OULUN SEUDUN
AMMATTIKORKEAKOULU

Antti Aho

TEKLA STRUCTURES -LIITOKSEN KEHITTÄMINEN

TEKLA STRUCTURES -LIITOKSEN KEHITTÄMINEN

Antti Aho
Tekla Structures -liitoksen kehittäminen
Syksy 2013
Tietojenkäsittely
Oulun seudun ammattikorkeakoulu

TIIVISTELMÄ

Oulun seudun ammattikorkeakoulu
Tietojenkäsittely, Ohjelmistokehitys

Tekijä(t): Antti Aho
Opinnäytetyön nimi: Tekla Structures -liitoksen kehittäminen
Työn ohjaaja(t): Anu Niva
Työn valmistumislukukausi ja -vuosi: Syksy 2013 Sivumäärä: *31 + 41 liitesivua*

Tämän opinnäytetyön tavoitteena oli kehittää Tekla Structures -rakennesuunnitteluohjelmistossa toimiva liitoskomponentti. Komponentin toimintatarkoituksena on luoda pohjalevy käyttäjän valitsemaan pilariin. Komponentin kehitystehtävä pitää sisällään koko ohjelmistokehitysprojektin aina vaatimusmäärittelystä valmiiseen tuotteeseen. Raportti sisältää Tekla Structures -rakennesuunnitteluohjelmistoon luodun esimerkkisovelluksen, jotta lukija saa käsityksen Tekla Structures -liitoskomponenttien kehityksestä. Raportti sisältää esimerkin myös käyttöliittymän lähdekoodista.

Opinnäytetyön tietosisällössä keskitytään esittelemään kehitysvaiheessa käytettyjä työkaluja ja menetelmiä. Lisäksi tietosisältö keskittyy esittelemään ohjemistokehitysprojektille ominaisia piirteitä ja käytäntöjä. Raportin tietoperustana on käytetty pääasiassa Tekla Structures -dokumentteja, ohjelmistokehitykseen liittyviä kirjoja sekä internet-lähteitä.

Opinnäytetyön kehitysvaiheen tuloksena syntyi Tekla Structures -rakennesuunnitteluohjelmistossa toimiva liitoskomponentti. Valmistunut liitoskomponentti täyttää sille asetetut vaatimukset. Liitoskomponentti toteutettiin käyttäen Microsoftin C# -ohjelmointikieltä Microsoftin Visual Studio -ohjelmointiympäristössä. Sovelluksen käyttöliittymä kehitettiin käyttäen Teklan omaa Inp-käyttöliittymätyyppiä.

Opinnäytetyön toimeksiantajana toimi Hopia Engineering Oy.

ABSTRACT

Oulu University of Applied Sciences
Business Information Technology, Software development

Author(s): Antti Aho
Title of thesis: Developing a plugin to Tekla Structures
Supervisor(s): Anu Niva
Term and year when the thesis was submitted: Fall 2013 Number of pages: 31 + 41 pages of appendices

---

The aim of this bachelor thesis was to develop a plugin for Tekla Structures -structural design software. The purpose of this plugin is to create a base plate to a user selected column. The plugin was developed following the general steps of a normal development project starting from requirements analysis all the way to a finished product. The report includes an example plugin so the reader can learn the necessary steps needed to develop a Tekla Structures -plugin. The report also includes an example source code from the UI (User Interface).

The content of the thesis focuses on presenting the tools and methods used during the development. The report also focuses on presenting specific features and practices generally used in software development projects. The material of this thesis comes mainly from Tekla Structures -documents, software development related books and internet-sources.

The result of this thesis was a fully functional plugin for Tekla Structures -structural design software. The completed plugin meets the requirements set for it. The plugin was programmed in Microsoft's C# -programming language using Microsoft Visual Studio IDE (Integrated Development Environment). The plugins UI was developed using Tekla's own Inp -user interface type.

The client of this bachelor thesis was Hopia Engineering Oy.

SISÄLLYS

1 JOHDANTO

Hopia Engineering Oy on Oulussa vuonna 2010 perustettu suunnittelutoimisto. Kaikki yrityksen työntekijät ovat kokeneita rakennesuunnittelijoita. Hopia Engineering Oy on hiljattain siirtynyt myös ohjelmistokehitykseen ja tuottaa asiakasyrityksilleen heidän tarpeisiinsa räätälöityjä liitoksia ja komponentteja Tekla Structures -rakennesuunnitteluohjelmistoon. Hopia Engineering Oy:n kolme pääasiallista alaa ovat rakennesuunnittelun palvelut, FEM laskenta ja ohjelmistokehitys.

Tekla Structures on Tekla Corporationin luoma rakennesuunnitteluohjelmisto. Tekla on suomalainen ohjelmistotalo, joka tuottaa kansainvälisille markkinoille mallipohjaiseen olioteknologiaan perustuvia suunnitteluohjelmistoja ja tietojärjestelmiä. Tekla Corporationin asiakaskunta on tarkasti rajattu. Tekla Structures -ohjelmistolla on käyttäjiä noin 80 maassa, joten kyseessä on erittäin laajalle levinnyt ohjelma. Tekla on kehittänyt oman avoimen ohjelmointirajapintansa nimeltä Tekla Open API. Tekla Open API mahdollistaa suunnittelijoiden työtä helpottavien liitosten kehittämisen Tekla Structures -rakennesuunnitteluohjelmistoon. Tällaisia liitoksia voi olla esimerkiksi tässä opinnäytetyössä esiteltävä liitos, joka luo pilarille pohjalevyn automaattisesti sille asetetuilla asetuksilla. Aiheena oleva liitos on kohtalaisen pieni, jos sitä verrataan esimerkiksi kokonaisen rakennuksen luovaan sovellukseen. Asiakaskuntana tämän tyylisellä kehitystyöllä ovat pääasiassa rakennesuunnitteluun keskittyneet yritykset ja konepajat.

Toiminnallisuus tähän liitokseen ohjelmoidaan käyttäen C-sharp (C#) ohjelmointikieltä. Lisäksi sovelluksen help-tiedosto tehdään HTML-kieltä käyttäen ja sovelluksen .inp-tyyppinen käyttöliittymä tehdään skriptillä.

Tietoperustana käytän pääasiassa Teklan dokumentteja ja kirjoja, kuten Macro programming manuaalia, Developers guidea sekä Modeling guidea. Muuna tietoperustana käytän ohjelmistokehitykseen liittyviä kirjoja ja artikkeleita.

2 KEHITTÄMISTEHTÄVÄN KUVAUS

Ohjelmistotuotanto voidaan jakaa osa-alueisiin, jotka tässä tapauksessa ovat määrittely, suunnittelu, ohjelmointi, testaus ja julkaisu. Opinnäytetyöni kattaa koko ohjelmistoprojektin aikataulun laatimisesta valmiin ohjelmiston julkaisuun.

Kehitystehtävänä on luoda Tekla Structures -rakennesuunnitteluohjelmistoon liitoskomponentti, joka luo pilarille pohjalevyn. Liitos käyttää valmiiksi ohjelmistosta löytyvää detaljia nimeltään Base Plate (1004). Käyttäjä määrittelee pohjalevylleen haluamansa ominaisuudet ja asetukset käyttöliittymän kautta.

2.1 Menetelmät

Ohjelmistonkehitysprosessille ja siinä käytettäville menetelmille on rajattomasti vaihtoehtoja. Mikään ratkaisu ei ole yleispätevä ja ratkaisu, joka toimii yhdessä tilanteessa, ei välttämättä toimi toisessa tilanteessa laisinkaan. Tutullakin sovellusalueella sovittuja menettelytapoja voidaan joutua muuttamaan jopa projektikohtaisesti. (Haikala & Märijärvi 2004, 22.)

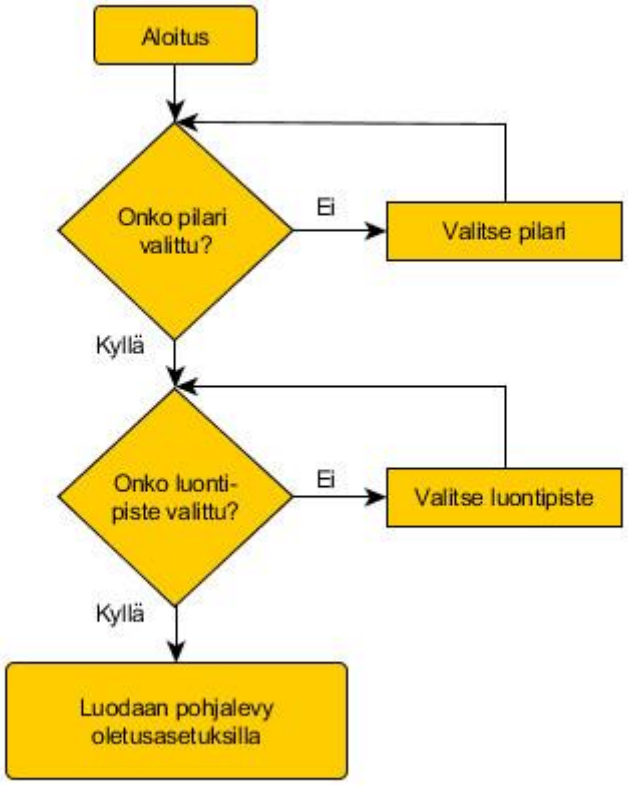
Tämä sovellus kehitetään Tekla Open API -ohjelmointirajapintaan käyttäen C#-ohjelmointikieltä. Tekla Open API sisältää paljon komentoja, jotka ovat hyödyksi ohjelmointitehtävässä. Sovelluksena ohjelmointitehtävässä käytetään MS Visual Studio 2010 ohjelmankehitysympäristöä (IDE). Sovelluksen .chm-päätteinen help-tiedosto tehdään HTML-kielellä. Sovelluksen käyttöliittymä luodaan Teklan Structuresin oman Inp-tiedostotyypin avulla.

2.2 Vuokaavio

Vuokaavioita on käytetty pitkään, jopa niin pitkään ettei ketään ole nimetty vuokaavion isäksi tai keksijäksi. Ohjelmointia käsittelevissä vuokaavioissa ei tarvitse kuin kuutta erilaista symbolia. Suorakaiteen muotoinen symboli kuvaa prosessia. Suorakaiteen sisään kirjoitetaan tietokoneen suorittama tehtävä. Suunnikas kuvaa tietokoneelle syötettävää tietoa tai tietokoneen tuottamaa tietoa (input/output). Salmiakkikuvio kuvaa päätöksen tekoa, joka perustuu ennalta määriteltyihin valintoihin. Suorakaide, jonka alareuna on aaltomainen, kuvaa tiedon varastointia kuten tulostamista. Pieni ympyrä kuvaa yhdistämistä. Ympyrää tarvitaan esimerkiksi vuokaavion tulostamisessa, jos

kaavio ei mahdu yhdelle paperille ja sitä joudutaan jatkamaan seuraavalle paperille. Nuolet kuvaavat suuntia, eli mikä seuraa mitäkin vaihetta. (Sarja, J. 2006).

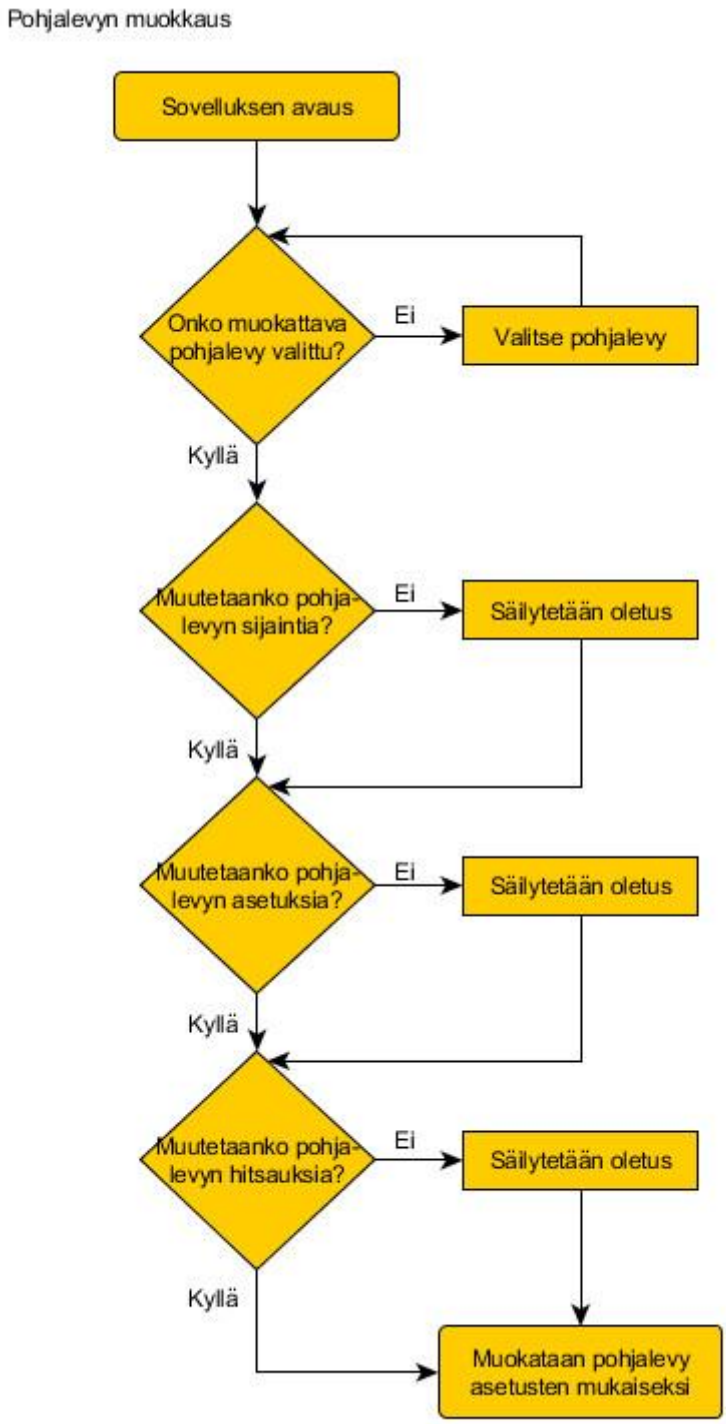
Kuvio 1 esittää vuokaaviona erittäin yksinkertaistetusti pohjalevyn luomisen.



KUVIO 1. Yksinkertaistettu vuokaavio kuvaa opinnäytetyösovelluksen toiminnallisuutta.

Pohjalevylle on kuitenkin mahdollista määrittää asetuksia. Kuvio 2 esittää vuokaaviona käyttäjän mahdollisuudet määritellä pohjalevylle haluamansa asetukset. Käyttäjän on mahdollista muokata seuraavat asetukset:

- Pohjalevyn sijainti ja lisättävä tukirauta
- Pohjalevyn paksuus, leveys, korkeus, materiaali ja luokka
- Pohjalevyn hitsaukset
- Pohjalevyn pulttien asetukset



KUVIO 2. Vuokaavio kuvaa pohjalevyn asetusten muokkaamista.

2.3 Työkalut

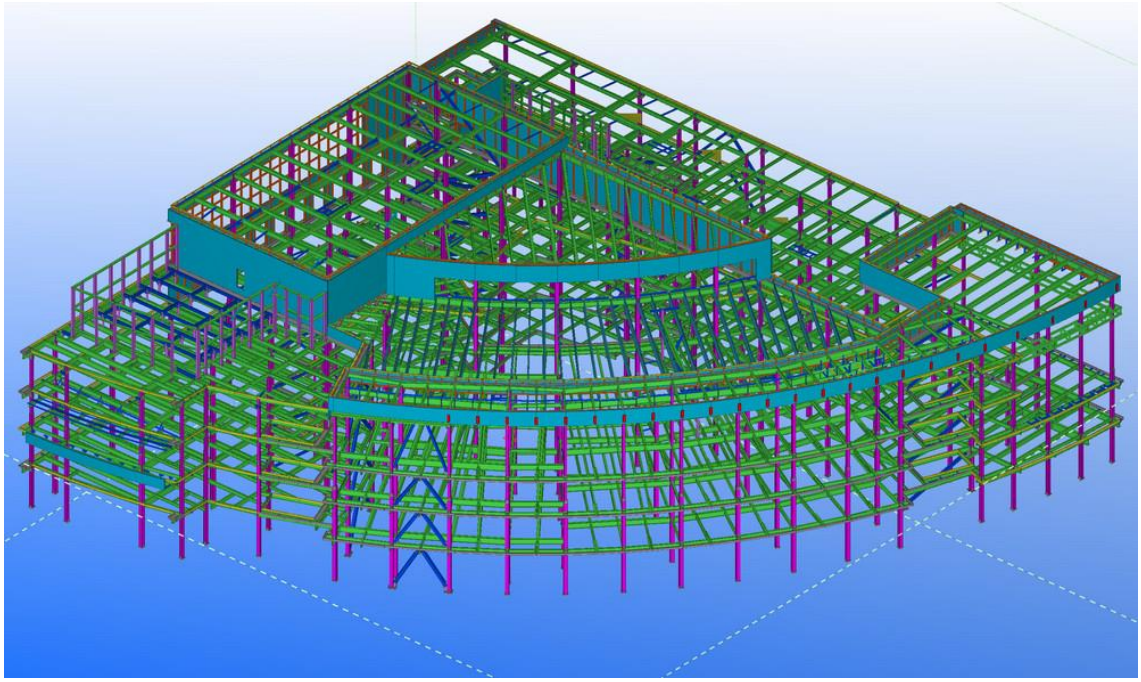
Tässä luvussa esitellään pintapuolisesti opinnäytetyöni kehittämistehtävässä käytössä olleita työkaluja. Ohjelmistokehityksessä on syytä tietää käytettävät työkalut ennen projektin aloittamista, jotta voitaisiin välttyä mahdollisilta yhteensopivuusongelmilta myöhemmässä vaiheessa projektia. Käytettävät työkalut on hyvä olla ohjelmistokehittäjän tiedossa myös ajan säästämiseksi. Tällöin ohjelmistokehittäjän ei tarvitse kesken projektin alkaa etsimään käyttötarkoitukseen soveltuvaa työkalua vaan voi keskittyä olennaiseen.

2.3.1 Kehitysympäristö

Visual Studio -kehitysympäristö on oletusympäristö C#-ohjelmointikielelle. C# ja siihen olennaisesti sidotut kirjastot eivät kuitenkaan ole sidottu pelkästään Windowsiin. Microsoftin virallinen tuki C#:lle kattaa Windows-työpöydän, Xbox 360 -pelikonsolin ja Windows Mobile laitteet. C#-ohjelmointikieli on perustettu Anders Heljsbergin johdolla vuonna 1999, ja se julkaistiin vuonna 2000. C#-ohjelmointikieli kehitettiin yhdistämään C++:n tehokkuus ja Java-kielen helppokäyttöisyys. Microsoft on pyrkinyt saamaan C#:lle virallisen standardoinnin. C#:sta tulikin ISO-standardi vuonna 2003. (Salonen, V. 2011).

2.3.2 Tekla Structures

Tekla Structures on Tekla Corporationin kehittämä tietomallinnuspohjainen rakennesuunnitteluohjelmisto (BIM), jolla on mahdollista luoda ja hallita erittäin suurella tarkkuudella detaljoituja, rakentamisen prosesseja tukevia rakennemalleja. Rakennemallit voivat olla kolmi- ja neliulotteisia. Teklamallia hyödynnetään kaikissa rakennusprosessin eri vaiheissa luonnossuunnittelusta valmistukseen, pystytykseen ja rakentamisen hallintaan. (Tekla Corporation. Tekla BIM). Kuvio 3 esittää kuvanruutukaappauksen Tekla Structures -mallista.



KUVIO 3. Kuvakaappaus Tekla Structures -mallista.

Mallista on mahdollista luoda viiden tyyppisiä piirustuksia:

- Yleispiirustukset (General arrangement drawing), jotka sisältävät näkymiä malliin. Esimerkiksi teräs- ja betonirakenteiden asennuskaaviot ja rakennetyyppien kuvaaminen tehdään yleispiirustuksilla.
- Teräs- ja puurakenteiden kokoonpanopiirustuksia (Assembly drawing), joissa esitetään mallin sisältämien osien liittyminen toisiinsa.
- Teräs- ja puurakenteiden osapiirustukset (Single part drawing), joissa esitetään osan valmistamiseksi tarvittavat tiedot.
- Betonielementtien piirustukset (Cast unit drawing), joissa esitetään elementin valmistamiseksi tarvittava mitta- ja raudoitustietous.
- Yhdistelmäpiirustukset (Multidrawing), jossa on mahdollista esittää enemmän kuin yksi kokoonpanopiirustus yhdellä paperilla. Yhdistelmäpiirustukset yleensä vaativat suuren paperikoon kuten A1. (Tekla Corporation. 2013).

2.3.3 Tekla Open API

Tekla Open API on Tekla Corporationin kehittämä ohjelmointirajapinta Tekla-ohjelmistoja varten. Tekla Open API -ohjelmointirajapinta on avoin rajapinta, joka mahdollistaa käyttäjän omien integroitujen ja itsenäisten sovellusten kehittämisen Tekla Structures -rakennesuunnitteluohjelmiston

pohjalta. Tekla Structures -rakennesuunnitteluohjelmisto sisältää avoimen Tekla Open API -ohjelmointirajapinnan. Rajapinnan avulla kolmannen osapuolen sovellukset voidaan integroida tiedonsiirtoa varten samaan mallinnusympäristöön. Rajapintaa hyödynnetään kaiken tyyppisten sovellusten kehittämisessä Tekla Structures -mallinnusympäristöön. Sovellustyyppejä ovat itsenäisesti suoritettavat sovellukset, Tekla Structures -rakennesuunnitteluohjelmiston toimintaa laajentavat sovellukset ja lisätyökalut.

Erikseen suoritettavia sovelluksia ovat:

- Microsoftin .NET-sovellukset
- COM-tekniikkaa käyttävät COM-sovellukset
- COM-tekniikkaa käyttävät VBA-makrot

Rakennesuunnitteluohjelmiston toimintaa laajentavia sovelluksia ovat:

- Valikkokomennolla suoritettava makrot
- Osaluettelosta käynnistettävät lisäsovellukset

Tekla Open API -ohjelmointirajapinta mahdollistaa:

- Rajapinnan toimintojen suorittamisen ja nauhoittamisen
- Mallinnuksen automatisointiin tarkoitettujen työkalujen luomisen
- Toisen toimittajan tiedostomuodossa olevien tietojen lisäämisen piirustuksiin
- Ohjelmiston integroimisen muihin ohjelmistoihin, kuten toimistotyökaluihin sekä analysointi- ja laskentaohjelmistoihin
- Uusien sovellustoimintojen luomisen
- Uusien tietojen luomisen RFI-hallintatoiminnon avulla

(Tekla Corporation. Avoin ympäristö sovelluskehittäjille).

2.3.4 Microsoft Visual Studio

Tämän opinnäytetyön sovellus on toteutettu käyttäen Microsoftin Visual Studio 2010 Express -ohjelmankehitysympäristöä. Microsoft Visual Studio on Microsoftin ohjelmankehitysympäristö, jossa voi käyttää useita ohjelmointikieliä. Tässä tapauksessa käytettiin C#-kieltä. Lisäksi Visual Studiossa voi käyttää esimerkiksi kieliä kuten Visual Basic, C++, ja J#. Microsoft Visual Studiossa graafisen käyttöliittymän luonti on tehty erityisen helpoksi. Tämän sovelluksen käyttöliittymä luodaan kuitenkin Teklan omaan Inp-käyttöliittymätyyppiin, joka on haastavampaa.

2.3.5 Versionhallinta

Ohjelmistoprojektin sisällön muuttuessa on hyödyllistä olla tallessa myös kehitettävän ohjelmiston vanhemmat versiot. Vanhemman version tiedostoista voi myöhemmässä kehitysvaiheessa olla hyötyä. Tallennetut versiot ovat ikään kuin paluupisteitä, joihin projektin voi palauttaa virheen sattuessa.

Versionhallinnalla tarkoitetaan teknistä ratkaisua, jolla projektien sisältämiin tiedostoihin tehdyt muutokset tallennetaan. Versionhallinnan avulla voidaan myös tarkastella aiemmin tehtyjä tallennuksia. Yleisimmin käyttökohde versionhallinnalle on ohjelmistoprojektien yhteydessä tehtävä versiointi. Murphyn lain mukaisesti versionhallinta on tarpeen juuri silloin, kun kuvitellaan, ettei sitä tarvita.

Tekemässäni projektissa versionhallintajärjestelmänä oli Subversion, lyhennettynä SVN. Muita avoimeen lähdekoodiin perustuvia versionhallintajärjestelmiä ovat RCS, CVS ja Git. Versionhallintajärjestelmien suurimmat erot löytyvät niiden verkko-ominaisuuksista, yhtäaikaisuuden hallinnasta ja operaatioista. SVN versionhallintajärjestelmän suurin tarkoitus on korjata CVS-versionhallintajärjestelmässä todettuja puutteita ja lisätä yleisesti tarpeelliseksi todettuja ominaisuuksia. SVN on siis rakennettu CVS:n pohjalta, minkä johdosta nämä kaksi ovat käytettävyydeltään hyvin samantyylisiä versionhallintajärjestelmiä.

SVN mahdollistaa esimerkiksi tiedostojen ja hakemistojen uudelleennimeämisten ja muun tiedostoihin liittyvän metadatan versioinnin. Tämän ominaisuuden johdosta tiedoston nimen tai attribuuttien muuttaminen tulkitaan samalla tavoin muutokseksi kuin esimerkiksi tiedoston sisällön muuttaminen. (Lenkkeri, J. Aureolis Oy).

3 DOKUMENTOINTI

Tässä kappaleessa esitellään opinnäytetyöprojektiin liittyvät dokumentit, jotka ovat vaatimusmäärittely ja testitapaukset. Vaatimusmäärittely tulee tehdä yhteistyössä asiakkaan kanssa, joka tässä tapauksessa on Hopia Engineering Oy. Vaatimusmäärittely sisältää ohjelmalle asetetut vaatimukset, eli miten valmiin ohjelman tulisi toimia. Esimerkillisiä ohjelmistotuotteelle asetettuja vaatimuksia voi olla esimerkiksi:

- Käyttöliittymätyyppi
- Mitä tietoja käyttäjä voi syöttää ohjelmaan
- Luotavat asennusmediat

Vaatimusmäärittely sijoittuu ohjelmiston kehittämisen alkuvaiheeseen. Vaatimusmäärittelyssä määritellään mitä kehitettävän ohjelmiston tulee tehdä, ja luodaan pohja ohjelmistokehityksen myöhemmille vaiheille. Vaatimusmäärittelyn tarkoituksena on antaa black-box -kuvaus ohjelmalta vaadittavista asioista. Ohjelmisto kuvitellaan ”mustana laatikkona”, kun implementoinnista ei ole tarkkaa tietoa. Vaatimusten keräämiseen liittyy myös niiden priorisointi. Sen lisäksi, että kaikki halutut vaatimukset saadaan erilaisilla tekniikoilla kerätyksi ja määritellyksi, täytyy myös päättää mitkä niistä toteutetaan ja miten ne liittyvät toisiinsa. Riittämätön vaatimusmäärittely on yleisin yksittäinen syy ohjelmistoprojektin epäonnistumiseen.

Systeemityön eri vaiheet ovat vaatimusmärittely, määrittely, suunnittelu, toteutus, testaus ja käyttöönotto. Vaatimusmäärittelyä käytetään usein ohjelmiston tarjouspyynnön liitteenä, josta asiakas voi helposti nähdä mitä tuote sisältää. (Lehtikuja, R. 2009). Opinnäytetyön vaatimusmäärittely löytyy raporttiosuuden liitteistä (Liite 1).

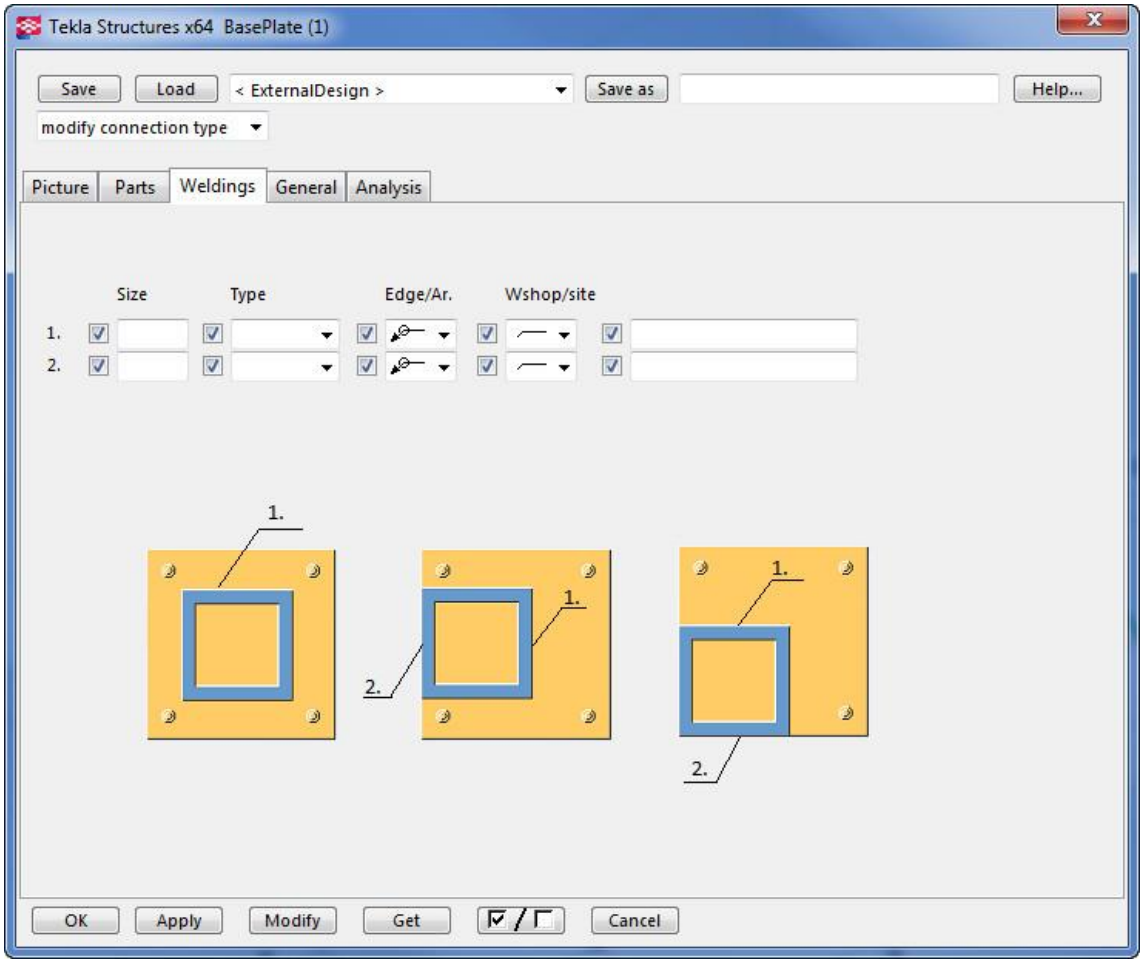
Testauksen tarkoituksena on testata kaikki ohjelman toiminnot niin kattavasti kuin mahdollista. Testitapausten luonnissa käytin White Box -testausmenetelmää. White Box -testausmenetelmää käytetään usein silloin kun ohjelmistorakenne ja toteutus ovat testaajan tiedossa. Testaaja siis tietää mitä reittejä haluaa ohjelmistokoodin kulkevan ja valitsee syötteet sen mukaisesti. Ohjelmointitaidot ovat välttämättömät White Box -testauksessa. Testauksessa syötetään oikeita sekä vääriä syötearvoja, jotka mahdollistavat virhetapausten tarkastelun. Koodin tarkasti testaamisessa on tärkeää, että sama henkilö joka on kirjoittanut koodin myös testaa sen. Näin voi varmistua siitä, että koodin kaikki polut ovat tulleet testatuksi. (Kassem A.S, 224–226). Liitteet osiosta löytyy opinnäytetyöni testitapaukset (Liite 2).

4 IMPLEMENTOINTI

Ohjelmointivaiheessa toteutetaan suunnittelun tuottamat kuvaukset käyttäen muun muassa komponentteja, ohjelmointikieliä sekä tietokantateknologioita. On erittäin epätodennäköistä, että implementointivaiheesta selviää ilman minkäänlaisia ongelmia.

4.1 Inp-käyttöliittymä

Tekla Structures plugin-ohjelmien käyttöliittymät toteutetaan joko Windows Forms -tyyppisenä käyttöliittymänä tai Inp-tiedostotyypin avulla. Inp-käyttöliittymän skripti on Teklan itse kehittämä kieli käyttöliittymän tekoon. Kuvio 4 esittää kuvankaappauksen opinnäytetyöni käyttöliittymän hitsaukset-välilehdestä.



KUVIO 4. Kuvakaappaus pohjalevy-sovelluksen käyttöliittymästä.

Seuraava ohjelmistokoodi esittää hitsaukset-välilehdestä otetun kuvankaappauksen takaa löytyvän ohjelmistokoodin käyttöliittymän osalta.

```
@"tab_page("""", ""Weldings"", 3)" + "\n" +
                    "{\n" +
                        @"attribute(""label_only1"", ""1."", label2, ""%s"",
none, none, ""0.0"", ""0.0"", 20, 80)" + "\n" +
                        @"attribute("""", ""jd_weld_Size"", label2, ""%s"",
none, none, ""0.0"", ""0.0"", 80, 50)" + "\n" +
                        @"parameter("""", ""w1_size"", distance, number, 80,
80, 64)" + "\n" +
                        @"attribute("""", ""jd_weld_Type1"", label2, ""%s"",
none, none, ""0.0"", ""0.0"", 180, 50)" + "\n" +
                        @"attribute(""w1_type"", """", option, ""%s"", none,
none, ""0.0"", ""0.0"", 180, 80, 96)" + "\n" +
                        "{\n" +
                            @"value(""w_type_0.xbm"", 1)" + "\n" +
                            @"value(""w_type_10.xbm"", 0)" + "\n" +
                            @"value(""w_type_4.xbm"", 0)" + "\n" +
                        "}\n" +
                        @"attribute("""", ""jd_weld_Edge_around"", label2,
""%s"", none, none, ""0.0"", ""0.0"", 315, 50)" + "\n" +
                        @"attribute(""w1_around"", """", option, ""%s"",
none, none, ""0.0"", ""0.0"", 315, 80, 64)" + "\n" +
                        "{\n" +
                            @"value(""w_around_0.xbm"", 0)" + "\n" +
                            @"value(""w_around_1.xbm"", 1)" + "\n" +
                        "}\n" +
                        @"attribute("""", ""jd_weld_Wshop_site"", label2,
""%s"", none, none, ""0.0"", ""0.0"", 420, 50)" + "\n" +
                        @"attribute(""w1_wtype"", """", option, ""%s"", none,
none, ""0.0"", ""0.0"", 420, 80, 64)" + "\n" +
                        "{\n" +
                            @"value(""w_workshop_1.xbm"", 0)" + "\n" +
                            @"value(""w_workshop_0.xbm"", 1)" + "\n" +
                        "}\n" +
                        @"attribute("""", ""jd_weld_Wshop_site"", label2,
""%s"", none, none, ""0.0"", ""0.0"", 420, 50)" + "\n" +
                        @"attribute("""", ""Reference text"", label2, ""%s"",
none, none, ""0.0"", ""0.0"", 530, 50)" + "\n" +
                        @"parameter("""", ""reference_text_1"", string, text,
530, 80, 200)" + "\n" +
                        @"attribute(""label_only2"", ""2."", label2, ""%s"",
none, none, ""0.0"", ""0.0"", 20, 105)" + "\n" +
                        @"parameter("""", ""w2_size"", distance, number, 80,
105, 64)" + "\n" +
                        @"attribute(""w2_type"", """", option, ""%s"", none,
none, ""0.0"", ""0.0"", 180, 105, 96)" + "\n" +
                        "{\n" +
                            @"value(""w_type_0.xbm"", 1)" + "\n" +
                            @"value(""w_type_10.xbm"", 0)" + "\n" +
                            @"value(""w_type_4.xbm"", 0)" + "\n" +
                        "}\n" +
                        @"attribute(""w2_around"", """", option, ""%s"",
none, none, ""0.0"", ""0.0"", 315, 105, 64)" + "\n" +
                        "{\n" +
                            @"value(""w_around_0.xbm"", 0)" + "\n" +
                            @"value(""w_around_1.xbm"", 1)" + "\n" +
                        "}\n" +
                        @"attribute(""w2_wtype"", """", option, ""%s"", none,
none, ""0.0"", ""0.0"", 420, 105, 64)" + "\n" +
                        "{\n" +
```

```
                    @"value(""w_workshop_1.xbm"", 0)" + "\n" +
                    @"value(""w_workshop_0.xbm"", 1)" + "\n" +
                "}\n" +
                @"parameter("""", ""reference_text_2"", string, text,
530, 105, 200)" + "\n" +

                @"picture(""BasePlate_Weldings.bmp"", 480, 200, 100,
210)" + "\n" +
            "}\n" +
```

4.1.1 Monikielisyys

Tekla Structures mahdollistaa liitosten luomisen monikieliseksi, joka siis tarkoittaa sitä, että liitos tunnistaa käyttäjän kielen ja lokalisoituu sen mukaan. Käyttöliittymän sisältämien tekstien lisäksi on mahdollista lokalisoida myös liitoksen nimi joka näkyy komponenttikatalogissa. Painikkeiden tekstejä ei voi lokalisoida, näitä ovat esimerkiksi ”OK” ja ”Cancel”. Painikkeiden kieli määräytyy käyttöliittymän kielen mukaiseksi.

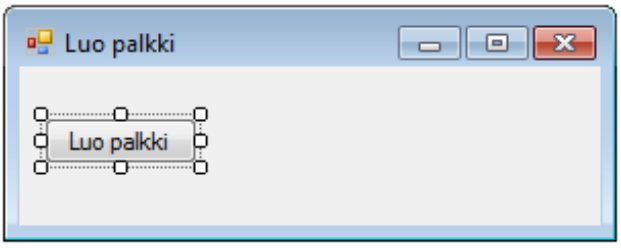
Tekla Structures -liitosten käännetyt tekstit sijaitsevat ohjelmiston asennuskansiossa sijaitsevassa tiedostossa nimeltä joints.ail. Asennuskansio on tässä tapauksessa C:\Tekla Structures\17.0\ messages. Tekstien kääntäminen tehdään XML-tiedostojen kautta, joita käyttää ainoastaan luotu liitos. Lokalisointitiedoston tulee tukea XML-sisältöä ja tekstiformaatin tulee olla UTF-8.

On suositeltavaa käyttää Microsoft Visual Studio -kehitysympäristöä lokalisointitiedostojen luomiseen ja muokkaamiseen. Visual Studio tarkistaa automaattisesti tiedostot virheiden varalta.
(Tekla Corporation 2011, 31—34.)

4.2 Esimerkki sovellus

Liitosten kehittäminen aloitetaan asentamalla vaadittu ohjelmointiympäristö työasemaan. Liitosten kehitystä varten tarvitsee Tekla Structures -rakennesuunnitteluohjelmiston ja voimassa olevan lisenssin sekä C#-ohjelmointikieltä tukevan ohjelmistonkehitysympäristön. Suositeltavaa on käyttää Microsoft Visual Studio -ohjelmaa. Lisäksi ennen ohjelmointiosuuden aloittamista täytyy tietysti tietää mitä aikoo tehdä. Teen tässä esimerkkinä yksinkertaisen ohjelman, joka luo palkin Tekla Structures -malliin.

Aluksi käynnistetään Microsoft Visual Studio- ohjelmointiympäristö ja luodaan uusi projekti. Käytän esimerkissäni Windows Forms -tyyppistä käyttöliittymää, joten luodaan uusi Windows sovellus. Kuvio 5 esittää esimerkkisovelluksen käyttöliittymän.



KUVIO 5. Esimerkkisovelluksen käyttöliittymä.

Seuraava vaihe on lisätä ohjelmistokoodi luotuun painikkeeseen. Ohjelmistokoodiin pääsee tuplaklikkaamalla luotua painiketta. Ennen varsinaisen ohjelmistokoodin kirjoitusta lisätään tarvittavat viitteet, jotta Microsoft Visual Studio osaa käyttää Teklan avoimen rajapinnan komentoja. Kuvio 6 esittää kuinka viitteet otetaan käyttöön koodissa, sekä viitteiden nimeämisen. Viitteitä nimetään lyhempään muotoon ohjelmistokoodin selkeyttämiseksi.

```
//Lisätyt viitteet
using Tekla.Structures;
using Tekla.Structures.Model;
//Viitteiden nimeäminen
using TSM = Tekla.Structures.Model;
using T3D = Tekla.Structures.Geometry3d;
```

KUVIO 6. Esimerkkisovelluksen ohjelmistokoodissa käytetyt viitteet.

Seuraava vaihe on avoimeen Tekla Structures -malliin yhdistäminen, palkin luominen ja palkin asetusten asettaminen. Kuvio 7 esittää kommentteineen ohjelmistokoodin näiden vaiheiden suorittamisesta.

```
private void button1_Click(object sender, EventArgs e)
{
    //Yhdistetään avoimeen malliin
    TSM.Model Malli = new Model();

    //Luodaan palkki
    TSM.Beam Palkki = new Beam();

    //Asetetaan palkin asetukset

    Palkki.Name = "PALKKI"; //Palkin nimi
    Palkki.Profile.ProfileString = "HEA300"; //Palkin profiili
    Palkki.Material.MaterialString = "S235JR"; //Palkin materiaali
    Palkki.Class = "3"; //Palkin luokka

    //Luodaan 1000m pitkä palkki (x, y, z,)
    Palkki.StartPoint = new T3D.Point(0, 0, 0); //Palkin aloituspiste
    Palkki.EndPoint = new T3D.Point(1000, 0, 0); //Palkin lopetuspiste

    //Asettaa palkin paikan mallissa
    Palkki.Position.Depth = Position.DepthEnum.BEHIND;
    Palkki.Position.Plane = Position.PlaneEnum.MIDDLE;
    Palkki.Position.Rotation = Position.RotationEnum.TOP;
```

KUVIO 7. Esimerkkisovelluksen ohjelmistokoodia.

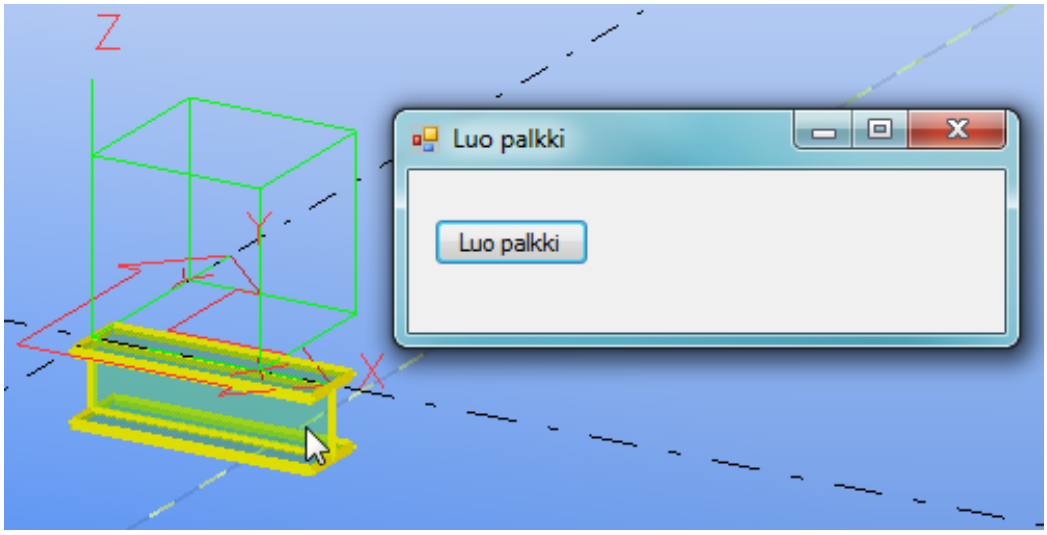
Tässä vaiheessa Tekla Structures -rakennesuunnitteluohjelmistolle on kerrottu mitä luodaan, millä asetuksilla luodaan ja mihin luodaan. Tämän jälkeen sovellukselle on annettava Insert-komento, joka lisää halutun laisen palkin malliin. Kuvio 8 esittää Insert-komennon käytön sekä CommitChanges-komennon käytön.

```
    //Insert-komento lisää palkin avoimeen malliin
    Palkki.Insert();

    // Mahdollistaa luotujen palkkien muuttamisen tai poistamisen mallista.
    Malli.CommitChanges();
}
```

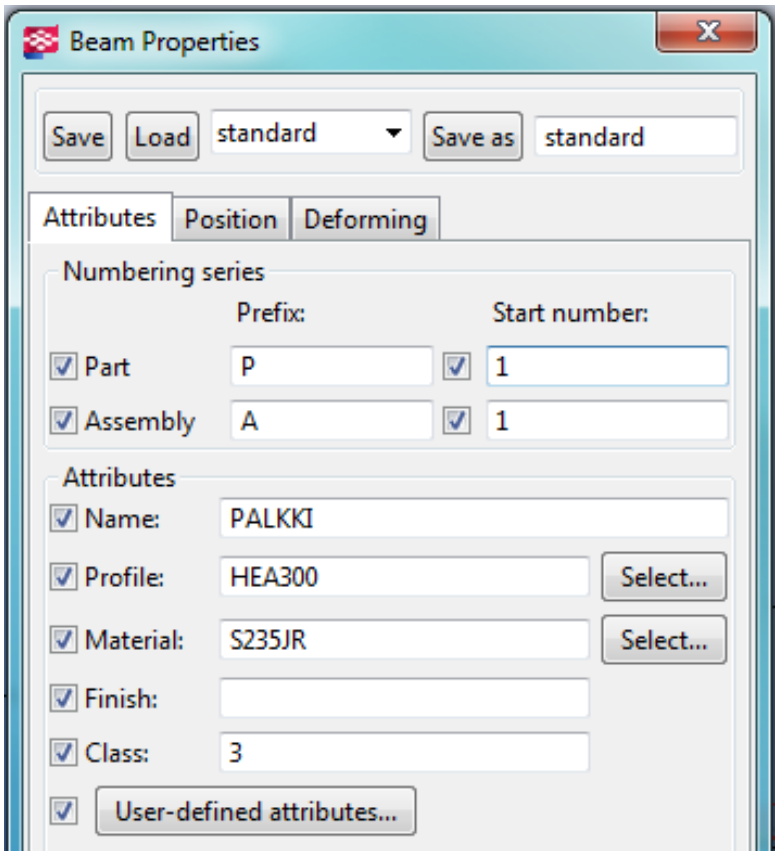
KUVIO 8. Esimerkkisovelluksen ohjelmistokoodia

Seuraavaksi avataan Tekla Structures ja malli johon palkki halutaan luoda. Tämän jälkeen käynnistetään sovellus Microsoft Visual Studio -ohjelmistonkehitysympäristöstä esimerkiksi painamalla F5-näppäintä. Kuvio 9 esittää sovelluksen luoman palkin avoimessa mallissa.



KUVIO 9. Esimerkkisovelluksen luoma 1000mm pitkä palkki Tekla Structures -mallissa.

Palkin asetuksista voi tarkistaa, loiko sovellus palkin oikeilla asetuksilla. Kuvio 10 esittää palkin asetukset Tekla Structures -rakennesuunnitteluohjelmistossa. Palkin asetukset ovat sellaiset kuin ne on ohjelmistokoodissa määritelty.

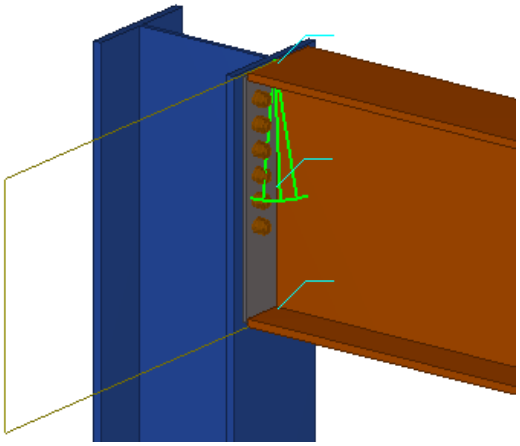


KUVIO 10. Esimerkkisovelluksen luoman palkin asetukset.

4.3 Komponenttien kutsuminen

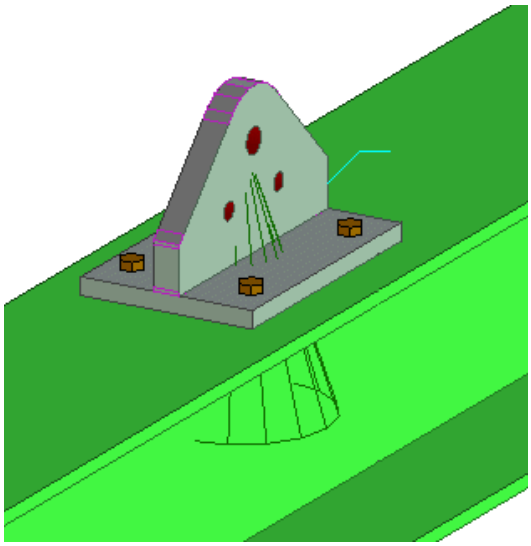
Tekla Structures -rakennesuunnitteluohjelmisto sisältää neljä erilaista komponenttityyppiä. Komponenttityypit ovat Connection, Detail, Part ja Seam. Liitosta kehitettäessä komponenttityyppi valitaan liitoksen käyttötarkoituksen mukaan.

Connection luo liitosesineitä ja yhdistää toissijaisten osien päät ensisijaiseen osaan. Ensisijainen osa voi jatkua liitoskohdasta. Komponentin symboli on vihreä. Kuvio 11 esittää esimerkin Connection-komponenttityypistä.



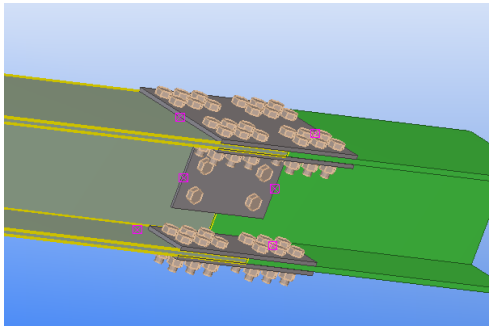
KUVIO 11. Esimerkki Connection-komponenttityypistä.

Detail luo yksityiskohtaisia esineitä ja yhdistää ne yksittäisestä osasta valittuun kohtaan. Komponentin symboli on vihreä. Kuvio 12 esittää esimerkin Detail-komponenttityypistä.



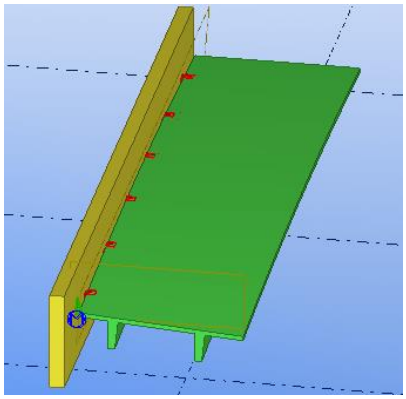
KUVIO 12. Esimerkki detail-komponenttityypistä.

Part luo esinejoukkoja, jotka voivat sisältää liitososia ja yksityiskohtaisia esineitä. Komponentilla ei ole symbolia. Kuvio 13 esittää esimerkin part-komponenttityypistä.



KUVIO 13. Esimerkki part-komponenttityypistä.

Seam luo saumaesineitä ja yhdistää osat linjaa pitkin, joka valitaan kahdella pisteellä. Osat ovat yleensä yhdensuuntaiset. Komponentin symboli on vihreä. Kuvio 14 esittää esimerkin seam-komponenttityypistä.



KUVIO 14. Esimerkki seam-komponenttityypistä.

Opinnäytetyön aiheena olevassa liitoskomponentissa kutsutaan detaljia nimeltä Base Plate (1004). Valmiiden komponenttien kutsumisesta saatava hyöty on suuri. Valmiiden komponenttien kutsuminen vähentää virheiden mahdollisuutta, koska kovakoodattujen valintojen toiminnallisuus tulee suoraa olemassa olevasta komponentista.

Tekla Structures määrää jokaiselle osalle ja kokoonpanolle tunnisteen, jota kutsutaan numeroinniksi. Huolellinen numerointi on tärkeää tuotannon, rakennusvaiheen ja logistiikan kannalta. Komponenttien kutsumisesta saatava hyöty on myös lisätä toistoja numerointiin.

4.4 Asennusmedian luonti

Sovelluksen asennusmedia luodaan Actual Installer -ohjelmistolla. Liitoksesta luodaan omat asennuspakettinsa 64- ja 32-bittisiin käyttöjärjestelmiin. Asennuspaketin luominen ja sen toiminta on yksinkertaista. Asennuspaketin tarkoitus on vain purkaa projektihakemistosta löytyvät kuvat ja projektin luoma dll-tiedosto (Dynamic-link library) Tekla Structures -rakennesuunnitteluohjelmiston asennuskansioon oikeisiin paikkoihin. Asennuspaketin luontia edeltävä toimenpide on luoda asennuspaketille oma kansio, jonka kansiorakenne vastaa Tekla Structures -rakennesuunnitteluohjelmiston kansiorakennetta siltä osin, mihin paketin on tarkoitus tiedostot purkaa. (Easy Desk, 2011).

4.5 Testaaminen

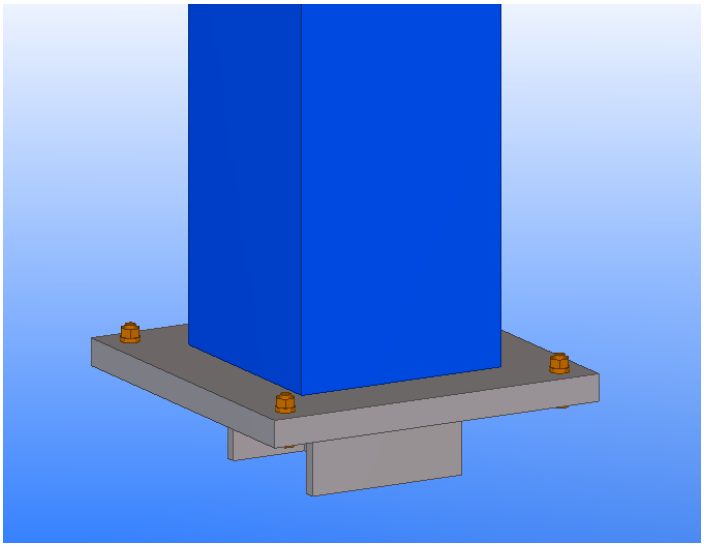
Ohjelmistojen testaukseen liittyviä työvaiheita ovat testauksen suunnittelu, johon sisältyy testaus suunnitelma ja testitapaukset, testiympäristön luonti, testin suorittaminen ja tulosten tarkastelu. Testauksen vaiheisiin ja testauksessa ilmenevien vikojen paikantamiseen kuluu normaalisti yli puolet projektin resursseista. Ohjelmistojen testauksen yhteydessä testausta määritellään perinteisesti suunnitelmalliseksi virheiden etsimiseksi ohjelmaa suorittamalla. Usein testaus tapahtuu ohjelman suorittamisella umpimähkäisesti joillain syöttöaineistoilla. Varsinkin jos testaajana on ohjelman tekijä, tavoitteena on osoittaa ohjelman toimivuus virheiden löytämisen sijaan. Testaukseen käytettävien työtuntien ja testitapausten määrä ei välttämättä ole suoraa verrannollinen testauksen tehokkuuteen. Muutamien tarkasti suunniteltujen testitapausjoukkojen testaaminen voi johtaa parempaan tulokseen kuin päiviä kestävä umpimähkäinen kokeilu.

Nykyään testaus määritellään hieman laajemmin laadun mittaamisen näkökulmasta siten, että testauksen katsotaan käsittävän ne kaikki menetelmät, joilla pyritään mittaamaan ja parantamaan ohjelman laatua. (Ohjelmistotuotanto, 283—284)

Opinnäytetyöni aiheena olevan ohjelman testaus on koko projektin ajan kestävä prosessi. Jo vaatimusmäärittelyssä pyritään miettimään testitapauksia, jotka osattaisiin ottaa huomioon jo ohjelmaa tehtäessä. Ohjelmaa testataan koko kehitysprosessin ajan Tekla Structures -rakennesuunnitteluohjelmassa.

5 TULOKSET

Opinnäytetyöni tavoitteena oli luoda Tekla Structures -rakennesuunnitteluohjelmistossa toimiva liitoskomponentti nimeltään Base Plate. Opinnäytetyöni piiriin kuului koko ohjelmistokehitysprosessi, aikataulujen suunnittelusta valmiin ohjelmiston testaukseen ja julkaisuun. Opinnäytetyön tuloksena valmistui tavoitteiden mukainen liitoskomponentti. Testaus vaiheessa käytiin läpi liitoksen toiminnot ja ne on todettu toimivan oikein. Testausvaiheen jälkeen luotiin asennusmediat 32bit- ja 64bit-käyttöjärjestelmiin, joista molemmat on testattu toimiviksi. Kaiken kaikkiaan opinnäytetyö oli haastava projekti, joka olikin yksi suurimmista syistä miksi sen valitsin. Halusin opinnäytetyöni kehittävän ammatillisia valmiuksiani mahdollisimman kattavasti. Kuvio 15 esittää kuvanruutukaappauksen kehitystehtävänä toimineen liitoksen luomasta pohjalevystä.



KUVIO 15. Valmiilla liitoksella luotu pohjalevy.

5.1 Toiminnallisuus

Pohjalevy-detalji luotiin yhdistämällä luotava detalji Tekla Structures -rakennesuunnitteluohjelmasta valmiiksi löytyvään vakioliitokseen nimeltään Base Plate (1004). Base Plate (1004) on pohjalevyn luotiin tarkoitettu työkalu. Base Plate (1004) sisältää erittäin paljon käyttäjälle suunnattuja valintoja. Pohjalevy-detaljin tarkoitus on karsia valintoja vähemmäksi ja tehdä detaljin käytöstä nopeaa ja selkeää. Ohjelmistokoodin (liite 3) luokkarakenne pyrittiin toteuttamaan niin, että koodi on selkeälukuista ja virheiden etsintä mahdollisimman vaivatonta.

Luokkarakenne:

- BasePlate.cs

  Sisältää lähdekoodin, joka pyytää käyttäjää valitsemaan palkin ja luontipisteen. Luokka sisältää myös Run-metodin, joka on komponentin päämetodi. Run-metodi suorittaa komponentin toiminnallisuuden ja lisää pohjalevyn käyttäjän valitsemaan pilariin. Mikäli pilaria ei ole valittu, Run-metodi palauttaa false-arvon.

- BasePlateDetail1004.cs

  Luokassa luodaan muuttujat, joille asetetaan Base Plate (1004) -vakioliitoksesta haetut arvot. Luokassa myös kova koodataan joidenkin muuttujien arvot. Insert-metodissa asetetaan komponentin ominaisuudet.

- BasePlateUI.cs

  Luokka sisältää komponentin käyttöliittymän. Käyttöliittymä on mahdollista asettaa osaksi Microsoft Visual Studio -projektia luomalla se .cs-tiedostomuotoon. Käyttöliittymän skripti on tarkoitettu olemaan .inp-tiedostomuotoinen, joten se on suoritettava yhtenä pitkänä merkkijonona.

- DeBasePlate1004.cs

  Tässä luokassa asetetaan arvot parametreille, joita komponentti käyttää. Luokka sisältää kaikki komponentin käyttämät parametrit. Luokka sisältää myös globalisoinnin, joka mahdollistaa syötteiden antamisen millimetreinä tuumien sijaan. Lisäksi luokassa on ohjelmistokoodit, joilla määritellään pohjalevyn sijainti.

- StructuresData.cs

  Luokka sisältää [StructuresField()]-attribuutteja. Nämä attribuutit osoittavat komponentille tietokannasta ne attribuutit, jotka komponentti lisää niille tarkoitettuihin kenttiin.

- WeldCatalog

  Tässä luokassa määritellään komponentin käyttämät hitsaustyypit. Tekla Structures sisältää 26 hitsaustyyppiä. Tämä luokka asettaa komponentille kaksi erityyppistä hitsaus vaihtoehtoa, numerot 10 ja 4. Numerolla 10 tarkoitetaan pienahitsiä ja numerolla 4 päittäishitsiä.

5.2 Testitapaukset

Pohjalevy-detaljin testaaminen tapahtui koko projektin aikana. Lisäksi pohjalevyn valmistuttua laadittiin dokumentti testitapauksista (liite 2). Testitapauksia laadittaessa pyrittiin kiinnittämään huomiota testauksen kattavuuteen ja luotettavuuteen. Ohjelmistokoodi sisältää tiettyjä polkuja joita pitkin sovelluksen suorittaminen voi kulkea läpi. Jokaiseen näistä poluista laadittiin testitapaus, jolla todettiin sen olevan toimiva. Testitapaukset pyrittiin laatimaan huomioiden kaikki mahdolliset valinta kombinaatiot, joita käyttäjä voisi pohjalevylleen valita.

6 POHDINTA

Rakennustekniikka ja rakennesuunnittelu ovat jatkuvasti kehittyviä aloja, joten niihin käytetyiden ohjelmistojen tulee kehittyä käsi kädessä ja vastata ammattilaisten tarpeisiin. Tämä opinnäytetyö esittelee yhden komponentin, joka nopeuttaa rakennesuunnittelijoiden päivittäistä työtä. Opinnäytetyössä esitellyn komponentin kaltaiset automatisoidut mallinnuksessa hyödynnettävät komponentit vähentävät inhimillisten virheiden mahdollisuutta mallinnuksessa, joten rakennesuunnittelijalla jää paremmin aikaa varsinaisen mallin suunnitteluun virheiden etsimisen ja tarkistamisen sijaan. Opinnäytetyöni komponentti on liitoskomponentti joka käyttää hyödykseen Tekla Structures -rakennesuunnitteluohjelmistosta valmiiksi löytyvää Base Plate (1004) -detaljia. Perimmäinen ajatus olikin saada Base Plate (1004) -detaljista räätälöityä paremmin Hopia Engineering Oy:n käyttöön sopiva versio. Käyttöliittymästä tehtiin yksinkertaisempi ja siitä tehtiin niin sanotusti ”monikielinen”, eli valinnat tapahtuvat pääasiassa kuvien avulla. Kuvista komponentin käyttäjä pystyy selvästi hahmottamaan mitä minäkin toiminto tekee.

Mielestäni liian automatisoiduissa komponenteissa on myös haittapuolensa. Suunnittelijan tulee mielestäni pystyä esimerkiksi tarkistamaan ja muuttamaan mittoja komponentin käyttöliittymää käyttäen, eikä niin, että komponentti hakee kaikki mitat automaattisesti ohjelmistokoodia käyttäen. Liian automatisoitu komponentti tekee komponentin käytöstä marginaalisempaa eikä sitä välttämättä voi hyödyntää riittävän useissa käyttökohteissa.

Opinnäytetyötä tehdessä olen oppinut valtavasti uusia asioita. Olen oppinut uutta niin Tekla Structures -rakennesuunnitteluohjelmiston käytöstä kuin ohjelmistokehityksestäkin. Hopia Engineering Oy:lle harjoitteluun mennessäni minulla ei ollut oikeastaan mitään kokemusta Tekla Structures -rakennesuunnitteluohjelmiston käytöstä. Aiempaa kokemusta rakennesuunnitelusta minulla oli aiemman suunnitteluassistentin koulutukseni kautta. Tekla Structures -rakennesuunnitteluohjelmisto on niin laaja kokonaisuus, että sen toiminnan hahmottaminen ja ymmärtäminen vaatii aikansa ennen kuin siihen voi alkaa kehittämään komponentteja. Minulla kävi tuuri, että pääsin ennen opinnäytetyöni aloittamista suorittamaan viiden kuukauden mittaisen harjoittelujakson Hopia Engineering Oy:lle ja sitä kautta tutustumaan Tekla Structures -rakennesuunnitteluohjelmistoon ja liitoskehitykseen.

Mielestäni Tekla Structures -rakennesuunnitteluohjelmiston sisältämä avoin rajapinta (Open API) täydentää käyttäjien tarpeita sekä nopeuttaa ohjelman kehittymistä. Liitoskehityksen kohdalla läpinäkyvä ja avoin ohjelmistokehitys luo nopeamman tavan verkostua ja jakaa kehitettäviä menetelmiä.

Opinnäytetyötä tehdessä huomasin, kuinka liitosten kehittäminen auttoi kehittymään myös mallinnuksen puolella. Koko opinnäytetyöprosessi on kehittänyt omia taitojani erittäin paljon.

LÄHTEET

Easy Desk. 2011. A DLL is a Dynamic Link Library. Hakupäivä 13.6.2013.
http://www.easydesksoftware.com/dll.htm

Haikala, I & Märijärvi, J. 2004. Ohjelmistotuotanto. Helsinki: Talentum.

Kassem A.S, Gopalawswamy R. 2009. Software Engineering.
http://books.google.fi

Lehtikuja, R. 2009. Vaatimusmäärittelyn taso ja työn jatkaminen. Hakupäivä 26.6.2013.
http://www.pcuf.fi/sytyke/lehti/kirj/st20092/ST092-26A.pdf

Lenkkeri, J. Aureolis Oy. Versionhallinta teoriassa. Hakupäivä 3.6.2013.
www.sugif.fi/arkisto2013/CVS_esitys.pptx

Salonen, V. 2011. Javasta C#:iin. Hakupäivä 28.6.2013.
http://villesalonen.fi/2011/javasta-ciin/

Sarja, J. 2006. ASP-ohjelmointi. Hakupäivä 13.6.2013.
http://www.verkkopedagogi.net/vanhat/fi/sisalto/materiaa-lit/asp/luku048b84.html?C:D=418946&selres=418946

Tekla Corporation. 2011. Developers Guide. Product Version 17.0.

Tekla Corporation. Tekla BIM. Hakupäivä 13.6.2013.
http://www.tekla.com/fi/products/tekla-structures/Pages/Default.aspx

Tekla Corporation. Avoin ympäristö sovelluskehittäjille. Hakupäivä 26.6.2013.
http://www.tekla.com/fi/solutions/building-construction/application-developers/Pages/Default.aspx

Tekla Corporation. 2013. Introduction to drawings. Hakupäivä 4.7.2013.
http://teklastructures.support.tekla.com/190/en/dra_introduction_to_tekla_structures_drawings

LIITTEET

Liite 1. Vaatimusmäärittely

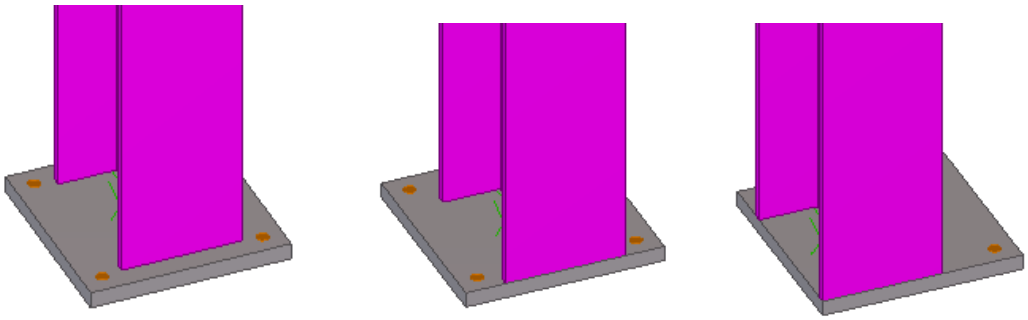
VAATIMUSMÄÄRITTELY: TEKLA STRUCTURES PLUGIN SPECIFIC BASE PLATE

SISÄLLYS

1 YLEISTÄ

Tuotteena on Tekla Structures -detalji, jonka tehtävä on luoda pohjalevy valittuun pilariin. Tuotteen nimi on Specific Base Plate.

Käyttäjän on mahdollista valita pohjalevyn sijainnille kuvan 1. mukaisesti kolme eri vaihtoehtoa, joko keskelle, reunaan tai kulmaan.



Kuva 1. Vaihtoehdot pohjalevyn sijainnille.

Käyttäjä määrittää pohjalevylle myös koon ja paksuuden. Käyttöliittymästä löytyy erikseen välilehdet myös hitsausten ja pulttien ominaisuuksille.

2 OHJELMAVAATIMUKSET

Ohjelmaversiot

- Tekla Structures 17.0
- Tekla Structures 18.0
- Tekla Structures 19.0

Ympäristö

- Detalji toimii kaikissa Tekla Structures ympäristöissä.

Kieli

- Detaljista tehdään monikielinen, eli sanoja ei kovakoodata käyttöliittymään vaan nehaetaan kielen mukaan.

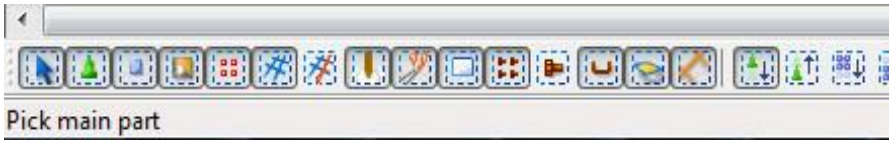
Toteutus

- .inp-tyyppinen käyttöliittymä
- Detalji luodaan connection base -luokkaan.

3 VAIHEET

Pohjalevyn luonti

- Valitaan pilari johon pohjalevy luodaan.
- Valitaan luontipiste joka vastaan pohjalevyn alapintaa.
- Sovellus luo pohjalevyn



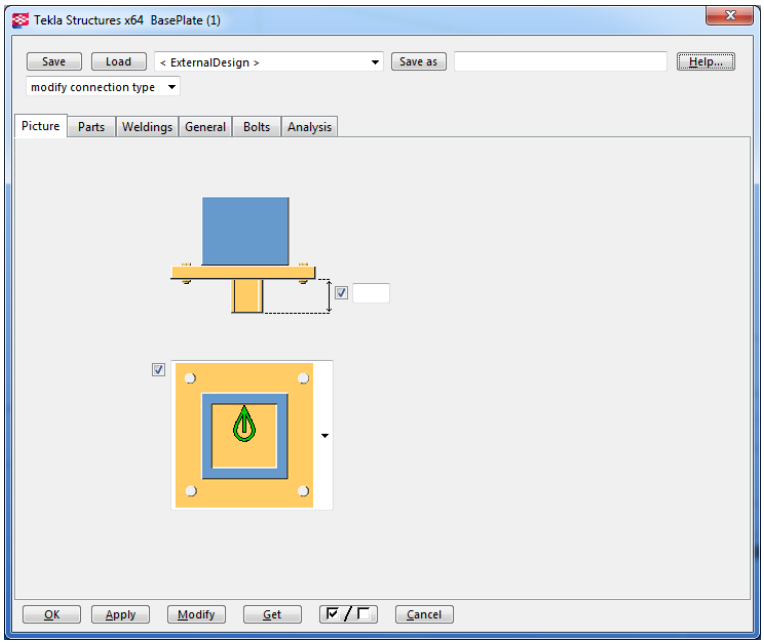
Kuva 2. Valitse pilari

Kuva 3. Valitse luontipiste

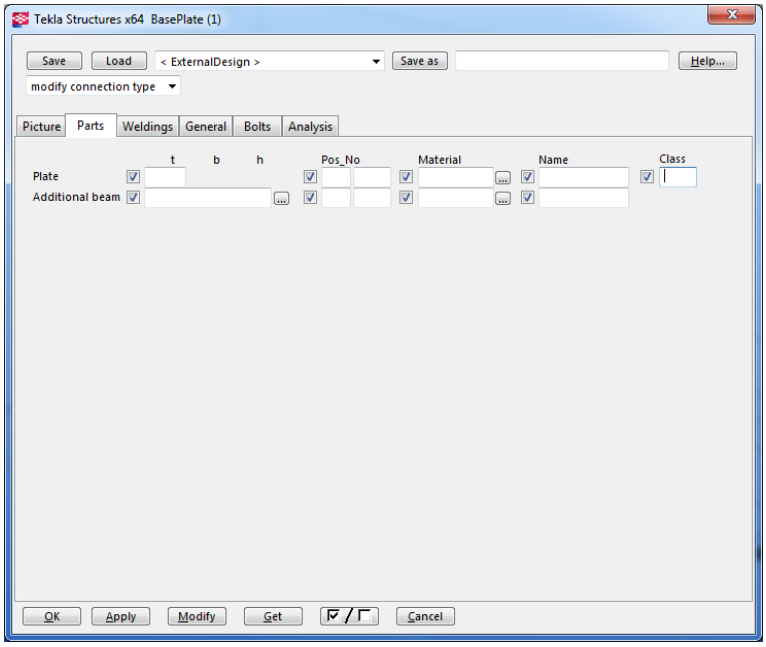
4 KÄYTTÖLIITTYMÄ

Käyttöliittymä

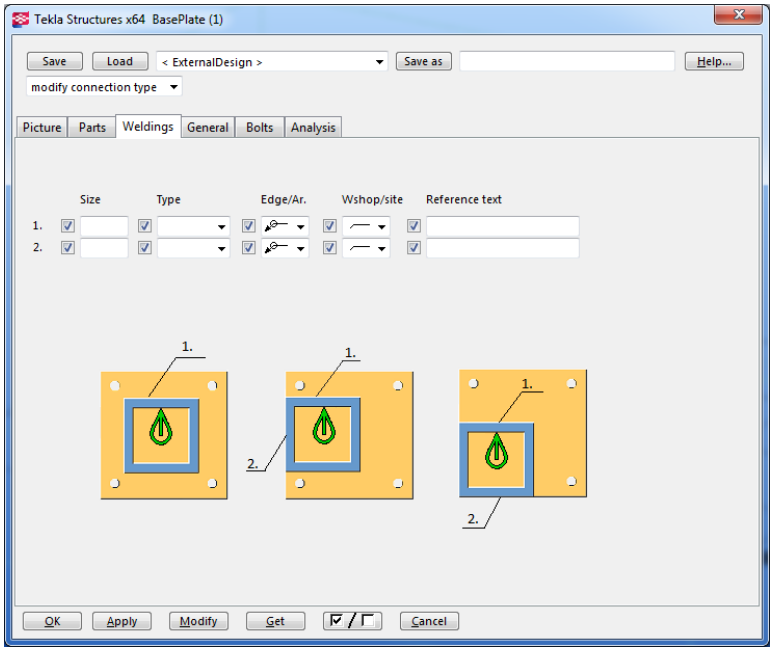
- Ikonit kokoina 16x16 pikseliä ja 32x32 pikseliä.
- Dialogi
- Välilehdet: Picture, Parts, Weldings, General ja Analysis



Kuva 4. Käyttöliittymän Picture-välilehti



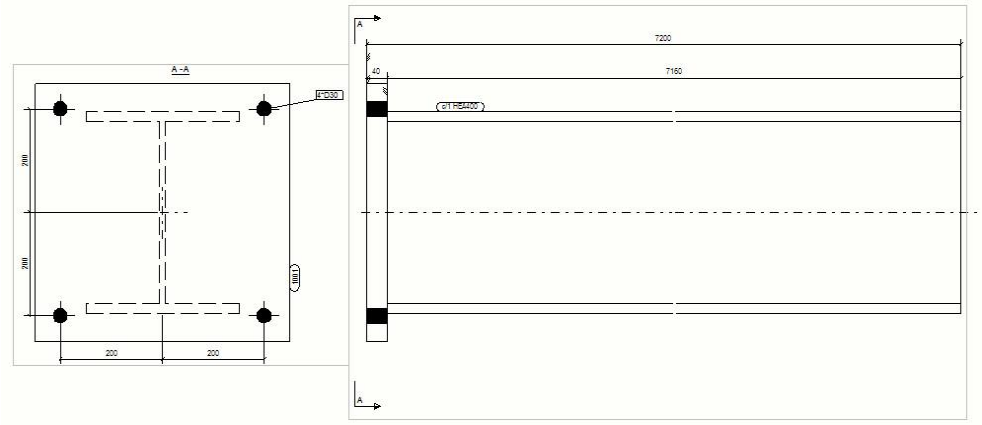
Kuva 5. Käyttöliittymän Parts-välilehti



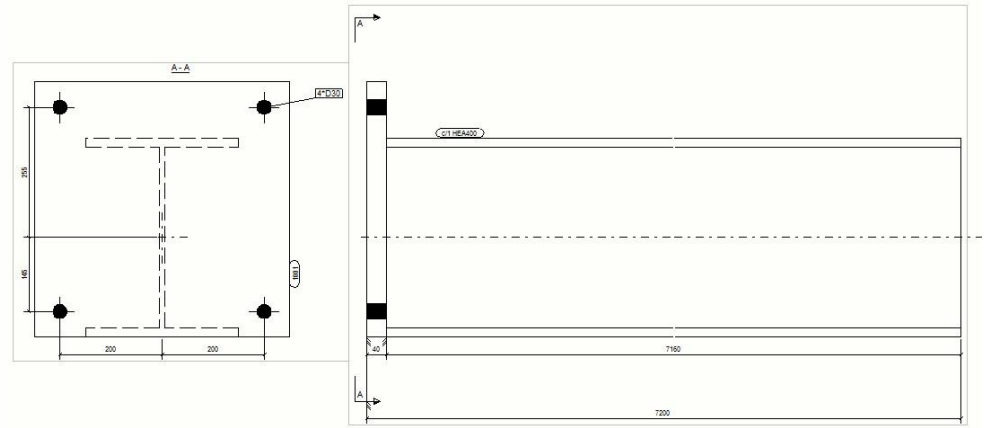
Kuva 6. Käyttöliittymän Weldings-välilehti

5 PIIRUSTUKSET

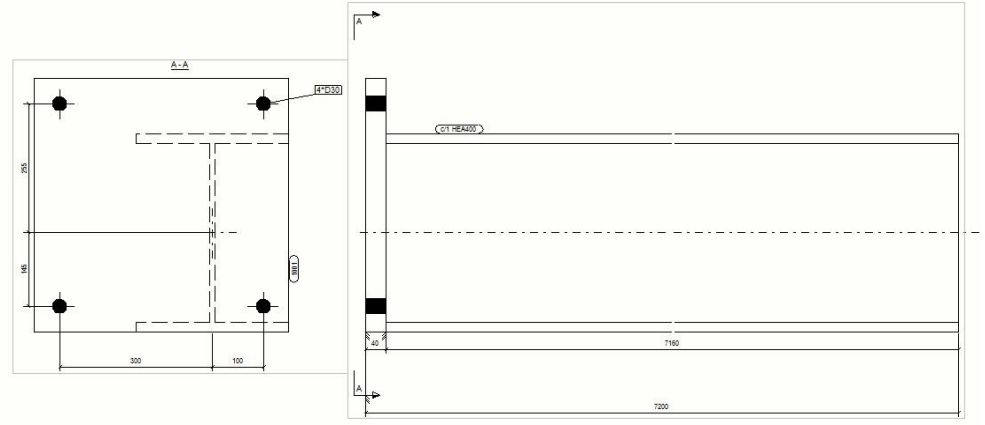
Kokoonpanopiirustus (Pohjalevyn sijainti ja sijoitus)



Kuva 2. Sijainti keksellä



Kuva 3. Sijainti reunassa



Kuva 4. Sijainti nurkassa

6 OHJETIEDOSTO

Ohjetiedosto (Help-tiedosto)

- Luodaan .chm muotoon.
- Luodaan HTML-kielellä.
- Sisältää kuvauksen luotavasta detaljista sekä käyttöliittymän kenttien oletusarvot.

7 TESTAUS

Testitapaukset (Cases)

Tehtävä	Tulos	Toteutuiko?	Kommentti
Klikkaa sovelluksen ikonia.	Sovellus käynnistyy.	Kyllä	-
Valitse pilari, johon pohjalevy luodaan.	Sovellus valitsee pilarin ja pyytää valitsemaan luontipisteen.	Kyllä	-
Valitse luontipiste.	Sovellus luo pohjalevyn.	Kyllä	-
Tuplaklikkaa valittua pohjalevyä	Sovellus avautuu	Kyllä	-
Vaihda pohjalevyn sijaintia Picture-välilehdellä olevasta pudotusvalikosta, jonka jälkeen paina Modify-painiketta.	Pohjalevyn sijainti muuttuu valittuun paikkaan.	Kyllä	-
Määritä Picture-välilehdeltä lisäpalkin pituus, jonka jälkeen määritä Parts-välilehdeltä lisäpalkille profiili, materiaali ja nimi. Tämän jälkeen paina Modify-painiketta.	Sovellus luo lisäpalkin annetuin asetuksin.	Kyllä	-

8 ASENNUSTIEDOSTOT

Luotavat asennusmediat

- 32 bit
- 64 bit

Ei vaadita pääkäyttäjäominaisuuksia eikä tarkisteta käyttöjärjestelmää. Oletuksena haetaan rekisteristä asennuskansio, jos ei löydy, käytetään kansiorakennetta C:\Tekla Structures\<versio>.

Luodaan uninstaller -rekisteri, jolloin:

- Käyttäjä voi poistaa asennetut tiedostot ohjauspaneelin lisää/poista sovellus -ohjelman kautta.
- Ajetaan kun käynnistetään asennus uudestaan.

Sovellusohjelmat niihin edellyttämiin kohdekansioihin

- Sovellukset ja sovelluskirjastot
- Kuvakkeet ja thumbnailit
- Help -tiedosto
- Attribuuttitiedostot

Versio: Tekla Structures 17.0
Environment: Default, Finland

BasePlate-tool_v1.0_32bit_TS17.0.exe
BasePlate-tool_v1.0_64bit_TS17.0.exe

TESTITAPAUKSET: TEKLA STRUCTURES PLUGIN SPECIFIC BASE PLATE

Versio	Muutokset	Pvm.	Tekijä
0.1	Draft	10.6.2013	Antti Aho
0.2	Testitapausten läpikäynti	12.6.2013	Antti Aho

TESTITAPAUKSET

Attributes

	Kuvaus	Odotettu tulos	Tulos	Kommentti
1	Avaa BasePlate -työkalu.	Työkalu aukeaa. Attributes -välilehdellä on standard -asetusta vastaavat arvot.	Odotettu	
2	Muokkaa attribute -asetuksia ja tallenna asetukset Save as painikkeella nimellä ”Test”.	”Test” niminen asetus luodaan ja se löytyy työkalun alasvetovalikosta.	Odotettu	
3	Valitse ”Test” alasvetovalikosta ja paina Load painiketta.	”Test” asetukset ladataan työkalun kenttiin ja ne vastaavat tallennettuja asetuksia.	Odotettu	
4	Muokkaa ”Testin” asetuksia ja paina Save.	Tehdyt muutokset asentuvat ”Testille”.	Odotettu	

Parts

	Kuvaus	Odotettu tulos	Tulos	Kommentti
1	Avaa Petra -työkalu standardiasetuksilla.	Työkalu aukeaa. Parameters -välilehdellä on standard -asetusta vastaavat arvot	Odotettu	
2	Muokkaa parameters -asetuksia ja tallenna asetukset Save as painikkeella nimellä ”Test”.	”Test” niminen asetus luodaan ja se löytyy työkalun alasvetovalikosta.	Odotettu	
3	Valitse ”Test” alasvetovalikosta ja paina Load painiketta.	”Test” asetukset ladataan työkalun kenttiin ja ne vastaavat tallennettuja asetuksia.	Odotettu	
4	Muokkaa ”Testin” asetuksia ja paina Save.	Tehdyt muutokset asentuvat ”Testille”.	Odotettu	

Uusi kappale

	Kuvaus	Odotettu tulos	Tulos	Kommentti
1	Avaa BasePlate -työkalu ja luo pohjalevy viiteen pilariin, muuttaen jokaisen kappaleen arvoja.	Pohjalevyt luodaan.	Odotettu	
2	Suorita numerointi.	Numerointi suoritetaan.	Odotettu	
3	Luo raporttipohja luoduille kappaleille.	Raporttipohja luodaan.	Odotettu	

		Listattujen kappaleiden attribuutit vastaavat luotuja kappaleita.		
4	Tallenna malli.	Malli tallennetaan.	Odotettu	

Olemassa oleva kappale

	Kuvaus	Odotettu tulos	Tulos	Kommentti
1	Käytä edellisessä tehtävässä luotua mallia.		Odotettu	
2	Suorita numerointi.	Numerointi suoritetaan.	Odotettu	
3	Avaa BasePlate dialogi ja aseta dialogiin eri arvot kuin tarkastettava komponentti. Tallenna ”Apply” + ”OK”.	Arvot tallentuvat dialogiin.	Odotettu	
4	Valitse mallista pohjalevy ja tuplaklikkaa sitä.	Dialogin arvot päivittyvät vastaamaan valittua komponenttia.	Odotettu	
5	Suorita valitun komponentin main partille inquire object.	Inquiren arvot vastaavat dialogin arvoja.	Odotettu	

Muokkaus olemassa olevalle kappaleelle

	Kuvaus	Odotettu tulos	Tulos	Kommentti
1	Käytä edellisessä tehtävässä luotua mallia.		Odotettu	
2	Valitse mallista pohjalevy ja tuplaklikkaa sitä.	Dialogin arvot päivittyvät vastaamaan valittua komponenttia.	Odotettu	
3	Muokkaa komponentin asetuksia ja paina modify.	Pohjalevy muokkautuu uusiin asetuksiin.	Odotettu	
4	Suorita valitun komponentin main partille inquire object.	Inquiren arvot vastaavat dialogin arvoja.	Odotettu	

Pohjalevyn asetukset

Pohjalevyn luonti oletusasetuksilla

	Kuvaus	Odotettu tulos	Tulos	Kommentti
1	Avaa BasePlate -työkalu ja luo pohjalevy muokkaamatta mitään asetuksia.	Pohjalevy luodaan Mitat: 550x460x10mm Levyn sijanti: Keskellä	Odotettu	

Pohjalevyn luonti niin, että pilari sijaitsee pohjalevyn reunassa

	Kuvaus	Odotettu tulos	Tulos	Kommentti

1	Avaa BasePlate -työkalu ja valitse käyttöliittymän picture -välilehden puottovalikosta pilarin sijainti pohjalevyn reunaan.	Pohjalevy luodaan niin, että pilari sijaitsee pohjalevyn reunassa	Odotettu	

Pohjalevyn luonti niin, että pilari sijaitsee pohjalevyn nurkassa

	Kuvaus	Odotettu tulos	Tulos	Kommentti
1	Avaa BasePlate -työkalu ja valitse käyttöliittymän picture -välilehden puottovalikosta pilarin sijainti pohjalevyn nurkkaan	Pohjalevy luodaan niin, että pilari sijaitsee pohjalevyn nurkassa.	Odotettu	

Lisäpalkin luonti

	Kuvaus	Odotettu tulos	Tulos	Kommentti
1	Valitse BasePlate -työkalulla luotu pohjalevy mallista. Aseta käyttöliittymän picture -välilehdellä sijaitsevaan lisäpalkin pituus kenttään arvo, jonka mukaan palkki luodaan. Valitse Parts-välilehdeltä additional beam riviltä kappaleen profiili. Paina ”Modify”.	Lisäpalkki luodaan annettun mitan mukaisesti.	Odotettu	

Pohjalevyn paksuus, materiaali, nimi ja luokka

	Kuvaus	Odotettu tulos	Tulos	Kommentti
1	Valitse BasePlate -työkalulla luotu pohjalevy mallista tuplaklikkaamalla. Aseta pohjalevyn Parts-välilehdeltä Plate kohdan arvot. Paina ”Modify”.	Annetut asetukset tallentuvat valittuun pohjalevyyn.	Odotettu	

Lisäpalkin porfiili, materiaali, nimi ja luokka

	Kuvaus	Odotettu tulos	Tulos	Kommentti
1	Valitse BasePlate -työkalulla luotu pohjalevy, jossa on lisäpalkki tuplaklikkaamalla sitä. Aseta pohjalevyn Parts-välilehdeltä Additional beam kohdan arvot. Paina ”Modify”.	Annetut asetukset tallentuvat lisäpalkiin.	Odotettu	

Pohjalevyn luonti, hitsaukset, pultit

Hitsaukset

	Kuvaus	Odotettu tulos	Tulos	Kommentti
1	Avaa BasePlate -työkalu ja luo pohjalevy oletusarvoni valittuun pilariin.	Pohjalevy luodaan.	Odotettu	
2	Valitse luotu pohjalevy tuplaklikkaamalla.	Käyttöliittymä avautuu.	Odotettu	
3	Mene käyttöliitymän Weldings -välilehdelle ja aseta rivin 1. hitsausasetukset. Klikkaa ”Modify”.	Hitsausten asetukset tallentuvat asetusten mukaisesti.	Odotettu	

Pultit

	Kuvaus	Odotettu tulos	Tulos	Kommentti
1	Avaa BasePlate -työkalu ja luo pohjalevy oletusarvoin valittuun pilariin.	Pohjalevy luodaan.	Odotettu	
2	Valitse luotu pohjalevy tuplaklikkaamalla.	Käyttöliittymä avautuu.	Odotettu	
3	Mene käyttöliittymän Bolts -välilehdelle ja aseta puottovalikkojen arvot: size, standard, tolerance ja thread in mat. Klikkaa ”Modify”	Pohjalevy tallentuu annettujen asetusten mukaiseksi.	Odotettu	
4	Valitse pohjalevy mallista ja mene takaisin Bolts-välilehdelle. Aseta pulttien väliset mitat ja pulttien lukumäärä käyttöliittymän kenttiin.	Pohjalevy tallentuu malliin annettujen asetusten mukaisesti.	Odotettu	

LÄHDEKOODI: TEKLA STRUCTURES PLUGIN SPECIFIC BASE PLATE

SISÄLLYS

BASEPLATE.CS

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Windows.Forms;

using Tekla.Structures;
using TSG = Tekla.Structures.Geometry3d;
using Tekla.Structures.Plugins;
using Tekla.Structures.Model.UI;
using Tekla.Structures.Catalogs;
using System.Collections;
using System.IO;
using Tekla.Structures.Model;
using Tekla.Structures.Geometry3d;
using Tekla.Structures.Datatype;

namespace BasePlate
{
    [Plugin("BasePlate")]
    [PluginUserInterface(BasePlateUI.BasePlatePlugin)]

    class BasePlate : PluginBase
    {
        private readonly StructuresData _data;
        private readonly Model _model;

        private DeBasePlate1004 basePlateDetail;

        //Constructor
        public BasePlate(StructuresData data)
        {
            this._data = data;
            _model = new Model();
        }

        public override List<InputDefinition> DefineInput()
        {
            List<InputDefinition> PickerList = new List<InputDefinition>();
            Picker Picker = new Picker();

            int i = 0;
            while (i < 1)
            {
                Beam pickedBeam = Picker.PickObject(Picker.PickObject-
Enum.PICK_ONE_OBJECT, "Pick a column") as Beam;

                if (pickedBeam != null)
                {
                    PickerList.Add(new InputDefinition(pickedBeam.Identi-
fier));

                    i++;
                }
            }
```

```
            ArrayList PickedPoints = Picker.PickPoints(Picker.Pick-
PointEnum.PICK_ONE_POINT);

            PickerList.Add(new InputDefinition(PickedPoints));

            return PickerList;
        }

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

            try
            {
                Identifier ID1 = (Identifier)((InputDefinition)Input[0]).Get-
Input();
                Beam pickedBeam = _model.SelectModelObject(ID1) as Beam;

                Point pickedPoint = Input[1].GetInput() as Point;

                if (pickedBeam != null)
                {
                        basePlateDetail = new DeBasePlate1004(_data);
                        result = basePlateDetail.Insert(pickedBeam,
pickedPoint);
                }
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.Message);
            }
            return result;
        }


    }
}
```

BASEPLATEDETAIL1004.CS

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using Tekla.Structures.Plugins;
using TSG = Tekla.Structures.Geometry3d;
using Tekla.Structures.Model;
using System.Windows.Forms;
using Tekla.Structures.Geometry3d;
using Tekla.Structures;

namespace BasePlate
{
    class BasePlateDetail1004
    {
        //Base Plate properties
        private double _tpl2;
        public double Tpl2
        {
            get { return _tpl2; }
            set {_tpl2 = value; }
        }

        private double _bpl2;
        public double Bpl2
        {
            get { return _bpl2; }
            set {_bpl2 = value; }
        }

        private double _hpl2;
        public double Hpl2
        {
            get { return _hpl2; }
            set {_hpl2 = value; }
        }

        private string _mat2;
        public string Mat2
        {
            get { return _mat2; }
            set {_mat2 = value; }
        }

        private string _pre2;
        public string Pre2
        {
            get { return _pre2; }
            set { _pre2 = value; }
        }

        private int _sno2;
        public int Sno2
        {
            get { return _sno2; }
            set { _sno2 = value; }
```

```
}

private string _partName2;
public string PartName2
{
    get { return _partName2; }
    set { _partName2 = value; }

}

private int _epClass;
public int EpClass
{
    get { return _epClass; }
    set { _epClass = value; }
}

//Weldings
private double _w1Size;
public double W1Size
{
    get { return _w1Size; }
    set { _w1Size = value; }
}

private int _w1Type;
public int W1Type
{
    get { return _w1Type; }
    set { _w1Type = value; }
}

private int _w1Around;
public int W1Around
{
    get { return _w1Around; }
    set { _w1Around = value; }
}

private int _w1WType;
public int W1WType
{
    get { return _w1WType; }
    set { _w1WType = value; }
}

public BasePlateDetail1004(StructuresData data)
{
    try
    {
        if( data != null)
        {
            WeldCatalog welds = new WeldCatalog();
            this.Tpl2 = data.tpl2;
            this.Bpl2 = data.bpl2;
            this.Hpl2 = data.hpl2;

            this.Pre2 = data.GussetPrefixPos;
            this.Sno2 = data.GussetStartNoPos;
```

```
                    if (data.mat2 != String.Empty) this.Mat2 = data.mat2;
                    else this.Mat2 = "S355J2H";
                    this.PartName2 = data.partName2;
                    this.EpClass = data.epClass;

                    // Weld dialog parameters
                    if (data.w1_size != Convert.ToDouble(Int32.MinValue))
this.W1Size = data.w1_size;
                    else this.W1Size = 6.0;
                    this.W1Type = welds.GetWeld(data.w1_type);
                    this.W1Around = data.w1_around;
                    this.W1WType = data.w1_wtype;
                }
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.Message);
            }
        }

        public bool Insert(Beam primary)
        {
            bool result = false;
            try
            {
                Detail currentDetail = new Detail();

                currentDetail.Name = "Base plate (1004)";
                currentDetail.Number = 1004;

                currentDetail.UpVector = new Vector(0, 0, 0);
                currentDetail.PositionType = PositionTypeEnum.MIDDLE_PLANE;
                currentDetail.AutoDirectionType = AutoDirectionTypeEnum.AUTO-
DIR_GLOBAL_Z;
                currentDetail.DetailType = DetailTypeEnum.END;

                // Values from dialog

                currentDetail.SetAttribute("tpl1", this.Tpl2);
                currentDetail.SetAttribute("bpl1", this.Bpl2);
                currentDetail.SetAttribute("hpl1", this.Hpl2);

                currentDetail.SetAttribute("prefix_pos1", this.Pre2);
                currentDetail.SetAttribute("startno_pos1", this.Sno2);

                currentDetail.SetAttribute("mat", this.Mat2);
                currentDetail.SetAttribute("partname", this.PartName2);

                currentDetail.Class = this.EpClass;

                // Set weld details
                currentDetail.SetAttribute("w1_size", this.W1Size);
                currentDetail.SetAttribute("w1_type", this.W1Type);
                currentDetail.SetAttribute("w1_around", this.W1Around);
                currentDetail.SetAttribute("w1_wtype", this.W1WType);

                // We don't want to show other welds for detail so set them 0
                currentDetail.SetAttribute("w1_type2", 0);
```

```
                currentDetail.SetAttribute("w2_type", 0);
                currentDetail.SetAttribute("w2_type2", 0);

                currentDetail.SetPrimaryObject(primary);
                double refPointX = 0.0;
                currentDetail.SetReferencePoint(new TSG.Point(-refPointX / 2,
0, 0));

                result = currentDetail.Insert();
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.Message);
            }
            return result;
        }
    }
}
```

DEBASEPLATE1004.CS

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using System.Collections;
using System.IO;
using System.Globalization;

using Tekla.Structures.Plugins;
using TSG = Tekla.Structures.Geometry3d;
using Tekla.Structures.Model;
using Tekla.Structures.Geometry3d;
using Tekla.Structures;
using Tekla.Structures.Datatype;
using Distance = Tekla.Structures.Datatype.Distance;

namespace BasePlate
{
    class DeBasePlate1004
    {
        // Baseplate

        private double _tpl1;
        public double Tpl1
        {
            get { return _tpl1; }
            set { _tpl1 = value; }
        }

        private double _hpl1;
        public double Hpl1
        {
            get { return _hpl1; }
            set { _hpl1 = value; }
        }

        private double _bpl1;
        public double Bpl1
        {
            get { return _bpl1; }
            set { _bpl1 = value; }
        }

        private string _mat;
        public string Mat
        {
            get { return _mat; }
            set { _mat = value; }
        }

        private double _pos1;
        public double Pos1
        {
            get { return _pos1; }
            set { _pos1 = value; }
```

```
}

private string _partname;
public string Partname
{
    get { return _partname; }
    set { _partname = value; }
}

private int _epClass;
public int EpClass
{
    get { return _epClass; }
    set { _epClass = value; }
}

// Bolts
private double _rb1;
public double Rb1
{
    get { return _rb1; }
    set { _rb1 = value; }
}

private double _rb2;
public double Rb2
{
    get { return _rb2; }
    set { _rb2 = value; }
}

private string _lbd;
public string Lbd
{
    get { return _lbd; }
    set { _lbd = value; }
}

private double _rw1;
public double Rw1
{
    get { return _rw1; }
    set { _rw1 = value; }
}

private double _rw2;
public double Rw2
{
    get { return _rw2; }
    set { _rw2 = value; }
}

private string _lwd;
public string Lwd
{
    get { return _lwd; }
    set { _lwd = value; }
}
```

```
private int _lwtyp;
public int Lwtyp
{
    get { return _lwtyp; }
    set { _lwtyp = value; }
}

private int _lbtyp;
public int Lbtyp
{
    get { return _lbtyp; }
    set { _lbtyp = value; }
}

private double _lwa;
public double Lwa
{
    get { return _lwa; }
    set { _lwa = value; }
}

private double _lba;
public double Lba
{
    get { return _lba; }
    set { _lba = value; }
}

private double _diameter;
public double Diameter
{
    get { return _diameter; }
    set { _diameter = value; }
}

private string _screwdin;
public string Screwdin
{
    get { return _screwdin; }
    set { _screwdin = value; }
}

private double _tolerance;
public double Tolerance
{
    get { return _tolerance; }
    set { _tolerance = value; }
}

private int _threadIn;
public int ThreadIn
{
    get { return _threadIn; }
    set { _threadIn = value; }
}

private int _assemblyType;
public int AssemblyType
{
```

```
    get { return _assemblyType; }
    set { _assemblyType = value; }
}

private double _longHoleX;
public double LongHoleX
{
    get { return _longHoleX; }
    set { _longHoleX = value; }
}

private double _longHoleY;
public double LongHoleY
{
    get { return _longHoleY; }
    set { _longHoleY = value; }
}

private int _holeType;
public int HoleType
{
    get { return _holeType; }
    set { _holeType = value; }
}

private int _holeDirection;
public int HoleDirection
{
    get { return _holeDirection; }
    set { _holeDirection = value; }
}

private int _nw;
public int Nw
{
    get { return _nw; }
    set { _nw = value; }
}

private int _nb;
public int Nb
{
    get { return _nb; }
    set { _nb = value; }
}

private string _estring;
public string Estring
{
    get { return _estring; }
    set { _estring = value; }
}

// Additional Beam
private double _adist1;
public double Adist1
{
    get { return _adist1; }
    set { _adist1 = value; }
```

```
}

private string _prof;
public string Prof
{
    get { return _prof; }
    set { _prof = value; }
}

private double _pos2;
public double Pos2
{
    get { return _pos2; }
    set { _pos2 = value; }
}

private string _mat2;
public string Mat2
{
    get { return _mat2; }
    set { _mat2 = value; }
}

private string _partname2;
public string Partname2
{
    get { return _partname2; }
    set { _partname2 = value; }
}

private int _beampos;
public int Beampos
{
    get { return _beampos; }
    set { _beampos = value; }
}

//Weldings
private double _w1Size;
public double W1Size
{
    get { return _w1Size; }
    set { _w1Size = value; }
}

private int _w1Type;
public int W1Type
{
    get { return _w1Type; }
    set { _w1Type = value; }
}

private int _w1Around;
public int W1Around
{
    get { return _w1Around; }
    set { _w1Around = value; }
}
```

```
private int _w1WType;
public int W1WType
{
    get { return _w1WType; }
    set { _w1WType = value; }
}

private double _w2Size;
public double W2Size
{
    get { return _w2Size; }
    set { _w2Size = value; }
}

private int _w2Type;
public int W2Type
{
    get { return _w2Type; }
    set { _w2Type = value; }
}

private int _w2Around;
public int W2Around
{
    get { return _w2Around; }
    set { _w2Around = value; }
}

private int _w2WType;
public int W2WType
{
    get { return _w2WType; }
    set { _w2WType = value; }
}

private double _w3Size;
public double W3Size
{
    get { return _w3Size; }
    set { _w3Size = value; }
}

private int _w3Type;
public int W3Type
{
    get { return _w3Type; }
    set { _w3Type = value; }
}

private int _w3Around;
public int W3Around
{
    get { return _w3Around; }
    set { _w3Around = value; }
}

private int _w3WType;
public int W3WType
{
```

```
            get { return _w3WType; }
            set { _w3WType = value; }
        }

        private double _w4Size;
        public double W4Size
        {
            get { return _w4Size; }
            set { _w4Size = value; }
        }

        private int _w4Type;
        public int W4Type
        {
            get { return _w4Type; }
            set { _w4Type = value; }
        }

        private int _w4Around;
        public int W4Around
        {
            get { return _w4Around; }
            set { _w4Around = value; }
        }

        private int _w4WType;
        public int W4WType
        {
            get { return _w4WType; }
            set { _w4WType = value; }
        }


        public DeBasePlate1004(StructuresData data)
        {
            try
            {
                if (data != null)
                {
                    WeldCatalog welds = new WeldCatalog();

                    //Weldings paramaters
                    if (data.w1_size != Convert.ToDouble(Int32.MinValue))
this.W1Size = data.w1_size;
                    else this.W1Size = 6.0;
                    this.W1Type = welds.GetWeld(data.w1_type);
                    this.W1Around = data.w1_around;
                    this.W1WType = data.w1_wtype;

                    if (data.w2_size != Convert.ToDouble(Int32.MinValue))
this.W2Size = data.w2_size;
                    else this.W2Size = 6.0;
                    this.W2Type = welds.GetWeld(data.w2_type);
                    this.W2Around = data.w2_around;
                    this.W2WType = data.w2_wtype;
                    this.Adist1 = data.adist1;
                    this.Prof = data.prof;
                    this.Pos2 = data.pos2;
                    this.Mat2 = data.mat2;
```

```
                this.Partname2 = data.partname2;
                this.Estring = data.estring;

                this.Tpl1 = data.tpl1;
                this.Hpl1 = data.hpl1;
                this.Bpl1 = data.bpl1;

                if (data.mat != System.String.Empty) this.Mat = data.mat;
                else this.Mat = "S355J2H";

                this.Partname = data.partname;
                this.EpClass = data.epClass;
                this.Beampos = data.beampos;

                this.Estring = data.estring;
                this.Rb1 = data.rb1;
                this.Rb2 = data.rb2;
                this.Rw1 = data.rw1;
                this.Rw2 = data.rw2;

                if (data.lbd != System.String.Empty) this.Lbd = data.lbd;
                else this.Lbd = "500";

                if (data.lwd != System.String.Empty) this.Lwd = data.lwd;
                else this.Lwd = "550";

                this.Nb = data.nb;
                this.Nw = data.nw;
                this.Diameter = data.diameter;
                this.Screwdin = data.screwdin;
                this.Tolerance = data.tolerance;
                this.ThreadIn = data.thread_in;
                this.AssemblyType = data.assembly_type;
                this.Lwtyp = data.lwtyp;
                this.Lwa = data.lwa;
                this.Lbtyp = data.lbtyp;
                this.Lba = data.lba;
                this.LongHoleX = data.longholex;
                this.LongHoleY = data.longholey;
                this.HoleType = data.holetype;
                this.HoleDirection = data.holedirection;
            }
        }
        catch (Exception ex)
        {
            MessageBox.Show(ex.Message);
        }
    }


    private ArrayList ReturnDistances(string lwd)
    {
        CultureInfo invC = CultureInfo.InvariantCulture;

        ArrayList FinalSpacesCollection = new ArrayList();
        string[] Lwd_Values = lwd.Split(' ');

        for (int splitnum = 0; splitnum < Lwd_Values.Length; splitnum++)
        {
```

```
                string NewValue = Lwd_Values.GetValue(splitnum).ToString();

                if (NewValue.Contains("*"))
                {
                    int NumberItems = int.Parse(NewValue.Split('*').Get-
Value(0).ToString());
                    double SpaceValue = double.Parse(NewValue.Split('*').Get-
Value(1).ToString());
                    for (int counter1 = 0; counter1 < NumberItems; coun-
ter1++)
                    {
                        if (SpaceValue != 0)
                        {
                            FinalSpacesCollection.Add(SpaceValue);
                        }
                    }
                }
                else
                {
                    if (double.Parse(NewValue) != 0)
                    {
                        FinalSpacesCollection.Add(double.Parse(NewValue));
                    }
                }
            }
            return FinalSpacesCollection;
        }


        private Tekla.Structures.Datatype.DistanceList GetDistanceList(string
DistList)
        {
            return Tekla.Structures.Datatype.DistanceList.Parse(DistList,
                                                    System.Globaliza-
tion.CultureInfo.InvariantCulture,
                                                    Tekla.Struc-
tures.Datatype.Distance.UnitType.Millimeter);


        }

        public bool Insert(Beam pickedBeam, Point pickedPoint)
        {
            bool result = false;

            try
            {
                Detail currentDetail = new Detail();

                currentDetail.Name = "BasePlate (1004)";
                currentDetail.Number = 1004;
                currentDetail.DetailType = DetailTypeEnum.END;

                //Baseplate arvot dialogista

                currentDetail.SetAttribute("tpl1", this.Tpl1);
                currentDetail.SetAttribute("hpl1", this.Hpl1);
                currentDetail.SetAttribute("bpl1", this.Bpl1);
                currentDetail.SetAttribute("pos1", this.Pos1);
```

```
                currentDetail.SetAttribute("mat", this.Mat);
                currentDetail.SetAttribute("partname", this.Partname);
                currentDetail.Class = this.EpClass;

                // Bolts arvot dialogista
                currentDetail.SetAttribute("rb1", this.Rb1);
                currentDetail.SetAttribute("rb2", this.Rb2);
                currentDetail.SetAttribute("lbd", this.Lbd);
                currentDetail.SetAttribute("rw1", this.Rw1);
                currentDetail.SetAttribute("rw2", this.Rw2);
                currentDetail.SetAttribute("lwd", this.Lwd);
                currentDetail.SetAttribute("lwtyp", this.Lwtyp);
                currentDetail.SetAttribute("lbtyp", this.Lbtyp);
                currentDetail.SetAttribute("lba", this.Lba);
                currentDetail.SetAttribute("lwa", this.Lwa);
                currentDetail.SetAttribute("estring", this.Estring);

                currentDetail.SetAttribute("diameter", this.Diameter);
                currentDetail.SetAttribute("screwdin", this.Screwdin);
                currentDetail.SetAttribute("tolerance", this.Tolerance);
                currentDetail.SetAttribute("thread_in", this.ThreadIn);
                currentDetail.SetAttribute("assembly_type", this.Assem-
blyType);

                currentDetail.SetAttribute("nb", this.Nb);
                currentDetail.SetAttribute("nw", this.Nw);
                currentDetail.SetAttribute("longholex", this.LongHoleX);
                currentDetail.SetAttribute("longholey", this.LongHoleY);
                currentDetail.SetAttribute("holetype", this.HoleType);
                currentDetail.SetAttribute("holedirection", this.HoleDirec-
tion);

                //Additional beam arvot dialogista
                currentDetail.SetAttribute("adist1", this.Adist1);
                currentDetail.SetAttribute("prof", this.Prof);
                currentDetail.SetAttribute("mat2", this.Mat2);
                currentDetail.SetAttribute("partname2", this.Partname2);
                currentDetail.SetAttribute("pos2", this.Pos2);

                // Set weld details
                currentDetail.SetAttribute("w1_size", this.W1Size);
                currentDetail.SetAttribute("w1_type", this.W1Type);
                currentDetail.SetAttribute("w1_around", this.W1Around);
                currentDetail.SetAttribute("w1_wtype", this.W1WType);

                currentDetail.SetAttribute("w1_type2", 0);
                currentDetail.SetAttribute("w2_type", 0);
                currentDetail.SetAttribute("w2_type2", 0);
                currentDetail.SetAttribute("w3_type", 0);
                currentDetail.SetAttribute("w3_type2", 0);
                currentDetail.SetAttribute("w4_type", 0);
                currentDetail.SetAttribute("w4_type2", 0);

                double h = 0.0;
                double w = 0.0;
                pickedBeam.GetReportProperty("PROFILE.WIDTH", ref w);
                pickedBeam.GetReportProperty("PROFILE.HEIGHT", ref h);

                ArrayList aDistances = new ArrayList();
```

```
                DistanceList distanceList = new DistanceList();

                     if (Distance.CurrentUnitType == Distance.UnitType.Milli-
meter)
                     {
                            distanceList = DistanceList.Parse(this.Lwd,Cul-
tureInfo.InvariantCulture, Distance.UnitType.Millimeter);
                            foreach (Distance distance in distanceList)
                                  aDistances.Add(distance.ConvertTo(Dis-
tance.UnitType.Millimeter));
                     }
                     else if (Distance.CurrentUnitType == Distance.Unit-
Type.Inch)
                     {
                            distanceList = DistanceList.Parse(this.Lwd, Cul-
tureInfo.InvariantCulture, Distance.UnitType.Inch);
                            foreach (Distance distance in distanceList)
                                  aDistances.Add(distance.ConvertTo(Dis-
tance.UnitType.Inch));
                     }

                double DistSum = 0.0;
                for (int ii = 0; ii < aDistances.Count; ii++)
                {
                    double currentDist = (double)aDistances[ii];
                    DistSum += currentDist;
                }

                ArrayList aDistances1 = new ArrayList();
                DistanceList distanceList1 = new DistanceList();

                if (Distance.CurrentUnitType == Distance.UnitType.Millimeter)
                {
                    distanceList1 = DistanceList.Parse(this.Lbd, Cul-
tureInfo.InvariantCulture, Distance.UnitType.Millimeter);
                    foreach (Distance distance in distanceList1)
                        aDistances1.Add(distance.ConvertTo(Distance.Unit-
Type.Millimeter));
                }
                else if (Distance.CurrentUnitType == Distance.UnitType.Inch)
                {
                    distanceList1 = DistanceList.Parse(this.Lbd, Cul-
tureInfo.InvariantCulture, Distance.UnitType.Inch);
                    foreach (Distance distance in distanceList1)
                        aDistances1.Add(distance.ConvertTo(Distance.Unit-
Type.Inch));
                }

                double DistSum1 = 0.0;
                for (int ii = 0; ii < aDistances1.Count; ii++)
                {
                    double currentDist1 = (double)aDistances1[ii];
                    DistSum1 += currentDist1;
                }

                currentDetail.SetAttribute("lwtyp", -2147483648);
                currentDetail.SetAttribute("lbtyp", -2147483648);

                //Nurkka
```

```
                if (this.Beampos.Equals(1))
                {
                    double lwa_value = Rw1 + DistSum + Rw2 - w;
                    double lba_value = Rb1 + DistSum1 + Rb2 - h;

                    currentDetail.SetAttribute("lwa", lwa_value / 2);
                    currentDetail.SetAttribute("lba", lba_value / 2);
                    currentDetail.SetAttribute("estring", "1");

                    currentDetail.SetAttribute("w1_size", this.W1Size);
                    currentDetail.SetAttribute("w1_type", this.W1Type);
                    currentDetail.SetAttribute("w1_around", this.W1Around);
                    currentDetail.SetAttribute("w1_wtype", this.W1WType);

                    currentDetail.SetAttribute("w2_size", this.W2Size);
                    currentDetail.SetAttribute("w2_around", this.W2Around);
                    currentDetail.SetAttribute("w2_wtype", this.W2WType);
                    currentDetail.SetAttribute("w2_type", this.W2Type);

                }

                //Reuna
                if (this.Beampos.Equals(0))
                {
                    double lwa_value = Rw1 + DistSum1 + Rw2 - w;
                    currentDetail.SetAttribute("lwa", lwa_value / 2);

                    currentDetail.SetAttribute("w1_size", this.W1Size);
                    currentDetail.SetAttribute("w1_type", this.W1Type);
                    currentDetail.SetAttribute("w1_around", this.W1Around);
                    currentDetail.SetAttribute("w1_wtype", this.W1WType);

                    currentDetail.SetAttribute("w3_size", this.W2Size);
                    currentDetail.SetAttribute("w3_around", this.W2Around);
                    currentDetail.SetAttribute("w3_wtype", this.W2WType);
                    currentDetail.SetAttribute("w3_type", this.W2Type);
                }


                currentDetail.SetPrimaryObject(pickedBeam);

                currentDetail.SetReferencePoint(pickedPoint);

                result = currentDetail.Insert();
            }

            catch (Exception ex)
            {
                MessageBox.Show(ex.Message);
            }

            return result;
        }

    }
}
```

STRUCTURESDATA.CS

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using Tekla.Structures.Plugins;

namespace BasePlate
{
    public class StructuresData
    {
        //Base Plate Detail 1004

        //Plate
        [StructuresField("tpl1")]
        public double tpl1;

        [StructuresField("bpl1")]
        public double bpl1;

        [StructuresField("hpl1")]
        public double hpl1;

        [StructuresField("pos1")]
        public double pos1;

        [StructuresField("mat")]
        public string mat;

        [StructuresField("partname")]
        public string partname;

        [StructuresField("beampos")]
        public int beampos;



        //Additional beam
        [StructuresField("adist1")]
        public double adist1;

        [StructuresField("prof")]
        public string prof;

        [StructuresField("pos2")]
        public double pos2;

        [StructuresField("mat2")]
        public string mat2;

        [StructuresField("partname2")]
        public string partname2;

        [StructuresField("startno_pos1")]
        public int GussetStartNoPos;

        [StructuresField("prefix_pos1")]
```

```
public string GussetPrefixPos;

[StructuresField("epClass")]
public int epClass;


//Weldings
[StructuresField("w1_size")]
public double w1_size;

[StructuresField("w1_type")]
public int w1_type;

[StructuresField("w1_around")]
public int w1_around;

[StructuresField("w1_wtype")]
public int w1_wtype;

[StructuresField("w2_size")]
public double w2_size;

[StructuresField("w2_type")]
public int w2_type;

[StructuresField("w2_around")]
public int w2_around;

[StructuresField("w2_wtype")]
public int w2_wtype;

[StructuresField("w3_size")]
public double w3_size;

[StructuresField("w3_type")]
public int w3_type;

[StructuresField("w3_around")]
public int w3_around;

[StructuresField("w3_wtype")]
public int w3_wtype;

[StructuresField("w4_size")]
public double w4_size;

[StructuresField("w4_type")]
public int w4_type;

[StructuresField("w4_around")]
public int w4_around;

[StructuresField("w4_wtype")]
public int w4_wtype;

//Bolts tab_page atribuutit

[StructuresField("diameter")]
public double diameter;
```

```
[StructuresField("screwdin")]
public string screwdin;

[StructuresField("tolerance")]
public double tolerance;

[StructuresField("thread_in")]
public int thread_in;

[StructuresField("assembly_type")]
public int assembly_type;

[StructuresField("longholex")]
public double longholex;

[StructuresField("longholey")]
public double longholey;

[StructuresField("holetype")]
public int holetype;

[StructuresField("holedirection")]
public int holedirection;

[StructuresField("nb")]
public int nb;

[StructuresField("nw")]
public int nw;

[StructuresField("lwtyp")]
public int lwtyp;

[StructuresField("lwa")]
public double lwa;

[StructuresField("lbtyp")]
public int lbtyp;

[StructuresField("lba")]
public double lba;

[StructuresField("rb1")]
public double rb1;

[StructuresField("rb2")]
public double rb2;

[StructuresField("rw1")]
public double rw1;

[StructuresField("rw2")]
public double rw2;

[StructuresField("lbd")]
public string lbd;

[StructuresField("lwd")]
```

```
        public string lwd;

        [StructuresField("estring")]
        public string estring;

    }
}
```

WELDCATALOG.CS

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace BasePlate
{
    class WeldCatalog
    {
        List<int> welds;

        public WeldCatalog()
        {
            welds = new List<int>
            {
                0, 10, 4
            };
        }

        public int GetWeld(int selectedIndex)
        {
            return welds[selectedIndex];
        }
    }
}
```

BASEPLATEUI.CS

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using Tekla.Structures.Plugins;

namespace BasePlate
{
    class BasePlateUI
    {
        public const string BasePlatePlugin = @"" +
            @"page(""TeklaStructures"","""")" + "\n" +
            "{\n" +
                "joint(1, BasePlate)\n" +
                "{\n" +
                    @"helpurl(""BasePlate.chm::/general.html"")" + "\n" +

@"tab_page("""", ""Picture"", 1)" + "\n" +
                    "{\n" +

                        @"picture(""BasePlate_ShearKey.bmp"", 173, 126, 180,
50)" + "\n" +
                        @"parameter("""", adist1, distance, number, 440, 160,
50)" + "\n" +
                        @"attribute(""beampos"", """", option, ""%s"",none,
none, ""0.0"", ""0.0"", 200,250,215)" + "\n" +
                            "{\n" +
                                @"value(""BasePlate_Beam_Position_1.xbm"",
2)"  + "\n" +
                                @"value(""BasePlate_Beam_Position_2.xbm"",
1)" + "\n" +
                                @"value(""BasePlate_Beam_Position_3.xbm"",
0)" + "\n" +
                            "}\n" +
                    "}\n" +

@"tab_page("""", ""Parts"", 2)" + "\n" +
                    "{\n" +
                        @"part (   ""j_plate"", tpl1, , , pos1, mat,  , part-
name)" + "\n" +
                        @"profile (""j_additional_beam"", prof, pos2, mat2,
, partname2)" + "\n" +
                        @"attribute(, ""Class"", label2, ""%s"", none, none,
""0.0"", ""0.0"", 840, 4)" + "\n" +
                        @"parameter("""", ""epClass"", integer, number, 840,
25, 50)" + "\n" +

                        //@"parameter("""", ""rw1"", distance, number, 490,
130, 50)" + "\n" +
                        //@"parameter("""", ""lwd"", distance, number, 590,
130, 50)" + "\n" +
                        //@"parameter("""", ""rw2"", distance, number, 690,
130, 50)" + "\n" +

                        //@"parameter("""", ""rb1"", distance, number, 770,
175, 50)" + "\n" +
```

```
                        //@"parameter("""", ""lbd"", distance, number, 770,
250, 50)" + "\n" +
                        //@"parameter("""", ""rb2"", distance, number, 770,
325, 50)" + "\n" +

                        //@"picture(""BasePlate_tbh1.bmp"", 520, 200, 50,
150)" + "\n" +
                    "}\n" +
@"tab_page("""", ""Weldings"", 3)" + "\n" +
                    "{\n" +
                        @"attribute(""label_only1"", ""1."", label2, ""%s"",
none, none, ""0.0"", ""0.0"", 20, 80)" + "\n" +
                        @"attribute("""", ""jd_weld_Size"", label2, ""%s"",
none, none, ""0.0"", ""0.0"", 80, 50)" + "\n" +
                        @"parameter("""", ""w1_size"", distance, number, 80,
80, 64)" + "\n" +
                        @"attribute("""", ""jd_weld_Type1"", label2, ""%s"",
none, none, ""0.0"", ""0.0"", 180, 50)" + "\n" +
                        @"attribute(""w1_type"", """", option, ""%s"", none,
none, ""0.0"", ""0.0"", 180, 80, 96)" + "\n" +
                        "{\n" +
                            @"value(""w_type_0.xbm"", 1)" + "\n" +
                            @"value(""w_type_10.xbm"", 0)" + "\n" +
                            @"value(""w_type_4.xbm"", 0)" + "\n" +
                        "}\n" +
                        @"attribute("""", ""jd_weld_Edge_around"", label2,
""%s"", none, none, ""0.0"", ""0.0"", 315, 50)" + "\n" +
                        @"attribute(""w1_around"", """", option, ""%s"",
none, none, ""0.0"", ""0.0"", 315, 80, 64)" + "\n" +
                        "{\n" +
                            @"value(""w_around_0.xbm"", 0)" + "\n" +
                            @"value(""w_around_1.xbm"", 1)" + "\n" +
                        "}\n" +
                        @"attribute("""", ""jd_weld_Wshop_site"", label2,
""%s"", none, none, ""0.0"", ""0.0"", 420, 50)" + "\n" +
                        @"attribute(""w1_wtype"", """", option, ""%s"", none,
none, ""0.0"", ""0.0"", 420, 80, 64)" + "\n" +
                        "{\n" +
                            @"value(""w_workshop_1.xbm"", 0)" + "\n" +
                            @"value(""w_workshop_0.xbm"", 1)" + "\n" +
                        "}\n" +
                        @"attribute("""", ""jd_weld_Wshop_site"", label2,
""%s"", none, none, ""0.0"", ""0.0"", 420, 50)" + "\n" +
                        @"attribute("""", ""Reference text"", label2, ""%s"",
none, none, ""0.0"", ""0.0"", 530, 50)" + "\n" +
                        @"parameter("""", ""reference_text_1"", string, text,
530, 80, 200)" + "\n" +
                        @"attribute(""label_only2"", ""2."", label2, ""%s"",
none, none, ""0.0"", ""0.0"", 20, 105)" + "\n" +
                        @"parameter("""", ""w2_size"", distance, number, 80,
105, 64)" + "\n" +
                        @"attribute(""w2_type"", """", option, ""%s"", none,
none, ""0.0"", ""0.0"", 180, 105, 96)" + "\n" +
                        "{\n" +
                            @"value(""w_type_0.xbm"", 1)" + "\n" +
                            @"value(""w_type_10.xbm"", 0)" + "\n" +
                            @"value(""w_type_4.xbm"", 0)" + "\n" +
                        "}\n" +
```

```
                        @"attribute(""w2_around"", """", option, ""%s"",
none, none, ""0.0"", ""0.0"", 315, 105, 64)" + "\n" +
                        "{\n" +
                            @"value(""w_around_0.xbm"", 0)" + "\n" +
                            @"value(""w_around_1.xbm"", 1)" + "\n" +
                        "}\n" +
                        @"attribute(""w2_wtype"", """", option, ""%s"", none,
none, ""0.0"", ""0.0"", 420, 105, 64)" + "\n" +
                        "{\n" +
                            @"value(""w_workshop_1.xbm"", 0)" + "\n" +
                            @"value(""w_workshop_0.xbm"", 1)" + "\n" +
                        "}\n" +
                        @"parameter("""", ""reference_text_2"", string, text,
530, 105, 200)" + "\n" +

                        @"picture(""BasePlate_Weldings.bmp"", 480, 200, 100,
210)" + "\n" +
                    "}\n" +
@"tab_page("""", ""Bolts"", 5)" + "\n" +
                    "{\n" +
                        @"parameter("""", ""diameter"", bolt_size, number,
170, 5, 150)" + "\n" +
                        @"parameter("""", ""screwdin"", bolt_standard, text,
170, 30, 150)" + "\n" +
                        @"parameter("""", ""tolerance"", distance, num-
ber,170,55,150)" + "\n" +
                        @"attribute(""thread_in"", """", option, ""%s"",
none, none, ""0.0"", ""0.0"",170,80,150)" + "\n" +
                        "{\n" +
                            @"value(""j_Default"", 2)" + "\n" +
                            @"value(""j_no"", 0)" + "\n" +
                            @"value(""j_yes"", 1)" + "\n" +
                        "}\n" +
                        @"attribute(""assembly_type"", """", option, ""%s"",
none, none, ""0.0"", ""0.0"",170,111,150)" + "\n" +
                        "{\n" +
                            @"value(""j_Default"", 2)" + "\n" +
                            @"value(""j_site"", 1)" + "\n" +
                            @"value(""j_workshop"", 0)" + "\n" +
                        "}\n" +
                        @"attribute(""bolt_size_lb"", ""j_bolt_size"", la-
bel2, ""%s"",  none, none, ""0.0"", ""0.0"",20,5)" + "\n" +
                        @"attribute(""bolt_standard_lb"", ""j_bolt_stand-
ard"", label3, ""%s"",  none, none, ""0.0"", ""0.0"",20,30)" + "\n" +
                        @"attribute(""tolerance_lb"", ""j_tolerance"", la-
bel2, ""%s"",  none, none, ""0.0"", ""0.0"",20,55)" + "\n" +
                        @"attribute(""thread_in_material_lb"",
""j_thread_in_material"", label3, ""%s"",  none, none, ""0.0"",
""0.0"",20,80)" + "\n" +

                        //@"attribute(""lwtyp"", """", option, ""%s"",  none,
none, ""0.0"", ""0.0"",250,185,90)" + "\n" +
                        //"{\n" +
                            //@"value(""j_Default"", 2)" + "\n" +
                            //@"value("""",0)" + "\n" +
                            //@"value(""j_left"", 0)" + "\n" +
                            //@"value(""j_middle"", 1)" + "\n" +
                            //@"value(""j_right"", 0)" + "\n" +
                        //"}\n" +
```

```
                        //@"parameter("""", ""lwa"", distance, num-
ber,250,208,60)" + "\n" +
                        //@"attribute(""lbtyp"", """", option, ""%s"",  none,
none, ""0.0"", ""0.0"",100,284,90)" + "\n" +
                        //"{\n" +
                            //@"value(""j_Default"", 2)" + "\n" +
                            //@"value("""", 0)" + "\n" +
                            //@"value(""j_top"", 0)" + "\n" +
                            //@"value(""j_middle"", 1)" + "\n" +
                            //@"value(""j_below"", 0)" + "\n" +
                        //"}\n" +
                        //@"parameter("""", ""lba"", distance, num-
ber,110,315,60)" + "\n" +

                        //@"picture(""slotted_hole_extended"", 155, 115, 326,
0)" + "\n" +
                        //@"parameter("""", ""longholex"", distance, num-
ber,444,20,60)" + "\n" +
                        //@"parameter("""", ""longholey"", distance, num-
ber,533,70,60)" + "\n" +
                        //@"parameter("""", ""holetype"", hole_type,
text,463,100,150)" + "\n" +
                        //@"parameter("""", ""holedirection"", hole_direc-
tion, text,463,125,150)" + "\n" +
                        //@"attribute(""hole_type_lb"", ""albl_Hole_Type_"",
label3 , ""%s"", none, none , ""0.0"", ""0.0"", 332, 100)" + "\n" +
                        //@"attribute(""hole_direction_lb"", ""albl_Ro-
tate_Slots_"", label3 , ""%s"", none, none , ""0.0"", ""0.0"", 332, 125)" +
"\n" +

                        @"picture(""bolts.bmp"", 146, 172, 180, 235)" + "\n"
+
                        @"parameter("""", ""rb1"", distance, num-
ber,410,265,60)" + "\n" +
                        @"parameter("""", ""nb"", integer, num-
ber,410,315,40)" + "\n" +
                        @"parameter("""", ""lbd"", distance_list_no_toggle,
text,455,315,135)" + "\n" +
                        @"parameter("""", ""rb2"", distance, num-
ber,410,365,60)" + "\n" +
                        @"parameter("""", ""rw1"", distance, num-
ber,190,419,60)" + "\n" +
                        @"parameter("""", ""rw2"", distance, num-
ber,310,419,60)" + "\n" +
                        @"parameter("""", ""nw"", integer, num-
ber,190,445,40)" + "\n" +
                        @"parameter("""", ""lwd"", distance_list_no_toggle,
text,235,445,135)" + "\n" +
                        //@"parameter("""", ""estring"", dis-
tance_list_no_toggle, text,455,445,135)" + "\n" +
                    "}\n" +
                "}\n" +
            "}\n";
    }
}
```