SAVONIA

OPINNÄYTETYÖ - **AMMATTIKORKEAKOULUTUTKINTO**
TEKNIIKAN JA LIIKENTEEN ALA

TEKLA STRUCTURES SÄHKÖASENNUSTEN MALLINNUSTYÖKALUN KEHITTÄMINEN OPEN API OHJELMOINTIRAJAPINNALLA

TEKIJÄ: Jussi Junkkarinen

SAVONIA-AMMATTIKORKEAKOULU

OPINNÄYTETYÖ
Tiivistelmä

Koulutusala Tekniikan ja liikenteen ala	
Koulutusohjelma Rakennustekniikan koulutusohjelma	
Työn tekijä(t) Jussi Junkkarinen	
Työn nimi Tekla Structures sähkötyökalun kehittäminen Open API ohjelmointirajapinnalla	
Päiväys 21.11.2014	Sivumäärä/Liitteet 47/45
Ohjaaja(t) lehtori Viljo Kuusela, lehtori Harry Dunkel	
Toimeksiantaja/Yhteistyökumppani(t) Rakennussuunnittelutoimisto Sormunen & Timonen Oy	

Tiivistelmä

Tämän opinnäytetyön tavoitteena oli kehittää Telka Structures -rakennesuunnitteluohjelmistossa toimiva työkalu. Työkalun tarkoituksena on pystyä mallintamaan seinäelementtien sähköistykset. Työkalun kehittämistehtävään kuuluivat työkalun ohjelmointi Open API –ohjelmointirajapinnalla sekä työkalulla tuotettujen sähköistysten esittäminen piirustuksissa. Opinnäytetyön toimeksiantajana toimi Rakennussuunnittelutoimisto Sormunen & Timonen Oy.

Opinnäytetyössä keskityttiin käsittelemään kehitystyössä käytettyjä työkaluja ja menetelmiä. Lisäksi on käsiteltiin työkalun kehittämisen vaiheita, kehitetyn työkalun toiminnallisuuksia sekä sen testausta. Raportoinnin perustana on käsitelty lähinnä Tekla Structures -ohjelmistosta olevia dokumentteja sekä aiheeseen liittyviä internet-lähteitä.

Opinnäytetyön tuloksena saatiin aikaan Tekla Structures ohjelmistossa käytettävä työkalu, jolla voidaan mallintaa seinäelementtien sähköistykset ja sisällyttää niistä saatava informaatio elementeistä tuotettaviin piirustuksiin. Työkalu kehitettiin käyttämällä Microsoft Visual Studio Express 2013 -ohjelmointotyökalua.

Ohjelmointikielenä käytettiin Microsoft C# -ohjelmointikieltä. Työkalun käyttöliittymän suunniteltiin käytettiin Microsoft .NET –arkkitehtuuria hyödyntävää Windows Forms –tekniikkaa.

Avainsanat
Tekla, tietomalli, API, plugin, sähköasennukset

SAVONIA UNIVERSITY OF APPLIED SCIENCES

THESIS
Abstract

Field of Study Technology, Communication and Transport			
Degree Programme Degree Programme In Construction Engineering			
Author(s) Jussi Junkkarinen			
Title of Thesis Development of Tekla Structures Electricity Tool with Open API Programming Interface			
Date	21 November 2014	Pages/Appendices	47/45
Supervisor(s) Mr. Viljo Kuusela, Senior Lecturer, Mr. Harry Dunkel, Senior Lecturer			
Client Organisation /Partners Rakennussuunnittelutoimisto Sormunen & Timonen Oy			

Abstract

This final year project was commissioned by Rakennussuunnittelutoimisto Sormunen & Timonen Oy. The aim of this Bachelor's Thesis was to develop a modeling tool for electrical installations in Tekla Structures designing software. The purpose of this tool is to enable electrical installations in precast wallpanels. The development process includes creating the tool with Tekla Open API -programming interface and showing the electrical installations in drawings.

This thesis focuses on discussing the tools and methods used in the development process. In addition to this the thesis also discusses the steps of creating the modeling tool, the tool itself and testing of the tool. This thesis is mostly based on documents about Tekla Structures -software and related internet sources. The plugin was created using Microsoft Visual Studio Express 2013 -programming tool. The programming language used was Microsoft C#. The user interface was created with Windows Forms -technology which exploits Microsoft .NET –architecture.

As a result, the development process generated a Tekla Structures modeling tool which is used to model electrical installations in precast wallpanels and transfer information from the created model objects to the drawings made of the precast elements.

Keywords
Tekla, BIM, API, plugin, electrical

ALKUSANAT

Kehitystyön aihe ja idea tulivat työn tilaajana toimineelta Rakennussuunnittelutoimisto Sormunen & Timonen Oy:ltä. Tilaajalla oli tarve kehittää Tekla Structures -ohjelman elementtirakenteiden sähköasennuksiin liittyviä työkaluja. Koska Tekla Structures -ohjelman omat työkalut sähköasennusten mallinnusta varten ovat varsin puutteelliset, päätettiin alkaa kehittämään omia työkaluja. Kehittämisen tarkoituksena oli myös voida luopua aikaisemmasta tavasta esittää sähköasennukset dwg -tiedostojen avulla.

Kehitystyön toteuttamiselle oli olemassa kaksi varteenotettavaa tapaa. Työkalu voitaisiin toteuttaa Tekla Structures -ohjelmiston Custom Component työkaluna. Custom Componentin valitseminen olisi kuitenkin asettanut työkalulle huomattavia rajoitteita ja sen toimillallisuudet olisivat rajoittuneet Custom Component Editorin asettamiin rajoihin. Työkalu päätettiin toteuttaa Tekla Oyj:n kehittämällä Open API -ohjelmointirajapinnalla, jolla työkalu voitaisiin toteuttaa huomattavasti Custom Component Editoria laajemmilla mahdollisuuksilla. Ainoana rajoitteena toimisi kehitystyön tekijän tietämys ohjelmoinnista ja Open API:n toiminnasta.

Työssä suurena apuna ovat olleet Tekla Oyj käyttäjätuen yhteyshenkilöt, joilla on ollut hyvin vastaanottava suhtautuminen työn tekijän loputtomiin kysymyksiin. Lisäksi tilaajan ymmärtäväinen suhtautuminen kehitystyön ajalliseen kestoon.

Yhteistyö osapuolia kiittäen
Kuopiossa syyskuussa 2014

Jussi Junkkarinen

SISÄLTÖ

1 JOHDANTO

1.1 Taustat ja tavoitteet

Rakennussuunnittelutoimisto Sormunen & Timonen Oy on vuonna 1979 perustettu yksityisessä omistuksessa oleva suunnittelutoimisto. Yhtiössä työskentelee noin 25 rakennussuunnittelun ammattilaista joista noin 2/3 työskentelee rakennesuunnittelutehtävissä. Yli 30-vuotisen historian aikana yhtiö on toteuttanut mittavan määrän eri tyyppisiä suunnittelukohteita Suomessa sekä ulkomailla. Vankan rakennussuunnittelu osaamisen ohella yhtiöllä on kiinnostusta kehittää toimintaansa myös kaiken aikaa kasvavassa tietomallinnuksessa. Yhtiössä on jo usean vuoden ajan ollut käytössä Tekla Structures -rakennesuunnitteluohjelmisto.

Opinnäytetyössä Tekla Structures -rakennesuunnitteluohjelmistoon kehitettävää sähköasennusten mallinnustyökalua päätettiin alkaa kehittämään Tekla Open API -ohjelmointirajapinnalla. Open API on Tekla Corporationin kehittämä olio-teknologiaan pohjautuva ohjelmointirajapinta. Sen avulla voidaan kehittää erilaisia mallinnus- ja piirustustyökaluja Tekla Structures -ohjelmistossa käytettäviksi. Open API -ohjelmointirajapinta valittiin kehitystyön välineeksi sen monipuolisten mahdollisuuksien ja tulevaisuuden kannalta suuren potentiaalin vuoksi.

Opinnäytetyön työelämäyhteys on suoraan verrattavissa kehitystyön tuloksena syntyvän työkalun toimivuuteen. Toimiessaan työkalulla voidaan saavuttaa merkittävää ajallista ja taloudellista säästöä seinäelementtien suunnittelussa.

1.2 Lyhenteet ja määritteet

API	Tekla Open API –ohjelmointirajapinta.
C#	C# eli C sharp on Microsoft-yhtion kehittämä ohjelmointikieli.
CAD	Computer Aided Design eli tietokone avusteinen suunnittelu.
CC	Tekla Structures –rakennesuunnitteluohjelmassa toimiva työkalu, jolla voidaan luoda objekteja, liitoksia, saumoja ja detaljeja.
CC Editor	Tekla Structures Custom Componenttien luomiseen tarkoitettu sovellus.
DWG	CAD –ohjelmien käyttämä tiedostomuoto.
IFC	Tietomalliohjelmistojen yhteinen mallien kuvaustapa.
Mallinnustila	Teklan tila, jossa käsiteltävää tietomallia työstetään.
Piirustustila	Teklan tila, jossa käsitellään Teklalla tuotettavia piirustuksia.
Plugin	Ohjelmiston tai sovelluksen kanssa vuorovaikutuksessa toimiva liitännäinen.
Template	Templateja käytetään älykkäiden taulukoiden ja graafisten kenttien tuottamiseen ja tiedon esittämisessä Teklan piirustuksissa.
Tietomalli	Digitaalisessa muodossa olevan rakennelman 3-ulotteinen esittäminen ominaisuustietoineen
TS / Tekla	Tekla Structures –rakennesuunnitteluohjelma.
VS	Microsoft Visual Studio ohjelmankehitysympäristö.

2 TYÖKALUN KEHITTÄMISTEHTÄVÄN KUVAUS

Laadukkaan betonielementtin valmistamiseen tarvitaan yhteistyötä työmaan, elementtitehtaan ja eri suunnitteluosapuolten välillä. Sähköasennusten virheet ja puutteet ovat työläitä korjata, joten suunnittelussa ja elementtien valmistuksessa tulee olla huolellinen. Elementtisuunnittelussa käytettävillä ohjelmistoilla ja työkaluilla on suuri vaikutus laadukkaan betonielementin suunnittelussa (Betonielementtien sähköasennukset 2012.)

Tässä opinnäytetyössä on tarkoitus tehdä kehitystyötä Tekla Structures – ohjelmistoon ja kehittää tarvittavat työkalut elementtirakenteiden sähköasennusten mallintamiseen. Kehittämistehtävän tavoite on voida mallintaa seinäelementtien sähköasennukset yleisimmin esiintyvien tapausten osalta. Lisäksi työn edetessä kehitettävän työkalun ominaisuuksia on tarkoitus lisätä sitä mukaa, kuin työn tekijän tietämys ja taidot työn edetessä kehittyvät. Mallinnustyökalun lisäksi työssä kehitetään tarvittavat menetelmät mallinnettujen sähköasennusten esittämiseksi elementtipiirustuksissa ja piirustusluetteloissa.

2.1 Kehittämistehtävän lähtökohdat

Työn lähtökohtana on tilaajayrityksen nykyinen tapa esittää sähköasennukset Teklan piirustuksissa niihin liitettävien dwg -piirustusten avulla. Nykyinen tapa tehdä piirustukset sähköistämättömistä elementeistä, lähettää ne sähkösuunnittelijalle ja palauttaa sähkösuunnittelijan piirustukset takaisin Teklaan on varsin työläs ja aikaa vievä tapa. Aikaa kuluu huomattavasti, kun elementtisuunnittelijan on käsiteltävä elementtien piirustukset kahteen kertaan sekä niitä luodessaan että sähköjä niihin siirrettäessä. Lisäksi aikaa kuluu piirustusten ollessa sähkösuunnittelijalla. Lisäksi virheiden mahdollisuus on huomattavasti suurempi, kun sähköjä ei ole voitu mallintaa tietomalliin jossa on helppo tehdä tarkastelua eri rakenne- ja talotekniikkaobjektien välillä.

2.2 Kehittämistehtävän tavoite

Kehitystyön tavoite on voida mallintaa sähköasennukset ja saada ne esitettyä elementtipiirustuksissa ja listata mallinnettujen sähköasennusobjektien tieto piirustusten luetteloihin. Mallina sähköasennusten esittämiseksi piirustuksissa käytettään tilaajan aikaisempien suunnittelukohteiden elementtipiirustuksia sekä betoniteollisuus ry:n betonielementtien sähköasennukset 2012 -ohjetta.

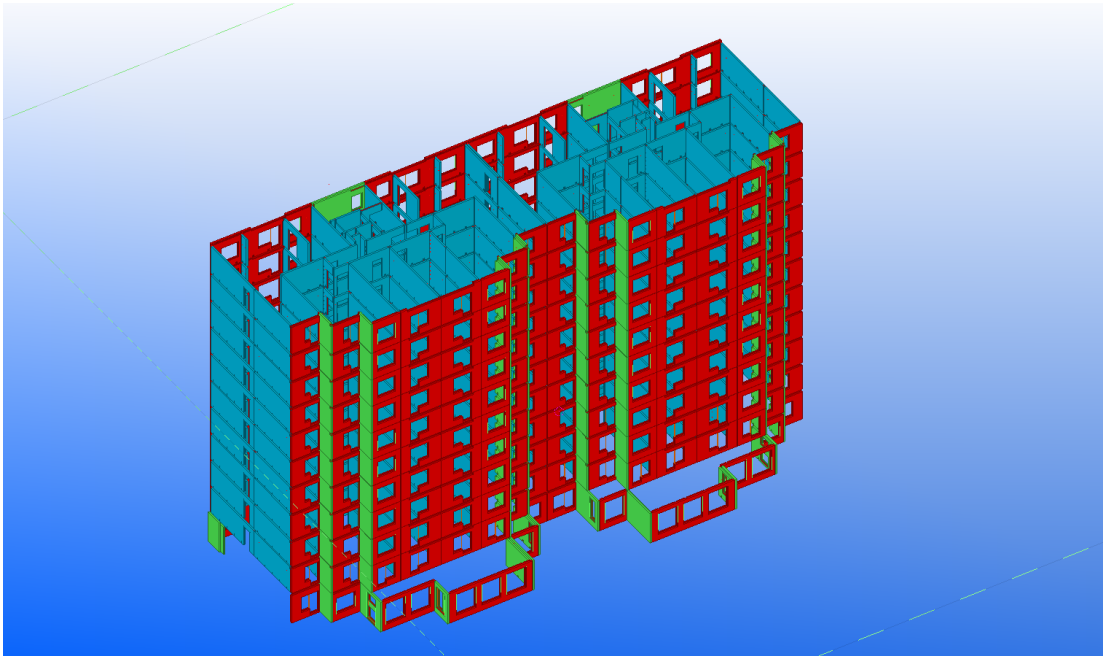
3 KEHITYSTYÖSSÄ KÄYTETYT TYÖKALUT

Ohjelmistokehitystyössä on syytä tuntea käytetyt työkalut ja niiden toiminnallisuudet, jotta voidaan välttyä projektin edetessä mahdollisesti ilmenevistä yhteensopivuusongelmista. Kehitystyön kohteena on Tekla Oyj: kehittämä Tekla Structures –rakennesuunnitteluohjelmisto. Tekla Oyj on myös kehittänyt Open API –ohjelmointirajapinnan, jolla ohjelmistokehittäjät ja ohjelmointia tuntevat Tekla Structures –käyttäjät voivat kehittää uusia työkaluja ja sovelluksia käytettäväksi Tekla Structures –ohjelmistossa.

3.1 Tekla Structures

Tekla Structures on Tekla Oyj:n kehittämä rakennesuunnitteluohjelmisto, jolla saadaan aikaan IFC-standardin mukainen rakennuksen rakenteiden tietomalli. Tekla Structures -ohjelmistolla on käyttäjiä yli 100 maassa. Teklan ohjelmistot ovat lähtökohtaisesti avoimia tietomalliohjelmistoja, joten ne toimivat yhteen useiden muiden ratkaisutoimittajien ohjelmistojen kanssa.

Tekla Structuresilla tuotetut mallit sisältävät sen tarkan, luotettavan ja yksityiskohtaisen tiedon, jota tarvitaan onnistuneeseen rakentamisen tietomallinnukseen ja toteutukseen. Tekla Structures tarjoaa sujuvamman työnkulun ja toteuttamiskelpoiset tietomallit. (TEKLA 2014a.)



Kuva 1 Kuvakaappaus Tekla Structures -seinäelementtimallista (Sormunen & Timonen, 2014)

3.2 Tekla Open API

Tekla Open API on Tekla Oyj:n kehittämä avoin ohjelmointirajapinta. API tulee sanoista Application Programming Interface. Open API mahdollistaa erilaisten sovellusten ja lisätoiminnallisuuksien kehittämisen Tekla Structures –rakennesuunnitteluohjelmistoon. Tekla Open API käyttää Microsoft Corporationin kehittämää .NET –teknologiaa. Open API:lla kehitettyjä työkaluja kutsutaan sovelluslaajennuksiksi.

Tekla Open API:lla esimerkiksi (Tekla 2014b.)

- nauhoittaa ja suorittaa käyttöympäristön toimintoja, joilla voidaan automatisoida rutiinin omaisia tehtäviä kuten päivittäisien raporttien tuottamista
- kehittää automatisoituja työkaluja usein tarvittujen mallinnusobjektien ja -tapausten luomiseksi
- luoda yhteyksiä Tekla Structuresin ja muiden ohjelmistojen välillä
- kehittää uusia työkaluja ja toiminnallisuuksia Tekla Structuresiin.

3.3 Microsoft Visual Studio

Kehitystyössä käytetään Microsoftin Visual Studio Express 2013 -ohjelmointityökalua. Visual Studio on ohjelmointityökalu, jossa voidaan käyttää useita eri ohjelmointikieliä. Sillä voidaan käyttää esimerkiksi kieliä C#, C++ ja Visual Basic. Visual Studiossa sovellusten käyttöliittymä voidaan rakentaa varsin helposti käyttäen Windows Form -käyttöliittymäkomponentteja.

3.4 Ohjelmointikieli

C# on moderni, olio-perustainen ja tyyppiturvallinen ohjelmointikieli. C#:n kehittymiseen ovat vaikuttaneet monet muut olio-orientoituneet kielet, kuten C++, SmallTalk ja Java. Tarkoituksena oli kehittää kieli, jossa on laajentuva tyyppijärjestelmä, 1. luokan komponenttien tuenta, luotujen ohjelmien vakaus ja näiden versioiden hallinta sekä yhteensopivuus aiemmin tehtyjen ohjelmistojen ja komponenttien kanssa. Tavoitteena oli myös kehittää kieli, joka yhdistää Microsoftin Visual Basicin tuottavuuden ja C++:n tehon (Sivonen, 2004, Helsingin yliopisto).

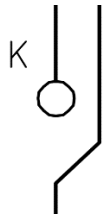
4 TYÖKALUN KEHITTÄMISPROSESSI

Kehitystyön toteuttaminen aloitetaan perehtymällä elementtien sähköasennuksista olemassa oleviin ohjeisiin ja tilaajan aikaisemmista suunnittelukohteista saataviin piirustuksiin. Kehitystyön ohjeeksi otettiin Betoniteollisuus ry:n vuonna 2012 julkaisema ohje betonielementtien sähköasennuksista. Kehitettävällä TS:n työkalulla on tarkoitus voida tuottaa tietomalliobjektit yleisimmin esiintyville sähköasennustapauksille.

Tekla Structures -ohjelmistoon kehitettävällä työkalulla on tarkoitus voida mallintaa seinäelementteihin seuraavat sähköasennusobjektit:

- Rasiat ja päätevaraukset
  Koje-, jako- ja kaksoiskojerasiat sekä varaukset tulee esittää elementtien muottikuvassa. Elementin etupinnassa olevat rasiat ja varaukset esitetään valkoisella. Taka- eli muottipinnassa ne tulee esittää mustalla.

- Putkitukset
  Putkitukset tulee esittää rasioiden tapaan rasteroituna niiden sijaintipinnan mukaan. Jos putken kokoa ei ole kerrottu sähköpiirustuksissa se on JM20 ja sen tunnusta ei myöskään tarvitse esittää elementtikuvissa. Mikäli putken koko on kerrottu ja se poikkeaa JM20:stä, on sen tunnus kerrottava myös elementtikuvissa. Putkien taitokset on voitava esittää elementin muottikuvassa, jotta putkitukset eivät ole ristiriidassa keskenään tai rasioiden kanssa.



Kuva 2 Esimerkki taittotapauksesta

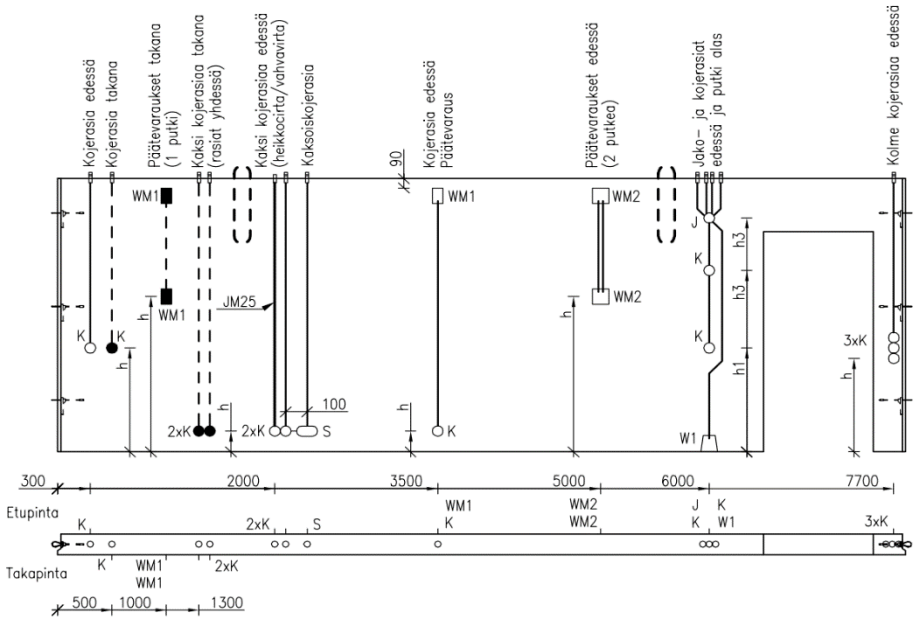
- Nysät
  Rasioihin liitettävät nysät putkituksille. Nysiä ei tarvitse esitetää elementtien muottikuvissa, mutta ne tulee esittää piirustusten sähkötaulukoissa.

- Pääteholkit
  Putkien yläpään pääteholkit. Pääteholkit tulee esittää elementtien muottikuvissa.

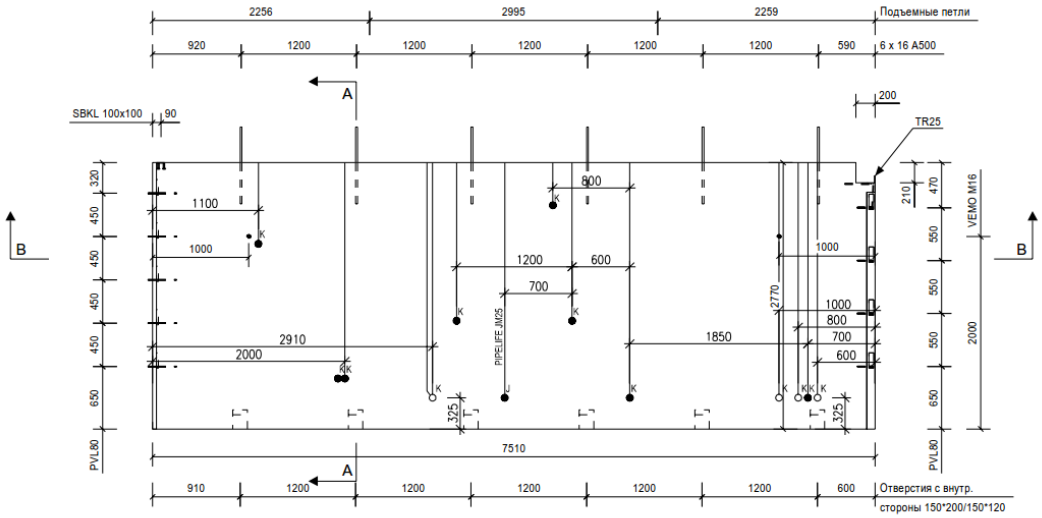
- Rasiat elementin päässä
  Rasiat tulee voida mallintaa myös seinäelementin päähän.

Alla on esimerkkikuva betonielementtien sähköasennukset -ohjeesta. Esimerkin kuvassa havainnollistetaan väliseinäelementin sähköasennusten esittämistapoja. Kuvassa on esitety sähköasennus -komponenteista rasiat, putkitukset ja varaukset sekä niiden tunnukset ja mitoitus. Sähköasennukset elementin etupinnassa on esitetty valkoisella ja elementin taka- eli muottipinnan sähköistykset mustalla. Yhtenäisellä esittämistavalla varmistutaan siitä, että elementtien sähköasennus -komponentit tulevat asennetuksi oikein elementtitehtaalla.

Tämä esimerkkikuva toimii kehitystyön pohjana yhdessä tilaajayrityksen aikaisemmista kohteista saatavien piirustusten kanssa. Kehitystyön Tekla Structures -ohjelmiston pluginilla on tarkoitus kyetä tuottamaan sähköasennus -objektit Teklan mallinnustilassa ja esittää niistä saatava tieto Teklan piirustuksissa.



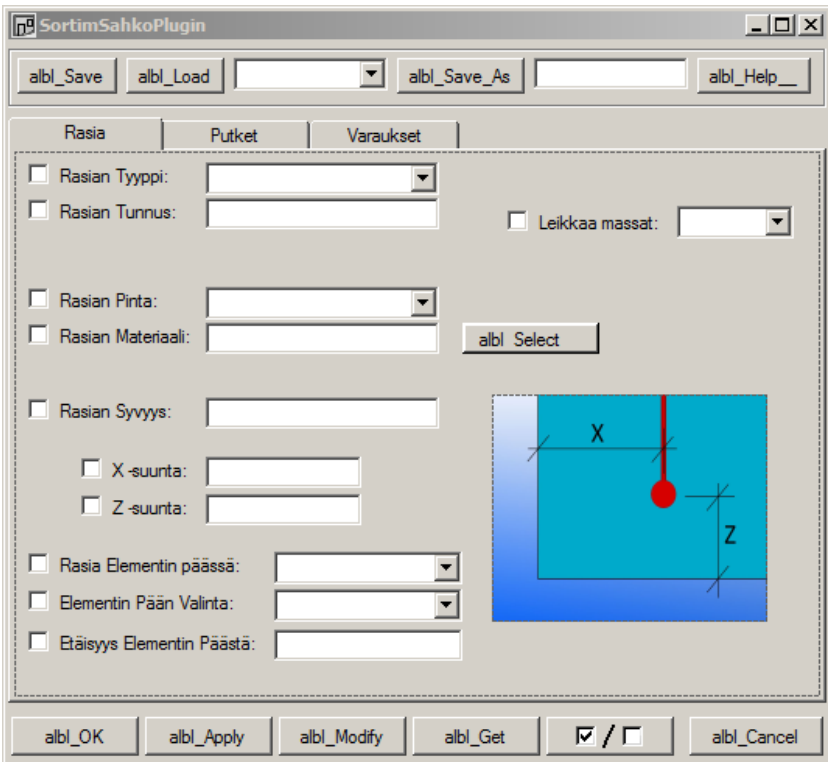
Kuva 3 Esimerkki piirustus (Betonielementtien sähköasennukset, 2012.)



Kuva 4 Esimerkki tilaajayrityksen aikaisemmasta kohteesta (Sormunen & Timonen, 2014.)

4.1 Käyttöliittymä

Kehitettävän työkalun käyttöliittymä toteutetaan Windows Forms -tyyppisenä. Käyttöliittymässä työkalun käyttäjä antaa pluginille tarvittavat tiedot, kuten esimerkiksi rasioiden ja varausten tyypit ja tunnukset sekä putkituskoot ja niiden taittotapaukset. Käyttöliittymän objektien harkitulla asettelulla on huomattavasti merkitystä työkalun käytettävyyden kannalta. Pluginin käyttöliittymän ensimmäinen välilehti on esitetty alla.



Kuva 5 Kehitettävän työkalun ensimmäinen välilehti (Junkkarinen 2014.)

Ensimmäiselle välilehdelle on määritetty käyttöliittymäobjektit, joilla saadaan haettua käyttäjältä vaadittavat tiedot pluginin rasioiden mallintamiselle. Jokaisella objektilla on oma nimi ja tunnus, joilla ne saadaan yhdistettyä osaksi pluginin ohjelmistokoodia. Alla esitettynä rasian tunnus -objektin nimeäminen ja tyypin määrittäminen. Nimeämisen jälkeen rasian tunnus -objekti on käytettävissä plugini ohjelmoitaessa.

Tekla Structures	
AttributeName	**RasiaTun**
AttributeTypeName	**String**
BindPropertyName	
IsFilter	False

Kuva 6 Esimerkki rasian tunnuksen nimeämisestä (Junkkarinen 2014.)

4.2 Työkalun ohjelmointi

Seuraavaksi käsitellään Teklan Structures -ohjelmistoon kehitettävän pluginin rakenteen perus edellytykset. Kehitystehtävän ohjelmointia havainnollistetaan rasioiden mallintamista varten tehdyllä ohjelmoinnilla ja käydään läpi tapauksia putkien taivutusten ohjelmoinnista. Nämä ovat työkalun tärkeimmät toiminnallisuudet sähköasennusten perustapausten mallintamisessa. Lisäksi käydään läpi pluginista saatavan tiedon kulku mallinnusobjekteilta tuotettaviin piirustuksiin.

4.3 Pluginin kehittämisen aloitus

Tekla Structures -ohjelmiston pluginilla on oltava seuraavat pakolliset toiminnallisuudet, jotta pluginin luokkarakenne toimisi oikein ja että pluginin tiedot saadaan välitettyä pluginilta isäntäohjelmistoon.

Pluginilla on oltava yhteys Tekla Open API:n referenssikirjastoihin. Referenssikirjastot (eng. namespace) mahdollistavat ohjelmointi objektien kutsumisen ja sen, ettei referenssiä tarvitse kutsua ohjelmoitaessa erikseen vaan kyetään käyttämään lyhyempää koodia. Alla esimerkki referenssien kutsumisesta ja uudelleen nimeämisestä:

```
using Tekla.Structures;
using Tekla.Structures.Model;
using TSM = Tekla.Structures.Model;
using Tekla.Structures.Model.UI;
using TSMUI = Tekla.Structures.Model.UI;
```

Pluginilla on oltava nimi. Nimen on oltava yksilöllinen. PluginUserInterface osoittaa pluginille käyttöliittymän Form/INP määritykset:

```
[Plugin("SortimSahkoPlugin")]
[PluginUserInterface("SortimSahkoPlugin.MainForm")]
```

Pluginille käyttöliittymän attribuutit on määritettävä StructuresData:ssa. StructuresData määrittää datan, joka voidaan tuoda käyttöliittymästä:

```
public class StructuresData
{
            [StructuresField("RasiaName")]
            public string RasiaName;
            [StructuresField("RasiaTun")]
            public string RasiaTun;
            [StructuresField("RasiaPinta")]
            public int RasiaPinta;
            [StructuresField("RasiaMaterial")]
            public string RasiaMaterial;
            [StructuresField("RasiaClass")]
            public string RasiaClass;
}
```

Plugin tarvitsee yhteyden pluginin attribuuttien eli StructuresData:n ja Tekla Structuresin välillä:

```
public SortimSahkoPlugin(StructuresData data)
{
            MyModel = new Model();
            Data = data;
}
```

InputDefinition metodi, jolla plugin hakee käyttäjältä toimenpiteet pluginin suorittamiseksi. Esimerkiksi kehittämistapauksen plugin pyytää käyttäjää valitsemaan mallista seinän johon sähköasennukset halutaan mallintaa:

```
public override List<InputDefinition> DefineInput()
{
List<InputDefinition> inputList = new List<InputDefinition>();
Picker Picker = new Picker();
int i = 0;
while (i < 1)
{
            Part P = Pick-
            er.PickObject(TSMUI.Picker.PickObjectEnum.PICK_ONE_OBJECT);

            if(P != null)
            {
                        inputList.Add(new InputDefini-
                        tion(P.Identifier));
                        i++;
            }
}
return inputList;
}
```

Lisäksi plugin tarvitsee päämetodin, joka suoritetaan kun käyttäjältä saatava syöte on toteutunut. Päämetodi on pluginin niin kutsuttu älykäs vaihe. Englanniksi käytetään termiä ”business logic”. Päämetodi kuvattu alla:

```
public override bool Run(List<InputDefinition> Input)
{
bool result = false;

            try
            {
                        bool PutkiResult = false;
                        try
                        {
                        // Pluginilla suoritettava koodi tähän
                        }
            }
            catch (Exception exc)
            {
            MessageBox.Show(exc.ToString());
            }
            return result;
}
```

Tekla Structuressissa pluginilla tuotettujen objektien luonti ja muokkaaminen suoritetaan aina samaa kaavaa käyttäen. Jos objekteja muutetaan tai käyttäjä painaa käyttöliittymän modify -painiketta, plugin suoritetaan uudelleen. Yleensä muokkaaminen toimii ilman ylimääräistä työtä. Tekla suorittaa Run() -metodin muuttuneilla parametreilla ilman käyttäjän inputin uudelleen saamista. Tekla myös automaattisesti huolehtii olemassaolevien objektien muokkaamisesta uusien luomisen sijaan (Iiro Ojala, 2012).

4.4 Ohjelmointiesimerkit

Esimerkeillä havainnollistetaan Tekla Open API:lla tuotetun pluginin toiminnallisuuksien takana suoritettavaa ohjelmointikoodia. Esimerkeissä käydään läpi sähkövarauksien rasioiden sekä putkitusten ja niiden taitosten ohjelmointia. Käsiteltävät koodin osat ovat pluginin päämetodista ja sen lisäksi ohjelmoiduista yksityisistä metodeista, joilla voidaan lyhentää varsinaisen päämetodin ohjelmointityötä.

4.4.1 Rasian ohjelmointi

Pluginin rasioiden luomiseksi on tehty yksityinen metodi (eng. private method), joka toimii perusrunkona pluginin rasioiden ohjelmoinnissa. Yksityisellä metodilla pluginille määrittää rasian luonnissa toistuvat tapaukset, kuten rasian tunnuksen, profiilin ja materiaalin hakeminen käyttöliittymästä sekä sen poikkeusten käsittely mikäli pluginia suoritettaessa tulee esiin ristiriitoja pluginin ohjelmistokoodissa.

Rasian yksityisessä metodissa määritettään, että se on yksityinen metodi joka luo rasian Teklan palkkina kahden pisteen väliin:

```
private Beam CreateRasia(Point RasiaStartPoint, Point RasiaEndPoint)
{
```

Rasian yksityinen metodi hakee käyttöliittymän muuttujat ja määrittää ne käytettäväksi rasian luonnissa.

```
GetValuesFromDialog();

Beam Rasia = new Beam(RasiaStartPoint, RasiaEndPoint);
Rasia.PartNumber.Prefix = _RasiaTun;
Rasia.AssemblyNumber.Prefix = _RasiaTun;
Rasia.Name = _RasiaName;
Rasia.Profile.ProfileString = _RasiaProfile;
Rasia.Material.MaterialString = _RasiaMaterial;
Rasia.Class = _RasiaClass;
```

GetValuesFromDialog() –metodi hakee rasian luonnissa käytettävät attribuutit käyttöliittymältä ja asettaa niille oletusarvot, mikäli käyttäjän arvoja ei ole saatavilla.

```
private void GetValuesFromDialog()
{
            _RasiaName = Data.RasiaName;
            _RasiaTun = Data.RasiaTun;

            if (IsDefaultValue(_RasiaName))
            _RasiaName = "K";
            if (IsDefaultValue(_RasiaTun))
            if(_RasiaName == "K")
            {
            _RasiaTun = "AU3.2";
            }
            if(_RasiaName == "J")
            {
            _RasiaTun = "AU19";
            }
            if(_RasiaName == "S")
            {
            _RasiaTun = "AU17.2";
            }
```

Rasian luomiseksi oikeaan asentoon ja sijaintiin seinäelementin koordinaatistossa, sille on määritettävä sijainnit luontipisteiden suhteessa rasian objektin syvyyteen ja luontitasoon nähden:

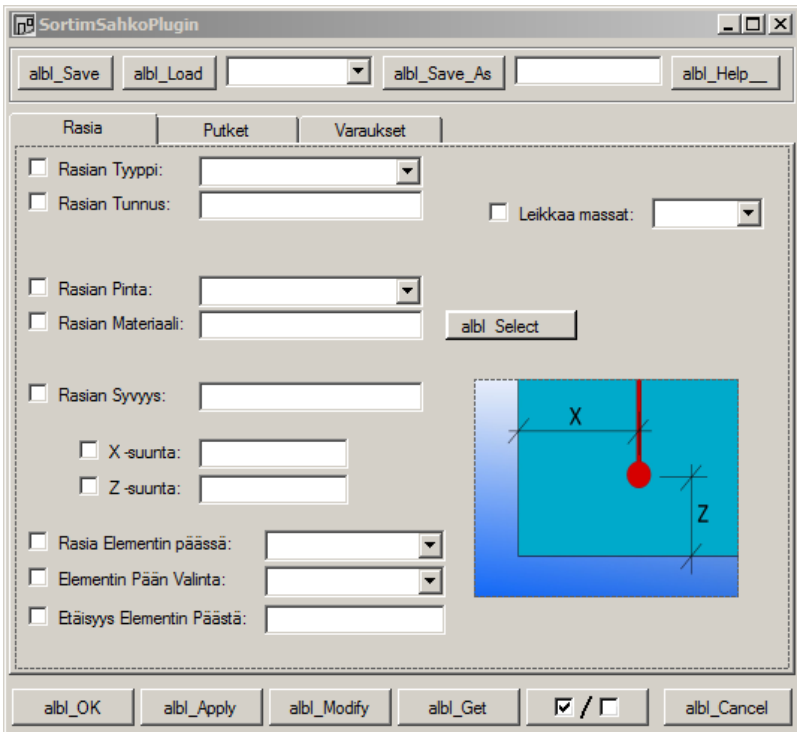
```
Rasia.Position.Depth = Position.DepthEnum.MIDDLE;
Rasia.Position.Plane = Position.PlaneEnum.MIDDLE;
```

Mikäli rasian luonnissa tapahtuu ristiriita pluginia suoritettaessa, plugin ilmoittaa käyttäjälle ”Failed to create electric box” –virheilmoituksen. Tämä on eräs tapa suorittaa ohjelmoinnin poikkeuskäsittely. Mikäli ristiriitaa ei tule, rasia luodaan normaalisti.

```
if(Rasia.Insert())
return Rasia;
else
{
            MessageBox.Show("Failed to create electric box");
            return null;
}
```

Käyttöliittymän Rasia -välilehdellä voidaan määrittää rasian tyyppi ja tunnus. Rasioiden tyyppi on määritetty vastaamaan yleisimmin käytettyjä rasiatyyppejä. Rasiatyypit ovat koje-, jako- ja kaksoiskojerasia. Rasian Pinta -valinnalla voidaan valita tuleeko rasia seinäelementin etu- vai takapintaan. Rasian pinta on sidottu rasian luokkaan joka määrittää rasian värin mallissa. Rasiat seinäelementin etupinnassa esitetään mallissa luokassa 2 eli ne näkyvät punaisina ja rasiat elementin taka- eli muottipinnassa luokassa 4 jolloin ne näkyvät sinisinä. Yhdessä rasioiden tyypin ja tunnuksen kanssa, rasioiden luokkaa voidaan käyttää yksilöimään rasioiden näkyvyys Teklan piirustustilassa.

Rasia -välilehdellä on myös muuttujat rasian sijainnin määrittämiseksi seinäelementin pituus- ja korkeusakselin suhteen. Rasia on myös mahdollista luoda seinäelementin päähän. Pään valinnalla pituus- eli x-akselin muuttuja vastaa elementin paksuuden suuntaista akselia.



Kuva 7 Pluginin käyttöliittymän Rasia -välilehti (Junkkarinen, 2014)

Koska pluginilla tuotetaan sähköasennusobjektit seinäelementti -objekteihin, on helpompi määrittää niiden mallintamiseen tarvittavat sijainnit seinäelementti -objektin omaan koordinaatistoon. Seinän oman koordinaatiston käyttö mahdollistaa myös sen, että sähköasennus -objektit on tuolloin sidottu seinäelementin koordinaatistoon ja täten myös käyttäytyvät seinää muokattaessa niin että ne pysyvät aina osana seinäelementtiä. Esimerkiksi jos seinäelementtiä kopioidaan tai käännetään se eri asentoon mallissa, seuraavat sähköasennus -objektit seinää sen muuttuneeseen sijaintiin tai kopioitaessa säilyttävät alkuperäisen sijaintinsa suhteessa kopioituun seinäelementtiin.

Seinäelementin koordinaatiston haku tapahtuu hakemalla ensin käyttäjän inputista saatava valittu seinäelementti -objekti ja sen jälkeen kutsumalla ja tallentamalla seinäelementti -objektin paikallinen koordinaatisto. Inputin ja koordinaatiston haku voidaan suorittaa ohjelmoinnillisesti seuraavasti:

```
Identifier ID1 = (Identifier)((InputDefinition)Input[0]).GetInput();
Part FatherPart = _model.SelectModelObject(ID1) as Part;

TransformationPlane CurrentTP =
_model.GetWorkPlaneHandler().GetCurrentTransformationPlane();

_model.GetWorkPlaneHandler().SetCurrentTransformationPlane(new Transfor-
mationPlane(FatherPart.GetCoordinateSystem()));
```

Koordinaatiston kutsumisen ja tallentamisen jälkeen määritetään vielä uudet muuttujat käytettäväksi luotavien objektien sijaintiattribuuttien yhdistämisessä seinäelementti -objektin koordinaatiston ääripisteisiin:

```
double MinX = FatherPart.GetSolid().MinimumPoint.X;
double MinY = FatherPart.GetSolid().MinimumPoint.Y;
double MinZ = FatherPart.GetSolid().MinimumPoint.Z;
double MaxX = FatherPart.GetSolid().MaximumPoint.X;
double MaxY = FatherPart.GetSolid().MaximumPoint.Y;
double MaxZ = FatherPart.GetSolid().MaximumPoint.Z;
```

Kun seinäelementti -objekti ja sen koordinaatisto on kutsuttu ja ne ovat käytettävissä ohjelmointitehtävässä, voidaan rasian luomiseksi määrittää sille sijaintipisteet seinän koordinaatistossa. Koska rasia luodaan Teklan palkki -objektina, täytyy sille määrittää alku- ja loppupiste, joiden väliin palkki luodaan. Rasia -objektin alku- ja loppupisteet määritetään määrittämällä uudet muuttujat RasiaStartPoint ja RasiaEndPoint ja määrittämällä niiden sijainnit seinäelementti -objektin paikallisessa koordinaatistossa.

```
Point RasiaStartPoint = new Point(MinX + _RasiaXdim, MinY + _RasiaYdim,
MaxZ);
Point RasiaEndPoint = new Point(MinX + _RasiaXdim, MinY + _RasiaYdim,
MaxZ - _RasiaSyvyys);
```

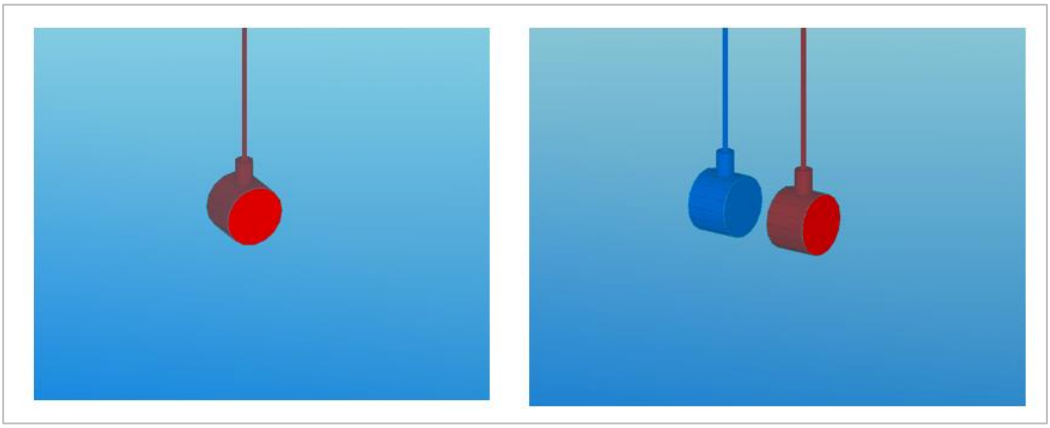
Käyttöliittymällä olevat attribuutit _RasiaXdim ja _RasiaYdim ovat käyttäjältä haettavat rasian X- ja Y-akselin suuntaiset sijainnit rasian luontipisteille ja ne on edellä olevassa tapauksessa yhdistetty seinän ääripisteisiin MinX ja MinY, jotka ovat seinäelementti -objektin minimi pisteet suhteessa X- ja Y-akseliin.

Kun rasian sijaintipisteet on määritetty ja sidottu käyttöliittymän attribuutteihin, voidaan rasia luoda. Rasian luonti tapahtuu kutsumalla objektia Beam eli palkki ja tässä tapauksessa nimeämällä se Rasiaksi. Tämän jälkeen kutsutaan aikaisemmin luotua rasian yksityistä metodia ja määritetään metodin käytettäväksi rasialle asetetut alku- ja loppupisteet. Tämä tapahtuu seuraavasti:

```
Beam Rasia = CreateRasia(RasiaStartPoint, RasiaEndPoint);
```

Kun rasia on luotu, se liitetään vielä osaksi seinäelementti -objektin kokoonpanoa (eng. assembly). Seinän kokoonpanoon liittämiseksi on määritettävä uusi muuttuja A, joka toimii seinäelementti -objektin kokoonpanon referenssinä. Muuttujan A määrittämisen jälkeen liitetään luotu rasia -objekti osaksi seinän kokoonpanoa A ja komennolla Modigy() määritetään seinäelementti -objektin kokoonpano suorittamaan tapahtuneet muutokset.

```
Assembly A = FatherPart.GetAssembly();
A.Add(Rasia.GetAssembly());
A.Modify();
```



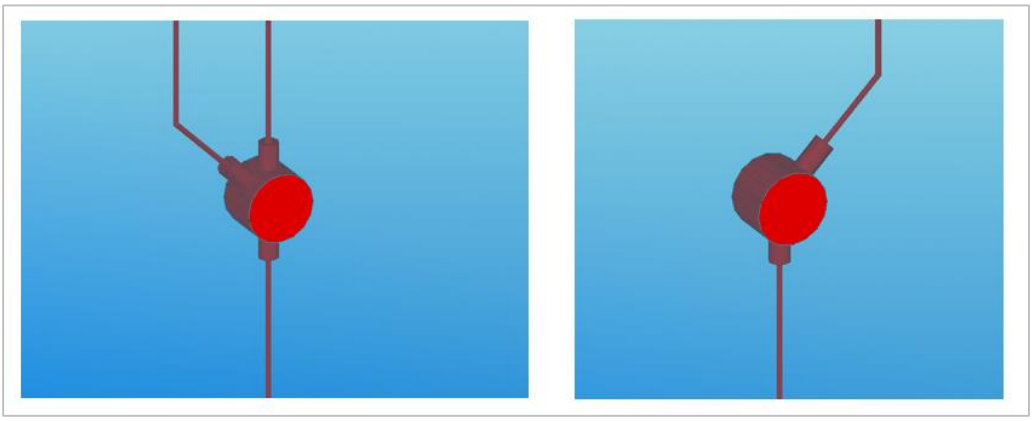
Kuva 8 Esimerkkejä pluginilla mallinnetuista rasioista (Junkkarinen, 2014)

Pluginilla mallinnettavat rasiat ovat varsin yksinkertaisia. Rasiat on kuitenkin tarkoituksen mukaista pitää geometrialtaan yksinkertaisina, jolloin ne eivät tee työstettävästä mallista liian raskasta. Kuvitellaan että työstettävässä mallissa on seinäelementtejä noin 400–500 kappaletta ja jokaiseen tulee keskimäärin kolmesta neljään rasiaa. Rasioiden määrä mallissa olisi tuolloin 1000:sta 2000:tta kappaletta. Jos rasiat olisivat geometrialtaan monimutkaisempia ja näin ollen vastaisivat todellisten rasioiden geometriaa, tekisivät ne työstettävästä mallista hyvin raskaan. Lisäksi tulevat putkitukset, pääteholkit ja muut mallinnettavat objektit nostavat sähköasennusten mallinnusobjektien määrän mallissa jo useisiin tuhansiin objekteihin. On siis varsin perusteltua, että kehitystyön pluginilla tuotettavat objektit ovat visuaalisesti yksinkertaisia. Tärkeämpää on saada riittävä tietosisältö yhdistettyä mallinnettaviin objekteihin ja vietyä se mallinnustilasta piirustuksissa käytettäväksi.

4.4.2 Putkitusten ohjelmointi

Sähköasennuksiin liittyvien putkitusten osalta kehitystyön pluginilla on voitava mallintaa suorat putkivedot rasialta elementin ylä- ja alalaitaan sekä taivutetut putkitukset, mikäli elementin rasiat sijaitsevat päällekkäin ja alapuoliselta rasialta tulevan putkituksen on kierrettävä sen yläpuolella sijaitseva rasia. Putkituksille on voitava määrittää niiden tunnus ja koko sekä putkitusten päähän tulevat pääteholkit tai -varaukset. Sähköistys -objektien piirustuksissa esittämisen helpottamiseksi on myös edunmukaista määrittää rasioille ja putkituksille muuttujat niiden sijaintipinnan mukaan, eli sijaitsevatko rasiat ja putket seinäelementin etu- vai takapinnassa.

Suoran putkituksen luonti tapahtuu ohjelmoinnillisesti vastaavalla tavalla, kuin rasian luonti. Putkitukset luodaan niin ikään Teklan palkkeina. Putkituksille ohjelmoidaan rasioiden yksityistä metodia vastaava oma metodi, jolla putkituksille haetaan niiden arvot dialogilta. Putkitukset määrittää siis putkituksiksi niiden tietosisältö, ei niinkään niiden visuaalinen muoto mallissa. Putkitusten yksityinen metodi on esitetty alla:



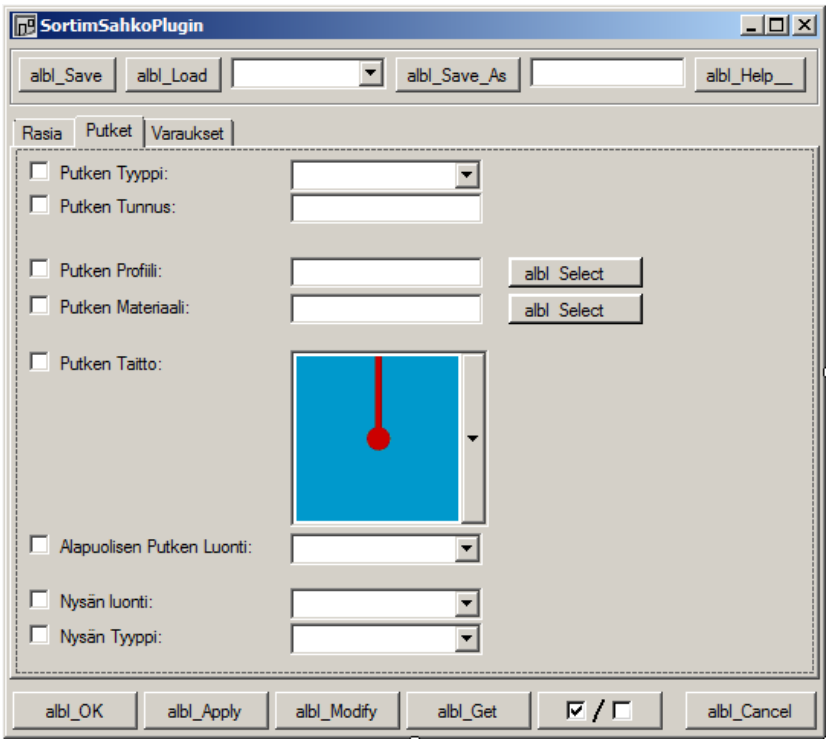
Kuva 9 Esimerkki mallinnetuista putkituksista (Junkkarinen, 2014)

```
private Beam CreateSuoraPutki(Point SuoraPutkiSP, Point SuoraPutkiEP)
{
            GetValuesFromDialog();

            Beam SuoraPutki = new Beam(SuoraPutkiSP, SuoraPutkiEP);
            SuoraPutki.Name = _PutkiName;
            SuoraPutki.PartNumber.Prefix = _PutkiTun;
            SuoraPutki.AssemblyNumber.Prefix = _PutkiTun;
            SuoraPutki.Profile.ProfileString = _PutkiProfile;
            SuoraPutki.Material.MaterialString = _PutkiMaterial;
            SuoraPutki.Class = _RasiaClass;
            SuoraPutki.Position.Depth = Position.DepthEnum.MIDDLE;
            SuoraPutki.Position.Plane = Position.PlaneEnum.MIDDLE;

            if(SuoraPutki.Insert())
            return SuoraPutki;
            else
            {
            MessageBox.Show("Failed to create straight top conduit");
            return null;
            }
}
```

Päämetodissa putkitukselle määritetään muuttuja putkituksen taittotapauksen mukaan, taittotapaus voidaan valita käyttöliittymältä sen mukaan, halutaanko mallintaa suora, oikealle tai vasemmalle taitettu tai näiden yhdistelmä putkitus. Yhdistelmiä käytetään, kun putkituksia on kaksi. Putkien taittotapaus valitaan käyttöliittymän ImageListComboBoxilta kohdasta Putken Taitto:



Kuva 10 Pluginin käyttöliittymän putket välilehti (Junkkarinen, 2014)

Käyttöliittymän Putket -välilehdellä putkitusten voidaan määrittää putkitusten tyypit ja tunnukset sekä putken profiili ja materiaali. Putkitusten profiili on oletusarvoisesti yhteydessä putkituksen tyypistä saatavaan profiiliin, mutta käyttöliittymässä on tehty mahdolliseksi putkitusten profiilin muokkaus käyttäjän toimesta. Välilehdellä voidaan myös valita putkitusten taittotapaukset sekä alapuolisen putken luonti. Putkitusten ja rasioiden väliin tulevien nysien luonnin ja tyyppien valinta voidaan myös määrittää tällä välilehdellä.

Putkituksien taiton ohjelmointi eroaa rasioiden ja suorien putkitusten ohjelmoinnista siten, että taivutetut putkitukset on ohjelmoitava Teklan polybeameinä rasioiden ja suorien putkitusten beamin sijaan. Polybeamien ohjelmointi eroaa beamin ohjelmoinnista siinä, että polybeamille on sen luontipisteille määritettävä niin sanottu tilapäinen holderi, johon polybeamin mallinnuspisteet tallennetaan ja josta ne kutsutaan käytettäväksi polybeamia luotaessa. Holderin määrittämiseksi kutsutaan ohjelmointitehtävässä ohjelmointi objektia Contour() ja määritetään sille mallinnuspisteet. Esimerkkitapauksen putkitustaitoksessa on määritetty holderille kolme pistettä:

```
Contour ContourPOikea = new Contour();

 Point PutkiOikeaSP = new Point(MinX + _RasiaXdim + 23, MinY + _RasiaZdim
 + 23, MaxZ - (_RasiaSyvyys / 2));

 Point PutkiOikeaMP = new Point(MinX + _RasiaXdim + 100, MinY +
 _RasiaZdim + 100, MaxZ - (_RasiaSyvyys / 2));

 Point PutkiOikeaEP = new Point(MinX + _RasiaXdim + 100, MaxY - _YPVarET,
 MaxZ - (_RasiaSyvyys / 2));

var contourPoint1 = new ContourPoint(PutkiOikeaSP, null);
var contourPoint2 = new ContourPoint(PutkiOikeaMP, null);
var contourPoint3 = new ContourPoint(PutkiOikeaEP, null);

ContourPOikea.AddContourPoint(contourPoint1);
ContourPOikea.AddContourPoint(contourPoint2);
ContourPOikea.AddContourPoint(contourPoint3);
```

Putkitus luodaan kutsumalla sille ohjelmoitua yksityistä metodia ja määrittämällä metodilla käytettäväksi edellä määritetty holderi, jolla mallinnuspisteet ovat. Lisäksi yhdistetään putkitus seinäelementti -objektin kokoonpanoon vastaavalla tavalla, kuin rasiaa ohjelmoitaessa.

```
PolyBeam PutkiOikea = CreatePutkiOikea(ContourPOikea);
Assembly A = FatherPart.GetAssembly();
A.Add(PutkiOikea.GetAssembly());
A.Modify();
```

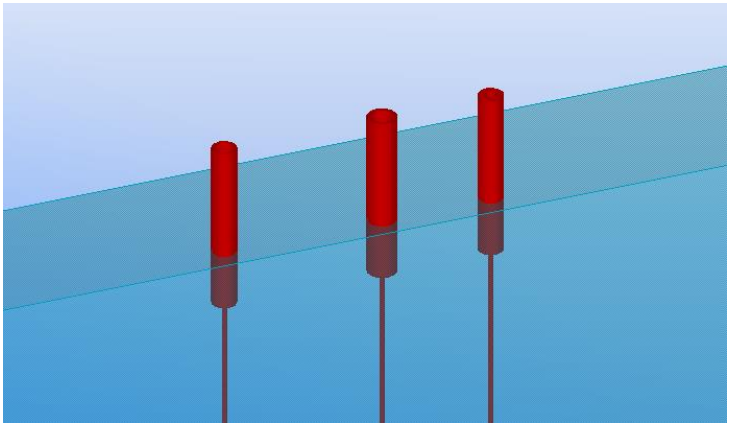
Kun putkitustapaukset on ohjelmoitu, voidaan ohjelmoida putkitusten yläpäähän tarvittavat pääteholkit. Pääteholkille on tehty oma yksityinen metodi, kuten rasioille ja putkituksille:

```
private Beam CreateYPHolkO(Point YPHolkOSP, Point YPHolkOEP)
            {
```

Pääteholkkien sijaintipisteet on päämetodissa määritetty riippuvaisiksi putkitustapauksista. Yläpään holkeille on määritetty mallinnuspisteet sen mukaan, mikä putkitustapaus on valittu käyttöliittymässä. Esimerkiksi putken taittotapauksessa jossa putkitus mallinnetaan taivutettuna oikealle, mallintuu yläpään holkki suhteessa putken yläpään sijaintiin. Taivutus tapauksessa, jossa putkituksia on kaksi, yläpään holkit mallintuvat useamman sijainnin mukaan. Esimekiksi kun mallinnetaan suora putkitus ja sen lisäksi rasiaan nähden vasemmalle taivutettu putkitus, mallintuvat holkit sekä suoran että taivutetun putkituksen yläpään sijainnin mukaan.

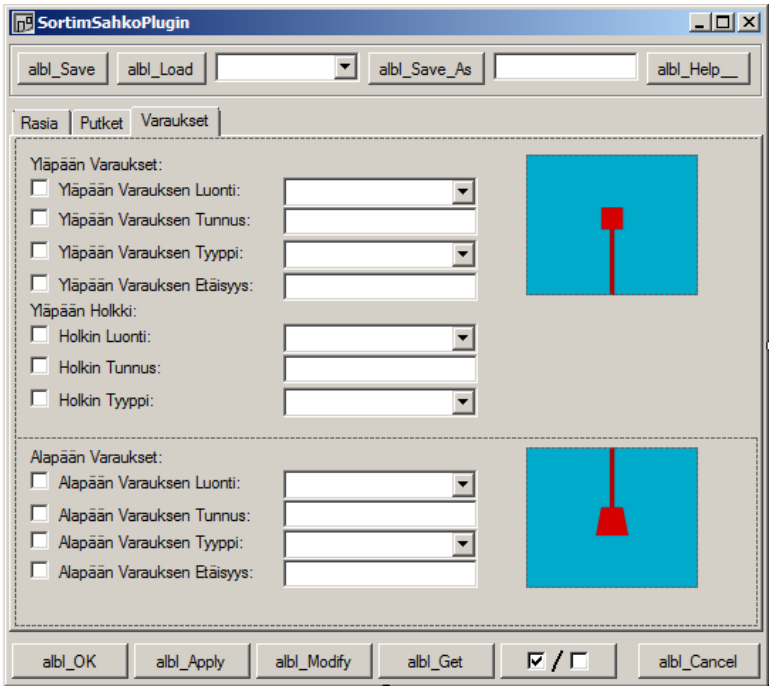
```
if (_YPHolkLuonti == 1)
{
            Point YPHolkSP = new Point(MinX + _RasiaXdim, MaxY - 50,
            MaxZ - (_RasiaSyvyys / 2));
            Point YPHolkEP = new Point(MinX + _RasiaXdim, MaxY + 100,
            MaxZ - (_RasiaSyvyys / 2));
            Point YPHolkVSP = new Point(MinX + _RasiaXdim - 100, MaxY -
            50, MaxZ - (_RasiaSyvyys / 2));
            Point YPHolkVEP = new Point(MinX + _RasiaXdim - 100, MaxY +
            100, MaxZ - (_RasiaSyvyys / 2));

            Beam YPHolk = CreateYPHolk(YPHolkSP, YPHolkEP);
            Beam YPHolkV = CreateYPHolkV(YPHolkVSP, YPHolkVEP);
            A.Add(YPHolk.GetAssembly());
            A.Add(YPHolkV.GetAssembly());
            A.Modify();
}
```



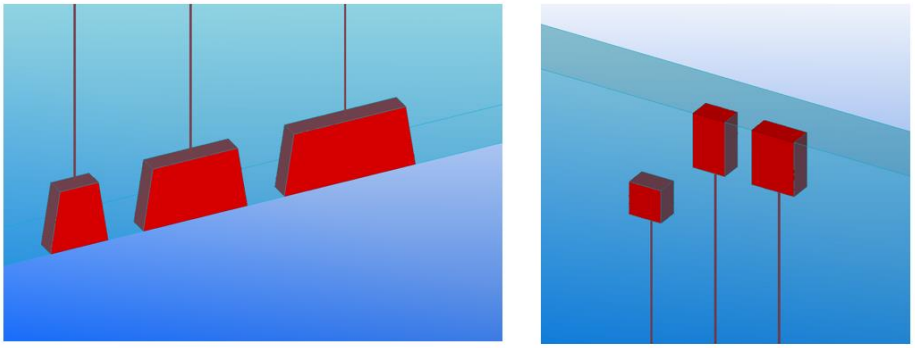
Kuva 11 Esimerkki putkien pääteholkeista (Junkkarinen, 2014)

Muuttujalla _YPHolkLuonti määritetään pääteholkkien luonti, eli halutaanko pääteholkit mallintaa vai jätetäänkö ne mallintamatta. Jos muuttuja saa arvon 1, pääteholkit mallintuvat. Jos ne taas saavat arvon 0, ne jätetään mallintamatta. Muuttujan arvo on sidottu päätehokkien mallinnukselle määritetylle valinnalle käyttöliittymän Varaukset välilehdelle kohtaan Holkin Luonti:



Kuva 12 Käyttöliittymän Varaukset -välilehti (Junkkarinen, 2014)

Käyttöliittymän Varaukset -välilehdellä voidaan myös määrittää tapauskohtaisesti ylä- ja alapään päätevarausten mallinnus. Päätevarauksille on määritetty muuttujat käyttöliittymään niiden mallintamiselle, tunnuksille, varausten tyypille sekä etäisyydelle elementin ylä- ja alalaidasta. Yläpään varaukset on tehty riippuvaisiksi putkitusten taittotapauksista, jolloin varaukset syntyvät putkitusten pään sijainnin mukaan ja mikäli putkituksia seinäelementin yläpäähän on kaksi, myös varauksia mallinnetaan kaksi. Varausten tyypit ovat vakiokokoisia ja ne on pluginin ohjelmistokoodissa määritetty vastaaviksi, kuin Betoniteollisuus ry:n betonielementtien sähköasennukset 2012 -ohjeessa. Lisäksi pluginilla voidaan mallintaa yleisesti venäjäkohteissa käytetty alapäänvaraustyyppi.



Kuva 13 Esimerkki mallinnetuista päätevarauksista (Junkkarinen, 2014)

5 TYÖKALUN TESTAUS

Ennen työkalun käyttöönottoa on tarkoituksen mukaista suorittaa erilaisia testitapauksia. Testitapauksissa plugin työkalua käytetään Tekla Structures -rakennesuunnitteluohjelmiston mallinnustilassa ja tehdään testejä liittyen pluginin käynnistymiseen, suorittamiseen ja muokkaamiseen. Testitapaukset sisältävät kaikki toimet, joita käyttäjä voi tehdä plugin työkalua käyttäessään. Testitapauksilla varmistetaan, että kaikki käyttäjän toimesta pluginilla tekemät muutokset ovat tarkoituksen mukaisia ja palvelevat käyttäjäänsä ilman virheitä.

Pluginin dialogin painikkeet		
Tehtävä	Tulos	Toteutuminen
Pluginin ikonin klikkaaminen	Plugin käynnistyy	Kyllä
Pluginin kuvan klikkaaminen component catalogissa	Plugin käynnistyy	Kyllä
OK -painikkeen klikkaus	Attribuutit tallentuvat ja dialogi sulkeutuu	Kyllä
Apply -painikkeen klikkaus	Attribuutit tallentuvat ja dialogi sulkeutuu	Kyllä
Modify -painikkeen klikkaus	Plugin suoritetaan uudelleen, muutokset tapahtuvat	Kyllä
Get -painikkeen klikkaus	Dialogin attribuutit haetaan valitulta pluginin objektilta	Kyllä
On/Off -painikkeen klikkaus	On/Off -valinnat päälle ja pois	Kyllä
Cancel -painikkeen klikkaus	Dialogi sulkeutuu, muutoksia ei suoriteta	Kyllä
Save -painikkeen klikkaus	Plugin tallentaa dialogin attribuutit	Kyllä
Load -painikkeen klikkaus	Plugin lataa dialogin attribuutit	Kyllä
Save As -painikkeen klikkaus	Plugin tallentaa dialogin attribuutit uudella nimellä	Kyllä
Rasian materiaali katalogin -painikkeen klikkaus	Materiaali katalogi avautuu, valittu materiaali tallentuu	Kyllä
Putken profiili katalogin -painikkeen klikkaus	Profiili katalogi avautuu, valittu materiaali tallentuu	Kyllä
Putken materiaali katalogin -painikkeen klikkaus	Materiaali katalogi avautuu, valittu materiaali tallentuu	Kyllä

Pluginin input ja suorittaminen		
Tehtävä	Tulos	Toteutuminen
Käyttäjän input	Plugin pyytää käyttäjää valitsemaan inputin	Kyllä
Plugin suoritetaan	Seinä valitaan, plugin suoritetaan, objektit luodaan	Kyllä
Objektien attribuuttien oikeellisuus	Pluginilla luoduilla mallinnusobjekteilla on halutut attribuutit	Kyllä

Käyttöliittymän Rasia -välilehti		
Tehtävä	Tulos	Toteutuminen
Rasian tyypin valinta	Rasian tyyppi muuttuu	Kyllä
Rasian tunnuksen valinta	Rasian tunnus muuttuu	Kyllä
Rasian pinnan valinta	Rasian pinta muuttuu	Kyllä
Rasian materiaalin valinta	Rasian materiaali muuttuu	Kyllä
Rasian syvyys	Rasian syvyyden arvo muuttuu	Kyllä
Rasian X-suunta	Rasian etäisyys muuttuu	Kyllä
Rasian Z-suunta	Rasian etäisyys muuttuu	Kyllä
Rasia elementin päässä	Rasian luonti elementin päähän	Kyllä
Elementin pään valinta	Elementin päähän luodun rasian sijainti vaihtuu	Kyllä
Etäisyys elementin päästä	Rasian sijainti muuttuu	Kyllä

Käyttöliittymän Putket -välilehti		
Tehtävä	Tulos	Toteutuminen
Putken tyypin valinta	Putken tyyppi muuttuu	Kyllä
Putken tunnuksen valinta	Putken tunnus muuttuu	Kyllä
Putken profiilin valinta	Putken profiili muuttuu	Kyllä
Putken materiaalin valinta	Putken materiaali muuttuu	Kyllä
Putken taitto	Putken taitto muuttuu	Kyllä
Alapuolisen putken luonti	Alapuolinen putki luodaan	Kyllä
Nysän luonti	Rasianysä luodaan	Kyllä
Nysän tyypin valinta	Rasianysän tyyppi muuttuu	Kyllä

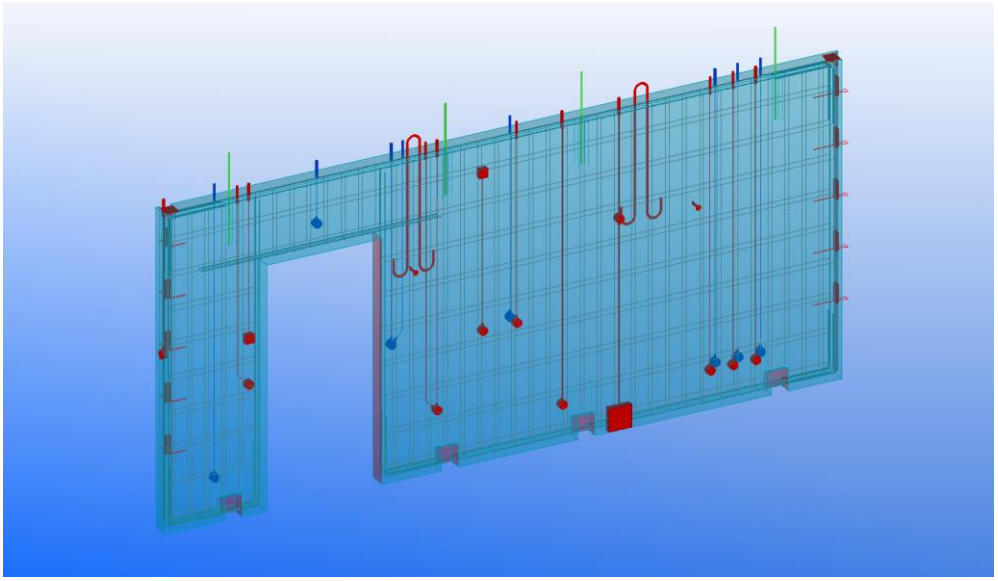
Käyttöliittymän Varaukset -välilehti		
Tehtävä	Tulos	Toteutuminen
Yläpään varauksen luonti	Yläpään varaus luodaan	Kyllä
Yläpään varauksen tunnuksen valinta	Yläpään varauksen tunnus muuttuu	Kyllä
Yläpään varauksen tyypin valinta	Yläpään varauksen tyyppi muuttuu	Kyllä
Yläpään varauksen etäisyys	Yläpään varauksen etäisyys muuttuu	Kyllä
Holkin luonti	Holkki luodaan	Kyllä
Holkin tunnuksen valinta	Holkin tunnus muuttuu	Kyllä
Holkin tyypin valinta	Holkin tyyppi muuttuu	Kyllä
Alapään varauksen luonti	Alapään varaus luodaan	Kyllä
Alapään varauksen tunnus	Alapään varauksen tunnus muuttuu	Kyllä
Alapään varauksen tyypin valinta	Alapään varauksen tyyppi muuttuu	Kyllä
Alapään varauksen etäisyys	Alapään varauksen etäisyys muuttuu	Kyllä

Muut testit		
Tehtävä	Tulos	Toteutuminen
Pluginilla luodun objektin kopiointi toiseen elementtiin	Objektit kopioituvat, objektit liittyvät uuden isäntäobjektin kokoonpanoon	Kyllä
Seinäelementin sijainnin muuttuminen	Sähköistysobjektit muuttavat sijaintiaan seinän mukana	Kyllä
Seinäelementin kääntäminen	Sähköistysobjektit kääntyvät seinän mukana	Kyllä
Seinäelementin peilaus	Sähköistysobjektit peilautuvat seinän mukana	Kyllä

6 TULOKSET

Kehitystyön tuloksena syntyi Tekla Structures -rakennesuunnitteluohjelmistossa toimiva plugin liitännäinen, jolla voidaan mallintaa yleisimmät seinäelementtien sähköasennuksissa esiintyvät mallinnusobjektit. Työkalu on varsin helppokäyttöinen ja sillä voidaan kohtuullisen nopeasti mallintaa seinäelementtien sähköasennukset sisältäen erilaiset koje-, jako- ja kaksoiskojerasiat, erilaiset putkitustapaukset ja niiden taitokset, rasioiden ja putkitusten väliin tulevat nysät sekä putkitusten päihin tulevat päätevaraukset ja -holkit.

Ajallisesti suurin etu saavutetaan sillä, että kehitystyön tuloksena syntyneellä pluginilla sähköistysobjektit tarvitsee ainoastaan mallintaa Teklan mallinnustilassa. Sähköistysobjekteihin sisällytetyn informaation avulla tapahtuva piirustustilan objektien yksilöiminen mahdollistaa sen, että piirustustilassa sähköistysobjekteista saatava tieto voidaan esittää piirustuksissa automaattisesti. Sähköistysobjektien luettelointi seinäelementtien piirustusten taulukoissa tapahtuu niin ikään automaattisesti. Muottikuvassa eli toisin sanoen elementtipiirustuksen pääkuvassa sähköistysobjektit esitetään automaattisesti niiden visuaalisen ilmeen sekä objektien tunnusten ja mitoituksen osalta. Toisin sanoen, Teklan piirustustilassa sähköistysobjektien oikeaoppinen esittäminen ei vaadi käyttäjältä erillisiä toimenpiteitä.



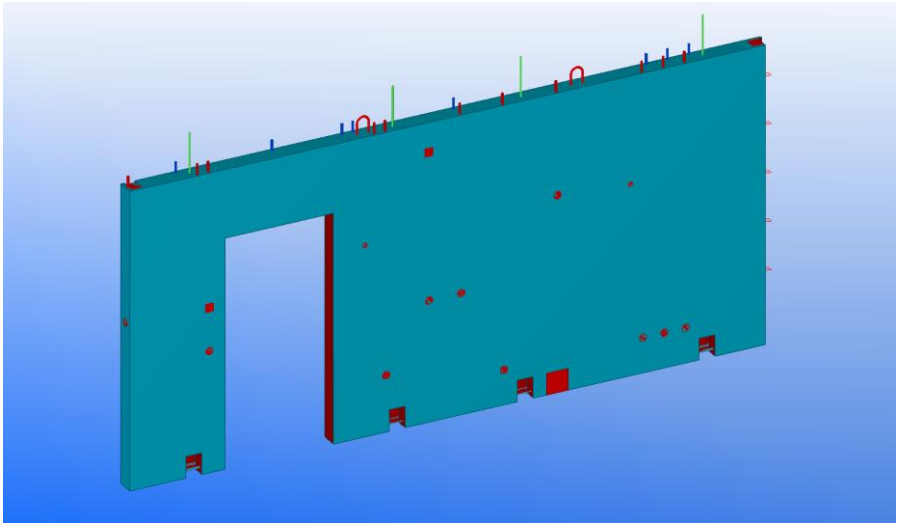
Kuva 14 Esimerkki pluginilla sähköistetystä väliseinäelementistä (Junkkarinen, 2014)

7 TYÖKALUN KÄYTÖN ALOITUS

Työkalun käytön aloitus käsittää työkaluun liittyvien asetusten määrittämisen Teklan mallinnustilassa sekä tarvittavien taulukoiden ja automatisoitujen toimintojen kehittämisen Teklan piirustustilaan. Mallinnustilassa tehtävät toimenpiteet käsittävät pluginin objekteille määritettävät suodatinasetukset, joita tarvitaan piirustustilassa objetkien esittämiseksi piirustuksissa ja taulukoissa. Piirustustilaan kehitetään sähköasennustarvikkeiden esittämiseksi tarvittavat taulukot ja mallinnustilassa määritettyjä suodattimia käyttäen tehdään tarvittavat asetukset sähköasennusten esittämiseksi seinäelementtien muottikuvissa ja leikkauksissa. Lisäksi luodaan Teklan työkalupalkkiin kuvake, josta työkalu voidaan käynnistää. Työkalua on myös mahdollista käyttää Teklan component catalogin kautta. Kuvassa 15 on esitetty työkalun kuvakkeet Teklan työkalupalkissa sekä component catalogissa.

Kuva 15 Työkalun kuvakkeet (Junkkarinen, 2014)

Tarvittavien asetusten määrittämistä tukemaan on mallinnettu esimerkki seinäelementti. Esimerkki elementtiin on mallinnettu sähköasennukset opinnäytetyössä kehitetyllä työkalulla. Elementtiä käytetään apuna tarvittavia näkymä suodattimia määritettäessä. Siitä tuotetaan myös esimerkki piirustus, johon tehdään tarvittavat taulukoinnit ja rasioiden graafista ilmettä säätelevät asetukset. Esimerkki elementtiin on mallinnettu kaikki sähköasennustapaukset, jotka työkalulla voidaan tuottaa. Esimerkki seinäelementti on esitetty kuvassa 16.

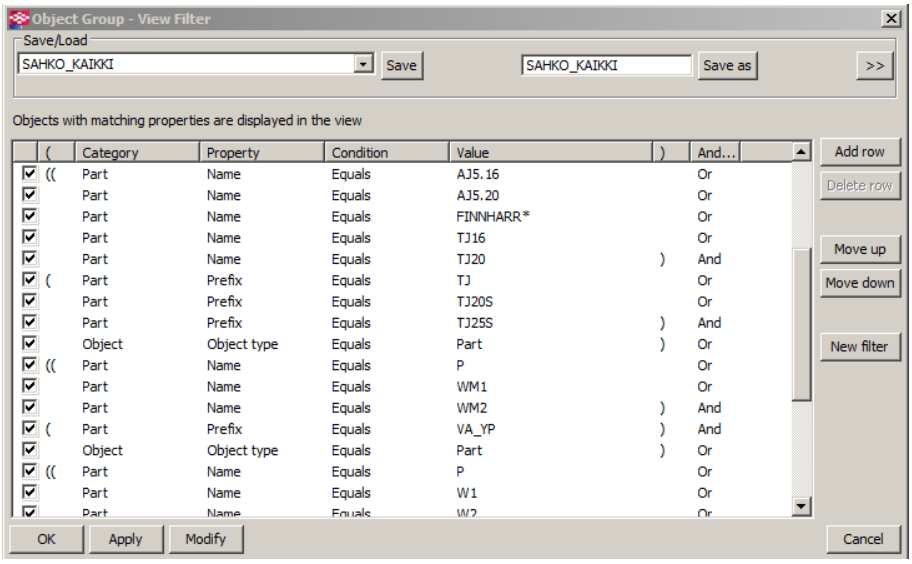


Kuva 16 Esimerkki seinäelementti (Junkkarinen, 2014)

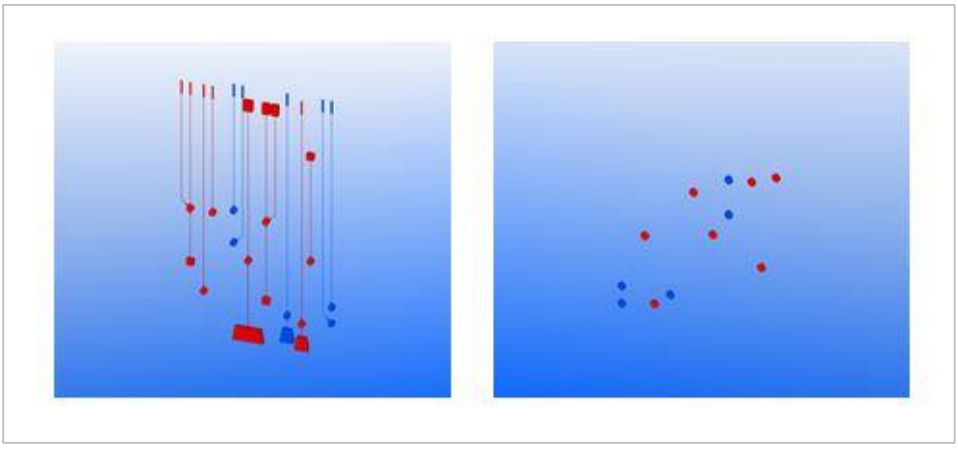
7.1 Mallinnustila

Teklan mallinnustilassa voidaan määrittää suodattimia, joilla voidaan suodattaa mallinnusobjekteja joko näkyviksi tai näkymättömiksi käyttäen näkymä suodattimia (view filter) tai objektien valitsemisen rajoittamiseksi valinta suodattimilla (selection filter). Näkymä suodattimilla objekteja voidaan piilottaa mallinnustilassa. Niitä käytetään, kun halutaan rajoittaa mallin objektien näkyvyyttä tai helpottaa mallintamista piilottamalla haluttujen objektien edessä olevia objekteja. Valinta suodattimilla voidaan vaikuttaa siihen, mitkä objektit ovat valittavissa ja mitkä eivät. Näiden käyttö helpottaa mallintamista esimerkiksi silloin, kun halutaan valita mallista yhtä aikaisesti useita samanlaisia objekteja ilman että ei halutut objektit tulevat valituksi. Näkymä suodattimet määritetään kaikille sähköasennusten mallinnus -työkalulla tuotettaville mallinnusobjekteille. Kun tarvittavat suodattimet on tehty, voidaan niitä hyödyntää piirustusten tuottamisessa.

Kuvissa 17 ja 18 on esitetty näkymä suodattimien määrittämiseksi tarkoitettu Teklan työkalu sekä havainnollistettu näkymä suodattimen toimintaa suodattamalla sähköasennuksista näkyviin ainoastaan rasiat.



Kuva 17 Teklan näkymä suodatin työkalu (Junkkarinen, 2014)



Kuva 18 Näkymä suodattimen toiminnan havainnollistaminen (Junkkarinen, 2014)

7.2 Templatet ja taulukot

Templatet ovat Teklan raporttien, taulukoiden ja graafisten tietokenttien esittämiseen tarkoitettuja taulukkopohjia. Templatejen avulla voidaan kehittää esimerkiksi älykkäitä raudoite- ja valutarvikeluetteloita, jotka tunnistavat mallinnustilan objektit automaattisesti sekä hakevat niihin sisällytettyä tietoa. Niitä voidaan käyttää myös raporttien tuottamiseen Teklasta. Templateja voisi verrata esimerkiksi Microsoft Excel -taulukkoihin. Templatet koostuvat Excel -taulukoiden tapaan niiden sisältämistä erilaisista kaavoista ja kaavojen välisistä yhteyksistä, joilla taulukoista tehdään älykkäitä.

Opinnäytetyössä kehitetyn mallinnustilan työkalun lisäksi on kehitettävä tarvittavat templatet sähköasennusobjektien esittämiseksi piirustusten taulukoissa. Templateissa jokaiselle sähköasennuksissa esiintyvälle objektityypille, esimerkiksi rasioille on tehty omat template -pohjan rivit. Rivit hakevat tietoa piirustus objekteilta ja vertaavat niitä keskenään. Samanlaiset objektit esitetään templatessa samalla rivillä. Riveillä on tehty objekteille niiden arvoja hakevia kenttiä (value field). Value fieldeillä määritetään, mitä tietoa objektilta halutaan hakea. Esimerkiksi kojerasioille on määritetty templatessa rivi, joka määrittää sille haettavien objektien tyypiksi kojerasiat. Sen jälkeen rivillä olevat value fieldit hakevat rasian nimikkeen, tunnuksen ja valmistajan sekä kappalemäärän.

Kuvassa 19 on esitetty kuvakaappaus sähköasennuksien taulukointia varten tehdystä templatesta. Template hakee sähköobjektien nimet ja valmistajan, tunnukset, mitat ja kappalemäärät sekä putkitusten tapauksessa putkien pituudet.

ÄÑÅ ÓÐÓÀÙ ÄÈß ÝÊÄÈÓÐÌÎÐÌÀÌÄÈÈ JM 20 ÄÑÈÈ ÍÅ ÇÀÍÀÐÈÈÐÌÄÀÍÙ
ÍÅ ÊÎÍÓÙ ÓÐÓÀ ÓÑÓÀÍÄÄÈÈÄÄÞÒÑß ÃÈÄÊÈÈ ÓÄÈÈÌÈÓÄÈÈ TJ20S/TJ25S

NAME_	TITLE	DIMENSIONS_	PCS	TOTAL_
field_NAME_2	field_TJ_1	DIMENSIONS_	fie	
ValueField_2	field_K_Rasia_1	ValueField_5	Val	
ValueField_7	field_J_Rasia_1	ValueField_10	Val	
ValueField_12	field_S_Rasia_1	ValueField_15	Val	
ValueField_17	field_Putket_1		Val	Valu m
ValueField_16	field_Putket_2		Val	Valu m
ValueField_75	field_Putket_3		Val	Valu m
ValueField_11	ValueField_6		Val	Valu m
ValueField_22	field_Varaukset_P	ValueField_25	Val	
ValueField_42	field_Varaukset_WM1	ValueField_45	Val	
ValueField_48	field_Varaukset_WM2	ValueField_51	Val	
ValueField_54	field_Varaukset_W1	ValueField_57	Val	
ValueField_60	field_Varaukset_W2	ValueField_63	Val	
ValueField_64	field_Varaukset_W3	ValueField_67	Val	
ValueField_9	field_Varaukset_H	ValueField_19	Val	
ValueField_58	field_Nysat_1	ValueField_70	Val	

Kuva 19 Sähköasennusten taulukointia varten kehitetty template (Junkkarinen, 2014)

Kuvassa 20 on esitetty, miltä templatella tuotettu sähköasennusobjektien taulukko näyttää Teklan piirustustilassa.

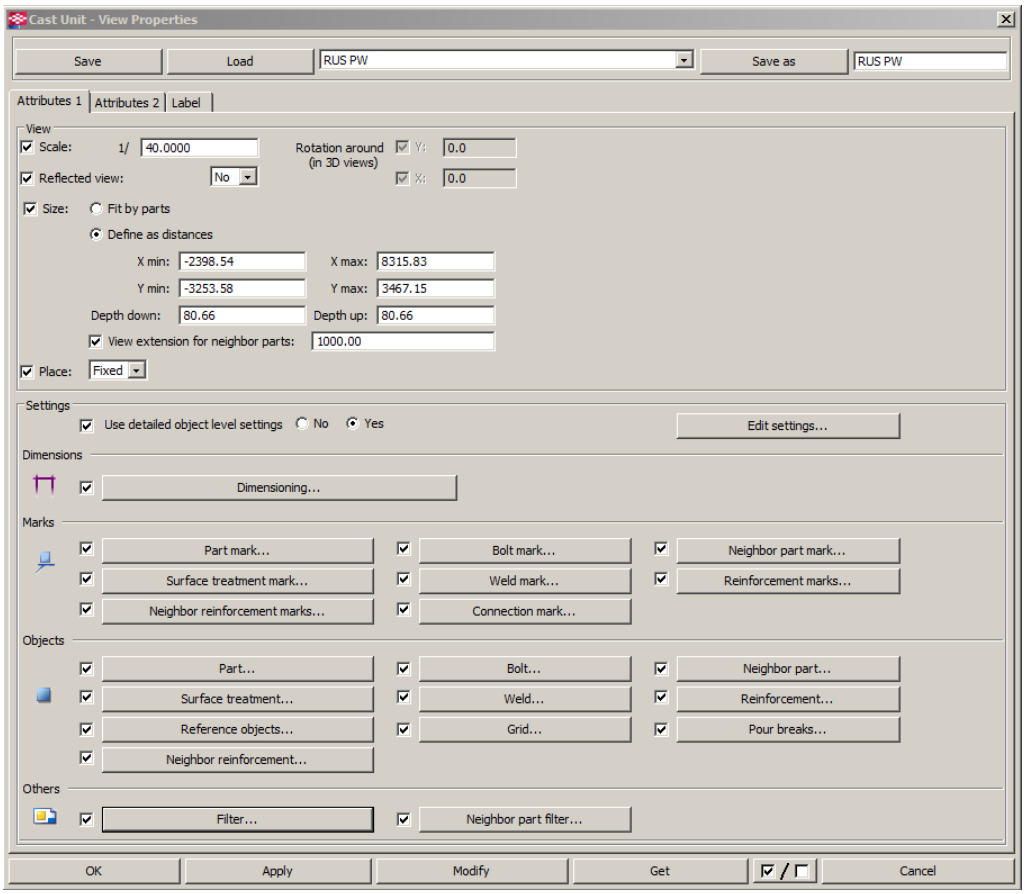
ВСЕ ТРУБЫ ДЛЯ ЭЛЕКТРОПРОВОДКИ JM 20 ЕСЛИ НЕ ЗАМАРКИРОВАНЫ
НА КОНЦЫ ТРУБ УСТАНАВЛИВАЮТСЯ ГИБКИЕ УДЛИНИТЕЛИ TJ20S/TJ25S

Обозначение	Название	Размеры	Кол.	Всего
ГИБКИЙ УДЛИНИТЕЛЬ	FINNHARR TJ20S		17	
FINNHARR TJ25S	FINNHARR TJ25S		2	
МОНТАЖНАЯ КОРОБКА	ABB AU3.2 + AS32		16	
СОЕДИНИТЕЛЬНАЯ КОРОБКА	ABB AU19 + PM107		1	
ЭЛЕКТРИЧЕСКАЯ ТРУБА	PIPELIFE JM20		18	33.76 m
ЭЛЕКТРИЧЕСКАЯ ТРУБА	PIPELIFE JM25		3	5.13 m
Ниша	H	200x200x50	1	
AN20	ABB AN20		16	
AN25	ABB AN25		3	

Kuva 20 Esimerkki template piirustuksessa (Junkkarinen, 2014)

7.3 Piirustukset

Ennen mallinnustilassa määritettyjen suodattimien hyödyntämistä sähköasennusten esittämisessä Teklan piirustuksissa, on sähköasennusobjekteille tehtävä asetukset jotka määrittävät niiden näkyvyyden piirustuksissa. Esimerkiksi rasioiden ja putkitusten väriin ja rasterointiin voidaan vaikuttaa näillä asetuksilla. Asetuksia voidaan muuttaa Teklassa elementtipiirustuksia varten olevalla Cast Unit -dialogilla

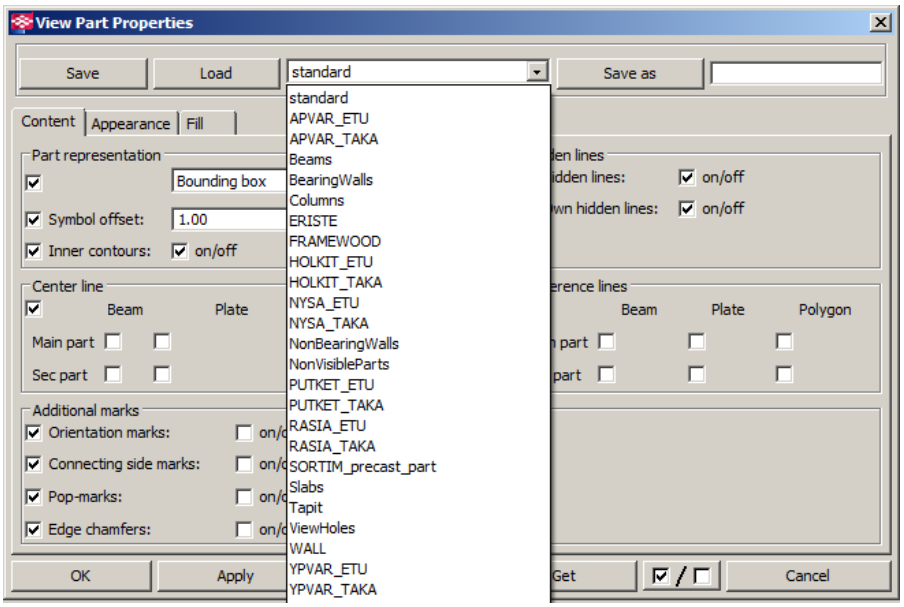


Kuva 21 Cast unit -dialogi (Junkkarinen, 2014)

Rasioille ja putkituksille tehdään asetukset niiden esittämiseksi elementin etu- ja takapinnassa. Etupinnassa oleville rasioille ja putkituksille tehdään asetukset, joilla ne esitetään niiden ääriviivojen mukaan ja väriltään valkoisina. Takapinnassa oleville rasioille ja putkituksille tehdään asetukset, joilla ne esittää mustaksi rasteroituina. Varauksille ja rasianysille tehdään asetukset, joilla ne voidaan esittää kuten rasiat ja putkitukset. Putkitusten yläpäähän tuleville pääteholkeille tehdään asetukset, joilla ne näkyvät sijaintipinnasta riippumatta pelkkinä ääriviivoina.

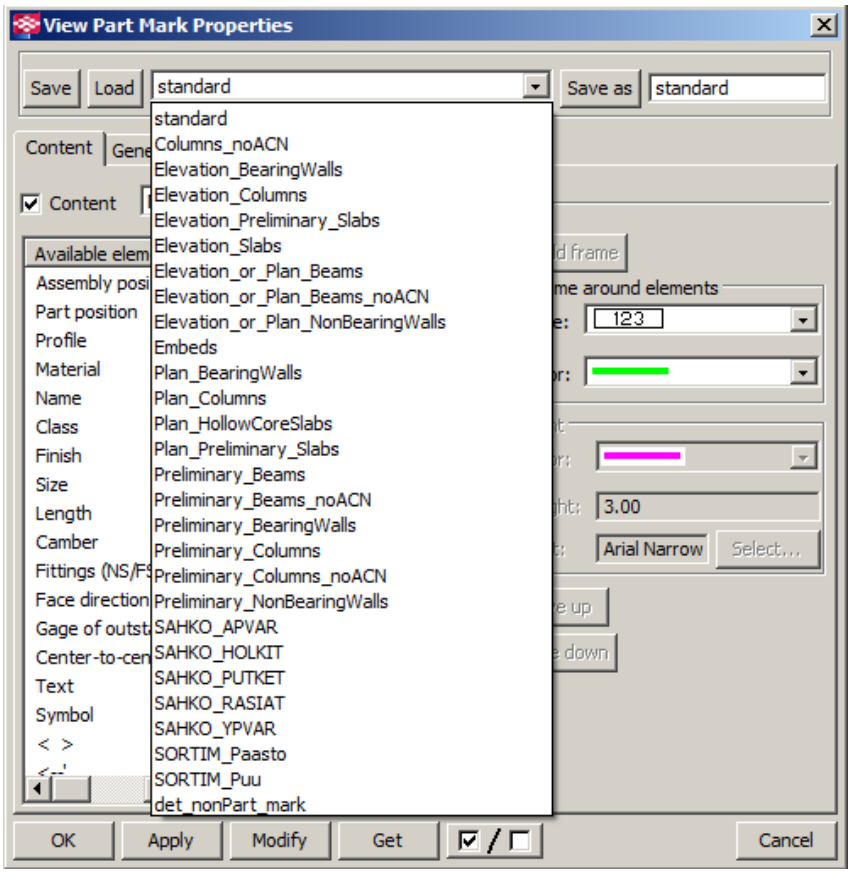
Piirustusobjektien asetuksia voidaan muokata ja määrittää Cast Unit -dialogilla. Piirustusobjektien graafiseen esittämiseen liittyviä asetuksia voidaan tehdä Cast Unit -dialogin Part -painikkeesta avautuvalla View Part Properties -dialogilla. Piirustusobjektien tunnuksia voidaan tehdä Part mark -painikkeesta avautuvalla View Part Mark Properties -dialogilla.

Kuvassa 24 on esitetty View Part Properties -dialogilla sähköasennusobjekteille määritetyt asetukset. Asetukset näkyvät dialogin alasvetovalikossa.



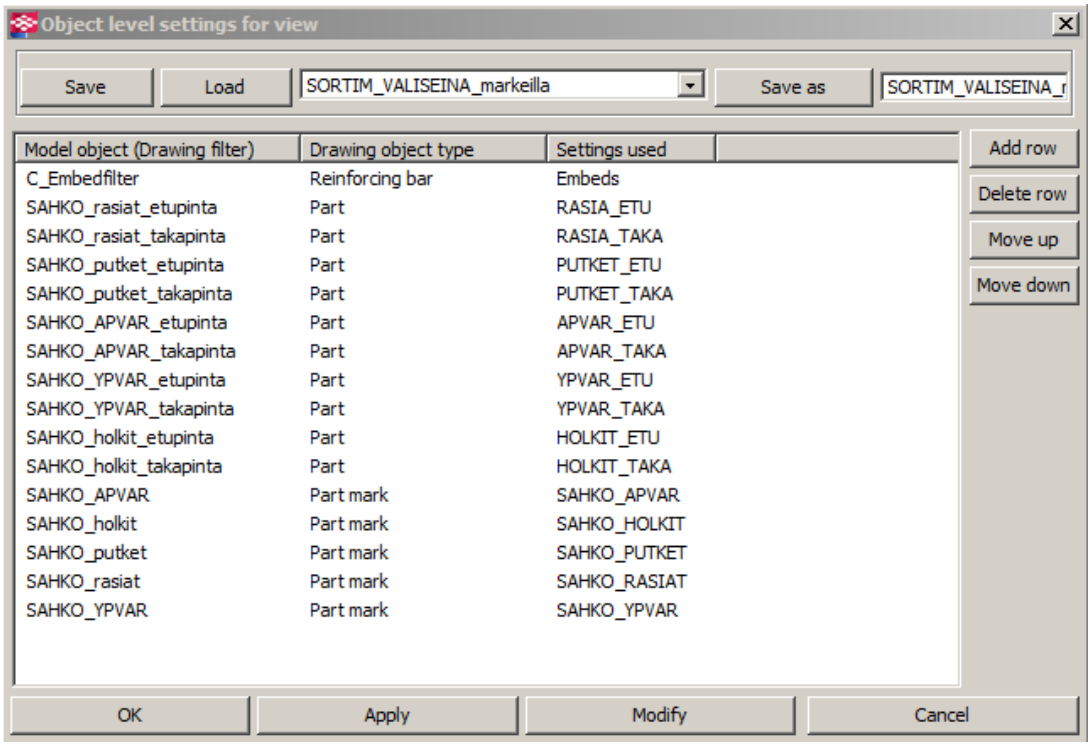
Kuva 22 (Junkkarinen, 2014)

Kuvassa 25 on esitetty View Part Mark Properties -dialogilla sähköasennusobjekteille tehdyt asetukset. Asetukset näkyvät dialogin alasvetovalikossa.



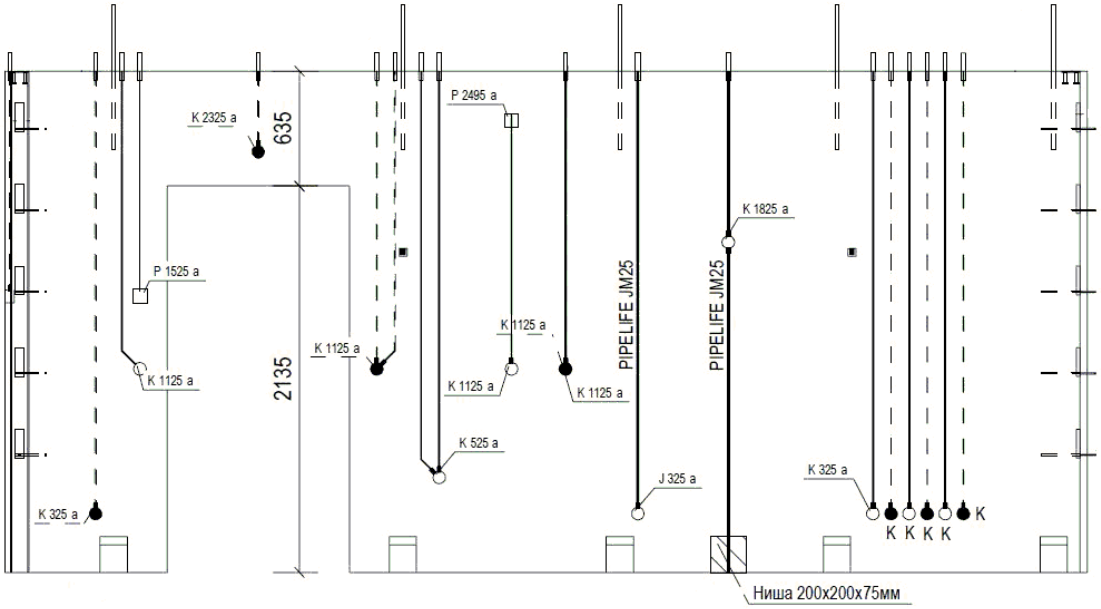
Kuva 23 (Junkkarinen, 2014)

Kun tarvittavat asetukset objektien grafiikan esittämiseksi on tehty, voidaan ne yhdistää aiemmin määritettyihin näkymä suodattimiin Cast Unit -dialogin Object level -asetuksista. Object level -dialogilla voidaan yhdistellä objekteille määritettyjä näkymäsuodattimia Part- ja Part Mark -asetusten kanssa. Object level -asetukset määrittävät objektien näkyvyyden niille asetettujen erillisten tasojen mukaan. Näkymä suodattimella määritetään tietynlaisille objekteille taso jolla ne näkyvät ja Part- ja Part Mark -asetuksilla määritellään kyseisellä tasolla olevien objektien ulkomuoto ja niille annettavat tunnukset. Näkymä suodattimien yhdistäminen Part- ja Part Mark -asetuksiin on esitetty kuvassa 26.



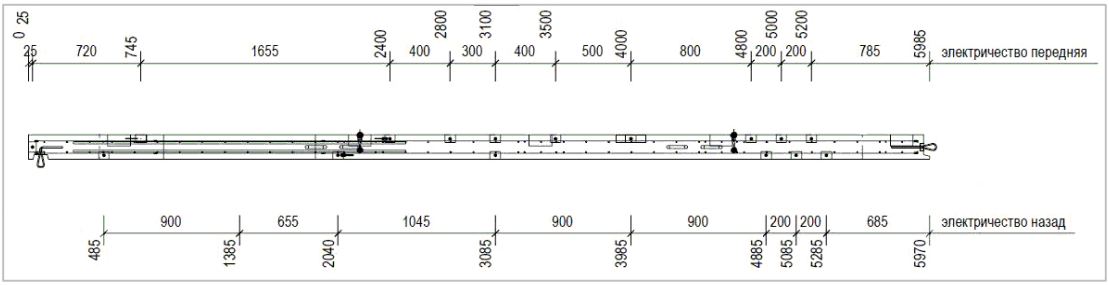
Kuva 24 Näkymä suodattimien ja objektien asetusten yhdistäminen (Junkkarinen, 2014)

Näkymä suodattimien ja piirustus objekteille tehtyjen asetusten yhdistämisen jälkeen voidaan sähköasennusobjektit esittää seinäelementtin muottikuvassa.



Kuva 25 Plugin työkalulla sähköistetyn elementin muottikuva (Junkkarinen, 2014)

Näkymä suodattimia on käytetty myös rasioiden vaakamitoituksen esittämisessä seinäelementin vaakaleikkauksessa.



Kuva 26 Sähköasennusten mitoitus vaakaleikkauksessa (Junkkarinen, 2014)

8 POHDINTA

Opinnäytetyön tavoitteena oli kehittää Tekla Structures -rakennesuunnitteluohjelmistossa toimiva betonielementtien sähköasennusten -mallinnustyökalu. Tuloksena kehittyi Tekla Structures -rakennesuunnitteluohjelmistossa toimiva plugin sovellusliitännäinen, jolla voidaan mallintaa sähköasennusten mallinnusobjektit. Opinnäytetyön kehittämisprosessi osoittautui varsin aikaa vieväksi ja työlääksi, mutta myös tekijäänsä kehittäväksi ja työn tuloksellisuuden myötä palkitsevaksi. Kehitetyllä työkalulla voidaan mallintaa seinäelementtien sähköasennukset niissä yleisimmin esiintyvien tapausten osalta. Kehitetty työkalu on myös helposti muokattavissa vastaamaan kehitysprosessin ulkopuolelle jääneitä sekä tulevia sähköasennustapauksia.

Kehitystyössä syntynyt työkalu mahdollistaa seinäelementtien sähköasennusten mallintamisen ja niiden esittämisen elementtipiirustuksissa huomattavasti aikaisempaa menetelmää helpommin ja nopeammin. Aikaisempi työmenetelmä, jossa elementti suunnittelija lähettää mallinnettujen elementtien piirustukset sähkösuunnittelijalle ja liittää sähkösuunnittelijalta palautuneet piirustukset takaisin alkuperäisiin elementtipiirustuksiin oli varsin työläs ja aikaa vievä menetelmä. Aikaa kului huomattavasti siihen, että elementtisuunnittelijan oli työstettävä samoja elementtipiirustuksia useampaan kertaan. Lisäksi määrittelemätön aika kului siihen, kun elementtipiirustukset olivat sähkösuunnittelijan työstettävinä.

Opinnäytetyön tuloksena syntynyt työkalu mahdollistaa seinäelementtien sähköistysten mallintamisen yhtäaikasesti seinäelementtien muiden mallinnusobjektien ohella. Lisäksi piirustustilassa vaadittava elementtipiirustusten työstä on huomattavasti vähäisempää, kuin aikaisemmassa menetelmässä. Mallinnetut sähköasennusobjektit esitetään Teklan piirustustilassa automaattisesti sekä elementtikuvissa, että piirustusten taulukoissa. Käyttäjältä vaaditaan kehitettyä työkalua käytettäessä sähköasennusten mallintaminen Teklan mallinnustilassa sekä objektien tarkistus Teklan piirustustilassa. Työkalun käytön yhteydessä työkalua ja siihen liittyviä muita Teklan ominaisuuksia voidaan kehittää vastaamaan entistä paremmin työkalun käyttäjän vaatimuksia.

Opinnäytetyön kehitystyön tuloksena syntyi siis huomattavan nopeakäyttöinen ja kustannustehokas seinäelementtien sähköasennusten mallinnustyökalu Tekla Structures -rakennesuunnitteluohjelmistossa käytettäväksi.

LÄHTEET

Betonielementtien sähköasennukset 2012 [ohje]. Viitattu [2014-10-04.]
Saatavissa: http://www.betoni.com/Download/22708/BET1104_78-81.pdf

Betonielementtien sähköasennukset 2012. Betoniteollisuus ry. [kuvat]. [Viitattu 2014-10-04.]

OJALA, Iiro, 2012. Tekla North American User Meeting 2012. [Viitattu 2014-09-28.]
Saatavissa: https://extranet.tekla.com/BC/tekla-structures-en/developers/openapi/Pages/Default.aspx

Sormunen & Timonen, 2014 [kuvat]. [Viitattu 2014-10-05.]

SIVONEN, Veli-Matti, Helsingin yliopisto, 2004 [seminaariesitelmä]. [Viitattu 2014-09-28.] Saatavissa: http://www.cs.helsinki.fi/u/pohjalai/k04/ohpe/seminar/Sivonen-CSharp.pdf

TEKLA 2014a. Tekla Structures tuote-esittely [verkkojulkaisu]. [Viitattu 2014-09-28.]
Saatavissa: http://www.tekla.com/fi/tuotteet/tekla-structures

TEKLA 2014b. Tekla Structures tuote-esittely [verkkojulkaisu]. [Viitattu 2014-09-28.]
Saatavissa: http://teklastructures.support.tekla.com/190/en/sys_tekla_open_api

Tekla Corporation 2013. Developers Guide. Product Version 19.1.

LIITE 1: PIIRUSTUSMERKINNÄT

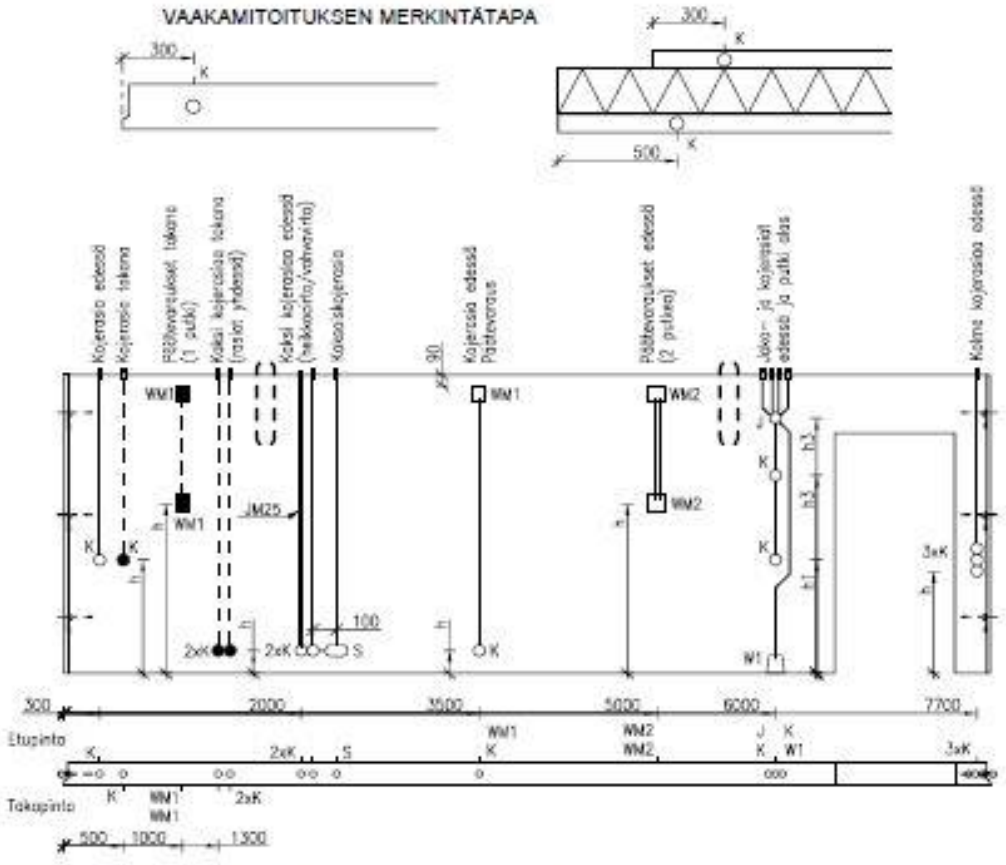
6 Piirustusmerkinnät

betoni
RT/ Betoniteollisuus ry

	Työn nro	
	Päiväys	Tekijä
Rakennuskohde/Käyttökohde VÄLISEINÄELEMENTIT SÄHKÖMERKINTÖJEN LUKUOHJE	Sisältö SÄHKÖVARUSTEIDEN MITOITUS VÄLISEINÄELEMENTISSÄ LUKUOHJE	

- Sähkövarusteiden mitoitus tehdään valmistuspiirustuksiin
- Mitat annetaan millimetreinä
- Vaakamitoitus merkitään elementin uloimmasta reunasta raslan keskelle
- Sandwich-elementissä vaakamitoitus merkitään erikseen sisä- ja ulkokuoren uloimmasta reunasta raslan keskelle
- Pystymitoitus merkitään elementin alareunasta raslan tai varauksen keskelle
- Yläreunaltaan vinojen elementtien pystymitoitus tehdään elementin alareunasta
- Varauksen sijoitus merkitään elementin yläpinnasta varauksen yläreunaan
- Sähköputket ovat vakiona JM20 ellei toisin mainita, esim. JM25 merkintä ja lisäksi suositellaan käyttämään paksumpaa viivaa
- Putki merkitään yhdellä viivalla
- Sähköputket varustetaan pääteholkilla tai talvutusjatkolla ja raslat nysillä

HUOM! Patteriseinissä vaakavedot, vinovedot ja vedot pinnasta pintaan eivät ole mahdollisia. Vaaka- ja vinovetojen käytöstä (vaakavaluseinät) sovittava erikseen kohteen aloituspalaverissa.



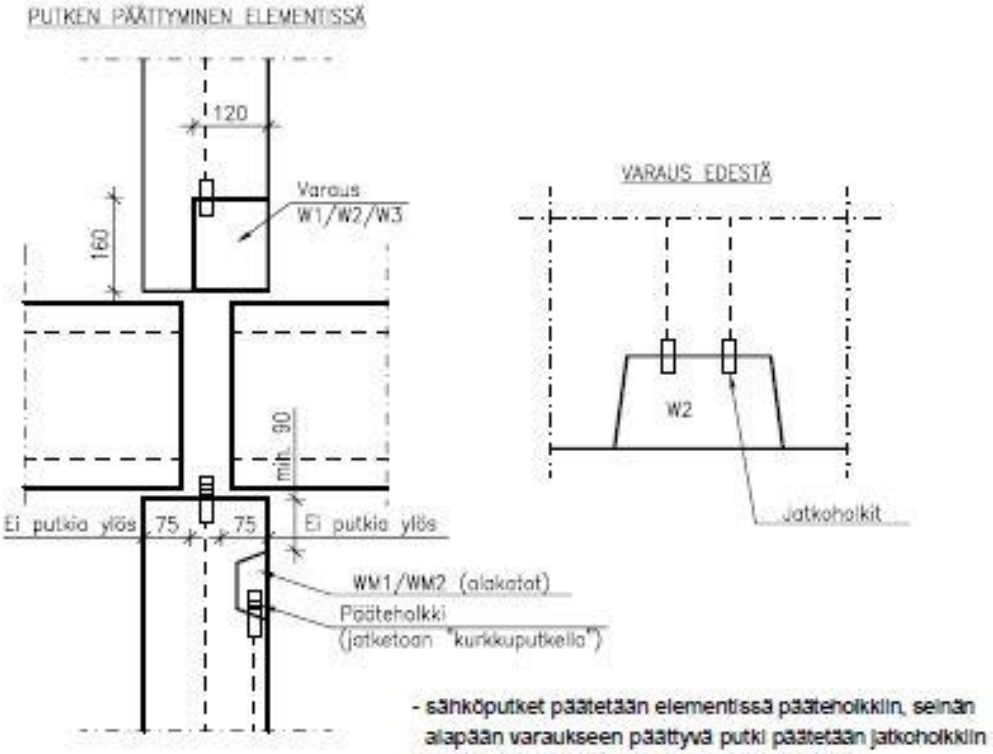
Kuva 10 Sähkövarusteiden mitoitus väliseinäelementissä.

betoni RT/ Betoniteollisuus ry	Työn nro Päiväys / Tekijä	
Rakennuskohde/Käyttökohde SUUNNITTELUOHJE VÄLISEINÄELEMETIT	Sisältö SÄHKÖTARVIKKEET SÄHKÖVARAUKSET PUTKITUS	

MERKINNÄT

○	Rasia edessä
●	Rasia takana
- - -	Putki takana, JM20 ellei kokoa merkitty
——	Putki edessä
K	Kojerasia
S	Kaksoiskojerasia
J	Katto- tai jakorasia
W1	Varaus 150x160x120 (1 putki)
W2	Varaus 270x160x120 (2 putkea)
W3	Varaus 340x160x120 (useampia putkia)
WM1	Päätevaraus 75x125x50 (1 putki)
WM2	Päätevaraus 110x125x50 (2 putkea)

Pyritään käyttämään ensisijaisesti päätevarauksia WM1 ja WM2



- sähköputket päätetään elementissä pääteholkkiin, seinän alapään varaukseen päättyvä putki päätetään jatkoholkkiin
- pääteholkista noin 50 mm esiin betonivalusta
- pääteholkki suljetaan aina suojatulpalla tai käytetään umpinaista talvutusjatkoa

Kuva 11 Sähkötarvikkeet ja -varaukset ja putkitukset väliseinäelementissä.

betoni
RT/ Betoniteollisuus ry

Työn nro

Päivitys | Tekijä

Rakennuskohde/Käyttökohde
SUUNNITTELUOHJE
VÄLISEINÄELEMENTIT

Sisältö
SÄHKÖTARVIKKEET
SÄHKÖVARAUKSET
SYMBOLILUETTELO

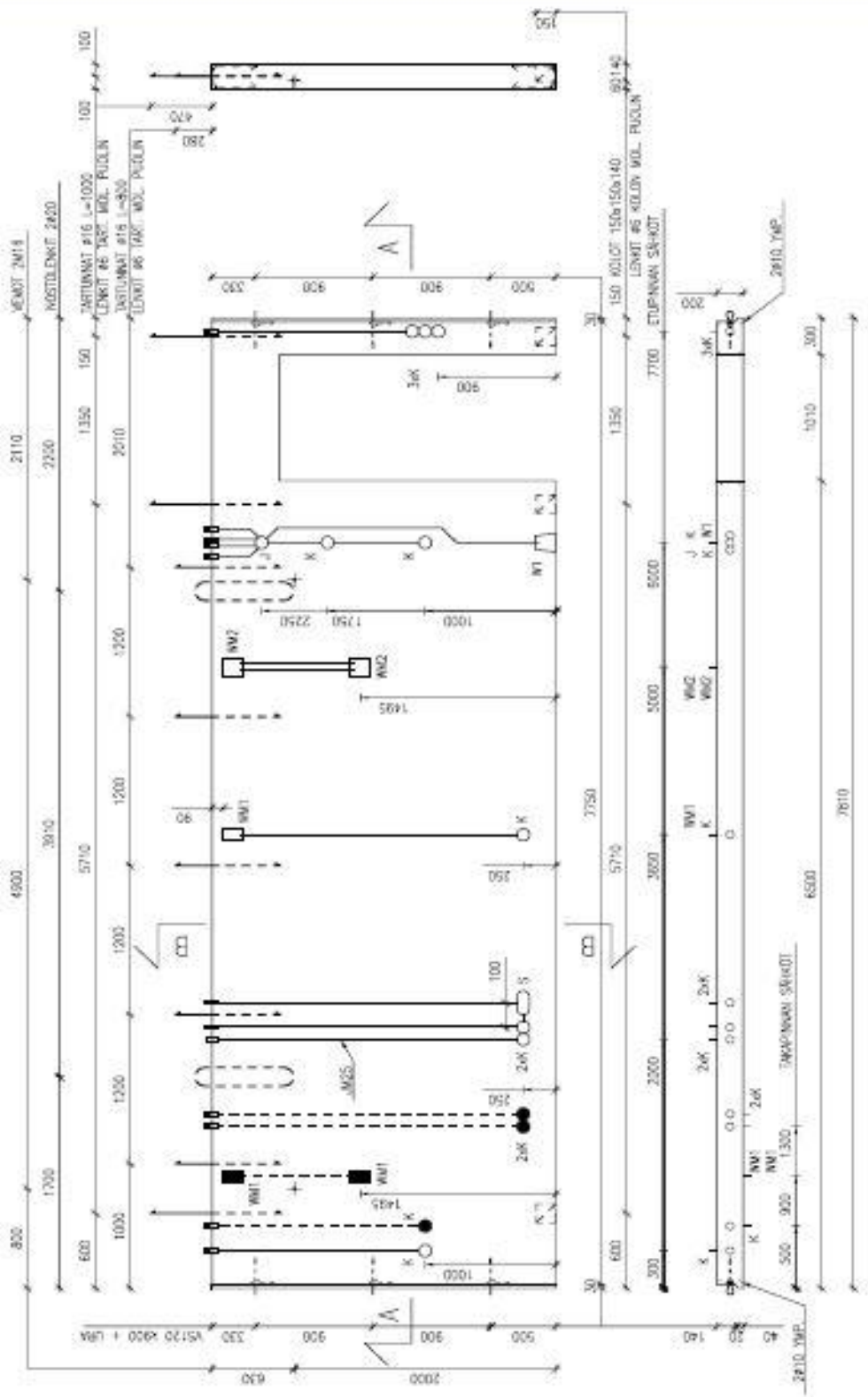
Etupinnassa	Takapinnassa	
K	K	Kojerasia (ABB AU3.2; Schneider Electric JR00)
2xK	2xK	2 kojerasiaa toisissaan kiinni Yhdyskappale ABB PMR71; Schneider Electric JL71)
K 100 K	K 100 K	Kojerasiat vakioetäisyydellä Yhdyskappaleet ABB PMR490, PMR502; SE JL85, JL100 (Heikkovirta-/vahvavirtarasiat)
S	S	Kaksoiskojerasia, huom. asennussuunta (ABB AU17.2; Schneider Electric JR20)
J	J	Jakorasia (ABB AU19; Schneider Electric JR08)
W1	W1	Varaus 150x160x120 (lev x kork x syv)
W2	W2	Varaus 270x160x120 (lev x kork x syv)
W3	W3	Varaus 340x160x120 (lev x kork x syv)
WM1	WM1	Päätevaraus 75x125x50 (lev x kork x syv), 1 putki (alakatot, kaapistojen ylälistat)
WM2	WM2	Päätevaraus 100x125x50 (lev x kork x syv), 2 putkea (alakatot, kaapistojen ylälistat)
		Jatkoshalkki (ABB AJ16, AJ20, AJ25; Schneider Electric RJM16, RJM20, RJM25)
		Pääteholkki (ABB AJ5.16, AJ5.20)
		Putkinysä (ABB AN16, AN20, AN25; Schneider Electric JN20, JN25)

Kuva 12 Sähkötarvikkeiden ja -varauksien symboliluettelo.

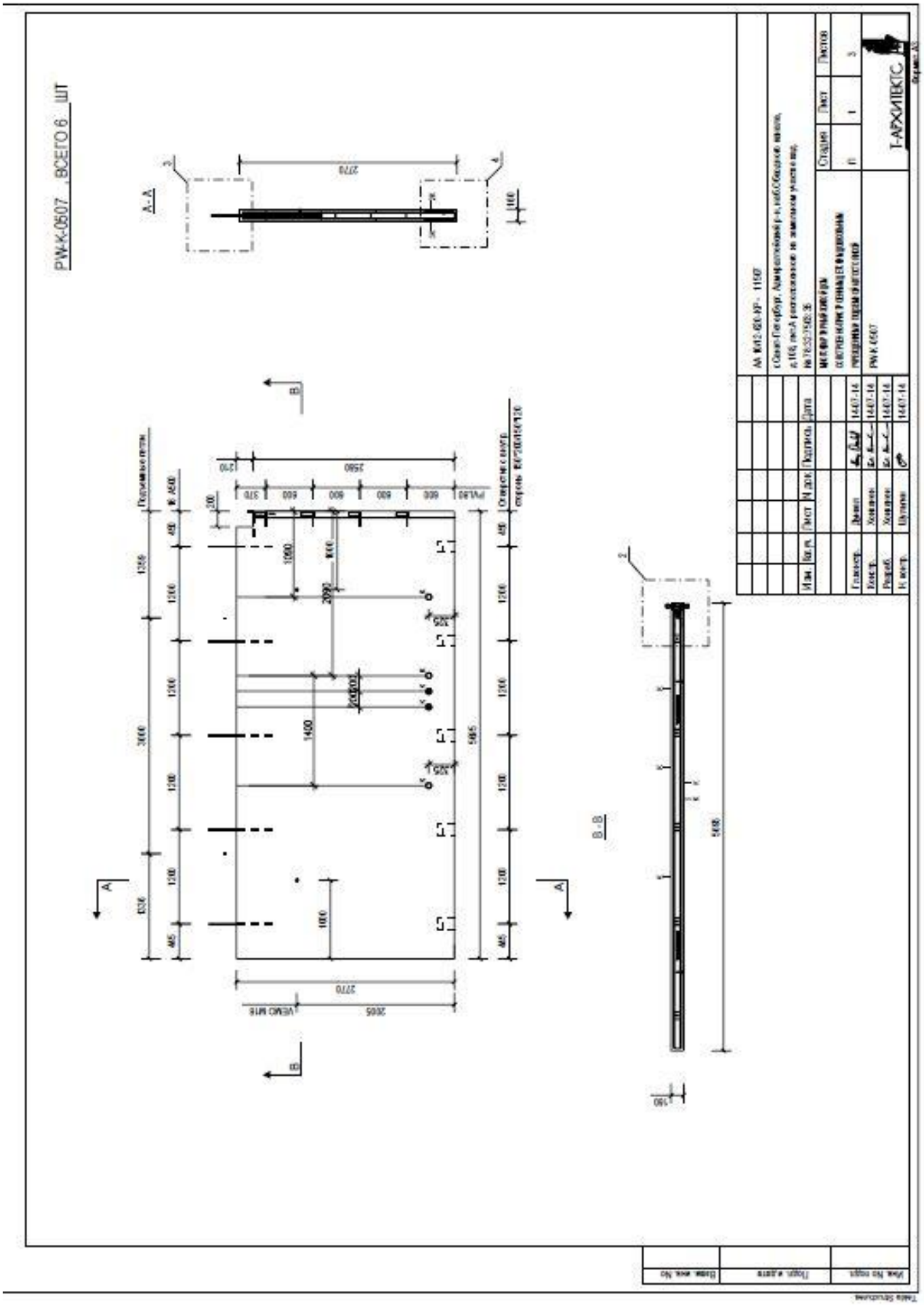
LIITE 2: MALLIELEMENTIT

Asennusesimerkki

Mallielementin sähköasennukset



Kuva 13 Mallielementti V-02.


РЖ-К-0507 , ВСЕГО 6 ШТ
А-А
В-В
Т-АРХИТЕКТС