, robotin tarttuja ja kaksi syöttölaitetta vanerilevyjen luontiin. Simuloidun linjaston läpi kulkee vanerilevyjä, jotka ovat syöttölaitteiden luomia simuloituja tuotekomponentteja. Simulaatiomallissa voidaan muuttaa nopeusasetuksia sekä vanerikokoja ja -laatuja.

2 RAUTE OYJ

Raute Oyj on perustettu vuonna 1908 nimellä Lahden Rauta- ja Metalliteollisuustehdas. Aluksi yritys valmisti sisävesilaivoja, höyrykattiloita sekä höyrykoneita, mutta 1930-luvulta alkaen mukaan tulivat muun muassa kehäsahat ja vanerikoneet. Sotakorvaustuotteiden vienti Neuvostoliittoon toimi pohjana Rauten muuntumiselle vientiyritykseksi. Tuotannon kasvaessa yrityksen tuotanto siirtyi ahtaaksi käyneestä Lahden keskustasta vähitellen Nastolaan nykyiselle paikalleen. Samoin tuotepaletti muuntui vähitellen nykyiseksi. Raute joutui käymään läpi rankan saneerauksen 1990-luvun alussa Neuvostoliiton kaupan romahduksen vuoksi. (Wikipedia 2016b; Lahden Kaupunginmuseo 2016.)

Nykyisin Raute on teknologia- ja palveluyritys, joka palvelee puutuotetoimialaa maailmanlaajuisesti. Raute toimittaa viilun, vanerin ja LVL:n eli viilupalkin valmistuksessa tarvittavia tuotantoprosesseja ja niihin liittyviä palveluja. Suurimmalla asiakasteollisuudenalallaan, vaneriteollisuudessa, Raute on maailmanmarkkinajohtaja. LVL-teollisuudessa yli puolet maailman tuotannosta tuotetaan Rauten toimittamilla koneilla. Raute tarjoaa asiakkailleen kokonaispalvelukonseptin laiteinvestointien koko elinkaaren ajalle. Asiakkaan laitehankkeet toteutetaan projektitoimituksina, olipa kyse yksittäisistä koneista tai koko tehtaan laitteistosta. Laitteiden elinkaarta tuetaan teknologiapalveluilla tarjoamalla varaosia sekä kunnossapito- ja modernisointipalveluita. (Raute Oyj 2016, 2, 16.)

Ilmastonmuutos ja ympäristötietoisuuden kasvu lisäävät Rauten tuotteiden kysyntää globaalisti, koska puun käyttö lisääntyy raaka-aineena ja prosesseilta vaaditaan parempaa energiatehokkuutta ja ympäristöystävällisyyttä. Rauten tuotteilla on mahdollista saavuttaa korkea automaatioaste, jolla voidaan vähentää niin energian kulutusta kuin raaka-ainehukkaakin. Uusien istutuspuulajien prosessointia tutkitaan ja kehitetään ja puutuotteiden laatua ja lujuutta maksimoidaan edistyksellisillä laadutusteknologioilla. Lisäksi Raute tarjoaa liimaa säästäviä vanerin ja LVL:n ladonta- ja liimausratkaisuja. Rauten modernin

mittaus- ja automaatiotekniikan käytöllä voitaisiin maailman vaneritehtaissa säästää puuta jopa puoli miljoonaa kuorma-autollista vuodessa. Samoin energiaa voitaisiin Rauten tekniikalla säästää ison ydinvoimalan lämpötehon verran vuodessa. (Raute Oyj 2016, 8 - 9, 16.)

Myös globaali talouskehitys vaikuttaa puutuotteiden kysyntään ja siten luo Rautelle uusia markkinoita. Kysynnän painopiste markkinoilla siirtyy yhä enemmän kehittyville markkinoille, joissa yleisen elintason noustessa panostukset rakentamiseen ja asumiseen kasvavat. Samalla kasvavat myös laatu- ja ergonomiavaatimukset, jolloin kustannusten noustessa automaation merkitys kasvaa. Rautella on erityisesti kehittyville markkinoille tarkoitettu Dragon-tuoteperhe viilusorvaukseen ja -kuivaukseen, mikä lisää automaatiota kehittyvien markkinoiden tuotannossa sekä edistää viilupohjaisten tuotteiden käyttöä niiden laadun parantuessa. (Raute Oyj 2016, 6, 8.)

Rauten kilpailuetuina markkinoilla toimivat oman alansa johtavat teknologiat ja innovaatiot, vanerin ja LVL:n tuotantoprosessin kokonaisosaaminen, pitkäaikainen kokemus projektitoimituksista kaikille markkina-alueille, kattavat teknologian, palveluiden sekä huolto- ja varaosapalvelujen tarjonnat, paikallinen läsnäolo globaalisti ja hyvä maine. Tavoitteena Rautella on vahvistaa edelleen maailmanlaajuista markkinajohtajuutta ja kuulua johtavien toimittajien joukkoon myös valituilla kehittyvillä markkinoilla. Rauten kilpailijat ovat tyypillisesti paikallisesti tai alueellisesti toimivia yhteen tai muutamaan prosessiin tai teknologiaan keskittyneitä pieniä tai keskisuuria yrityksiä. (Raute Oyj 2016, 10, 16 - 17.)

Puutuotteiden kokonaismarkkinat ovat 150 miljardia euroa vuodessa, ja Rauten asiakkaiden investoinnit ovat noin 600 miljoonaa euroa vuodessa. Näistä investointimarkkinoista Rauten osuus on noin 15 – 20 %. Liikevaihto vuonna 2015 oli 127,3 miljoona euroa, jossa oli kasvua 35 % edelliseen vuoteen verrattuna. Liikevaihdosta projektitoimituksien osuus oli 67 % ja teknologiapalveluiden 33 %. Rautella on noin 650 työntekijää, joista suurin osa työskentelee Nastolassa, jossa sijaitsee päätuotantoyksikkö ja konsernihallinto. (Raute Oyj 2016, 2, 6, 10, 16.)

Lahden ammattikorkeakoulu (LAMK) ja Raute Oyj ovat jo vuosia tehneet yhteistyötä. Opiskelijoille on järjestetty yritysvierailuja Rauten tiloihin Nastolaan, ja Rauten yritysprojekteja on ollut osana opintoja ja opinnäytetöitä. 22.4.2015 päivätyn uutisen (Lahden ammattikorkeakoulu Oy 2015) mukaan LAMK ja Raute Oyj ovat kyseisellä päivämäärällä allekirjoittaneet puitesopimuksen yhteistyön tiivistämisestä. Yhteistyön tarkoituksena on tukea molempien osapuolten kilpailukyvyn kasvattamista, strategisia ja operatiivisia tavoitteita sekä henkilöstön ja opiskelijoiden hyvinvointia. Yhteistyön puitteissa kehitetään Rauten henkilöstön lisä- ja täydennyskoulutusmahdollisuuksia sekä tehostetaan LAMK:n opiskelijoiden harjoittelumahdollisuuksia.

3 SIMULOINTI

> *Simulointi tai simulaatio on todellisuuden jäljittelyä* (Wikipedia 2014).

Simulointi on prosessi, jossa todellisen maailman järjestelmät muunnetaan virtuaalisiksi malleiksi. Näitä malleja käytetään työkaluina testauksessa ja järjestelmien käyttäytymisen arvioinnissa. Laajemmassa katsannossa simulaatio voi olla mitä hyvänsä yksinkertaisesta roolipelistä vaikuttavaan virtuaalimaailman graafiseen esitykseen. Näiden kahden ääripään väliin mahtuu useita erilaisia simulointikategorioita. (Lahtinen, Salmela, Koukka 2012, 577.)

Simuloinnin historia alkoi vuonna 1777 Georges Leclerc de Buffonin neulakokeella, jossa hän tasolle neuloja heittämällä arvioi piin likiarvoa. Hänen työtään jatkoi Pierre Simon Laplace vuonna 1819. Lähes vuosisata myöhemmin vuonna 1908 Guinnessin panimossa työskennellyt William Sealy Gosset yhdisti kehittämänsä analysointimenetelmän Studentin t-jakauman karkeaan manuaaliseen simulaatioon ja kehitti näin teollisen simulaation. Tietokonepohjainen simulaatio kehitettiin 1940-luvulla elektronisten tietokoneiden kehityksen myötä aluksi sotilaallisten sovellusten muodossa. Keith Douglas Tocher kehitti ensimmäisen yleiskäyttöisen tehdassimulaatio-ohjelman GSP (General Simulation Program) 1950-luvulla Southamptonin yliopistossa. (Goldsman, Nance & Wilson 2009, 1 - 2.)

3.1 Simulointi osana opetusta ja opiskelua

Simulointia voidaan hyödyntää monissa eri opetustarkoituksissa, joista tyypillisimmin käytetään laitesimulaattoreita, kuten lentosimulaattoria ja metsäkonesimulaattoria. Toisaalta simulaatioita hyödyntäen voidaan opiskella vaikkapa taktiikkaa, johtamista, vuorovaikutusta ja prosesseja. (Räsänen 2004.) Erilaiset simulointityökalut tarjoavat loputtomasti mahdollisuuksia myös insinöörikoulutuksen parantamiseen (Lahtinen ym. 2012, 577).

Insinöörikoulutuksessa simuloinnin kokoluokka voi alkaa yksittäisestä piiristä päätyen kokonaiseen tuotantolaitokseen. Simulaatiotyökaluilla käytännön oppiminen voidaan suorittaa kustannustehokkaasti ja turvallisesti. Esimerkkeinä PLC-ohjelmoinnin (PLC = programmable logic controller, ohjelmoitava logiikka) harjoittelussa voidaan hyödyntää 3DCreate-simulaatiomalleja oikeiden koneiden sijaan, ABB:n Robot Studio -ohjelmalla voidaan simuloida robottisoluja ja ohjelmoida robotteja ja Feston Fluidsim -ohjelmalla voidaan simuloida hydrauliikkaa ja pneumatiikkaa. (Lahtinen ym. 2012, 577 - 579.)

3.2 Tehdassimulointi

Vaikka länsimaissa olemme ohittaneet teollisen aikakauden ja siirtyneet informaatioaikakaudelle, valmistaminen säilyy tärkeänä osana globaalia taloutta. Viimeisten viiden vuosikymmenen aikana teollisen valmistuksen tehokkuutta on pyritty parantamaan hyödyntämällä simulointia. Vaikka edistystä onkin tapahtunut ja simulaatiomalleja on käytetty kasvavissa määrin tehtäessä päätöksiä valmistusjärjestelmistä, on niiden käyttö yhä satunnaista monissa valmistusympäristöissä. (Fowler & Rose 2004, 1.)

3.2.1 Simuloinnin hyödyt

Modernit korkean teknologian valmistusjärjestelmät voivat olla erittäin monimutkaisia. Monimutkaisuus johtuu useista syistä: esimerkiksi tehdään useita tuotteita samalla linjalla, käytetään lukuisia, jopa satoja valmistusvaiheita ja käytetään hyvin monimutkaista välineistöä. Monimutkaisuuden ja korkeiden kustannusten vuoksi ei ole järkevää luottaa vain kokemuksen tuomaan näkemykseen vaan systeemimallien käyttö on välttämätöntä. Mallien tarkoitus on tukea päätöksiä, mutta yksi malli ei yleensä riitä kaikkien päätösten tueksi, vaan tarvitaan useita malleja erilaisiin näkökantoihin. Tässä voidaan hyödyntää tietokonepohjaisia simulaatiomalleja, jotka antavat tarkempaa ja siten parempaa tietoa, miten monimutkainen valmistusjärjestelmä toimii. (Fowler & Rose 2004, 2.)

Simuloinnilla saavutetaan monia käytännön etuja. Simulaatiota voidaan ajaa moninkertaisella nopeudella reaalimaailmaan verrattuna, jolloin tunnit tiivistyvät minuuteiksi. Simulaatiomalleilla voidaan testata erilaisten komponenttien yhteistoimintaa sekä erilaisia hypoteettisia tai vaarallisia järjestelmiä ilman taloudellisia tai fyysisiä riskejä. Lisäksi on mahdollista toistaa kokeita erilaisilla järjestelmillä samanlaisissa olosuhteissa tai samanlaisilla järjestelmillä erilaisissa olosuhteissa. Kaikki nämä voidaan tehdä tarkasti kontrolloidusti ja monitoroidusti. (Fowler & Rose 2004, 3.)

3.2.2 Simulaatiomallin suunnittelu ja rakentaminen

Simulaatioprosessissa mallin määrittely on tärkeä mutta usein ylenkatsottu osa. Tässä vaiheessa projektin osallistujat tunnistetaan, projektin tavoitteet määritellään ja projektisuunnitelma tehdään. Jos tässä vaiheessa ei olla huolellisia, voidaan valita tarpeettoman yksityiskohtainen simulaatiomalli. Vaikka mallin uskottavuus voi tällöin kasvaa, mallin rakentaminen, virheenkorjaus, ymmärtäminen ja ylläpito voivat vaikeutua. Suunnitteluvaiheessa on siis ensiarvoisen tärkeää määritellä mallin tarkkuus. (Fowler & Rose 2004, 5.)

Kun alustavat työt on tehty ja konseptimalli on suunniteltu, seuraavana tehtävänä on itse simulaatiomallin teko. Tämä käsittää mallintamisen näkökulman valinnan, mallin rakentamisen, mallin verifioinnin ja validoinnin. Oikein valitulla mallintamisen näkökulmalla voidaan saavuttaa merkittäviä etuja niin itse mallin teossa kuin mallin suoritusajoissakin. (Fowler & Rose 2004, 7.)

Mallintamista voidaan lähestyä joko perinteisen tehtäväkeskeisen näkökulman tai vaihtoehtoisen työsykleihin perustuvan eli resurssikeskeisen näkökulman kautta. Tehtäväkeskeisessä näkökulmassa seurataan yksittäisten tuotteiden matkaa prosessiketjun läpi. Jokaiselle simuloidulle tuotteelle luodaan oma tietueensa, johon tallennetaan prosessiajat ja jonka avulla tuotteen kulkua läpi prosessin seurataan. Resurssikeskeisessä näkökulmassa seurataan prosessilaitteiden toimintaa ja tallennetaan yksinkertaista laskennallista tietoa työkierroista ja läpi

kulkevien tuotteiden määristä. Tämä on huomattavasti kevyempää kuin jokaisen läpi menevän tuotteen yksilöllinen seuraaminen. Suurissa ja monimutkaisissa simulaatiomalleissa kannattaakin tilanteen salliessa hyödyntää yksinkertaisempaa ja moninkertaisesti suorituskykyisempää resurssikeskeistä lähestymistapaa. Näitä lähestymistapoja voidaan myös hyödyntää yhdessä siten, että laajemman skaalan resurssikeskeisellä simulaatiomallilla ensin tunnistetaan prosessista keskeisiä parannusmahdollisuuksia, joita sitten tarkemmin tutkitaan tarkemmilla tehtäväkeskeisillä simulaatiomalleilla. (Fowler & Rose 2004, 7 - 9.)

Mallintamisen näkökulman valinnan jälkeen vuorossa on varsinaisen mallin teko. Tämä vaihe voi viedä hyvin paljon aikaa, joten siinä on myös potentiaalia merkittäville ajansäästöille. Tietokonesimuloinnin alkuaikoina useimmat valmistusjärjestelmien simulaatiot olivat tapahtumakohtaisia ja ne oli ohjelmoitu joko konekielellä tai korkeamman tason kielellä, kuten Fortranilla. Tällöin mallit olivat verrattain tehokkaita suorittaa, koska vain halutut tapahtumat oli mallinnettu. Toisaalta jokaiselle uudelle mallille piti kirjoittaa uusi ohjelma, koska uudelleenkäytettävyyttä ei juurikaan ollut. Tätä varten kehitettiin simulaatiokielet, jotka mahdollistivat simulaatiomallien uudelleenkäytön. (Fowler & Rose 2004, 9.)

Mikrotietokoneiden yleistyminen 1980-luvulla johti valmistusjärjestelmien simuloinnin kasvuun. Tämä johti simulaatio-ohjelmayhteisön kehittämään kaksi toisiinsa liittyvää, mutta jokseenkin erilaista lähestymistapaa simulaatiomallien teon tehostamiseksi. Ensimmäisenä tulivat käyttöliittymät, jotka mahdollistivat useamman perusmallin yhdistämisen yhdeksi korkeamman tason malliksi. Toinen lähestymistapa oli simulaattorien kehittäminen. Koska simulaattorit tarjosivat valmiita malleja, saattoi niiden käytöllä säästää merkittävästi aikaa. (Fowler & Rose 2004, 9 - 10.)

Vaikka edellä mainituilla keinolla onkin jo saatu merkittävästi lyhennettyä simulaatiomallien rakennusaikaa, parannuksille on yhä tilausta. Yksi potentiaalinen tapa tehostaa mallien rakennusta on hyödyntää olemassa olevaa dataa esimerkiksi MES-järjestelmistä (Manufacturing Execution

System, suom. valmistuksenohjausjärjestelmä). Toisinaan tämä tieto voidaan sisällyttää simulaatioon ja toisinaan simulaatiomalli voidaan näillä tiedoilla generoida automaattisesti. Automaattisen generoinnin pitäisi myös vähentää mallin verifiointiaikaa, koska ohjelmakoodin testausaika vähenee. Myös animaation käytöllä voidaan vähentää verifiointiaikaa. (Fowler & Rose 2004, 10 - 11.)

3.2.3 Simulaatiomallin käyttö

Yksinkertaisten valmistusjärjestelmien simulaatioiden ajo voi olla hyvinkin nopeaa, mutta monimutkaisemmissa järjestelmissä yhden työsyklin suoritus on voinut viedä tunteja. Tämä on rajoittanut simulaatioajojen ja siten simulaatiosta saatavan datan määrää. Mallintamisen lähestymistavan muuttaminen saattaa auttaa tässä ongelmassa. (Fowler & Rose 2004, 11.) Toisaalta tämän opinnäytetyön simulaatioprosessin kanssa on havaittu, että ohjelmaversion vaihtaminen uudempaan voi ainakin tässä tapauksessa kasvattaa simulaation suoritusnopeutta erittäin paljon. Datan analysointia voidaan tehostaa kuvaajien ja taulukoiden automaattisella generoinnilla, mutta lopulta ihmisen on suoritettava varsinainen analysointityö (Fowler & Rose 2004, 12).

Simulaatiomalleja voidaan hyödyntää monella eri tavalla. Uusia ideoita voi esimerkiksi helposti visualisoida ja testata simulaatiomallin rakentamisen aikana. Simulaatiomalleja voidaan myös hyödyntää PLC-ohjelmien testaukseen tuotannon nopeutusta varten. Myyntihenkilöstö voi hyödyntää parametroituja simulaatiokomponentteja tuotteiden esittelyssä asiakkaille. (Lahtinen ym. 2012, 581.)

4 SIMULOINTI KÄYTÄNNÖSSÄ

Simulaatiomallien käytöstä saadaan merkittäviä hyötyjä projektin alusta asti. Simulaatio-ohjelman on oltava riittävän helppokäyttöinen, ettei ylimääräistä aikaa kulu toissijaisiin asioihin. Jos projektin toteutuksessa käytetään standardiosia, tai vähintään riittävän samankaltaisia osia, voidaan simulaatiomalleja hyödyntää jo tarjousvaiheessa. Tällöin myyjä rakentaa valmiista komponenteista alustavan simulaatiomallin asiakkaalle näytettäväksi ja laskee siitä alustavan tarjouksen. (Lahtinen 2016.)

Esisuunnitteluvaiheessa kannattaa hyödyntää karkeita simulaatiomalleja. Kun on asiakkaan kanssa ensin selvitetty asiakkaan tarpeet, tehdään karkea simulaatiomalli, jota käydään asiakkaan kanssa läpi, ja tarkistetaan, löytyvätkö kaikki halutut toiminnot ja toimilaitteet. Tämän karkean mallin perusteella voidaan jo alustavasti tehdä myös tilatarkastelu ja muokata suunnitelmia tarpeen mukaisiksi. Myös suunnittelijat voivat tässä vaiheessa tehdä huomioitaan ja parannusehdotuksiaan lopullisiin suunnitelmiin. (Lahtinen 2016.)

Kun karkeaan simulaatiomalliin perustuva suunnitelma on hyväksytty, tehdään varsinainen suunnittelutyö. Lopullisten geometrioiden perusteella tehdään uusi tarkempi simulaatiomalli, jota voidaan hyödyntää lukuisissa suunnittelutehtävissä. Mekaniikkasuunnittelija voi tarkistaa omia ajatuksiaan simulaatiomallin avulla, ohjelmoijat voivat hyödyntää simulaatiomallin ohjelmia tai testata PLC-ohjelmiaan simulaatiomallin kanssa. Asentajille voidaan esitellä simulaatiomallista valmiin linjaston toimintaa, ja jos linjastolla on käytössä robotti, voidaan tehdä robotin ulottuvuustarkastelu. (Lahtinen 2016.)

Raute on käyttänyt Visual Componentsin 3D-simulaatiotuotteita vuodesta 2006 asti. Simulaatiomalleja hyödynnetään tutkimuksessa, tuotekehityksessä, myynnissä ja markkinoinnissa. Lisäksi Raute tarjoaa simulaatiopalveluja. (Lahtinen ym. 2012.) Suunnittelupäällikkö Janne Kousan (2016a) mukaan Rauten tulisi tulevaisuudessa hyödyntää simulointia enemmänkin uusien linjakonseptien suunnittelun yhteydessä.

5 3DCREATE

Opinnäytetyö toteutettiin 3DCreate 3D-simulaatio-ohjelmalla, jota käytetään Rautella simulaatiomallien toteuttamiseen.

5.1 Visual Components

Visual Components on suomalainen ohjelmistoyritys, joka on perustettu vuonna 1999 suomalais-amerikkalaisten simulaatioasiantuntijoiden toimesta. Yritys havaitsi ensimmäisten joukossa bisnesmahdollisuuden 3D-simuloinnin valtavirtaistamisessa uudelleenkäytettävin simulaatiokomponentein. (Visual Components 2016f.)

Visual Componentsin tuoteperheeseen kuuluvat 3DRealize, 3DRealize R, 3DSimulate, 3DCreate ja 3DAutomate (Visual Components 2016g). 3DRealize on lähtötason 3D-simulointiohjelma, jolla voidaan nopeasti visualisoida tuotantolinjoja tuhansista valmiista komponenteista. Koska 3DRealize ei vaadi erityisosaamista, se soveltuu erityisen hyvin myyntihenkilöstön käyttöön. (Visual Components 2016c.) 3DRealize R lisää edellisen ominaisuuksiin mahdollisuuden robottipohjaisten mallien luontiin valmiilla yli 800 robotin kirjastolla (Visual Components 2016d). 3DSimulate lisää edellisiin analyysi- ja raportointityökalut helpottamaan mahdollisten pullonkaulojen selvittämistä (Visual Components 2016e).

3DCreate on työkalu 3D-simulointimallien luontiin tyhjästä tai hyödyntäen olemassa olevia CAD-tiedostoja. Lisäksi se tarjoaa ohjelmointityökalut komponenttien ohjelmointiin Python ohjelmointikielellä. (Visual Components 2016b.) 3DAutomate on Visual Componentsin lippulaivatuote erittäin suurien tuotantolinjastojen simulointiin (Visual Components 2016a).

5.2 Kokemuksia 3DCreatesta

Opinnäytetyössä käytettiin 3DCreatea (versiot 2010 ja 2014), joka on mielestäni suhteellisen helposti lähestyttävä ja monipuolinen työkalu 3D-simulaatiomallien luontiin. LAMK:n lehtori Henri Koukan (2016) mukaan

sellainen käyttäjä, joka ei ole aiemmin käyttänyt simulaatio-ohjelmia, saattaa kokea 3DCreaten hieman hankalakäyttöisenä, mutta käytettävyys verrattuna moniin muihin simulaatiotyökaluihin on kohtuullisella tasolla. Projektiryhmän jäsen Ossi Ruotsalainen koki ohjelman monipuoliseksi ja hyvin tuotantolinjojen suunnitteluun ja demonstraatioihin soveltuvaksi (Ruotsalainen 2016).

Ohjelmaversioiden 2010 ja 2014 välillä suurin muutos liittyy suorituskyvyn merkittävään parantumiseen. Opinnäytetyön osalta simulaationopeus kasvaa noin 30-kertaiseksi käytettäessä ohjelmaversiota 2014 version 2010 sijaan. Lisäksi robottien toimintapisteet sisältävä xml-pohjainen rsl-tiedosto ei ole taaksepäin yhteensopiva versiosta 2014 versioon 2010. Koukka (2016) toteaa, että ensimmäisestä käyttämästään versiosta (2007) lähtien ohjelman käyttöliittymä on pysynyt melko samana, mutta kehitystä on tapahtunut ohjelman vakauden ja asioiden toimivuuden suhteen. Hänen mukaansa muutokset ovat olleet ohjelman peruskäytössä melko näkymättömiä, mutta esimerkiksi lisenssimalli on vaihtunut konekohtaisesta verkkolisenssiin, mikä on tärkeää oppilaitosympäristössä.

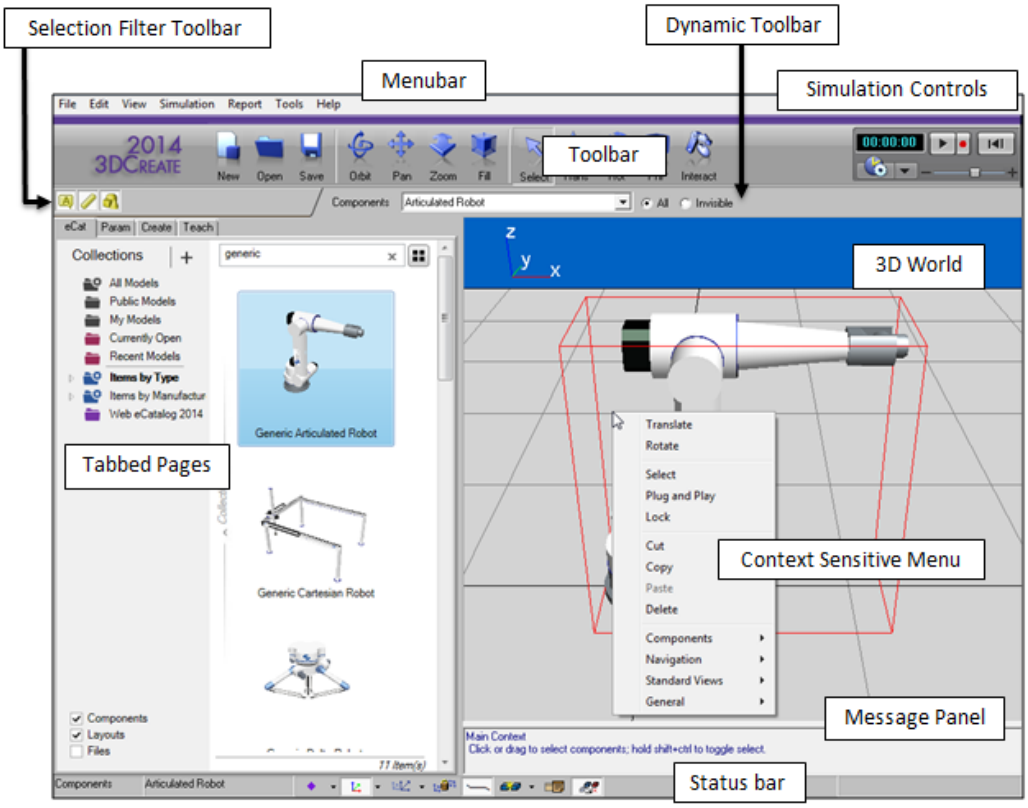
Opiskelijan näkökulmasta 3DCreatessa olisi joitakin selkeitä parannuskohteita alkaen lisensoinnista. Vaikka lyhytaikaisia testilisenssejä onkin saatavilla, olisi parempi, jos 3DCreatesta olisi mahdollista saada ilmainen tai hyvin edullinen lisenssi opiskeluajaksi kotikäyttöön, kuten on monen muun suunnitteluohjelmiston kohdalla. Tämä mahdollistaisi asiasta oikeasti kiinnostuneille kunnollisen perehtymisen ohjelman saloihin ja täten loisi uusia alan asiantuntijoita yritysten käyttöön. Opinnäytetyötä varten Visual Componentsilta saatiin puolen vuoden lisenssit 3DCreatesta kotikoneille. Toinen parannuskohde on käyttöohje, jossa kehitys toki onkin ollut huimaa versioiden 2010 ja 2014 välillä. Moniin toimintoihin kaivattaisiin kuitenkin edelleen kattavampia ohjeita tarkempaa säätöä varten sekä etenkin Pythonin käyttöön enemmän 3DCreate-spesifisiä koodiesimerkkejä.

Ohjelman isoimmat ongelmat peruskäytössä liittyvät kuitenkin epävakauteen ja satunnaisiin vikoihin. Ossi Ruotsalainen (2016) toteaa,

että 3DCreatessa on niin sanottuja mysteerivikoja, jotka häviävät joko käynnistämällä ohjelman tai koko tietokoneen uudestaan, ja näiden takia kuluu turhaa aikaa vianhaussa. Lisäksi useamman 3DCreaten yhtäaikainen käyttö aiheuttaa jumiutumisia ja jopa kaatuilua, pääasiassa tallennettaessa tiedostoja.

5.3 Ohjelman esittely

Koska 3DCreate on toiminnoiltaan monipuolinen, esitellään siitä pääasiassa vain niitä toimintoja, joita on käytetty opinnäytetyön simulaatiomallin luonnissa.



KUVIO 1. Käyttöliittymä (Visual Components 2015a)

Kuviossa 1 esitetään 3DCreaten käyttöliittymä, jonka toiminnot alla selitettyinä:

- Menu bar (valikkopalkki): Päästää käyttäjän käsiksi ohjelman toimintoihin.
- Toolbar (työkalupalkki): Näyttää yleisimmät komennot painikkeina.

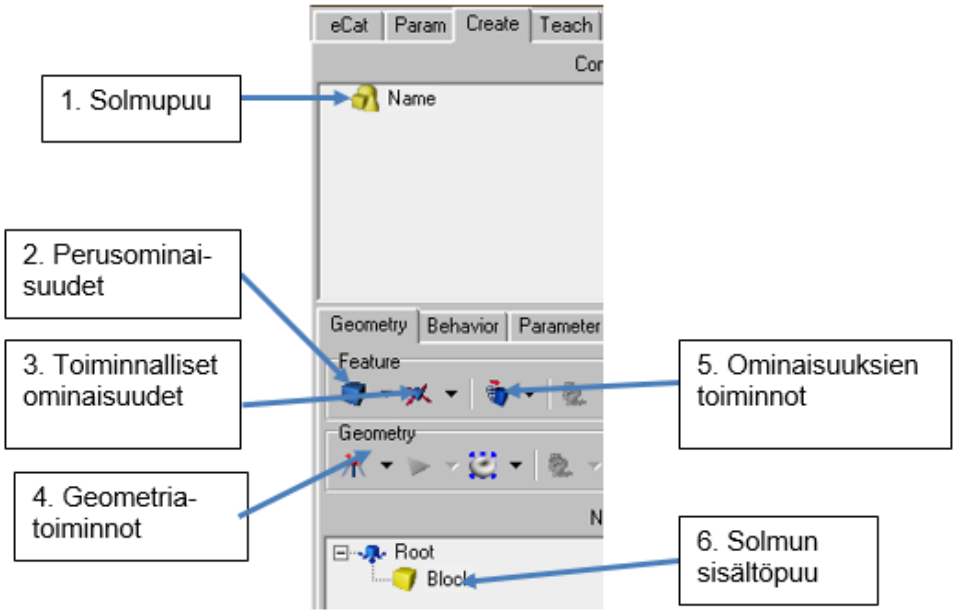
- Simulation Controls (simulaation ohjaus): Ohjataan kuten CD-soitinta.
- Selection Filter Toolbar (toimintojen suodatuspalkki): Valintatapojen suodatus.
- Dynamic Toolbar (Dynaaminen työkalupalkki): Näyttää mahdolliset optiot aktiiviselle komennolle.
- Tabbed Pages (välilehdet): Näyttää kontekstiriippuvaiset työkalut.
- 3D World (3D-maailma): On täysin kolmiulotteinen näyttöalue.
- Message Panel (viestipaneeli): Näyttää virheet ja muut viestit.
- Status bar (tilapalkki): Näyttää valintatilan, valitun komponentin nimen sekä sisältää valintapainikkeita.
- Context Sensitive Menu (kontekstiriippuvainen valikko): Näyttää komennot, joita kyseisessä tilanteessa voidaan käyttää. Saadaan esiin hiiren kakkospainikkeella.
  (Visual Components 2015.)

3DCreatea käytettäessä lähes kaikki toiminnot alkavat oikean välilehden valinnalla (kuvio 1, Tabbed Pages). Ensimmäisenä vasemmalta lukien on eCat-välilehti, joka toimii komponenttien, layoutien ja tiedostojen sähköisenä kirjastona (Visual Components 2015). Tältä välilehdeltä siis valitaan käyttöön halutut valmiit komponentit, layoutit ja tiedostot. eCat-kirjasto pitää sisällään niin käyttäjän itse luomat, kuin ohjelman mukanakin tulevat mallit, joita on vajaat 2000, esimerkkeinä useiden valmistajien robottimallistot.

Param-välilehti näyttää valitun komponentin muokattavat ominaisuudet. Jos mitään komponenttia ei ole valittuna, on näkymä tyhjä. Create-välilehti tarjoaa toiminnallisuudet komponenttien luonnille. Kun tämä välilehti on valittuna, on ohjelma asetettuna luontitilaan. Koska tämä välilehti on olennainen simulaation luonnissa, se esitellään tarkemmin seuraavassa luvussa. Teach-välilehdeltä löytyy toiminnat robotti- ja servo-ohjainten opettamiseen ja ohjelmointiin. (Visual Components 2015.)

5.4 Create-välilehti

Tässä luvussa esitellään pääasiassa toimintoja, joita on käytetty tämän opinnäytetyön teon yhteydessä.

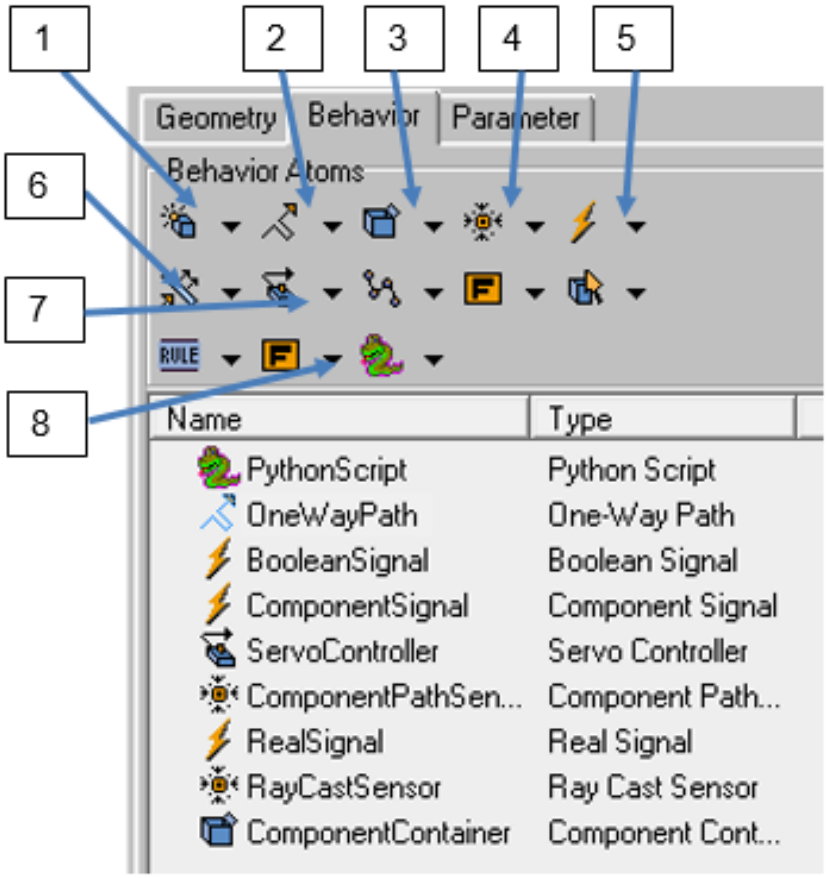


KUVIO 2. Create-Geometry-välilehti (Kuvakaappaus ohjelmasta 3DCreate 2014)

Kuvioon 2 on merkitty Create-välilehden Geometry-välilehden sisältöä seuraavasti:

1. Solmupuu (Component Node Tree) on komponentin hierarkkinen rakenne (Visual Components 2015). ”Name” korvataan yleensä komponentin nimellä, esimerkiksi Kuljetin. Solmupuun solmuihin voidaan laittaa toimintoja Geometry, Behaviour ja Parameter-välilehdiltä (Visual Components 2015).
2. Perusominaisuudet-valikko (Primitive Features) sisältää komponenttien geometrioiden luontiin tarjotut muodot, kuten särmiöt ja kartiot.
3. Toiminnalliset ominaisuudet -valikko (Operation Features) sisältää geometriisin muotoihin liittyviä toimintoja, kuten kloonauksen ja peilauksen.

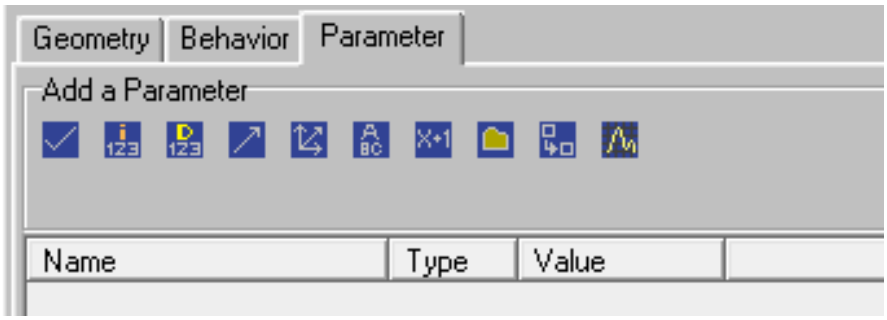
4. Geometriatoimintojen valikoista löytyy lisätoimintoja geometrioihin, esimerkiksi täältä liitetään valmiista CAD-malleista luotuihin komponentteihin materiaalit.
5. Ominaisuuksien toiminnot -valikon (Feature Operations) toiminnolla hajotetaan tai yhdistetään geometrioita.
6. Solmun sisältöpuu (Node Feature Tree) näyttää kunkin solmun sisällön riippuen valilusta välilehdestä (Geometry, Behaviour, Parameter).



KUVIO 3. Create/Behavior-välilehti (Kuvakaappaus ohjelmasta 3DCreate 2014)

Create-välilehden Behavior-välilehdellä annetaan komponentille sen toiminnalliset ominaisuudet, kuten liitännät, polut ja signaalit, sekä kirjoitetaan sen ohjelma Python Script -editorilla. Kuvioon 3 on merkitty Create-välilehden Behavior-välilehden valikot, joiden sisältöön tämän opinnäytetyön teossa on tutustuttu:

1. Toiminnot, joilla ”luodaan” tuotekomponentteja mallikomponenteista simulaatioajon aikana (Creators).
2. Polut (Paths) sisältää toiminnot tuotekomponenttien reiteille.
3. Säiliöt (Containers) sisältää erilaisia säiliöitä, joihin esimerkiksi voidaan kiinnittää tuotekomponentteja simulaatioajon aikana.
4. Sensorit eli anturit, jotka ovat erilaisten objektien tunnistukseen.
5. Signaalit ja signaalikartat (Signal Maps), sisältää erilaiset signaalit, kuten boolean, kokonaisluku ja reaaliluku, joita voidaan käyttää tiedonsiirtoon simulaatiomallissa.
6. Liitännät (Interfaces) sisältää toiminnot, joiden avulla komponentit liitetään toisiinsa layoutissa ja joiden kautta signaalit liikkuvat komponenttien välillä.
7. Ohjaimet (Controllers) sisältää servo- ja robottiohjaimet.
8. Tämä valikko sisältää sekalaisia toimintoja, joista tärkeimpänä tämän työn kannalta Python Script.



KUVIO 4. Create/Parameter-välilehti (Kuvakaappaus ohjelmasta 3DCreate 2014)

Create-välilehden Parameter-välilehdellä voidaan luoda parametroitavia ominaisuuksia, joita sitten hyödynnetään joko suoraan tai Python Scriptin kautta. Opinnäytetyössä käytettiin Boolean-, Integer- ja Real-parametrejä, jotka ovat vasemmalta lukien ensimmäinen, toinen ja kolmas painike. Boolean-parametrilla luodaan valintaruutu, joka joko on tai ei ole valittu. Kokonaisluku- ja reaalilukuparametreilla muutetaan lukuarvoja.

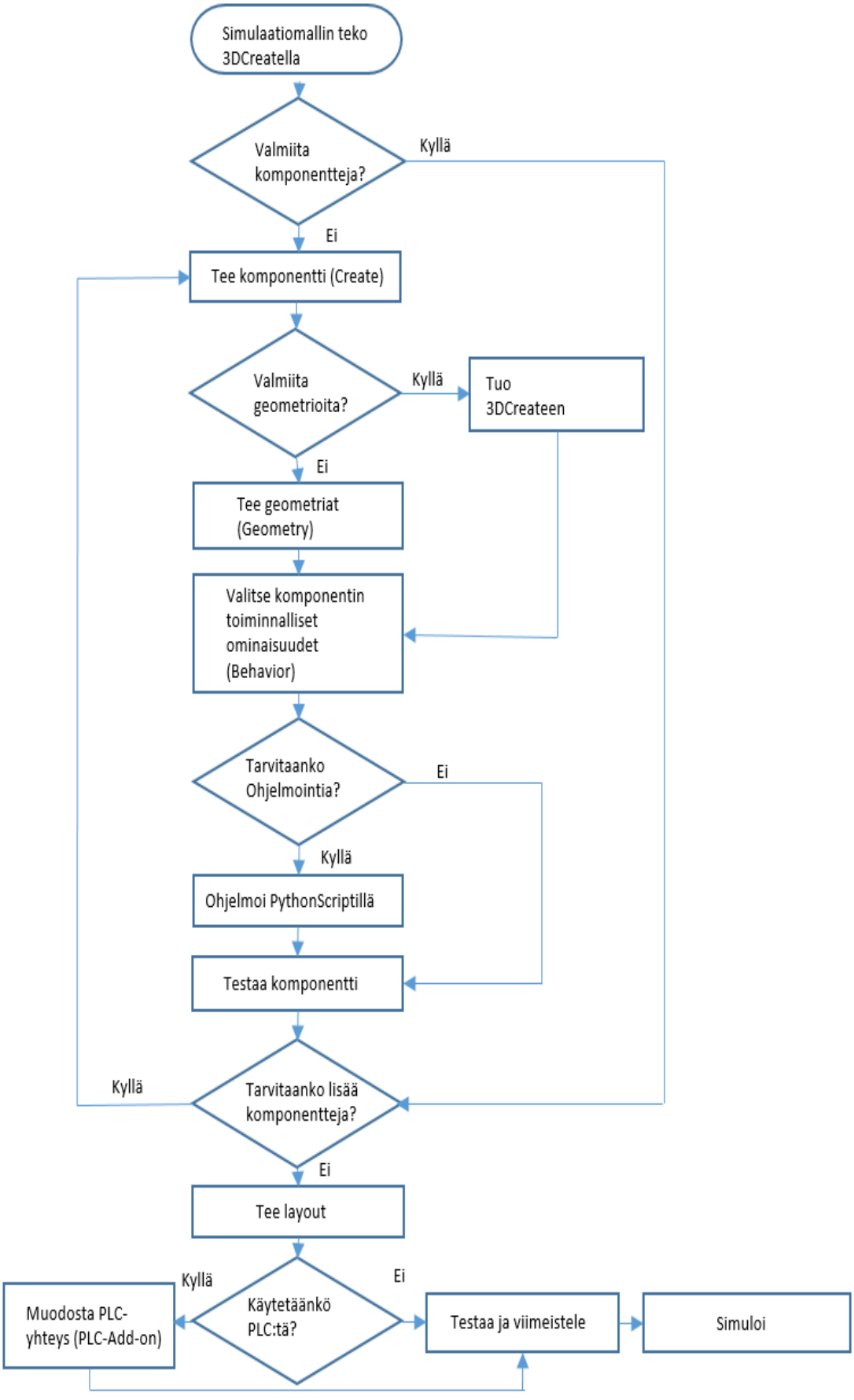
5.5 Simulaatiomallin luonti 3DCreatella

Uuden simulaatiomallin luonti 3DCreatella on varsin suoraviivainen tapahtuma. Kuviossa 5 on esitetty vuokaaviona malli, jonka avulla simulaatiomallin luontia voidaan 3DCreatessa lähestyä. Kaaviosta on jätetty pois 3DCreateen liittymättömät vaiheet, kuten esisuunnittelu.

Karkeasti yksinkertaistettuna työn kulku on seuraava: Ensimmäiseksi tutkitaan, löytyykö suunnitelman mukaisia komponentteja valmiina komponenttikirjastosta. Jos kaikki tarvittavat komponentit löytyvät komponenttikirjastosta, päästään suoraan rakentamaan niistä layout.

Jos komponentit täytyy luoda itse, vaihtoehtoina on käyttää joko muusta suunnitteluohjelmistosta tuotuja CAD-malleja tai luoda geometriat 3DCreatella. Komponentille valitaan toiminnalliset ominaisuudet ja tarvittaessa kirjoitetaan Python Script -ohjelmat. Lopuksi luotu komponentti testataan ja siirrytään tarvittaessa seuraavan komponentin luontiin.

Kun kaikki tarvittavat komponentit on luotu, kootaan niistä layout. Mikäli simulaatiomallin kanssa käytetään PLC-ohjausta, luodaan PLC-yhteys. Lopuksi simulaatiomalli testataan ja viimeistellään, minkä jälkeen aloitetaan simulaatiokäyttö.



KUVIO 5. Yksinkertaistettu vuokaavio simulaatiomallin luonnista

6 PYTHON

Guido van Rossum julkaisi alkuperäisen Python-ohjelmointikielen helmikuussa 1991 USENETissä (Internetin keskusteluryhmät). Nimensä kieli on saanut Monty Pythonin lentävästä sirkuksesta. Nykyään Pythonin kehityksestä vastaa ja oikeudet omistaa voittoa tavoittelematon organisaatio The Python Software Foundation. Pythonin lähdekoodit ovat vapaasti saatavilla osoitteesta https://www.python.org/downloads/. Pythonista kehitetään rinnakkain kahta eri versiota: vanhakantaisempaa 2.x:ää ja uudempaa, osin epäyhteensopivaa 3.x:ää. Näistä versio 2.x on vielä toistaiseksi käyttökelpoisempi paremman sovellus- ja laajennusvalikoiman ansiosta. (The Python Software Foundation 2016.) 3DCreatessa komponentit ohjelmoidaan Python-ohjelmointikielen 2.5-versiolla (Visual Components 2015).

Python on tulkattava, interaktiivinen olio-ohjelmointikieli (The Python Software Foundation 2016). Tulkattavuus tarkoittaa, että ohjelmat ovat heti valmiita ajettaviksi ilman erillistä kääntämistä. Tämä helpottaa ohjelmointia ja testausta, mutta heikentää suorituskykyä verrattuna esimerkiksi C/C++-kieliin (Wikipedia 2016a). Kielenä se sisältää moduulit, poikkeukset, dynaamisen tyypityksen, korkean tason dynaamiset tietotyypit ja luokat (The Python Software Foundation 2016). Dynaaminen tyypitys tarkoittaa, että toisin kuin staattisella tyypityksellä, muuttujan tyyppiä ei tarvitse erikseen esitellä, vaan muuttuja saa tyyppinsä siihen tallennettavan objektin perusteella (Wikipedia 2016a).

Python on korkean tason yleiskäyttöinen kieli, jota voidaan käyttää monissa eri ympäristöissä ja monien eri tason ongelmien ratkaisuissa. Kielen mukana tulee suuri standardikirjasto, joka kattaa muun muassa merkkijonojen käsittelyn, internet-protokollat ja käyttöjärjestelmien rajapinnat. Myös muiden tekemiä laajennuksia on saatavilla laaja valikoima. (The Python Software Foundation 2016.)

Johtuen yksinkertaisesta ja johdonmukaisesta syntaksistaan, Python on hyvä valinta ensimmäiseksi opeteltavaksi ohjelmointikieleksi. Muuttujien

dynaaminen tyypitys poistaa yhden aluksi ylimääräisen mutkikkuuden ja siten nopeuttaa siirtymistä olennaisten ohjelmointitekniikoiden, kuten silmukoiden ja proseduurien opetteluun. Hyödyntämällä laajoja standardikirjastoja Pythonin kanssa päästäänkin nopeasti ohjelmoimaan realistisia sovelluksia. (The Python Software Foundation 2016.)

Seuraavana esitellään 3DCreatella tehty Python Script -esimerkkikoodi. Kyseinen koodi on eräästä kuljettimesta, ja sillä määritellään, koska kuljetin on käynnissä (path.Enabled = True tai False). Käynnistävänä metodina toimii ”triggerCondition(lambda: lahto.Value == True)”, jossa odotetaan robotilta tulevan boolean-tyyppisen signaalin, ”lahto/bRobotti”, tilan muuttumista. Kuljetin pysähtyy, kun se on pyörinyt 18 sekuntia (delay(18)).

```
#-----------------------------------
# Kalle Torniainen
# MEK12
#-----------------------------------
from vcScript import *

def OnSignal( signal ):
  #print signal.Name, "=", signal.Value
  pass

def OnRun():
  comp = getComponent()
  levy = comp.findBehaviour("cRobotti")
  path = comp.findBehaviour("Path")
  kontti = comp.findBehaviour("Container")
  lahto = comp.findBehaviour("bRobotti")

  path.Enabled = False

  while True:
    # Odotetaan signaalin arvon muuttumista
    triggerCondition(lambda: levy.Value != None)
    temp = levy.Value

    # Odotetaan signaalin arvon muuttumista, jonka jälkeen
      käynnistetään kuljetin
    triggerCondition(lambda: lahto.Value == True)
    path.Enabled = True
    path.grab(temp)

    # Viive pyörimiselle, että levynippu ehtii siirtyä pois
      levyjen laskupaikasta
    delay(18)
    lahto.signal(False)
    delay(11)
    path.Enabled = False
    temp = None
```

7 SIMULAATIOMALLIN TOTEUTUS

Simulaatioprojekti toteutettiin kaksivuotisena projektina, jonka ensimmäisen vuoden tärkeimpänä tavoitteena oli oppia Visual Componentsin 3DCreate-ohjelmiston sujuva käyttö. Projektiryhmässä oli alusta asti sama kokoonpano: opinnäytetyön tekijä, Eppu Huusko ja Ossi Ruotsalainen sekä työn ohjaaja laboratorioinsinööri Timo Lahtinen. Ryhmän jäsenten valintaperusteina käytettiin vanhoja näyttöjä sekä aitoa kiinnostusta aiheeseen.

Lähtötilanteessa kukaan ryhmän opiskelijajäsenistä ei omannut simulointikokemusta eikä merkittävää ohjelmointitaitoa lukuun ottamatta koulussa opittu PLC-ohjelmointi. Koska simulaatiomallien tekeminen oli aiheena kaikille uusi (kuuluu vasta neljännen vuoden opintoihin), lähdettiin liikkeelle 3DCreaten perusteista. Aluksi opeteltiin komponenttien mekaniikkojen mallintamista ja niiden ohjelmointia Python-kielellä. Lisäksi harjoiteltiin servojen käyttöä, matriisiohjausta sekä erilaisten signaalien ja anturien käyttöä.

Kun perusteet oli omaksuttu, tehtiin ensimmäisen vuoden aikana kaksi projektityötä, joiden tavoitteena oli yhdessä tarjota interaktiivinen, helppokäyttöinen opiskeluympäristö PLC-ohjelmoinnin opiskeluun. Tämä toteutettiin Siemens S7-300 -ympäristössä hyödyntäen 3DCreaten PLC-Addonia, joka mahdollistaa simulaatiologiikan kytkennän 3DCreaten signaaleihin. Opiskeluineen näihin käytettiin vajaat 200 tuntia tekijää kohden (jokainen teki oman versionsa).

Varsinainen opinnäytetyöprojekti alkoi käynnistyä opinnäytetyön ohjaavan opettajan Teijo Lahtisen neuvoteltua asiasta Raute Oyj:n kanssa huhtikuussa 2015. Toukokuussa sain ensimmäisen alustavan linjamallin ja projektin aloitusajaksi sovittiin elokuu 2015.

7.1 Projektin aloitus

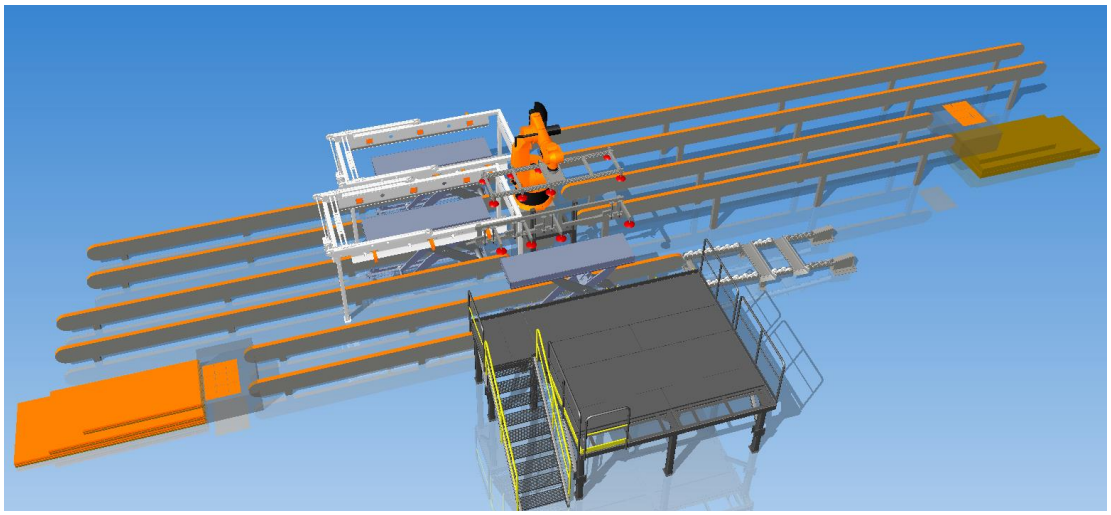
Projekti käynnistettiin aloituspalaverilla Rauten tiloissa 26.8.2015. Tällöin sovittiin Rauten yhteyshenkilön, Janne Kousan kanssa projektin

suuntaviivoista ja siitä miltä toteutuksen tulisi näyttää. Tämän jälkeen Rautelta toimitettiin linjaston toimintakuvaus, ensimmäinen oikea tasopiirustus ja komponenttien 3D-mallit.

Projektin toteutus käynnistettiin 11.9.2015 koulun lukuvuoden alkaessa. Ensimmäiset kaksi viikkoa käytettiin pääasiassa tarvittavien taitojen kertaukseen sekä samalla Master-vaihto-oppilaan, Sebastian Krummin, opastukseen 3DCreaten käytössä hänen omaa projektiaan varten.

7.2 Toteutusprosessi

Toteutusprosessi aloitettiin viikolla 40/2015 jakamalla tehtävät siten, että jokaiselle saatiin sopivasti haasteita ja tekemistä. Opinnäytetyön tekijän tehtäviksi tulivat kokonaisuuden hallinta, nostolavat, Kuka KR500 -robotin ohjelmointi, kuljettimien ohjelmointi, 3DCreate-layoutin (3D-linjamalli) suunnittelu ja käyttöliittymä. Valmis 3D-linjamalli, eli simulaatiomalli, on esitetty kuviossa 6. Ossi Ruotsalainen vastasi kuljettimien mallintamisesta, robotin aputarttujasta, robotin työkalun mallintamisesta ja syöttölaitteista (ennalta määriteltyjä tuotekomponentteja ”tyhjästä” luova komponentti). Eppu Huusko vastasi pinkkarista ja vapaarullastosta.



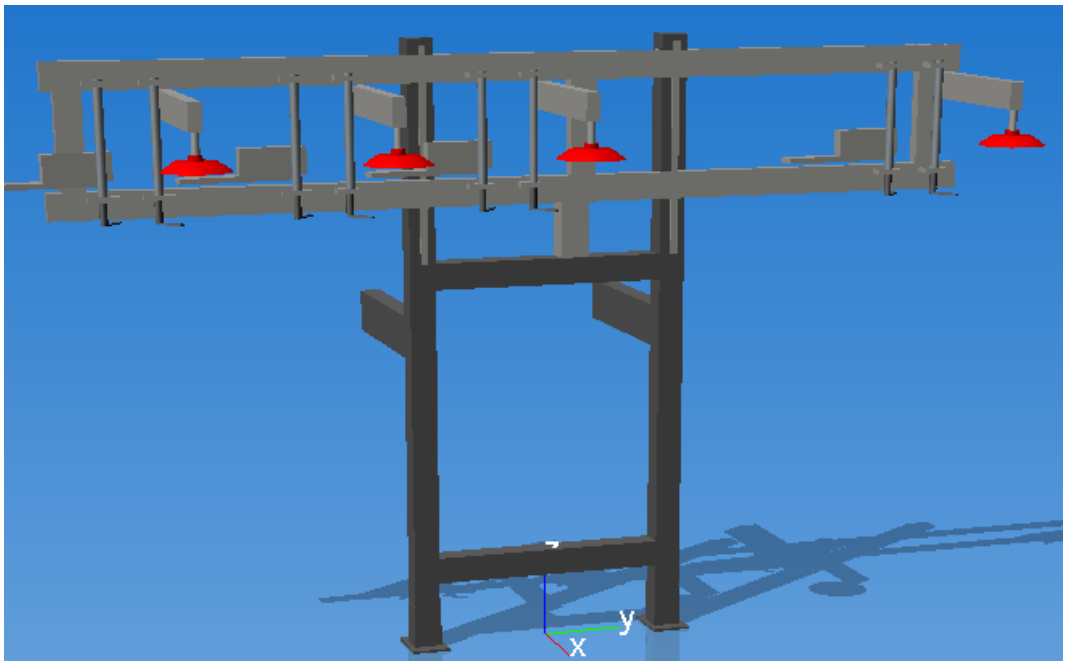
KUVIO 6: Yleiskuva tehdystä simulaatiomallista

Toteutus jaettiin lisäksi kolmeen työvaiheeseen: 1. komponenttien mallinnus ja ohjelmointi, 2. komponenttien kasaaminen yhdeksi simulaatiomalliksi ja 3. simulaatiomallin testaus.

Työn tuloksena syntyi joulukuussa 2015 saadun tasopiirustuksen mukainen simulaatiomalli. Mallissa voidaan parametrisesti asettaa kuljettimien nopeudet ja robotin erilaisia nopeussäätöjä. Erilaisia vanerilevyjä voidaan valita kolmea eri kokoa ja niitä kutakin kolmea eri paksuutta. Vanerilevyjen laatuvaihtelu on määritettävissä 1-laadun, 2-laadun ja ”reject”-laadun suhteen vapaasti valittavin numeroarvoin. Lisäksi mallissa voidaan asettaa sekunteina operaattorin vanerilevykohtainen tarkastusaika sekä simulaation kesto. Käyttöliittymä parametrien asettamiseen löytyy simulaatiomallin hoitotaso-komponentista. Simulaatiomalli tuottaa lopputuotteena txt-muotoisen raportin, josta käyvät ilmi sekä kyseisen simulaatioajon asetukset että läpimenevien kappaleiden määrät. Komponentit ja toteutusprosessi on tarkemmin kuvattu seuraavissa luvuissa.

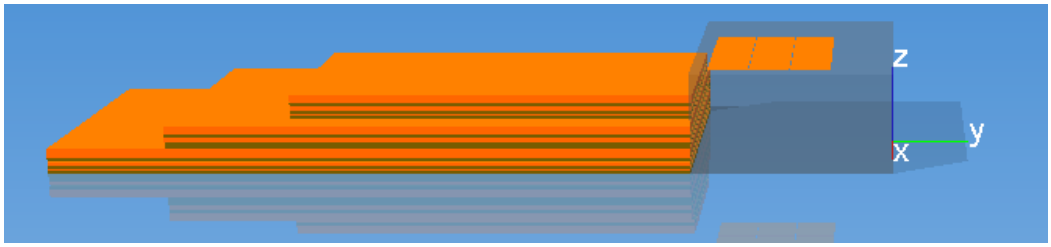
7.2.1 Simulaatiomallin komponentit

Simulaatiomalli sisältää 17 komponenttia, jotka esitellään tarkemmin jäljempänä. Komponentit ovat: viisi kuljetinta, vapaarullasto, Kuka KR500 -robotti, kaksi pinkkaria, kolme nostolavaa, aputarttuja, hoitotaso, robotin tarttuja ja kaksi kappaletta vanerilevyjä luovia syöttölaitteita.



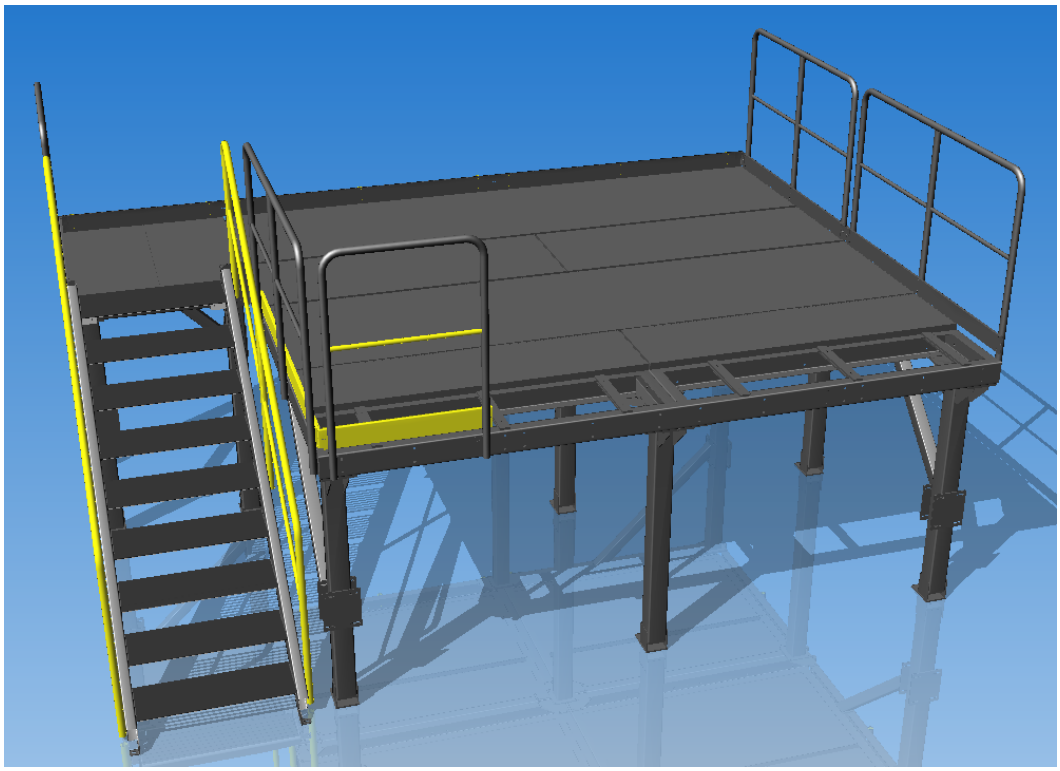
KUVIO 7. Aputarttuja

Kuviossa 7 on aputarttuja, jonka tehtävänä on nostaa vanerilevy irti levypinkasta, koska vanerilevyillä on reaalimaailmassa taipumus takertua toisiinsa.



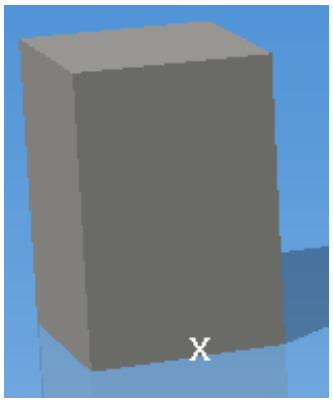
KUVIO 8. VaneriFeeder

Kuviossa 8 on VaneriFeeder eli vanerin syöttölaite. Syöttölaite on komponentti, jolla luodaan tuotekomponentteja ”tyhjästä” ennalta luotujen mallien pohjalta. Tässä simulaatiomallissa syöttölaitteilla luodaan vanerilevynippuja sekä niille aluslevynippuja.



KUVIO 9. Hoitotaso

Kuviossa 9 on hoitotaso. Hoitotaso on simulaatiomallin kannalta passiivinen komponentti. Simulaatiomallin käyttöliittymä rakennettiin toimimaan hoitotason kautta.

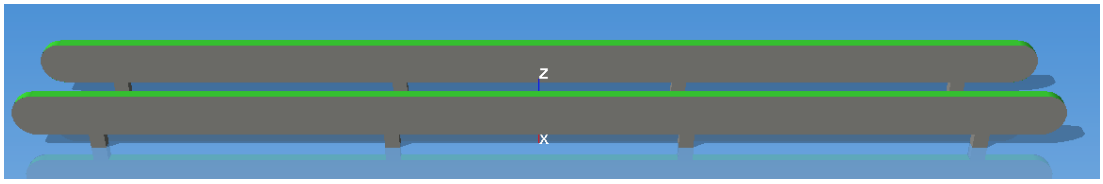


KUVIO 10. Robotin jalusta

Kuviossa 10 on robotin jalusta. Tässä simulaatiomallissa se on maailman keskipiste, johon muiden komponenttien sijainti on sidottu. Se toimii myös signaalikeskittimenä robotin ja muiden komponenttien välillä.

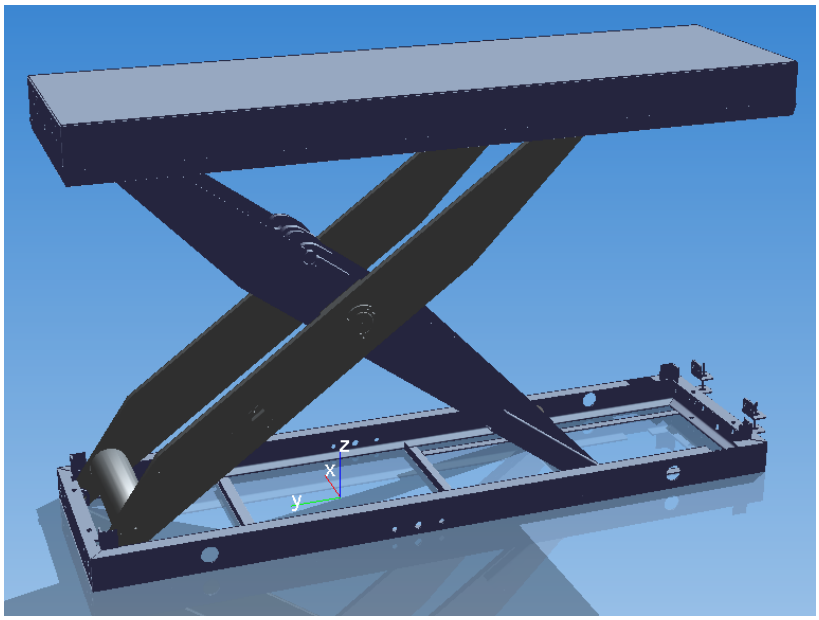
KUVIO 11. Kuka KR500 -robotti ja tarttuja

Kuviossa 11 on simulaatiomallissa käytetty Kuka KR500 -robotti, joka on tarjolla 3DCreate 2010:ssä kirjastokomponenttina. Robotissa on kiinni lisäksi imukuppitoiminen alipainetarttuja.



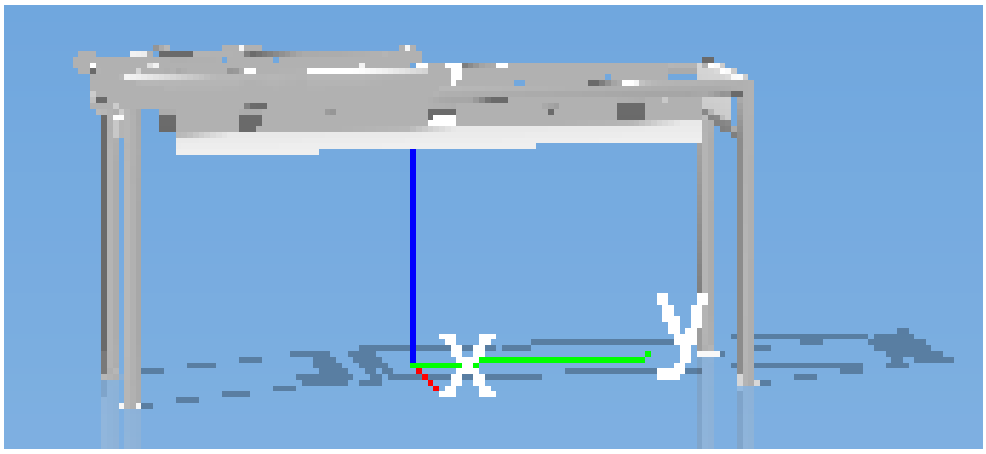
KUVIO 12. Kuljetin

Kuviossa 12 on yksi simulaatiomallin kuljettimista. Tässä simulaatiomallissa on viisi samankaltaista kuljetinta. Kun simulaatioajo on käynnissä, yläpinnassa näkyvä vihreä väri indikoi, että kuljetin pyörii. Vastaavasti keltainen väri indikoi, että kuljetin on pysähtynyt.



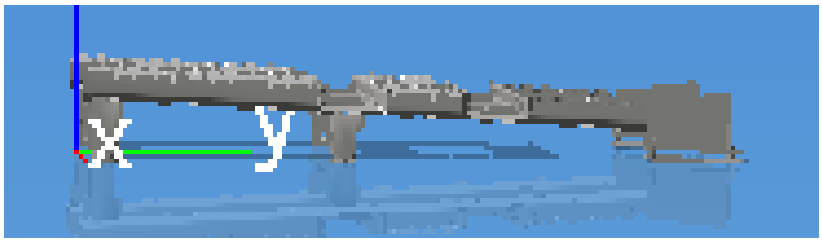
KUVIO 13. Nostolava

Kuviossa 13 on nostolava. Simulaatiomallissa on kolme geometrialtaan identtistä nostolavaa. Syöttökuljettimen yhteydessä oleva nostolava nostaa vanerilevynipun ylös aputarttujan työkorkeudelle. Pinkkarien yhteydessä olevat nostolavat pitävät vanerilevypinon korkeuden vakiona ja lopulta laskevat valmiin pinon alas kuljettimelle.



KUVIO 14. Pinkkari

Pinkkareita (kuvio 14) on simulaatiomallissa kaksi kappaletta. Pinkkari tasaa nostolavalle pudotetut vanerilevypinot hyödyntäen kuutta sylinteriä ja yhtä tasauslevyä.



KUVIO 15. Vapaarullasto

Saapuvien vanerinippujen tyhjät aluslevyt siirtyvät syöttökuljettimelta vapaarullastolle (kuvio 15).

7.2.2 Komponenttien mallinnus ja ohjelmointi

3DCreatessa komponenttien tekeminen alkaa aina niiden mekaniikan mallintamisesta. Komponenttien mallinnus voidaan suorittaa joko 3DCreatella itsellään tai siihen voidaan tuoda valmiita 3D-malleja muista ohjelmista, kuten SolidWorksista. Projektin käynnistyessä harkittiin kumpaakin vaihtoehtoa.

Ensimmäisen vaihtoehdon etuna voidaan pitää parempaa oppimistulosta, mutta vastaavasti työmäärä varsinkin monimutkaisten komponenttien osalta kasvaa huomattavasti. Lisäksi 3DCreaten omalla mallinnustyökalulla monien yksityiskohtien mallinnus on joko hankalaa tai täysin mahdotonta, jolloin malleista tulee helposti yksinkertaistettuja. Jälkimmäisen vaihtoehdon etuina ovat ajan säästö sekä varmasti millimetrintarkkojen komponenttien luonti.

Rautelta saamiamme 3D-malleja tutkittiin ja todettiin, että malleja ei sinällään voida suoraan siirtää 3DCreateen komponenteissa olleiden koonti- ja asemointivirheiden vuoksi. Tästä syystä päädyttiin ensin tekemään komponenttien mallinnus 3DCreatella. Pian havaittiin näiden virheiden häiritsevän myös joidenkin komponenttien mitoittamista SolidWorksissa, minkä seurauksena tutkittiin myös muiden 3D-piirtosovellusten, kuten PTC-Creon, käyttöä mitoituksen apuvälineenä, mutta tulokset jäivät laihoiksi.

Viikolla 44/2015 tehtiin huomio, että jos viallisesta 3D-mallista poistetaan virheitä aiheuttavia komponentteja sekä suoristetaan komponentin koordinaatisto aputasojen avulla, voidaan komponentit viedä 3DCreateen toimivina, joskin vähän puutteellisina malleina. Näin toimittiin nostolavojen, pinkkarien, vapaarullaston, hoitotason ja robotin tarttujan kanssa. 3DCreatessa komponentteihin mallinnettiin mahdolliset puuttuvat osat, jos niillä oli simuloinnin kannalta merkitystä. Esimerkiksi nostolavojen sylinterit jäivät tästä syystä pois simulaatiomallista.Tämän jälkeen projektityön toteutus saatiin kunnolla vauhtiin. Mekaniikan mallinnuksen osalta viimeiset komponentit (kuljettimet) valmistuivat viikolla 51/2015.

Komponenttien ohjelmointi aloitettiin viikolla 45 perustoimintojen lisäämisellä. Tutkittiin, voidaanko liikkeitä toteuttaa servoilla vai tarvitaanko matriisiohjausta. Servo-ohjauksen etuina voidaan pitää helppoutta, se perustuu 3DCreaten valmiiseen toimintoon, mutta hankaluutena voidaan pitää useamman yhtäaikaisen liikkeen toteuttamista synkronoituina. Servo-ohjauksessa liikenopeuksille ei myöskään ole rajoituksia. Matriisiohjaus perustuu Python-ohjelmakoodiin, jolloin sen toteutus on työläämpää, mutta sillä toiminnot saadaan synkronoitua tarkasti. Ongelmaksi tulee liikkeiden tarkkuus. Mitä nopeampi on liikenopeus, sitä enemmän on epätarkkuutta, koska liikkuvan kappaleen sijaintia tarkastellaan ohjelmakierron nopeudella. Tästä seuraa siis se, että mitä pidempiä ohjelmat ovat tai mitä raskaampi kokonaisuus on, sitä epätarkempia liikkeet ovat.

Koska sekä nostotasossa, pinkkarissa että aputarttujassa oli kaikissa tarvetta yhtäaikaisen liikkeen mallinnukselle, päädyttiin aluksi käyttämään matriisiohjausta niissä kaikissa. Aputarttujan osalta päädyttiin kuitenkin pian karsimaan pois simulaatiomallin kannalta epäoleellinen, ainoastaan visuaalinen liike, jolloin siitä saatiin helposti servotoiminen.

Viikolla 49/2015 saatiin Rautelta uusi versio linjaston tasopiirustuksesta, mikä aiheutti muutostarpeita niin simulaatiolayoutin komponenttien sijoituksiin, robotin ja pinkkarinkin toimintaan. Alkuperäisellä oletuksella pinkkarin ajateltiin keskittävän vanerilevyt, mutta uuden layoutin

perusteella ajatusmallia ja siten toteutusta jouduttiin muuttamaan. Robotin osalta jalustan korkeus ja osa kohdepisteistä muuttui.

Ohjelmoinnin kannalta haastavimmiksi komponenteiksi osoittautuivat Ossi Ruotsalaisen työstämät vaneri- ja aluslevyjen syöttölaitteet. Koska 3DCreaten valmiita Component Creator/Filler -toimintoja ei onnistuttu saamaan toimimaan halutulla tavalla, kyseiset toiminnot toteutettiin täysin ohjelmallisesti. Alkuperäisenä ideana oli, että pystyttäisiin luomaan minkä kokoisia vanerilevyjä hyvänsä annettujen speksien sisällä. Viikkojen 52/2015 – 1/2016 aikana tehdyissä testeissä kuitenkin ilmeni, että tuntemattomasta syystä johtuen levyjen koot eivät aina muuttuneet halutusti. Lisäksi koska pinkkarilla kappaleiden paikoitus tasopiirustuksen perusteella riippui kappaleen koosta, toimintaa muutettiin siten, että käytettävissä oli yhdeksän erikokoista levyä, joiden koot oli jo valmiiksi muutettu mallikappaleisiin.

Myös pinkkarin toimintavarmuuden kanssa oli ongelmia. Matriisipohjainen ohjaus aiheutti pahoja ongelmia, mutta koska kummankin puolen sylinterien yhtäaikaista liikettä pidettiin tärkeänä, jatkettiin yrityksiä ongelman ratkaisemiseksi. Toiseksi ongelmalähteeksi paljastuivat kyseisen komponentin vaatimat raycast-anturit, jotka riittävällä signaalinpäivitystahdilla hidastivat mallin toimintaa merkittävästi. Yhdistettynä matriisiohjaukseen tuloksena oli liian satunnaisesti ja hitaasti toimiva pinkkari.

Tasopiirustuksen muututtua viikolla 49 pinkkarin toimintaperiaatetta piti vielä muuttaa keskittävästä kappaleen koon mukaan paikoittavaksi. Koska matriisiohjaus yhdessä raycast-anturien kanssa osoittautui hankalaksi toteuttaa ja toiminnaltaan liian epätarkaksi ja hitaaksi, päätettiin vaihtaa toteutustapa servo-ohjaukseen viikolla 03/2016. Tämän jälkeen pinkkarista saatiin nopea ja tarkasti toimiva versio valmiiksi viikolla 6/2016.

Robotin ja kuljettimien periaatteellista toimintaa ohjelmoitiin yksinkertaistetun 3D-linjamallin avulla, jossa oli oikea määrä kuljettimia ja siirrettäviin kappaleisiin ohjelmoitu erilaisia laatuja. Tämän testimallin

avulla voitiin helposti testata signaalien liikkeitä robotin tarttujalta kuljettimille, robotin liikekäskyjä ja nopeuksia. Lisäksi kuljettimien ohjelmakoodit saatiin perustoiminnoiltaan valmiiksi.

Myös robotin liikkeiden toteutuksessa jouduttiin miettimään erilaisia vaihtoehtoja, koska toisen hakupisteen ja kaikkien jättöpisteiden korkeuskoordinaatti oli vaihteleva. Vaihtoehtoina olivat esimerkiksi matriisipohjainen laskennallinen metodi ja suhteellinen lineaariliike hyödyntäen raycast-anturia. Koska jälkimmäinen oli erittäin yksinkertainen toteuttaa, valittiin se.

7.2.3 Simulaatiomallin kasaus ja testaus

Alustavia versioita komponenteista saatiin valmiiksi vähitellen siten, että viikolla 50/2015 päästiin aloittamaan simulaatiomallin kasaaminen. Tässä vaiheessa aloitettiin viimeistelemään komponenttien ohjelmia niiden sujuvaa yhteistoimintaa ajatellen. Koska jokaisella on oma tapansa tehdä komponentteja eikä simulaatiomallin toimintasuunnitelmakaan ollut täysin onnistunut, jouduttiin osiin tekemään muutoksia simulaatiomallin kasaamisessa. Yllätyksenä tuli robotin ja tarttujan osittainen toimimattomuus, koska näiden ohjelmat oli jo testattu testimallilla. Ratkaisuksi tuli lisätä komponenttisignaalien pakkoresetointeja ja laukaisuehtoja ohjelmaan, jolloin toiminta vastasi lähes testimallin vastaavaa.

Joulukuun alussa sovittiin välipalaveri Rauten tiloissa päivämäärälle 18.12.2015. Olin alussa asettanut tavoitteeksi, että alustava simulaatiomalli olisi valmis viikon 51/2015 loppuun mennessä. Robotin ongelmien ja muiden pienten alkuvaikeuksien jälkeen saatiin tehtyä rajoitetusti toimiva 3DCreate-malli, josta vielä puuttui kokonaan pinkkarien työliike, hoitotaso sekä parametrinen käyttöliittymä. Tätä vaillinaista mallia esiteltiin kyseisessä välipalaverissa toimeksiantajalle, joka vaikutti tyytyväiseltä näkemäänsä. Sovittiin tavoitteeksi, että simulaatiomalli valmistuisi tammikuun aikana. Erilaisia ongelmia kuitenkin ilmeni

jatkokehityksen aikana sen verran, että tästä aikataulusta myöhästyttiin lähes kolme viikkoa.

Edellä mainitussa välipalaverissa esitelty simulaatiomalli oli siis vielä varsin karsittu. Tämän mallin jatkotestaus aloitettiin viikolla 52/2015 ja jatkettiin komponentteja vähitellen viimeistellen viikolle 06/2016. Syöttölaitteissa olleen ohjelmavirheen vuoksi simulaatiomallista tuli jokaisen simulaatioajon ja sen jälkeisen tallentamisen myötä entistä raskaampi ja hitaampi, mutta koska tämä tapahtui vähitellen, ei sitä havaittu moneen viikkoon. Tämä hidasti simulaatiomallin toimintaa ja aiheutti ongelmia testauksessa, kunnes Ossi Ruotsalainen huomasi ongelman syyn: Syöttölaitteen vanhan version tunnettu ongelma oli, että se teki ylimääräisiä kappaleita layoutin reunaan. Selvisi, että se oli tehnyt niitä satoja, ja nämä hidastivat mallin toimintaa vähitellen.

Toiseksi merkittäväksi ongelmaksi muodostui ongelmat koulussa käytettävissä olevan 3DCreaten version 2010 kanssa. Versio 2014, joka oli käytössä määräaikaisella lisenssillä projektiryhmäläisten kotikoneilla, ei ollut robotin liikepisteiden suhteen alaspäin yhteensopiva version 2010 kanssa. Tämän lisäksi simulaatiomallin toimintanopeus oli vain murto-osa verrattuna koulun verkon ulkopuoliseen 2014-versioiseen koneeseen. Tämän vuoksi työtä ei enää voitu jatkaa koulun koneilla, vaan ainoastaan kotikoneilla. Muut ongelmat kuitenkin ratkesivat vähitellen, jolloin päästiin käyttöliittymän ja raportointimetodin tekoon.

Tämä viimeinen työvaihe käynnistyi viikolla 06/2016 ja osoittautui toimintatestausten vuoksi yllättävän aikaa vieväksi. Parametrisia asetuksia lisättiin käyttöliittymään vähitellen ja tehdyt muutokset piti aina testata. Samoin oli raportointityökalun teon kanssa. Samalla havaittiin ja korjattiin muistakin komponenteista viimeisiä pieniä virheitä.

Ensimmäinen julkaisukelpoinen simulaatiomalli lähetettiin Rautelle suunnittelupäällikkö Janne Kousan ja tuotepäällikkö Antti Mäkisen arvioitavaksi 19.2.2016 vajaat kolme viikkoa joulukuun palaverissa annettua aikataulua jäljessä. Simulaatiomalliin tehtiin korjauksia Antti

Mäkisen toiveiden mukaisesti: Lisättiin parametrinen säätö robotin akselinopeuksille, kuljettimien nopeuksille ja operaattorin tarkastusajalle. Tämä versio toimitettiin Rautelle 23.2.2016, jonka jälkeen Antti Mäkinen (2016) kommentoi mallin vaikuttavan hyvältä. Viikolla 15/2016 tehtiin simulaatiomalliin vielä viimeisiä pieniä hienosäätöjä liittyen raporttitiedostoon.

Koska kyseinen opinnäytetyö oli myös neljännen vuoden projektityö, piti ratkaista yhteensopivuusongelmat versioiden 2014 ja 2010 välillä, jotta robotin liikepisteet saadaan siirrettyä näiden kahden version välillä ilman kynää ja paperia, ja siten toimiva versio toimitettua myös koululle arvioitavaksi. Ongelma saatiin ratkaistua rsl-tiedoston avulla. Rsl-tiedosto on tekstimuotoinen xml-koodattu tiedosto, johon tallennetaan robotin liikepisteet siirtoa varten. 2014-version tiedoston perusteella muokattiin tarvittavat tiedot 2010-version tiedostoon.

Projektiryhmä pääsi 26.4.2016 tutustumaan Rauten tiloihin koekäyttöä varten rakennettuun vanerinlajittelulinjastoon, jonka suunnitelmien perusteella simulaatiomalli oli toteutettu. Oikean robotin liikeradat poikkesivat osaksi simulaatiomallin vastaavista. Tilaisuudessa Rauten Janne Kousa (2016b) totesi, että koekäytössä robotin liikeratoja jouduttiin rajoittamaan ja muokkaamaan. Lopullinen hienosäätö tehdään asennuskohteessa, jolloin nähdään, voidaanko hyödyntää simulaatiomallin liikeratoja ja -nopeuksia.

8 YHTEENVETO

Työssä toteutettiin Raute Oyj:n vanerinlajittelulinjaston simulaatiomalli. Tavoitteena oli tehdä joustava työkalu linjaston läpimenoaikojen tarkasteluun sekä tutkia 3DCreate-ohjelmiston hyödyntämistä tällaisessa tehtävässä. Työ toteutettiin kolmen insinööriopiskelijan voimin kaksivuotisena opiskeluprojektina, josta ensimmäinen vuosi käytettiin 3DCreaten opiskeluun.

Tuloksena syntyi resurssikeskeinen simulaatiomalli, jossa voidaan seurata vanerilevyjen läpimenoaikoja koko linjaston läpi. Rauten suunnittelupäällikön Janne Kousan (2016a) mukaan simulaatiomallin tulokset vaikuttavat oikeilta ja realistisilta ja simulaatiomalli oikeiden arvojen pohjalta laaditulta ja luotettavalta. Täten toimeksiantaja on tyytyväinen työn tulokseen. Oikean linjaston koekäyttöversioon nähden simulaatiomallin robotin liikeradat olivat kuitenkin osin poikkeavat.

3DCreate osoittautui aloittelevien käyttäjien silmissä suhteellisen nopeasti omaksuttavaksi ja monipuoliseksi työkaluksi, jota voi hyödyntää hyvin tuotantolinjojen suunnittelussa ja demonstroinnissa. Ongelmia ilmeni eri versioiden yhteensopivuuksissa, version 2010 suorituskyvyssä ja ohjelman vakauden kanssa. Positiivista oli ohjelman selkeys ja monipuolisuus. Ohjelmalta toivottiin kattavampaa ja helppolukuisempaa ohjeistusta.

Suurimpina haasteina simulaatiomallin luonnissa olivat ongelmat valmiiden geometrioiden tuonnissa 3DCreateen, joista selvittiin muokkaamalla niitä SolidWorksilla ennen tuontia, ongelmat joidenkin komponenttien matriisiohjauksissa, jotka ratkaistiin siirtymällä servo-ohjaukseen, sekä ongelmat syöttölaitteiden toteutuksessa, joita ratkaistiin niin ohjelmallisesti kuin kiinteillä geometrioillakin. Erityisesti jouduttiin työstämään pinkkarien ja syöttölaitteiden toimintaperiaatteita.

Aikataulullisesti projektissa onnistuttiin melko hyvin. Lopun testaus- ja viimeistelyvaihe vei odotettua enemmän aikaa, mistä opittiin, että testaukseen on varattava tarpeeksi aikaa.

Opinnäytetyön tekemisen tärkeintä antia oli simulointimallien tekoprosessin oppiminen ja hyvän tiimin kanssa työskenteleminen. Simulointityön tekeminen on ollut erittäin mielenkiintoista, motivoivaa ja mieluista. Simulaatiomallien tärkeyttä ei voida liikaa korostaa osana teollisuusprosessien tuotekehitystä ja suunnittelua.

LÄHTEET

Fowler, J. & Rose, O. 2004. Grand Challenges in Modeling and Simulation of Complex Manufacturing Systems. Saatavissa: http://www.thesimguy.com/GC/papers/scsgc-02.pdf

Goldsman, D., Nance, R. & Wilson, J. 2009. A brief history of simulation. Teoksessa Rossetti, M. D., Hill, R. R., Johansson, B., Dunkin, A. & Ingalls, R. G. (toim.) Proceedings of the 2009 Winter Simulation Conference. Saatavissa: www.informs-sim.org/wsc09papers/028.pdf

Koukka, H. 2016. Lehtori. Lahden ammattikorkeakoulu Oy. Haastattelu 12.4.2016.

Kousa, J. 2016a. RE: Simulaatioprojekti 8150 [sähköpostiviesti]. Vastaanottaja Torniainen, K. Lähetetty 30.3.2016.

Kousa, J. 2016b. Suunnittelupäällikkö. Raute Oyj. Haastattelu 26.4.2016

Lahden ammattikorkeakoulu Oy. 2015. Kumppaneina kohti entistä vahvempaa kilpailukykyä [viitattu 15.4.2016]. Saatavilla: http://www.lamk.fi/ajankohtaista/Sivut/kumppaneina-kohti-entista-vahvempaa-kilpailukykya.aspx

Lahden Kaupunginmuseo. 2016. Metalliteollisuus sotien jälkeen [viitattu 28.3.2016]. Saatavissa: http://www.lahdenmuseot.fi/kuka-mita-lahti/lahden-historia/teollisuuskaupunki/metalliteollisuus-sotien-jaelkeen/

Lahtinen, T. 2016. Laboratorioinsinööri. Lahden ammattikorkeakoulu Oy. Haastattelu 16.3.2016.

Lahtinen, T., Salmela, A., Koukka, H. 2012. Enhancing engineering education and university-industry collaboration by simulation tools. Teoksessa Brörkqvist, J., Laakso, M-J., Roslöf, J., Tuohi, R. & Virtanen, S. (toim.) International Conference on Engineering Education 2012 – Proceedings. Saatavissa: http://julkaisumyynti.turkuamk.fi/filemanager/productfiled/1163file3Upload.pdf

Mäkinen, A. 2016. RE: Simulaatioprojekti (8150). Vastaanottaja Torniainen, K. Lähetetty 25.2.2016.

Raute Oyj. 2016. Vuosikertomus 2015 [viitattu 10.4.2016]. Saatavilla: www.raute.fi/documents/10157/866789/Raute2015_FI.pdf/7301d3db-10be-42a6-b2b7-a1b4dfc0ed98

Ruotsalainen, O. 2016. 3DCreate. Vastaanottaja Torniainen, K. Lähetetty 19.4.2016.

Räsänen, S. 2004. Verkko-opetuksen tietotekniikkaa – Simulaatio opetuksessa [viitattu 17.4.2016]. Kuopion Yliopisto, Tietojenkäsittelytieteen laitos. Saatavissa: http://www.cs.uku.fi/tutkimus/publications/reports/B-2004-3.pdf

The Python SoftWare Foundation. 2016. General Python FAQ [viitattu 3.4.2016]. Saatavissa: https://docs.python.org/3.5/faq/general.html

Visual Components. 2015. 3DCreate 2014 SP3 (ohjelma). User Manual and Reference Guide. Saatavissa (demo): http://www.visualcomponents.com/downloads/#2

Visual Components. 2016a. 3DAUTOMATE [viitattu 17.4.2016]. Saatavissa: http://www.visualcomponents.com/products/3dautomate/

Visual Components. 2016b. 3DCREATE [viitattu 17.4.2016]. Saatavissa: http://www.visualcomponents.com/products/3dcreate/

Visual Components. 2016c. 3DREALIZE [viitattu 17.4.2016]. Saatavissa: http://www.visualcomponents.com/products/3drealize/

Visual Components. 2016d. 3DREALIZE R [viitattu 17.4.2016]. Saatavissa: http://www.visualcomponents.com/products/3drealizer/

Visual Components. 2016e. 3DSIMULATE [viitattu 17.4.2016]. Saatavissa: http://www.visualcomponents.com/products/3dsimulate/

Visual Components. 2016f. About us [viitattu 6.3.2016]. Saatavissa: http://www.visualcomponents.com/about-us/

Visual Components. 2016g. Products [viitattu 17.4.2016]. Saatavissa: http://www.visualcomponents.com/products/

Wikipedia. 2014. Simulointi [viitattu 6.3.2016]. Saatavissa: https://fi.wikipedia.org/wiki/Simulointi

Wikipedia. 2016a. Python (ohjelmointikieli) [viitattu 3.4.2016]. Saatavissa: https://fi.wikipedia.org/wiki/Python_%28ohjelmointikieli%29

Wikipedia. 2016b. Raute [viitattu 28.3.2016]. Saatavissa: https://fi.wikipedia.org/wiki/Raute