SAVONIA

OPINNÄYTETYÖ - **AMMATTIKORKEAKOULUTUTKINTO**
TEKNIIKAN JA LIIKENTEEN ALA

LIITTOPALKKIMALLINNUS-TYÖKALUN KEHITTÄMINEN TEKLA STRUCTURES OPEN API -OHJELMOINTIRAJAPINNALLA

TEKIJÄ/T: Aleksi Maunula

SAVONIA-AMMATTIKORKEAKOULU

OPINNÄYTETYÖ
Tiivistelmä

Koulutusala Tekniikan ja liikenteen ala	
Koulutusohjelma Rakennustekniikan koulutusohjelma	
Työn tekijä(t) Aleksi Maunula	
Työn nimi Liittopalkki mallinnustyökalun kehittäminen Tekla Structures Open API ohjelmointirajapinnalla	
Päiväys 3.12.2015	Sivumäärä/Liitteet 45/49
Ohjaaja(t) lehtori Viljo Kuusela, lehtori Harry Dunkel	
Toimeksiantaja/Yhteistyökumppani(t) Lujabetoni Oy	

Tiivistelmä

Tämän opinnäytetyön tavoitteena oli kehittää mallinnustyökalu Tekla Structures -rakennesuunnitteluohjelmistoon LujaBetonin valmistamalle LujaBeam-liittopalkille ja piilokonsolille. Työkalut kehitettiin käyttäen Open API -ohjelmointirajapintaa. Opinnäytetyön toimeksiantajana toimi LujaBetoni Oy.

Opinnäytetyötä aloitettiin kehittämään Teklan tarjoamalla Open API -rajapinnalla. Työkalut olisi voinut kehittää Tekla Structuresissa käyttäen Custom Componentia, mutta Open APIn avulla työkalusta saatiin monikäyttöisempi. Opinnäytetyössä käsitellään kehitystyössä käytettyjä työkaluja ja menetelmiä. Lisäksi selostetaan kuinka työkalut kehitettiin, niiden toiminnallisuudet ja testaustapaukset. Opinnäytetyön perustana käytettiin Tekla tarjoamia dokumentteja, sekä muita aiheesta tehtyjä opinnäytetöitä ja internet-lähteitä. Työkalun ohjelmoinnissa käytettiin C# -ohjelmointikieltä. Käyttöliittymä toteutettiin Windows Forms -tyyppisenä, joka käyttää hyväkseen Microsoftin tarjoamaa .NET-arkkitehtuuria.

Työn tuloksena saatiin kehitettyä toimivat työkalut LujaBetonin valmistamalle liittopalkille, sekä piilokonsoliliitokselle Tekla Structures -ohjelmistoon, sekä ohjeet niiden käytölle.

Avainsanat
Tekla, mallinnustyökalu, API, C#, .NET,

SAVONIA UNIVERSITY OF APPLIED SCIENCES

THESIS
Abstract

Field of Study Technology, Communication and Transport			
Degree Programme Degree Programme In Construction Engineering			
Author(s) Aleksi Maunula			
Title of Thesis Development of Tekla Structures Composite Beam with Open API Programming Interface			
Date	3 December 2015	Pages/Appendices	45/49
Supervisor(s) Mr. Viljo Kuusela, Lecturer, Mr. Harry Dunkel, Lecturer			
Client Organisation /Partners Lujabetoni Oy			

Abstract

The objective of this thesis was to develop a modeling tool for the composite beam and one connection in the Tekla Structures designing software. This tool was created with the Tekla Open API, programming interface. This thesis was commissioned by Lujabetoni Oy.

This thesis was decided to be developed using the Tekla Open API programming interface. It would have been possible to create the tool with Tekla Structures Custom Component, but Open API is more efficient to create tools in Tekla. First this thesis goes through what kind of tools and methods were needed in development. Secondly this thesis undergo how the modeling tools were created, their functionality and testing. This thesis is mostly based on documents from Tekla, similar theses and Internet sources. C# was used as the programming language. Furthermore the user interface will be created using the Windows Forms technology which exploits Microsoft .NET architecture.

As a result, the development process generated modeling tools for composite beam and connection in Tekla Structures and instructions on how to use them.

Keywords
Tekla, modeling tool, API, .NET

SISÄLTÖ

1 JOHDANTO

1.1 Tausta ja tavoitteet

Lujabetoni kuuluu Luja-yhtiöihin, joka on yksi Suomen suurimmista rakennusalan yrityksistä. Luja on toiminut rakennusalalla jo 60 vuoden ajan. Lujabetoni palvelee asiakkaitaan luotettavasti kaikessa betonirakentamisessa. Tietomallintaminen on yleistymässä, joten Lujabetoni haluaa olla osana kehittämässä tätä suunnittelualuetta. (Kokkonen 2015-05-20.). Lujabetoni haluaa kehittää kaikista valmistamistaan tuotteistaan mallinnustyökalut Tekla Structures -ohjelmistoon. Nämä tuotteet on tarkoitus laittaa ladattavaksi Teklan tarjoamaan pilvipalveluun nimeltä Tekla Warehouse, josta rakennesuunnittelijat voivat ladata tarvitsemiaan työkaluja. Näin ollen suunnittelijoiden ei tarvitse lähteä itse mallintamaan jokaista pientä rakennusosasta, josta saattaa tulla aikaa ja siten rahaa vievä suunnitteluvaihe. Tämän opinnäytetyön aiheena oleva liittopalkki ja piilokonsoliliitos ovat siis vain osa Lujabetonin tarjoamista tuotteista, jotka tulevat lyötymään Tekla Warehousesta.

Rakennusten tietomallit ovat yleistymässä, mikä vaikuttaa koko rakennusalan suunnittelumenetelmiin. Totutut toimintatavat ja käytännöt kokevat jatkuvaa muutospainetta suunnittelun työkalujen kehittyessä. Tietomallien käyttöä on tutkittu ja ohjeistettu kansallisissa tietomallintamista käsitelleissä hankkeissa. Tietomallipohjaisesta suunnittelusta saa parhaan mahdollisen hyödyn irti jos se otetaan käyttöön heti suunnittelutyön alkaessa. Tehokkuutta tietomallintamiseen saadaan myös lisää mitä useampi suunnittelija ja toimija tietomallia hyödyntävät. Näin ollen hyödyt jakaantuvat yksittäisen suunnittelualan sisäisistä hyödyistä laajemmin koko rakentamisprosessille. (Elementtisuunnittelu, mallintava suunnittelu.)

Opinnäytetyön tavoitteena on kehittää toimivat mallinnustyökalut Tekla Structures -rakennesuunnittelu-ohjelmistoon LujaBetonin valmistamalle liittopalkille ja piilokonsoliliitokselle ja ohjeet niiden käytölle. Kehitystyö tehdään Tekla Structures- rakennesuunnitteluohjelmiston tarjoamalla Tekla Open API -ohjelmointirajapinnalla. Tekla Open API on avoin ohjelmointirajapinta, joka perustuu olio-teknologiaan. Sen avulla voidaan kehittää mallinnus- ja piirustustyökaluja Tekla Structures -ohjelmistoon. Junkkarinen (2014) toteaa Savonia-ammattikorkeakouluun tekemässään opinnäytetyössään, että Open API tarjoaa monipuolisen kehitysympäristön ja on tulevaisuuden kannalta varsin potentiaalinen kehitystyön väline Tekla Structures -ohjelmistossa. Tämän vuoksi opinnäytetyössä kehitettävät työkalut päätettiin lähteä tuottamaan Teklan tarjoamaa avointa rajapintaa käyttäen.

1.2 Lyhenteet ja määritteet

BIM (Building Information Model) = rakennuksen tietomalli

API (Application Programming Interface) = ohjelmointirajapinta

.NET Framework = Microsoftin kehittämä ohjelmistokomponenttikirjasto

C# (C sharp) = Microsoftin luoma ohjelmointikieli

Plugin = liitännäinen, joka toimii yhdessä isäntäsovelluksen kanssa

Tietomalli = rakennuksen ja sen rakennusosien tietojen kokonaisuus digitaalisessa muodossa

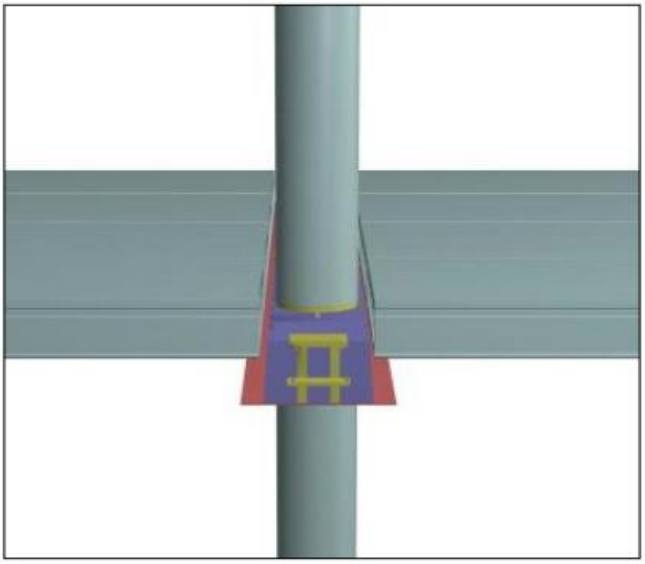
Tekla = Tekla Structures -rakennesuunnitteluohjelma

Visual Studio = Microsoftin tarjoama ohjelmankehitysympäristö

2 MALLINNUSTYÖKALUJEN KEHITYSTARPEET

Lujabetoni Oy valmistaa LujaBeam–runkojärjestelmää, johon kuuluu LujaBeam-palkit, LujaBeam-pilarit ja Luja-ontelolaatat. Lujabetoni on myös kehittänyt oman Luja-piilokonsolijärjestelmän, jota se käyttää liitoksinaan perinteisten runkorakenteiden liitosten lisänä.

LujaBeam-palkki ei rajoita putkisto-, kanavisto- ym. asennuksia tai tilan hyödyntämistä kerroskorkeutena, sillä se jää laataston sisään. Sen päälle asennettavia palkkeja tai ontelolaattoja ei tarvitse tukea väliaikaisesti, joten se vastaa pienten jännebetonipalkkien asentamista. Laatat asennetaan suoraan teräslaipan päälle. Laatan lisäksi palkille asennetaan tarvittavat raudoitukset ja saumavalut. Palotilanteessa palkki on suunniteltu toimimaan ilman teräksistä alalaippaa. Saumavalun, raudoituksen ja muodon yhteistoiminnalla onnistutaan siirtämään kuormat palkille ongelmitta. Näin ollen palkkia ei tarvitse palosuojata. (Suunnitteluohje LujaBeam-järjestelmä)
Lujabetoni valmistaa kahta erilaista LujaBeam–palkkia: LBL - ja LB -palkkia. LBL-palkissa on yhtenäinen teräslevy pohjan alueella ja se jatkuu reunojen yli muodostaen leuat. LB-palkissa taas on kaksi erillistä teräsleukaa, jotka muostuvat kulmaraudoista. Molempien palkkien päälle asennetaan työmaalla ontelo- tai kuorielementtilaatasto. Elementtilaatoista ja palkista muodostuu liittorakenne kun laatta ja palkki on raudoitettu ja betonoitu.



Kuva 1 LBL-palkki asennettuna (Suunnitteluohje LujaBeam–järjestelmä)

2.1 Vastaavat liittopalkit ja konsoliliitokset markkinoilla

Liittopalkeille ja konsoliliitoksille löytyy useita valmistajia markkinoilta. Liittopalkkeja valmistaa muun muassa Peikko Group, Betonimestarit, Ruukki ja Lujabetoni. Konsoliliitoksia tarjoaa muun muassa Peikko, Anstar ja Lujabetoni. Kilpailijat ovat pääosin tehneet tarvittavat komponentit Tekla Structuresiin tietomallintamista varten.

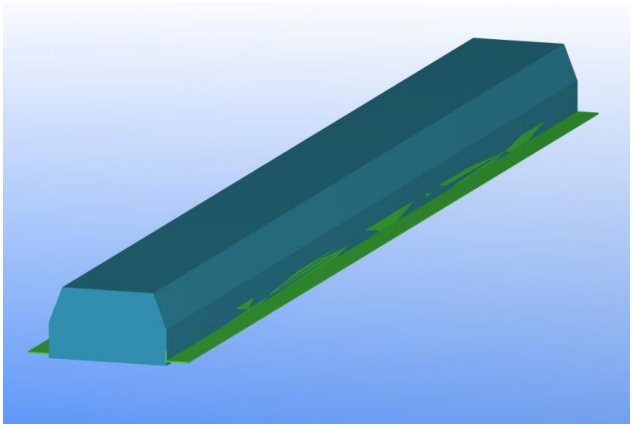
2.1.1 Peikko Oy:n Deltapalkki

Deltapalkki on Peikko Oy:n valmistama liittopalkki, jota käytetään betonirakenteiden yhdistämisessä väli- ja yläpohjissa. Se on markkinoiden suosituimpia liittopalkkeja ja sitä on valmistettu vuodesta 1990 saakka. Valmistajan mukaan deltapalkkien avulla on rakennettu yli 10 000 rakennusta. Peikko tarjoaa tällä hetkellä useita suunnittelua helpottavia työkaluja Tekla Structuresissa. Deltapalkista löytyy tällä hetkellä profiilit Tekla Structuresin profiilikirjastosta. (Peikko.fi.)

Kuva 2 Deltapalkki (Peikko.fi)

2.1.2 Betonimestarit Oy:n BM-palkki

BM-palkki on Betonimestarit Oy:n valmistama jännebetonileukapalkki. Siihen kiinnittyy harjaterästappien välityksellä teräslaippa, joka muodostaa leuan. Tämän leuan päälle tapahtuu ontelo- tai kuorilaataston asennus. BM-palkki on ladattavissa Tekla Structures ohjelmistoon Tekla Warehousesta. (Betonimestarit.fi.)

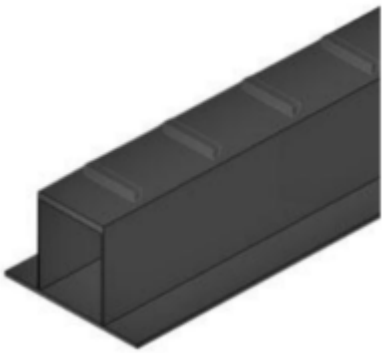


Kuva 3 BM-palkki (Maunula 2015)

2.1.3 Ruukki Oy:n CWQ-palkki

CWQ–palkki kuuluu osana Ruukin valmistamaan monikerrosrakentamisen järjestelmäkokonaisuutta. Se muodostaa liittorakenteen sen päälle asennettavien ontelo- tai kuorilaattojen ja betonivalun

kanssa. Sitä käytetään ylä- ja välipohjissa ja se on nopea asentaa. Palkki on matala ja jää laataston sisään, joten se ei vaikuta pienentävästi kerroskorkeuteen. Näkyviin jää alalaippa valmistuneissa rakenteissa. CWQ-profiili lyötyy Tekla Structuresin profiilikirjastoista. (Ruukki.fi.)



Kuva 4 CWQ-palkki (Ruukki.fi)

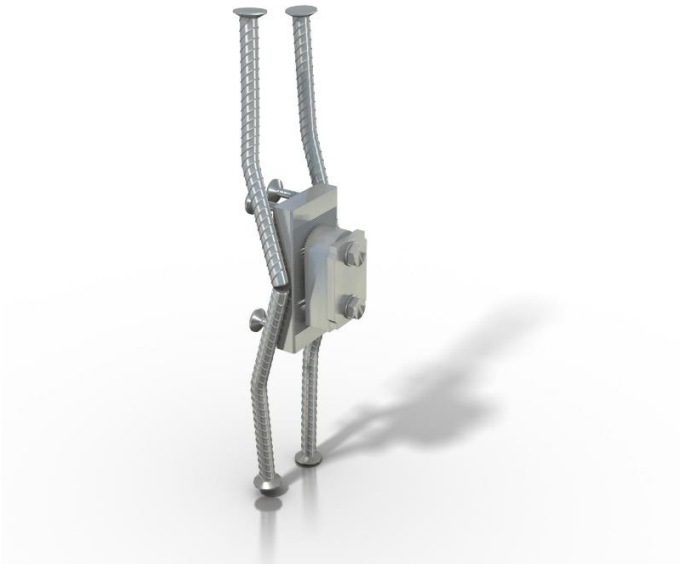
2.1.4 Anstar Oy:n AEP-konsoliliitos

Käytetään liittämään erilaisia betonipalkkeja pilariin. Liitos suunnitellaan nivelellisenä, mutta se on kuitenkin jäykkä väännölle. AEP-konsoliliitoksen avulla ontelolaatat voidaan asentaa betonipalkille ilman tuentaa. AEP-piiloksonsoleista löytyy komponentit Tekla Structuresissa. (Anstar.fi.)

Kuva 5 AEP-konsoliliitos (Anstar.fi)

2.1.5 Peikko Oy:n PCs-konsoliliitos

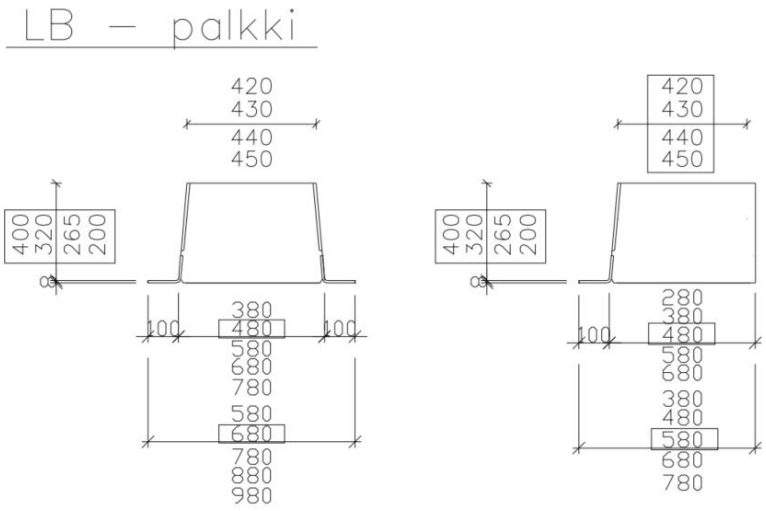
PCs–konsoli on modulaarinen piilokonsoli pilarille. Se on tarkoitettu betoni- ja teräs palkkien kiinnittämiseen pilareihin tai seiniin. Liitoksen korkeutta ja sivusuuntaa voidaan muokata, jonka vuoksi liitos on varsin hyödyllinen mittavirheitä sattuessa. Myös PCs-liitoksesta löytyy työkalut Tekla Structuresissa. (Peikko.fi.)



Kuva 6 PCs-konsoliliitos (Peikko.fi)

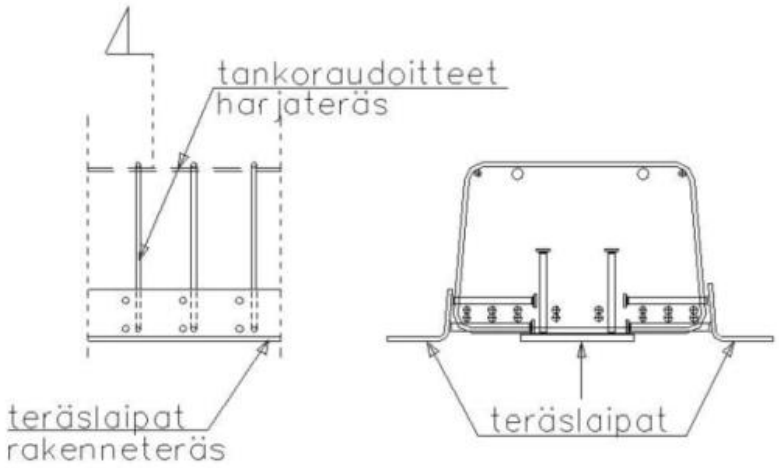
2.2 Lähtökohta LB-palkin mallinnustyökalun kehittämiselle

Lujabetoni tarjoaa suunnittelijalle erikokoisia palkkivaihtoehtoja. LB-palkkia on tarjolla yksipuoleisella, sekä kaksipuoleisella leualla. Suunnittelija voi valita kohteeseensa erikorkuisia ja -levyisiä palkkeja, mitkä soveltuvat parhaiten käyttötarkoitukseensa. Kuvasta 7 esitetty LB-palkin tyypillisimmät koot. Palkin kokonaiskorkeus muodostuu laipan ja betoniosan summasta. Reunapalkilla suositusleveys on 480 mm tai 580 mm, jolloin betoniosan leveydeksi tulee 480 tai 580mm. Suositusleveys keskipalkin alareunalle on 680 mm ja betoniosalla 480 mm. Vaarnatuille sivuille kaltevuus on 1/13. Piirustuksiin palkki merkitään esimerkiksi LB 208 * 280/380 * 6000 eli palkkityyppi * korkeus * betoniosan leveys * teräsosan leveys * pituus.



Kuva 7 Kuva 1 LB-palkit: keskipalki (vas.) ja reunapalkki (oik.) mittasuosituksineen (Suunnitteluohje LujaBeam–järjestelmä)

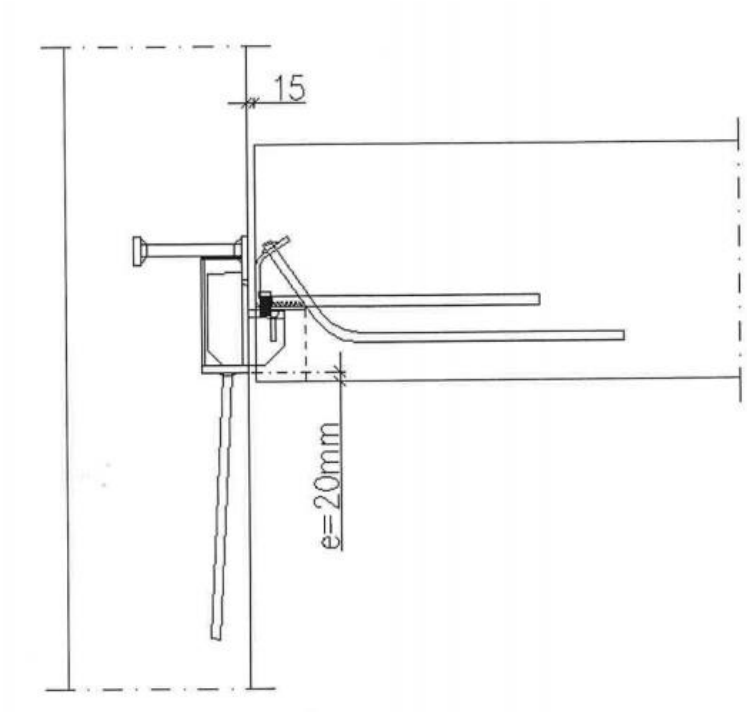
Palkit voidaan suunnitella rakenteissa myös jatkuvina. Tässä tapauksessa palkin ja raudoitusosan käsiteltävyys ja valmistus rajoittaa palkin mitan 18 metriin. Palkit suunnitellaan pilareihin nähden keskeisesti, jotta pilarille ei synny ylimääräisiä rasituksia. Teräsmateriaalina LB–palkeissa käytetään SFS-EN 10025 (1993) mukaista teräslaatua S355J2G3 tai parempaa, jonka alempi myötöraja on vähintään 355 MPa. Betoniosaan liittyvien laippojen vaarnojen teräsmateriaalina käytetään SFS-EN 10025 (1993) mukaista teräslaatua S235JR tai S355J0, jonka alempana myötörajana on vähintään 235 MPa. Betoniteräksenä käytetään standardin SFS 1215 mukaista teräslaatua A500HW. Jänneteräksenä käytetään standardin SFS 1265 mukaista vähintään lujuusluoka St 1550/1750 jännepunosta halkaisijaltaan 12,5 mm. Betonin lujuusluokka on vähintään K50 betoniosalla ja palkille tulevilla ontelolaatoilla. Saumavalut ja pintalaatta tulee olla vähintään K30-2. Kuvassa 8 on esitetty LB-palkin raudoitus.



Kuva 8 LB-palkin raudoiteosa (Suunnitteluohje LujaBeam–järjestelmä)

2.3 Lähtökohta LK-piilokonsoliliitoksen kehittämiselle

LK-piilokonsoliliitosta käytetään LujaBetoni Oy:n valmistamassa LujaBeam -palkissa ja sen liittymisessä eri korkuisiin pystyrakenteisiin. Liitos siirtää palkilta tulevat kuormat pilarille tai seinälle nivellisesti. Kuvassa 9 on LK-piilokonsoliliitos asennettuna pilariin ja palkkiin.

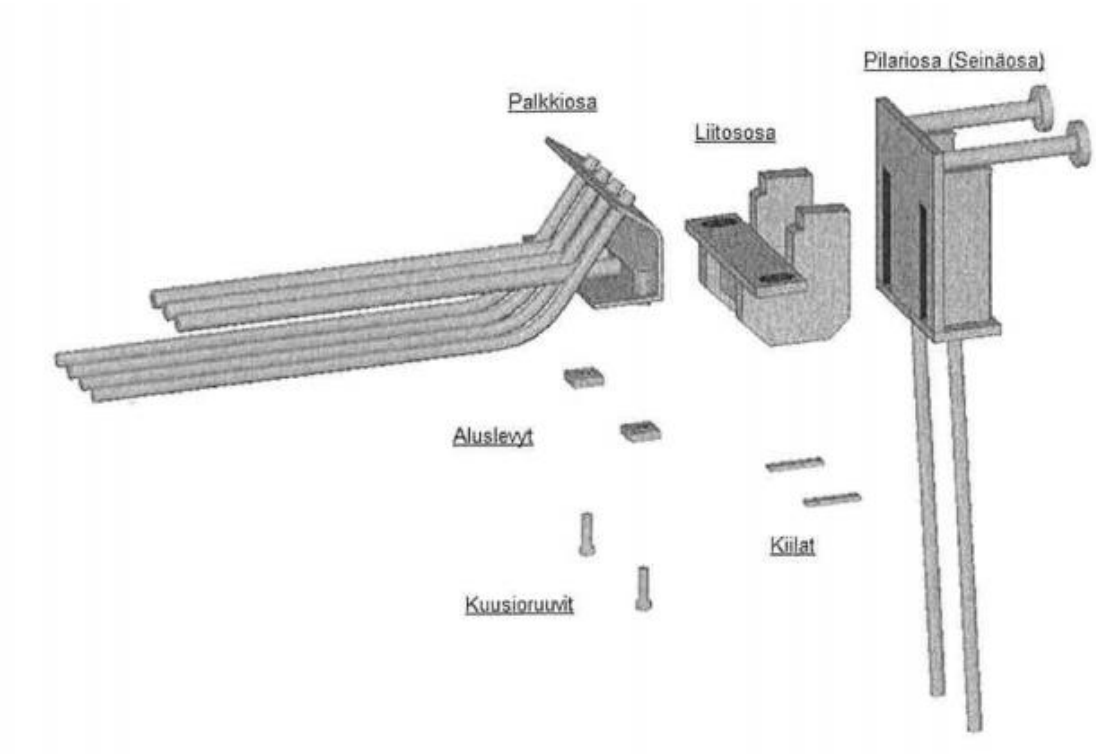


Kuva 9 LK-konsoliliitos sivusta (Suunnitteluohje LujaBeam–järjestelmä)

LK-piilokonsoliliitos koostuu kolmesta eri teräsosasta: palkkiosa, liitososa ja pilari- tai seinäosa. Kuvassa 10 on esitetty LK-piilokonsoliliitos kokonaisuudessaan. Jokaista osaa on neljää eri kokoa, jotka on esitetty taulukossa 1. Koko määräytyy kestävyyden mitoitusarvon mukaan, joka ilmoitetaan osan tunnuksessa numerolla. Yhtenäiset mitoitusarvot omaavat liitokset ovat yhteensopivia keskenään, mutta niitä ei voida sekoittaa keskenään.

Taulukko 1 LK-konsoliliitoksen koko vaihtoehdot (Maunula 2015)

Palkkiosa	Liitososa	Yksipuolinen pilariosa
LK400P	LK400L	LK400PI
LK600P	LK600L	LK600PI
LK800P	LK800L	LK800PI
LK1100P	LK1100L	LK1100PI



Kuva 10 LK-konsoliliitoksen osat (Suunnitteluohje LujaBeam–järjestelmä)

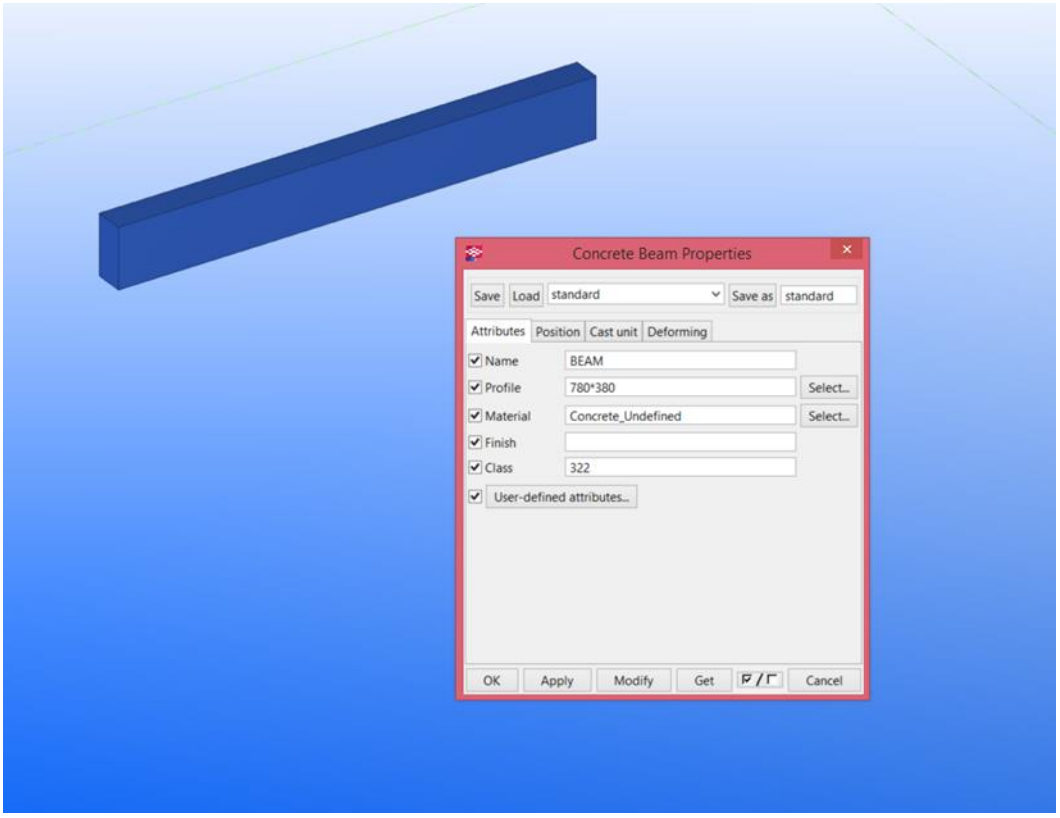
3 VAATIMUKSET KEHITETTÄVILLE TYÖKALUILLE

Tietomallien tarkkuus ja sisältövaatimukset muuttuvat suunnittelun eri vaiheissa. Rakennemalliin mallinnetaan kaikki kantavat rakenteet sekä ei-kantavat betonirakenteet. Lisäksi on mallinnettava sellaiset tilaa vievät rakennustuotteet, joiden koolla ja sijainnilla on merkitystä muille suunnittelijoille. Rakenteet tulee mallintaa siten, että tietoa siirrettäessä rakennusosan sijainti, nimi/tyyppi ja geometria siirtyvät rakennusosan mukana. (Yleiset tietomallivaatimukset 2012. Osa 5. Rakennesuunnittelu.)

Palkin kehittämisen tavoitteena on kehittää työkalu, jonka avulla luodaan tietomalliin parametrisoitu LB–palkki. Työkalulle syötettävät tiedot tulevat olemaan samanlaisia, kuten Tekla Structuresin tarjoamalla oletuspalkilla, joka on esitetty kuvassa 11.

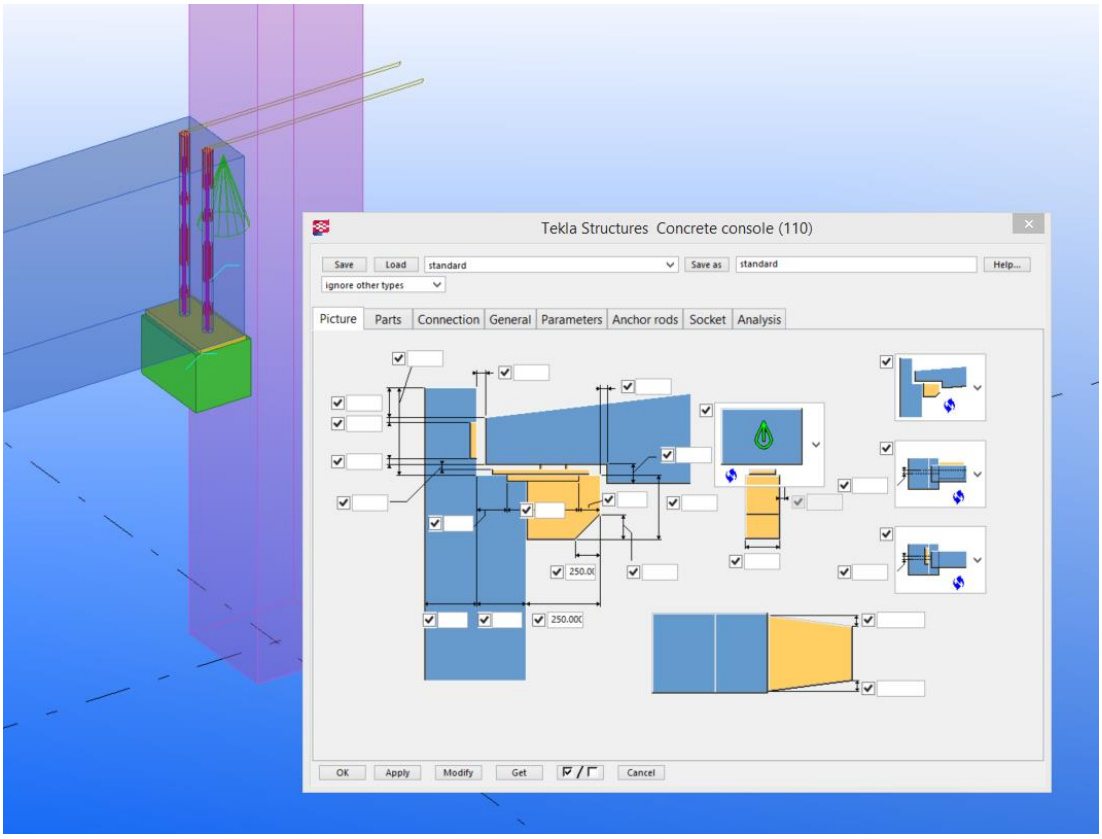
Tekla Structuresin betonipalkille syötettäviä tietoja ovat:

- nimi
- profiili
- materiaali
- pääte
- luokka
- UDA (palkin tietojen tarkempi määrittely)
- etuliite
- aloitusnumero.



Kuva 11 Esimerkki betonipalkin dialogista (Maunula 2015)

Piilokonsoliliitoksen kehittämisessä on tavoitteena kehittää työkalu, jonka avulla luodaan tietomalliin LK-piilokonsoliliitos. Esimerkkinä mitä tietoja liitos tulee sisältää, voidaan katsoa Tekla Structuresin tarjoamaa betoni-pilari-konsoliliitosta, joka on esitetty kuvassa 12. Tekla Structuresin tarjoamiin oletusliitoksiin ei ole mahdollista syöttää attribuutteja, sillä ne sisältävät vain syötteitä mitoille. Mittojen avulla liitos saadaan asetettua oikealle paikalleen.



Kuva 12 Esimerkki liitoksen dialogista (Maunula 2015)

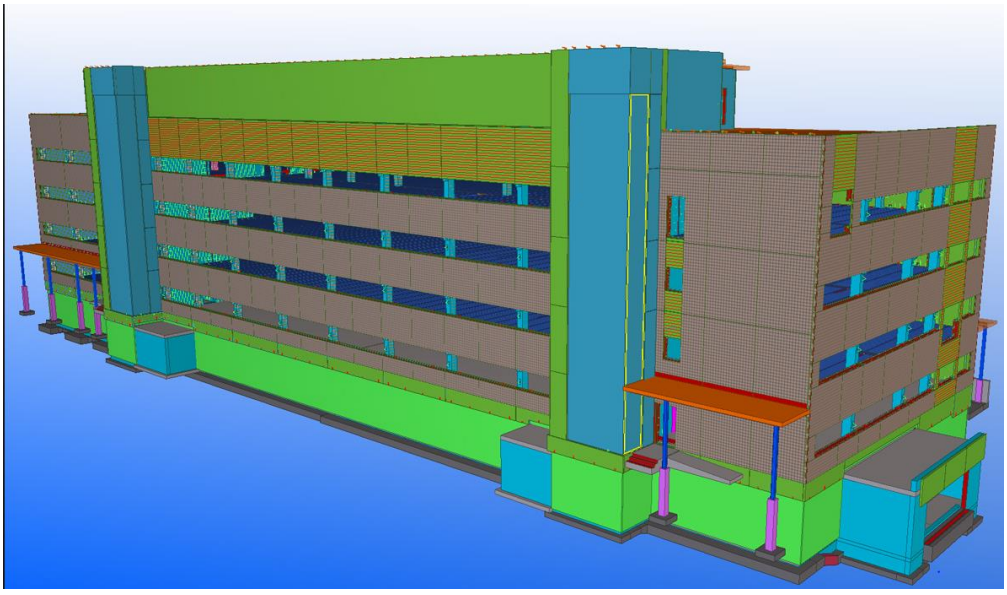
Kehitettävä LK-piilokonsoliitos tulee sisältämään syötteitä mitoille, mutta myös mahdollisuudet syöttää muutamia attribuutteja. Nimeäminen on tärkeää liitoksille, sillä niiden avulla pystytään automatisoimaan tulostusasetuksia ja luetteloita.

4 KEHITYSTYÖN VAATIMAT TYÖKALUT

Komponentti luodaan Tekla Structures -rakennesuunnitteluohjelmiston tarjoamalla Open API -ohjelmointirajapinnalla. Tällä ohjelmointirajapinnalla rakennesuunnittelija voi luoda itse suunnittelua helpottavia työkaluja ja sovelluksia.

4.1 Tekla Structures

Tekla tuottaa tietomallinnusohjelmistoja rakennus-, infrastruktuuri- ja energiatoimialoille. Asiakkaita Teklalla on yli 100 maassa ja ohjelmiston kehityksestä vastaa lähes 600 työntekijää. Teklasta tuli vuonna 2011 heinäkuussa osa Trimble-konsernia. Teklan luomia rakennustietomallinnusohjelmistoja kutsutaan BIM-ohjelmistoiksi ja ne on luotu tehostamaan työtä rakennusalalla. BIM-ohjelmistoja käytetään ympäri maailmaa monenlaiseen rakentamiseen silloista asuinrakennuksiin ja öljynporauslauttoihin. Niillä pystytään luomaan, yhdistämään, jakamaan ja hallitsemaan laajoja, monimutkaisia ja tarkkoja tietomalleja. (Tekla 2015b.)



Kuva 13 Rakennuksen tietomalli (Tekla 2015)

4.2 Tekla Open API

Teklan kehittämä ohjelmointirajapinta, joka mahdollistaa sovellusten ja lisätoimintojen kehittämisen Tekla Structures -rakennesuunnitteluohjelmistossa ja integroida ne omaan ympäristöösi. Tekla Open API toteutetaan Microsoftin kehittämällä .NET-tekniikalla. Tekla Open API:lla kehitetyt sovellukset, jotka toimivat yhdessä Tekla Structuresin kanssa kutsutaan laajennuksiksi. Tekla Open API:lla voi esimerkiksi nauhoittaa ja suorittaa käyttöympäristön toimintoja. Niiden avulla voidaan automatisoida rutiininomaisia tehtäviä kuten päivittäisien raporttien luomista. OPEN API:lla voi myös luoda automatisoituja työkaluja usein tarvituille työkaluille kuten esimerkiksi luomaan perus rakenneosa tai lisäämään tyypillinen detalji piirustuksiin. Open API:a ja .NET:a voidaan käyttää hyödyksi, kun siirretään tietoa Tekla Structuresin ja toisen suunnitteluohjelmiston välillä. (Tekla 2015a.)

4.3 Microsoft Visual Studio

Komponentin kehittämisessä käytettiin Microsoftin luomaa Microsoft Visual Studio 2013 -ohjelmointityökalua. Microsoft Visual Studio on ohjelmankehitysympäristö, jossa voidaan käyttää monia eri ohjelmointikieliä. Komponentin luomisen suoritettiin C#-kielellä. Visual Studio olisi myös mahdollistanut Visual Basic -, C++ - ja Java -kielten käytön. Komponentti luotiin Windows Forms -tyyppisenä käyttöliittymänä.

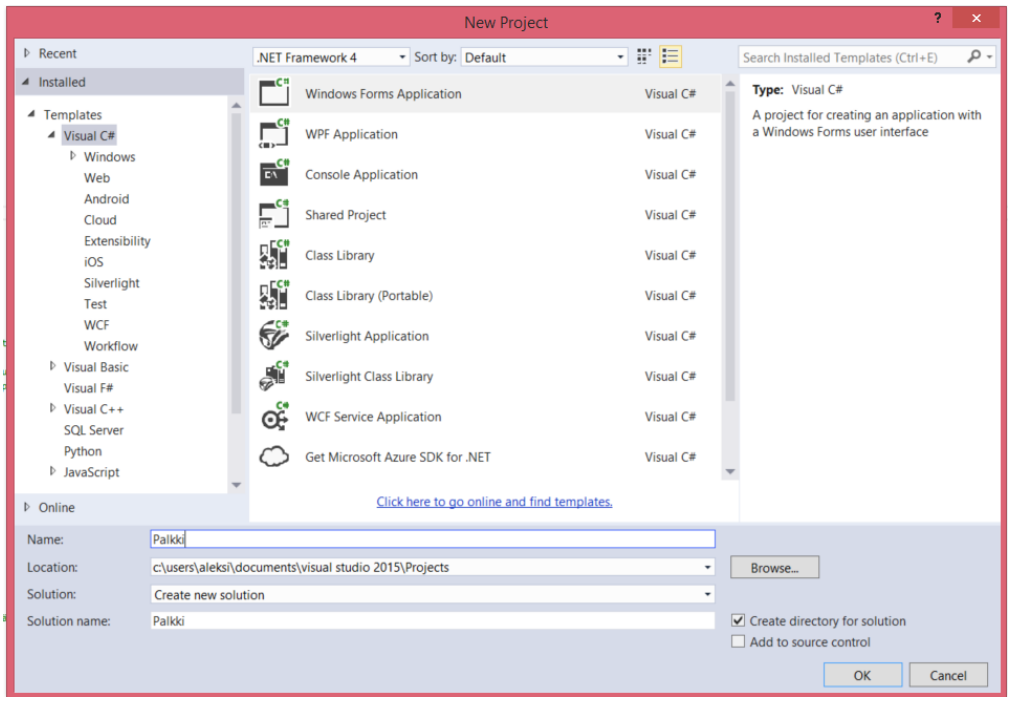
4.4 Ohjelmointikieli

C# on vahvasti ja staattisesti tyypitetty (nimisidonnat tehdään tyypityksen käännösaikana) olio-orientoitunut ohjelmointikieli. C# kehittymiseen on vaikuttanut muut olio-orientoituneet kielet kuten Java ja C++. Kielen suunnittelun pyrkimyksenä oli säilyttää C++:n ja C:n hyvät ominaisuudet ja muuttaa tai laajentaa kielen lauseoppia vain, jos se on tarpeellista. C# -kieli tukee kaikkia ohjelmointikielen periaatteita, jotka ovat periytyminen, luokittelu ja periytyminen. Näiden lisäksi se tukee kapselointia ja monimuotoisuutta, joten se täyttää kaikki modernit ohjelmointikielen tunnusmerkit. (Sivonen, 2004, Helsingin yliopisto)

4.5 Esimerkkisovellus

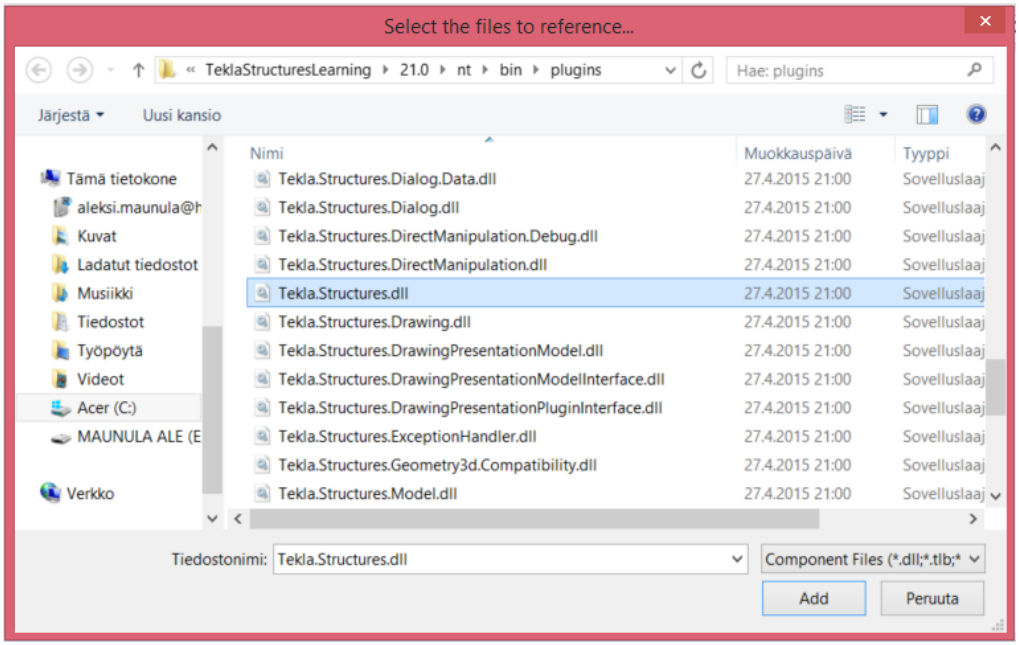
Tässä esimerkissä kehitetään erittäin yksinkertainen .NET applikaatio. Tämä applikaatio hyödyntää Open API:a, jonka avulla saadaan luotua palkki Tekla Structuresin malliin nappia painamalla. Tekla Structuresin tulee olla käynnissä applikaatiota suoritettaessa.

Ensiksi luodaan uusi projekti Visual Studiossa ja annetaan sille nimi Palkki. Esimerkkisovellus kehitetään Windows Forms -tyyppisenä ja annetaan projektille nimeksi Palkki, kuten kuvasta 16 voi nähdä.



Kuva 14 Uuden projektin luominen (Maunula 2015)

Kun projekti on saatu luotua, siirrytään lisäämään referenssejä, jotta pystytään kutsumaan tarvittavia objekteja näistä referenssikirjastoista. Tähän applikaatioon tarvitsemme Tekla.Structures.dll ja Tekla.Stuctures.Model.dll referenssit. Kuva 15 esittää kuinka referenssejä lisätään.



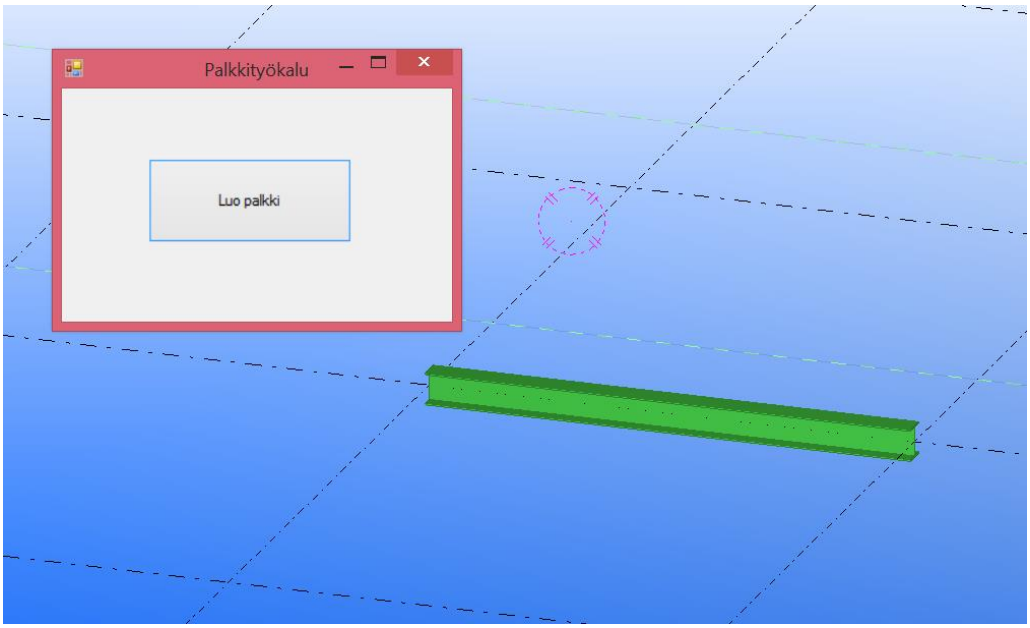
Kuva 15 Referenssien valinta (Maunula 2015)

Seuraavaksi luodaan painike, jonka perään kirjoitetaan suoritettava koodi. Painikkeen lisäys tapahtuu Toolboxin kautta tuplaklikkaamalla Button–käskyä. Kun painike on luotu ja nimetty haluamaksi tuplaklikataan painiketta, jotta päästään kirjoittamaan koodia sen alle.

```
namespace Palkki
{
    public partial class PalkkiForm : Form
    {
        public PalkkiForm()
        {
            InitializeComponent();
        }

        private void button1_Click(object sender, EventArgs e)
        {
            // Luodaan yhteys avoinna olevaan malliin
            Model Malli = new Model();

            Beam Palkki = new Beam(); // Luodaan palkki
            Picker myPicker = new Picker(); // Valitaan palkille luontipisteet

            Palkki.StartPoint = myPicker.PickPoint(); // Palkin aloituspiste
            Palkki.EndPoint = myPicker.PickPoint(); // Palkin lopetuspiste

            // Palkin asettelu
            Palkki.Position.Depth = Position.DepthEnum.MIDDLE;
            Palkki.Position.Plane = Position.PlaneEnum.MIDDLE;
            Palkki.Position.Rotation = Position.RotationEnum.TOP;

            // Palkin tiedot
            Palkki.Name = "Palkki";
            Palkki.Profile.ProfileString = "HEA400";
            Palkki.Material.MaterialString = "S355JR";
            Palkki.Class = "3";
```

```
            // Luodaan palkki malliin
            Palkki.Insert();

            // Varmistetaan että muutokset, joita mahdollisesti tehdään päivittyy myös malliin
            Malli.CommitChanges();
        }
    }
}
```

Applikaation luominen tapahtuu Depug-painiketta klikkaamalla tai painamalla F5-näppäintä. Näin ollen syntyy painike Tekla Structuresiin, jonka avulla voidaan luoda palkki malliin. Kun Luo palkki -painiketta on klikattu, annetaan palkille aloitus- ja lopetuspiste. Palkin luominen onnistuu, kuten kuvasta 16 voi nähdä.



Kuva 16 Luo palkki -painike ja palkki luotuna (Maunula 2015)

5 TYÖKALUN KEHITTÄMISPROSESSI

5.1 Käyttöliittymä

Windows Forms

- sisältää save & load, apply, get komennot
- tukee valintoja katalogeista (pultit, profiili, jne.)
- tukee etäisyysyksiköitä, double, int ja string komennot
- on tehokkaampi kuin INP.

INP

- samaa formaattia käytetään custom componenteille ja system componenteille
- sisältää save & load, apply ja get komennot
- tukee valintoja katalogeista (pultit, profiili, jne.)
- tukee kaikkia datatyyppejä ja tyyppitarkasteluja.

(Plugins and Dialogs, Tekla –materiaalit)

5.2 Työkalun ohjelmoinnin perusteet

Open API voidaan luoda kolmella eri tapaa:

- Macro
  - tallennetut skriptit
  - versio riippumaton
- Application
  - itsenäisesti suoritettavia sovelluksia
  - suoritetaan Teklan ulkopuolella
  - lisääntynyt joustavuus
- Plug-in
  - suoritetaan Tekla Structuresin sisäpuolella
  - päivittää muutokset automaattisesti

(Plugins and Dialogs, Tekla-materiaalit)

Pluginit eli laajennukset ovat työkaluja joita käytetään automatisoimaan mallissa tai piirustuksissa esiintyviä objekteja. Nämä laajennukset ovat älykkäitä eli ne ovat muutettavissa ja ovat riippuvaisia niihin syötetyistä osista. Pluginit on ladattu Tekla Structuresin prosessiin. Mallissa käytettävät pluginit löytyvät Component Catalogista kun taas piirustuksissa käytettävät löytyvät työkaluriviltä.

Tekla Structures mahdollistaa kaksi eri tapaa luoda pluginin. Ensimmäinen on PluginBase-menetelmä, jolla voidaan luoda abstrakti luokka komponenttien työkaluille. PluginFormBase taas käyttää Windows Forms- tyyppiä.

Plugin-tyypit:

- PluginBase
  - kuten yleinen komponentti
    - portaat, tikkaat, ristikot, jne.
  - syöte voidaan määritellä vapaasti
    - mikä tahansa määrä esineitä
    - mikä tahansa määrä pisteitä
  - voi olla riippuvainen tai riippumaton syöte.

- ConnectionBase
  - yksityiskohdat
    - pohjalevy, nostokoukut jne.
    - yksi osa ja yksi piste
  - liitokset
    - jäykiste, hitsaukset, liitokset, jne
    - yksi tai useampi toissijainen osa.

PluginBase perusrakenne:

- Structuresdata = rakenteen data
  - määrittää tiedot, jotka voidaan syöttää käyttöliittymältä
- Attributes = attribuutit
  - määrittele nimi ja käyttöliittymä
- Constructor = muodostin
  - alustaa pluginin
  - vastaanottaa nykyisen StructuresDatan
- DefineInput = määritä syöte
  - määrittää pluginin vaatiman syötteen
- Run = Suorita
  - suorittaa kun syöte on vastaanotettu.

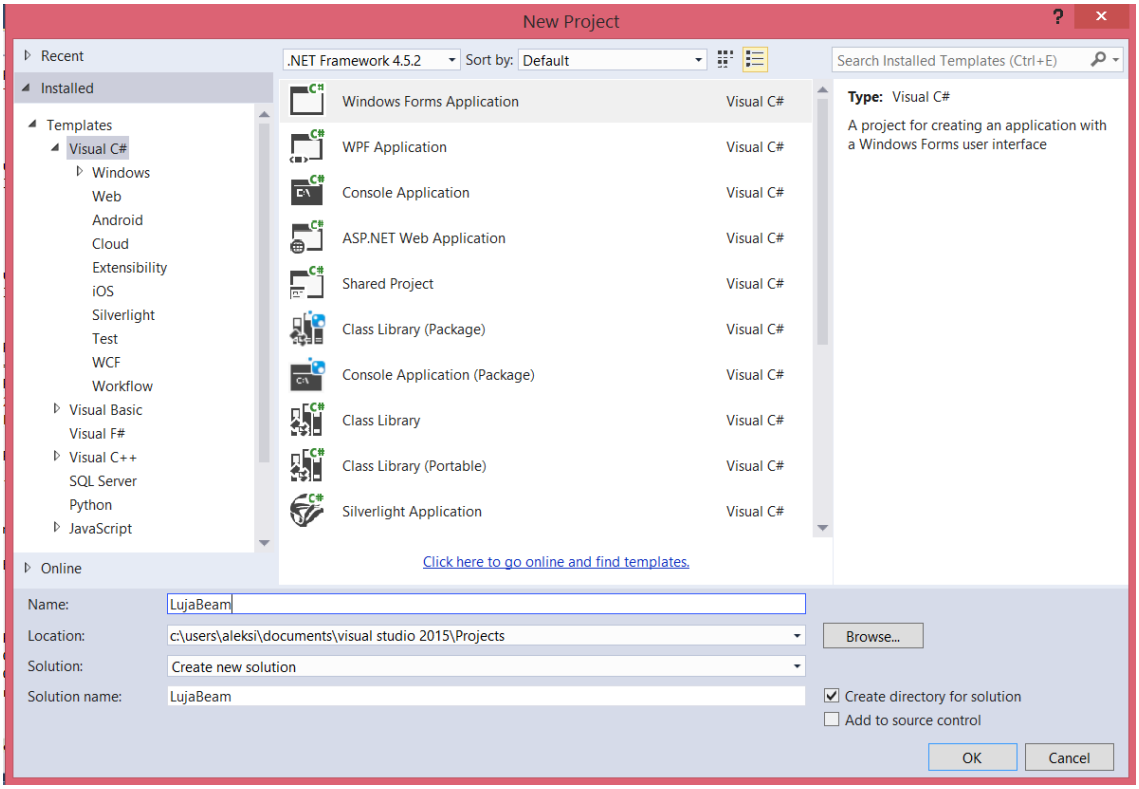
ConnectionBase perusrakenne:

- Structuresdata = rakenteen data

- määrittää tiedot, jotka voidaan syöttää käyttöliittymältä
- Attributes = attribuutit
  - määrittele nimi ja käyttöliittymä, sekundaarien määrä, törmäyksen tyyppi ja auto up -suunta
- Constructor = muodostin
  - alustaa liitoksen
  - vastaanottaa nykyisen StructuresDatan
- DefineInput = määritä syöte
  - määrittää pluginin vaatiman syötteen
- Run = Suorita
  - suorittaa kun syöte on vastaanotettu.

5.3 LB-palkki pluginin kehittäminen

LB-palkin luominen aloitettiin luomalla uusi .NET projekti Visual Studiossa, kuten kuvasta 17 voi nähdä.



Kuva 17 Uuden projektin aloittaminen Visual studiossa (Maunula 2015)

Kun projekti saatiin luotua, lisättiin pluginiin tarvittavat referenssit, joita Tekla Structures käyttää. Nämä referenssit ovat välttämättömiä, jotta vuoropuhelu ja luokkarakenne toimivat oikein Tekla Structuresin kanssa. On suositeltavaa, että referenssejä nimetään uudelleen. Esimerkiksi TSMUI on lyhenne Tekla.Structures.Model.UI referenssistä. Koodiin ei tällöin tarvitse kirjoittaa kirjaston koko nimeä vaan sen lyhenne, joten koodi selkeytyy. Kuvassa 18 on esimerkki, kuinka uudelleennimeäminen onnistuu.

```
using Tekla.Structures;
using Tekla.Structures.Catalogs;
using Tekla.Structures.Datatype;
using TSData = Tekla.Structures.Datatype;
using Tekla.Structures.Dialog;
using Tekla.Structures.Model;
using TSM = Tekla.Structures.Model;
using Tekla.Structures.Model.UI;
using TSMUI = Tekla.Structures.Model.UI;
using Tekla.Structures.Geometry3d;
using T3D = Tekla.Structures.Geometry3d;
using Tekla.Structures.Plugins;
```

Kuva 18 Referenssien lisäys (Maunula 2015)

StructuresData toimii porttina käyttöliittymälle. Attribuuttien nimet käyttöliittymällä tulee täsmätä pluginin sisältämään koodiin, kuten kuvassa 19 on esitetty. Tuetut datan tyypit ovat string, double ja integer.

```
[StructuresField("BeamPrefix")]
public string BeamPrefix;
[StructuresField("BeamSNro")]
public int BeamSNro;
[StructuresField("BeamName")]
public string BeamName;
[StructuresField("BeamProfile")]
public string BeamProfile;
[StructuresField("BeamMaterial")]
public string BeamMaterial;
```

Kuva 19 Attribuuttien sisällyttäminen koodiin (Maunula 2015)

Tässä kohtaa nimetään pluginimme. Palkki nimettiin LujaBeamiksi ja sillä nimellä se löytyy Tekla Structuresin Component Catalogista. Nimen pitää olla ainutlaatuinen, sillä Tekla Structures ei hyväksy toista samannimistä pluginiä. PluginUserInerface-luokkaa käytetään tallentamaan pluginin käyttöliittymän kuvaus järjestelmään.

```
[Plugin("LujaBeam")]
[PluginUserInterface("LujaBeam.LujaBeamForm")]
```

Kuva 20 Pluginin nimeäminen (Maunula 2015)

Kuvasta 21 voi nähdä kuinka plugin attribuuteille luodaan yhteys StructuresData:n ja Tekla Structures:n välille.

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

Kuva 21 Yhteyden luominen StructuresDatan ja Tekla Structuresin välille (Maunula 2015)

Define input on julkinen metodi ja se tarjoaa yhden tavan tehdä pluginin. Tämä metodi määrittää pluginin syötteet. Piirustuksen pluginit tukevat useampia syötteitä, mutta myöskin toimivat ilman. Syötteet annetaan Run- metodin kautta InputDefinition-listalle. InputDefinitionFactory luo ja hakee syötteet. Ne voivat olla pisteitä tai piirustusobjekteja. Metodi suoritetaan kun käyttäjä on painanut pluginin kuvaketta. Opinnäytetyössä on käytetty InputDefinition:a määrittämään palkille alku- ja loppupiste, kuten kuvasta 22 voi nähdä. Työkalu käynnistyy kun käyttäjä on painanut LujaBeam-kuvaketta Component Catalogista.

```
public override List<InputDefinition> DefineInput()
{

    //User input here
    Picker picker = new Picker();
    List<InputDefinition> PointList = new List<InputDefinition>();

    Point P1 = picker.PickPoint("Pick startpoint for LujaBeam");
    Point P2 = picker.PickPoint("Pick endpoint for LujaBeam");
    InputDefinition Input1 = new InputDefinition(P1);
    InputDefinition Input2 = new InputDefinition(P2);
    PointList.Add(Input1);
    PointList.Add(Input2);

    return PointList;
}
```

Kuva 22 Syötteiden määritys (Maunula 2015)

Päämetodi suoritetaan kun kun on saatu määritettyä syötteet DefineInputissa. Tätä kutsutaan pluginin päämetodiksi ja sen tulisi sisältää kaikki todellinen toteutus. Kuvassa 23 on esitetty pluginin päämetodi.

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

    try
    {
        GetValuesFromDialog();

        Point StartPoint1 = (Point)(Input[0]).GetInput();
        Point EndPoint1 = (Point)(Input[1]).GetInput();

        Keskipalkki

        Oikea palkki

        Vasen palkki

    }
    catch (Exception exc)
    {
        MessageBox.Show(exc.ToString());
    }
    return result;
}
```

Kuva 23 Päämetodi (Maunula 2015)

5.3.1 Palkin ohjelmointi

Ensiksi määritetään ohjelma luomaan uusi palkki avainsanalla ”new”. Sitten asetetaan uudelle palkille sen asetukset kuvan 24 esittämällä tavalla. Palkin aloitus- ja lopetuspiste määräytyy käyttäjän valitsemien pisteiden välille. Käyttäjä määrittää sitten palkille etuliitteen, numeroinnin, nimen, materiaalin ja luokan. Palkki luodaan keskeisesti malliin kahden pisteen välille. Koodissa olevassa sijaintikohdassa määritetään miten päin palkki luodaan malliin.

```
//Keskipalkki

Beam LujaBeam = new Beam();
LujaBeam.StartPoint = StartPoint1;
LujaBeam.EndPoint = EndPoint1;
LujaBeam.PartNumber.Prefix = _BeamPrefix;
LujaBeam.PartNumber.StartNumber = _BeamSNro;
LujaBeam.AssemblyNumber.Prefix = _BeamPrefix;
LujaBeam.AssemblyNumber.StartNumber = _BeamSNro;
LujaBeam.Name = _BeamName;
LujaBeam.Profile.ProfileString = "VAULT" + _BeamHeight + "*" + _BeamWidth + "-" + _BeamChamferH + "*" + _BeamChamferV;
LujaBeam.Material.MaterialString = _BeamMaterial;
LujaBeam.Class = _BeamClass;
LujaBeam.Position.Depth = Position.DepthEnum.FRONT;
LujaBeam.Position.Plane = Position.PlaneEnum.MIDDLE;
LujaBeam.Position.Rotation = Position.RotationEnum.TOP;
LujaBeam.Insert();
```

Kuva 24 Palkin määrittäminen (Maunula 2015

Palkin reunaan tuleva kulmarauta luotiin samalla periaatteella kuin palkki. Kulmarauta saatiin oikeaan sijaintiin Offset- käskyllä, jonka arvoksi annettiin puolet palkin leveydestä. Kulmarauta lisätään osaksi palkin kokoonpanoa Assembly-käskyllä. Kuvassa 25 on esitetty kulmaraudan kutsuminen koodissa.

```
Beam SteelPlate1 = new Beam(P1, P2);
SteelPlate1.StartPointOffset.Dz = (_BeamWidth / 2);
SteelPlate1.EndPointOffset.Dz = (_BeamWidth / 2);
SteelPlate1.Name = _Plate1Name;
SteelPlate1.Profile.ProfileString = "L" + _Ledge + "*" + _Ledge + "*" + _Plate1Thic;
SteelPlate1.Material.MaterialString = _Plate1Material;
SteelPlate1.Class = _Plate1Class;
SteelPlate1.Position.Depth = Position.DepthEnum.FRONT;
SteelPlate1.Position.Plane = Position.PlaneEnum.LEFT;
SteelPlate1.Position.Rotation = Position.RotationEnum.FRONT;
SteelPlate1.Insert();
Assembly A = LujaBeam.GetAssembly();
A.Add(SteelPlate1.GetAssembly());
A.Modify();
```

Kuva 25 Teräslaipan määrittäminen (Maunula 2015)

GetValuesFromDialog()-metodi hakee rasian luonnissa käytettävät attribuutit käyttöliittymältä ja asettaa niille oletusarvot, mikäli käyttäjän arvoja ei ole saatavilla. Käyttöliittymälle syötetyt attribuutit haetaan GetValuesFromDialog()- käskyllä kuvan 26 esittämällä tavalla. Mikäli käyttäjä ei anna dialogille mitään attribuutteja, voidaan sille asettaa jotkin oletusarvot IsDefaultValue- käskyllä. Esimerkkinä koodissa on palkin nimi, jonka oletusarvo on LB.

```
private void GetValuesFromDialog()
{

    //LujaBeam
    _BeamPrefix = Data.BeamPrefix;
    _BeamSNro = Data.BeamSNro;
    _BeamName = Data.BeamName;
    _BeamProfile = Data.BeamProfile;
    _BeamMaterial = Data.BeamMaterial;
    _BeamClass = Data.BeamClass;

    _BeamHeight = Data.BeamHeight;
    _BeamWidth = Data.BeamWidth;
    _BeamChamferH = Data.BeamChamferH;
    _BeamChamferV = Data.BeamChamferV;


    if (IsDefaultValue(_BeamPrefix))
        _BeamPrefix = "LB";
    if (IsDefaultValue(_BeamSNro))
        _BeamSNro = 1;
    if (IsDefaultValue(_BeamName))
        _BeamName = "LUJABEAM";
```

Kuva 26 Attribuuttien määrittäminen, jos käyttäjä ei niitä valitse (Maunula 2015)

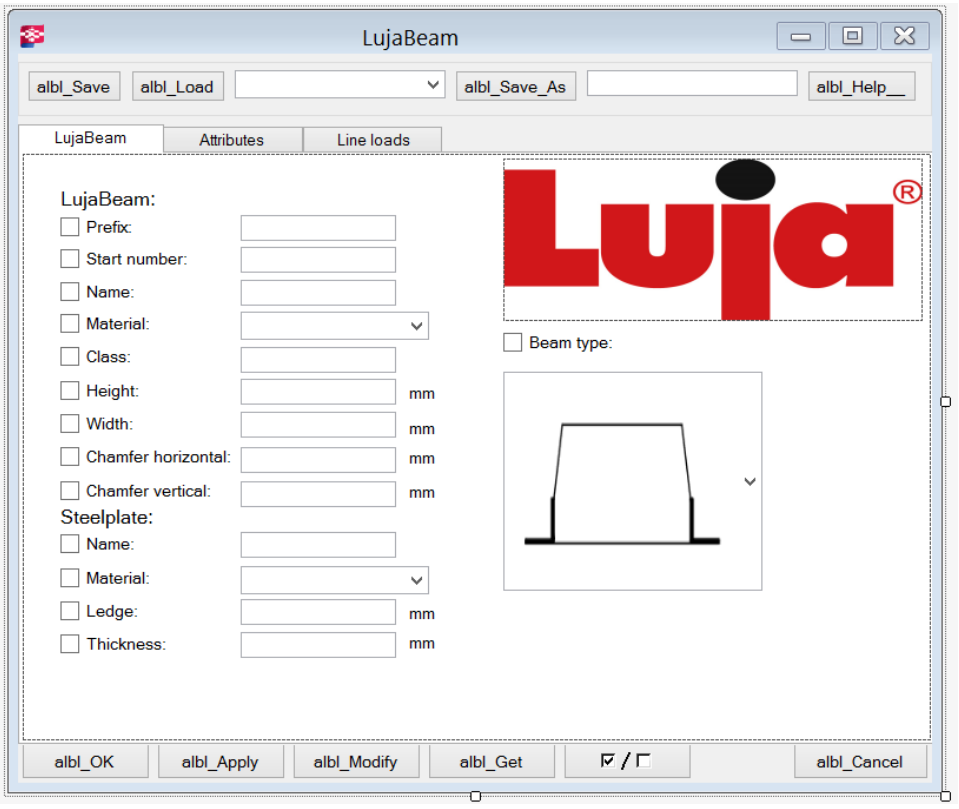
Käyttöliittymän LujaBeam-välilehdellä voidaan määrittää LB-palkin tiedot, kuten kuvasta 27 voi nähdä. Betoniosalle voidaan antaa seuraavat tiedot:

- etuliite
- aloitusnumero
- nimi
- materiaali
- luokka
- korkeus
- leveys
- horisontaali viiste
- vertikaali viiste.

Teräsosalle voidaan syöttää:

- nimi
- materiaali
- leuka
- paksuus.

LB-palkkia on kahta erilaista tyyppiä: keskipalkki ja reunapalkki. Keskipalkki sisältää kulmaraudat palkin molemmin puolin ja reunapalkki vain toispuoleisesti. Päämetodissa palkille on määritetty kolme muuttujaa: keskipalkki, reunapalkki oikealle ja reunapalkki vasemmalle. Palkin tyyppi valitaan käyttöliittymän ImageListComboBoxilta kohdasta Beam type.

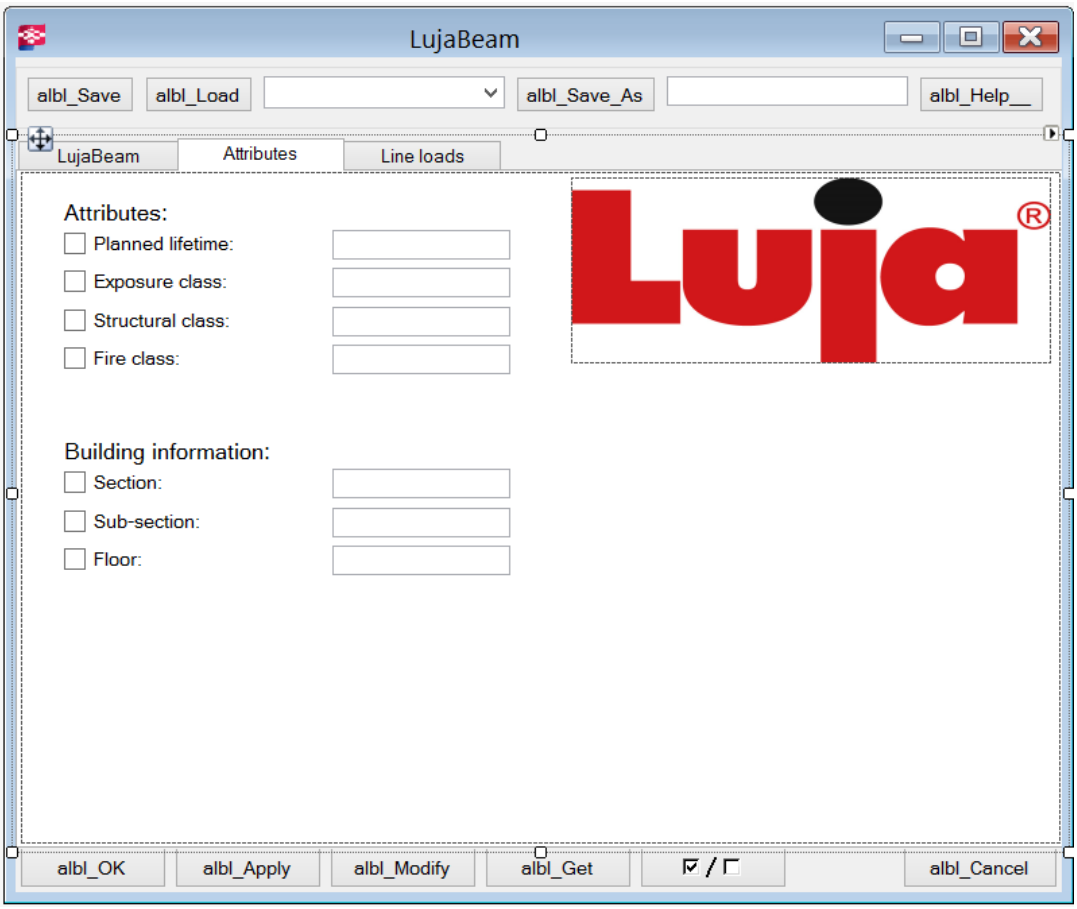


Kuva 27 LB-palkin ensimmäinen välilehti (Maunula 2015)

LB-palkkia on kahta erilaista tyyppiä: keskipalkki ja reunapalkki. Keskipalkki sisältää kulmaraudat palkin molemmin puolin ja reunapalkki vain toispuoleisesti. Päämetodissa palkille on määritetty kolme muuttujaa: keskipalkki, reunapalkki oikealle ja reunapalkki vasemmalle. Palkin tyyppi valitaan käyttöliittymän ImageListComboBoxilta kohdasta Beam type.

Käyttöliittymän Attributes-välilehdellä, joka on esitetty kuvassa 28, voidaan määrittää attribuutteja ja rakennuskohteen tietoja palkille kuten:

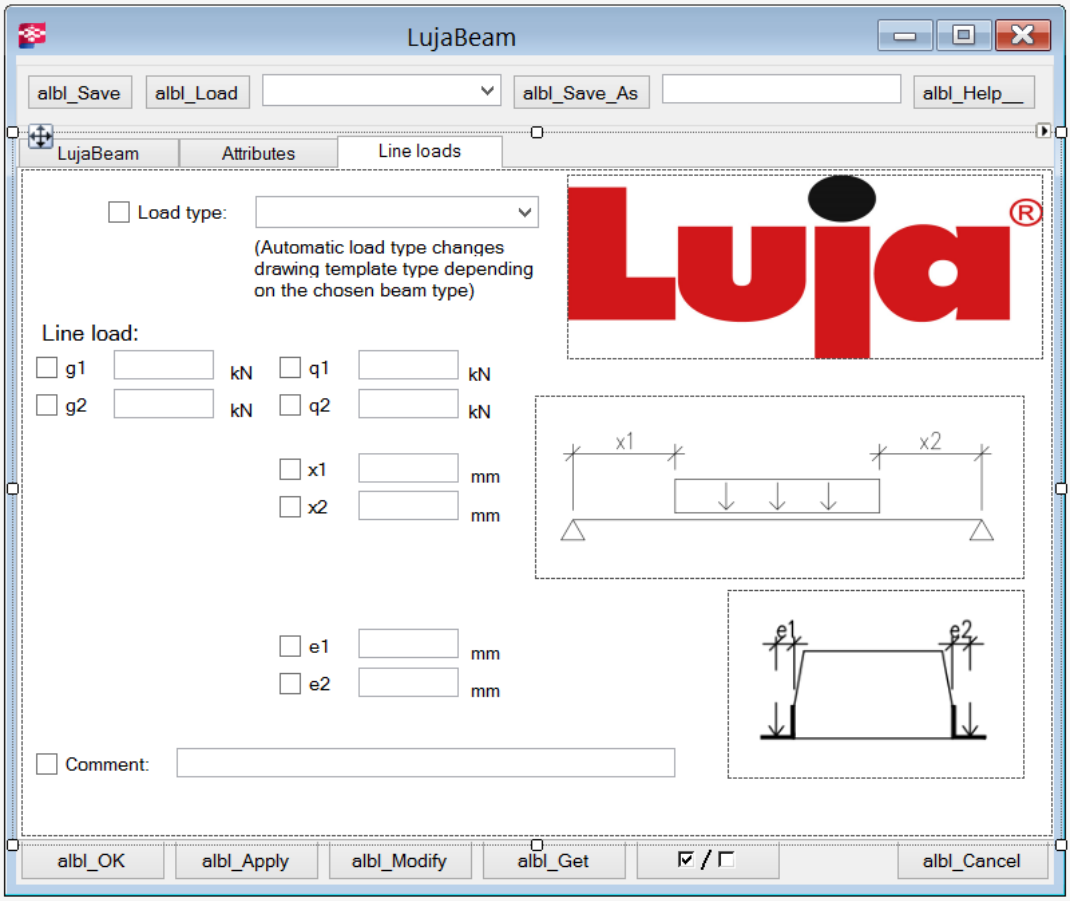
- käyttöikä
- rasitusluokka
- rakenneluokka
- paloluokka
- rakennusosa
- alaosasto
- kerros.



Kuva 28 LB-palkin Attributes-välilehti (Maunula 2015)

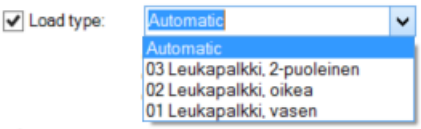
Käyttöliittymän Line loads -välilehdellä, kuten kuvassa 29 on esitetty, voidaan määrittää palkille tulevia kuormia ja etäisyyksiä:

- g1
- g2
- q1
- q2.
- x1
- x2
- e1
- e2.



Kuva 29 LB-palkin Line Loads -välilehti (Maunula 2015)

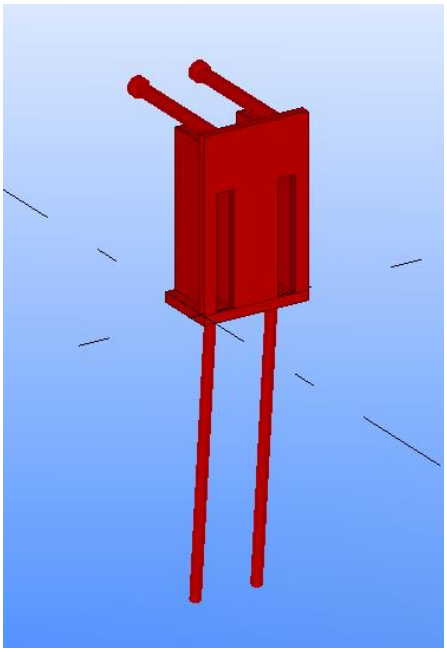
Load typessä valitaan palkin tyyppi, jota käytetään kuormien templaten määrittämisessä. Kuvassa 30 on esitetty kuormatyypin valinta.



Kuva 30 Kuormavaihtoehdot (Maunula 2015)

5.4 LK-piilokonsoliliitos pluginin kehittäminen

LK–piilokonsolin kehittäminen alkoi siitä, että luotiin liitoksessa tarvittavat osat Tekla Structuresissa. Kun liitokset saatiin luotua, tehtiin niistä Custom Componentteja. Näitä Custom Componentteja kutsutaan sitten koodissa. Kuvassa 31 on LK-piilokonsoliliitoksen pilariosa mallinnettuna.



Kuva 31 Custom Component valmiina (Maunula,2015)

Suurin eroavaisuus koodissa on siinä, että LB-palkki luotiin kahden pisteen välille, mutta liitos piti mallintaa pilarin ja palkin välille. Tästä syystä liitos pitää asettaa kahden eri koordinaatiston mukaan. Tästä aiheutuu ongelma että liitokset syntyivät monesti väärään päähän palkkia. Asia saatiin korjattua luomalla laskentasääntö koodiin, jonka mukaan valitaan palkin oikea pää. Laskentasääntö toteutettiin muuttamalla annetun pilarin aloituspiste palkin koordinaatistoon. Näin saatiin liitokselle luotua sääntö, jonka mukaan se mallintuu oikeaan päähän palkkia. Kuvassa 32 on esitetty koordinaatiston muutos.

```
//Tässä asetetaan koordinaatisto palkin mukaan
TransformationPlane originalTransformationPlane = _Model.GetWorkPlaneHandler().
    GetCurrentTransformationPlane();
_Model.GetWorkPlaneHandler().SetCurrentTransformationPlane(new TransformationPlane
    (Palkki.GetCoordinateSystem()));

//Tässä asetetaan matriisi muuttamaan paikallinen koordinaatisto
//aktiiviseen koordinaatistoon
Matrix TransformationMatrix = MatrixFactory.ToCoordinateSystem
    (Palkki.GetCoordinateSystem());
//Tässä muutetaan pilarin piste Piste11 aktiiviseen koordinaatistoon
Point Piste11 = TransformationMatrix.Transform(PilariSP);
```

Kuva 32 Koordinaatiston muutos (Maunula,2015)

LK-piilokonsolin osille piti luoda omat ehdot sille, kumpaan päähän palkkia ne luodaan ja minkä kokoinen liitos on. Seuraavassa kuvassa 34 on kutsuttu LK400P-palkkiosa.

```
if (Pilari != null && Palkki != null)
{
    #region Palkin alkupiste lähempänä
    if ((DistanceSP < DistanceEP) && (_LKKoko == 0))
    {
        CustomPart LK400P = new CustomPart();
        LK400P.Name = "EB_LK400P";
        LK400P.Position.Plane = Position.PlaneEnum.RIGHT;
        LK400P.Position.Rotation = Position.RotationEnum.FRONT;
        LK400P.Position.Depth = Position.DepthEnum.MIDDLE;
        LK400P.SetInputPositions(new Point(MinX, MinY + 151.59 + _LKHeight, MidZ + _LKE),
            new Point(MaxX, MinY + 151.59 + _LKHeight, MidZ + _LKE));
        LK400P.Insert();
        A_PA.Add(LK400P);
```

Kuva 33 Ehdot liitoksen koolle (Maunula 2015)

6 TYÖKALUN TESTAUS

Pluginille on tärkeä suorittaa muutamia testitapauksia ennen käyttöönottoa. Testitapaukset sisältävät kaikki toiminnot jotka käyttäjä voi suorittaa pluginilla. Testaamisella varmistetaan, että työkalu toimii halutun mukaisesti. Aluksi testataan dialogin painikkeet, sitten eri välilehtien painikkeet ja lopuksi kuinka plugin toimii muissa tapauksissa.

6.1 LB-palkin testaus

Tässä osassa on kehitetty testitapaukset LB-palkki työkalulle.

Taulukko 2 Pluginin dialogin painikkeet (Maunula 2015)

Pluginin dialogin painikkeet		
Tehtävä	Tulos	Tulos
Pluginin klikkaaminen Component Catalogissa	Plugin käynnistyy	Odotettu
OK-painikkeen klikkaus	Plugin käynnistyy	Odotettu
Apply-painikkeen klikkaus	Plugin käynnistyy	Odotettu
Modify-painikkeen klikkaus	Plugin käynnistyy	Odotettu
Get-painikkeen klikkaus	Plugin käynnistyy	Odotettu
On/Off-painikkeen klikkaus	On/Off –valinnat päälle ja pois	Odotettu
Cancel-painikkeen klikkaus	Dialogi sulkeutuu, muutoksia ei suoriteta	Odotettu
Save-painikkeen klikkaus	Plugin tallentaa dialogin attribuutit	Odotettu
Load-painikkeen klikkaus	Plugin lataa dialogin attribuutit	Odotettu
Save As-painikkeen klikkaus	Plugin tallentaa dialogin attribuutit uudella nimellä	Odotettu

Taulukko 3 Pluginin input ja suorittaminen (Maunula 2015)

Pluginin input ja suorittaminen		
Tehtävä	Tulos	Tulos
Käyttäjä valitsee LujaBeamin aloitus – ja lopetuspisteen	Palkki syntyi annettujen pisteiden välille	Odotettu
Attribuuttien oikeellisuus	Pluginilla on halutut attribuutit	Odotettu

Taulukko 4 Käyttöliittymän LujaBeam-välilehti (Maunula 2015)

Käyttöliittymän LujaBeam-välilehti		
Tehtävä	Tulos	Tulos
LujaBeamin etuliitteen valinta	Etuliite muuttuu	Odotettu
LujaBeamin aloitusnumeron valinta	Numerointi muuttuu	Odotettu
LujaBeamin nimen valinta	Nimi muuttuu	Odotettu
LujaBeamin materiaalin valinta	Materiaalikatalogi avautuu ja materiaali muuttuu	Odotettu
LujaBeamin luokan valinta	Luokka muuttuu	Odotettu
LujaBeamin korkeuden valinta	Korkeus muuttuu	Odotettu
LujaBeamin leveyden valinta	Leveys muuttuu	Odotettu
LujaBeamin vaakasuoran viisten valinta	Viiste muuttuu	Odotettu
LujaBeamin pystysuoran viisteen valinta	Viiste muuttuu	Odotettu
Kulmaraudan nimen valinta	Nimi muuttuu	Odotettu
Kulmaraudan materi-	Materiaali muuttuu	Odotettu

aalin valinta		
Kulmaraudan leuan valinta	Leuka muuttuu	Odotettu
Kulmaraudan paksuuden valinta	Paksuus muuttuu	Odotettu

Taulukko 5 Käyttöliittymän Attributes-välilehti (Maunula 2015)

Käyttöliittymän Attributes-välilehti		
Tehtävä	Tulos	Tulos
Suunnitellun käyttöiän valinta	Käyttöikä muuttuu	Odotettu
Käyttöluokan valinta	Käyttöluokka muuttuu	Odotettu
Rakenneluokan valinta	Rakenneluokka muuttuu	Odotettu
Paloluokan valinta	Paloluokka muuttuu	Odotettu
Rakennuksen lohkon valinta	Lohko muuttuu	Odotettu
Rakennuksen alilohkon valinta	Alilohko muuttuu	Odotettu
Rakennuksen kerroksen valinta	Kerros muuttuu	Odotettu

Taulukko 6 Käyttöliittymän Line Loads -välilehti (Maunula 2015)

Käyttöliittymän Line Loads -välilehti		
Tehtävä	Tulos	Tulos
Omapaino g1 valinta	Omapaino g1 muuttuu	Odotettu
Omapaino g2 valinta	Omapaino g2 muuttuu	Odotettu
Hyötykuorman q1 valinta	Hyötykuorman q1 muuttuu	Odotettu
Hyötykuorman q2 valinta	Hyötykuorman q2 muuttuu	Odotettu
Etäisyyden x1 valinta	Etäisyys x1 muuttuu	Odotettu

Etäisyyden x2 valinta	Etäisyys x2 muuttuu	Odotettu
Kuorman tyypin valinta	Kuorman tyyppi muuttuu	Odotettu
Kommentin valinta	Kommentti muuttuu	Odotettu

Taulukko 7 Muut testit (Maunula 2015)

Muut testit		
Tehtävä	Tulos	Tulos
Palkin liikuttaminen	Palkki liikkuu	Odotettu
Palkin kopioiminen	Palkki kopioituu	Odotettu

6.2 LK-piilokonsoliliitoksen testaus

Tässä osassa on kehitetty LK–piilokonsoliliitokselle testitapaukset.

Taulukko 8 Pluginin dialogin painikkeet (Maunula 2015)

Pluginin dialogin painikkeet		
Tehtävä	Tulos	Tulos
Pluginin klikkaaminen Component Catalogissa	Plugin käynnistyy	Odotettu
OK-painikkeen klikkaus	Plugin käynnistyy	Odotettu
Apply-painikkeen klikkaus	Plugin käynnistyy	Odotettu
Modify-painikkeen klikkaus	Plugin käynnistyy	Odotettu
Get-painikkeen klikkaus	Plugin käynnistyy	Odotettu
On/Off-painikkeen klikkaus	On/Off-valinnat päälle ja pois	Odotettu

Cancel-painikkeen klikkaus	Dialogi sulkeutuu, muutoksia ei suoriteta	Odotettu
Save-painikkeen klikkaus	Plugin tallentaa dialogin attribuutit	Odotettu
Load-painikkeen klikkaus	Plugin lataa dialogin attribuutit	Odotettu
Save As-painikkeen klikkaus	Plugin tallentaa dialogin attribuutit uudella nimellä	Odotettu

Taulukko 9 Pluginin input ja suorittaminen (Maunula 2015)

Pluginin input ja suorittaminen		
Tehtävä	Tulos	Tulos
Käyttäjä valitsee pilarin ja palkin	Liitos syntyi valitun pilarin ja palkin välille	Odotettu

Taulukko 10 Käyttöliittymän Attributes-välilehti (Maunula 2015)

Käyttöliittymän Attributes-välilehti		
Tehtävä	Tulos	Tulos
Nimen valinta	Käyttöikä muuttuu	Odotettu
Liitoksen koon valinta	Liitoksen koko muuttuu	Odotettu

Taulukko 11 Käyttöliittymän Parameters 1 -välilehti (Maunula 2015)

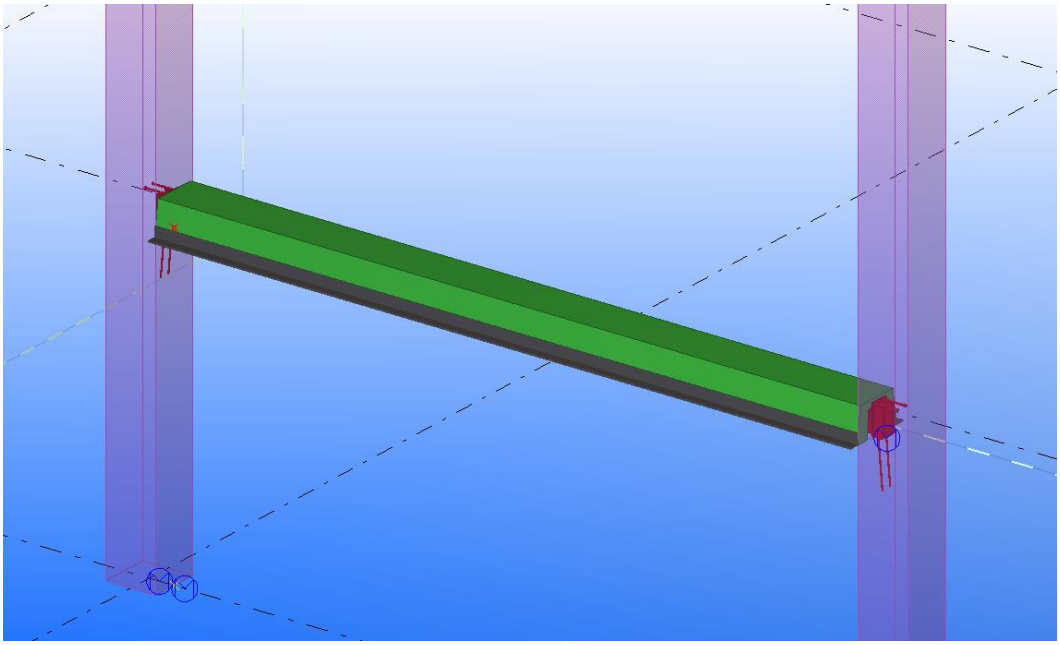
Käyttöliittymän Parameters 1 -välilehti		
Tehtävä	Tulos	Tulos
Liitoksen osien välinen etäisyys	Etäisyys muuttuu	Odotettu
Liitoksen etäisyys palkin alareunasta	Etäisyys muuttuu	Odotettu
Liitoksen epäkeskisyyden valinta	Epäkeskisyys muuttuu	Odotettu

Taulukko 12 Käyttöliittymän Parameters 2 -välilehti (Maunula 2015)

Käyttöliittymän Parameters 2 -välilehti		
Tehtävä	Tulos	Tulos
Kolon korkeus valinta	Korkeus muuttuu	Odotettu
Kolon syvyys valinta	Syvyys muuttuu	Odotettu
Kolon leveys valinta	Leveys muuttuu	Odotettu
Kolon epäkeskisyys valinta	Epäkeskisyys muuttuu	Odotettu

6.3 Kehitettyjen työkalujen yhteensopivuuden testaus

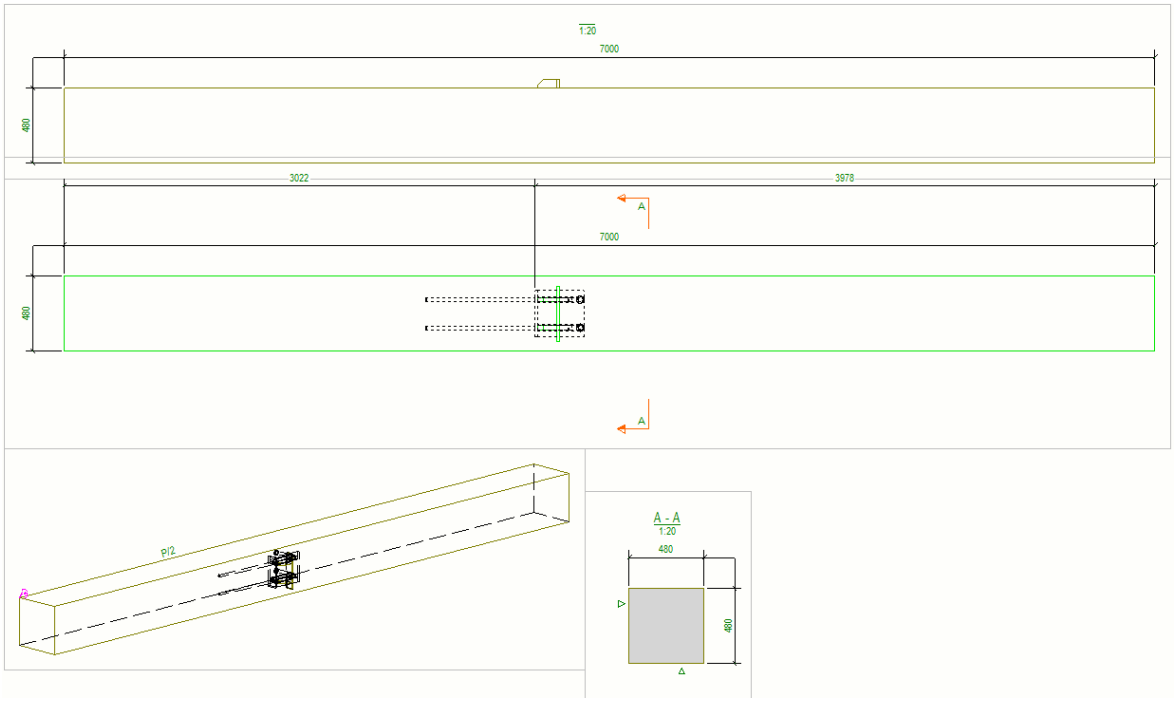
Seuraavaksi testataan vielä kuinka kehitetyt työkalut toimivat yhteistyössä Tekla Structuresissa. Aloitetaan käynnistämällä Tekla Structures, sitten mallinnetaan LB-palkki ja pilarit palkin molempiin päihin. Kun palkki ja pilarit on mallinnettu, lisätään LK-konsoliliitos. Tilanne on kuvan 34 mukainen.



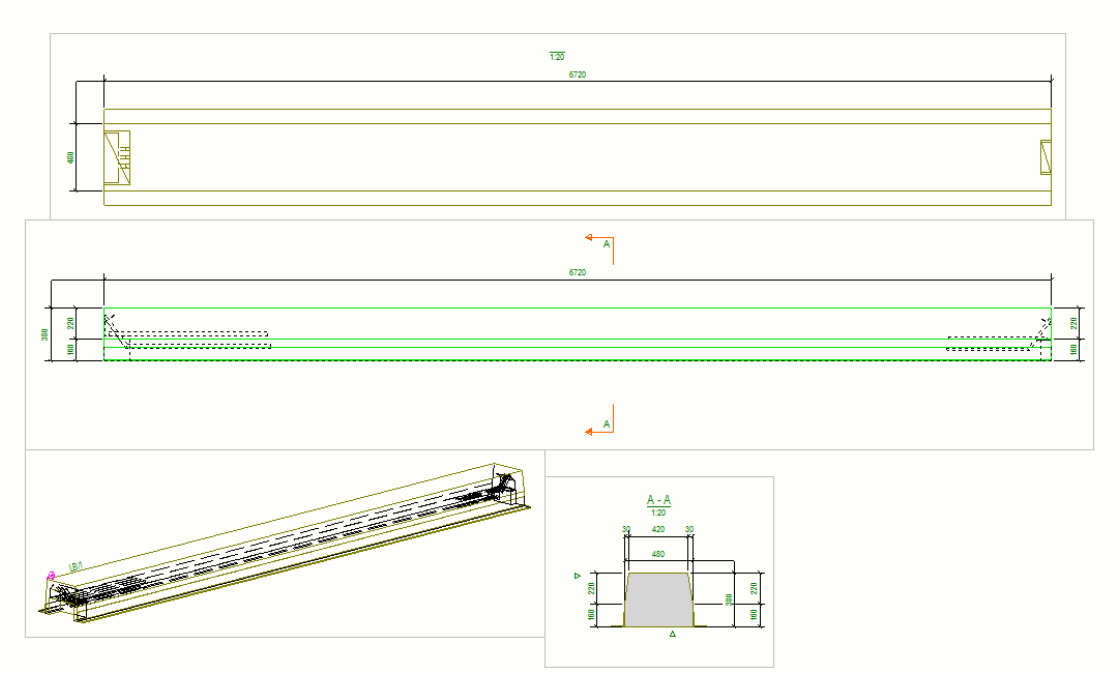
Kuva 34 Pilari ja palkki mallinnettuna (Maunula 2015)

Muutetaan palkin ja liitoksen kokoa ja huomataan että työkalut toimivat halutun mukaisesti. Seuraavaksi siirrytään tarkistamaan, että objektit näkyvät oikein myös tulostusnäkymässä. Palkista ja pilarista luodaan omat piirustukset. LK-liitoksesta tulisi näkyä pilarin piirustuksessa liitos- ja pilariosa ja palkin piirustuksessa vain palkkiosa.

Kuten kuvasta 35 voidaan huomata, pilariosa ja liitososa näkyvät pilarin piirustuksessa. Kuvasta 36 taas huomataan, että palkkiosa ja kolot näkyvät palkin piirustuksessa. Yhteenvetona voidaan sanoa, että opinnäytetyössä kehitetyissä työkaluissa on tarvittavat toiminnallisuudet ja ne toimivat oikein.



Kuva 35 Pilarin mittakuva (Maunula 2015)



Kuva 36 LB-palkin mittakuva (Maunula 2015)

7 KÄYTTÖÖNOTTO

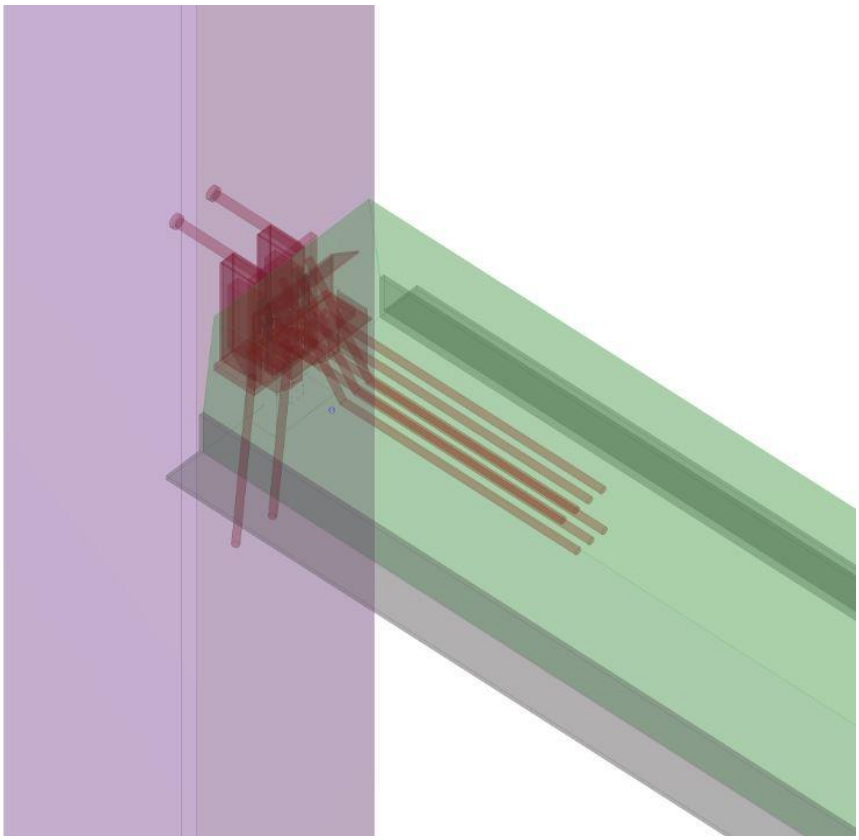
7.1 Liite 1. LB-palkin käyttöönotto- ja mallinnusohje

7.2 Liite 2. LK-konsoliliitoksen käyttöönotto- ja mallinnusohje

8 TULOKSET

Kehitystyön tuloksena syntyivät toimivat mallinnustyökalut Lujabetonin valmistamalle LB-palkille ja LK-piilokonsoliitokselle. Mallinnustyökalut kehitettiin mahdollisimman yksinkertaisena, jotta niiden käyttö ei vaikeudu liikaa. LB-palkin mallinnustyökalun avulla käyttäjä voi mallintaa LB-palkin haluamillaan tiedoilla ja muuttamaan niitä vaivatta. LK-piilokonsoliliitoksen mallinnustyökalu taas mahdollistaa liitoksen nopean luomisen käyttäjän valitsemien palkkien ja pilarien välille. Testaustapausten mukaan molemmat mallinnustyökalut toimivat oikein, joten ne pitäisi saada käyttöön nopeasti.

Tekla tarjoaa käyttäjille pilvipalvelun Tekla Warehousen, josta käyttäjät pystyvät lataamaan toisten käyttäjien ja valmistajien kehittämiä työkaluja. LujaBetonin tavoitteena on luoda kaikista valmistamistaan tuotteista mallinnustyökalut Tekla Warehouseen, jotta rakennesuunnittelijoiden työ helpottuu. Opinnäytetyössä kehitetty LB-palkki ja LK-piilokonsoliliitos on mahdollista ladata Tekla Warehousesta tulevaisuudessa.



Kuva 37 Kuvassa mallinnettu LB-palkki ja LK-liitos

9 POHDINTA

Opinnäytetyö päätettiin kehittää Tekla Structuresin tarjoamalla Open API -rajapinnalla. Opinnäytetyö osoittautui varsin haastavaksi, koska sovellus vaati ohjelmointitaitoa, jota itselläni ei ollut entuudestaan. Ohjelmoinnin osaamattomuuden vuoksi opinnäytetyöprosessi alkoi ohjelmoinnin opettelulla. Tekla tarjoaa aloituspaketin, josta löytyy erinäisiä harjoitustöitä Open API:iin liittyen. Harjoitustöillä pääsee hyvin alkuun, mutta itse olisin toivonut kattavampaa tarjontaa niiden osalta. Samasta aiheesta opinnäytetyön tehnyt rakennusinsinööri Jussi Junkkarinen onneksi tarjosi apuaan ja työkalut saatiin tehtyä.

Opinnäytetyössä kehitettiin mallinnustyökalut Lujabetonin valmistamalle LB-palkille ja LK-piilokonsoliliitokselle. Tuloksena kehittyivät toimivat parametrisoidut komponentit, joilla on tarvittavat ominaisuudet tietomallintamiseen. Kehitetyillä työkaluilla pyritään suunnittelun nopeuttamiseen ja tietomallintamisen kehittämiseen. Kehitetyt työkalut ovat kuitenkin vain pieni osa tietomallintamista ja Lujabetonin tuotteita. Kehitystyötä on siis Lujabetonilla vielä edessä, jotta he pystyvät tarjoamaan tarvittavia tietomallinnuskomponentteja. Lujabetoni on muita kilpailijoita jäljessä mallinnustyökalujen tarjoamisessa Tekla Structuresissa. Monet kilpailijat tarjoavat jo mallinnustyökaluja liittopalkeille ja konsoliliitoksille. Tietomallintaminen on tärkeässä osassa tämän päivän rakennesuunnittelua joten on hyvä, että Lujabetoni lähtee myös kehittämään mallinnuskomponentteja tuotteistaan.

Ohjelmointitaidon opettelu vei varsin paljon aikaa ja siksi opinnäytetyöprosessista tuli varsin pitkäkestoinen. Open API -rajapinnan opettelu kuitenkin oli hyödyllistä, koska sillä pystytään valmistamaan varsin monimutkaisia työkaluja Tekla Structures -rakennessuunnitteluohjelmistoon. Moni saattaa kyseenalaistaa rakennusinsinöörin tarpeen opetella ohjelmointia, sillä sitä ei opeteta ainakaan vielä rakennusinsinööreille. Maailma kuitenkin digitalisoituu nopeaa tahtia ja vanhat työskentelytavat menettävät merkitystään. Opinnäytetyössä perehdyttiin vain vähän Open API:n tarjoamiin mahdollisuuksiin. Kehitettyjä mallinnuskomponentteja voitaisiin viedä vielä paljon eteenpäin sisällyttämällä niihin enemmän älykkyyttä. Tarkoituksena onkin jatkaa vielä työskentelyä Open API:n parissa.

LÄHTEET JA TUOTETUT AINEISTOT

Anstar.fi [verkkoaineisto]. [Viitattu 2015-11-24.] Saatavissa:
http://www.anstar.fi/
Polku: Anstar.fi. Tuotteet.

Betonimestarit.fi [verkkoaineisto]. [Viitattu 2015-11-24.] Saatavissa:
http://www.betonimestarit.fi/
Polku: Betonimestarit.fi. Tuotteet.

Elementtisuunnittelu.fi a [verkkoaineisto]. [Viitattu 2015-11-02.] Saatavissa:
http://www.elementtisuunnittelu.fi/fi/suunnitteluprosessi/mallintava-suunnittelu

JUNKKARINEN, Jussi 2014. Tekla Structures sähköasennusten mallinnustyökalun kehittäminen Open API ohjelmointirajapinnalla. Savonia-ammattikorkeakoulu. Rakennustekniikan koulutusohjelma. Opinnäytetyö. [Viitattu 2015-11-03]. Saatavissa:
http://www.theseus.fi/xmlui/browse?value=Junkkarinen%2C+Jussi&type=author

KOKKONEN, Kari 2015-05-20. Lujabetoni projektipäällikkö, elementit. [haastattelu]. Kuopio: Savonia-ammattikorkeakoulu.

Peikko.fi [verkkoaineisto]. [Viitattu 2015-11-24.] Saatavissa:
http://www.peikko.fi
Polku: Peikko.fi. Tuotteet.

Ruukki.fi [verkkoaineisto]. [Viitattu 2015-11-24.] Saatavissa:
http://www.ruukki.fi/
Polku: Betonimestarit.fi. Rakentaminen. Teräsrunkorakenteet. CWQ-palkkijärjestelmä.

SIVONEN, Veli-Matti. Helsingin yliopisto 2004 [seminaariesitelmä]. [Viitattu 2015-11-02.] Saatavissa:
http://www.cs.helsinki.fi/u/pohjalai/k04/ohpe/seminar/Sivonen-CSharp.pdf

Suunnitteluohje LujaBeam-järjestelmä [verkkoaineisto]. [Viitattu 2015-11-02.] Saatavissa:
http://www.lujabetoni.fi/instancedata/prime_product_julkaisu/luja/embeds/lujabetoniwwwstructure/17568_Lujabeam-runkojarjestelma_aineisto.pdf

Tekla Corporation 2013. Developers Guide. Product Version 20.0.

TEKLA 2015a. Tekla Structures tuote-esittely [verkkojulkaisu]. [Viitattu 2015-11-03.] Saatavissa:
http://teklastructures.support.tekla.com/190/en/sys_tekla_open_api

TEKLA 2015b. Tekla Structures tuote-esittely [verkkojulkaisu]. [Viitattu 2015-11-03.] Saatavissa:

http://www.tekla.com/fi/tuotteet/tekla-structures

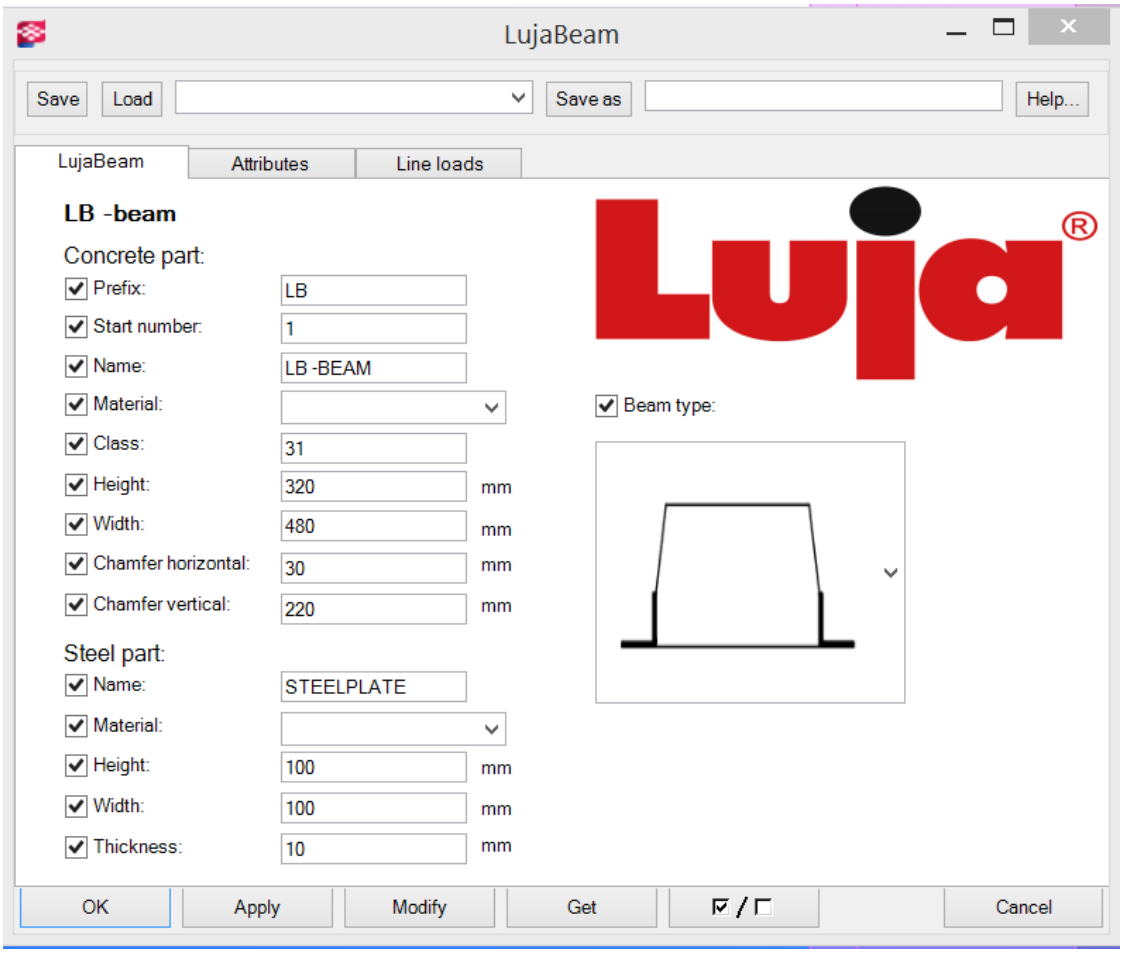
Yleiset tietomallivaatimukset 2012. Osa 5. Rakennesuunnittelu. [online]. [Viitattu 2015-11-03.] Saatavissa:

https://www-rakennustieto-fi.ezproxy.savonia.fi/kortistot/tuotteet/108097.html.stx

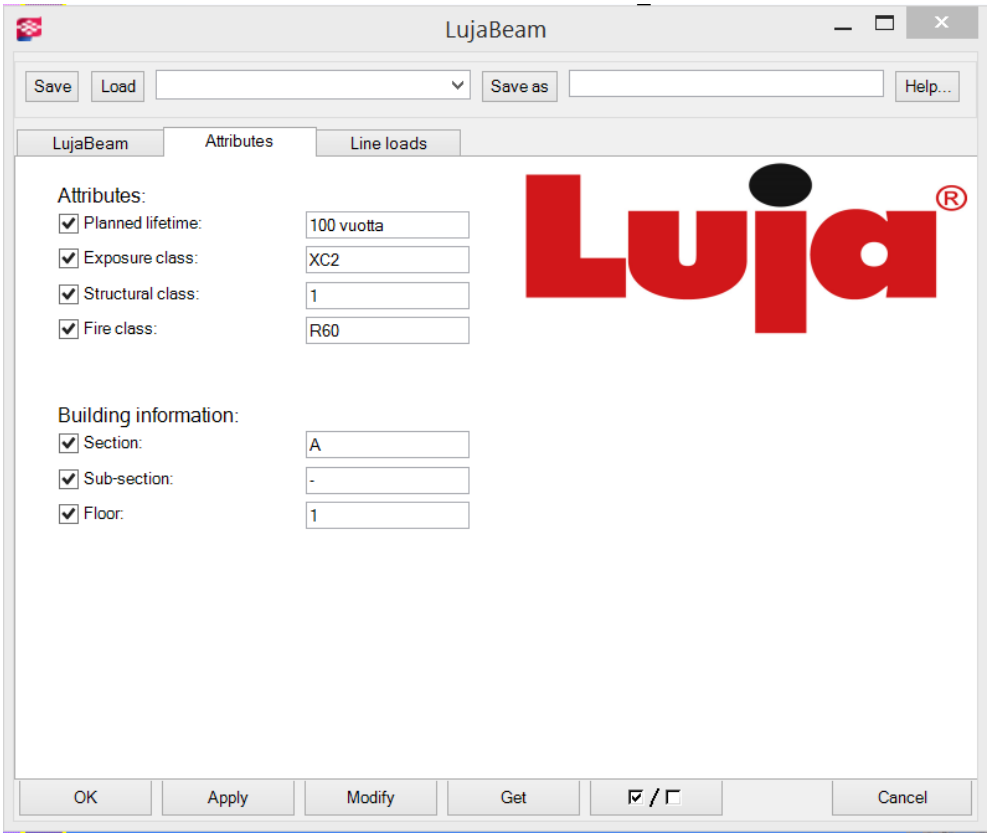
LIITE 1: LB-PALKIN KÄYTTÖÖNOTTO JA MALLINNUSOHJE

- Viedään ladadusta LujaBeam-kansiosta
  C:\Users\aleksi\Desktop\LujaBeam\LujaBeam\bin\Debug LujaBeam.dll tiedosto paikalliseen Tekla -kansioon C:\Program Files (x86)\TeklaStructures\21.0\nt\bin\plugins
- Käynnistä Tekla Structures
- Avaa Component Catalog (Ctrl+F)
- Kirjoita hakukenttään LB–beam
- Tuplaklikkaa LB–beam kuvaketta → työkalu avautuu
- Syötä kenttiin haluamasi tiedot

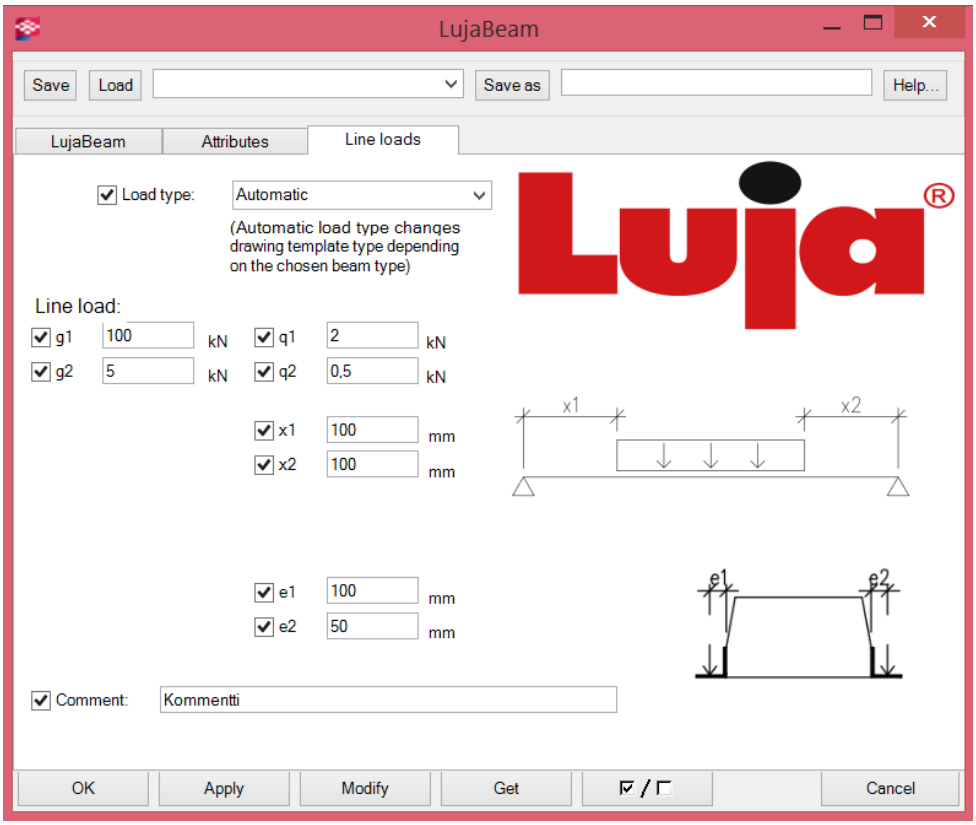
LujaBeam-välilehdellä voit antaa tiedot LB-palkin betoni- ja teräsosille. Beam type -kohdassa voit määrittää onko palkki kaksileukainen vai yksileukainen.



Attributes-välilehdellä voit antaa palkille suunnittelutietoja ja sen sijaintiin perustuvia tietoja.



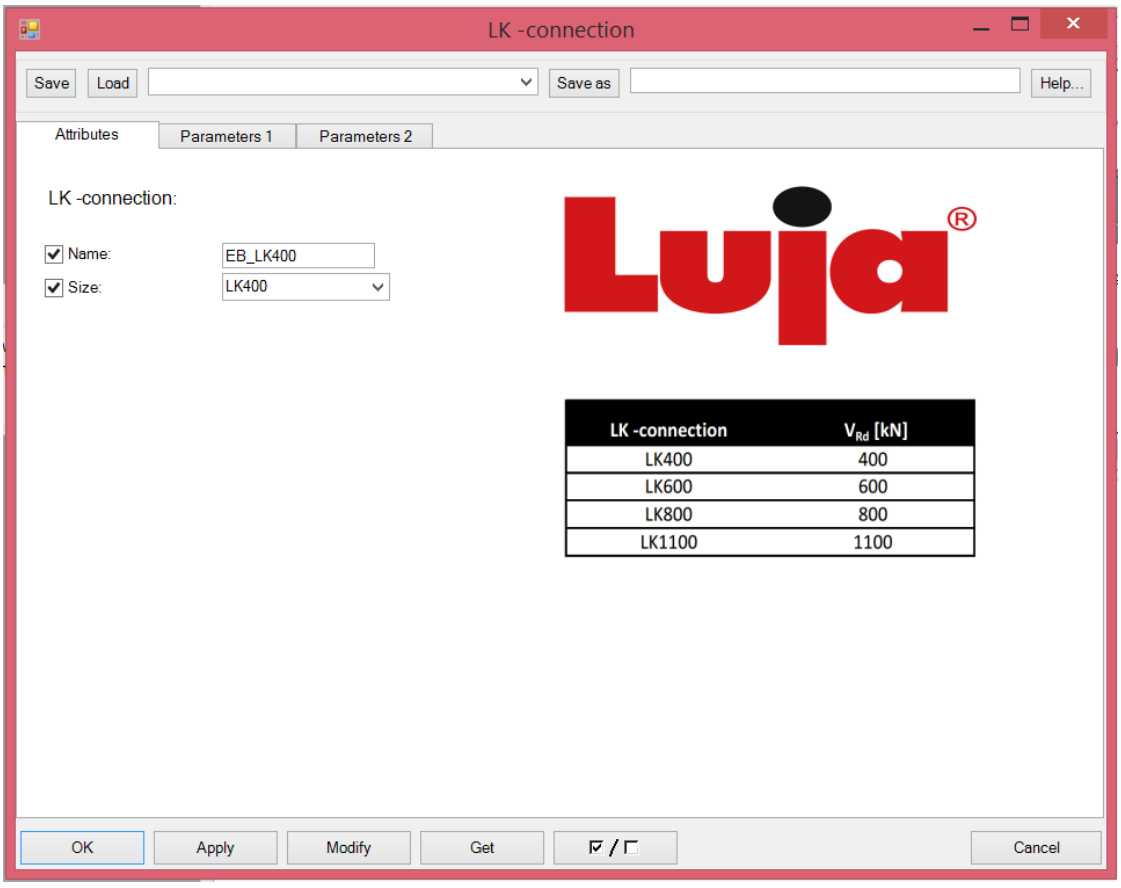
Line loads -välilehdellä palkille syötetään kuormat. Jos kyseessä on tarjouslaskenta laita kuormiksi nolla, jotta ei synny väärinkäsityksiä laskennan suhteen. Ota myös tulostussivulta kuormien template pois.



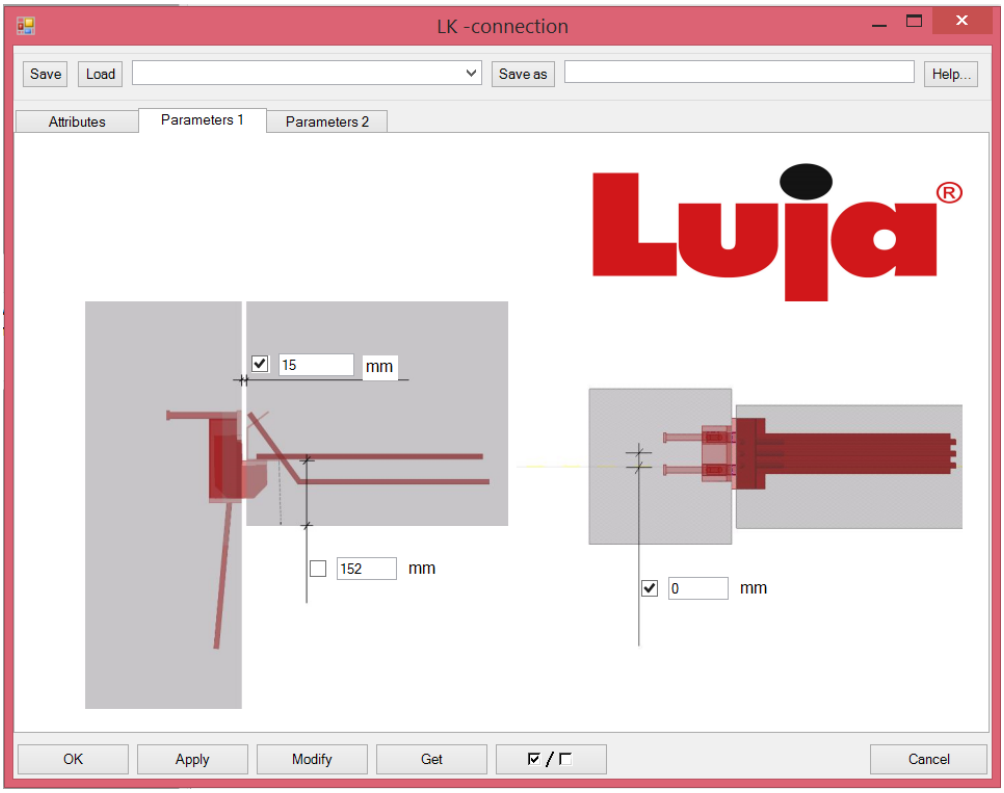
LIITE 2: LK-KONSOLIN KÄYTTÖÖNOTTO JA MALLINNUSOHJE

- Viedään ladadusta Harjoitus-kansiosta C:\Users\aleksi\Documents\Visual Studio 2015\Projects\Harjoitus\Harjoitus\bin\Debug Harjoitus.dll paikalliseen Tekla-kansioon C:\Program Files (x86)\TeklaStructures\21.0\nt\bin\plugins
- Käynnistä Tekla Structures
- Avaa Component Catalog (Ctrl+F)
- Tuplaklikkaa LK -connection kuvaketta → työkalu avautuu
- Syötä kenttiin haluamasi tiedot

Attributes-välilehdellä voit antaa liitokselle nimen ja määrittää liitoksen koon. Taulukosta voit nähdä liitoksien ominaislujuudet ja määritellä sen avulla tarvitsemasi liitoksen vahvuuden.



Parameters 1 -välilehdellä voit sijoitella LK-liitosta haluamallesi kohdalle. Kuvista näkee mittaviivoista, mihin mitat vaikuttavat. Liitos tulee automaattisesti palkin päähän, joten se tulee mallintaa oikein heti alusta. Palkin pään tulee olla 15mm irti pilarista ja keskeisesti palkiin nähden. Tällä dialogilla voit muuttaa pilari- ja liitososan etäisyyttä palkista, liitoksen korkeus ja keskisyys sijaintia. Palkkiosan etäisyys palkin alareunasta tulisi täsmätä Parameters 2-välilehdellä annettavaan reiän korkeuteen.



Parameters 2 -välilehdellä voit määrittää liitososan tarvitseman kolon suuruuden. Taulukosta voit nähdä, mitä arvoja kyseisen liitoksen kololle tulee antaa. Korkeus voi olla suurempikin, mutta sen tulee täsmätä Parameters 1 -välilehdellä annettuun palkkiosan etäisyyteen palkin reunasta.