SeAMK
SEINÄJOEN AMMATTIKORKEAKOULU
SEINÄJOKI UNIVERSITY OF APPLIED SCIENCES

Petri Kasari

Mallitilojen kopioinnin automatisointi

Opinnäytetyö
Kevät 2023
Insinööri (AMK), Automaatiotekniikka

SEINÄJOEN AMMATTIKORKEAKOULU

Opinnäytetyön tiivistelmä

Tutkinto-ohjelma: Insinööri (AMK), Automaatiotekniikka

Suuntautumisvaihtoehto: Sähköautomaatio

Tekijä: Petri Kasari

Työn nimi: Mallitilojen kopioinnin automatisointi

Ohjaaja: Raine Kauppinen

Vuosi: 2023 Sivumäärä: 45 Liitteiden lukumäärä: 3

Tämän opinnäytetyön tarkoituksena on selvittää ja kehittää talotekniikan tietomallintamisen automatisointiin ohjelma visuaalisella ohjelmointialustalla Dynamolla. Se on ohjelma, jonka avulla voidaan automatisoida samanlaisten huoneiden taloteknisten laitteiden ja putkien piirtoa. Automatisoinnin kehityksen tarve on syntynyt Swecon LVI-kehitysorganisaatiolla. Tarvittiin ohjelma, jonka avulla automatisoinnin avulla tehostettaisiin ja varmistettaisiin suunnittelua ja mahdollistettaisiin mm. luonnosvaiheen suunnitelmien nopea valmistaminen. Sweco Finland Oy on osa kansainvälistä rakennetun ympäristön ja teollisuuden johtavaa asiantuntijayritystä Sweco AB:ta.

Tässä opinnäytetyössä perehdytään aluksi kehitystyön ja ohjelman kannalta tärkeimpiin asioihin, kuten siihen mitä talotekniikan suunnittelu ja tietomallintaminen on ja mitä niissä pitää ottaa huomioon. Tämän jälkeen tutustaan tietomallintamisen tärkeimpiin ohjelmistoihin ja tarkemmin siihen, mitä tässä työssä tehtävän automatisointiohjelman kehittämiseen vaaditaan.

Kehitystyön aikana ohjelman valmistuksessa ilmenneihin ongelmiin löydettiin ratkaisut. Merkittävimmät ongelmat olivat laitteiden yhdistämiseen tarvittavien liittimien puutteelliset tiedot. Liittimien yhdistämiseksi liittimet tuli yksilöidä niiden geometrian perusteella, minkä jälkeen liittimien parien etsintään käytettiin Python-skriptiä. Lopputuloksena on valmis ohjelma, joka vastaa ohjelmalle annettuihin tavoitteisiin. Ohjelman kehitystyö jatkuu tulevaisuudessa Swecon alaisuudessa.

Asiasanat: tietomallit, automaatio, Revit Dynamo, talotekniikka, ohjelmointi

SEINÄJOKI UNIVERSITY OF APPLIED SCIENCES

Thesis abstract

Degree programme: Automation Engineering

Specialisation: Electric Automation

Author: Petri Kasari

Title of thesis: Automating the copying of model rooms

Supervisor: Raine Kauppinen

Year: 2023 Number of pages: 45 Number of appendices: 3

The purpose of the thesis was to discover and develop a program for building service engineering with a programming environment called Dynamo to automate a part of building information modelling. The main object of the program was, that we could, for example, automate the modelling of similar rooms in a project. The need for this kind of development had risen in a Swecos development team. They needed a program that could bring help and quality to modelling building technology. With the program it would also be easier to produce draft stage plans for customers. Sweco Finland Oy is a part of Sweco AB an leading international expert and consulting company in the field of built environment and construction industry.

First the thesis studied the main points of making the program, for example what building service engineering and building information modelling are, and what needs to be considered when working with them. Secondly, the thesis focused on the kind of software needed for building information modelling and more specifically for the automatization program developed during the thesis project. What are the main requirements for developing it and how can they be met.

In conclusion, solutions to the problems that occurred during the development of the program were found. The main problems were the lack of information on the connectors needed to connect the devices. To connect the connectors, it was necessary to identify the connectors based on their geometry and to use Python script to find the couplers for the connectors. As the result there was a working program that achieved the objectives given to it. The development of the program will be continued under Sweco.

Keywords: building service engineering, building information model, automation, program, Revit Dynamo

SISÄLTÖ

Opinnäytetyön tiivistelmä 1
Thesis abstract 2
SISÄLTÖ 3
Kuvioluettelo 5
1 JOHDANTO 8
1.1 Työn tausta 8
1.2 Työn tavoite 8
1.3 Työn rakenne 9
1.4 Sweco Finland Oy 9
2 TALOTEKNIIKAN SUUNNITTELU JA TIETOMALLINTAMINEN 11
2.1 Talotekniikan suunnittelu 11
2.2 Talotekniikan tietomallintaminen 12
2.3 Talotekniikan tietomallivaatimukset 13
2.4 Autodesk Revit 14
2.5 MagiCAD 15
2.6 Autodesk Dynamo 16
3 VISUAALINEN OHJELMOINTI DYNAMOLLA 17
3.1 Dynamon käyttäminen 17
3.2 Listojen käyttö Dynamo-ohjelmoinnissa 19
3.3 Koodilohkot ja DesignScript Dynamossa 21
3.4 Pythonin käyttö Dynamossa 22
3.5 C# käyttö Dynamossa 24
3.6 Dynamo-skriptin luominen ja testaaminen 25
4 MALLITILOJEN KOPIOINNIN AUTOMATISOINTI 28
4.1 Ohjelman vaatimukset ja rajaukset 28
4.2 Mallitilojen kopioinnin periaatteet 28
4.3 Ohjelman haasteet ja rajoitukset 34
5 MALLIHUONE DYNAMON KÄYTTÄMINEN OSANA MALLINTAMISTA . 36

5.1 Dynamo-skriptin käyttöönotto 36
5.2 Ohjelman koekäyttö ja jatkokehitys 37
6 TULOKSET 38
6.1 Työn tavoitteiden toteutuminen 38
6.2 Työssä saadut tulokset 38
7 YHTEENVETO JA POHDINTA 39
7.1 Yhteenveto 39
7.2 Pohdinta 40
LÄHTEET 41
LIITTEET 44

Kuvioluettelo

Kuvio 1. Kolmiulotteinen tietomalli toimistorakennuksesta. 12

Kuvio 2. MagiCADin tuoma toiminto lämpöpatterin ja putkiston painehäviön laskentaan. . 15

Kuvio 3. Dynamon työtilan päänäkymä. 16

Kuvio 4. Dynamon solmuja yhdistettyinä toisiinsa johdoilla. 17

Kuvio 5. MEPover-kirjaston sisältämä solmu, joka tuo tiedot putkien ja osien liittimistä. ... 18

Kuvio 6. Listojen käsittely ja listan kerroksien rajaus. 20

Kuvio 7. Tavallisten solmujen ja koodilohkon vertailu. 22

Kuvio 8. Python-ohjelmakoodi pinta-alojen vertailuun. 23

Kuvio 9. C#-ohjelmakoodi pinta-alojen vertailuun. 24

Kuvio 10. C#-ohjelmakoodi tuotuna Dynamoon. 25

Kuvio 11. Esimerkki solmuista, joilla haetaan halutut tilaelementit projektista. 26

Kuvio 12. Tilojen pinta-alatietojen haku ja vertailu Python-ohjelmakoodilla. 26

Kuvio 13. Revit-tietokannan hierarkiakaavio. 29

Kuvio 14. Ilmanvaihdon päätelaitteen geometriatietojen haku. 30

Kuvio 15. Listat kopioitavista elementeistä ja niiden geometrioista. 31

Kuvio 16. Kanavien lähtötietojen hakeminen ja syöttö piirtosolmulle. 32

Kuvio 17. Laitteiden ja putkistojen liittimien listojen ja liittimien yhdistäminen. 32

Kuvio 18. Ohjelman automatisoinnin lähtötilanne. 33

Kuvio 19. Toimivan ohjelman automatisoinnin lopputulos. 33

Kuvio 20. Dynamo Playerin näkymä Dynamoiden käyttöön. 36

Käytetyt termit ja lyhenteet

3DM	3D-mallitiedostojen tiedostomuoto, jota käytetään monissa eri 3D-suunnitteluohjelmistoissa.
API-rajapinta	Application Programming Interface on ohjelmointirajapinta, joka määrittelee eri ohjelmistojen välisen tiedonvaihdon ja yhteensopivuuden.
BIM	Building Information Modelling on menetelmä rakennusten suunnitteluun, toteutukseen ja ylläpitoon tietomallinnuksen avulla.
CAD	Computer-Aided Design tarkoittaa tietokoneavusteista suunnittelua. Se on sunnitteluprosessi, jossa tietokonetta käytetään suunnittelun ja dokumentoinnin tukena.
ELSE	Tietokoneohjelmoinnin ehtolause, joka mahdollistaa suoritettavien ohjeiden määrittämisen, mikäli IF-ehto ei toteudu.
IF	Tietokoneohjelmoinnin ehtolause, joka suorittaa tiettyjä ohjeita, jos tietty ehto toteutuu.
IFC	Industry Foundation Classes on rakennusalan standardoitu tiedostomuoto. Se on avoin ja kansainvälinen tiedostostandardi, joka mahdollistaa tietomallinen siirtämisen eri ohjelmistojen välillä.
IV	Ilmanvaihto.
Jäähdytinpalkki	Jäähdytinpalkki on ilmastointilaite, joka käyttää ilmanvaihdon ilmavirtauksia levittämään lämpöä tai viilennystä lämmönvaihtimen lävitse huoneeseen.
LJ	Lämmitys ja jäähdytys.
Moduuli	Tietokoneohjelmoinnin käsite, joka tarkoittaa itsenäistä ohjelman osaa, joka sisältää tietyn toiminnallisuuden ja voidaan liittää osaksi laajempaa ohjelmaa.

OBJ	OBJ on yleinen tiedostomuoto 3D-mallinnuksessa ja se sisältää 3D-mallin tiedot, kuten muodon, geometrian ja materiaalit.
ODBC	Open Database Connectivity on standardi, joka mahdollistaa tietokantojen hallinnan ja käytön ohjelmistoista riippumatta.
Puhallinkonvektori	Puhallinkonvektori on ilmastointilaite, joka käyttää puhallinta levittämään lämpöä tai viilennystä lämmönvaihtimen lävitse huoneeseen.
RAU	Rakennusautomaatio.
Skripti	Tietokoneohjelmoinnin käsite, joka tarkoittaa kirjoitettua ohjelmakoodia, joka suoritetaan suoraan sovelluksessa.
VV	Vesi ja viemäri.

1 JOHDANTO

1.1 Työn tausta

Työn taustalla on tarve tietomallintamisen automatisoinnista Revit-mallinnusohjelmistolle. Automatisoinnin kehitystarve on syntynyt Swecon LVI-kehitystiimillä, ja tähän kehitystarpeeseen pyritään löytämään ratkaisu tässä opinnäytetyössä. Automatisoinnilla tässä kehitystarpeessa tarkoitetaan taloteknisten järjestelmien laitteiden, putkien, johtokourujen yms. automaattista piirtoa ja kopiointia mallihuoneen perusteella muihin vastaavanlaisiin huoneisiin. Tässä opinnäytetyössä esitellään ohjelmointiympäristö, jossa automatisointia voidaan tehdä. Lisäksi käsitellään visuaalisen ohjelmoinnin avulla tehtävää Dynamo-ohjelman kehittämistä.

Esimerkki automatisoidun talotekniikan piirron käyttökohteesta on toimistorakennus. Toimistorakennuksissa on tavallista, että toimistotilat ovat samankaltaisia keskenään. Suunniteltaessa tiloja tulee suunnittelijan piirtää jokainen tila erikseen tai kopioida yhden huoneen järjestelmät toisiin huoneisiin käsityönä. Mikäli huoneisiin tulisi merkittäviä muutoksia lähtötietojen muuttuessa, esimerkiksi tilan puhallinkonvektori vaihtuisi jäähdytinpalkkiin, tulisi jokaiseen huoneeseen vaihtaa käsin kyseiset laitteet. Automatisoidulla mallihuoneen kopioinnilla/piirrolla riittää, että tekee tarvittavat muutokset huoneeseen, joka toimii mallina, ja Dynamo-ohjelma hoitaa lopun samankaltaisiin huoneisiin. Automatisoidun piirron avulla helpotettaisiin merkittävästi esimerkiksi luonnosvaiheen suunnitelmien luomista tietomallista ilman pelkoa suuresta korjaustyöstä, joka aiheutuisi suunnitelmien muuttuessa projektin aikana.

1.2 Työn tavoite

Työn tavoitteena on kehittää ja valmistaa ohjelma visuaalisella ohjelmointikäyttöliittymällä Dynamolla, joka vastaisi tietomallintamisen automatisoinnin tarpeisiin. Ohjelman tärkein tavoite on automatisoida talotekniikan tietomallintamisen piirtoa mahdollisimman tarkasti ja laajasti. Lisäksi kehityksen aikana selvitetään mahdollisia piirron automatisointiin liittyviä rajoitteita ja ongelmia.

1.3 Työn rakenne

Luvussa 2 käydään yleisesti läpi tämän opinnäytetyön kannalta oleellisimmat asiat. Luvussa käsitellään sitä, mitä talotekniikan suunnittelu ja talotekniikan tietomallintaminen on, mitä vaatimuksia niillä on ja mitkä ovat tärkeimmät ohjelmistot niiden käytössä.

Luvussa 3 perehdytään tarkemmin visuaalisen ohjelmoinnin työympäristöön Dynamoon. Luvussa kerrotaan mitkä ovat Dynamon tärkeimmät piirteet ja eroavaisuudet perinteisiin ohjelmointitapoihin verrattuna.

Luvussa 4 selvitetään, mitä opinnäytetyöhön tehtävällä ohjelmalla pyritään saavuttamaan ja miten. Käydään läpi ohjelman pääpiirteitä ja sitä, mitä rajoituksia tai ongelmia ohjelman valmistuksen aikana ilmeni. Osa luvun sisällöstä on liitteessä, joka ei ole julkinen.

Luvussa 5 kerrotaan, miten ohjelman käyttöönotto osaksi tietomallintamista tapahtuu, mitä puutteita tai rajoituksia koekäytön aikana ilmeni ja mitkä ovat ensimmäiset kehityksen kohteet ohjelmassa.

Luvussa 6 läpikäydään työn tulokset, mihin tavoitteista päästiin ja mitkä ovat ohjelman kannalta seuraavat askeleet. Ohjelman lähdekoodi on erillisissä liitteissä, jotka eivät ole julkisia.

Luvussa 7 tarkastellaan työn yhteenvetoa sekä sitä, miten työ eteni ja mitä työn aikana ilmeni. Lisäksi pohditaan, mitä työssä olisi voitu tehdä toisin ja miksi.

1.4 Sweco Finland Oy

Sweco Finland Oy on osa kansainvälisen rakennetun ympäristön ja teollisuuden johtavaa asiantuntijayritystä Sweco AB:ta (Sweco, i.a.). Swecon ydinliiketoimintana on tarjota laaja-alaista ja laadukasta suunnittelu- ja konsultointipalvelua. Kansainvälisyyden myötä Sweco toteuttaa kymmeniätuhansia projekteja vuosittain ympäri maailmaa.

Swecolla työskentelee yli 18 500 asiantuntijaa, joista 3 000 työskentelee Suomessa (Sweco, i.a.). Sweco on listattuna Tukholman pörssiin (Nasdaq), ja sen liikevaihto on noin

2,2 mrd. €. Sweco tarjoaa palveluitaan rakentamisen jokaiseen vaiheeseen. Asiantuntijat auttavat projektinjohdon ja rakennuttamisen eri vaiheissa. Arkkitehdit ja eri alojen suunnittelijat laativat tarvittavat arkkitehti-, rakennus-, energia- ja talotekniikkasuunnitelmat.

2 TALOTEKNIIKAN SUUNNITTELU JA TIETOMALLINTAMINEN

2.1 Talotekniikan suunnittelu

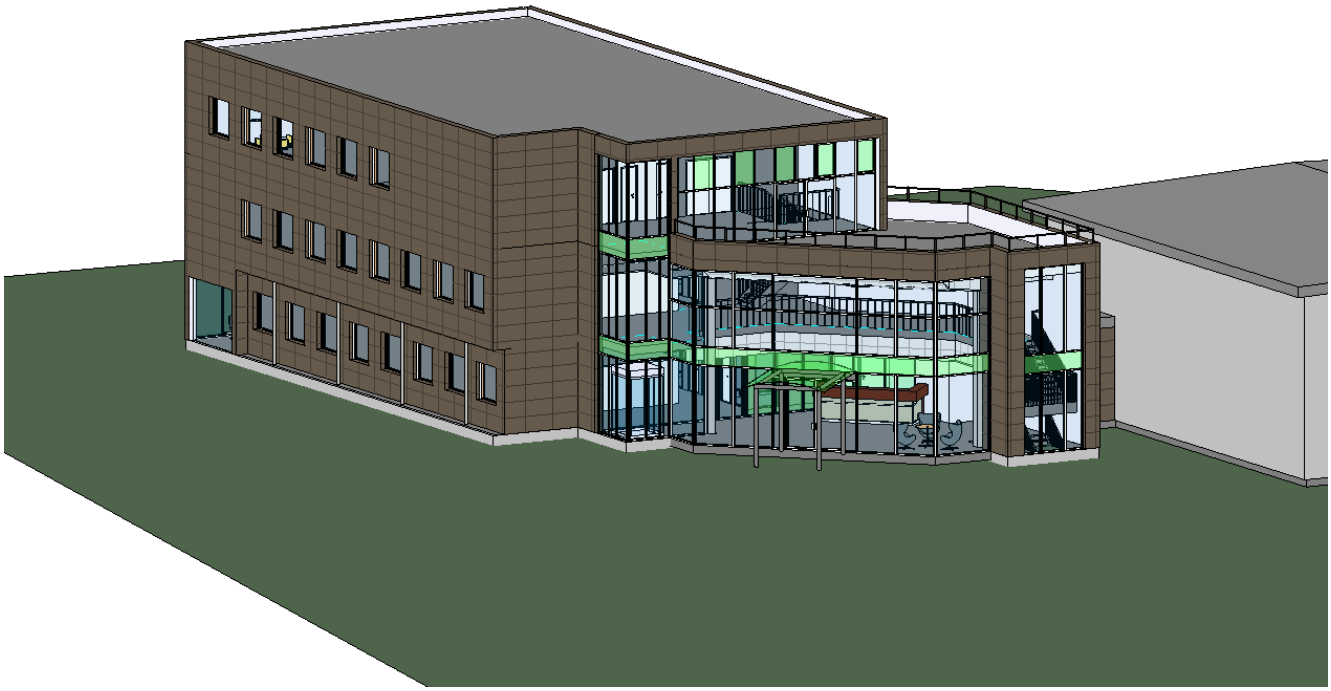
Talotekniikan suunnittelulla tarkoitetaan rakennuksen teknisten järjestelmien, kuten lämmitys-, ilmanvaihto-, sähkö-, rakennusautomaatio-, vesi- ja viemärijärjestelmien suunnittelua. Talotekniikan suunnittelu takaa sen, että rakennuksissa on tärkeät välttämättömyydet, kuten juokseva vesi, lämmitys ja jätevesien viemäröinti (Talteka, i.a.). Ammattitaitoisella suunnittelulla varmistetaan näiden välttämättömyyksien oikeaoppinen toiminta. Talotekniikan suunnittelulla varmistetaan myös mm. sisätilojen olosuhteiden viihtyvyys ja se, etteivät järjestelmät aiheuta toiminnallaan rakennukselle vahinkoa.

Ilmaston lämpeneminen viime vuosikymmenen aikana on muuttanut rakennusten lämmityksen ja jäähdytyksen tarpeita (Ilmatieteenlaitos, 2015). Lämmitystarpeen vähentyessä jäähdytyksen tarve on kasvanut, mikä näkyy jäähdytysjärjestelmien määrän kasvuna. Tämän lisäksi energiatehokkuuden parantaminen mm. rakennusautomaation avulla on lisääntynyt (Talotekniikka-lehti, 2019). Talotekniikan järjestelmien monimutkaistuessa ja laajentuessa tulee eri alojen suunnittelijoiden olla entistä tiiviimmin yhteistyössä keskenään, jotta varmistetaan rakennuksien ja niiden käyttäjien puolesta laadukas ja toimiva lopputulos.

Rakennusten käyttäjien vaatimuksien lisääntyessä talotekniikan suunnittelussa pyritään enemmän ja enemmän käyttäjälähtöiseen suunnitteluun (Rakennusinsinöörien liitto (RIL), 2015, s. 3). Tämä tarkoittaa, että rakennukset ja niiden järjestelmät suunnitellaan niin, että ne toimivat entistä enemmän vuorovaikutuksessa käyttäjän kanssa eli ovat reaktiivisia (RIL, 2015, s. 20). Esimerkiksi isoissa toimistorakennuksissa käyttäjälähtöisillä järjestelmillä mahdollistetaan kaikille työskentelijöille yksilölliset olosuhteet tilakohtaisten mittauksien ja säätöjen avulla. Lisäksi tilakohtaisilla mittauksilla ja säädöillä saavutetaan parempi energiatehokkuus, kun niillä voidaan estää tilojen turhat jäähdytykset ja lämmitykset. Käyttäjälähtöisen ja energiatehokkaan järjestelmän suunnittelussa korostuukin rakennusautomaatiojärjestelmän merkitys (Talotekniikka-lehti, 2019).

2.2 Talotekniikan tietomallintaminen

Talotekniikan tietomallintaminen on osa kokonaisuutta rakennuksen tietomallissa. Tietomallintamista tehdään erilaisilla 3D-mallinnusohjelmilla tai erityisillä tietomallinnusohjelmilla, jotka on tarkoitettu talotekniikan suunnitteluun ja dokumentointiin. Tietomallintamisella tarkoitetaan rakennuksen tietomallintamista, jossa rakennuksesta tehdään digitaalinen ja kolmiulotteinen malli, joka sisältää paljon erilaista tietoa ja dataa rakennetusta ympäristöstä (Solibri, 2022). Kuviossa 1 on Autodesk Revit -mallinnusohjelmistolla tehty tietomalli toimistorakennuksesta. Tietomallintamisesta puhuttaessa käytetään myös yleisesti lyhennettä ”BIM”, joka tulee sanoista Building Information Modelling. Tietomalli voi sisältää esimerkiksi dataa huonetilojen korkeuksista ja pinta-aloista sekä siitä, mitä kaikkea rakennus käytännössä sisältää (Laine, 2008, s. 7).



Kuvio 1. Kolmiulotteinen tietomalli toimistorakennuksesta (Kasari, 2023).

Talotekniikan tietomallintamisella saavutetaan luotettavampi ja tehokkaampi suunnitteluprosessi, esimerkiksi rakennuksen sisäilman ja olosuhteiden laadun varmistamisella erilaisten analysointiohjelmien avulla (Laine, 2008, s. 13). Muita tärkeitä tietomallin tuomia hyötyjä perinteiseen suunnitteluun verrattuna ovat (mts. 13)

- vaihtoehtotarkasteluiden ja simulointien tehokkaampi käyttömahdollisuus
- visuaalisuuden takia suunnitelmat loppukäyttäjän kannalta helpommin luettavia loppukäyttäjälle
- rakennuksen elinkaarikustannusten ja ympäristövaikutusten hallinnan ja rakentamiseen tarvittavien materiaalien laskennan mahdollistaminen ja helpottaminen
- rakentamisen helpottaminen tarkempien ja visuaalisten mallien avulla.

2.3 Talotekniikan tietomallivaatimukset

Vaatimuksilla määritellään esimerkiksi se mitä tietoja tietomalliin tulee sisällyttää ja miten tiedot on esitettävä. Tietomallintamisen tavoitteena on parantaa suunnittelun ja rakentamisen laatua, turvallisuutta ja tehokkuutta sekä tukea kestävän kehityksen mukaista hanke- ja elinkaariprosessia (Building Smart Finland, 2012b, s. 5). Tietomallivaatimuksilla varmistetaan, että näihin tavoitteisiin päästään, ja se, että tietomalli on yhtenäinen ja tiedot vastaavat todellisuutta.

Suunnittelusopimuksissa viitataan lähes poikkeuksetta ”YTV - Yleiset tietomallivaatimukset 2012” -ohjeeseen. Sen on laatinut kehittämishanke COBIM. Tarve tietomallivaatimuksille syntyi rakennusalalla nopeasti kasvavasta tietomallintamisen käytöstä (Building Smart Finland, 2012a, s. 2). Hankeen osapuolina toimii lukuisia suunnittelu- ja asiantuntijayrityksiä, rakennusyrityksiä ja ohjelmistoyrityksiä. ”Yleiset tietomallivaatimukset 2012” tulee saamaan lähitulevaisuudessa merkittävän päivityksen, kun parhaillaan suomennettavana oleva rakentamista koskeva kansainvälinen standardi ISO 19650 julkaistaan (Nordic BIM Group, i.a.-b). Merkittävimpänä muutoksena tulee olemaan, että tuleva ”Yleiset tietomallivaatimukset” tulee entistä enemmän pohjautumaan soveltavin osin kansainvälisiin standardeihin ja näin lisäämään tietomallintamisen yhdenvertaisuutta kansainvälisellä tasolla.

Toinen tärkeä vaatimus tietomallinnuksessa on IFC-standardi. IFC-standardin on kehittänyt ja sitä valvoo voittoa tavoittelematon yhteistyöfoorumi buildingSMART (Nordic BIM Group, i.a.-a). Standardin tarkoituksena on varmistaa avoimen tiedostomuodon avulla eri osapuolien käyttämien ohjelmistojen yhteensopivuus.

Parhaillaan kehitteillä oleva kehityshanke RAVA3pro tulee osaltaan myös vaikuttamaan tietomallintamisen sisällön määrittelyihin (Rava3pro, i.a.). RAVA3pro-hanke on Helsingin kaupungin johtama ja valtiovarainministeriön rahoittama kehityshanke, jossa on lisäksi mukana 23 kuntaa. Kehityshankkeen tarkoituksena on kuntien rakennusvalvonnan sähköisten lupaprosessien kehittäminen ja prosessien automatisointi. Hankkeella pyritään automatisoimaan rakennuslupatarkastusten tarkistaminen ohjelmistolla ja näin nopeuttamaan lupakäsittelyä. Prosessien automatisoinnin mahdollistamiseksi hankkeessa määritellään kansallisten IFC-tietomallien ja rakennusvalvonnan käyttötapauksien tietosisältömäärittelyt sekä keskeisempien ohjelmistojen vaatimukset IFC-mallien hallintaan ja tietojen välityksiin rajapintojen avulla.

2.4 Autodesk Revit

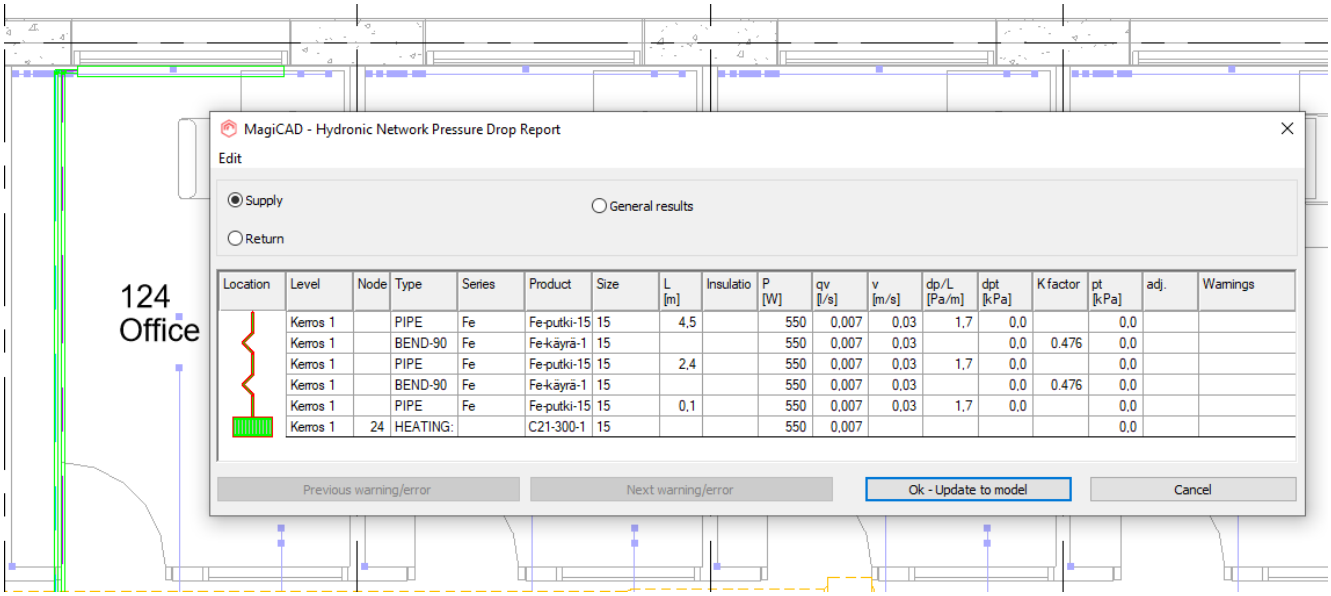
Tietomallintamisessa Autodesk Revit on yksi käytetyimpiä tietomallinnuksen ohjelmistoja (Construction Global, 2020). Autodesk Revit -ohjelmistoa käytetään arkkitehti-, rakenne- ja talotekniikkasuunnitteluun (Autodesk, i.a.-c). Sillä voidaan mallintaa rakenteita ja järjestelmiä kolmiulotteisesti parametriarvoja hyödyntämällä hyvinkin monipuolisesti.

Autodesk Revit -ohjelmiston tärkeimpiä ominaisuuksia ovat (Autodesk, i.a.-b):

- Thteentoimivuus yleisesti käytettyjen BIM- ja CAD-tiedostomuotojen kanssa, kuten IFC, 3DM ja OBJ,
- Parametrinen mallintaminen. Kaikilla mallintamisessa käytetyillä objekteilla on suuri määrä dataa esimerkiksi geometriasta. Parametrinen mallintaminen helpottaa mm. laskentaa ja rakennuksen analysointia,
- Kaikki tietomallin tiedot tallentuvat yhteen tietokantaan, mikä helpottaa tiedostonhallintaa ja parantaa yhteistyön toimivuutta eri suunnittelualojen välillä. Tietomallien tietokantoja voidaan myös linkittää esimerkiksi ODBC-tietokantaan tai Microsoft Exceliin, jolloin tietomallin tietoja voidaan käyttää mm. ulkopuolisissa laskenta- ja kiinteistönhallintatyökaluissa (Autodesk, i.a.-a).
- Kehittäjän työkalut, Revit-ohjelmiston toimintoja ja mallintamista voidaan laajentaa mm. Dynamolla ja API-rajapinnalla, joista tarkemmin tämän luvun kohdassa 2.6.

2.5 MagiCAD

MagiCAD on Autodesk Revit- ja AutoCAD-ohjelmistoihin integroitu lämpö-, vesi-, ilmanvaihto-, sähkö-, sprinkleri- ja rakennusautomaatiosuunnittelun tietomallinnusohjelmisto (MagiCAD, i.a.-a). MagiCAD tuo monipuolisempia ja laajempia työkaluja talotekniikan tietomallintamiseen. Näistä merkittävimpiä ovat erilaisten järjestelmien mallinnus- ja laskentatoiminnot. Esimerkkinä laskentatoiminnoista kuviossa 2 on laskettu lämpöpatterin ja siihen kytkettyjen putkistojen painehäviöt. Lisäksi MagiCADin tuoman kirjaston avulla voidaan käyttää johtavien kansainvälisten laitevalmistajien objekteja, jolloin saadaan tietomallista tarkempi ja täsmällisempi.



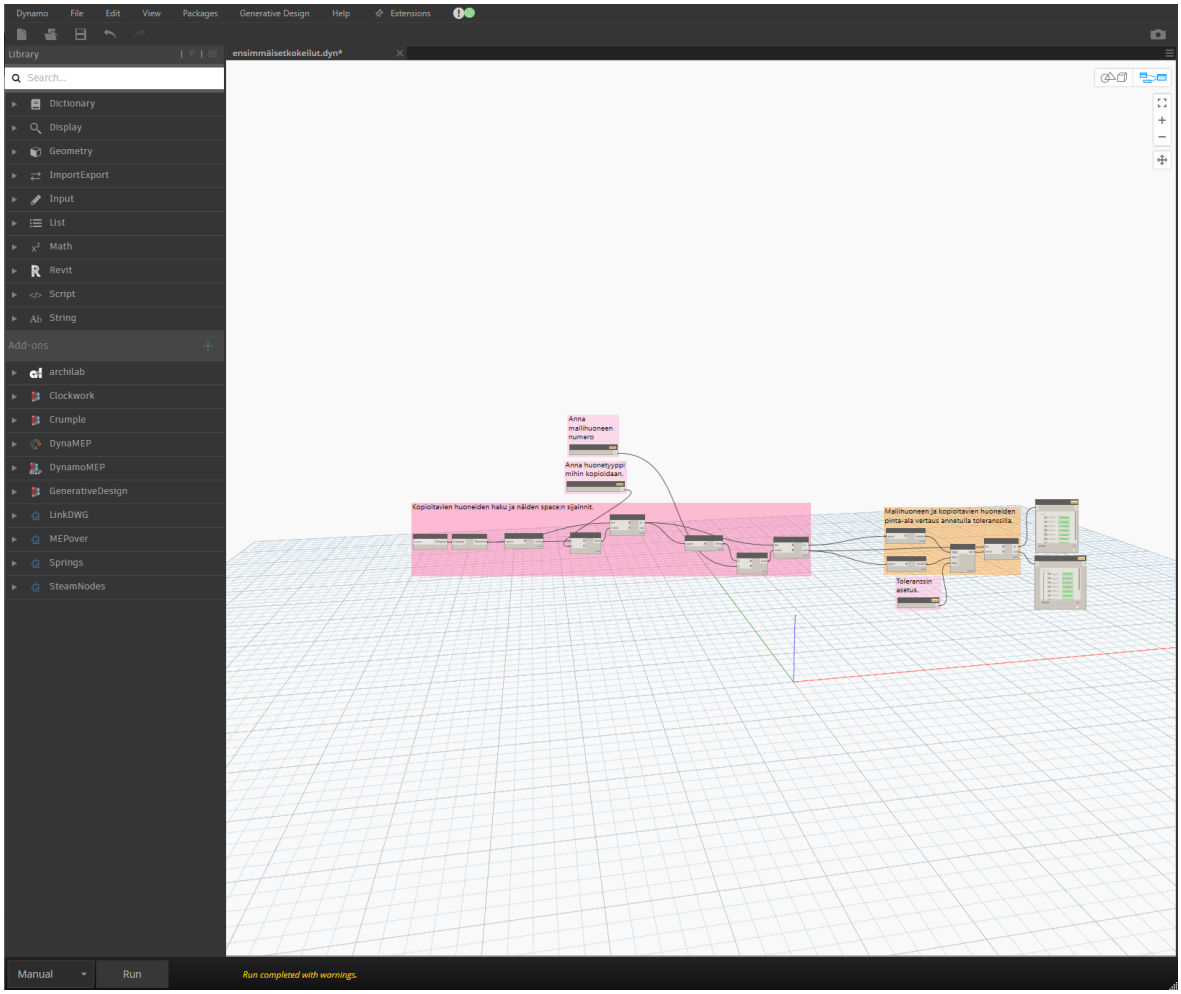
Kuvio 2. MagiCADin tuoma toiminto lämpöpatterin ja putkiston painehäviön laskentaan (Kasari,2023).

MagiCAD on jaettu kuuteen eri suunnittelualan sovellukseen, joista kolme yleisintä ovat MagiCAD Ventilation, MagiCAD Piping ja MagiCAD Electrical (MagiCAD, i.a.-b). MagiCAD Ventilation- ja Piping-sovellukset tuovat työkaluja ilmanvaihdon, lämmityksen, käyttöveden ja viemäröinnin suunnitteluun ja mallinnukseen. MagiCAD Electrical -sovelluksella puolestaan saadaan työkalut sähkö-, valaistus, tele- ja datajärjestelmien suunnitteluun ja laskentaan.

2.6 Autodesk Dynamo

Dynamo on graafinen ja avoimeen lähdekoodiin perustuva ohjelmointialusta (DynamoBIM, i.a.-k). Dynamo on integroituna Revit-ohjelmistoon, mutta sitä voidaan käyttää myös erillisenä ja itsenäisenä ohjelmana. Dynamoa voidaan myös käyttää avuksi muissakin Autodesk-ohjelmistoissa, joista löytyy API-rajapinta. Dynamon tavoitteena on olla ohjelmointityökalu, joka ei vaadi käyttäjältä aiempaa ohjelmointitaustaa, mutta antaa mahdollisuuden monimutkaistenkin ohjelmien kirjoittamiseen perinteisillä ohjelmointikielillä.

Dynamon graafisella ohjelmoinnilla voidaan luoda elementtejä yhdistämällä toimintojen sarjoja, jotka muodostavat algoritmin (DynamoBIM, i.a.-k). Algoritmien avulla voidaan esimerkiksi etsiä ja lukea jotain tiettyä tietoa tietomallista tai automatisoida jonkin objektin piirtoa. Kuviossa 3 on Dynamon ohjelmointiympäristön työtilan päänäkymä ja esimerkki Dynamosta, joka hakee projektista tilaelementeistä tietoja pinta-alojen vertailuun.

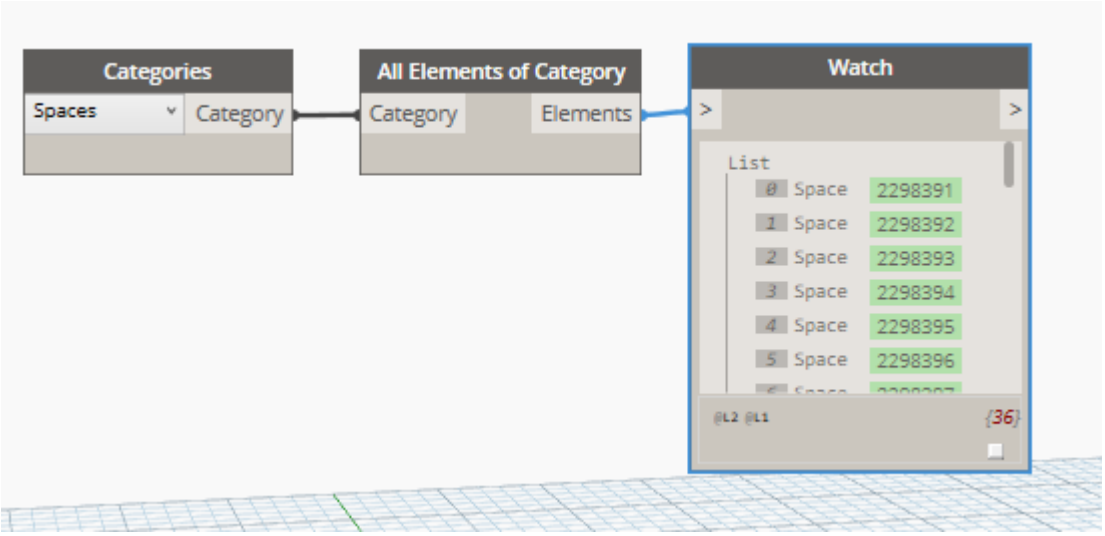


Kuvio 3. Dynamon työtilan päänäkymä (Kasari, 2023).

3 VISUAALINEN OHJELMOINTI DYNAMOLLA

3.1 Dynamon käyttäminen

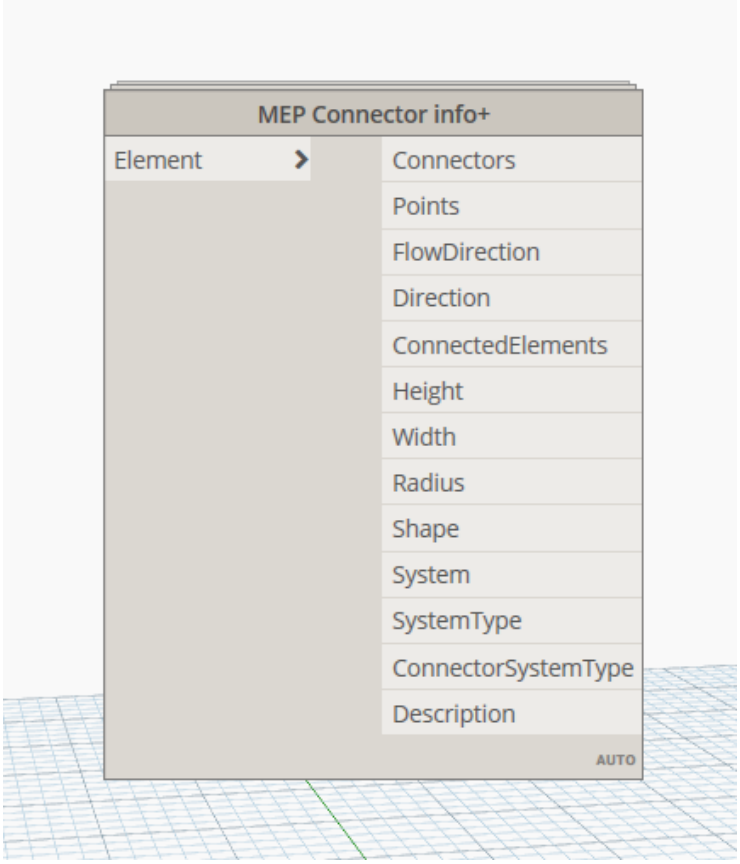
Dynamo on otettavissa heti käyttöön sen ollessa integroituna Revit-ohjelmistoon. Visuaalinen ohjelmointi Dynamolla tarkoittaa, että siinä ei tarvitse kirjoittaa ohjelmointikielillä mitään, mutta ohjelmointikieliä voidaan käyttää tarvittaessa monimutkaisempiin ohjelmiin. Visuaalinen ohjelmointi on loogisen kulun rakentamista yhdistämällä solmuja (nodes) ja johtoja (wires) toisiinsa (DynamoBIM, i.a.-a). Kuviossa 4 on kategoriasolmu ”Categories”, jolla valitaan, mitä elementtejä projektista haetaan. Valittu kategoria välitetään yhdistetylle solmulle ”All Elements of Category”, joka hakee kaikki valitun kategorian elementit projektista. Ohjelmien rakentamiseen tarvittavat solmut löytyvät suurimmaksi osaksi Dynamon vakiokirjastoista. Pakettienhallintatyökalun avulla voi lisäksi hakea tarvittavia erikoiskirjastoja (DynamoBIM, i.a.-d). Tämän lisäksi solmuja voi valmistaa itse, esimerkiksi tekemällä monimutkaisen solmujen verkoston yhdeksi solmuksi, jolla voi selkeyttää ja yksinkertaistaa ohjelmaa merkittävästi.



Kuvio 4. Dynamon solmuja yhdistettyinä toisiinsa johdoilla (Kasari, 2023).

Mallintaessa taloteknisiä järjestelmiä jäävät Dynamon omat kirjastot nopeasti suppeiksi toimintojensa puolesta. Esimerkiksi putkien ja kaapelihyllyjen piirtoon ei Dynamon omista kirjastoista löydy kunnollisia solmuja. Yksi hyvä kirjasto, josta löytyy paljon toimintoja nimenomaan talotekniikan mallintamiseen, on MEPover. Lisäksi tärkeä toiminto ladattavissa

kirjastoissa on putkia ja putkiosia mallinnettaessa tarvittavien liittimien (connectors) tiedonhaku. Liittimiä tarvitaan putkien ja putkiosien yhdistämiseen keskenään. Kuviossa 5 on MEPover-kirjaston solmu, joka hakee sille välitettyjen elementtien tiedot liittimistä, kuten liittimien sijainnin, niiden suunnan, koon, muodon ja paljon muuta hyödyllistä tietoa. Liittimien sijaintitietoa tarvitaan esimerkiksi siihen, että voidaan yhdistää kaksi liitintä samassa pistesijainnissa toisiinsa. Mikäli näitä ei yhdistetä, jäävät liittimet avonaisiksi eivätkä putkistot ole yhtenäisiä ja niiden virtauksia yms. ei voida laskea.



Kuvio 5. MEPover-kirjaston sisältämä solmu, joka tuo tiedot putkien ja osien liittimistä (Kasari, 2023).

Solmut ovat objekteja, jotka suorittavat erinäisiä toimintoja (DynamoBIM, i.a.-f). Nämä toiminnot voivat olla esimerkiksi yksinkertaisia numeroiden tai tekstien syöttöjä tai monimutkaisempia geometriamuunnoksien laskutoimintoja. Solmut eroavat toisistaan toimintojensa lisäksi myös tulojen ja lähtöjen mukaan. Nämä tulot ja lähdöt toimivat portteina (ports) toisille solmuille, jotka liitetään johtojen avulla toisiinsa. Johdot siis mahdollistavat solmujen tiedonkulun toiselle solmulle. Jotta solmujen toiminnot toimisivat moitteetta, on porteille annettu vaatimuksena tietotyypit, jotka täytyy täsmätä yhdistettävien solmujen välillä.

Esimerkkinä yleisimpiä tietotyyppejä ovat kokonaisluku (int), liukuluku (double) ja merkkijono (string).

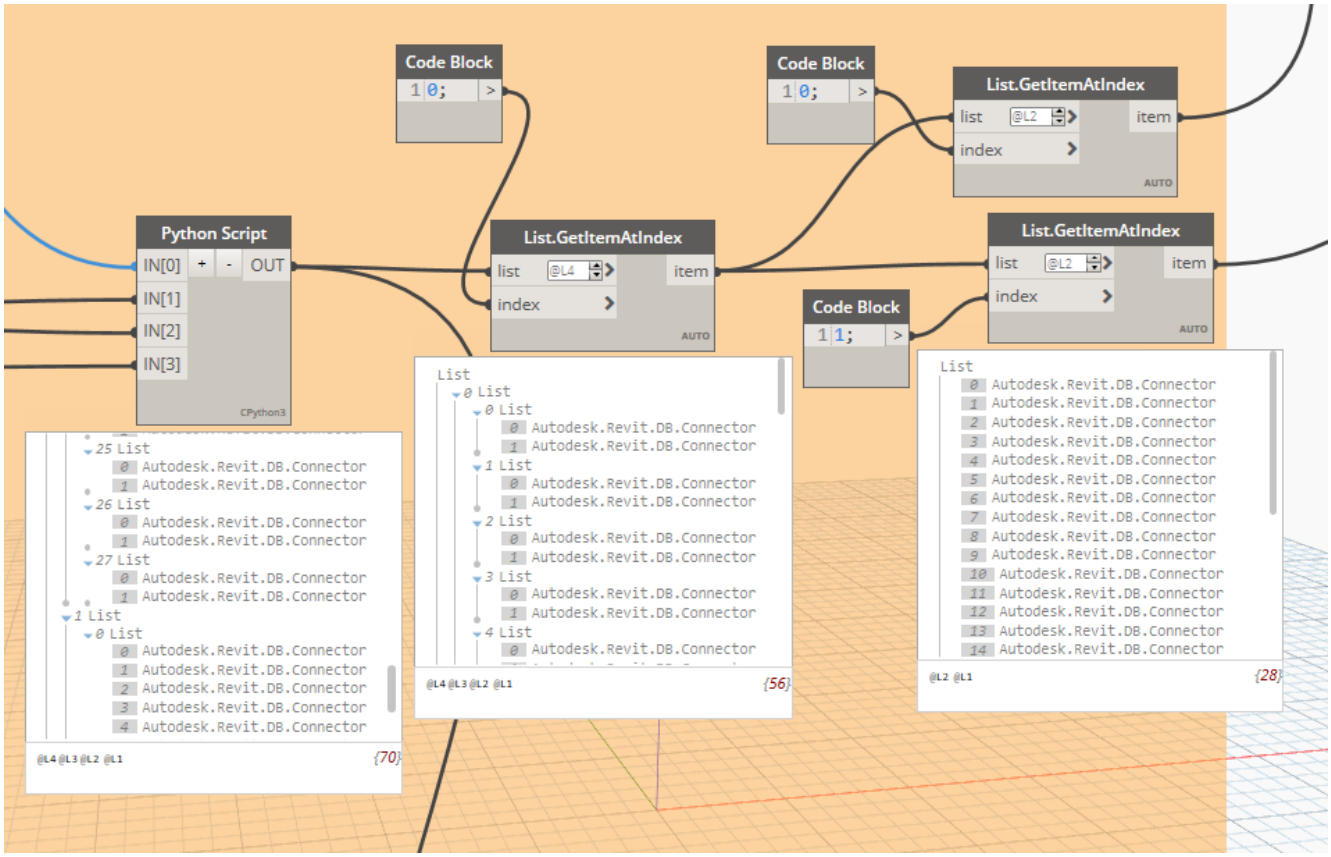
Ohjelman looginen kulku kulkee vasemmalta oikealle. Solmuja voi sijoitella Dynamon työtilassa vapaasti, mutta niiden tulot ovat aina vasemmalla puolen ja lähdöt oikealla puolen (DynamoBIM, i.a.-o). Tällöin ohjelman rakenne muodostuu lähes automaattisesti kulkemaan vasemmalta oikealle.

3.2 Listojen käyttö Dynamo-ohjelmoinnissa

Listat ovat yksi tärkeimmistä dynaamisista tietotyypeistä, joita Dynamo käyttää tietojen hallinnassa ja käsittelyssä. Listojen käyttö Dynamossa on tärkeää, koska ne mahdollistavat useiden eri toimintojen suorittamisen useilla arvoilla samanaikaisesti sekä mahdollistavat usean elementin välityksen solmujen välillä (DynamoBIM, i.a.-m).

Dynamossa listojen indeksointi tapahtuu samalla lailla kuin muillakin ohjelmointikielillä, eli listojen indeksit alkavat numerosta nolla (DynamoBIM, i.a.-m). Jos listassa on esimerkiksi viisi elementtiä [1, 2, 3, 4, 5], olisivat niiden indeksit [0, 1, 2, 3, 4]. Solmuille, joille voidaan syöttää useampikin tulo, syötetyt listat voivat olla keskenään rakenteeltaan ja kooltaan erilaisia. Mikäli listat eivät ole keskenään samanlaisia, täytyy solmun asetukset listojen rakenteille määritellä halutun lopputuloksen mukaisesti tai listat täytyy käsitellä ennen syöttöä solmulle.

Listojen käsittelyssä merkittävimmät toiminnot ovat listojen kerroksien valinta sekä listojen läpikäynnin algoritmit. Listan kerroksien valinta voidaan suorittaa solmujen tulojen asetuksilla (lacing), jolloin tulolle määritellään, miltä listan kerrokselta tiedot haetaan. Vaihtoehtoisesti voidaan käyttää myös siihen tarkoitettuja solmuja, kuten kuviossa 6 olevaa solmua ”List.GetItemAtIndex”. Listojen läpikäynnin eri algoritmit puolestaan määrittelevät, kuinka kaksi listaa läpikäydään keskenään (DynamoBIM, i.a.-m). Kahta eri kokoista listaa läpikäydessä täytyy valita, käydäänkö listat lyhyimmän vai pisimmän mukaan. Lisäksi on mahdollista käyttää ristituloa, jolloin listan yksi jokaista arvoa vertaillaan/käytetään listan kaksi arvojen kanssa. Listoista saadaan tällöin muodostettua tai tarkastettua jokainen mahdollinen yhdistelmä.



Kuvio 6. Listojen käsittely ja listan kerroksien rajaus (Kasari, 2023).

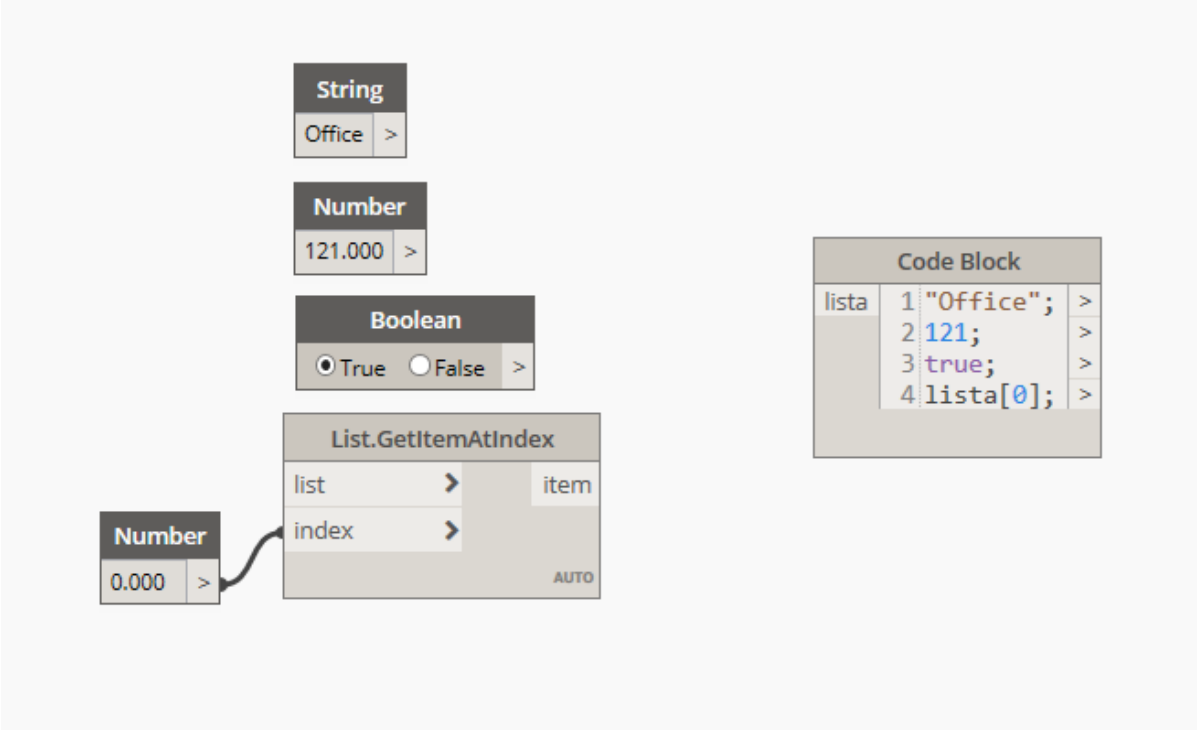
Esimerkkinä listojen käytöstä kuviossa 6 on Python-skripti tuottanut liittimistä listan, jonka sisällä on kaksi listaa. Listassa nolla (yksi) on 28 listaa, joissa on kaksi liitintä jokaisessa listassa. Listassa yksi (kaksi) puolestaan on yksi lista liittimistä, joille ei löytynyt toisista liittimistä pareja. Käytettäessä solmua ”List.GetItemAtIndex” saadaan valittua skriptin tuottamasta listasta vain ensimmäisen lista. Solmulle ”List.GetItemAtIndex” annetaan indeksin asetukseksi nolla. Solmu hakee tällöin tiedot listan paikalta nolla. Solmu tarvitsee lisäksi listan syötön asetukseksi ”L4”, jolloin solmu hakee listan uloimmalta kerrokselta tiedot paikalta nolla. Näin saamme eroteltua halutun listan 28 listasta liittimiä. Kuviossa 6 oikealla solmut ovat hakeneet liittimet edellisen listan kerrokselta ”L2” indeksin nolla ja yksi paikalta ja tehneet liittimistä omat listansa.

Listojen käsittelyyn on edellä mainittujen toimintojen lisäksi lukuisia muita toimintoja. Listoilta voidaan laskea mm. niiden elementtien tai listojen alilistojen lukumäärät solmulla ”List.Count” sekä tarkastella listasta jotain tiettyä arvoa tai elementtiä. Lisäksi listojen rakenteita voidaan kääntää ympäri mm. niiden indeksien tai kerroksien perusteella.

3.3 Koodilohkot ja DesignScript Dynamossa

Dynamon koodilohkot ovat tapa lisätä Dynamon omaa ohjelmointikieltä DesignScriptiä visuaaliseen ohjelmointiin (DynamoBIM, i.a.-b). DesignScript on ohjelmointikieli, joka toimii Dynamon perustana ja joka on helposti luettavaa sen ytimekkään ja yksinkertaisen rakenteen vuoksi (DynamoBIM, i.a.-n). DesignScriptin käyttäminen koodilohkoissa on helppoa, koska sen ohjelmointikielen toiminnot ovat lähes identtiset tavallisten solmujen toimintoihin ja nimityksiin verrattuna. Dynamosta löytyy myös toiminto ”Node to Code”, joka muuntaa tavalliset solmut DesignScript-ohjelmointikieleksi ja näin opettaa ohjelmoijaa koodilohkojen ja DesignScriptin käytössä omassa työssään.

Koodilohkoja voidaan käyttää yksinkertaisimmillaan lähtötietojen, esimerkiksi tekstin (string) ja numeroiden (number), asettamiseen seuraaville solmuille toimimaan monimutkaisempien solmujen tavoin. Niitä voidaan käyttää esimerkiksi hakemaan listasta halutun indeksin mukainen elementti tai suorittamaan kahden geometriapisteen mukainen viiva (DynamoBIM, i.a.-n). Käytettäessä koodilohkoja osana visuaalista ohjelmointia saadaan parhaimmillaan yksinkertaistettua ohjelmaa helpommin luettavaksi, kuten kuviossa 7, jossa on kolmen eri tietotyypin asetus sekä listasta haettavan indeksin mukainen haku suoritettu tavallisilla solmuilla verrattuna yhteen koodilohkoon.

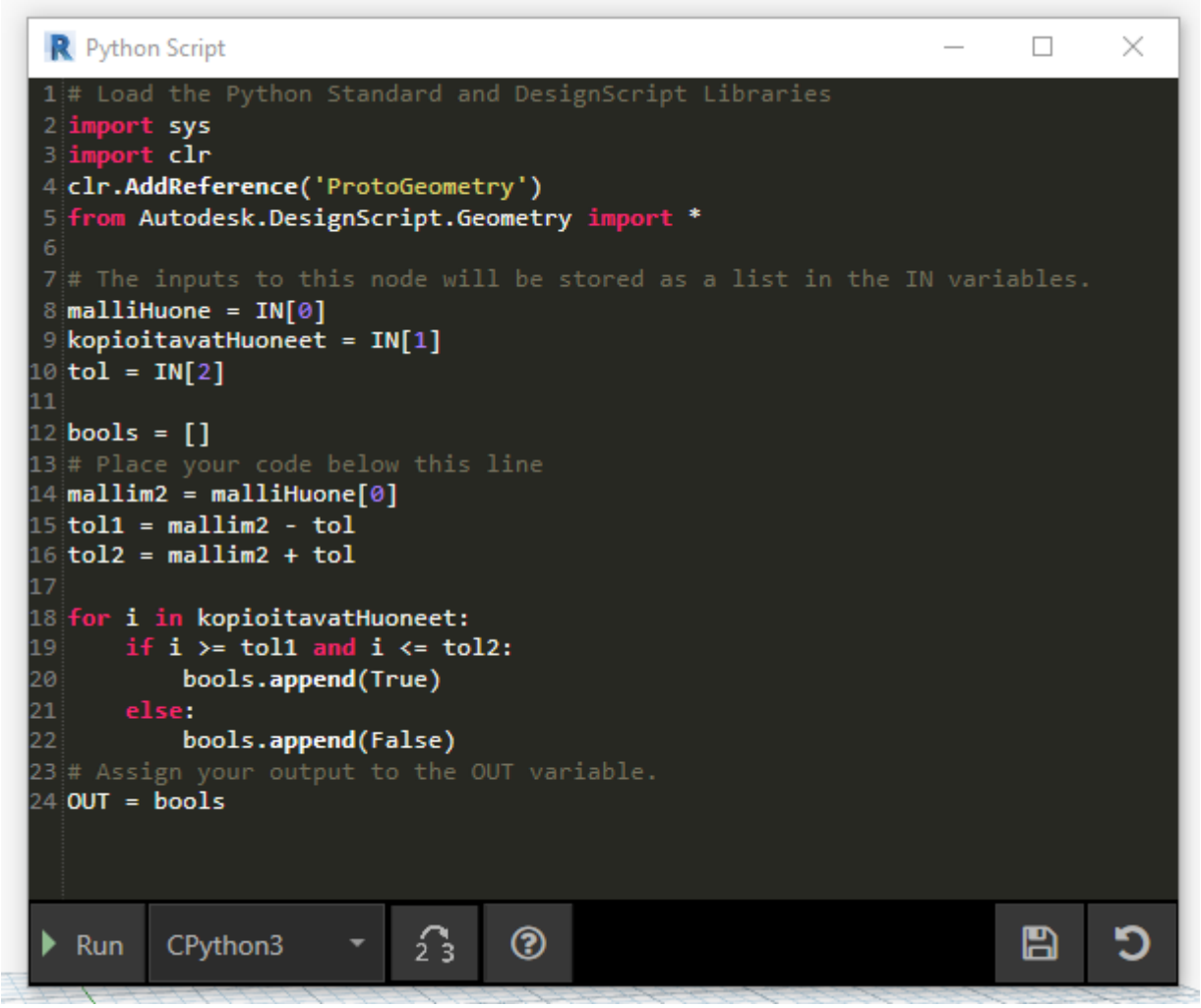


Kuvio 7. Tavallisten solmujen ja koodilohkon vertailu (Kasari, 2023).

3.4 Pythonin käyttö Dynamossa

Dynamon oman DesignScript:ohjelmointikielen lisäksi Dynamossa voidaan käyttää tavanomaisempiakin ohjelmointikieliä kuten Pythonia. Python on laajasti käytetty ohjelmointikieli, jonka suosio perustuu sen syntaksiin (DynamoBIM, i.a.-g). Se on helposti luettavaa ja helpommin opittavissa kuin monet muut ohjelmointikielet. Pythonin suuria hyötyjä ovat myös sen laajat ja ilmaiset kirjastot, joista löytyy paljon valmiiksi tehtyjä moduuleja nopeuttamaan ja helpottamaan ohjelmointia (Python, i.a.).

Visuaalisen ohjelmoinnin haittapuoli on joskus sen sekava lopputulos ja se, että visuaalisen ohjelmoinnin työkaluista ei aina löydy tarvittavia ominaisuuksia tai toimintoja monimutkaisempien ohjelmien valmistamiseen (DynamoBIM, i.a.-g). Ohjelmaan on esimerkiksi helpompi toteuttaa ehdollisia toimintoja, kuten ”if” ja ”else” sekä silmukoita (loop) Pythonin avulla. Python on tehokas työkalu tehostamaan ja yksinkertaistamaan visuaalista ohjelmointia Dynamolla. Esimerkiksi kuviossa 8 suoritetaan Python-ohjelmakoodilla yksinkertainen huonetilojen pinta-alan vertailu mallihuoneen ja muiden valittujen huoneiden välillä annetun toleranssin perusteella.



Kuvio 8. Python-ohjelmakoodi pinta-alojen vertailuun (Kasari, 2023).

Tietomallin automatisointiohjelmaa tehtäessä ilmenikin, että esimerkiksi listojen läpikäynti kaksoiskappaleiden vuoksi tai listojen vertailu keskenään on huomattavasti helpompaa ja tehokkaampaa Python-koodilla kuin Dynamon omilla solmuilla.

Python-koodia voidaan kirjoittaa ja käyttää Dynamossa omissa ”koodilohkoissaan” (DynamoBIM, i.a.-g). Python-lohkoihin yhdistettyjen Revit-kirjastojen avulla saadaan yhteys Revitin ohjelmointirajapintaan, jolloin Python-ohjelmakoodilla voidaan toteuttaa kaikkia samoja toimintoja kuin Revitillä itsessään (DynamoBIM, i.a.-h). Python-lohkoihin täytyy aina määritellä tulot ja lähtö. Tuloja (IN) voi olla useita, mutta lähtöjä (OUT) voi olla vain yksi. Ulkoisesti Python-lohko toimii samalla lailla kuin normaalit solmut, joihin liitetään johdoilla muita solmuja.

3.5 C# käyttö Dynamossa

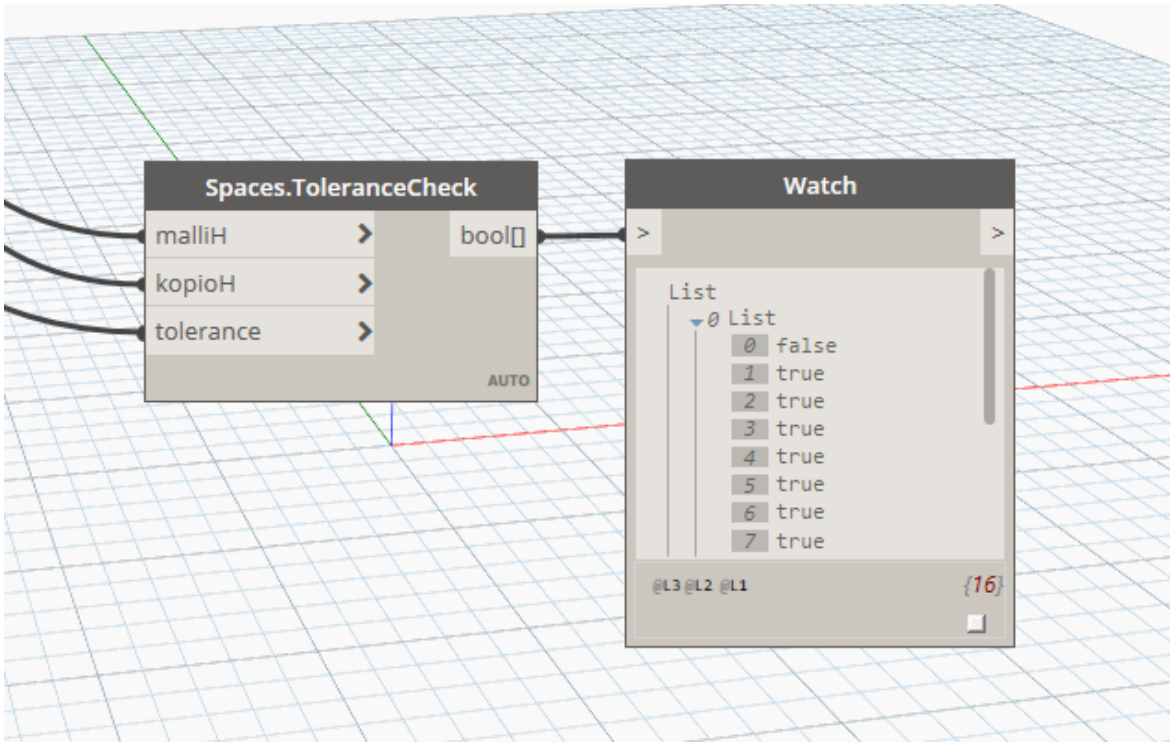
Python-ohjelmointikielen lisäksi Dynamossa voidaan käyttää C#:n-ohjelmointikieltä. Käytettäessä C#-ohjelmointikieltä ero Pythoniin on se, että C#-ohjelmointikieltä käytetään lähinnä Dynamon ohjelmointiympäristön ulkopuolella ja se käännetään Dynamossa solmuiksi ja toiminnoiksi (DynamoBIM, i.a.-c). C#-ohjelmakoodin käyttäminen Dynamossa mahdollistaa erityisesti uusien räätälöityjen solmujen kehittämisen tai Revit:toimintojen ajamisen ohjelmakoodilla.

```
using System;
using Autodesk.DesignScript.Geometry;
using System.Collections.Generic;
using Dynamo.Applications;


namespace CustomNodes
{
    0 references
    public class Spaces
    {
        0 references
        public static List<bool> ToleranceCheck(double malliH, List<double> kopioH, double tolerance)
        {
            List<bool> huoneLista = new List<bool>();
            double tolerance1;
            double tolerance2;

            tolerance1 = malliH - tolerance;
            tolerance2 = malliH + tolerance;

            foreach (double i in kopioH)
            {
                if (i>=tolerance1 && i <= tolerance2)
                {
                    huoneLista.Add(true);
                }
                else
                {
                    huoneLista.Add(false);
                }
            }

            return huoneLista;
        }
    }
}

```

Kuvio 9. C#-ohjelmakoodi pinta-alojen vertailuun (Kasari, 2023).

Käytettäessä esimerkiksi C#-ohjelmointikieltä kirjoitetaan haluttu toiminto ohjelmakoodina Microsoftin ohjelmointiympäristössä Visual Studiossa ja käännetään ohjelmakoodi tiedostomuotoon .dll. Kuviossa 9 on kirjoitettu sama pinta-alojen vertailu C#-ohjelmakoodilla Visual Studiossa kuin kuviossa 8 Python-ohjelmakoodilla. Tiedostomuoto .dll:llä on koottu kirjasto, joka sisältää kokoelman menettelyjä ja ohjaimia, joiden avulla useat ohjelmat voivat ajaa jaettuja toimintoja yhteisten kirjastojen kautta (Microsoft, 2022). Kun haluttu ohjelmakoodi on käännetty .dll-tiedostomuotoon, voidaan se ladata Dynamoon omaksi solmukseen ja käyttää sitä kuten muitakin solmuja (DynamoBIM, i.a.-l). Kuviossa 10, on kuvion 9 ohjelmakoodi tuotuna .dll-tiedostomuodossa Dynamoon. Tätä menetelmää käytettäessä

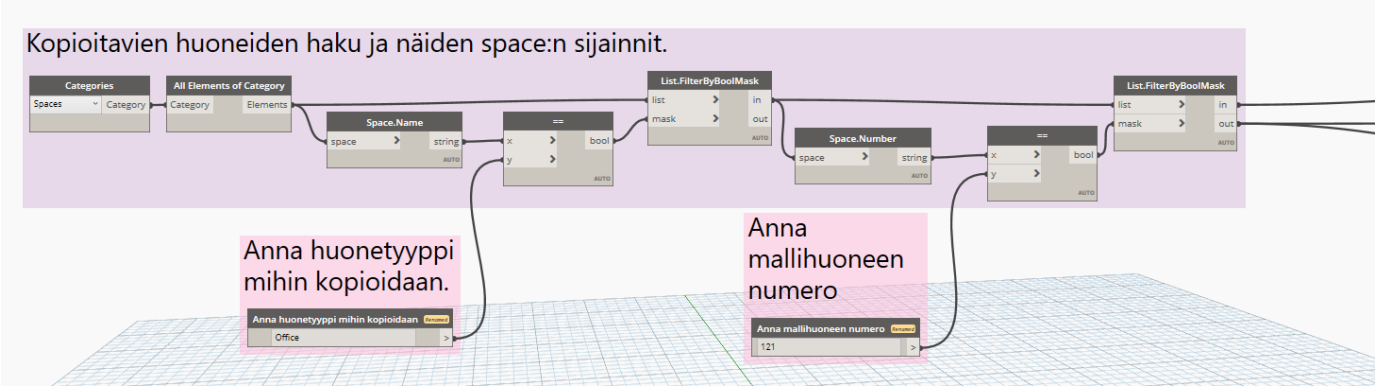
puhutaan ”Zero-Touch”:menetelmästä, joka viittaa yksinkertaiseen osoita ja napsauta -menetelmään tuontia varten. Käytettäessä Zero Touch -menetelmää voidaan Dynamoon tuoda muitakin kirjastoja, joita ei välttämättä ole edes suunniteltu Dynamon käyttöä varten, mutta joita voidaan hyödyntää mallinnuksessa Dynamon kautta.



Kuvio 10. C#-ohjelmakoodi tuotuna Dynamoon (Kasari, 2023).

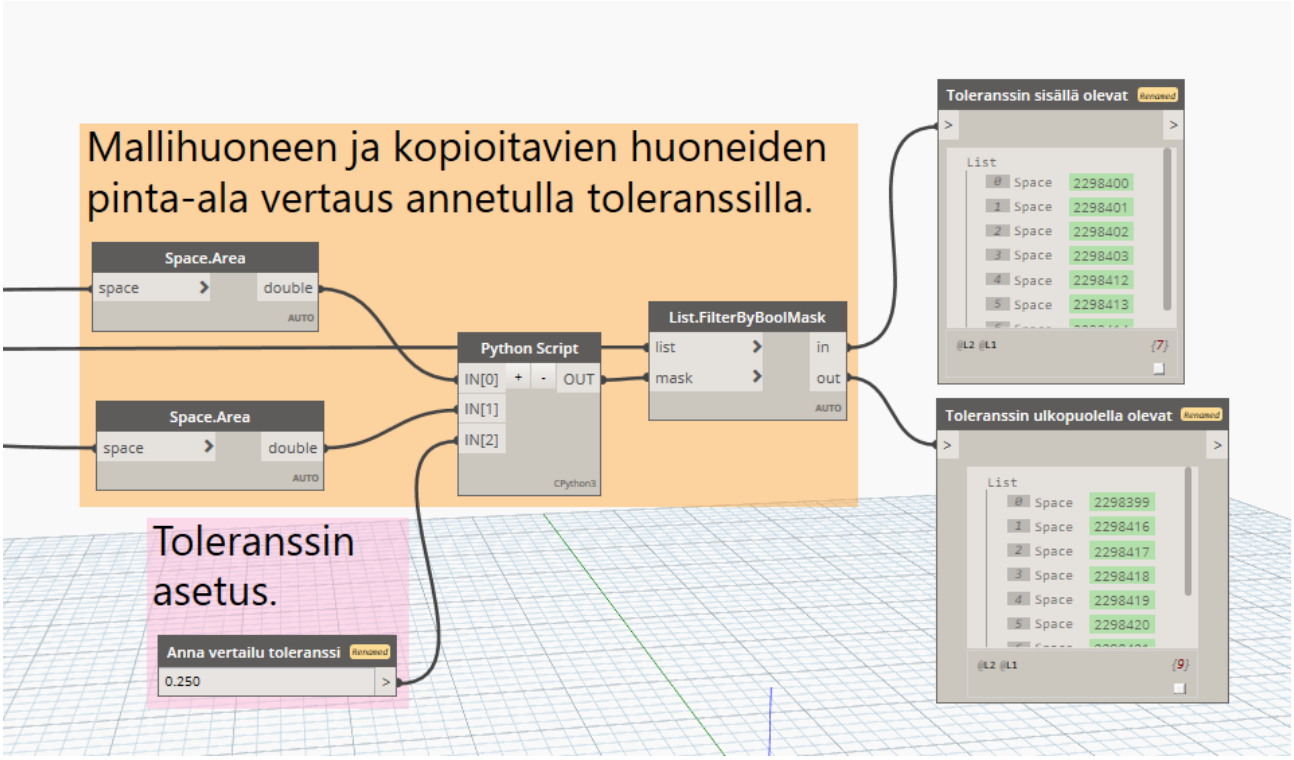
3.6 Dynamo-skriptin luominen ja testaaminen

Ensimmäisenä luodaan Dynamo-projekti, jonka jälkeen voidaan aloittaa ohjelman tekeminen, ja kun Dynamo on käynnistetty Revitin sisällä, on Revit ja Dynamo automaattisesti yhdistettynä toisiinsa. Ensimmäisenä solmuna tulee monesti solmu, jolla voidaan valita projektista jotain tiettyä tietoa, esimerkiksi elementtejä tietyn tason tai kategorian perusteella, kuten kuviossa 11 vasemmalla on haettu projektin kaikki tilat ”space”:kategorian alta. Tätä dataa voidaan sitten jatkokäyttää esimerkiksi erilaisiin laskentoihin, vertailuihin tai lähtötietona seuraaville toiminnoille. Kuviossa 11 esimerkiksi on tilaelementeistä valittu tilat, jotka ovat annetun tilatyypin ”Office” mukaiset.



Kuvio 11. Esimerkki solmuista, joilla haetaan halutut tilaelementit projektista (Kasari, 2023).

Esimerkiksi kun projektista on haettu halutut tilat ja näiden joukosta on valittu mallihuone sen tilanumeron perusteella, voidaan tämän jälkeen vertailla muiden tilojen pinta-aloja mallihuoneen pinta-alaan ja näin muodostaa kaksi listaa tiloista. Kuviossa 12 on tilaelementit välitetty solmuille ”Space.Area”, jotka lukevat tilaelementeistä näiden pinta-alat ja välittävät ne pinta-alaa vertaavalle Python-solmulle. Python-solmu ajaa ohjelmakoodia, joka on esitetty kuviossa 8. Lopputuloksena saamme listan huoneista, jotka mahtuvat mallihuoneen ja sille annetun toleranssin sisälle, ja listan huoneista, jotka ovat joko liian suuria tai pieniä mallihuoneeseen nähden.



Kuvio 12. Tilojen pinta-alatietojen haku ja vertailu Python-ohjelmakoodilla (Kasari, 2023).

Kehitettäessä ja tehdessä visuaalista ohjelmaa Dynamolla on sen luettavuuden ja selkeyden kannalta oleellista, että solmut ja johdot asetellaan kohdistetusti ja ryhmitellään organisoidusti (DynamoBIM, i.a.-j). Erityisesti ohjelman kasvaessa suuremmaksi ja monimutkaisemmaksi korostuu selkeyden merkitys ohjelman jatkokehityksen ja vianetsinnän helpottamiseksi. Dynamosta löytyy myös toiminto ”Cleanup Node Layout”, jolla voidaan uudelleenjärjestää pieniä määriä solmuja ja näin parantaa ohjelman rakenteen selkeyttä.

Toinen merkittävä keino parantaa ohjelman rakennetta on käyttää ryhmiä. Kun valitaan yksi solmu tai enemmän, voidaan solmut ryhmittää ”Create Group” -toiminnolla yhdeksi ryhmäksi (DynamoBIM, i.a.-e). Solmuista tulee yksi liikuteltava lohko, jolle voidaan antaa mm. otsikko ja jonka väriä voidaan muuttaa.

Lohkojen värityksille voi yrityksillä olla olemassa omia sisäisiä määritelmiä. Näin standardisoidaan yrityksen sisäisiä Dynamo-ohjelmia samaan rakenteeseen ja helpotetaan ohjelmien käyttöä ja muokattavuutta yrityksen käyttäjien kesken. Tässä työssä tehdyn ohjelman lohkojen värimääritelmissä on mm. käytetty kuvissa kymmenen ja yksitoista käytettyjä värejä. Kuvissa solmut on ryhmitetty niiden toimintojen perusteella. Violetilla värillä on ryhmitetty solmut, jotka hakevat tietoa projektista, pinkillä värillä ryhmitetyt puolestaan merkitsevät käyttäjän syötteitä ja oranssilla värillä ryhmitetyt merkitsevät ohjelman loogista prosessia.

4 MALLITILOJEN KOPIOINNIN AUTOMATISOINTI

4.1 Ohjelman vaatimukset ja rajaukset

Suunnittelupalaverissa Swecon kanssa sovittiin automatisointiohjelman keskeisimmät toiminnot. Ohjelmalle sovitut tärkeimmät toiminnot olivat seuraavat:

- Ohjelma kopioi sille annetun huonetyypin sisältämät laitteet ja putkistot saman huonetyypin huoneisiin.
- Kohdehuoneet rajataan mallihuoneen pinta-alan ja siihen lisättävän toleranssin mukaan.
- Kopioinnin seurauksena ei kohdehuoneiden laitteisiin ja putkistoihin saa jäädä avonaisia liittimiä.
- Rajauksena ohjelman ei tarvitse tässä vaiheessa yhdistää putkistoja päärunkoihin.

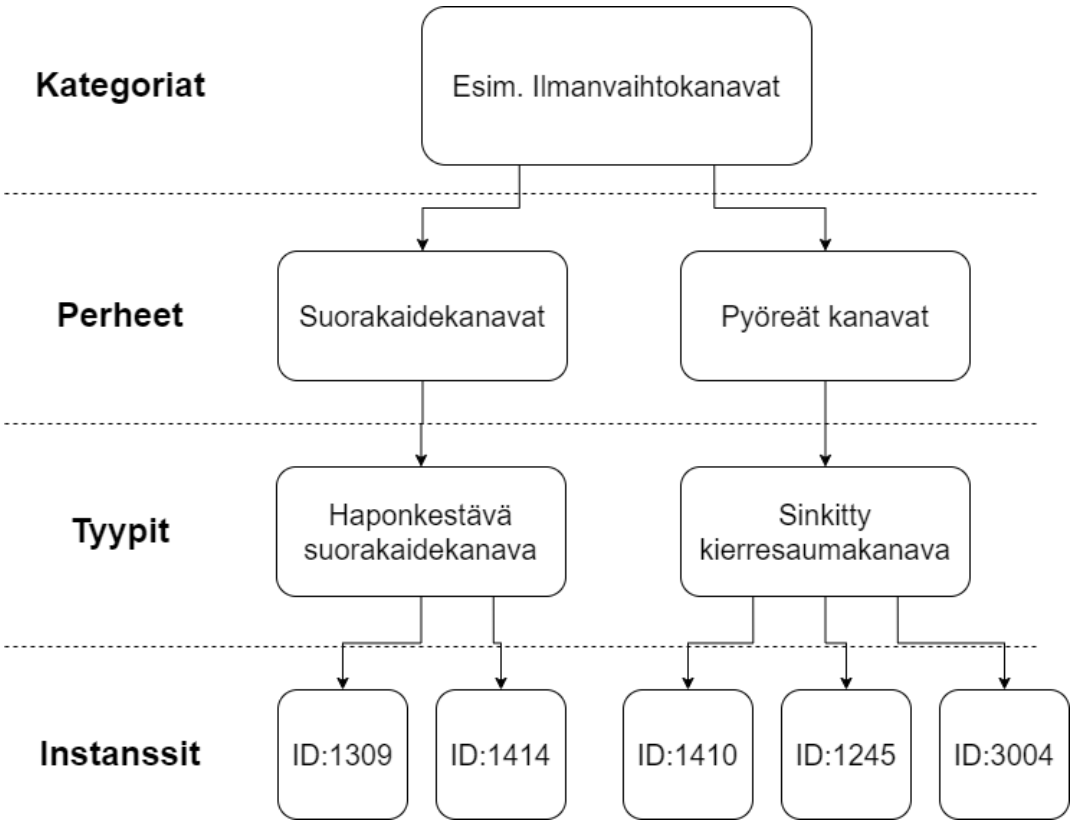
Lisäksi, mikäli ohjelma saataisiin toimimaan, lisättäisiin siihen toiminto jo olemassa olevien laitteiden siirtämiseen ja toiminto huoneiden vertailuun ylimääräisen kopioinnin estämiseksi.

4.2 Mallitilojen kopioinnin periaatteet

Lähtiessä suunnittelemaan ja kehittämään ohjelmaa, joka automatisoi jotain työvaihetta, täytyy ensin ajatella ohjelman niin sanottuja tuloja ja lähtöjä, joita tarvitaan, että ohjelma pystyy prosessoimaan ja tuottamaan haluttuja toimintoja ja asioita. Tässä tapauksessa, kun kehitetään ohjelmaa, joka automatisoi talotekniikan piirtoa ja joka kopioi mallihuoneen talotekniset järjestelmät muihin huoneisiin, täytyy etsiä kohteesta ensimmäisenä lähtökohta, joka pitää sisällään halutut kopioitavat elementit eli huoneen/tilan.

Kaikki mitä Revit-tietomalli pitää sisällään on elementtejä, esim. seiniä, pöytiä, lämmitysputkistoja tai vain huoneita. Elementit Revitissä on jaettu omiin kategorioihinsa, jotka pitävät sisällään niin sanottuja perheitä (DynamoBIM, i.a.-i). Perheen sisällä puolestaan elementit jakautuvat omiin tyyppeihinsä, ja kun elementti sijoitetaan tietomalliin, tulee siitä sen

perheen instanssi, jolla on aina yksilöllinen ID-tunnus, kuten kuviossa 13 olevassa Revitin hierarkiakaaviossa on esitetty.



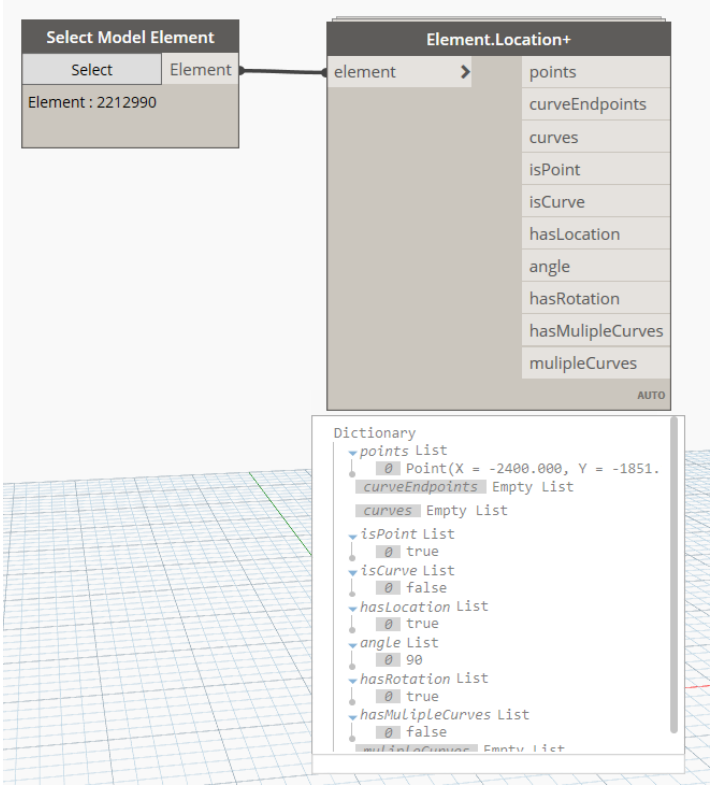
Kuvio 13. Revit-tietokannan hierarkiakaavio (Kasari, 2023).

Ensimmäinen asia, joka projektikannasta haetaan, on huone-elementit, ja ne löytyvät ”Spaces” -kategorian alta. Revitistä löytyy myös kategoria ”Rooms”, mutta sitä käytetään enemmän sisustussuunnittelun ja arkkitehtisuunnittelun näkökulmasta. ”Spaces” eli tilaelementeillä on enemmän ominaisuuksia, joita tarvitaan talotekniikan mallintamisessa. Tietokannasta haetuista tilaelementeistä suodatetaan käyttäjän antaman syötteen mukaiset tilatyypit, ja lisäksi valitaan käyttäjän antaman tilanumeron mukainen huone mallihuoneeksi niin kuin kuviossa 11 on tehty. Oikeista tilatyypeistä täytyy vielä rajata pois väärän kokoiset huoneet.

Seuraava tärkeä asia kopioinnin kannalta on valita mallihuoneesta nollapiste, jota käytetään mallihuoneen elementtien sijaintien laskentaan. Kehitystyön aikana käytettiin aluksi tilaelementeiltä saatavaa keskipistettä, mutta keskipiste osoittautui huonoksi vaihtoehdoksi, koska keskipisteet eivät ole lukittuja ja niitä pystyy helposti siirtämään

huomaamatta. Ratkaisuksi nollapisteelle löytyi solmu, joka hakee tilaelementiltä kyseisen tilan rajat ja rajojen alkupisteen.

Mallihuoneen sisältämät elementit voidaan hakea solmulla, joka hakee kaikki elementit tilaelementin rajojen sisältä. Koska tämä hakee kaikki elementit rajojen sisältä, täytyy elementtien lista jälleen suodattaa ylimääräisistä elementeistä. Kun listasta on suodatettu halutut elementit, välitetään sen elementit Clockwork-kirjastosta löytyvälle solmulle, joka hakee elementeiltä lukuisia geometriatietoja. Kuviossa 14 haetuilla geometriatiedoilla pystymme laskemaan päätelaitteiden sijaintien erotukset huoneen nollapisteeseen verrattuna.

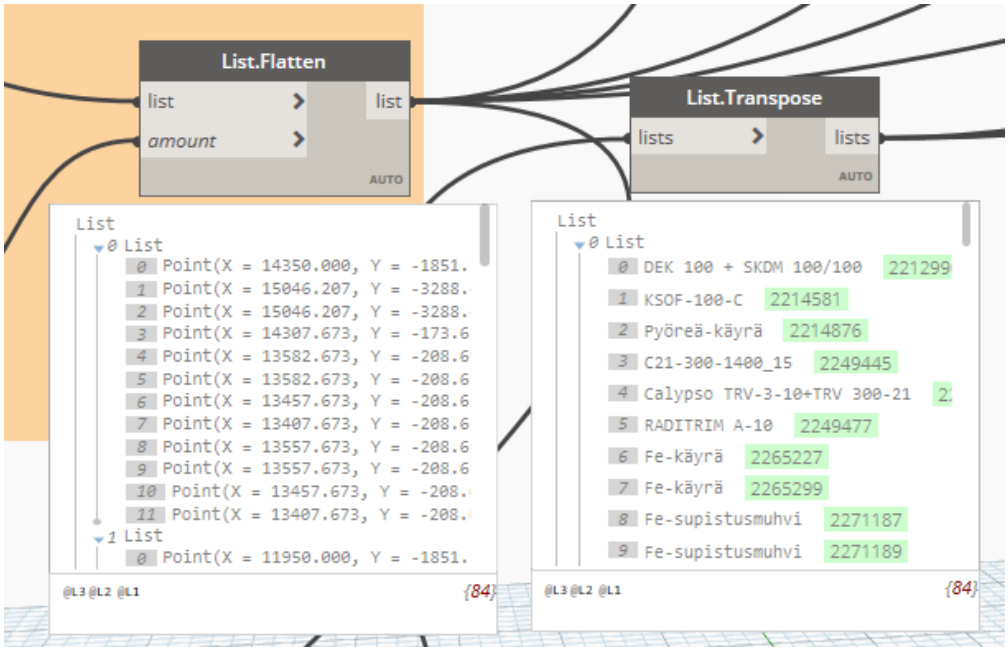


Kuvio 14. Ilmanvaihdon päätelaitteen geometriatietojen haku (Kasari, 2023).

Helpommaksi ja järkevämmäksi tavaksi suorittaa mallihuoneen elementtien sijaintien erotukset nollapisteeseen nähden oli tehdä Python-skripti, joka laskee erotukset ja tekee lasketuista arvoista listan. Tämä lista välitetään seuraavalle Python-skriptille, joka vähentää kopioitavien huoneiden nollapisteistä mallihuoneen elementtien erotukset jokaiselle kopioitavalle elementille. Tällä saadaan kopioitaviin huoneisiin sijoitettaville elementeille oikeat sijainnit jokaiselle huoneelle. Talotekniikan elementtejä kopioitaessa on selkeämpää

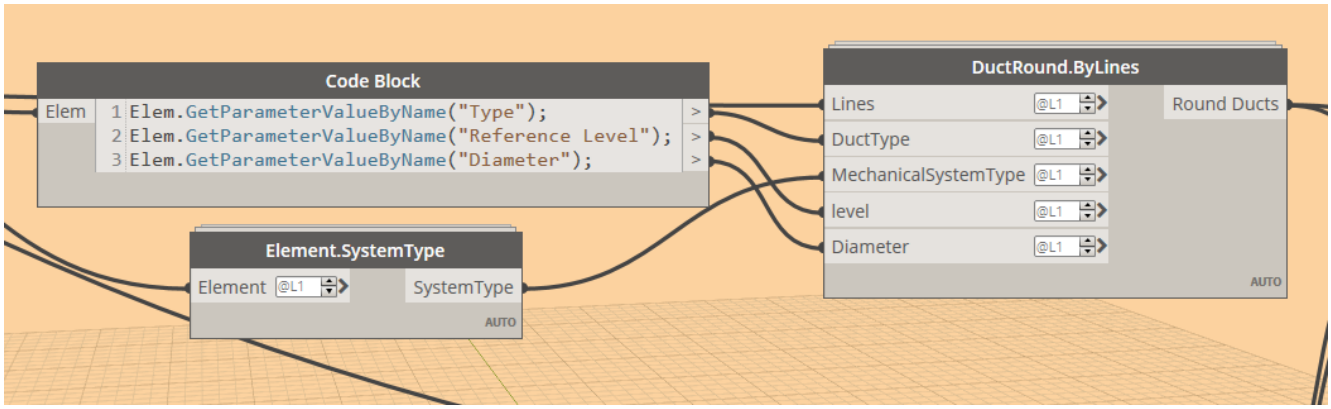
laskea laitteiden ja putkistojen yms. kourujen geometriatiedot omilla skripteillään, koska laitteet käyttävät geometrian pistesijaintia ja putkistot ja kourut puolestaan viiva- ja kaarisijaintia.

Elementit, joilla on pistesijainti, kopioidaan solmulla, joka tarvitsee tuloihinsa kopioitavat elementit ja niiden pistegeometriat. Välitettäessä näitä tietoja solmulle on tärkeää, että elementtien ja sijaintien listat ovat rakenteeltaan ja järjestykseltään identtiset. Elementit eivät muuten kopioidu oikeille paikoilleen. Kuviossa 15 elementtien ja sijaintien listat on järjestetty huoneiden perusteella eli elementit ja niiden geometriat on jaettu omiin listoihinsa listan sisällä.



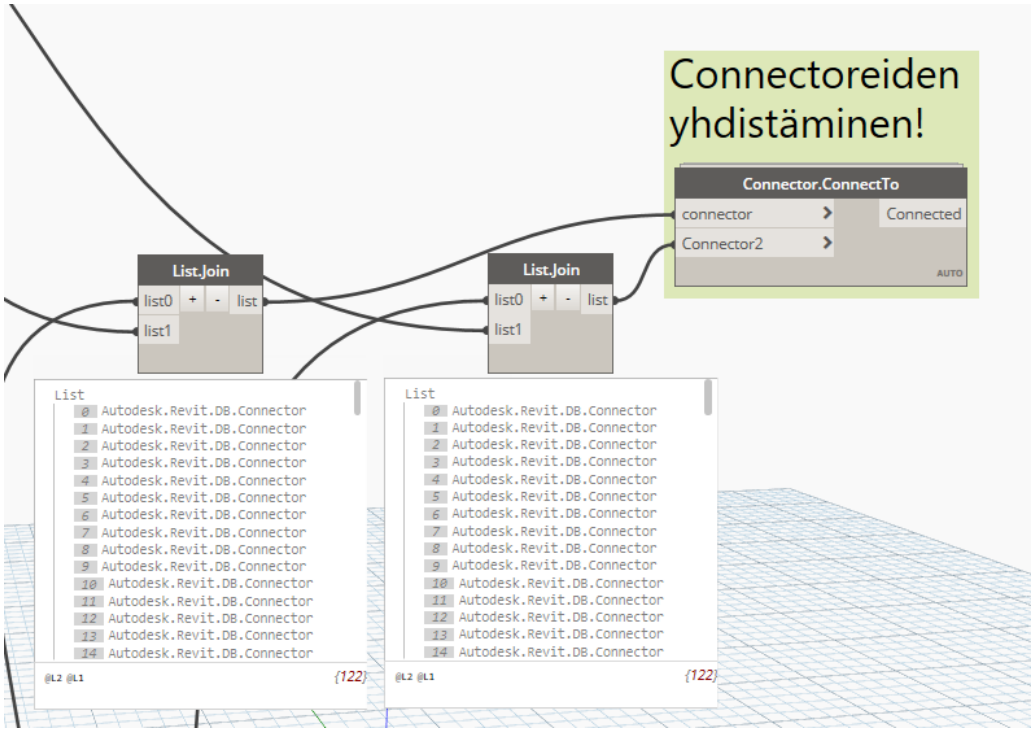
Kuvio 15. Listat kopioitavista elementeistä ja niiden geometrioista (Kasari, 2023).

Kanavien ja putkistojen mallintamiseen saadaan kuvion 16 mukaisella koodilohkolla ja ”Element.SystemType” -solmulla haettua tarvittavat parametrit kanavien piirtosolmulle ”DuctRound.ByLines”.



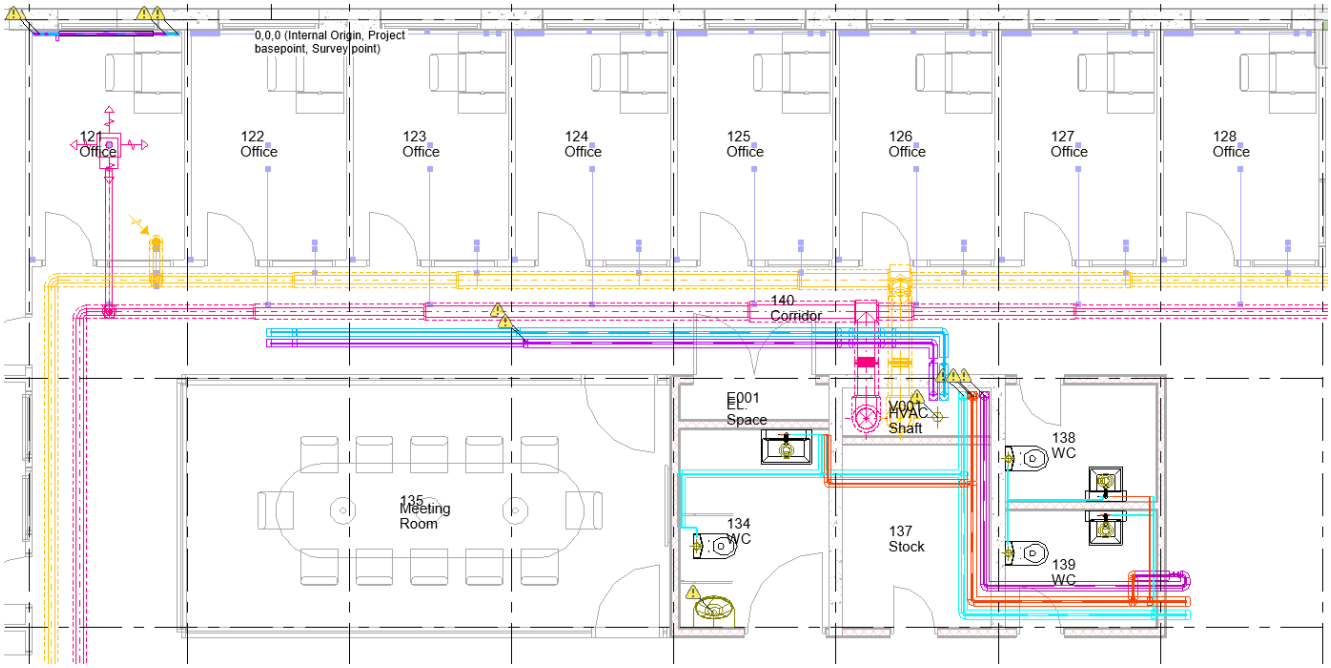
Kuvio 16. Kanavien lähtötietojen hakeminen ja syöttö piirtosolmulle (Kasari, 2023).

Kun mallihuoneen laitteiden ja putkistojen kopiointi sekä mallinnus toimivat moitteetta, tulee seuraavaksi saada yhdistettyä kaikki avoimet liittimet. MEPover-kirjasto on lähes välttämätön yhdistettäessä putkia ja osia toisiinsa. Kuviossa 5 olevalla ”MEP Connector Info+” -solmulla saadaan haettua kaikki tarvittavat tiedot elementeiltä liittimien yhdistämiseen. Tässä työssä tehdyn ohjelman haastavimpia vaiheita oli saada elementit yhdistettyä toisiinsa virheettömästi. Toisin kuin kaikilla elementeillä, liittimillä ei ole yksilöllisiä ID-tunnuksia, mikä aiheuttaa haasteita niiden vertailuun keskenään. Liittimien yhdistämisestä ja vertailusta tarkemmin liitteessä 3. Kuviossa 17 on liittimien listat yhdistetty ja välitetty MEPover -kirjaston ”Connector.ConnectTo” -solmulle yhdistämistä varten.

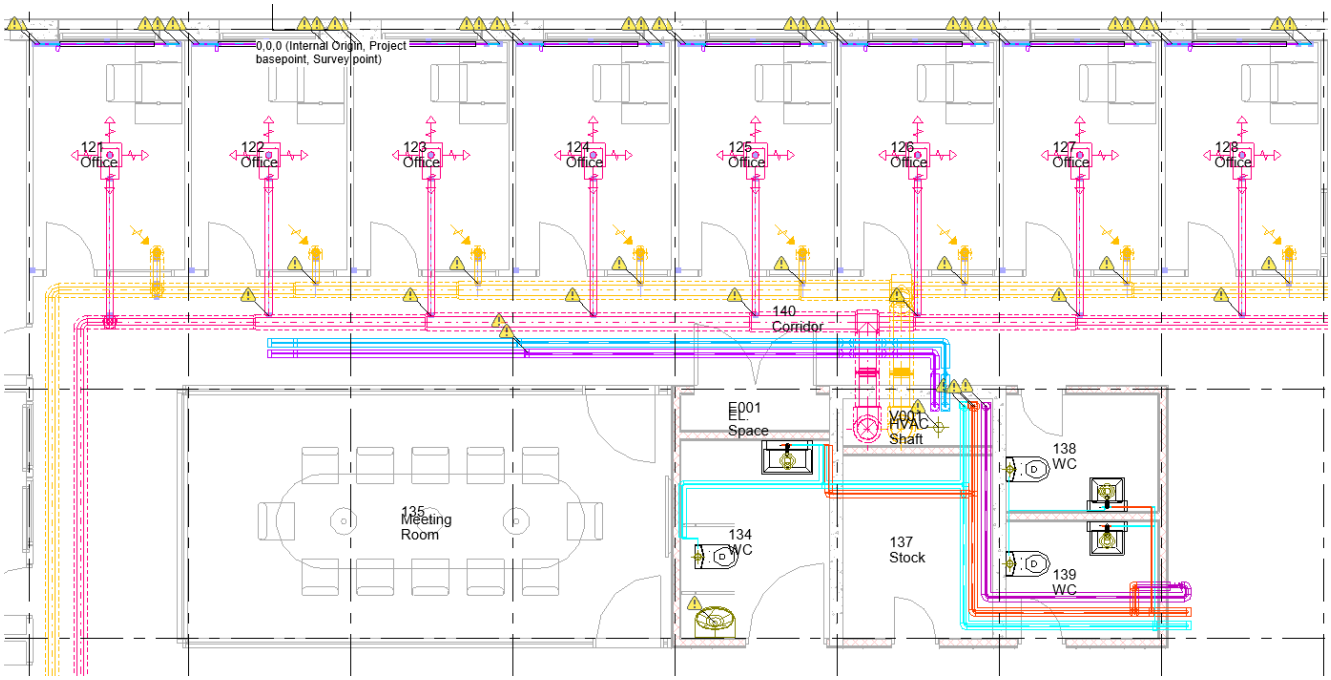


Kuvio 17. Laitteiden ja putkistojen liittimien listojen ja liittimien yhdistäminen (Kasari, 2023).

Toimivan automatisointiohjelman jälkeen päästään kuviossa 18 automatisoinnin lähtötilanteesta eli mallihuoneesta 121 lisättyjen laitteiden ja putkistojen kopiointiin ja piirtoon muihin huoneisiin. Automatisoinnin lopputulos näkyy puolestaan kuviossa 19.



Kuvio 18. Ohjelman automatisoinnin lähtötilanne (Kasari, 2023).



Kuvio 19. Toimivan ohjelman automatisoinnin lopputulos (Kasari, 2023).

Kun ohjelma toimii halutusti, lisättiin ohjelman rakenteeseen vielä olemassa olevien elementtien siirto. Periaatteena oli, että mikäli muihin huoneisiin oli jo kopioitu mallihuoneen laitteet ja putkistot ja niitä siirrettäisiin eri kohtaan huoneessa, reagoisi ohjelma muutokseen siirtämällä tarvittavia elementtejä kopioiduissa huoneissa. Tämä saavutettiin lisäämällä ohjelmaan kopioitavien elementtien ja kopioinnin kohteena olevien huoneiden sisältämien elementtien vertailu. Toiminnaltaan siirto toimii niin, että ohjelma etsii kohdehuoneista jo olemassa olevat elementit ja siirtää niitä niihin kopioitavien elementtien arvojen mukaan. Lisäksi samalla vertailulla pystyttiin lisäämään toiminto, joka estää kohdehuoneiden ylimääräisen kopioinnin ja piirron. Ohjelma siis kopioi ja piirtää mallihuoneesta vain ne elementit, joita kohdehuoneissa ei vielä ennestään ole.

4.3 Ohjelman haasteet ja rajoitukset

Dynamo-ohjelmaa kehitettäessä listojen ja alilistojen ymmärtäminen on merkittävässä osassa, koska lähes poikkeuksetta kaikki solmujen data liikkuu listoina. Onkin tärkeää osata suodattaa ja etsiä listoista haluamiaan elementtejä, joten Dynamon listoihin ja niiden toimintoihin kannattaa perehtyä kunnolla. Automatisointiohjelmaa kehitettäessä huomattiin, että Python-skriptillä pystytään tehokkaammin ja pienemmällä vaivalla käsittelemään listoja kuin Dynamon perinteisillä solmuilla.

Kehitystyön aikana tulleet merkittävimmät haasteet olivat laitteiden ja putkistojen liittimien yhdistäminen. Tähän vaikuttaa useampikin tekijä. Ensimmäisenä haasteena oli se, että liitinelementit eivät sisällä yksilöllistä ID-tunnusta, mikä vaikeuttaa niiden käyttämistä suodatuksissa. Toisena haasteena selvitettäessä liittimien vertailua niiden geometria-arvojen perusteella huomattiin myös, että vaikka kahden liittimen geometria-arvot näyttäisivät Dynamon työtilassa identtisiltä, on niissä kumminkin hyvin pienet desimaaliset erot. Tämä selviää tallentamalla liittimien geometria-arvot esimerkiksi Python-skriptillä täysillä desimaaleilla. Ilman näitä pieniä desimaalieroja pelkällä skriptillä olisi erittäin helppo etsiä liittimille parit. Vertailevan skriptin tueksi täytyykin käyttää Dynamon omaa solmua ”Geometry.ClosestTo”, kuten aiemmissa kappaleissa mainittiin.

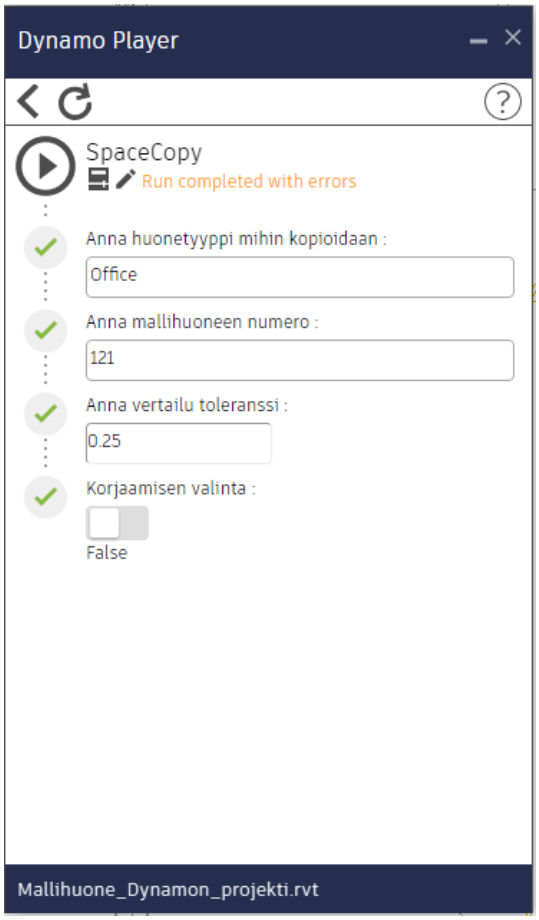
Ohjelmassa on tällä hetkellä ylimääräisen kopioinnin vertailussa rajoituksena se, että kun laitteet ja putkistot on kopioitu huoneisiin ja mikäli mallihuoneeseen lisätään heti sen

jälkeen jokin laite ja kopiointi ajetaan uudelleen, tulee kopioinneissa päällekkäisyyksiä ja virheitä. Tämä johtuu siitä, että olemassa olevien laitteiden lista ei päivity ensimmäisen ajon jälkeen. Rajoitus voidaan välttää ajamalla ohjelma uudelleen heti ensimmäisen kopioinnin jälkeen, jolloin listat päivittyvät ajan tasalle.

5 MALLIHUONE DYNAMON KÄYTTÄMINEN OSANA MALLINTAMISTA

5.1 Dynamo-skriptin käyttöönotto

Valmista Dynamoa voidaan käyttää kahdella tapaa, suoraan joko ohjelmointiympäristöstä tai sitten Dynamo Playerillä. Kun käytetään Dynamo Playeriä, ei tarvitse aukaista itse Dynamon ohjelmointiympäristöä ollenkaan. Dynamo-ohjelmaan voidaan laittaa solmuja tuloiksi tai lähdöiksi, jotka tulevat näkyviin ja käytettäviksi Dynamo Playeriin. Kuviossa 20 on Dynamo Playerin perusnäkymä. Ohjelmalle on annettu tuloiksi huonetyyppi, mallihuoneen numero, vertailutoleranssi ja valinta, halutaanko ohjelman korjaavan laitteiden sijainteja.



Kuvio 20. Dynamo Playerin näkymä Dynamoiden käyttöön (Kasari, 2023).

5.2 Ohjelman koekäyttö ja jatkokehitys

Kun mallihuone Dynamo toimi halutulla tavalla, otettiin se sisäiseen koekäyttöön Swecon kehitysorganisaatiolla. Koekäytön tarkoituksena on testata mm. ohjelman toimintaa erilaisilla lähtötiedoilla ja järjestelmillä laajemmassa projektissa. Testauksen perusteella ohjelmasta löydetään merkittävimmät epäkohdat ja kehityskohteet, jotta ohjelma saataisiin käyttöön Swecon kaikille suunnittelijoille.

Koekäytön aikana ohjelmasta löydettiin mm. seuraavia puutteita ja bugeja:

- Kopioitujen poistokanavien eriste ei kopioidu kanavien mukana.
- Poistokanavia siirtäessä kanavat siirtyvät kaksinkertaisesti.

Koekäytön aikana ilmenneet merkittävimmät kehityskohteet:

- Mallihuoneen elementtien kopiointiin tarvitaan valinnat eri järjestelmien laitteille, jotka kopioidaan, esim. LJ, IV, VV, sähkö, RAU yms.
- Kopioitavien huoneiden valitsemiseen lisätään vaihtoehto huoneiden valitsemiseen ns. käsin klikkaamalla.
- Kanavat ja putket pitää yhdistää lähimpiin runkoihin.
- Ohjelman rakennetta pyritään selkeyttämään ja poistamaan kaikki ylimääräinen ohjelman ajon nopeuttamiseksi.

Lisäksi valmiin Dynamo-ohjelman käytettävyyden kannalta on merkittävää, että sen käyttäminen ei tarvitsisi erillisiä kirjastoja, koska muuten jokaisen käyttäjän tulisi ensimmäisenä ladata oikeat versiot ohjelman vaatimista kirjastoista. Useimpien kirjastojen solmut voidaan niin sanotusti purkaa osiin, jolloin kirjaston solmusta tulee joukko Dynamon omia solmuja ja mahdollisesti jokin Python-lohko, josta kyseinen räätälöity solmu on tehty. Käytettävyyden kannalta toinen merkittävä asia on ohjelman ohjeistuksien ja dokumenttien laadinta.

6 TULOKSET

6.1 Työn tavoitteiden toteutuminen

Työn tavoitteena oli kehittää ja valmistaa talotekniikan mallintamiseen Revit-mallinnusohjelmistolle ohjelma, joka kopioisi yhden mallihuoneen sisältämät talotekniset laitteistot ja putkistot muihin samantyyppisiin huoneisiin. Ohjelma toteutettaisiin visuaalisella ohjelmointityökalulla Dynamolla. Valmiilla ohjelmalla vastattaisiin tarpeeseen automatisoida osaa talotekniikan mallintamista ja näin tuoda lisää tehokkuutta suunnitteluun. Lisäksi ohjelmalla helpotettaisiin luonnosvaiheen suunnitelmien laatimista sekä mahdollistettaisiin suunnitelmien vaihtoehtotarkastelua.

Työssä saatiin kehitettyä ja valmistettua Dynamo-ohjelma, joka toiminnaltaan vastaa sille annettuihin tavoitteisiin. Lisäksi ohjelmalle saatiin tehtyä jo käytettävyyden kannalta merkittäviä toimintoja, kuten ylimääräisen kopioinnin estäminen.

6.2 Työssä saadut tulokset

Tämän kehitystyön tuloksena saatiin valmistettua ohjelma, joka vastaa sille annettuihin tavoitteisiin ja luvussa 4.1 esitettyihin vaatimuksiin. Ohjelma saatiin Swecon kehitysorganisaatiolle koekäyttöön, jonka avulla sille löydettiin merkittävimmät puutteet ja kehityskohdat, jotta ohjelma saadaan kehitettyä laajempaan käyttöön. Kuvat valmiin ohjelman sisällöstä löytyvät erillisestä liitteestä yksi, ja ohjelmassa käytetyt Python-skriptit löytyvät liitteestä kaksi.

7 YHTEENVETO JA POHDINTA

7.1 Yhteenveto

Tämän opinnäytetyön tarkoituksena oli kehittää ja valmistaa ohjelma visuaalisella ohjelmointikäyttöliittymällä Dynamolla, millä vastattaisiin yhteen tietomallintamisen keskeisimmistä automatisointitarpeista. Ohjelman avulla pystyttäisiin kopioimaan ja mallintamaan vaivatta keskenään samankaltaisia huoneita ja näin lisäämään tehokkuutta suunnitteluun.

Kehitystyö alkoi ohjelman vaatimuksien selvityksellä ja ohjelman rakenteen suunnittelulla. Tämän jälkeen aloitettiin itse ohjelman ohjelmointi. Kehitystyön aikana merkittävimpiä haasteita olivat listojen käsittelyt sekä laitteiden ja putkistojen liittimien yhdistäminen. Listojen käsittelyissä huomattiin, että Python-skripteillä saatiin selkeytettyä ja helpotettua listojen lukua ja vertailua merkittävästi. Liittimien yhdistämisien tuomat vaikeudet puolestaan johtuvat osaltaan Revit-ohjelmiston rajoituksista ja puutteista. Suurimpana puutteena esimerkiksi on, että liittimillä ei ole muiden elementtien tapaan yksilöllistä ID-tunnusta, joilla helpotettaisiin liittimien käyttöä merkittävästi. Elementtien geometria-arvojen erittäin pienet desimaalierot toivat osaltaan vaikeuksia etsittäessä liittimien pareja. Desimaalierot tulivat selville vasta, kun ne tallennettiin Python-skriptillä tekstimuodossa listaan. Ratkaisuiksi haasteisiin olivat solmut, joilla voitiin vertailla kahden listan arvoista keskenään lähimpiä arvoja sekä solmut, jotka kertovat, osuuko kaksi arvoa geometrialtaan toisiinsa. Näiden lisäksi tehdyillä Python-skripteillä saatiin liittimien yhdistämiset lopulta toimimaan. Kun ohjelman kopioinnin toimiessa halutulla tavalla, kehitettiin ja lisättiin ohjelmaan käytettävyyden parantamiseksi toiminnot olemassa olevien laitteiden siirtelyyn ja vertailut kopioitavien ja olemassa olevien laitteiden välille, jotta vältetään turhat kopioinnit.

Kehitystyön lopputuloksena saatiin valmis ohjelma, joka vastaa sille annettuja vaatimuksia ja joka voitiin ottaa koekäyttöön Swecon kehitysorganisaatiossa. Koekäytön tuloksena ohjelmasta löydettiin merkittävimmät puutteet ja tärkeimmät jatkokehityksen kohteet. Ohjelman kehitystyö jatkuu Swecon kehitysorganisaation alaisuudessa.

7.2 Pohdinta

Tämä opinnäytetyö osoitti tietomallintamisen tuomat mahdollisuudet suunnittelun tukena. Tietomallinnuksessa voidaan automatisointia käyttää hyvinkin laajasti, mutta suunniteltaessa ohjelmaa täytyy perehtyä ohjelman vaatimaan työmäärään perusteellisesti. Kaikkea ei kuitenkaan ole taloudellista ja kannattavaa automatisoida työmäärän vuoksi.

Mikäli ohjelman kehittämiseen olisi ollut käytettävissä enemmän aikaa, olisi ohjelman rakennetta ollut mielenkiintoista yksinkertaistaa monimutkaisemmilla Python-skripteillä. Skriptien avulla ohjelma olisi selkeytynyt ja supistunut merkittävästi. Monimutkaisempien Python-skriptien tekeminen olisi toisaalta vaatinut huomattavan määrän lisää perehtymistä. Lisäksi ohjelmaa olisi saanut selkeytettyä käyttämällä tehokkaammin DesignScriptiä ja koodilohkoja, mutta näiden käyttäminen tuli selvemmäksi vasta kehitystyön loppuvaiheessa.

Tulevaisuudessa olisi mielenkiintoista selvittää vielä enemmän, mihin kaikkeen tietomallintamisen automatisointia voitaisiin käyttää, koska tietomallintaminen tulee tulevaisuudessa entisestään lisääntymään ja korvaamaan perinteistä suunnittelua tasokuvien ja säätökaavioiden osalta. Ennen kuin suunnittelu siirtyy kokonaan tietomallintamiseen, olisi mielenkiintoista selvittää, kuinka tietomallia voitaisiin tällä hetkellä hyödyntää perinteisen suunnittelun rinnalla, esimerkiksi rakennusautomaation säätökaavioiden kanssa. Näiden asioiden ohella aion tulevaisuudessa perehtyä, kuinka tehdä C#-ohjelmointikielellä työkaluja mallintamiseen ja suunnittelun tueksi, sekä selvittää enemmän Revitin skriptikirjastosta pyRevitistä. PyRevit-kirjasto mahdollistaa Python-skriptien ajon suoraan Revitissä ilman Dynamoa. PyRevitin etuna on myös, että sen toiminnot saadaan helposti käyttöön, koska ne saadaan lisättyä Revitin työkaluihin omalle välilehdelleen, josta Pythonilla tehdyt toiminnot voidaan ajaa.

LÄHTEET

Autodesk. (i.a.-a) *Exporting to ODBC.* Haettu. 12.2.2023. https://help.autodesk.com/view/RVT/2021/ENU/?guid=GUID-57CE0EAA-A33B-4DB9-A5F2-81A66F68353B

Autodesk. (i.a.-b). *Revitin tärkeimmät ominaisuudet.* Haettu 11.1.2023, https://www.autodesk.fi/products/revit/features

Autodesk. (i.a.-c). *Revit: BIM-ohjelmisto suunnittelijoille, rakentajille ja tekijöille: Mikä on Revit?* Haettu 11.1.2023, https://www.autodesk.fi/products/revit/overview?term=1-YEAR&tab=subscription

Building Smart Finland. (27.3.2012a). *YTV – Yleiset tietomallivaatimukset 2012: Osa 1. Yleinen osuus. https://drive.buildingsmart.fi/s/7FPE7tGocYZw8BY*

Building Smart Finland. (27.3.2012b). *YTV – Yleiset tietomallivaatimukset 2012: Osa 4. Talotekninen suunnittelu.* https://drive.buildingsmart.fi/s/S2p59nX27yZ2LzM

Construction Global. (16.3.2020). *The Brilliance of BIM: The best of BIM.* https://constructiondigital.com/technology-and-ai/the-brilliance-of-bim

DynamoBIM. (i.a.-a). *Anatomy of a visual program.* https://primer.dynamobim.org/03_Anatomy-of-a-Dynamo-Definition/3_anatomy-of-a-dynamo-definition.html

DynamoBIM. (i.a.-b). *Code Blocks and DesignScript.* https://primer.dynamobim.org/07_Code-Block/7_Code-Blocks-and-Design-Script.html

DynamoBIM. (i.a.-c). *Developing for Dynamo.* https://developer.dynamobim.org/03-Development-Options/3-0-developing-for-dynamo.html

DynamoBIM. (i.a.-d). *Dynamo Library.* https://primer.dynamobim.org/03_Anatomy-of-a-Dynamo-Definition/3-3_dynamo_libraries.html

DynamoBIM. (i.a.-e). *Managing Your Program.* https://primer.dynamobim.org/03_Anatomy-of-a-Dynamo-Definition/3-4_best_practices.html

DynamoBIM. (i.a.-f). *Nodes.* https://primer.dynamobim.org/03_Anatomy-of-a-Dynamo-Definition/3-1_dynamo_nodes.html

DynamoBIM. (i.a.-g). *Python.* https://primer.dynamobim.org/10_Custom-Nodes/10-4_Python.html

DynamoBIM. *(i.a.-h). Python and Revit.* https://primer.dynamobim.org/10_Custom-Nodes/10-5_Python-Revit.html

DynamoBIM. (i.a.-i). *Selecting.* https://primer.dynamobim.org/08_Dynamo-for-Revit/8-2_Selecting.html

DynamoBIM. (i.a.-j). *The Workspace.* https://primer.dynamobim.org/02_Hello-Dynamo/2-3_the_workspace.html

DynamoBIM. (i.a.-k). *What is Dynamo?* Haettu 16.1.2023, https://primer.dynamo-bim.org/en/01_Introduction/1-2_what_is_dynamo.html

DynamoBIM. (i.a.-l). *What is Zero-Touch?* https://primer.dynamobim.org/11_Packages/11-5_Zero-Touch.html

DynamoBIM. (i.a.-m). *What's a List?* https://primer.dynamobim.org/06_Designing-with-Lists/6-1_whats-a-list.html

DynamoBIM. (i.a.-n). *What's Code Block?* https://primer.dynamobim.org/07_Code-Block/7-1_what-is-a-code-block.html

DynamoBIM. (i.a.-o). *Wires.* https://primer.dynamobim.org/03_Anatomy-of-a-Dynamo-Definition/3-2_wiring_programs.html

Ilmatieteen laitos. (22.12.2015). *Ilmastonmuutos vähentää rakennusten lämmitysenergian tarvetta.* https://www.ilmatieteenlaitos.fi/uutinen/126771699

Laine, T. (2008). *Tuotemallintaminen talotekniikkasuunnittelussa.* Rakennustieto.

MagiCAD. (i.a.-a). *MagiCAD LVIS-suunnitteluun.* Haettu 11.1.2023, https://www.magi-cad.com/fi/mita-magicad-tarjoaa-lvis-suunnitteluun/

MagiCAD. (i.a.-b). *MagiCAD LVIS-suunnitteluun: MagiCAD-sovellukset.* Haettu 11.1.2023, https://www.magicad.com/fi/mita-magicad-tarjoaa-lvis-suunnitteluun/

Microsoft. (4.12.2022). *What is DLL.* https://learn.microsoft.com/en-us/troubleshoot/windows-client/deployment/dynamic-link-library

Nordic BIM Group. (i.a.-a). *Tietomallinnuksen ABC: Mitä tiedostoja BIM hyödyntää.* https://www.nordicbim.com/fi/bim-tietomallinnuksen-abc

Nordic BIM Group. (i.a.-b) *Tietomallinnuksen tärkein ISO-standardi suomennetaan – Mitä se tarkoittaa?* https://www.nordicbim.com/fi/bimblogi/tietomallinnuksen-t%C3%A4rkein-iso-standardi-suomennetaan-mit%C3%A4-se-tarkoittaa

Python. (i.a.). *What is Python? Executive Summary.* https://www.python.org/doc/essays/blurb/

Rantala, E., Mäkinen, R., Piikkilä, V., Siren, K., Piira, K., Hast, J., Federley, M., Seisto, A., Sarja, A., Åström, G. (2015). *Käyttäjälähtöinen älyrakennus: Suunnittelu, rakentaminen, käyttö ja ylläpito.* Suomen Rakennusinsinöörien Liitto RIL.

Rava3pro. (i.a.). *Mikä on RAVA3pro?* http://www.rava3pro.fi/

Solibri. (24.3.2022). *BIM ja tietomallit rakentamisessa: Mitä on BIM?* https://www.solibri.com/fi/ajankohtaista/bim-ja-tietomallit-rakentamisessa

Sweco. (i.a.). *Tietoa Swecosta.* Haettu 16.1.2023, https://www.sweco.fi/tietoa-swecosta/

Talotekniikka-lehti. (1.11.2019). *Älykäs taloautomaatio säästää energiaa ja helpottaa arkea.* https://talotekniikka-lehti.fi/alykas-taloautomaatio-saastaa-energiaa-ja-helpottaa-arkea/

Talotekninen teollisuus ja kauppa (Talteka). (i.a.). *Talotekniikka on elämisen tekniikkaa.* https://talteka.fi/tietoiskut/talotekniikka-on-elamisen-tekniikkaa/

LIITTEET

Liite 1. Mallihuone Dynamon rakenne (Vain Swecon sisäiseen käyttöön)

Liite 3. Mallihuone Dynamon tarkemmat selostukset (Vain Swecon sisäiseen käyttöön)

Liite 3. Mallihuone Dynamon tarkemmat selostukset (Vain Swecon sisäiseen käyttöön)