Saimaan ammattikorkeakoulu
Tekniikka, Lappeenranta
Tietotekniikka
Ohjelmistotekniikka

Jan Pount

SAIMAAN ERISTYS OY:N TELINELASKENTAJÄRJESTELMÄ

Opinnäytetyö 2010

TIIVISTELMÄ

Jan Pount

Saimaan Eristys Oy:n telinelaskentajärjestelmä, 46 sivua, 1 liite
Saimaan ammattikorkeakoulu, Lappeenranta
Tekniikka, Tietotekniikan koulutusohjelma
Ohjelmistotekniikka
Opinnäytetyö 2010
Ohjaajat: Lehtori Martti Ylä-Jussila, Saimaan ammattikorkeakoulu,
Tekninen johtaja Antero Väätämöinen, Saimaan Eristys Oy

Opinnäytetyö on tehty Saimaan Eristys Oy:lle, jonka toimialaan kuuluvat erilaisten pintojen ääni-, kylmä- ja lämpöeristykset sekä erilaisten rakennustelineiden suunnittelu ja pystytys teollisuuden eri kohteisiin.

Opinnäytetyön tavoitteena on kehittää asiakasyritykselle helppokäyttöinen rakennustelineiden suunnittelujärjestelmä. Sovelluksen päätehtävänä on erilaisten rakennusten sekä niiden ympärille tai sisälle rakennettavien telineiden mallintaminen, tarvikemäärien laskenta. Muihin ominaisuuksiin kuuluvat tietojen ylläpito.

Järjestelmä on ohjelmoitu pääasiassa C#-ohjelmointikielen sekä SQL-tietokantakielen avulla. Toteutuksessa on käytetty Apache-palvelinta, johon oli integroitu MySQL-tietokantapalvelin sekä PHP-tulkki. Visuaalisessa toteutuksessa on käytetty Adobe Photoshop-kuvakäsittelyohjelmaa ja Visual Studion työkaluja.

Opinnäytetyö on rajattu ja painotettu käyttöliittymän suunnitteluun sekä mallinnussovelluksen kehitykseen. Opinnäytetyön lopputuloksena on saatu aikaan määrittelydokumentti sekä prototyyppi ohjelmasta, joka kattaa yleisimmät rakennuskohteet ja niiden rakennustelineiden suunnittelun.

Avainsanat: 3D-mallinnus, DirectX, DirectX:n piirto-ominaisuus, MySQL, MOV’AMP, C#

ABSTRACT
Jan Pount
Saimaan Eristys Oy:n Scaffolding Calculating Program, 46 pages, 1 appendi
Saimaa University of Applied Sciences, Lappeenranta
Technology, Degree Programme in Information Technology
Software Engineering
Bachelor thesis 2010
Instructors: Lecturer Martti Ylä-Jussila, Saimaa University of Applied Sciences
Technical manager Antero Väätämöinen, Saimaan Eristys Oy

This thesis report contains information about developing modeling software for an existing company named Saimaan Eristys Oy. This company produces and sells sheet-metal. However, the main task is installation of sound, cold and heat insulations for different surfaces. Also this company mounts different scaffoldings around and inside various industry objects.

So far Saimaan Eristys Oy has managed all the calculations needed to order a proper amount of scaffolds with a spreadsheet computation program. For calculating a proper amount of scaffolds needed in a building site the company used different algorithms. Each algorithm is suitable only with a specific target object. The customer desires new students to develop new algorithms which would be suitable with many different targets.

The previous version of this program was made by Miika Puroharju in 2009. The program contained data managements and a section where the user can apply different algorithms for counting scaffolds.

The new version of the software has been developed through this project by two students. Both have made this project as a thesis. According to the feasibility study it has been agreed that developing new algorithms would be just a pure waste of time. There are target structures of many different types. Developing general algorithm to match demands of many objects would be almost impossible. It has been agreed that the new project group will create a program, which contains both a universal modeling feature for common types of constructions and management tools. The new user-friendly software should help customer's employees in planning tasks. The software's main function is to model target structures and scaffolds, which are placed inside or outside the target objects. The software should be able to calculate needed amount of scaffolds.

As a coding language this project used C#- language and database commands have been formed with SQL. The project required Apache-server, which contains MySQL- servers and PHP-interpreter. For a visual implementation Adobe Photoshop CS3 has been used.

The project was confined to both interface and modeling tool development. This document explains in details the project stages.

Keywords: 3D-modeling, DirectX, DirectX Draw, MySQL, MOV'AMP, C#.

SISÄLLYS

TERMIT JA LYHENTEET

Broker	Mercus Softwaren järjestelmä, jolla voidaan hallita yrityksen tarjouspyyntörekisteriä, tarjouslaskentaa, aikataulua ja jälkilaskentaa.
Elementti	Teline-elementti, joka koostuu rakennustelineen osista.
Kohde	Rakennus tai muu vastaava, jonka ympärille tai sisäpuolelle kootaan valmiit elementit.
Kärki	Vektorin kärki.
Lähetyslista	Taulukko, jossa on tilattavien telineenosien numerot, nimet, painot ja tarvittavat kappalemäärät.
Malli	Malli eli telinelaskelma sisältää 3D-mallinnuksen kohteesta ja telineestä, pitää sisällään myös käyttäjän piirrokselle antamat ominaisuudet kuten telineen nimen ja korkeuden (ei lasketa 3D-mallista).
MD5	Merkkijonojen vakiomittaiseen 128-bittiseen tiivistykseen käytettävä algoritmi, käytännöllinen salasanojen tallennuksessa tietokantaan.
MySQL	SQL-tietokannan hallintajärjestelmä, joka sisältää rajapinnan usealle eri ohjelmointikielelle.
Objekti	Objekti on kokoelma kohteen tai elementtien osia kaksi ja kolmeulotteisessa avaruudessa.
Offline	Tietokone ei ole yhdistetty verkkoon, jolloin se ei voi lähettää eikä vastaanottaa tietopaketteja verkosta.

Online	Tietokone on onnistuneesti yhdistetty verkkoon ja kykenee lähettämään ja vastaanottamaan tietopaketteja.
Palvelin	Tietokone, johon on asennettu MySQL-palvelin, mahdollisesti phpMyAdmin-graafinen tietokantaliittymä sekä tarvittavat ajurit. Tietokone varastoi Teline-tietokannan tietoja.
PDF	Portable Document Format, käyttöjärjestelmäriippumaton siirrettävä tiedostomuoto, jota käytetään pääasiallisesti sähköiseen julkaisemiseen, tulostamiseen ja painamiseen.
phpMyAdmin	Selaimen kautta käytettävä MySQL-tietokannan hallintatyökalu.
Prototyyppi	Prototyyppimallin tulos, joka on toiminnaltaan epätäydellinen. Tulos selventää asiakkaan tarpeet sekä visuaalisesta että toiminnallisesta näkökulmasta.
Refaktorointi	Prosessi, joka vaihtaa tietokoneohjelman lähdekoodia muuttamatta sen toiminnallisuutta.
Sovellus	Teline-sovellus, ellei käsitettä yhdistetä asiayhteydessä muuhun sovellukseen.
SQL	Structured Query Language, standardoitu ja yleisin käytössä oleva kyselykieli, jolla käsitellään relaatiotietokantoja.

Teline-tietokanta	Tietokantapalvelin, johon järjestelmän koneet ovat yhteydessä. Tietokanta sisältää ohjelman tarvitsemat tiedot.
UML	Unified Modeling Language, standardoitu graafinen mallinnuskieli, joka käsittää erilaisia kaavioita. Kaavioilla kuvataan rakennetta, käyttäytymistä ja vuorovaikutusta.
Vektori	Perusmuoto 3D-mallinnuksessa, joka muodostuu alku- ja loppukärjestä.
Yhdyshenkilö	Henkilö, joka toimii linkkinä asiakasyritykseen.
2D ja projektio	Kaksiulotteinen kuva, joka esittää 3D-kuvaa tietystä suunnasta, eli projektio. Tässä projektissa on käytetty kolmea suuntaa: ylä-, sivu- ja etupuoli.
3D	Kolmiulotteinen kuva, jota voidaan tarkastella eri kulmista.

1 JOHDANTO

Opinnäytetyön aiheena on telinesuunnitteluohjelman kehitys Saimaan Eristys Oy:n käyttöön (1).

Asiakasyrityksen toimialaan kuuluvat muun muassa erilaisten telinerakenteiden suunnittelu ja pystyttäminen. Telineiden suunnitteluun kuuluu huomattavan paljon työtä ja siksi se halutaan automatisoida.

Opinnäytetyöprojektin tavoitteena on kehittää ohjelma, jolla asiakasyrityksen työntekijät voivat mallintaa tarkasti tarvittavat kohderakenteet ja telineosat. Sovelluksen ominaisuuksiin kuuluvat myös telinerakenteiden määrälaskenta sekä erilaisten tietojen ylläpito ja taltiointi. Ohjelman on oltava helppokäyttöinen, koska yrityksen työntekijöiden kokemus tietokoneohjelmista on niukka.

Opinnäytetyö keskittyy mallinnussovelluksen kehittämiseen. Muihin kehitettyihin ominaisuuksiin kuuluvat erilaiset hallintaominaisuudet.

Tausta ja tarkoitus

Miika Puroharju oli tehnyt ohjelmoinnin harjoitustyönä telinelaskentaohjelman ensimmäisen prototyypin vuoden 2009 alkupuolella. Kyseiset dokumentit ja prototyyppi koskivat vain lieriömäisten rakennelmien (rakennukset, säiliöt) ympärille rakennettavien telineosien määrälaskentaa.

Kuva 1.1 Rakennustelineet

Tämän opinnäytetyön tehtävänä on kehittää ohjelma, joka tukisi kaikkia yleisimpiä rakennustyyppejä. Rakennuskohteet voivat olla hyvin erimuotoisia ja -mittaisia, joten yleispätevän laskentamallin kehittäminen olisi hyvin vaikeaa, miltei mahdotonta. Jokainen rakennustyyppi on uniikki ja jokaiselle rakennukselle olisi tehtävä oma laskenta-algoritmi, mikä pätee ainoastaan siihen rakennusmalliin, jotta saataisiin tarkat lukemat. Esitutkimuksessa tulimme siihen tuloksen, että on kehitettävä mallinnustyökalun, jolla mallinnettaisiin rakennuskohteet ja sen ympärille koottavat telineet ja joka mahdollistaisi tarvittavien telineosien laskennan.

2 ASIAKKAAN TOIMINNAN KUVAUS

Saimaan Eristys Oy on teollisuusyritys, joka on perustettu vuonna 1982. Yritys työllistää noin 100 erityisalan ammattilaista. Yritys on erikoistunut lämpö- ja kylmä- ja äänieristyksien asennukseen putkistoihin. Pääasiassa eristettäviin kohteisiin kuuluvat säiliöt, kattilat, kanavat, erilaiset laitteet sekä sähkösuotimet. Yritys myös valmistaa ja myy peltiosia sekä tekee telineasennuksia.

Tällä hetkellä yrityksellä on vain yksi toimipiste, joka sijaitsee Lappeenrannan Eteläkadulla.

Yrityksen markkina-alue on laaja. Yrityksellä on asiakkaita paperi- ja sellu-, kemian-, elintarvike- ja mekaanisessa puunjalostusteollisuudessa sekä voimalaitoksissa.

Rakennustelineiden suunnittelu ja rakentaminen

Saimaan Eristys Oy:n nykyinen telinesuunnitteluprosessi on monivaiheinen. Ennen urakkasopimuksen tekoa yrityksen työntekijät käyvät paikan päällä mittamassa sovitun kohderakennuksen ulottuvuudet. Saatujen mittojen perusteella lasketaan tarvittavien telineosien lukumäärät, jonka jälkeen tehdään tilaus telinevalmistajalle (2.). Lieriömuotoiselle kohderakennukselle on olemassa laskenta-algoritmi, joka helpottaa laskentaprosessia. Kaikki laskennat hoidetaan taulukkolaskentaohjelmalla, joten uusi sovellus olisi tarpeen.

Valmiin telineeseen tarvitaan telinekortti, josta käy ilmi telineen pystyttäjä, tarkastaja sekä tapahtuma-ajankohta. Lisäksi kortissa ilmoitetaan kuorman kesto. Kortti käännettynä toisinpäin tarkoittaa, että teline on käyttökiellossa.

Tarvittavia tulosteita ovat telinedokumentit, asennusohjeet sekä tarjouksen tuotossa tarjouspyynnön tulostus. Tarjousten tuottaminen eli hinnoittelu kuuluu

3 OHJELMISTON KEHITYSMENETELMÄT

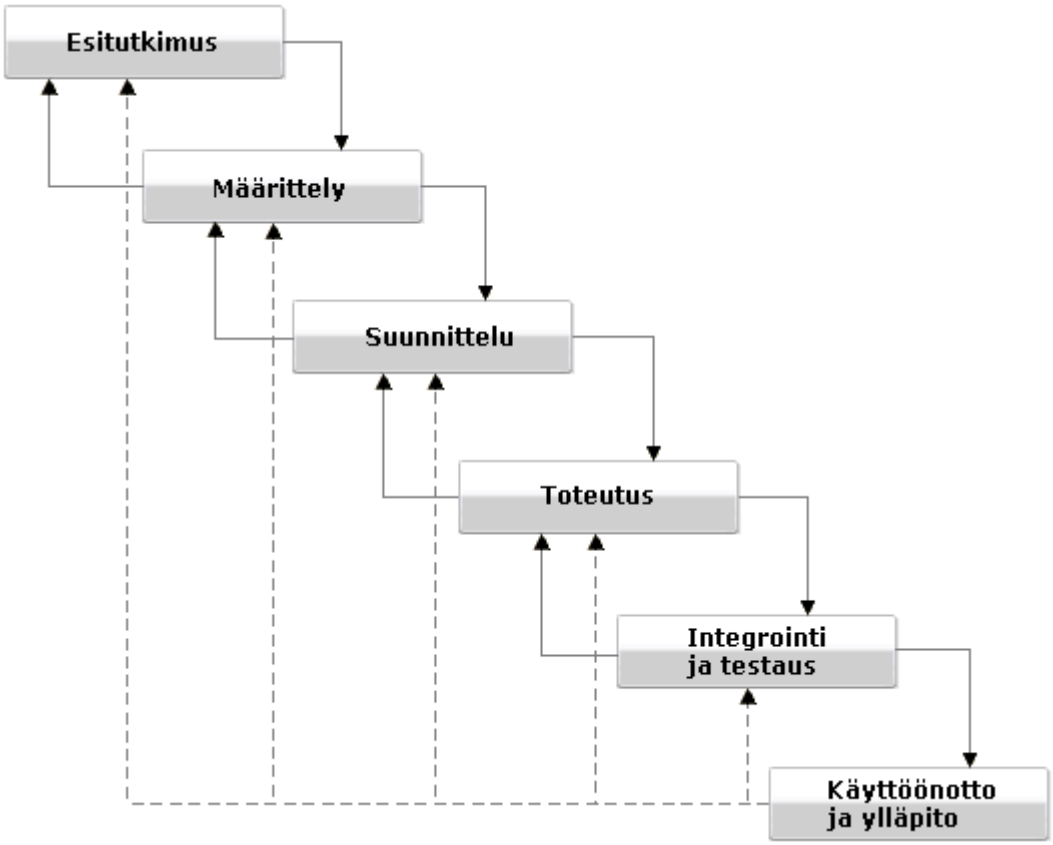
Ohjelmistokehitys esiteltiin terminä ensimmäisen kerran 1960-luvulla ohjelmistokriisin aikaan. Kriisin taustalla oli se, etteivät entiset menetelmät enää sopineet uusille, tehokkaammille tietokoneille, vaan kehitysprojektit olivat jopa vuosia myöhässä, ylittivät kustannuksensa ja tuloksena syntyneet ohjelmistot olivat hankalia ylläpitää. (3.) Ohjelmistokriisin jälkeen ohjelmien kehitystyöhön alettiin etsiä avuksi prosesseja ja malleja, joiden avulla voitaisiin kehittää tehokkaampia ohjelmistojen tuotantomenetelmiä. Ensimmäiset ohjelmistokehitysmenetelmät syntyivät muista insinööritieteistä haettujen menetelmämallien pohjalta. (4.)

3.1 Vesiputousmalli

Vesiputousmalli pidetään ensimmäisenä varsinaisena prosessimallina. Kyseinen malli jakaa prosessin lineaarisiin vaiheisiin. Vesiputousmalli perustuu hyvään dokumentaatioon sekä syntyneiden dokumenttien huolelliseen tarkasteluun.

Winston Roycea on ensimmäisenä määrittänyt vesiputousmallia 1970-luvulla. Malli on nopeasti saavuttanut suosiota, sillä se vastasi senaikaisten ohjelmistoammattilaisten käsitys oikeasta prosessista. (5.)

Mallia on helppo mieltää. Se koostuu kuvassa (Kuva 3.1) esitetyistä prosesseista.



Kuva 3.1 Vesiputousmallin etenemisprosessit.

Esitutkimus on ensimmäinen dokumentti, joka syntyy projektia vastaanottaessa. Esitutkimus vastaa kysymyksiin, miksi järjestelmä tarvitaan, mitä järjestelmän tulisi tehdä, miten järjestelmä tai sen tietyt prosessit tekevät ja onko hanke ylipäätään kannattava aloittaa. Esitutkimuksesta kirjoittaessa on tärkeintä asiakkaan todellisten tarpeiden ymmärtäminen.

Määrittelyvaiheessa johdetaan asiakasvaatimuksista ohjelmistovaatimukset. Määrittelyn tuloksena syntyy toiminnallinen määrittely, joka kuvaa tarkasti asiakkaan tarpeet, toteutusmenetelmät sekä laitteistovaatimukset. Myöhempien vaiheiden epäselvyyksien välttämiseksi dokumentti kirjoitetaan mahdollisimman tarkasti, kuvataan jokaisen toiminnon yksityiskohtaiset kuvaukset. Tietokannat sekä rajapinnat on kuvattava laajasti, jotta tuleva soveltaminen on helppoa.

Suunnittelun tarkoituksena on muuntaa toiminnallinen määrittely tekniseksi määrittelyksi. Suunnittelun tehtävä on suunnitella kuvattujen toimintojen

toteutus. Suunnitteluvaihe jaetaan kahteen osaan: arkkitehtuuri- ja moduulisuunnitteluun.

Toteutus on vaihe, jolloin toiminnallisen määrittelyn sekä suunnittelun toiminnot toteutetaan jollakin ohjelmointikielellä. Toteutetaan määritysten mukaiset moduulit ja yhdistetään ne toimivaksi kokonaisuudeksi. Toteutusvaiheessa joudutaan huomioimaan laatuun vaikuttavat seikat, muun muassa toiminnallisuus, luotettavuus, siirrettävyys sekä ylläpidettävyys.

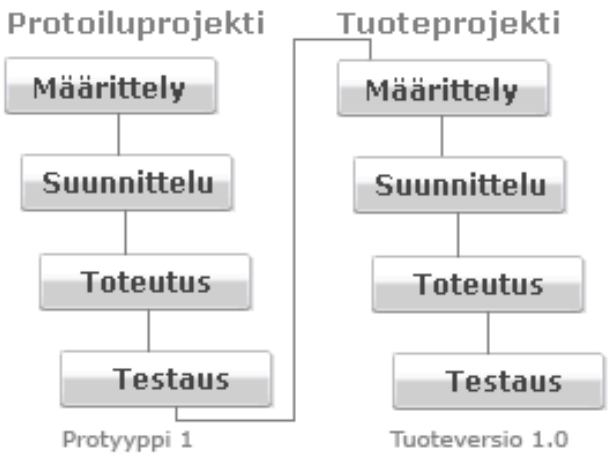
Testauksen tarkoituksena on löytää ohjelmistossa esiintyvät virheet. Testaussuunnitelma luodaan ennen toteutusvaihetta.

Kun järjestelmä on testattu ja toteutettu, voidaan se ottaa käyttöön. **Käyttöönottoon** kuuluvat muun muassa seuraavat olennaiset seikat: olemassa olevien tietojen siirtäminen uuteen järjestelmään, vanhan järjestelmän mahdollinen rinnakkaiskäyttö sekä käyttäjien ja ylläpitäjien kouluttaminen.

Ylläpito ei varsinaisesti kuulu kehitysprojektiin vaan on käytönaikaista toimintaa. Sen tavoitteina ovat muun muassa ohjelmistossa ilmenneiden virhetilanteiden korjaaminen sekä asiakkaan tarpeiden mukaisen muotoon päivittäminen.

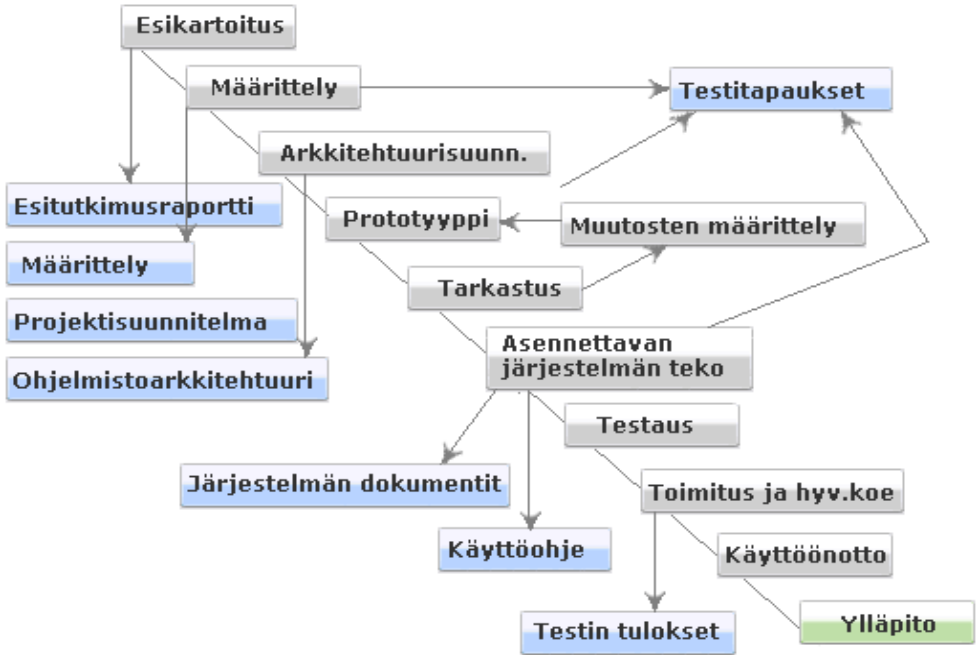
3.2 Protoilumalli

Protoilumallille on ominaista luoda yksittäisiä prototyyppejä, jotka soveltuvat erityisesti uusien teknisten ratkaisujen kokeiluun sekä asiakkaan epäselvien vaatimuksien selvittämiseen. Prototyyppimallista syntyvä prototyyppi on yleensä nopeasti luotu havainnollinen demo, joka toiminnallisuudellaan ei kata haluttuja toimintoja vaan selventää asiakastarpeet. Jos prototyyppi on hyväksytty, voidaan siitä jalostaa toimiva komponentti (Kuva 3.2).



Kuva 3.2 Protoilumallin tyyppi 1

Vaihtoehtoisesti voidaan aloittaa sovelluksen kehitys alusta ottaen huomioon kaikki prototyypissä ilmi tulleet positiiviset sekä negatiiviset seikat. (Kuva 3.3)



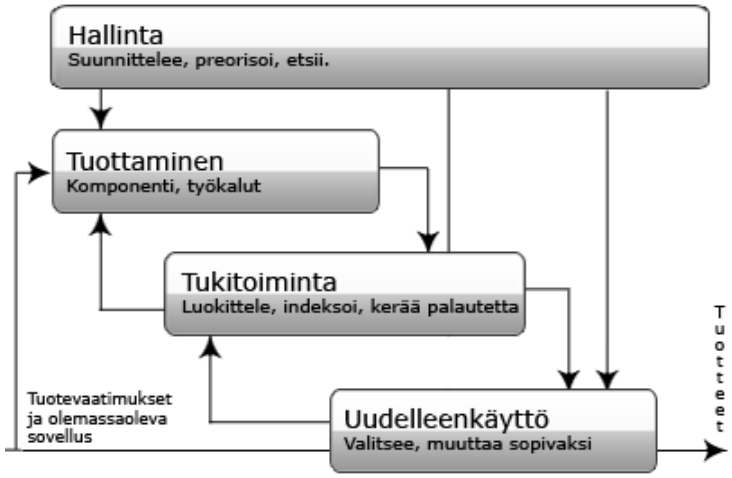
Kuva 3.3 Protolumallin tyyppi 2

Prototyyppiä voidaan käyttää lopullisen järjestelmän ytimeen. Nopeasti tehdystä komponentista tulee huonosti ylläpidettävä korjaamattomien virheiden takia.

3.3 Komponenttiperustainen malli

Komponenttiperustaiselle mallille ominaiset piirteet ovat järjestelmällisyys ja uudelleenkäytettävyys. Kyseinen malli koostuu neljästä prosessista (6;7):

- *Hallinta* sisältää muiden prosessien suunnittelun ja seurannan, mm. prioriteettien asettaminen.
- *Tuottaminen* sisältää mm. kohdealueen määrittämisen, uusien komponenttien kehittämisen, testaamisen sekä välineiden kehittämisen.
- *Tukitoiminta* ylläpitää komponenttikirjastoa, luokittele, indeksoi sekä kerää palautetta hyväksikäyttäjältä.
- *Uudelleenkäyttö* Valitsee komponentteja käyttöön sekä muuntaa niitä sopiviksi.



Kuva 3.4 Komponenttiperustainen kehittämistoiminta.

Komponenttiperustainen menetelmä suosii uudelleenkäytettävyyttä. Uudelleenkäytettävyydellä tarkoitetaan minkä tahansa komponentin tai sitä koskevan suunnitelman käyttämistä hyväksi samaa sovellusta kehittäessä tai toisessa projektissa. Uudelleenkäytettävä komponentti voidaan käyttää sellaisenaan, muunnella sen sisältö tarpeiden täyttämiseksi tai mukauttaa parametreja käyttäen. (5,8.)

4 OHJELMISTON KEHITYSTEKNIIKAT

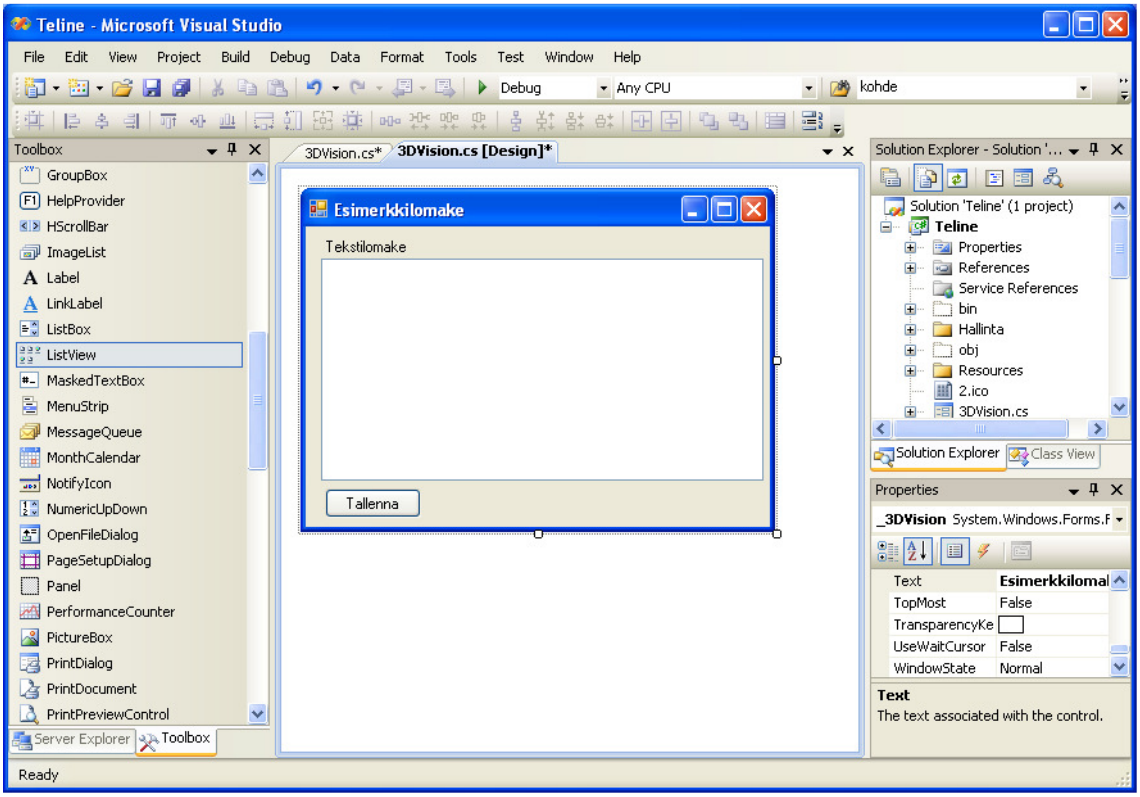
Ohjelman kehitys jakautuu graafisen ja toiminnallisen kehitykseen. Ennen kaikkea ohjelman on oltava selkeä, joten graafinen sommittelu on tärkeää. Ohjelman looginen rakenne on toteutettava selkeästi, jotta jatkokehitys olisi helppoa.

Ohjelman suunnitteluvaiheessa on tärkeää päättää tekniikat, joiden perusteella voidaan tutkia mahdolliset alusta-, ympäristö- sekä kielivaihtoehdot.

4.1 Microsoft Visual Studio 2008

Microsoft Visual Studio 2008 (VS2008) on Microsoftin kehittämä ja markkinoima ohjelmistojen kehitysympäristö (IDE), jota voidaan käyttää sekä konsoli-sovellusten että graafisten käyttöliittymäsovellusten kehittämiseen. Sovellus-tyyppejä ovat muun muassa tietokoneohjelmat ja Internet-sovellukset.

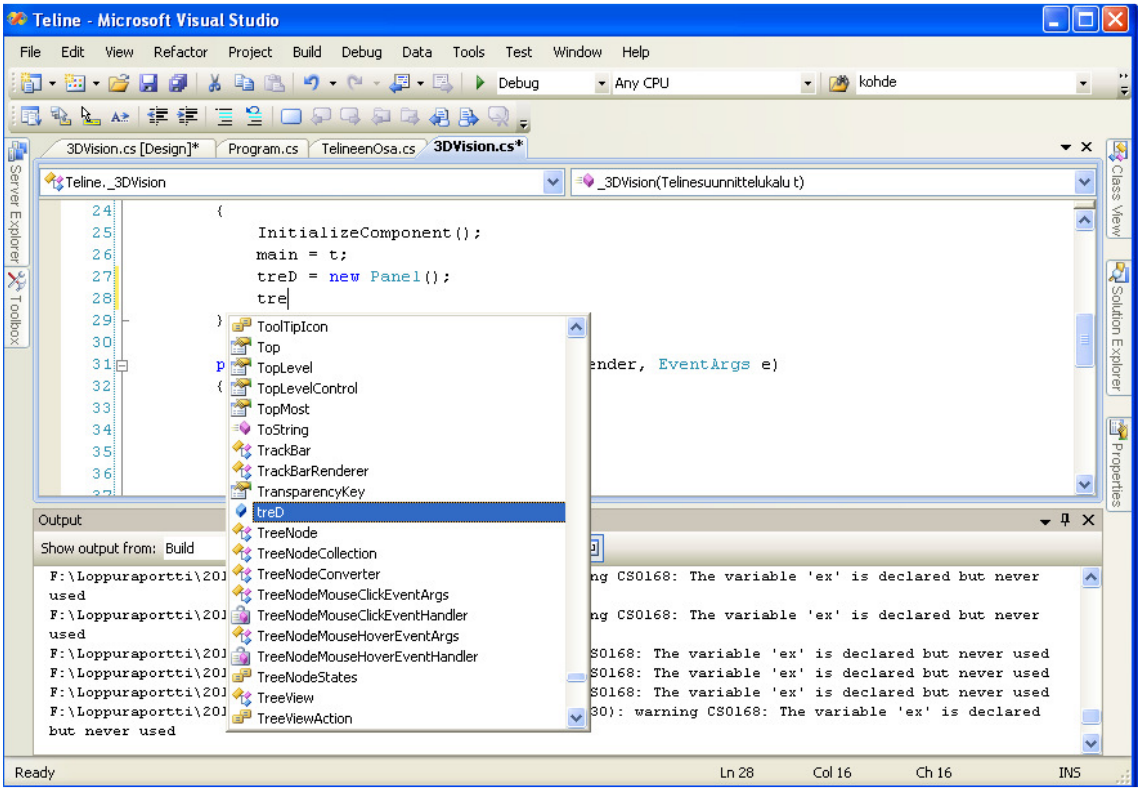
VS2008:ssa on graafinen käyttöliittymä. Seuraavassa (Kuva 4.1) esitetään tavallisen lomakkeen luominen. Sovelluksessa on erotettu komponentit, niiden ominaisuudet, luokat tai kirjastot ja muut ominaisuudet.



Kuva 4.1 Visual Studio 2008-käyttöliittymä.

Työnäkymä sijaitsee keskellä ikkunaa. Näkymän tilaa voidaan vaihdella lähdekoodinäkymän ja visuaalisen näkymän välillä. Lisättävät objektit on sijoitettu vasempaan laitaan. Projektitiedostot on listattu oikeapuolisessa valintaikkunassa, jonka alla on objektien ominaisuuksia käsittelevä valintaikkuna.

VS2008:ssa on lähdekoodieditori, joka tukee IntelliSense täydennysominaisuutta (Kuva 4.2) sekä refaktorointia. Sovellukseen on integroitu debuggeriohjelma, jota voi käyttää sekä lähde- että konekielisen ohjelmakoodin virheiden etsintään. Muita sisäänrakennettuja työkaluja ovat muun muassa graafiset käyttöliittymäominaisuudet sekä verkko- ja tietokantasuunnittelutyökalut.

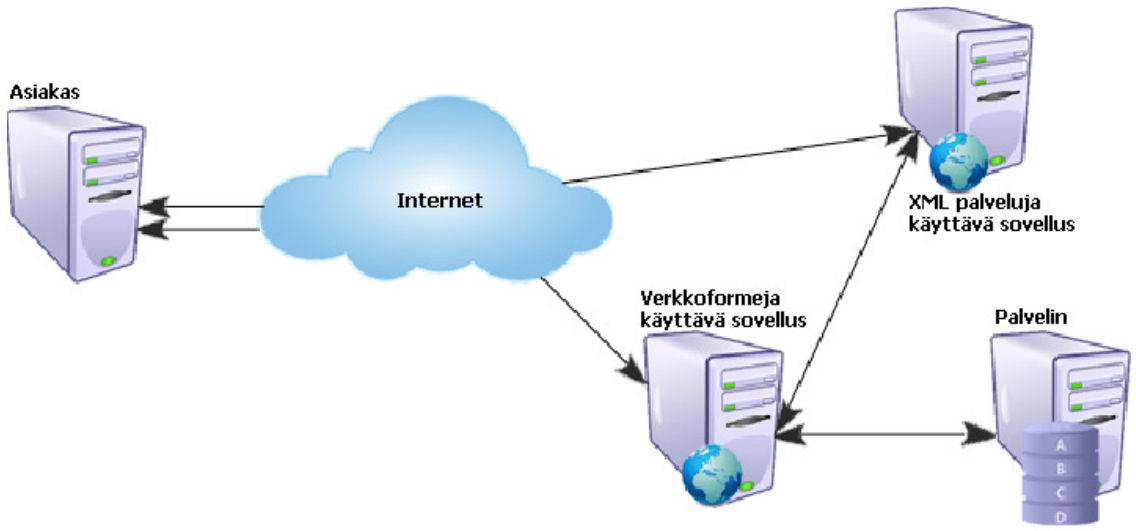


Kuva 4.2 IntelliSense-ominaisuus.

VS2008 tukee useita ohjelmointikieliä, kuten C, C#, C++, VB.NET, F#, M, Python, Ruby, XML, HTML/XHTML, Javascript, CSS.

4.2 .NET Framework

.NET Framework on kehitys- ja toteutusympäristö, joka mahdollistaa erilaisten ohjelmointikielien ja kirjastojen ongelmattoman ja sujuvan toimivuuden keskenään kehittäessä uusia Windows-, Web-, Mobiili- sekä Office-sovelluksia.

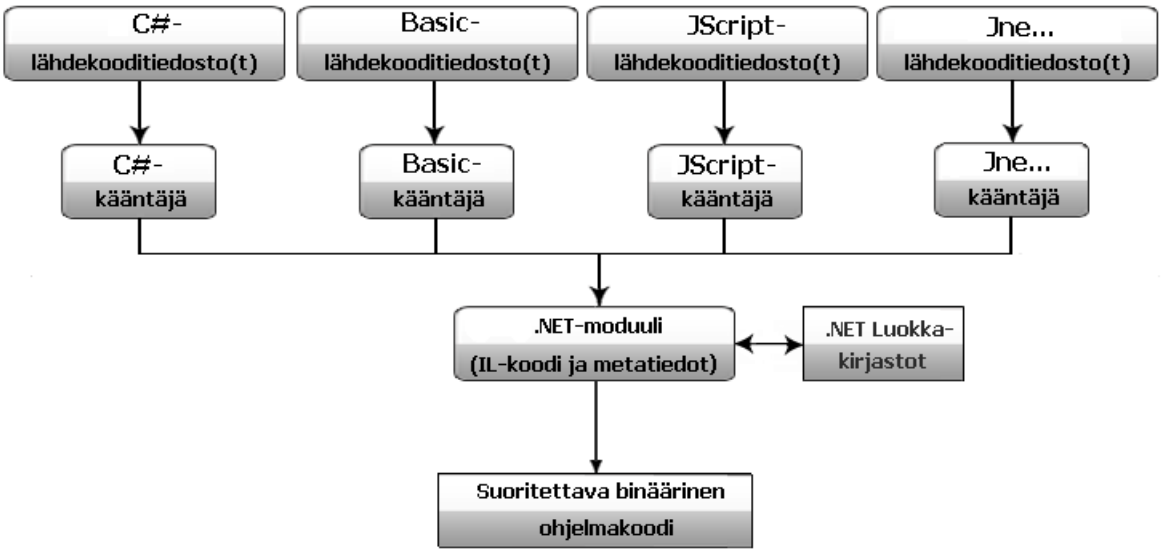


Kuva 4.3 .NET-ympäristön toimintaperiaate.

Kuvassa 4.3 esitetään .NET ympäristön toimintaperiaate verkon sisällä. .NET sisältää suuren määrän kirjastokokoelmia. Ympäristön tarkoituksena on taata sovellusten toimivuus kehityskielestä ja fyysisestä sijainnista riippumatta. (9.)

.NET Framework -kehitysympäristö muodostuu kahdesta osasta, jotka ovat ohjelmointikielille yhteinen ajoympäristö (CLR) ja luokkakirjasto (FCL).

CLR on ajonaikainen ympäristö useille erilaisille ohjelmointikielille (Kuva 4.4), joka vastaa kokoonpanojen koodin ajamisesta. Ajonaikainen ympäristö luo turvallisen ajoympäristön pakottamalla ohjelmoijaa noudattamaan tarkkoja turvallisuusmääräyksiä sekä tarjoaa erilaisia palveluja ohjelmille. Näitä palveluja ovat muun muassa muistin ja säikeiden hallinta, jotka mahdollistavat rinnakkaistehtävien suorittamista sekä eri sovelluksissa sijaitsevien olioiden kommunikoinnin. CLR sisältää virtuaalikoneen, joka kääntää kehitysympäristön tuottaman esikäännetyn IL-ohjelmakoodin binäärimuotoon, jota käyttöjärjestelmä voi lukea sekä suorittaa. CLR-ominaisuudet ovat käytettävissä kaikissa siihen sovelletuissa ohjelmointikielissä. Ajon aikana CLR ei tiedä lähdeohjelman ohjelmointikieltä, joten on mahdollista käyttää tarkoituksenmukaisinta kieltä. CLR-otsikko sisältää informaation, joka tekee tiedostosta hallittavan moduulin. Otsikosta käy ilmi muun muassa CLR-versio, moduulin aloituskohdan metadata, sen sijainti, koko ja paljon muuta.



Kuva 4.4 Lähdekoodin kääntäminen binäärimuotoon .NET-ympäristössä.

IL-ohjelmakoodi on käännöksen yhteydessä syntynyt välikoodi. Yhdessä JIT-kääntäjän kanssa välikoodi käännetään suorittimen ymmärtämään muotoon. JIT-kääntäjä (JustIn Time Compiler) optimoi koodia, joka nopeuttaa ohjelman toimintaa suorittimen tasolla.

NET Framework Class Library on luokkakirjasto, johon kuuluu useita tuhansia tyyppimäärittelyjä ja jokainen tyyppi tarjoaa toimintoja. CLR ja FCL yhdessä mahdollistavat seuraavanlaisten sovellusten rakentamisen: XML-pohjaiset verkkopalvelut, HTML-pohjaiset sovellukset ja Windows-lomakkeet, konsoli-sovellukset sekä palvelut.(10.)

4.3 C#

C#-ohjelmointikieli on julkaistu vuonna 2000. C#-ohjelmointikieli on johdettu C-, C++- ja Java-ohjelmointikielistä, mutta sen kehitys aloitettiin täysin alusta. C#-kieli on täysin oliopohjainen ja siinä on automaattien roskienkeruuominaisuus. C#-ohjelmointikieltä voidaan soveltaa muun muassa pelien, käyttöjärjestelmien, pöytäsovellusten tai verkkosivujen tekemiseen sekä päätelaitteiden hallintaan.

```
1  //Sisällykset
2  using System;
3  using System.Windows.Forms;
4  //Nimiavaruus
5  namespace WindowsFormsApplication1
6  {
7      //Luokan Form2 esittely alkaa
8      public partial class Form2 : Form
9      {
10         //Luokan sisäinen Integer - muuttuja
11         private int luku = 0;
12         /// <summary>
13         /// Oletusmuodostin
14         /// </summary>
15         #region
16         public Form2()
17         {
18             //Visuaalisten objektien piirtämien
19             //automaattinen luonti
20             InitializeComponent();
21             //Kutsutaan Näytä-metodia
22             Nayta();
23         }
24         #endregion
25         /// <summary>
26         /// Nayta - metodi
27         /// </summary>
28         public void Nayta()
29         {
30             Globaalit.Luku = 4;
31             luku = 3;
32             //Tulostus debuggerikomentoriviin
33             Console.WriteLine("Globaali luku: " + Globaalit.Luku +
34                 " Paikallinen luku " + luku + "\nGlobaali teksti "
35                 + Globaalit.teksti);
36         }
37     }
38     //Luokan Form2 esittely päättyy
39     //Luokan Globaali esittely alkaa
40     public static class Globaalit
41     {
42         //Globaali merkkijonomuuttuja
43         public static string teksti = "Globaali teksti";
44         //Globaali Integer-tyyppinen muuttuja
45         static int luku;
46         //Globaali Integer-ttyyppisen muuttujan saantimetodit
47         public static int Luku
48         {
49             set { luku = value; }
50             get { return luku; }
51         }
52     }
53     //Luokan Globaali esittely päättyy
54 }
```

Kuva 4.5 C#-kielen syntaksi.

C#-kieli on C++- ja Java-ohjelmointikielen tyyppinen. Jokaisen tiedoston alussa on mainittava, mitä kirjastoja käytetään kyseisessä tiedostossa. Kirjastoesittelyn jälkeen määritetään nimiavaruus, jonka sisälle toteutetaan luokat. Luokkien sisällä esitellään muuttuja sekä metodit. Edellinen (Kuva 4.5) esittää luokan lähdekoodinäkymän. Esimerkissä on kuvattu paikallinen ja globaaliluokka.

Paikallisessa luokassa kutsutaan globaaliluokan metodia. Globaalista luokasta ei tarvitse tehdä ilmentymää new-sanalla, toisin kuin staattisia luokkia käyttäessä.

C#-lähdekoodin käännösvaiheessa käännetään ohjelma välikoodiksi (IL), jonka jälkeen CLR ja FCL yhdessä luovat metadatatiedoston, johon tallennetaan suorittavat komennot. JIT-kääntäjä optimoi ja kääntää välikoodin konekieleksi (10;11;12.)

4.4 MOV'AMP 0.5

MOV'AMP on ilmainen kehitystyökalu, jonka tarkoituksena on helpottaa kehittäjien sovelluksen suunnittelutyötä. Kehittäjien tarkoituksena on ollut nopean ja helppokäyttöisen työkalun luominen, joka soveltuu pääasiassa verkkosivujen tekemiseen, mutta käytännössä sitä voidaan soveltaa minkä tahansa ohjelmointikielen yhteydessä. MOV'AMP toimii Windows 2000-, XP-, Vista- ja 7-alaisuudessa. Sovellus voidaan asentaa muistitikulle tai suoraan tietokoneelle. Työkalu sisältää Apache- ja MySQL-serverit, PHP-tulkin sekä PhpMyAdmin-työkalun. Työkalu vaatii MySQLConnector- ajurin toimiakseen VS2008-ympäristössä.

Apache-palvelimella on ollut tärkein rooli Internet-verkon kehityksessä, koska suurin osa verkkosivuista on perustanut toimintansa tälle alustalle. Apachea on käytetty enimmäkseen tarjoamaan sekä staattista sisältöä, että dynaamisia sivuja verkon sisällä (13.).

PHP-tulkki tarjoaa dynaamisten sivujen kehittämisympäristön. Kyseinen palvelu on kehitetty 1990-luvulla. Tarkoituksena oli yhdistää HTML-esityskieli tietokantaan sekä korvata joitakin Perl-kielen koodiosioita. PHP-lähdekoodi sisällytetään HTML-esityskieleen. PHP-kielen tärkeimpänä tehtävänä on tiedon siirtäminen palvelinkoneen ja etäkäyttäjän välillä. PHP-kielen avulla voidaan muokata sekä poistaa tietoa joko tietokannasta tai palvelinkoneen määrätystä kansiosta.

PhpMyAdmin on verkkopohjainen sovellus, joka helpottaa tietokantojen sekä niiden tietojen ylläpidon. PhpMyAdmin on verkkosivulomake, joka luettelee käyttäjälle kyseisellä serverillä olevat tietokannat sekä mahdollistaa niiden muokkauksen ja poiston. Käyttäjän ei välttämättä tarvitse osata SQL-kieltä tietokantojen ylläpitämiseen.

MySQLConnector on MySQL:n kehittämä rajapinta, joka mahdollistaa MySQL-serverin yhteensopivuuden erilaisten ohjelmakielien kanssa.

4.5 Microsoft Access 2003

Microsoft Access 2003 on Microsoftin tarjoama tietokantahallintajärjestelmä. Järjestelmä koostuu graafisesta käyttöliittymästä sekä Microsoft Jet -tietokanta-moottorista, jota on käytetty yleisesti Microsoftin työkaluissa. Microsoft Access tarjoaa tarvittavat ominaisuudet tietojen säilyttämiseen, indeksointiin, lajitteluun ja hakuun. Toimiakseen Visual Studio -ympäristössä Microsoft Access vaatii Microsoft Jet Ole DB -ajurin. Kyseinen ajuri on Microsoftin kehittämä rajapinta, joka mahdollistaa Jet-moottoria käytettävien tietokantojen toimivuuden muiden ohjelmien kanssa.

4.6 DirectX 9.0

DirectX on ohjelmointirajapinta (kokoelma kirjastoja), joka mahdollistaa tietokoneiden resurssien tai komponenttien käytön Microsoftin alustoilla. Ennen kaikkea DirectX on tunnettu peleistä, sillä suurin osa Windows alustoille suunniteluista viihdykkeistä käyttävät juuri tätä kokonpanoa.

DirectX on Microsoft alustoille asennettava rajapinta, joka tarjoaa kirjastoja muun muassa multimedia-, piirto- sekä pelisovelluksille (15.).

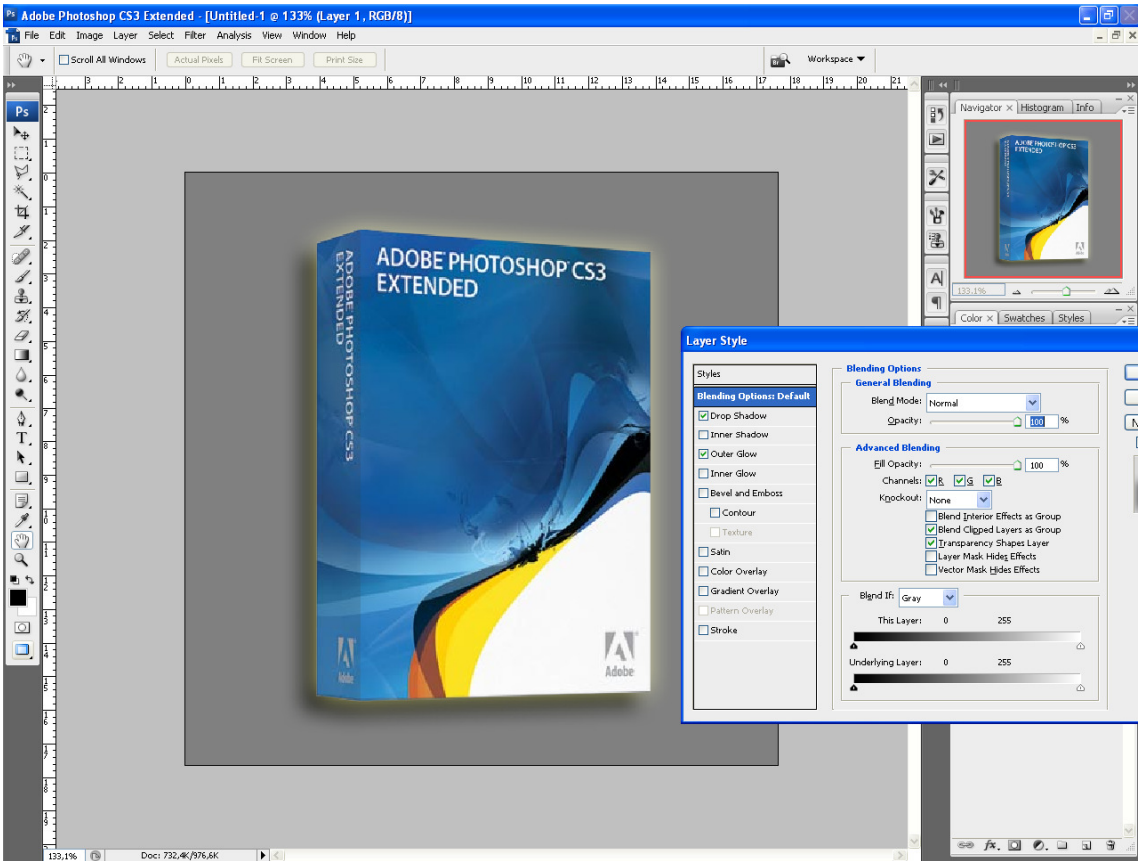
DirectInput on DirectX:n rajapinta, jota käytetään päätelaitteista (hiiri, näppäimistö, peliohjain) saatujen tietojen käsittelyyn.

DirectDraw on DirectX:n rajapinta, joka tarjoaa tuen 2D-grafikoiden hallintaan. DirectDraw-kirjasto on erikoistunut laitteisto- sekä näyttölaitteiden muistin hallintaan ja käyttää muistin kiihdytystä aina, kun on mahdollista. Kyseisen kirjaston avulla voidaan hallita sekä fyysistä että graafista muistia.

Direct3D on matala-asteisten graafisten ohjelmien rajapinta, jolla voidaan esittää kolmiulotteisia malleja. Kyseistä rajapintaa on käytetty muun muassa XBOX-konsoleissa.

4.7 Adobe Photoshop CS3

Photoshop CS3 (Kuva 4.6) on Adobe-perheeseen kuuluva visuaalinen kuva-editori. Kyseinen ohjelma on hyvin suosittu ammattilaisten keskuudessa sen helppokäyttöisyyden ja laajan tarjonnan ansiosta (16.).



Kuva 4.6 Adobe Photoshop CS3

Photoshop CS3 tarjoaa kuvamuokkaimen, jolla voidaan suunnitella ja piirtää kuvia ja kaavioita.

5 SAIMAAN ERISTYS OY:N TELINELASKENTAOHJELMAN KEHITYSPROJEKTIN VAIHEET

Järjestelmän kehitysprojektissa on sovellettu vesiputousmallia, protoilumallia sekä komponenttiperustaista mallia riippuen siitä, mitä osa-aluetta on kehitetty.

5.1 Projektin organisointi

Opinnäytetyöprojektiin osallistui neljä jäsentä: asiakasyrityksen yhdyshenkilö, projektin ohjaaja sekä kaksi toteuttajaa. Ohjausryhmä nimesi työhön kaksi opinnäytetyöntekijää (Valtteri Lantta, Jan Pount), koska ongelmaan ei ollut tiedossa helppoa ratkaisua ja työ tulisi vaatimaan paljon tutkimista, kokeilua ja protoilua. Projektiryhmän päälliköksi nimettiin Valtteri Lantta. Ohjauspalavereita pidettiin säännöllisesti viikoittain. Asiakaspalaverit pidettiin kuukausittain. Palavereissa käsiteltiin muun muassa edellisen jakson työt, projektin riskit, tekemättömät työt sekä projektin tilanne ja suunniteltiin seuraavan jakson työt.

5.2 Esitutkimus

Ensimmäisen asiakaspalaverin jälkeen projektiryhmä teki muutokset Miika Puroharjun tekemään esitutkimukseen vastamaan muuttunutta tilannetta. Esitutkimuksessa rajattiin tavoite sekä projektiryhmän työalueet. Valtteri Lantan tehtävänä oli kehittää objektien lisäys-, muokkaus-, pyöritys- sekä poisto-ominaisuudet ja valita piirtoalueen värikoodit. Jan Pountin päätehtävä oli kehittää käyttöliittymä sekä looginen rakenne. Muihin tehtäviin kuuluvat muun muassa palvelimeen ja ohjelman välisen yhteyden luominen sekä objektien tallennus- ja latausominaisuuksien kehittäminen.

5.3 Projektisuunnitelma

Esitutkimuksen jälkeen päivitettiin projektisuunnitelma. Kyseisessä dokumentissa on kerrottu yleisesti projektin taustoista, rajauksista, ympäristöstä sekä tuloksista. Myös on otettu karkeasti kantaa projektin työmenetelmiin, päättämiseen, asiakkaan kouluttamiseen sekä projekti aikatauluun.

5.4 Määrittely

Määrittelyvaiheen alussa päivitettiin Miika Puroharjun tekemät vaatimus- sekä toiminnallinen määrittely vastamaan nykyistä tilannetta. Määrittelyä ei kuitenkaan voitu kirjoittaa kokonaan, koska projektiryhmä prototyyppejä kehittäessä samalla tutki, miten toteutus tulisi tehdä. Ehdotuksina oli muun muassa erilaisten ilmaisohjelmien sulauttaminen järjestelmään. Projektiryhmä ei kuitenkaan ole löytänyt mitään sopivaa mallinnusohjelmaehdokasta, joka vastaisi projektin vaatimuksia. Siinä vaiheessa todettiin, että uuden mallinnussovelluksen kehitys on välttämätön. Määrittelyä on päivitetty protoilun edetessä.

5.5 Protoiluvaihe

Protoilu tuotti ohjelmasta prototyypivestion, jonka tehtävänä on selventää, mitä asiakas tarkalleen tarvitsee. Jokaisesta ominaisuudesta on tehty oma demoversio, jota on hiottu vastamaan asiakkaan toivomuksia. Lopuksi jokaisesta demosta on joko kehitetty lopullinen versio tai muutettu joitakin lomakkeita demosta saatujen palautteiden perusteella.

6 KÄYTTÖLIITTYMÄN JA TOIMINNALLISUUDEN ESITTELY

Asiakasvaatimusten mukaan sovelluksen on oltava helppokäyttöinen, joten projektin edetessä suunniteltiin useita erilaisia toiminnallisia ja visuaalisia vaihtoehtoja ja jokaisesta vaihtoehdosta luotiin prototyyppi. Valmista prototyyppiä on testattiin ja arvioitiin. Parhaasta prototyypistä jalostettiin täydellisesti tarpeita vastaava komponentti, tai toteutettiin sovelluksen osa-alue uudestaan prototyypistä saatujen tulosten perusteella.

6.1 Käyttöliittymä

Käyttöliittymä on toteutettu osittain VS2008:n tarjoaman käyttöliittymä-suunnittelutyökalun avulla. Käyttöliittymän visuaalinen puoli on toteutettu Adobe Photoshop CS3 -koululisenssillä. Käyttöliittymän looginen toiminnallisuus on toteutettu C#-kielellä. Ideoita on lainattu myös muista kielistä, muun muassa Javascript-ohjelmointikielestä, joka on yleisesti käytössä verkkosivuilla. Käyttö-liittymä on toteutettu noudattaen sekä protoilumallin että komponentti-perustaisen mallin etenemisperiaatteita.

6.2 Toiminnallisuus

Toiminnallisuus eli ohjelman looginen rakenne on toteutettu C#-ohjelmointi-kielellä.

Toteutetun ohjelman ensimmäinen käyttötapaus on sisäänkirjautuminen, jonka perusteella käyttäjälle avataan juuri sille käyttäjätyypille ominainen valikkorivi. Valikon avulla käyttäjä näkee sallitut ominaisuudet sekä voi käyttää niitä vapaasti.

Käyttäjän kirjautuessa onnistuneesti järjestelmään sovellus vertaa paikallisen- ja palvelimen tietokannan (jos mahdollista), tiettyjä tauluja. Jos taulujen tietuerivit eivät täsmää tai muuten on jotakin päivitettävä, sovellus ilmoittaa siitä käyttäjälle sekä samalla tekee yhteenvedon päivettävistä kohdista.

Erilaiset hallintaominaisuudet on päivitettävissä suoraan tietokantaan. Hallintavälilehdet listaavat sille välilehdelle ominaiset tietotyypit, joita käyttäjä voi muokata riippuen käyttäjäoikeuksista. Erilaiset raportit voidaan tulostaa sekä PDF-muotoon että suoraan paperille.

Sovelluksen keskeisin osa on 3D-mallinnustyökalu, jolla mallinnetaan tarvittavat kohteet ja niiden ympärille tai sisälle koottavat teline-elementit sekä lasketaan elementtien määrä. Mallinnussovelluksen päätyönäkymä on 3D-paneeli, jossa käyttäjä mallintaa tarvittavat työkohteet kolmiulotteisessa avaruudessa. Kaikkea 3D-avaruudessa ei kuitenkaan voida toteuttaa. Esimerkiksi mallinnusvaiheessa erilaisten objektien sijainteja muuttaessa on mahdollista käyttää joko näppäimistöä tai hiirtä. Näppäimistöllä koordinaattien muuttaminen vaati matemaattisia taitoja ja tarkkoja lukuja. Hiirellä siirrettäessä koordinaatit lasketaan automaattisesti, mutta yhden elementin käyttäjän näkökulmasta ollessa oikeassa paikassa eri sivusta katsottuna objekti voi olla jonkun koordinaatin mukaan väärässä paikassa. Ratkaisuksi on sovittu 2D-projektionäkymät. Tähän sovellusosaan on kehitetty kolme 2D-projektiota, jotka on periytetty 3D-avaruudessa mallinnetuista objekteista, mutta joitakin ominaisuuksia on karsittu, jotta piirros olisi käyttäjän helposti ymmärrettävissä. Kaksiulotteiset projektiot helpottavat objektien yhdistämistä, monistamista sekä paikalle asettamista. Suunniteltuaan valmiin objektin, käyttäjä voi tallentaa sen kirjautumisprofiilin mukaan joko palvelimeen tai paikalliselle kiintolevylle.

7 TELINELASKENTAOHJELMAN ESITTELY

Telinelaskentaohjelman kaksi tärkeintä tehtävää ovat kohteiden sekä niiden ympärille tai sisäpuolelle koottavien elementtien mallintaminen. Sovelluksen on mallinnettava käyttäjän suunnittelemat esineet 3D- ja 2D-avaruudessa kolmena projektioina. Mallista selviää kohteen ympärille pystytettävien teline-elementtien lukumäärä sekä niiden ominaisuudet (pituus, paino, tuotenumero). Malli voidaan tallentaa tietokantaan, joko paikallisesti tai palvelimelle.

7.1 Käyttäjätyypit

Telinelaskentaohjelma on tunnuksella ja salasanalla suojattu ohjelma, joten jokaisella käyttäjällä on oltava oma tili, jotta ohjelmaa voidaan käyttää. Ohjelmassa on myös erilaisia hallintaominaisuuksia, kuten yhdyshenkilöiden, tilaajien, tehtaiden ja käyttäjien hallintaosiot.

Telinelaskentasovelluksella on kaksi eri käyttäjätyyppiä: järjestelmänvalvoja ja tavallinen käyttäjä. Tavallisen käyttäjän oikeudet ovat rajalliset. Tavallisen käyttäjän mahdollisuudet ovat seuraavat:

- kirjautuminen
- lähetyslistan tulostus
- telinelaskelmien hallinta: telinelaskelmien lisäys, itse lisättyjen telinelaskelmien muokkaus ja poisto sekä omistuksen siirto
- osien hallinta: osien lisäys, muokkaus ja poisto
- kohteiden hallinta: kohteiden lisäys, muokkaus ja poisto
- tehtaiden hallinta: tehtaiden lisäys, muokkaus ja poisto
- tilaajien hallinta: tilaajien lisäys, muokkaus ja poisto
- yhdyshenkilöiden hallinta: yhdyshenkilöiden lisäys, muokkaus ja poisto.

Järjestelmänvalvojan tehtävät ovat virheiden poistaminen ja ongelmien ratkaisut. Järjestelmänvalvojalla on täydet käyttöoikeudet telinesuunnittelusovel-

luksessa ja hänellä on peruskäyttäjän oikeuksien lisäksi oikeus käyttäjien hallintaan: käyttäjien lisäykseen, muokkaukseen ja poistoon.

7.2 Järjestelmään kirjautuminen

Jotta ohjelmaa voi käyttää, on käyttäjän ensin kirjauduttava ohjelmaan omalla tunnuksella ja salasanalla (Kuva 7.1). Ellei tavallisella käyttäjällä ole tunnusta eikä salasanaa, hänen on pyydettävä ne järjestelmänvalvojalta.

Kuva 7.1 Kirjautumislomake

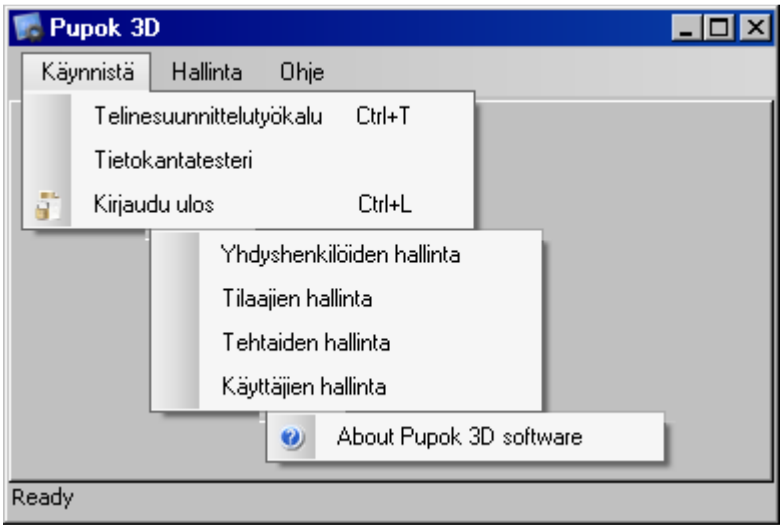
Ohjelman on pystyttävä toimimaan sekä offline- että online-tiloissa. Tietokoneen ollessa offline-tilassa kaikki tiedot tallennetaan paikallisesti sovellusta suorittavan tietokoneen kiintolevylle. Online-tilassa tallennettavat tiedot siirretään palvelimeen. Molemmat tilat synkronoidaan aina kirjautuessa onnistuneesti online-tilaan.

7.3 Aloitussivu

Kirjauduttuaan sovellukseen onnistuneesti käyttäjä näkee kuvassa (Kuva 7.2) esitetyt valikot. Telinesuunnittelutyökalu on se sovelluksen osa, jolla mallinnetaan kohteet ja teline-elementit. Kirjaudu ulos -valinta sulkee yhteyden ja ohjelman. Jos käyttäjä kirjautuu ulos sulkematta kaikkia välilehtiä, ohjelma ehdottaa tallentamista, joka voidaan hyväksyä, kieltää tai jättää huomioimatta.

Hallinta-välilehden alla sovellus tarjoaa käyttäjälle erilaiset hallintaominaisuudet. Käyttäjien hallinta on näkyvissä vain järjestelmänvalvojalle.

Koko työikkuna koostuu valikosta, jonka avulla käyttäjä valitsee haluamansa ominaisuudet, sekä taulukosta, johon käyttäjän valitsemat ominaisuudet avataan välilehdittäin. Kyseinen taulukko asettaa tarvittavat parametrit avatuille lomakkeille, jotta käyttöliittymä olisi mielekkään näköinen.



Kuva 7.2 Telinesuunnittelusovelluksen aloitussivu.

Työikkunan alalaidassa on tekstirivikenttä, jossa ilmaistaan ohjelman tila. Ready-sana tarkoittaa sitä, että ohjelma toimii ongelmitta. Virheen ilmetessä virheraportti tallennetaan HTML-tiedostoon ja käyttäjälle näytetään ainoastaan tietohäiriöitä.

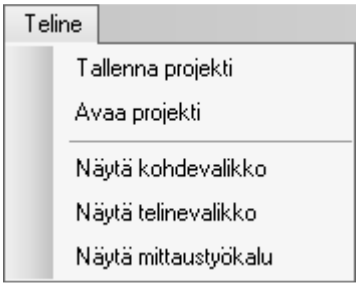
7.4 Mallinnus

Mallinnustyökalu on toteutettu omana lomakkeena, joka käyttää opinnäytetyössä kehitettyjä komponentteja, (luokkia ja objekteja). Mallinnustyökalu sisältää itsessään piirto-ominaisuudet sekä objektien tallentamisominaisuuden, joten riippuvuus päälomakkeesta on hyvin olematon.

7.4.1 Käyttöliittymä

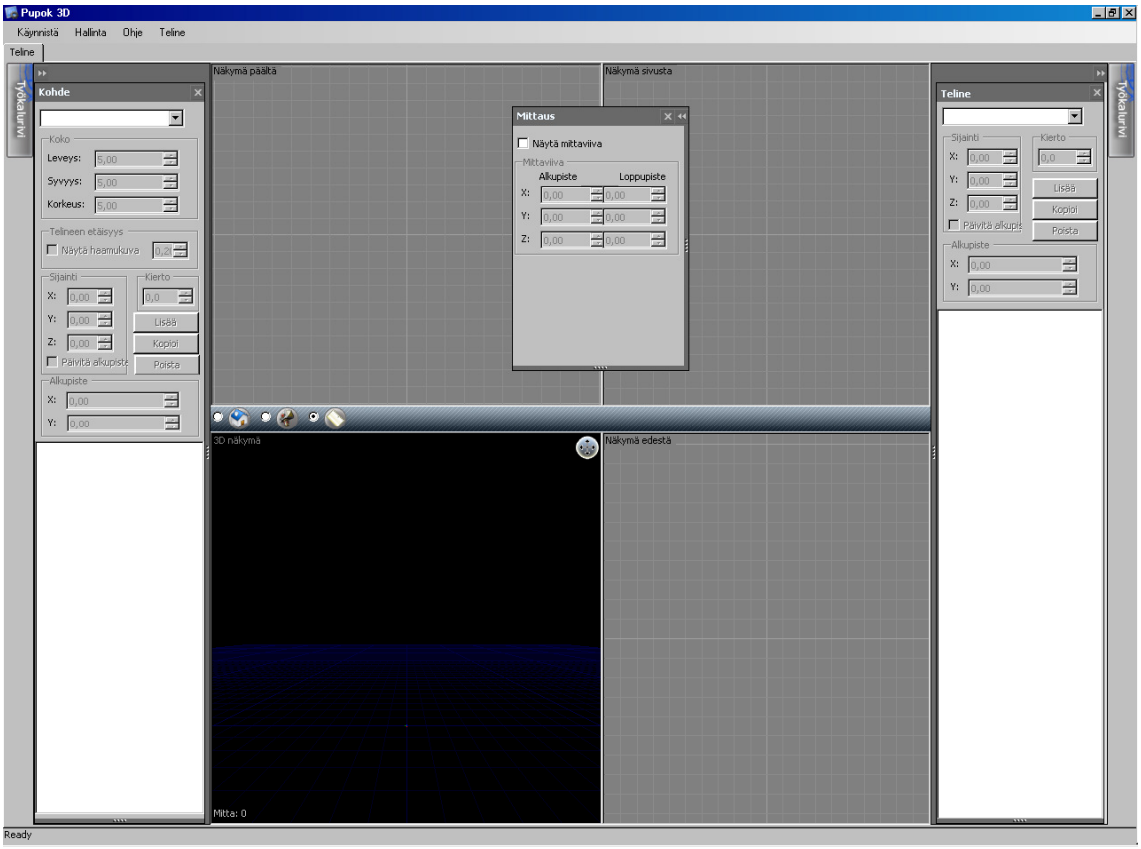
Käyttöliittymä koostuu ajonaikaisesti lisättävästä valikosta (Kuva 7.3), neljästä paneelista, jotka toimivat piirtoalustana (Kuva 7.4) ja muista paneeleista, jotka vastaavat objektien osien muokkaamisesta sekä valikoiden säilyttämisestä.

Teline-valikko avataan aina mallinnussovellusta avattaessa ja suljetaan aina kyseistä sovellusta suljettaessa.



Kuva 7.3 Mallinnussovelluksen ajoaikaisesti lisätty valikko.

Näytä kohdevalikko -valinta avaa valikon, joka sisältää kohteiden suunnitteluun tarvittavat työkalut. Näytä telinevalikko -valinta avaa teline-elementtivalikon. Näytä mittaustyökalu -painike avaa mittaustyökaluvalikon, jonka avulla käyttäjä voi mitata välimatkoja kahden pisteen välillä.



Kuva 7.4 Mallinnussovelluksen aloitusnäkymä.

Liukuvien paneelien käyttämiseen liittyvä idea on saatu HTML-esittelykielen sekä Javascript-verkkosovelluksien toiminnallisuuden toteutuskielen yhdistelmästä. Rakenne on sekä helposti toteutettavissa, että käyttäjän ymmärrettävissä.

7.4.2 Rakenne

Mallinnussovelluksessa on kolme erilaista valintaikkunaa: Mittaus- Kohde-, Teline-valintaikkuna. Graafisesti nämä valintaikkunat ovat toteutettu samalla periaatteella käyttäen aluksi protoilumallia, josta on sitten jalostettu uudelleenkäytettävä komponentti. Graafinen suunnittelu on toteutettu Adobe Photoshop CS3:n avulla. Valintaikkuna pitää sisällään erialaisia tietokenttiä, joiden avulla käyttäjä voi suunnitella objekteja. Valintaikkunoiden looginen rakenne käyttää osittain yhteisiä ja osittain parametreilla tai ohjelmakoodilla muunneltuja komponentteja.

7.4.3 Valintaikkuna

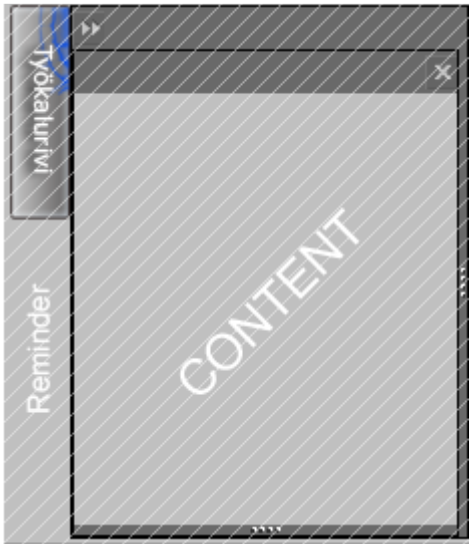
Valintaikkunoiden fyysinen sijainti ei ole suoranaisesti sidottu mihinkään tiettyyn paikkaan. Tämä ratkaisu antaa käyttäjälle vallan vapaasti päättää oman työnäkymän järjestyksestä. Halutessa käyttäjä voi siirtää valintaikkunan ruudulle, pienentää sen, liimata sovelluksen sivuille tai poista kokonaan näkyvistä. Tiettyä valintaikkunaa suljettaessa ajonaikaisesti luodun Teline-valikon (Kuva 7.3) tietyt osat aktivoituvat. Suljettuja valintaikkunoita voi käynnistää uudelleen Teline-valikon kautta.

Kuva 7.5 Valintaikkunan visuaalinen toteutus.

Näkyvillä olevat valintaikkunat voidaan sijoittaa säiliöpaneeleihin (Kuva 7.6), jotka sijaitsevat aloitusnäkymän (Kuva 7.4) sovelluksen vasemmassa ja oikeassa laidassa. Jos säiliöpaneeli ei sisällä yhtään valintaikkunaa, se piilotetaan näkyvistä. Jos valintaikkuna vedetään sovelluksen nurkkaan säiliöpaneelin ollessa piilossa, näytölle ilmesty muistuttajapaneeli (Reminder Kuva 7.6).

7.4.4 Säiliöpaneelit

Säiliöpaneelit ovat toteutettu samalla periaatteella kuin valintaikkunat sekä graafisella että loogisella tasolla. Säiliöpaneelien tehtävä on pitää sisällään valintaikkunoita. Säiliöpaneeleja voidaan kutistaa tai venyttää käyttäjän haluamaan kokoon.



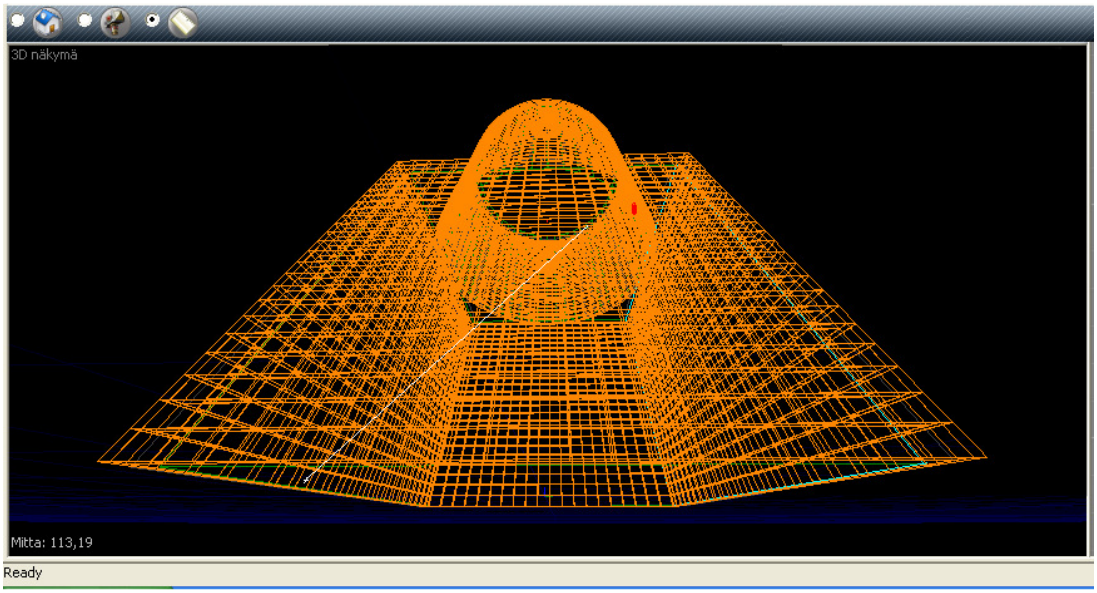
Kuva 7.6 Säiliöpaneelin visuaalinen toteutus.

Säiliöpaneeli on käytännöllinen ratkaisu siinä mielessä, jos valintaikkunat halutaan saada nopeasti näkyviille tai pois näkyvistä sulkematta niitä.

7.4.5 Työnäkymä

Työnäkymä koostuu neljästä paneelista, joita käytetään objektien suunnitteluun. Jokainen paneeli on muista riippumaton objekti. Ainoa yhtenäinen tekijä kaikille paneeleille on taulukko, johon käyttäjän suunnittelemat objektit on väliaikaisesti tallennettu. Jokainen paneeli esittää suunnitellut objektit (kohteet, elementit tai mittausviivan) paneelikohtaisten määritysten mukaisesti.

Kolmiulotteista avaruutta mallinnettava paneeli (Kuva 7.7) mahdollistaa jokaisen objektin tarkastelua käyttäjän määrittämien etäisyyksien ja katselukulmien mukaisesti. 3D-näkymä tarjoaa kokonaiskuvan tilanteesta. Kaikkea ei kuitenkaan ole mahdollista mallintaa pelkästään 3D-avaruudessa, koska ihmisen havainnointikyky ei siihen riitä. Sitä varten oli kehitettävä kaksiulotteiset näkymät.

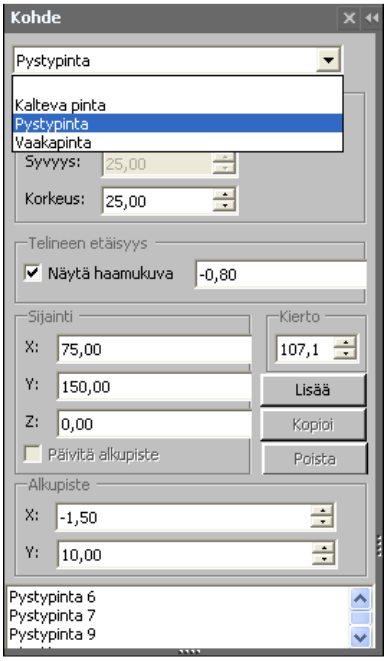


Kuva 7.7 Kolmeulotteista maailmaa esittävä paneeli.

Kaksiulotteista avaruutta simuloivia paneeleja on kolme. Niistä jokainen esittää käyttäjän mallinnetut objektit jostakin tietystä suunnasta. Mittaustyökalun toimintaperiaatetta selventävästä kuvasta (Kuva 7.11), nähdään 2D-paneelien toiminnallisuuden. 2D-paneelien päätarkoituksena on helpottaa suunnittelua. Erilaisten objektien kopiointi, sijainnin tai ominaisuuksien muuttaminen on helpompaa, kun kamera osoittaa samaan kohteeseen kolmesta eri suunnasta. Objekteja siis voidaan tarkastella kaikista kolmesta suunnasta yhtäaikaisesti katselukulmaa muuttamatta.

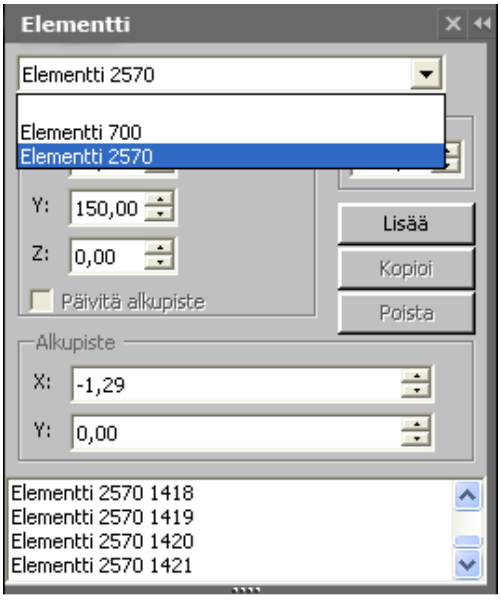
7.4.6 Objektien lisäys

Kohteet mallinnetaan viivojen sekä erilaisten pintojen avulla, joiden ominaisuudet ovat muokattavissa ajonaikaisesti koko suunnitteluprosessin aikana. Kohdevalikon Lisää-näppäimestä lisätään uusi objekti välimuistiin, jonka jälkeen se piirretään käyttäjälle. Kopioi-painike kopioi valitut objektit ja lisää ne luettelon loppuun. Poista-painikkeen avulla voidaan poistaa valitut objektit.



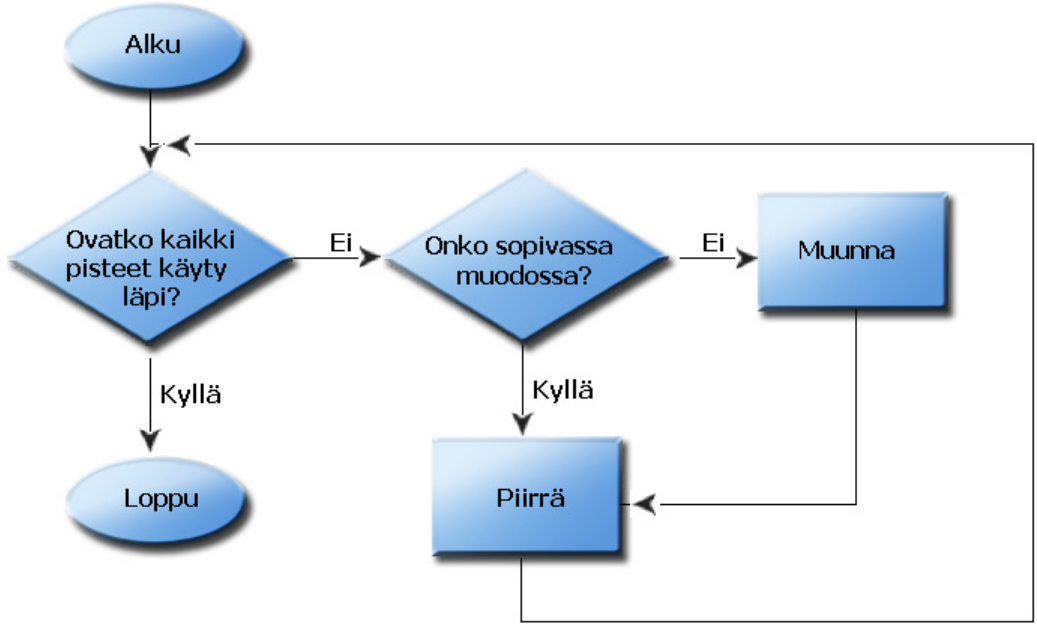
Kuva 7.8 Kohdevalintaikkuna

Elementtikokonaisuudet mallinnetaan samalla periaatteella kuin kohteet, ainoana erona on se, että elementit ovat valmiiksi mallinnettu. Elementti-valikon pudotusvalikosta voidaan valita valmiit komponentit, joita sovelletaan mallinnuksessa.



Kuva 7.9 Elementtivalintaikkuna.

Kaikki piirrettävät osat luodaan ajonaikaisesti. Kaikki objektit tallennetaan välimuistiin, josta sitten sovellus hakee tarvittavat tiedot piirtovaiheessa (Kuva 7.10). Kaikki osat piirretään samasta lähteestä. Ainoana erona on se, että ennen kaksiulotteisen mallin piirtämistä lähdettä on muokattava sopivaan muotoon.

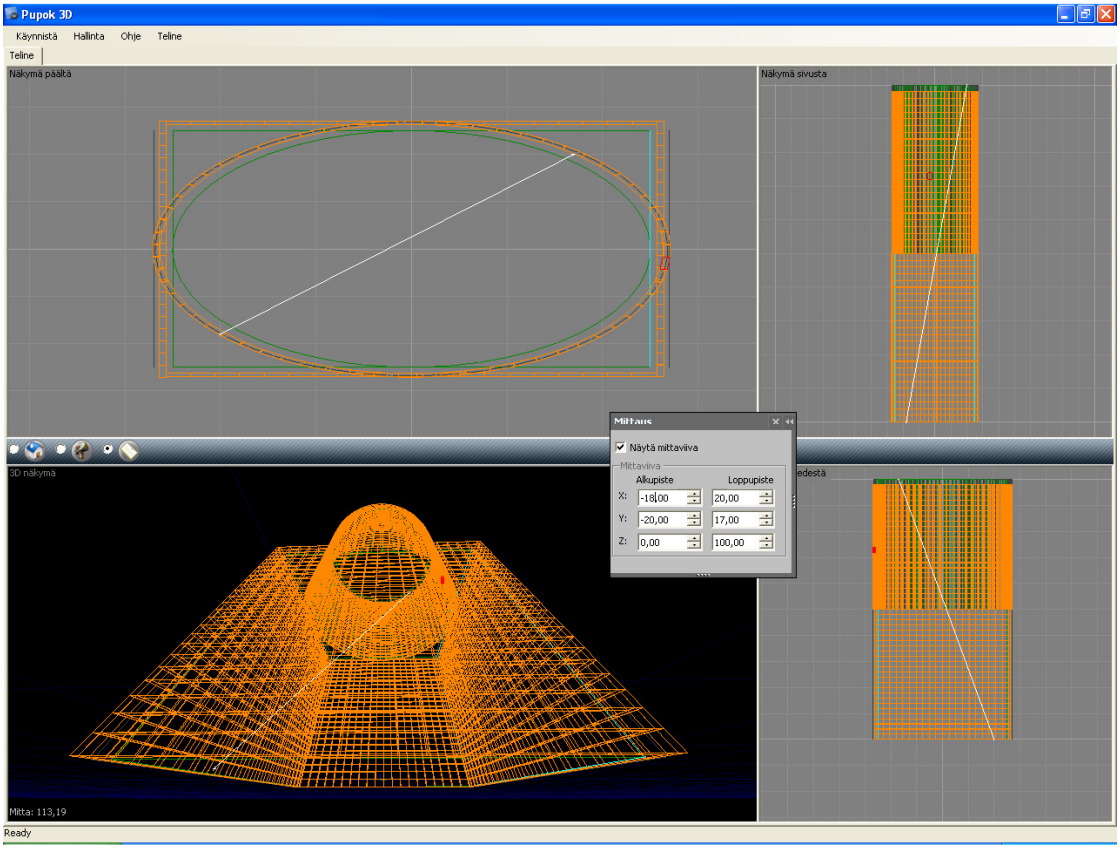


Kuva 7.10 Objektien tulostusperiaate.

Kolmiulotteisen mallin esittämiseen tarvitaan kolmea eri koordinaatistotasoa. Kaksiulotteisessa avaruudessa käytetään vain kahta koordinaatistotasoa. Edellisen kuvan (Kuva 7.10) Muunna-toiminto asettaa tietyn koordinaatiston kaikki arvot nolliksi.

7.4.7 Mittaus

Halutessaan käyttäjä voi ottaa tarkistusmittoja mittaustyökalulla. Työkalun ainoana tehtävänä on mitata kahden pisteen etäisyys (Kuva 7.11). Valkoinen viiva esittää mittaustyökalua.



Kuva 7.11 Mittaustyökalun toimintaperiaate.

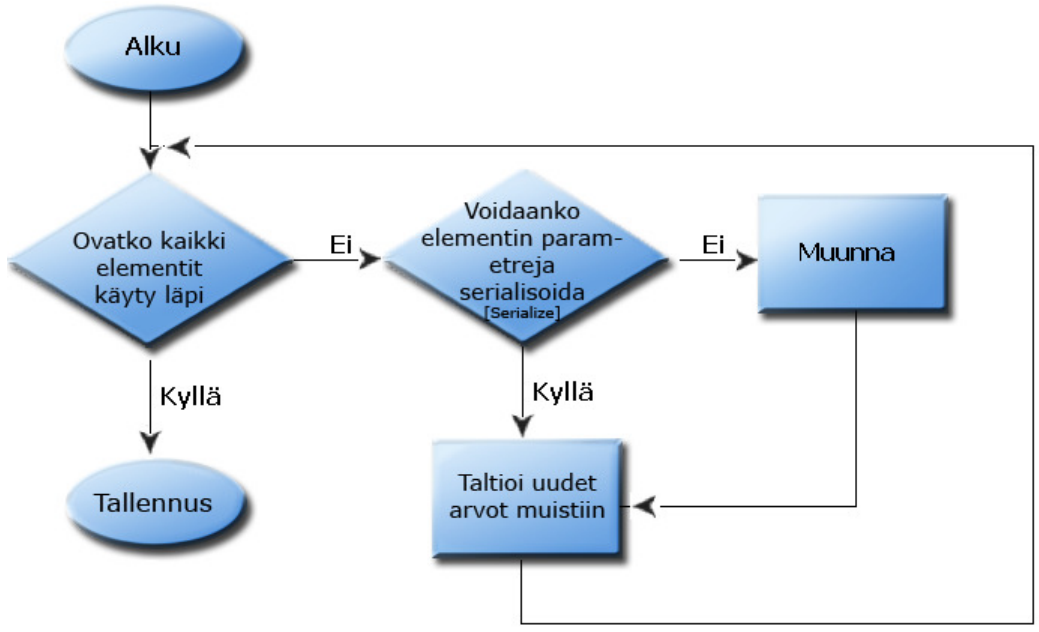
Mittaustyökalu laskee kahden pisteen, jotka ovat käyttäjän siirrettävissä, välinen etäisyys, jonka jälkeen tulostaa luvun kolmiulotteiseen näkymään sijoitettuun kenttään.

7.5 Tietojen taltiointi

Sovelluksen miltei jokainen osio on sidottu jollakin tapaa tietokantaan. Järjestelmään kirjautuminen ei onnistu ilman tietokantayhteyttä. Tässä projektissa on käytetty kahta erilaista tallennusmenetelmää. Paikalliseen tietojen talletukseen on käytetty Accessin (Microsoft Access 2003) tarjoamia työkaluja. Tietokantamoottorina on käytetty Jet-tietokantamoottoria. Tallennus-tapana on Accessin tarjoama perinteinen binääritiedosto (.mdb). Jotta kyseinen teknologia toimisi Microsoft Visual Studio ympäristössä, on tietokoneelle asennettava Microsoft Jet Ole Db -tietokantarajapinta. Tässä sovelluksessa on käytetty kyseisen kirjaston neljättä versiota.

Palvelin on simuloitu paikallisen palvelimen, MOV'AMP 0.5:n, avulla. Kyseisen työkalun tarjoamista työkaluista on tässä projektissa käytetty PhpMyAdminiä ja MySQL-serveriä. Tämä työkalu vaatii MySQL Connector/Net -tietokanta-rajapinnan toimiakseen VS2008-ympäristössä.

Käyttäjän kirjautuessa järjestelmään sovellus tarkistaa, onko yhteyden luominen palvelimeen mahdollista, ellei ole, järjestelmä ottaa paikallisen tietokannan käyttöön. Sisäänkirjautunut henkilö voi muokata omia tietoja Käyttäjien Hallinta-välilehdestä. Tehtaiden, tilaajien sekä yhdyshenkilöiden hallinnoille on omat lomakkeensa, joiden avulla tietoja voidaan päivittää suoraan tietokantaan. Kaikki objektit tallennetaan ensin binääriseen muotoon, jonka jälkeen ne viedään tietokantaan. Ennen tallennusta on kuitenkin muutettava objektien ominaisuudet, koska joitakin DirectX:n tarjoaamia komponentteja ei ole mahdllista tallentaa sellaisenaan. Tämän takia tallennusvaiheessa on käytävä jokainen elementtiolio läpi ja vaihdettava sen ominaisuudet kelvolliseksi ja vasta sen jälkeen tallennettava.



Kuva 7.12 DirectX–komponenttien tallennus levylle

Tiedoston avaamisessa mallinnustyökalun sisällä on edettävä samalla periaatteella päinvastaisessa järjestyksessä.

7.6 Virheraportti

Ajonaikaisesti syntyneet virheet taltioidaan HTML-tiedostoon tuleva kehitystä varten. Tiedosto sisältää yleiset virhetilanteet ja paikan, missä kukin virhe on tapahtunut. Virheet erotellaan päivämäärällä ja kellonajalla.

7.7 Tulosteet

Sovelluksella voi tulostaa lähetyslistan, josta selviää telineeseen tarvittavien osien lukumäärä ja ominaisuudet. Tuloste on PDF-muodossa.

7.7.1 Lähetyslista

Lähetyslista on dokumentti, josta selviää telineeseen tarvittavien osien lukumäärä ja ominaisuudet (tuotenumero, paino ja pituus). Lähetyslista voidaan joko tulostaa tai tallentaa Excel-taulukkomuodossa.

8 POHDINTA

Alussa työn tavoite oli lisätä olemassa olevaan telinelaskentaohjelmaan uusia rakennuskohdetyyppejä ja niille laskenta-algoritmeja. Varsin nopeasti todettiin, että kohdetyyppejä olisi rajaton määrä, joten tarvitaan ohjelma, jolla voidaan mallintaa kohde ja sen ympärille toimiva rakennusteline.

Ohjausryhmän näkemys asiasta oli, että aihe on joka tapauksessa niin mielenkiintoinen ja kehittävä, että tutkimusta ja protoilua kannattaa ehdottomasti jatkaa opinnäytetyönä ja että tämän projektin jälkeen aloitetaan tarvittaessa uusi projekti, joka tekee ohjelmasta lopullisen version.

Työmääräarvion tekeminen oli alussa mahdotonta, koska ohjelma oli määrittelemättä, eikä ollut edes varmuutta siitä, oliko ohjelma toteutettavissa yleispätevänä ratkaisuna.

Projektin tekemiseen varattu aika osoittautui toteutukseen riittämättömäksi, koska kului paljon aikaa tiedonkeruuseen, opiskeluun sekä suunnitteluun. Prototyyppiversioita on projektin aikana syntynyt lukuisia määriä. Työn tämänhetkinen tilanne riskien suhteen näyttää hyvältä, koska vaikein osa ohjelmasta on todistettu prototyypillä toteutuskelpoiseksi.

Ohjelmaa kehittäessä on kokeiltu erilaisia toteutustapoja. Joitakin asioita olen suunnitellut pitkään, mutta kun niitä on yritetty integroida ohjelmaan, niistä ei välttämättä saatu toivottua tulosta. Hyvänä esimerkkinä on DirectX-vektoreiden tallennus binääritiedostoon. Miltei joka kielessä on Serialization-ominaisuus, ainakin jossakin muodossa. Periaatteessa kyseinen ominaisuus mahdollistaa miltei kaiken olemassa olevan tiedon tallentamisen (muuttujat, objektit, luokat yms.). Tätä ajatusta soveltaen olen yrittänyt tallentaa vektorit tiedostoon. Tiedosto on kyllä täyttynyt, mutta väärillä tiedoilla. Kaiken lisäksi ohjelman suoritusvaiheessa virheraporttilistaan on tulostunut teksti, jonka mukaan eräät DirectX-komponentit eivät ole merkitty binäärimuotoon tallennettaviksi, vaikka koko luokka kaikkine komponentteine mukaan lukien DirectX-kirjastot olivat merkitty tallennettavaksi. Vahingossa yhtenä iltana löysin materiaalin, jonka mukaan DirecX-komponenteilla ei ole Serialization-ominaisuutta. Nämä ovat yksittäisiä tapauksia, mutta kuitenkin harmittavat. Tästä on otettu opikseen sen verran, että teknologiat tai ominaisuudet on tutkittava perin pohjin ennen soveltamista.

Visuaalinen suunnittelu oli helpoimpia asioita tässä projektissa, koska olen käyttänyt kuvakäsittelyohjelmia aiemmin, joten suunnittelu ja kuvien tuottaminen kävivät nopeasti. Vierityspalkit ja siirrettävät valintaikkunat olivat käytännöllinen ratkaisu, koska ei tarvinnut ottaa kantaa siihen, miten käyttäjä haluaisi toteuttaa valikon tai sen sijainnin. Melkein kaikki käyttöliittymän ominaisuudet ovat muokattavissa, joten jokainen käyttäjä voi kehittää mieleisensä työnäkymän.

Palvelimen kanssa ei ollut mitään ongelmia. Kaikki sovellukset tähän mennessä on toteutettu samalla periaatteella sekä melkein kaikki työvälineet ovat olleet samoja.

Vaativin osio oli kolmiulotteisen maailman kehitys. Yleensä suunnitteluun kuluu enin osa ajasta projektissa. Tätä ohjelma tehdessä suunnittelu ja toteutus edistyivät yhtäaikaisesti. Jos joku idea on tullut mieleen, se piti toteuttaa. Esimerkiksi tämänhetkisessä versiossa kohderakennuksia suunnitelessa käyttäjällä on ainoastaan kolme elementtiä. Ohjelmalla on koemielessä suunniteltu erilaisia objekteja. Miltei jokainen muoto voidaan mallintaa tehdyllä prototyyppiohjelmalla, mutta erilaiset lisäelementit olisivat helpottaneet ja nopeuttaneet mallintamista.

Projektin alussa ohjelmassa käytetyistä tekniikoista on ollut todella vähän tietoa, joten miltei kaikki on jouduttu selvittämään itse. Jos verrataan ohjelman ensimmäisiä kokeiluversioita viimeiseen prototyyppiin, voidaan rehellisesti todeta kehityksen olevan melkoinen. Projektin alkuvaiheessa ohjelman toteuttaminen on tuntunut miltei mahdottomalta. Kuitenkin vielä viimeisen version prototyyppiä tehdessä taitoa ja tietoa on kertynyt ja erilaisia toteutustekniikoita on kehitetty.

Tämän projektin ansiosta on opittu paljon erilaisia ohjelmointikielen ominaispiirteitä, joista on varmasti hyötyä tulevaisuudessa. Kaiken kaikkiaan projekti oli mielenkiintoinen ja antoisa.

Ohjelman kehitys jatkuu Miika Puroharjun ja Pekko Meurosen ohjelmointiprojektityönä ja heidän tavoitteenaan on tehdä ohjelman lopullinen käyttöönotettava versio.

KUVAT

KUVIOT

(http://ta.ramk.fi/~tauno.tepsa/TJS/Luennot%20wk%2047.pdf)

Kuva 3.4 Komponenttiperustainen kehittämistoiminta., s.15

(http://ta.ramk.fi/~tauno.tepsa/TJS/Luennot%20wk%2047.pdf).

LÄHTEET

1.Saimaan eristys Oy http://www.saimaaneristys.fi/ (Luettu 12.4.2010)

2. Layher Oy http://www.layherusa.com/allround_scaffolding.html

3. Sommerville, I. (1992): Software Engineering, Fourth Edition. Addison-Wesley

4. Wirgentius, M. (2003): Agile-menetelmät ja Crystal Clear ohjelmistotuotantoprojektissa. Insinöörityö 15.3.2003. Espoo-Vantaan teknillinen ammattikorkeakoulu.

5. Tepsa T.Prosessimallit http://ta.ramk.fi/~tauno.tepsa/TJS/Luennot%20wk%2047.pdf (Luettu 1.5.2009)

6. Karjalainen S. Prosessimallit ja ohjelmistoprojektinonnistuminen toimittajan näkokulmasta http://www.cs.uta.fi/research/theses/masters/Karjalainen_Sampo.pdf (Luettu 1.5.2009)

7. Immonen J. Johdanto ohjelmointituotantoon http://cs.joensuu.fi/~jimmonen/jot_moniste/jot_moniste_121.html (Luettu 15.4.2010)

8. Leppänen M. Tietokantasovelluksen suunnittelu ja toteutus http://users.jyu.fi/~mauri/tjtst12/Luku6.pdf (Luettu 1.5.2009)

9. Microsoft http://msdn.microsoft.com/en-us/library/zw4w595w(VS.71).aspx (Luettu 19.5.2010)

10. Jeffrey Richter (2002) Applied Microsoft .NET Framework programming

11. Microsoft http://msdn.microsoft.com/fi-fi/vcsharp/dd919145(en-us).aspx (Luettu 11.5.2010)

12. Microsoft http://download.microsoft.com/download/3/8/8/388e7205-bc10-4226-b2a8-75351c669b09/CSharp%20Language%20Specification.doc (Luettu 19.5.2010)

13. Wikipedia http://en.wikipedia.org/wiki/Apache_HTTP_Server#cite_note-1 (Luettu 01.04.2010)

14. MOV'AMP http://extensions.joomla.org/extensions/tools/standalone-servers/4565 (Luettu 10.11.2009) + paikallisen serverin (MOV'AMP) tarjoamia info sivuja (info.php)

15. Riemer's 2D & 3D XNA Tutorials
http://www.riemers.net/eng/Tutorials/DirectX/Csharp/Series1/tut1.php (Luettu 3.3.2010)

16. Adobe www.adobe.com (Luettu 3.2.2010)