KYMENLAAKSON AMMATTIKORKEAKOULU

Ohjelmistotekniikan koulutusohjelma

Mikko Tapaninen & Ville Liljeqvist

STRATEGIAPELIN SUUNNITTELU JA TOTEUTUS

Opinnäytetyö 2010

TIIVISTELMÄ

KYMENLAAKSON AMMATTIKORKEAKOULU

Ohjelmistotekniikka

TAPANINEN, MIKKO	Strategiapelin suunnittelu ja toteutus
LILJEQVIST, VILLE	
Opinnäytetyö	77 sivua + 2 liitesivua
Työn ohjaaja	Koulutusohjelmavastaava Teemu Saarelainen
Toimeksiantaja	Kymenlaakson ammattikorkeakoulu
Maaliskuu 2010	
Avainsanat	Peliohjelmointi, OpenGL, C++, SDL

Pelaamisen suosion kasvaessa myös mielenkiinto pelien tekemiseen on nousussa. Tekniikoiden kehitys avaa uusia mahdollisuuksia pelien kehitykseen entistä pienemmillä resursseilla. Nykyään myös innovatiiviset pelit voivat olla menestyksekkäitä, eivätkä ne vaadi suurten pelitalojen tukea taakseen.

Opinnäytetyön aiheena on suunnitella ja toteuttaa 3D-reaaliaikastrategiapeli hyödyntäen OpenGL-grafiikkakirjastoa. Työn yhtenä lähtökohtana pidettiin, ettei valmiina löytyviä ratkaisuja käytettäisi. Peli suunniteltiin käyttöjärjestelmästä riippumattomaksi, joten se toimii sekä Windows- että Linux-järjestelmissä.

Peli on toteutettu käyttäen C++-kieltä sen tarjoaman oliopohjaisuuden ja suoritusnopeuden vuoksi. Grafiikan piirtoon pelissä käytetään avoimeen lähdekoodiin perustuvaa OpenGL-grafiikkakirjastoa, muiden toimintojen suorittamiseen käytetään SDL-kirjastoa. Pelin 3D-mallinnus on toteutettu käyttäen Blender-ohjelmistoa, josta saatavat mallit muunnetaan pelille sopivaksi erillisellä muunnosohjelmalla. Pelikartat on toteutettu erillisellä Simple Tile Map -editorilla.

ABSTRACT

KYMENLAAKSON AMMATTIKORKEAKOULU

University of Applied Sciences

Software Engineering

TAPANINEN, MIKKO

LILJEQVIST, VILLE

Design and Development of a Strategy Game

Bachelor's Thesis — 77 pages + 2 pages of appendices

Supervisor — Teemu Saarelainen, Programme Head

Commissioned by — Kymenlaakson Ammattikorkeakoulu

March 2010

Keywords — OpenGL, C++, SDL, game programming

The increasing popularity of computer games has had a positive impact on game development. New technologies open up new opportunities for game developers with fewer resources. Today, also small innovative games can be successful and do not require support from major video game publishers.

The purpose of this study was to design and implement a 3D real-time strategy game using the OpenGL graphics library. One starting point for the study was to not use any existing solutions. The game was designed to be independent from operating systems, so it works with both the Windows and Linux systems.

The game was developed by means of the C++ programming language because of its object-oriented structure and its high performance. The graphics were drawn using the open-source OpenGL graphics library. For other functions, the game uses the SDL library. 3D modeling was carried out using the Blender software. Blender models were imported to a separate conversion program, which converted the models to a form that the game understands. The game maps were developed with Simple Tile Map Editor-application.

TERMIT JA LYHENTEET

A* Hakualgoritmi, jota reitinhaussa on käytetty.

Alfa-kanava
Alfakanava tai läpinäkyvyyskanava on mahdollinen lisäkanava värikanavien lisäksi tietokoneella esitettävissä kuvissa.

Binääri Tietokoneen pienin mahdollinen tiedonmuoto, joka käyttää tiedon esittämiseen vain kahta merkkiä. Yleisesti käytetyt symbolit ovat 1 ja 0.

Blackbox Eräs ohjelmistojen testaukseen käytettävä menetelmä. Testauksessa testit suoritetaan näkemättä ohjelman toteutusta ulkopuolisen tarkkailijan silmin.

Bug tracking system
Virheidenseuraamisjärjestelmä, jolla pidetään kirjaa löydetyistä virheistä sekä niiden tilasta.

C++ Yleiskäyttöinen järjestelmäohjelmointiin painottuva ohjelmointikieli.

Code::Blocks
Ilmainen alustariippumaton C++-ohjelmistoympäristö, jota opinnäytetyön tekemisessä käytettiin.

CSS (Cascading Style Sheets) on erityisesti www-sivustoilla käytetty tyylisäännöstö, joka määrittelee sivuston ulkoasun.

CVS (Concurrent Versions System) Versionhallintajärjestelmä.

Frame limiter
Ruudun päivitysnopeuden rajoitin. Rajoittaa, kuinka monta kertaa ruutua päivitetään sekunnin aikana.

Gcc (GNU Compiler Collection) Ilmainen tietokoneohjelma, joka kääntää ohjelmointikielisen lähdekoodin konekieliseen binääriseen muotoon.

GLU (OpenGL Utility Library) OpenGL:n apuna käytettävä kirjasto, joka sisältää funktioita eri grafiikkatoimintojen suorittamiseen.

Hud (Heads up display) Nimitys, jota käytetään pelienkehityksessä kuvaamaan erilaisia tietoruutuja, jotka ovat jatkuvasti esillä pelimaailman päällä.

Komentorivi
Komentorivi on käyttöympäristö, jossa tietokoneelle annetaan tekstikomentoja.

Minimap Peleissä käytettävä kartan pienoismalli, josta ilmenee myös vihollis- ja omien pelaajien sijainti.

MVC (Model View Controller) Arkkitehtuurimalli, joka erottaa toiminnan ja näkyvän sisällön toisistaan, jolloin ohjelman loogisuus ja tietoturva parantuvat. Myös ohjelman visuaalisen ulkoasun päivittäminen helpottuu huomattavasti.

Ogg Vorbis
Patentiton ja lisenssimaksuton äänenpakkausmenetelmä, joka pohjautuu avoimeen lähdekoodiin.

OpenGL (Open Graphics Library) Käyttöjärjestelmäriippumaton kirjasto 2D- ja 3D-grafiikan piirtoon.

PHP (PHP: Hypertext Preprocessor) Ohjelmointikieli dynaamisten Internet-sivujen luomiseen web-palvelinympäristössä.

PLEI-pelimoottori
Opinnäytetyöhön kehitetty pelimoottori.

PNG (Portable Network Graphics) Häviötön bittikarttagrafiikan tallennusformaatti, jota käytetään kaikissa PPC:n grafiikoissa.

PPC (Pulse Phase Chronicles) Lyhenne opinnäytetyönä tehdystä pelistä.

Repository (Versioarkisto) Nimitystä käytetään versionhallintajärjestelmissä palvelimella sijaitsevasta varastotilasta.

SDL (Simple DirectMedia Layer) C-kielellä toteutettu multimediakirjasto.

Skripti Pieni tietokoneohjelma, jota ei käännetä, vaan se suoritetaan käsky käskyltä.

SVN (Subversion) Versionhallintajärjestelmä.

TrueType Nykyään yleisin käytössä oleva kirjasinformaatti eli fonttityyppi.

Zend Framework
PHP:llä toteutettu vapaan lisenssin luokkakirjasto.

W3C (World Wide Web Consortium) Kansainvälinen yritysten ja yhteisöjen yhteenliittymä, joka ylläpitää ja kehittää WWW:n standardeja ja suosituksia.

Web-palvelin
Tietokone tai ohjelmisto, joka jakaa dokumentteja HTTP-protokollalla asiakasohjelmille ja koneille.

XHTML (Extensible Hypertext Markup Language) Internet-sivuissa käytetty kuvauskieli, joka täyttää XML:n vaatimukset.

XML (Extensible Markup Language) Rakenteellinen kuvauskieli, joka auttaa jäsentämään tietomassoja selkeästi.

SISÄLLYS

1 JOHDANTO

Opinnäytetyön tarkoituksena oli suunnitella ja toteuttaa ohjelmistoprojekti, jossa käytetään luovasti opittuja ohjelmointikieliä ja menetelmiä. Valmiina saatavilla olevia ratkaisuja pyrittiin välttämään, ja mahdolliset ongelmat pyrittiin ratkaisemaan omatoimisesti.

Aihevalintaan vaikuttivat edelliset ohjelmistoprojektit, jotka olivat myös pienimuotoisia pelejä. Opinnäytetyöstä haluttiin haastava mutta mielenkiintoinen, joten peliaiheinen ohjelmistoprojekti oli luonteva valinta. Edellisistä projekteista poiketen opinnäytetyössä käytetään tekoälyä ja pelitapahtumat sijoittuvat 3D-maailmaan.

Pelille asetettuja vaatimuksia olivat mm. usean tekoälypelaajan toiminta samanaikaisesti, 3D-maailma, pelikenttien ja 3D-mallien lataus tiedoista sekä monipuolisesti toimiva pelimoottori jatkokehitystä ja mahdollisia tulevia projekteja varten. Lisäksi pelimoottorin sekä pelin tulisi toimia käyttöjärjestelmästä riippumatta. Koska peli osallistuu vuoden 2010 ASSEMBLY gamedev -kilpailuun, sen tulee täyttää kilpailuun osallistumisen vaatimukset.

Pelimoottorin suunnittelussa huomioitiin erityisesti sen laajennettavuus, suoritusnopeus sekä helppokäyttöisyys. Oliopohjaisuuden ja suoritusnopeuden vuoksi ohjelmointikieleksi valikoitui C++, joka on vastaavissa ohjelmistoprojekteissa yleisin käytetty kieli. Grafiikan piirtoon valittiin OpenGL-grafiikkakirjasto, joka on suosittu ohjelmistorajapinta graafisia toimintoja varten. OpenGL-kirjaston tukiessa vain grafiikan piirtoa. Muiden toimintojen suorittamiseen opinnäytetyössä käytettiin SDL-kirjaston tarjoamia toimintoja.

Toteutus aloitettiin pelimoottorin tärkeimpien osa-alueiden ohjelmoimisella ja käytettävien kirjastoiden liittämisellä pelimoottoriin. Ennen varsinaisen pelin ohjelmoinnin aloitusta pelimoottorin tulee olla toimiva ja todettu vakaaksi alustaksi sitä hyödyntävälle sovellukselle. Pelin vaatimusten lisääntyessä myös pelimoottoria täytyi laajentaa vastaamaan tarvittavia ominaisuuksia.

Pelin nimeksi valikoitui Pulse Phase Chronicles, joka on kolmas osa Pulse Phase -pelisarjaa, johon kuuluu myös kaksi edellistä ohjelmistoprojektia. Pul-

se Phase Accretion on pelisarjan ensimmäinen tuotos, joka on sivusuunnasta kuvattu 2D-peli. Toinen tuotos Pulse Phase Brotherhood puolestaan on dynaaminen internetpohjainen strategiapeli, joka on toteutettu PHP-ohjelmointikielellä hyödyntäen Zend Framework -alustaa.

Pelin toteutus alkoi valikkorakenteen suunnittelulla ja sen toteutuksella pelimoottoria hyödyntäen, minkä jälkeen peliin lisättiin 3D-mallit sekä toteutettiin pelissä käytettävä 3D-maailma. Pelimaailman valmistuessa valmiista pelistä puuttuu pelilogiikka sekä tekoäly, jotta sitä voitaisiin testata pelattavana kokonaisuutena.

Testaus on tärkeä osa kaikkia ohjelmistoprojekteja. Pelin testaus suoritettiin suljettuna blackbox-testauksena, jossa testaajat eivät tiedä, mitä pelin tulisi tehdä, vaan dokumentoivat mahdollisesti tapahtuvat virheet sekä epäloogisuudet. Testauksessa ilmenneet virheet raportoitiin projektissa käytettyyn ohjelmointivirheiden seurantajärjestelmään, josta ilmenee projektissa vielä korjaamattomat virheet sekä kaivatut muutokset.

Työ on tehty Ohjelmistoakatemialla, Datariinassa, Kotkassa. Edistymistä on kiinnostuneena seurannut sekä antanut hyvää palautetta ja ehdotuksia, Ohjelmistoakatemian väki.

2 KÄYTETYT TEKNOLOGIAT

PPC on ohjelmoitu käyttäen C++-ohjelmointikieltä, ja apuna on ollut grafiikka- ja multimediakirjastoja. Kehitystyökaluna käytettiin vapaan lähdekoodiin Code::Blocks -ohjelmaa apunaan gcc-kääntäjä, jolla itse ohjelma käännettiin.

Projektin ollessa laaja sen lähdekoodien hallintaan käytettiin versionhallintajärjestelmää. Versionhallinnalla pystyttiin kontrolloimaan usean käyttäjän samanaikaisesti tekemiä muutoksia sekä pitämään tallessa vanhojen versioiden lähdekoodit.

Opinnäytetyönä toteutetun pelin Internet-sivusto päätettiin tehdä käyttäen PHP-ohjelmointikieltä sen monimuotoisuuden vuoksi. PHP:n lisäksi sivuston toteuttamisessa käytettiin Zend Framework -kirjastoa, sen ollessa kevyt sekä yksinkertainen.

2.1 C++

C++ on laajassa käytössä oleva ohjelmointikieli, jonka kantakielenä toimii C-kieli. C++ tunnetaan erityisesti sen luotettavuudesta sekä erityisen hyvästä suoritusnopeudesta, jonka vuoksi C++ on suosittu ohjelmointikieli esimerkiksi pankkialan sovelluksissa ja grafiikkaa hyödyntävissä ohjelmistoissa.

C++-kieli kehitettiin C-kielestä 1980-luvulla Bjarne Stroustrupin toimesta. Kieltä suunnitellessa painotettiin kielen määritelmän yksinkertaisuuteen sekä yhteensopivuuteen C-kielen kanssa. [1]

Oliopohjaisuus on C++:n tärkeimpiä ominaisuuksia. Oliot ovat luokkien ilmentymiä, jotka pitävät sisällään tietoa ja toiminnallisuutta. Jokaista oliota voidaan pitää omana kokonaisuutenaan, joka pystyy vastaanottamaan tietoa ja välittämään sitä edelleen toisille olioille. [2]

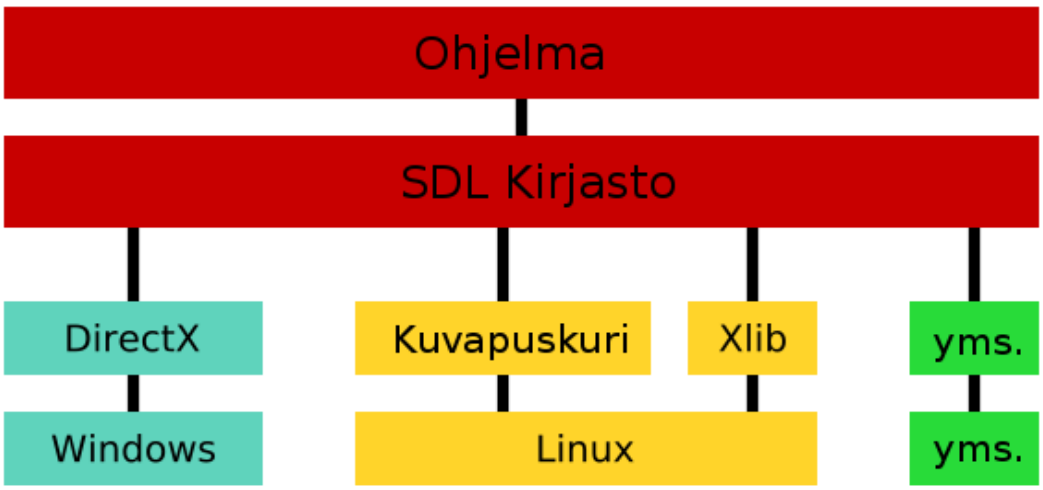
Alustariippumattomuuden ja skaalautuvuuden vuoksi C++ soveltuu suuriin ja pieniin ohjelmistoprojekteihin kaikenlaisilla suoritinalustoilla. Ohjelmoijalle C++:n valmiit standardikirjastot tarjoavat varmatoimiset työkalut projektien toteuttamiseen eri alustoilla.

C++:n muistinhallinta on manuaalista, joka tarkoittaa, että ohjelmoijan tulee varata ja vapauttaa muisti ohjelmassaan. Muistin hallinta on manuaalista, koska manuaalinen muistinhallinta on suorituskyvyltään tehokkaampaa. Riskinä on kuitenkin, että muisti unohdetaan vapauttaa, joka johtaa muistivuotoihin, tai sitä ei muisteta varata, jolloin ohjelma toimii odottamattomalla tavalla.

C++-kääntäjä muuntaa ohjelmakoodin käyttöjärjestelmän ymmärtämään binäärimuotoon. Kaikki standardia noudattavat C++-ohjelmakoodit voidaan muuntaa kaikille suoritinalustoille sopiviksi sovelluksiksi. [1]

2.2 SDL

Simple DirectMedia Layer on käyttöjärjestelmästä riippumaton multimediakirjasto, joka toimii eräänlaisena kerroksena PC-laitteiston ja ohjelmiston välillä (Kuva 1). SDL:n kehitti Sam Lantinga vuonna 1998 työskennellessään Loki Softwarella. Kirjasto sopii hyvin 2D-grafiikanpiirtoon sekä syötteiden lukuun. SDL on ohjelmoitu C-kielellä, mutta sitä voi hyödyntää myös C++:ssa ilman muutoksia. SDL toimii myös seuraavilla kielillä: Ada, C#, D, Eiffel, Erlang, Euphoria, Guile, Haskell, Java, Lisp, Lua, ML, Objective C, Pascal, Perl, PHP, Pike, Pliant, Python, Ruby, Smalltalk, ja Tcl.



Kuva 1. SDL-tason sijainti ohjelmiston ja PC-laitteiston välissä.

SDL -kirjastoon voidaan liittää myös useita lisäkirjastoja, jotka hallinnoivat muun muassa äänenkäsittelyä, verkkoliikennettä, kuvanlatausta sekä fonttienlatausta. SDL liitetään usein myös OpenGL -grafiikkakirjaston rinnalle tuomaan ohjelmistoon rajapinnan näppäimistön sekä hiiren kontrollien lukemiseen, joita OpenGL ei kykene hallinnoimaan. [3]

2.3 OpenGL

Open Graphics Library on SDL:n tavoin laitteistosta riippumaton ohjelmointirajapinta, jota käytetään grafiikan piirtoon. OpenGL toimii laitteiston ja ohjelmiston välillä, mutta se sisältää vain grafiikan piirtoon tarvittavat toiminnot. Silicon Graphics julkisti OpenGL:n vuonna 1992. Siitä lähtien OpenGL on kehittynyt hiljalleen yhdeksi suosituimmaksi 2D- ja 3D-piirtoon käytetyksi ohjelmointirajapinnaksi. OpenGL tukee muun muassa seuraavia kieliä: C, C++, Fortran, Ada, Java.

OpenGL:ää pidetään hyvin vakaana ja luotettavana rajapintana. Se tarjoaa nopeasti kehittyvän ympäristön, joka päivittyy jatkuvasti uusien laitteisto innovaatioiden mukana. OpenGL:n käyttö ei rajoitu vain PC-laitteistoihin, vaan sitä hyödynnetään, niin sulautetuissa järjestelmissä kuin supertietokoneissakin. Ohjelmistokehittäjien kannalta OpenGL on suunniteltu helppokäyttöiseksi, sillä se suorittaa kaikki laitteistopuolen toiminnot, ja ohjelmoija voi keskittyä pelkästään rajapinnan tarjoamien funktioiden käyttämiseen. [4]

Mobiililaitteisiin on olemassa räätälöity versio OpenGL:stä. OpenGL ES (Open Graphics Library Embendded Systems) on sulautettuihin järjestelmiin tarkoitettu kevennetty versio ohjelmointirajapinnasta, jota käytetään yleisesti älypuhelimissa sekä pelikonsoleissa. OpenGL:stä on olemassa myös versio kriittisille järjestelmille, kuten sotilas- ja ilmailualan sovelluksiin. OpenGL SC (Open Graphics Library Safety Critical) on muokattu versio OpenGL:stä, josta on poistettu vain pelikäyttöön tarkoitetut funktiot, tehden siitä yksinkertaisen ja luotettavan käyttää. [5;6]

2.4 PHP / Zend Framework

PHP: Hypertext Preprocessor on laajalti käytetty vapaaseen lähdekoodiin perustuva ohjelmointikieli, jota käytetään dynaamisen sisällön tuottamiseen Internet-sivuille. PHP on tulkattava kieli, joka perustuu C-kieleen, Perl-kieleen, sekä Javaan. Sivuston PHP-kielinen sisältö suoritetaan palvelimella ennen sivun lähettämistä eteenpäin, eli PHP ei vaadi sivuston käyttäjää asentamaan lisäosia.

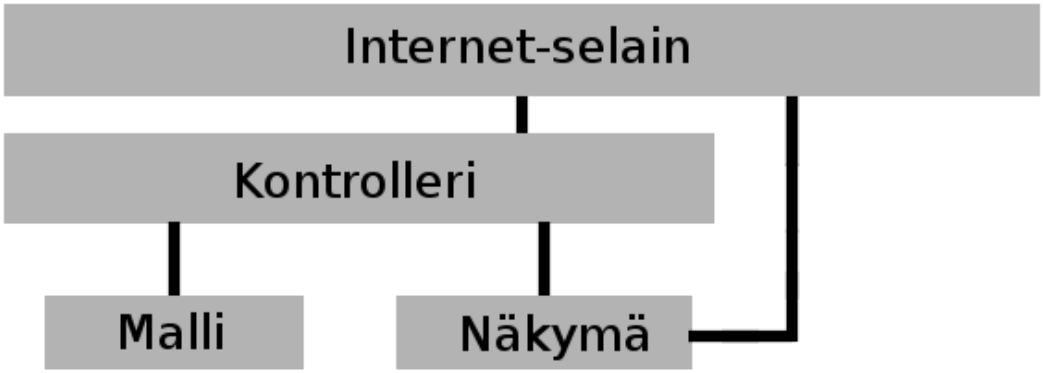
Rasmus Lerdorf julkaisi vuonna 1994 kirjoittamansa C-kieliset skriptit, joista myöhemmin kehittyi suosittu PHP-ohjelmointikieli. Nykyään tunnettu PHP sai varsinaisesti alkunsa vuonna 1998, kun kehitettiin PHP 3.0. Uuden version ydin kirjoitettiin lähes kokonaan uudelleen, ja sen testaamiseen käytettiin paljon aikaa. Tällä hetkellä PHP:stä on kehitteillä kuudes versio, mihin on päädytty useiden päivitysten jälkeen, mutta versioiden väliset muutokset ovat lähinnä olleet ominaisuuksien lisäämisiä.

PHP on alustasta riippumaton kieli, jota voidaan käyttää web-palvelimessa, tai komentorivipohjaisena tulkattavana kielenä. PHP:n voi asentaa Windows-, Unix-, Linux-, Mac-, BeOS-alustoille, ja se toimii samalla tavoin kaikilla kokoonpanoilla.

Web-kehityksen kannalta PHP:n tärkeimpiä ominaisuuksia ovat sen laaja tietokanta- ja viestintäprotokollatuki. Tietokantatyypeistä PHP tukee muun muassa seuraavia: MySQL, PostgreSQL, SQLite, sekä Mssql. [7;8]

Zend Framework on yksi PHP-kielen suosituimpia kehysrakenteita, jonka tarkoituksena on yksinkertaistaa PHP:n avulla tehtyjen sovellusten toteutus. Se sisältää oliopohjaisen rakenneratkaisun ja on erittäin hyvin dokumentoitu. Zend on hyvin monipuolinen työkalu, sillä se sisältää useita hyödyllisiä funktioita PHP-sivustojen rakentamiseen sekä päivittyy jatkuvasti uusien standardien ja trendien mukaan. [9]

Zend Frameworkin käyttäminen pakottaa tekijän oikeaoppiseen luokkarakenteen käyttämiseen ja käyttämään MVC-mallia (Kuva 2). MVC-mallin mukaan toteutetut sivustot ovat varmatoimisia sekä helppoja päivittää.



Kuva 2. MVC-mallin rakenne yksinkertaistettuna.

MVC-mallissa Internet-selain lähettää pyynnön, jonka kontrolleri vastaanottaa. Kontrolleri noutaa tarvittavat sivulla näytettävät tiedot mallin kautta. Malli huolehtii sovelluksen tiedon tallentamisesta, ylläpidosta ja käsittelystä. Näkymä näyttää sivuston ulkoasun ja käsittelee mallilta saamansa tiedot. Näkymä lähettää näytettävän sivun eteenpäin Internet-selaimelle. [10]

Yritysten kannalta Zend Framework on järkevä valinta kehysrakenteeksi, sillä se lisää ohjelmointitehokkuutta sekä sisältää hyödylliset Web 2.0 ominaisuudet. Zend Frameworkin lisenssi on yritysystävällinen, eli yritykset voivat käyttää sitä vapaasti omaan hyötykäyttöönsä.

Zend Framework lisää sivustojen turvallisuutta. Sen koodipohjan testaamisessa ovat olleet mukana useat henkilöt ympäri maailmaa, ja se on todettu erittäin varmatoimiseksi. Myös käyttäjien omien lisäosien valmistus ja testaaminen on tehty helpoksi, joka lisää Zend Frameworkin monipuolisuutta ja laajennettavuutta entistäkin enemmän.[9]

2.5 Subversion

SVN eli subversion on avoimeen lähdekoodiin perustuva versionhallintajärjestelmä, jota käytetään yleisesti ohjelmistokehityksessä hallinnoimaan projektin tiedostoja, kansioita ja niihin liittyviä muutoksia. Tämä mahdollistaa vanhojen versioiden palauttamisen uudelleen käyttöön sekä helpottaa työskentelyä projekteissa, joihin osallistuu useampi henkilö. [11]

SVN:n kehitys alkoi 2000-luvun alkupuolella, kun CollabNet ryhtyi tekemään korvaajaa CVS-versionhallintajärjestelmälle. Perusidea oli tuttu CVS-versionhallintajärjestelmästä, mutta tavoitteena oli toteutus, jossa ei ole ohjelmointivirheitä tai huonosti toimivia ominaisuuksia. SVN-versionhallintajärjestelmän on suunnitellut Karl Fogel, jolla oli jo olemassa suunnitelma paremmasta versionhallintajärjestelmästä. SVN otettiin käyttöön ensimmäisen kerran vuoden 2001 aikana, kun CollabNet ryhtyi käyttämään sitä SVN-projektin versionhallintajärjestelmänä. [11]

SVN:n versioarkisto (Repository) sijaitsee palvelimella, johon kaikki muutokset tehdään. Jokaisella projektiin osallistuvalla tietokoneella on asennettuna asiakasohjelma, jolla tehdyt muutokset lähetetään versioarkistoon. [11]

SVN-versionhallintajärjestelmä sisältää useita hyödyllisiä ominaisuuksia. Versioiden päivittäminen palvelimelle tapahtuu atomisesti, eli muutosten päivitys ei voi jäädä keskeneräiseksi, vaan kaikki version tiedostot päivittyvät tai yksikään tiedosto ei päivity versioarkistoon. Tilansäästämiseksi SVN päivittää vain versioiden väliset muutokset, sekä pakkaa ne pienempään tilaan. [11]

Käytetyimmät SVN-ohjelmat ovat Linux-käyttöjärjestelmällä CollabNet:n kehittämä svn, sekä sen päälle rakennettu graafinen versio esvn. Windows-käyttöjärjestelmällä suosituin ohjelma on TortoiseSVN. [11]

2.6 Code::Blocks

Code::Blocks on avoimeen lähdekoodiin perustuva alustasta riippumaton kehitysympäristö C- ja C++-ohjelmointikielille. Code::Blocks itsessään sisältää vain perustoiminnot, mutta sen toiminnallisuutta pystytään lisäämään laajennusosilla, joita on tarjolla useita moniin eri tarkoituksiin. Code::Blocks perustuu liitännäisarkkitehtuuriin, eli se on pyritty pitämään mahdollisimman kevyenä. Laajennusosia on runsaasti tarjolla, joten jokainen voi räätälöidä Code::Blocks -ohjelmasta omanlaisensa tarpeidensa mukaan, ja siihen on myös helppo kehittää omia laajennusosia.

Code::Blocks on suunniteltu toimimaan usealla eri kääntäjällä, joten sitä voidaan käyttää monilla eri kehitysalustoilla. Code::Blocks hyödyntää myös moniydintekniikkaa, eli se pystyy kääntämään ohjelmaa usealla ytimellä samaan aikaan, joka nopeuttaa sen kääntämistä huomattavasti.

Kääntäjät, joita Code::Blocks tukee, ovat GNU GCC, MinGW GCC, MSP430 GCC, TriCore GCC, PowerPC GCC, Apple GCC, Microsoft Visual C++ Toolkit 2003, Microsoft Visual C++ 2005, Borland's C++ Compiler 5.5, DigitalMars C/C++, OpenWatcom, Intel C++ Compiler, Small Device C Compiler, DigitalMars D, GDC D Compiler. [12]

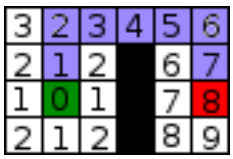
3 PLEI-PELIMOOTTORI

PLEI-pelimoottori on suunniteltu niin, ettei se ole sidottuna vain yhteen projektiin, tässä tapauksessa opinnäytetyönä tehtyyn peliin. Se sisältää tarvittavat ominaisuudet ja funktiot useimpien peli-projektien toteuttamiseen. Pelimoottorin eri osa-alueet ovat itsenäisiä kokonaisuuksia, eli niitä voidaan tehdä helposti lisää tai laajentaa jo olemassa olevia.

Toteutuksessa käytettiin Code::Blocks-kehitysalustaa sekä C++-ohjelmointikieltä, jonka avulla pelimoottoriin rakennettiin looginen luokkarakenne. Luokkarakenne helpottaa pelimoottorin toimintojen käyttöä, laajennettavuutta sekä sen myötä myös käyttömahdollisuuksia.

3.1 Reitinhaku

Reitinhaku on toteutettu A*-hakualgoritmia käyttäen. A* on yksi tunnetuimmista reitinhakualgoritmeista ja on laajalti käytössä peleissä sen nopeuden vuoksi. Pelimoottorissa A*:n käyttöön päädyttiin juurikin sen nopeuden vuoksi, sillä peleissä reitinhakua kuormitetaan todella paljon, ja reitinhaun hitaus vaikuttaisi pelin pelattavuuteen mahdollisesta dramaattisesti.



Kuva 3. Esimerkki A*-hakualgoritmin hakemasta reitistä ja sen käyttämistä kustannuksista.

A*-algoritmi käyttää reitin laskemiseen etäisyyttä ja kustannusta (Kuva 3). Kustannusfunktio g(x) on kustannusarvo aloituspisteestä nykyiseen pisteeseen, kun taas etäisyysfunktio h(x) tarkoittaa heuristista arviota matkan pituudesta päätepisteeseen. Etäisyyttä voidaan laskea usein eri tavoin, pelimoottorissa käytetään Manhattan-tapaa etäisyyden laskemiseen. Lyhin reitti määritellään funktion f(x) = g(x) + h(x) mukaan.

Manhattan-tavassa lasketaan yhteen vaaka- ja pystysuoraan liikuttujen ruutujen määrä, joka tarvitaan päätepisteeseen siirtymiseen ja kerrotaan tämä kustannuksen määrällä. Kun mahdolliset reitit ja niiden kokonaiskustannukset on laskettu, valitaan edullisimman reitin sisältämät pisteet.

A* luokittelee mahdolliset pisteet kahteen eri listaan: avoimiin ja suljettuihin. Aluksi kaikki pisteet kuuluvat avoimeen listaan, pisteitä siirretään suljettujen listaan aina, kun niihin ei voida kulkea tai pisteen kustannus on laskettu. Reitinhaku lopetetaan, kun päätepiste siirretään suljettujen listalle eli reitti on löytynyt tai jos ei ole enää avoimia pisteitä jäljellä eli mahdollista reittiä ei ole olemassa. [13]

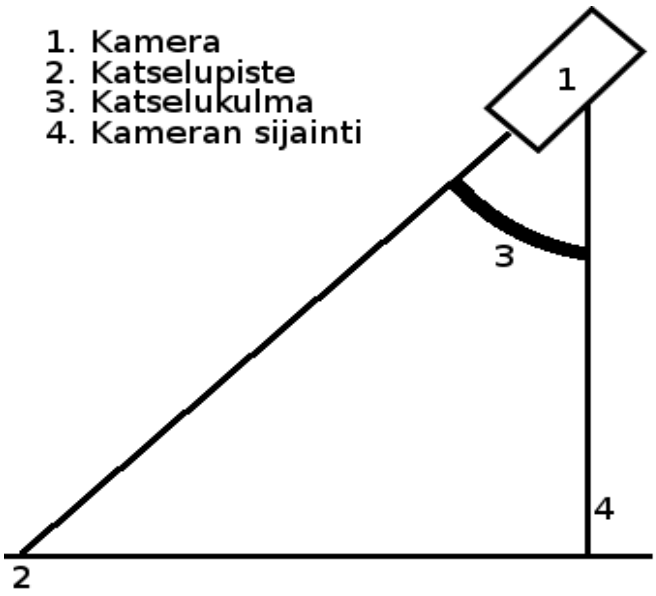
Pelimoottorissa päätepisteen ollessa varattu, reitinhaku etsii lähimpänä vapaana olevan pisteen, jota reitinhaku pitää päätepisteenä ja palauttaa reitin kyseiseen pisteeseen. Reitinhaku huomioi myös kartan pisteen nopeuskertoimen, joka vaikuttaa pisteen kustannukseen. Näin ollen karttaan voidaan tehdä erityyppisiä alueita, kuten vettä, hiekkaa, tai nurmea.

Reitinhaun raskauden vuoksi pelimoottorissa otetaan huomioon myös sen hetkellinen kuormitus. Reitinhaun käytössä on laskuri, joka rajoittaa hakujen suorittamisen määritettyyn määrään jokaisella ruudun päivityskerralla. Suorituskertojen määrä on helposti vaihdettavissa tarvittaessa.

3.2 Kamera

Kamera on suunniteltu mahdollisimman helpoksi käyttää. Se sisältää tarvittavat funktiot kameran liikuttamiseen, zoomaamiseen sekä katselukulman vaihtamiseen. Kamera on aina kohdistettuna katselupisteeseen, joka määräytyy kameran sijainnin sekä katselukulman mukaan (Kuva 4).

Kameran sijainnin ja kulman muutosten matemaattiset toimitukset on toteutettu matriisilaskennalla. Matriisilaskenta nopeuttaa 3D-maailman laskutoimituksia, ja selkeyttää myös ohjelmakoodin luettavuutta ja ymmärrettävyyttä.



Kuva 4. Kameraluokan keskeisimmät käsitteet.

Kamera siirtämiseen ja kiertämiseen on asetettu rajat, jottei kamera pääse liikkumaan epäloogisiin asentoihin. Kameran liikkumisessa otetaan huomioon myös pelimaailman koko, eikä näin ollen kamera voi liikkua pelialueen ulkopuolelle. Kiertäminen on myös rajoitettu maksimissaan 90-asteeseen eli suoraan ylhäältäpäin olevaan katselukulmaan sekä minimissään 15-asteeseen.

Kameraluokassa sijaitsevat myös funktiot, jotka palauttavat kameran sijainnin, sekä katselupisteen, jotka mahdollistavat kameran sijainnin mukaan määräytyvien toimintojen toteuttamisen sovelluksissa.

3.3 Asetukset

Pelimoottori sisältää 3 erilaista asetustyyppiä: Config, joka on tarkoitettu pysyvien asetustietojen tallentamiseen; Rekisteri, jota käytetään tietojen tallentamiseen sovelluksen käynnissäoloaikana; Tulokset, jota käytetään mahdollisten huipputulosten tallentamiseen ja ylläpitämiseen.

Config on asetustyyppi, joka tallentaa tiedot binäärityyppiseen tiedostoon. Config lukee asetustiedoston ohjelman käynnistyessä. Mikäli asetustiedostoa ei löydy, se lataa sille määritetyt oletusmuuttujat. Config pitää tietojaan yllä avain-arvo-periaatteella, eli jokaiselle tiedolle on olemassa indeksi, jolle anne-

taan arvo. Esimerkiksi indeksille ”ikkunan korkeus” voidaan antaa arvo ”600”, jolloin asetustiedostoon tallentuu ikkunan korkeudeksi arvo 600. Config-asetustiedoille on asetettu rajoitukseksi, että avaimen täytyy olla merkkijono ja arvon tulee olla kokonaisluku. Ohjelman sulkeutumisen yhteydessä config-asetustyypin tieto tallennetaan erilliseen tiedostoon tietokoneen kiintolevylle.

Rekisteri puolestaan toimii config-asetustyypin tavoin, mutta se ei tallenna tietoa erilliseen tiedostoon, vaan unohtaa tietonsa ohjelman sulkeutumisen yhteydessä. Eroa config-asetustyyppiin on myös se, että rekisteriin syötettävä arvo on merkkijono. Rekisteriä käytetään esimerkiksi tallentamaan tietoa, jota käytetään vain väliaikaisesti, eikä sitä tarvitse tallentaa pysyvästi mihinkään.

Tulokset asetustyyppiä käytetään tulosten ylläpitämiseen. Tieto tallennetaan config-asetustyypin tavoin binäärimuotoiseen tiedostoon, josta se ohjelman käynnistyksen yhteydessä ladataan käyttöön. Tulosta lisätessä ohjelma huolehtii tulosten paremmuusjärjestyksestä ja lajittelee ne annetun pistemäärän mukaan. Ohjelman sulkeutuessa tulokset tallennetaan tietokoneen kiintolevylle, mikäli tietoihin on tullut muutoksia.

3.4 Tapahtuma-luokka

Tapahtuma-luokka kuuntelee käyttöjärjestelmän lähettämiä viestejä, esimerkiksi sulkemisviestiä ja ikkunakoon muutosviestiä. Käyttöjärjestelmä lähettää viestit esimerkiksi painettaessa ikkunan ylälaidassa olevista painikkeista. Lisäksi tapahtuma-luokka kuuntelee asetettuja näppäinkomentoja, kuten F1-näppäimen painallusta.

Tapahtuma-luokka päivittää omaa tilaansa joka päivityskerralla hakemalla tiedot käyttöjärjestelmältä. Päivitykseen käytetään SDL:n tarjoamia toimintoja, joilla näppäinten- ja hiiren tilat luetaan luokan käyttöön jatkokäsittelyä varten. Päivityksen yhteydessä edellisellä päivityskerralla olleet tilat tallennetaan tilojen vertailua varten.

Näppäimistöpainallusta luettaessa tarkistetaan, että painettu näppäin on olemassa näppäimistökartassa. Näppäintä painettaessa oletuksena luokka käsittelee painalluksen jokaisella päivityskerralla. Tämä ei kuitenkaan ole aina toi-

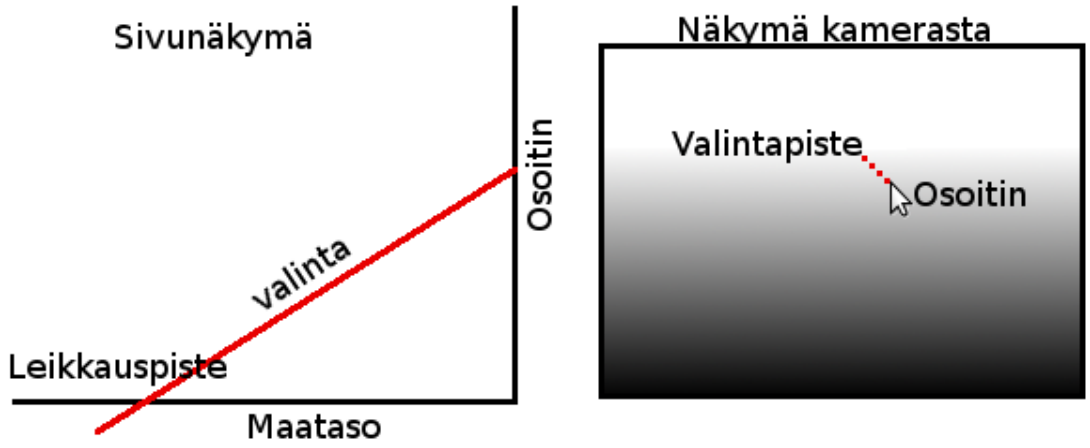
vottavaa, esimerkiksi valikoissa liikkuminen on hankalaa. Ongelman ratkaisuksi luokkaan on toteutettu toiston esto, joka vertailee tilojen eroja. Tämä mahdollistaa näppäinpainalluksen käsittelyn vain kerran yhden painalluksen aikana.

Hiiren nappien painallusten sekä sijainnin käsittely on toteutettu vastaavalla tavalla näppäinpainallusten käsittelyn kanssa, mutta hiiren nappien painallustietoja pystyy muokkaamaan myös manuaalisesti. Näin ollen painallustietoja voidaan mahdollisesti käsitellä vain kerran yhden päivityskerran aikana.

Tapahtuma-luokka sisältää myös sulkemistoiminnon, jonka kautta itse sovellus suljetaan. Sulkemis-toiminto voidaan asettaa aktiiviseksi näppäinkomennoilla, ikkunan ylälaidassa olevista painikkeista tai valikkopainikkeilla. Sovellus lukee jokaisella päivityskerralla sulkemistoiminnon tilan, jos sulkemistoiminto on aktiivinen, sovellus suljetaan.

3.5 Valinta-luokka

Valinta-luokkaa käytetään selvittämään 3D-maailmassa sijaitseva piste johon hiiren osoittimella klikataan, sekä valitsemaan pisteessä mahdollisesti sijaitseva pelaajan objekti.



Kuva 5. Kuva selventää hiiren osoittimen pisteen 2D-3D muunnosta.

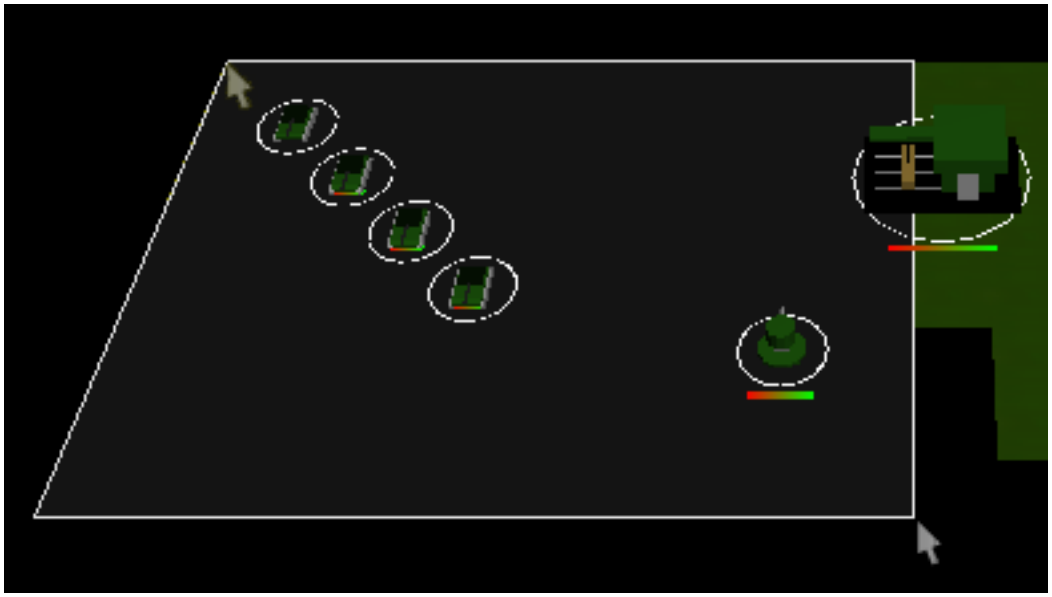
Kameranäkymässä hiiren osoittimen sijainti ei ole sama kuin pelimaailmassa oleva 3D-maataso. Osoittimen koordinaatit täytyy muuttaa ikkunan 2D-

koordinaateista pelimaailman 3D-koordinaatteihin, ennen kuin niitä voidaan hyödyntää.

Osoittimen 2D-koordinaateista ammutaan säde kameran kulman mukaisesti kohti 3D-maailman maatasoa (Kuva 5). Valintapiste on maatason ja ammutun säteen leikkauspisteessä, josta johtuen korkeuskoordinaatti on aina nolla.

Luokalle annetaan pelaajan objektilista, joka sisältää kaikki pelaajalle kuuluvat rakennukset ja joukot, jotka tallennetaan luokkaan. Objektien valintaan käytetään muistissa olevaa objektilistaa.

Päivityksen yhteydessä, mikäli hiiren painiketta on painettu, haetaan 3D-koordinaatit painetulle pisteelle. Napin ollessa painettuna valitaan objektit, jotka ovat painamispisteen alun ja lopun sisältämällä alueella. Suorakulmion muotoinen valinta-alue muodostuu painamispisteen alun ja lopun 3D-koordinaateista. Valitun objektin ei tarvitse olla kokonaan valinta-alueen sisäpuolella, vaan riittää, että objektin suoja-alue leikkaa valinta-alueen rajan (Kuva 6).



Kuva 6. Valinta-alueen muodostuminen ja objektien valinta.

Valinta-alueen sisäpuolelle jäävät objektit valitaan ja lisätään valittujen objektien listaan. Valittujen objektien listaa käytetään objektien eri toimintojen toteuttamiseen sovelluksen muissa vaiheissa, kuten valittujen objektien tietojen esittämiseen.

3.6 Poikkeukset

Poikkeuksia käytetään ilmaisemaan ohjelman ajon aikana tapahtuneita virheitä. Poikkeukset sisältävät virheiden kuvauksen, joka mahdollisesti kirjoitetaan lokiin. Virhetilanteessa sovellus antaa poikkeuksen, joka käsitellään myöhemmin koodissa. Poikkeuksen käsittelemättä jättäminen aiheuttaa ohjelman sulkeutumisen, ja sattuneen virheen jäljittäminen on haastavaa.

Poikkeuksia on kolmea eri tyyppiä: Kriittisiä, jotka estävät sovelluksen käytön jatkamisen ja aiheuttavat sovelluksen hallitun sulkemisen; Normaaleja, jotka vaikuttavat olennaisesti sovelluksen toimintaan, mutta sen suoritusta voidaan vielä jatkaa; Pienimuotoiset, jotka eivät varsinaisesti vaikuta näkyvällä tavalla sovelluksen suorittamiseen, mutta ne on hyvä huomioida. Esimerkiksi asetustiedoston puuttuessa virhe kirjataan ja uusi asetustiedosto luodaan.

Pelimoottorissa on myös olemassa kriittisen poikkeuksen erikoistapaus, jota käytetään mahdollisen tiedoston puuttumisen yhteydessä. Tämä poikkeus antaa virheilmoituksen tiedoston puuttuessa, jossa mainitaan puuttuvan tiedoston nimi. Tämä helpottaa huomattavasti sattuneen virheen etsimistä.

3.7 Frame limiter

Frame limiter on rajoitin, jolla säädetään ruudun päivityskertoja sekunnin aikana. Erilaisten tietokonetehojen vuoksi, jotta sovellus toimisi samoin tavoin kaikilla tietokoneilla, käytetään frame limiteriä tasaamaan ruudunpäivitysnopeus asetettuun arvoon kaikentehoisilla tietokoneilla.

Frame limiteriä käytetään yleisesti peleissä, joiden ruudunpäivitysnopeuden tulee pysyä vakiona hyvän pelikokemuksen saavuttamiseksi. Ilman frame limiteriä pelin kevyissä kohdissa ruutua päivitettäisiin liian nopeaan tahtiin, jolloin pelin tapahtumat etenisivät liian nopeasti. Rajoittimen käyttö myös säästää tietokoneen prosessoria turhalta rasitukselta, jolloin muiden toimintojen suorittamiseen jää prosessointiaikaa.

Frame limiter on pelimoottorissa toteutettu mahdollisimman yksinkertaisesti ja kevyesti. Rajoitin laskee ruudun piirron aloituksen sekä lopetuksen välisen ajan ja lisää uuden piirron väliin viivettä asetetun ruudunpäivitys tavoitteen mukaan.

Rajoitin laskee myös sekunnissa tehtävien ruudunpäivitysten määrän, jota voidaan käyttää esimerkiksi tulostamaan kyseinen arvo ruudulle. Ruudunpäivitysnopeuden näytölle tulostaminen helpottaa sovellusten mahdollisten hitaiden kohtien havaitsemista.

3.8 Kuva-luokka

Sovelluksissa käytetään yleisesti paljon kuvia tuomaan parempaa visuaalista ilmettä ja helpottamaan käyttöliittymän suunnittelua ja toteutusta. Lisäksi kuvia käytetään objektien ulkonäön parantamiseen ja teksturointiin.

Kuvan latauksessa sovellukseen käytetään SDL:n tarjoamaa funktiota. SDL:n kuvanlataus tukee useita eri kuvaformaatteja, kuten BMP, GIF, JPEG, LBM, PCX, PNG, PNM, TGA, TIFF, XCF, XPM ja XV. Kuvanlatauksella on kuitenkin yksi rajoitus, sillä kuvakoon tulee olla jaollinen kahdella, koska pelimoottorin haluttiin tukevan OpenGL:n vanhempia versioita. Kuvan latauksessa käytetään OpenGL:n tarjoamia suodattimia kuvan reunakäyttäytymiseen ja pehmennykseen. [14]

Pelimoottorissa on toteutettuna 2D-kuvanpiirtofunktio, joka on helppo ja yksinkertainen keino tulostaa ruudulle kuva, halutulle alueelle, halutun kokoisena. Funktiossa kuva piirretään 2D-tason päälle tekstuurina, jonka jälkeen taso siirretään halutun kokoisena annettuihin koordinaatteihin. Kuvaan voidaan määrittää myös läpinäkyvyyttä, jolla voidaan luoda erilaisia efektejä ja asettaa kuvia toistensa päälle. 3D-sovelluksissa ei yleensä käytetä 2D-kuvanpiirtoa, vaan kuva piirretään erikseen suoraan 3D-maailmaan.

PLEI-Pelimoottori pitää muistissaan kaikki ladatut kuvat. Kuvat on sijoitettu listaan, josta niitä voidaan helposti hallita. Kuvia ladattaessa pelimoottori tarkistaa, onko kyseistä kuvaa ladattu aiemmin, mikäli kuva on jo ladattuna pelimoottorin listaan, sitä ei ladata uudelleen. Mikäli kuvaa ei löydy listasta, kuva ladataan ja lisätään listaan tulevaa käyttöä varten.

Muistinkäytön minimoimiseksi pelimoottori pyrkii pitämään mahdollisimman vähän kuvia muistissa kerrallaan. Tarpeen mukaan kuvat poistetaan muistista, mutta jätetään listaan tulevien latausten nopeuttamiseksi.

Sovelluksen ikkunakoon muuttuessa käytössä olevat kuvat joudutaan lataamaan uudelleen mahdollisten kuvanpiirrossa esiintyvien virheiden välttämiseksi. Pelimoottorissa on toteutettuna uudelleenlatausfunktio, joka poistaa ladatut kuvat muistista ja lataa listassa olevat kuvat uudelleen muistiin.

Pelimoottorin kuvanlatauksessa on pyritty toteuttamaan kokonaisuus, joka toimisi useimmissa näytönohjaimissa, jotka tukevat OpenGL-grafiikkakirjastoa. Toteutuksessa on otettu huomioon myös vanhempien näytönohjaimien rajoitukset ja pyritty yhdistämään nämä rajoitukset uusimpien näytönohjaimien tarjoamaan teknologiaan.

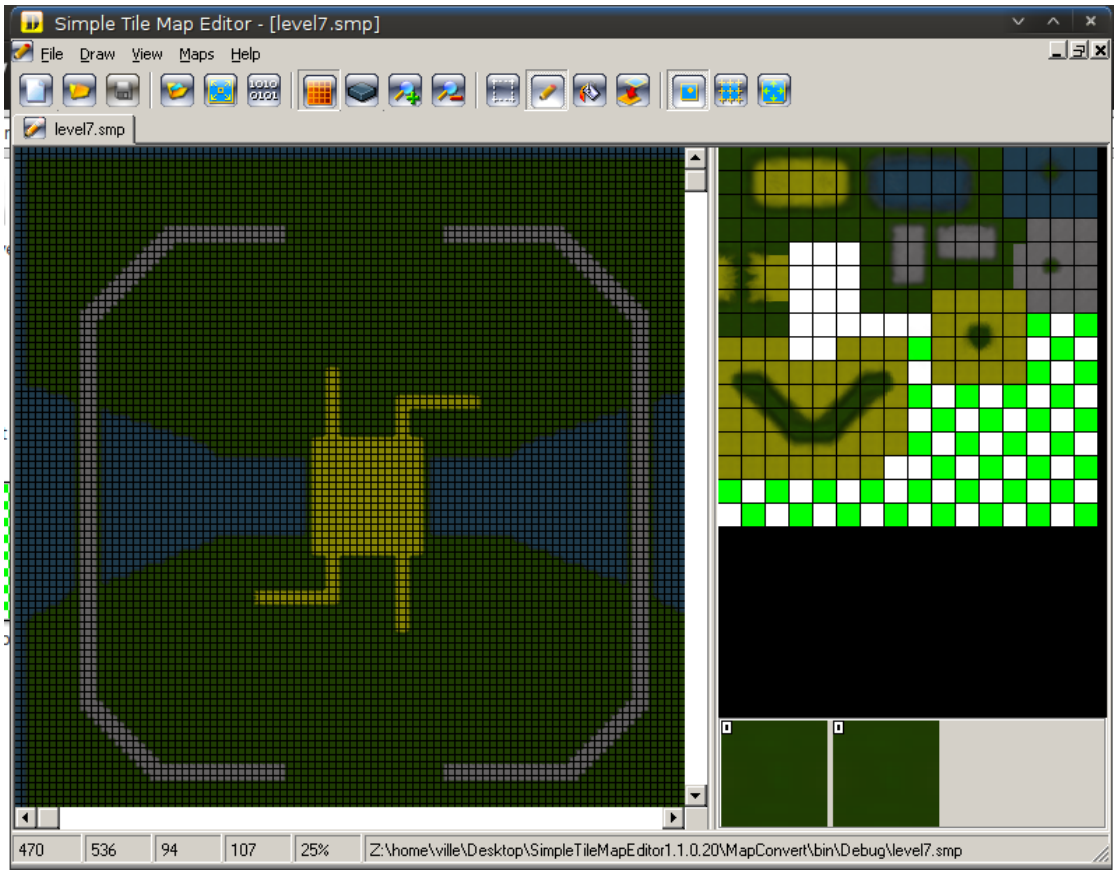
3.9 Kentät

Pelimoottorin käyttämien kenttien teossa ja käsittelyssä on pyritty yksinkertaiseen ja kevyeen toteutukseen. Jotta kenttien käsittely olisi mahdollisimman kevyttä, on kenttien maaston korkeuserot karsittu pois.

Kentät on toteutettu laattakartta-tekniikalla, jossa kartta muodostetaan useasta pienemmästä neliön muotoisesta laatasta. Laattaan piirretään kuva, jolloin kartta muodostuu useasta vierekkäin sijoitetusta pienemmästä kuvasta.

3.9.1 Teko

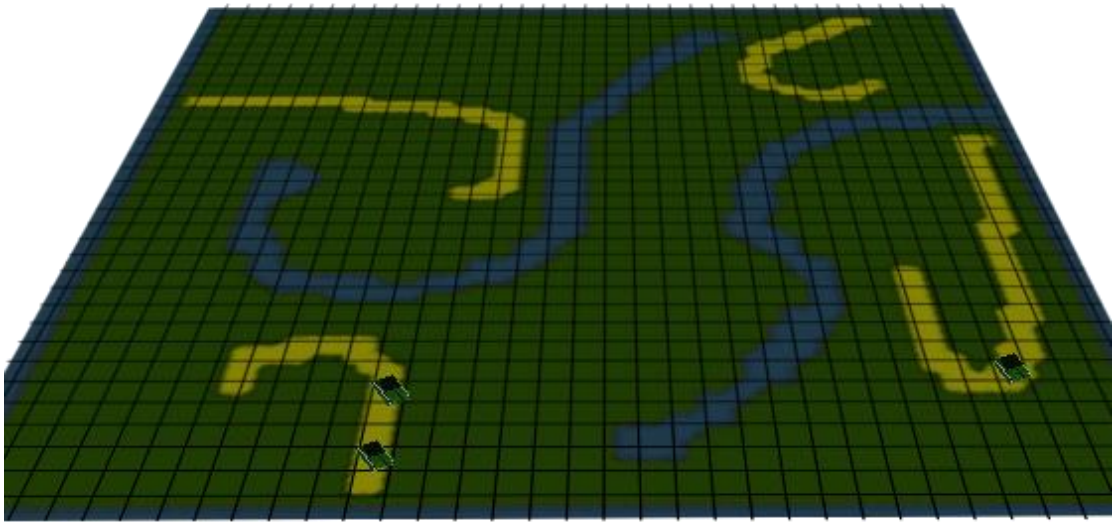
Pelimoottorissa kenttien tekemiseen käytetään Simple Tile Map Editor -ohjelmaa, joka on vapaasti käytettävissä. Ohjelmassa luodaan uusi kartta, jolle määritetään kartan- ja laatan koko ja laattojen piirtoon käytettävä kuvatiedosto (Kuva 7).



Kuva 7. Ruudunkaappaus Simple Tile Map Editor-ohjelmasta ja sen käyttö.

Ohjelma muodostaa käytettävästä kuvatiedostosta laattakoon mukaisia pienempiä kuvia, joita käytetään itse kartan tekemiseen. Kartta rakennetaan siirtämällä ohjelman muodostamia pienempiä kuvia eli laattoja valinta-alueelta kartta-alueelle. Kentän ollessa valmis se tallennetaan ohjelman asettamalla tiedostomuodolla.

Kartan teko on vain ensimmäinen askel kentän valmistamisessa. Kenttään lisätään kartan lisäksi objekteja, jotka luodaan myös Simple Tile Map Editor -ohjelmalla. Ohjelmalla luodaan uusi tiedosto, jossa käytetään laattojen piirtoon objektit sisältävää tiedostoa. Objektikarttaan lisätään samoin tavoin kartan teon kanssa objektilaatat haluttuihin koordinaatteihin (Kuva 8).



Kuva 8. Kentän muodostuminen kartta- ja objektitiedostosta.

Molemmat Simple Tile Map Editorin valmistamat tiedostot syötetään erikseen tehtyyn muuntimeen, joka yhdistää annetut tiedostot yhdeksi karttatiedostoksi, joka ladataan sovellukseen.

3.9.2 Käsittely

Sovellukseen ladataan karttojen teossa käytetty kuvatiedosto, josta pelimoottori muodostaa pienten kuvien OpenGL-listan, joka nopeuttaa huomattavasti kuvien piirtoa. Listaa käytetään myöhemmin sovelluksen kartan piirtämiseen.

Sovellukseen ladataan muunnostyökalulla luotu tiedosto, joka sisältää kentän kartta- ja objektitiedot. Latauksen yhteydessä luetaan kentän leveys ja korkeus sekä niiden mukaan varataan muistia kentän tarvitsema määrä (Liite 2). Tämän jälkeen kentän tiedot ladataan muistiin, ja objektit lisätään 3D-maailmaan. Seuraavaksi kartta piirretään muistissa olevista karttatiedoista käyttäen listaan ladattuja kuvia.

Pelimoottorissa kartan piirron nopeuttamiseksi piirretään vain kameran kuvakulmasta näkyvä alue. Piirto ottaa huomioon kuvakulman vaihdokset ja kameran liikkumisen. Sen vuoksi näkymän ulkopuolelle jäävää aluetta ei piirretä, ja sovelluksissa voidaan käyttää suuriakin kenttiä.

Kuva-luokan tapaan kuvakoon muuttuessa kartan kuvatiedot joudutaan lataamaan uudelleen. Tähän tarkoitukseen on toteutettu kartan uudelleenlataamisfunktio, joka poistaa vanhan listan, sekä poistaa kuvan muistista ja lataa ne uudelleen. Pelimoottori sisältää myös funktiot, joilla voidaan saada selville kentän leveys, korkeus sekä sen maksimi muistipaikka. Näitä funktioita voidaan hyödyntää pelimoottorissa ja pelimoottoria käyttävässä sovelluksessa.

3.9.3 Ruutu-tietotyyppi

Ruutu-tietotyyppi sisältää kartassa sijaitsevien laattojen ominaisuuksia. Näitä ominaisuuksia ovat näkyvyys, varaus, tilaus, laatan numero, laatalla oleva objekti sekä objektin sijainti ja kokotiedot. Näitä ominaisuuksia käytetään laatan piirrossa ja mahdollisesti pelimoottoria hyödyntävässä sovelluksessa.

Näkyvyys-ominaisuus määrittelee, piirretäänkö laattaa. Varaus-ominaisuus ilmaisee, onko laatalla objektia, sillä laatalla voi olla vain yksi objekti kerrallaan. Mikäli laatalla on varaus, reitinhaku ei voi määrittää kyseiselle laatalle päätepistettä. Tilaus-ominaisuus kertoo, onko reitinhaku määrittänyt kyseiselle laatalle päätepistettään, mutta laatalla ei ole vielä objektia.

3.10 Loki

Lokia käytetään kirjaamaan ohjelmistossa tapahtuvat tapahtumat, esimerkiksi virheet, erilaiset lataukset ja poistot. Lokin avulla voidaan seurata ohjelman toimintaa sekä jäljittää sovelluksessa mahdollisesti tapahtuvia virheitä.

Lokia voidaan kirjata kolmella eri tavalla. Tiedostoon tietojen tallentaminen on kirjaustapa, jolla sovelluksessa tapahtuvat tapahtumat tallennetaan erilliseen tiedostoon ohjelman sulkeutuessa. Sovelluksen tallentamaa tiedostoa voidaan tarkastella myöhemmin erillisessä tekstinkäsittelyohjelmassa.

Toinen kirjaustapa, joka pelimoottorin lokista löytyy, on komentoriville tulostus. Tämä tapa kirjaa tapahtumat välittömästi niiden sattuessa komentoriville. Komentoriville tulostaminen mahdollistaa välittömän reagoimisen toteutuneisiin tapahtumiin.

Viimeinen kirjaustavoista tallentaa toteutuneet tapahtumat vain muistiin ja jättää ne tulostamatta. Tapaa käytetään sovelluksen ollessa valmis, jolloin tapahtumien ei haluta tulostuvan eikä näkyvän sovelluksen käyttäjälle.

Jokaisesta tapahtumasta lokiin kirjataan tapahtuma-aika, viesti sekä tyyppi, joita ovat muun muassa testaus, kaikki, normaali, kriittinen ja ei lainkaan. Tapahtuman tyyppiä käytetään suodattamaan tulostuksessa käytetyt, ei halutut, virheilmoitukset. Tällöin lokiin kirjataan vain halutun tyyppiset tapahtumat.

Testaus-tyyppi tarkoittaa sovelluksen testauksessa käytettyjä viestejä, esimerkiksi lataus- ja poistoviestit. Normaali-tyyppi osoittaa sovelluksen kannalta ei-kriittiset tapahtumat, kuten normaali-tyyppiset virheet. Kriittinen-tyyppi käsittää vain kriittiset tapahtumat, kuten kriittiset virheet. Nimensä mukaisesti kaikki-tyyppi kuvaa kaikkia tapahtumia, ja ei lainkaan -tyyppi tarkoittaa, ettei loki kirjaa sitä lainkaan.

Kirjaustapoja voidaan käyttää myös useaa samaan aikaan, jolloin sovellus voi kirjata tiedot muun muassa lokiin ja komentoriville samanaikaisesti. Tämä mahdollistaa muun muassa tietojen välittömän tarkastelun sekä niiden myöhemmän katselun lokitiedostosta.

3.11 Matematiikka

Pelimoottoriin on toteutettu ohjelmointikielestä puuttuvat matemaattiset funktiot, kuten matriisi- ja vektorilaskenta. Matriisilaskentaa tarvitaan erityisesti 3D-maailman koordinaattien laskemiseen, esimerkiksi kameran siirron laskemiseen ja muihin operaatioihin. 3D-maailmassa vektoreita käytetään erilaisiin laskutoimituksiin sekä kuvaamaan objektien sijaintia.

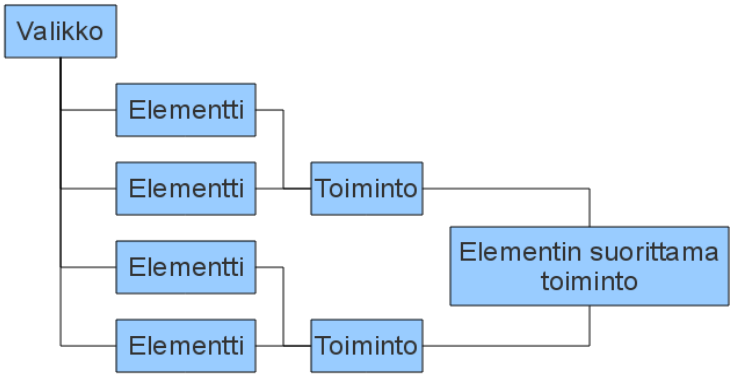
Matriisilaskennassa matriisien kokoa ei ole rajoitettu, jolloin se ei ole sidottuna vain yhteen käyttötarkoitukseen. Lisäksi siihen on toteutettu kaikki matriisilaskennan perusoperaatiot, jotka helpottavat matriisilaskennan käyttöä.

Myös vektorilaskenta sisältää kaikki vektorilaskennan perusoperaatiot, kuten vektoritulon ja skalaaritulon. Pelimoottorissa vektoreita käytetään pääsääntöisesti objektien sijainnin koordinaattien määrittämiseen.

3.12 Menu

Pelimoottori pitää sisällään valikkorakenteen tekemiseen tarkoitetut elementit ja toiminnot. Valikko koostuu rungosta, joka on kuin tyhjä kangas, jolle maalataan siihen haluttuja elementtejä. Runko huolehtii elementtien välisistä toiminnoista ja pitää kirjaa kaikista kyseisessä valikossa olevista elementeistä. Valikkorakenne on suunniteltu skaalautuvaksi erilaisiin sovelluksiin, ja sitä on helppo laajentaa omiin käyttötarkoituksiin.

Elementtien toiminnot suoritetaan niiden ulkopuolella erillisessä toimintoluokassa, joka mahdollistaa useiden eri toimintojen suorittamisen samantyyppisellä elementillä (Kuva 9).



Kuva 9. Esimerkkitapaus valikkorakenteen koostumisesta sekä toimintojen suorittamisesta.

Kaikkien menuelementtien piirto on suoritettu OpenGL:n tarjoamilla piirtofunktioilla, mikä tekee elementtien piirron suorittamisen kevyeksi. Tämä mahdollistaa myös suurien valikkorakenteiden toimivuuden pienitehoisilla tietokoneilla.

Valikossa on valittuna yksi elementti aina, jos se on mahdollista. Valintaa voidaan muuttaa joko viemällä hiiren kursori toisen elementin päälle tai näppäimistön nuolinäppäimillä liikkuen. Itse valinta suoritetaan painamalla hiiren vasemmanpuoleista painiketta tai näppäimistön Enter-näppäintä painamalla.

3.12.1 Valintaruutu

Valintaruutu on neliönmuotoinen valikkoelementti, jolla valinta voidaan laittaa päälle tai pois päältä. Valintaruudulle voidaan antaa myös seliteteksti, joka sijaitsee valintaruudun oikealla puolella. Valinnan ollessa päällä valintaruutuun ilmestyy valintaa osoittava täyttö (Kuva 10).

Kuva 10. Valintaruutu valinnan ollessa pois päältä ja päällä.

3.12.2 Pudotusvalikko

Pudotusvalikkoa käytetään, kun mahdollisia valittavia vaihtoehtoja on useita. Pudotusvalikko säästää myös paljon tilaa, koska se sisältää kaikki valittavissa olevat vaihtoehdot ja näyttää ne vain pudotusvalikon ollessa auki. Sen leveys myös skaalautuu sisällön mukaan sopivaksi.

Pudotusvalikon ollessa suljetussa tilassa se näyttää sen hetkisen valinnan, muut valinnat saadaan näkyviin painamalla pudotusvalikosta. Valinta suoritetaan painamalla uudelleen halutun valinnan kohdalta, jolloin pudotusvalikko sulkeutuu valiten painetun vaihtoehdon. Pudotusvalikolle voidaan antaa myös seliteteksti, joka sijaitsee sen oikealla puolella (Kuva 11).

Kuva 11. Pudotusvalikko sen ollessa suljettuna ja avattuna.

3.12.3 Syöttökenttä

Syöttökenttä on alue, johon voidaan kirjoittaa, ja kirjoitettu teksti voidaan tallentaa myöhempää käsittelyä varten. Syöttökenttää voidaan käyttää esimerkiksi käyttäjän syötteiden lukemiseen (Kuva 12).

Kuva 12. Esimerkkitapaus syöttökentän käytöstä.

Syöttökentän ollessa valittuna se tulkitsee näppäinpainalluksia ja tunnistaa yleisimmät merkit, kuten numerot ja kirjaimet. Rajoituksina syöttökentälle ovat 18 merkin maksimipituus sekä skandinaavisten merkkien puuttuminen.

Syöttökentälle voidaan antaa seliteteksti, joka sijaitsee vasemmassa ylälaidassa. Kirjoituksen seuraamisen helpottamiseksi syöttökenttä sisältää ilmaisimen, joka osoittaa sen hetkisen kirjoituskohdan.

3.12.4 Vierityspalkki

Vierityspalkki on vaakasuunnassa oleva, skaalattava, säätöelementti. Sen avulla on helpompi valita tietyllä alueella olevia arvoja, esimerkiksi äänenvoimakkuuden valinta alueelta 0-100 (Kuva 13).

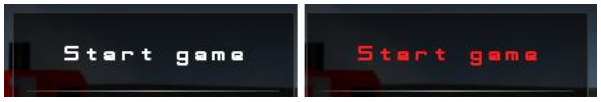
Kuva 13. Vierityspalkin väritys muuttuu sen arvon mukaan.

Vierityspalkkiin on mahdollista asettaa valinta-alueen rajat, jolloin se skaalautuu automaattisesti annettujen rajojen mukaan. Elementin koko on myös asetettavissa, kuten myös sen seliteteksti, joka sijaitsee vierityspalkin oikealla puolella.

Valintaa muutettaessa vierityspalkin väritys muuttuu valitun arvon mukaisesti. Elementti sisältää myös funktion, jolla vierityspalkin sen hetkinen arvo voidaan tallentaa myöhempää käsittelyä varten.

3.12.5 Teksti

Teksti-elementti on nimensä mukaisesti tekstiä sisältävä elementti, jota käytetään esimerkiksi valikon valintapainikkeissa. Teksti-elementti muuttaa väriään ollessaan valittuna, jotta valittu elementti erottuisi muista (Kuva 14).



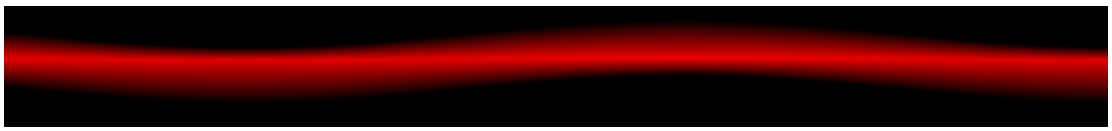
Kuva 14. Teksti-elementti valitsemattomana ja valittuna.

Teksti-elementti voidaan asettaa myös tilaan, jossa sitä ei voida valita, esimerkiksi asetuksissa muutoksien tekemiseen käytettävän päivitys-painikkeen ei tarvitse olla aktiivinen, jos asetuksiin ei ole tehty muutoksia.

3.12.6 Aalto

Aalto-elementti on täysin suunniteltu ainoastaan olemaan valikoiden koriste-elementti, jolla ei varsinaisia toimintoja ole ollenkaan. Se on elementti, jota ei voi valita, eikä valikossa liikkuessa sitä oteta huomioon.

Aalto-elementti on animoitu koriste, joka etenee aaltoilumaisesti kiemurrellen sille asetettujen arvojen mukaisesti. Efekti on toteutettu OpenGL-piirtoa hyväksikäyttäen muuttamalla sen arvoja jokaisella piirtokerralla, jolloin efektistä tulee jatkuvasti päivittyvä (Kuva 15).



Kuva 15. Aalto-elementin efekti.

3.13 Mallit

Malleja käytetään ilmaisemaan objekteja visuaalisessa muodossa. Pelimoottorissa käytetään sille suunniteltuja 3D-malleja, jotka toteutetaan erillisellä ohjelmalla, kuten Blender tai 3D Studio Max. 3D-mallit muodostuvat pisteistä, jotka sijaitsevat 3D-avaruudessa. Nämä pisteet muodostavat erityyppisistä muodoista, kuten kolmioista ja neliöistä.

3.13.1 Mallien lataus

3D-mallit ladataan pelimoottoriin siihen kehitetystä tiedostomuodosta, joka sisältää mallin version, mallissa käytetyt värit, mallin muodostavat pisteet sekä kokotiedot. Ladattavan mallin tulisi sijaita origossa, jotta mallin käsittely sovelluksessa toimisi halutulla tavalla. Mallit saadaan kyseiseen tiedostomuotoon muuntamalla 3D-mallinnusohjelmasta saatu vrlm-tiedosto OpenGL-käskyiksi, jotka siirretään pelimoottorille kehitettyyn muuntotyökaluun. Muuntotyökalu tulkitsee OpenGL-käskyt ja muuntaa ne pelimoottorin ymmärtämään muotoon. Tämän jälkeen malli tallennetaan pelimoottorille kehitettyyn tiedostomuotoon.

Pelimoottorissa mallitiedosto luetaan, ja sen sisältämä tieto puretaan käsittelyä varten. Näistä tiedoista luodaan OpenGL-lista jokaiselle pelaajalle, joka sisältää pelaajan väritiedot. Väritietojen lisäksi listoihin asetetaan valaistuksessa käytettävät tiedot, jotka määräytyvät käytettyjen värien mukaan.

Pelimoottorissa käytetään OpenGL-listoja tietojen tallentamiseen, koska ne ovat nopeampia kuin perinteiset piirtämisfunktiot. Listojen avulla OpenGL-piirtoja ei tarvitse suorittaa kuin niiden luontivaiheessa. Listat sijaitsevat näytönohjaimen muistissa, ja OpenGL optimoi ne parhaimmalla mahdollisella tavalla. Listojen muokkaaminen jälkikäteen ei ole mahdollista, joten listat on luotava aina muutosten tapahtuessa uudelleen. Mallien piirtoa testattaessa listojen käyttö paransi piirtonopeutta noin 50 prosenttia, mutta luku vaihtelee konekohtaisesti. [15]

3.13.2 Objektien perustiedot

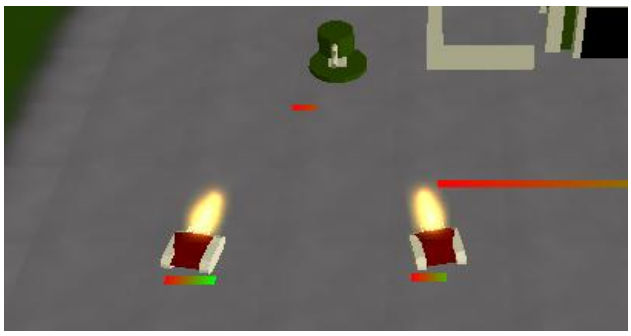
Pelimoottorin mallit sisältävät kaikille objekteille perustiedot, joita ovat sijainti, kulma, koko, malli, energia, maksimi energia, valittu, tuhottu ja rakennettu. Objektien perus-luokka sisältää myös paljon erilaisia toimintoja objektien tietojen muokkaamiseen. Kaikki sovelluksessa olevat objektit saavat nämä ominaisuudet ja toiminnot automaattisesti, jolloin kaikki objektit käyttäytyvät samalla tavalla.

Sijainti sisältää 3D-koordinaatit, jotka ilmaisevat pisteen, jossa objekti sijaitsee. Kulma pitää sisällään arvon jokaisen akselin kiertokulmasta, eli kuinka paljon mallia käännetään akselin suuntaisesti. Koko ilmaisee ladatun mallin leveyden, korkeuden ja pituuden. Mallitieto osoittaa objektissa käytettävään malliin, joka ladataan objektia luotaessa. Energiatiedot sisältävät objektin kunnon eli arvon, joka ilmaisee objektin tilan sekä maksimi arvon, jonka objekti voi saavuttaa. Valittu, tuhottu ja rakennettu ovat totuusarvomuuttujia, joita käytetään ilmaisemaan, onko objekti valittu, tuhottu tai rakennettu.

Objektien perus-luokan sisältämiä toimintoja voidaan käyttää asettamaan arvoja objekteihin ja hakemaan tietoja niistä. Lisäksi luokka sisältää toimintoja, joilla objektia voidaan kiertää 3D-akseleiden ympäri sekä asettaa objektin sijainnin 3D-maailmassa.

3.14 Partikkelit

Partikkeleiksi kutsutaan pieniä kappaleita, joita käytetään luomaan erilaisia efektejä, kuten liekkejä ja räjähdyksiä. Partikkelit ovat itsenäisesti liikkuvia, ja efektien luomiseen niitä käytetään useita. Partikkeleita voidaankin verrata esimerkiksi pölypilven hiukkasiin, mitä enemmän niitä on, sitä suurempi ja tarkempi efekti on (Kuva 16).



Kuva 16. Esimerkki käytetystä liekki-efektistä.

Pelimoottorissa partikkelit ovat toteutettu piirtämällä pieniä kuvia, jotka osittaisen läpinäkyvyytensä sekä värinvaihtokykynsä vuoksi mahdollistavat erilaisten efektien luomisen. Partikkelit on toteutettu kuvia piirtämällä, koska kuvaa vaihtamalla toiseen, voidaan luoda helposti erinäköisiä efektejä.

Näkyäkseen, partikkeleille tulee antaa erilaisia arvoja, joiden mukaan efekti käyttäytyy. Partikkeleita käytetään lisäämällä lähetyspiste, josta efektin suoritus alkaa. Lähetyspiste on 3D-maailman koordinaatistossa, ja sitä ei voida muuttaa efektiä suorittaessa. Lähetyspisteelle annetaan kulma, jonka mukaan partikkeleita lähetetään eteenpäin ja lähetysnopeus joka määrittää kuinka usein lähetyspisteestä lähetetään partikkeli. Partikkeleille määritetään sen koko, eli leveys ja korkeus. Partikkeleille annetaan myös voima, eli arvo joka kertoo kuinka nopeasti se liikkuu. Toimiakseen partikkelit vaativat myös elinikäarvon, joka tarkoittaa kuinka pitkään partikkelia näytetään.

3.15 Äänet

Ääniefektit ja musiikki luovat sovelluksiin tunnelmaa. Pelimoottorin ääni-luokka on suunniteltu tukemaan mahdollisimman useaa eri ääniformaattia, jolloin sen käyttö olisi mahdollisimman laajaa. Suunnittelussa on huomioitu myös äänitehosteiden ja musiikin käyttötarkoitusten erot, jonka vuoksi niiden käytön toteutukset poikkeavat hieman toisistaan.

Äänet toimivat omassa prosessissa erillään pelimoottorista. Tämän myötä äänien käsittely ei vaikuta rasittavasti muiden pelimoottorin toimintojen suorittamiseen, eikä myöskään pelimoottorissa tapahtuva hetkellinen nykiminen vaikuta äänentoistoon.

Pelimoottorissa äänien käyttö on rajattu kahteen erilaiseen käyttömahdollisuuteen, ääniefekteihin ja musiikkiin. Ääniä voidaan soittaa suoraan erillisellä funktiolla, tai mahdollisesti luomalla soittolistoja, joiden avulla tietyt äänikokonaisuudet voidaan yhdistää. Ääniä tai musiikkia voidaan toistaa useilla eri kanavilla, joka mahdollistaa useiden eri äänien samanaikaisen toiston.

Pelimoottorissa äänten käsittelyyn ja toistoon käytetään SDL:n tarjoamia toimintoja ja funktioita. Jotta ääni saataisiin toistettavaan muotoon, se tarvitsee ladata muistiin kokonaisuudessaan. Latauksessa otetaan huomioon tiedoston nimi sekä äänityyppi, eli onko äänitiedosto musiikkia vai äänitehoste. Jos latausta tehdessä tapahtuu virhe, luodaan sovellukseen poikkeus.

Äänitiedostoa soitetaan ensimmäisellä vapaassa käytössä olevalla kanavalla. Soittofunktio tunnistaa soitettavan äänitiedoston tyypin, ja käyttää sen soittamiseen SDL:n tarjoamaa, juuri siihen käyttöön tarkoitettua soittofunktiota. Äänien soittofunktioon voidaan antaa myös haluttu toistomäärä, jolloin kyseistä äänitiedostoa soitetaan halutun monta kertaa peräkkäin, oletuksena tämä arvo soittaa ääniefektejä ikuisesti, ja musiikkityyppiä vain kerran.

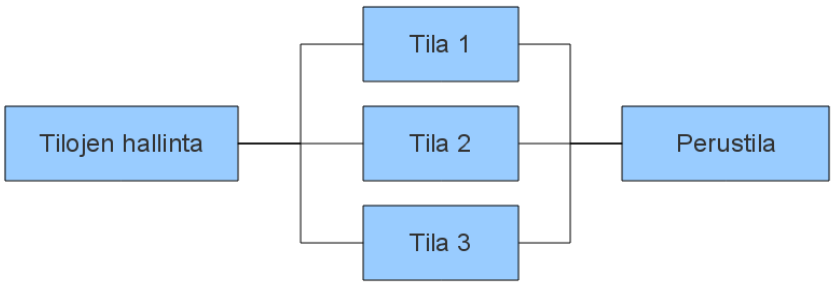
Musiikkityyppisten äänitiedostojen ollessa kooltaan suuria, ei ole haluttavaa ladata useaa tiedostoa samanaikaisesti muistiin. Tämän vuoksi musiikkitiedostoja pyritään pitämään muistissa vain yhtä kerrallaan, joten ääni-luokkaan on toteutettuna funktio jonka avulla musiikkitiedosto poistetaan muistista.

3.16 Tilat

Tilat ovat sovelluksen eri vaiheita, joita ylläpitää tilojenhallinta-luokka. Tilat selventävät sovelluksen toimintaa sekä erottavat sovelluksen eri osa-alueet toisistaan. Esimerkiksi sovelluksessa, jossa on valikko ja pelitila, nämä tilat ovat toisistaan riippumattomia kokonaisuuksia, joten ne tulisi erottaa toisistaan.

Tilojenhallinta-luokka pitää muistissaan sen hetkistä käytössä olevaa tilaa, ja suorittaa sen päivitys- ja piirtofunktion. Tilojenhallinta-luokka myös suorittaa itse tilan vaihdon, eli se poistaa muistista käytössä olevan tilan ja lataa muistiinsa uuden käytettävän tilan.

Kaikki pelimoottorissa käytettävät eri tilat pohjautuvat pelimoottorissa määritettyyn perustilaan. Perustila on tila, joka sisältää perusfunktiot, joita tilojenhallinta-luokka hallinnoi. Pelimoottorissa perustila sisältää piirto- ja päivitysfunktion, joita sovelluksen tilat käyttävät omien funktioiden suorittamiseen (Kuva 17).



Kuva 17. Kaavio eri tiloista, ja niiden yhteyksistä toisiinsa.

3.17 Teksti-luokka

Teksti luokkaa käytetään piirtämään ruudulle näkyvää tekstiä sille annetuista merkeistä. Teksti-luokka voi käyttää useaa eri fonttia, joita ladataan kuvatiedostoista. Pelimoottorissa ei käytetä tekstin piirtämiseen TrueType -fontteja, koska kuvatiedostosta ladattu fontti mahdollistaa laajemman käyttöjärjestelmätuen ja sen piirtäminen on sovelluksen kannalta kevyempää.

Merkit ladataan pelimoottoriin erillisestä kuvatiedostosta, ja luetaan kirjasinlistaan. Kuvatiedostossa olevien merkkien tulisi olla valkoisia, jotta tekstille voidaan antaa haluttu väri tulostusvaiheessa. Kirjasinlistasta jokaiselle merkille löytyy sitä vastaava kuva eli ruudulla näkyvä kirjain. Eri listoja eli fontteja voidaan ladata muistiin useita samaan aikaan.

Teksti tulostetaan käyttämällä funktiota jolle annetaan haluttu sijainti, tekstin väri, tulostettava teksti ja fontti. Tekstin tulostus alkaa annetuista koordinaa-

teista, johon ensimmäinen merkki tulostuu. Merkit haetaan kirjasinlistasta eli annetusta fontista, jonka jälkeen ne piirretään vierekkäin annetun tekstin mukaisesti. Tulostus funktiolle annetaan väriarvot punaiselle, vihreälle ja siniselle. Teksti tulostetaan annettujen osavärien mukaisesti.

Tekstin tulostuksen perustuessa kuvien piirtoon, myös kirjasinlistat joudutaan luomaan uudelleen sovelluksen kuvakoon muuttuessa. Tällöin vältetään kuvien tulostuksen tavoin mahdollisesti aiheutuvista piirto-ongelmista. Edellä mainittu on toteutettu erillisellä funktiolla joka suorittaa uudelleenlatauksessa tarvittavat toiminnot.

3.18 Ikkuna-luokka

Ikkuna-luokka on yksi pelimoottorin tärkeimmistä luokista, sillä se sisältää sovellusikkunan luomisen ja sen käsittelyn. Muita ikkuna-luokan sisältämiä tärkeitä toimintoja ovat skaalausfunktiot, joita käytetään ikkunakoon vaihtuessa skaalaamaan ruudulla näkyvä materiaali sopivaan kokoon.

Sovellusikkunan alustus on toteutettu käyttäen SDL:n tarjoamia funktioita, koska se tarjoaa alustariippumattoman sovellusikkunan luomisen. Ikkuna alustetaan käyttämään OpenGL-grafiikkakirjastoa sekä näytönohjaimella suoritettava 3D-kiihdytys otetaan käyttöön, jos se on mahdollinen kyseisellä tietokone kokoonpanolla.

Alustus jatkuu määrittämällä ikkunan koko ja kameran perspektiivi 3D-maailmassa. Piirrossa käytetään tuplapuskurointia, jossa kuvan piirto toteutetaan kahdella eri piirtopinnalla. Toista piirtotasoa näytettäessä, taustalla valmistellaan piirtoa varten seuraavaa piirtotasoa. Taustalla olevan piirtotason ollessa valmis, piirtotasot vaihdetaan keskenään, jolloin taustalla valmisteltu kuva näytetään. Tämä estää ruudun välkkymisen sovelluksen suorituksen aikana. Aina ikkunakoon muuttuessa, alustus suoritetaan uudelleen.

Ikkuna-luokka sisältää funktiot ikkunan tyhjentämiseen, ikkunan otsikkotekstin vaihtamiseen, ruudunpäivitysnopeuden näyttämiseen, ikkunakoon hakemiseen sekä kokonäyttötilan vaihtamiseen. Funktiot ovat staattisia, eli niitä voidaan käyttää sovelluksessa missä tahansa.

3.19 Sekalaiset

Pelimoottorissa on myös sekalaisia toimintoja, jotka eivät varsinaisesti sisälly mihinkään edellä mainittuihin osa-alueisiin. Nämä toiminnot ovat koottuna erilliseen sekalaiset-kansioon, josta niitä käytetään mahdollisesti sovelluksien tekemiseen.

Kenties käytetyin toiminto on jaettu osoitin, joka pitää muistissaan C++:n käyttämän olion osoittimen. Osoitin pidetään muistissa kunnes yksikään olio ei käytä kyseistä osoitinta sovelluksessa. Tämä toiminto helpottaa olioiden kanssa toimintaa ja vähentää sovellusten ohjelmointivirheriskejä.

Sekalaisiin toimintoihin sisältyy myös erilaisia muuntimia, joilla muunnetaan muun muassa numeroita tekstiksi sekä tekstiä numeroiksi. Näitä toimintoja käytetään tekemään tyyppimuunnoksia, joita tehdään useasti C++:n ollessa tarkka sille syötetyistä tyypeistä.

Yhteentörmäyksentunnistus on hyödyllinen toiminto, jota käytetään lähinnä peleissä tunnistamaan objektien väliset yhteentörmäykset. Jokaisella objektilla on oma ympyränmuotoinen törmäysalueensa, joka muodostuu objektin leveydestä ja korkeudesta. Mikäli yhteentörmäysalueet leikkaavat toisiaan, on yhteentörmäys tapahtunut.

Sekalaisiin toimintoihin sisältyy myös erilaisia tietovarastoja, joita käytetään esimerkiksi varastoimaan objektien sijaintia ja kokoa. Näiden käyttöä voidaan myös soveltaa kaiken tyyppisiin käyttötarkoituksiin sovelluksissa.

4 PULSE PHASE CHRONICLES

Pulse Phase Chronicles on reaaliaikastrategiapeli, jonka pelitapahtumat sijoittuvat futuristiseen avaruusaikaan. Opinnäytetyönä toteutettu PPC jatkaa koulussa tehdyn pelisarjan tarinaa. Tarinan tapahtumat jatkuvat pelityypilleen luontevasti hyvän ja pahan taisteluna, joka käydään erään planeetan pinnalla. Peliä pelatessa pelaaja johtaa joukkojaan tietokonepelaajaa vastaan rakentaen omaa tukikohtaansa ja hyökäten vihollisen kimppuun. Pelin tavoitteena on tuhota vihollisjoukot ja heidän tukikohtansa.

Pelityypiksi valikoitui reaaliaikastrategia, koska pelityyppi on tiedettävästi haastava toteuttaa sen koostuessa monesta eri osa-alueesta. Näiden osa-alueiden täytyy toimia yhdessä nopeasti ja luotettavasti pelin sujuvan toimivuuden takaamiseksi. Lisäksi peliä varten kehitettiin pelimoottori, jonka monipuolisuus mahdollistaa sen käytön useissa muissakin pelityypeissä ja sovelluksissa.

Peli on pyritty pitämään mahdollisimman kevyenä ja laitteistosta riippumattomana, jotta se toimisi useimmissa eri laitteistokokoonpanoissa. Graafisesta ilmeestä on jätetty kokonaisuudessaan tekstuurit pois, jotta saavutettaisiin mahdollisimman hyvin toimiva kokonaisuus.

Vastustajien toteutukseen oli kaksi varteenotettavaa vaihtoehtoa, tekoälyn ohjaamat tietokonepelaajat tai verkkopeli, jossa vastustajina toimivat muut verkon pelaajat. Peliin toteutettiin tekoälyn ohjaamat tietokonepelaajat, koska silloin peliä voi pelata, koska tahansa joutumatta odottamaan toisia pelaajia liittymään peliin.

4.1 Pelitilat

Pelitilat hyödyntävät pelimoottorin tilat-ominaisuutta, jolla peleihin saadaan rakennettu erillisiä kokonaisuuksia helpottamaan sen toteutusta. PPC:ssä on useita eri tiloja, jotka ovat tekijätila, pelin aloitustila, pelitila, valikkotila, pistetila, voittotila ja häviötila.

4.1.1 Tekijätila

Tekijätila on tila, joka listaa pelinkehityksessä mukana olleet tekijät ja projektissa auttaneet henkilöt. Tekijätilaan siirrytään voittaessa pelin viimeisen kentän tai halutessaan siihen voi siirtyä myös pelin päävalikosta. Tekijätilassa tekijöiden nimet vierivät ruudun alalaidasta kohti ylälaitaa. Tilasta poistutaan painamalla Esc-näppäintä tai kun kaikki nimet on näytetty.

4.1.2 Pelin aloitustila

Pelin aloitustila näytetään sovelluksen käynnistyksen yhteydessä ennen varsinaiseen päävalikkoon siirtymistä. Aloitustilasta ei voi siirtyä pois painamalla mitään näppäimiä, vaan tilasta siirrytään päävalikkoon, kun aloitustilassa suoritettavat toiminnot on näytetty. PPC:ssä aloitustilassa näytetään pelintekijöiden logo, joka häivytetään ruudulle ja takaisin mustaan.

4.1.3 Pelitila

Pelitilassa suoritetaan kaikki pelitoiminta. Siihen voidaan siirtyä päävalikosta painamalla Start game -nappia, jolloin peli alkaa ensimmäisestä kentästä tai painamalla Load game -nappia mikäli mahdollista, jolloin peli jatkuu aiemmin tallennetusta kentästä. Pelitilaan voidaan siirtyä myös TotalWar-valikosta, jossa suoritetaan halutut valinnat aloitettavan kentän ja vihollismäärän suhteen. Pelitilasta voidaan poistua painamalla Esc-näppäintä sekä voittamalla tai häviämällä kyseinen taistelu.

4.1.4 Valikkotila

Valikkotila sisältää sen hetkisen valitun valikon ja kontrolloi valikoiden toimintaa. Valikkotilasta ei poistuta siirryttäessä valikosta toiseen, vaan valittu valikko vaihtuu. Tilaan voidaan asettaa halutessa taustamusiikki, joka aloittaa musiikin soittamisen, kun tilaan saavutaan, ja lopettaa soittamisen tilasta poistuttaessa. Valikkotilasta myös nähdään sen hetkinen pelin versio.

4.1.5 Pistetila

Pistetilaa käytetään esittämään pelin maksimipisteet ja niiden tehnyt pelaaja. Pistetaulukko ladataan erillisestä tiedostosta ja tulostetaan ruudulle paremmuusjärjestyksessä. Pistetilaan ei siirrytä automaattisesti koskaan, vaan sinne saavutaan aina päävalikon kautta. Poistuminen pistetilasta tapahtuu Esc- tai Enter-näppäimellä sekä klikkaamalla Continue-nappia hiirellä.

4.1.6 Voittotila

Voittotilaan saavutaan voitetun taistelun jälkeen. Voittotila sisältää erilaista tietoa voitetusta pelistä, kuten pelatun ajan, rakennetut joukot, tuhotut joukot, tuhoutuneet joukot, aiheutettu tuho ja aiheutunut tuho. Tilassa ilmaistaan myös, että käyty taistelu oli voitokas. Voittotilasta poistutaan painamalla Esc- tai Enter-näppäintä ja klikkaamalla Continue-napista hiirellä. Poistumisen yhteydessä voittotila tallentaa taistelusta saadut pisteet sekä seuraavaksi pelattavan kentän pelin latausta varten.

4.1.7 Häviötila

Häviötila sisältää paljon samoja toimintoja kuin voittotila. Tilaan saavutaan hävityn taistelun jälkeen, ja se myös ilmaisee tietoja käydystä taistelusta. Häviötila tallentaa saadut pisteet, mutta ei pelattavaa kenttää. Tila kertoo käydyn taistelun olleen tappiollinen, ja tilasta poistutaan samoin tavoin kuin voittotilastakin.

4.2 Menu

Pelin valikkokokonaisuutta kutsutaan menuksi. Menussa käytettävät valikot on rakennettu käyttäen pelimoottorin tarjoamia elementtejä, jotka mahdollistavat yhtenäisen valikkorakenteen toteuttamisen. Pelissä oleva menu koostuu seuraavista valikoista päävalikko, asetukset, grafiikka-asetukset, ääniasetukset ja TotalWar.

4.2.1 Päävalikko

Päävalikko toimii koko menun keskipisteenä (Kuva 18). Päävalikosta siirrytään eri tiloihin sekä asetukset- ja TotalWar-valikoihin. Valikko sisältää painikkeet pelin aloittamiseen, piste- ja tekijätilaan siirtymiseen sekä pelistä poistumiseen. Päävalikossa on myös syöttökenttä, jota käytetään vastaanottamaan pelaajan nimi pisteiden kirjaamista varten. Päävalikosta voidaan sulkea peli myös painamalla Esc-näppäintä.

Kuva 18. PPC:n päävalikko, josta ilmenee pelissä käytettyjen valikoiden tyyli.

4.2.2 Asetukset

Asetukset-valikko sisältää painikkeet grafiikka- ja ääniasetusvalikoihin siirtymiseen. Asetukset-valikkoon saavutaan päävalikosta, ja se sisältää myös painikkeen, jolla sinne voidaan palata takaisin. Tämä voidaan suorittaa myös painamalla Esc-näppäintä.

4.2.3 Grafiikka-asetukset

Grafiikka-asetukset-valikko sisältää valintoja näytön tarkkuuden ja kokoruututilan asettamiseen. Näytön tarkkuuden valinta tapahtuu pudotusvalikosta, joka sisältää käytettävissä olevat tarkkuusvaihtoehdot. Valitut asetukset tallennetaan muistiin, ja niiden mukaan suoritetaan muiden pelissä käytettävien grafiikoiden skaalaus. Asetukset tallennetaan valikosta poistuessa, joka tapahtuu klikkaamalla Back-painiketta hiirellä tai painamalla Esc-näppäintä. Valikosta poistutaan takaisin asetukset-valikkoon.

4.2.4 Ääniasetukset

Ääniasetukset-valikkoon saavutaan grafiikka-asetusten tavoin asetukset-valikosta. Valikko sisältää vierityspalkkielementit pelin ääniefektien- ja musiikinvoimakkuuden asettamiseen. Asetukset tallennetaan valikosta poistuessa, joka tapahtuu myös grafiikka-asetusten tavoin Back-painikkeella tai Esc-näppäintä. Myös asetetut arvot vaihdetaan valikosta poistuttaessa, jolloin uudet ääniasetukset astuvat voimaan välittömästi. Valikosta poistutaan takaisin asetukset-valikkoon.

4.2.5 TotalWar

TotalWar-valikko sisältää TotalWar-pelimoodin alustukseen käytettävät valinnat. Valinnat koostuvat kahdesta pudotusvalikosta, joita käytetään valitsemaan haluttu palattava kenttä ja vihollispelaajien määrän. Valikkoon saavutaan päävalikosta, johon voidaan palata Back-painikkeella tai Esc-näppäintä painamalla. Pelin aloitus kyseisellä pelimoodilla ja asetetuilla valinnoilla tapahtuu painamalla Start game painiketta.

4.3 Hud

Hud näyttää tietoa pelimaailman tapahtumista, joita pelaaja voi hyödyntää pelinkulun aikana. Hudin näyttämä tieto on riippuvainen pelin tapahtumista, eli sen näyttämä tieto vaihtuu eri tapahtumien myötä.

4.3.1 Päätietoruutu

Päätietoruutu sijaitsee ruudun ylälaidassa. PPC:ssä päätietoruutu on jaettu kolmeen osaan: pelin logoon, tietoalueeseen ja mittarialueeseen. Logoaluetta käytetään parantamaan hudin visuaalista ilmettä. Tietoalue näyttää valituista pelaajan objekteista riippuen erilaista tietoa. Mittarialueella sijaitsevat energia- ja tasapainomittarit, jotka ilmaisevat pelaajan sähköntuotannon tilannetta ja pelaajan joukkojen vahvuutta vihollispelaajiin verrattuna.

Reaaliaikastrategiapeleissä tietoalue on hudin tärkein ja hyödyllisin osa-alue sen tarjoaman kriittisen tiedon vuoksi. Pelissä yhden objektin ollessa valittuna tietoalueella näytetään kyseisen objektin nimi ja kunto. Lisäksi Rakennus-tyyppisillä objekteilla näytetään myynti-painike. Joukko-tyyppisillä objekteilla näytetään tulivoima-arvo, vapaa-ajomoodi-painike ja mittari, joka osoittaa ampumisen latausaikaa.

Kun objekteja on valittuna kahdesta kymmeneen kappaletta, tietoalueella näytetään valittujen joukko-tyyppisten objektien yhteenlaskettu kunto- ja tulivoima-arvo. Näiden tietojen lisäksi jokaisesta valitusta objektista tyypistä riippumatta näytetään kuva, jonka yläpuolella näkyy objektin kunto graafisessa muodossa. Objektin kuvaa klikkaamalla kyseinen objekti valitaan pelimaailmasta ja tietoruudussa näytetään vain kyseisen objektin tiedot.

Valittujen objektien määrän ollessa suurempi kuin kymmenen kappaletta, tietoalueella näytetään valittujen objektien määrä, joukko-tyyppisten objektien yhteenlaskettu kunto- ja tulivoima-arvo. Kunto- ja tulivoima-arvot näytetään vain, jos ne ovat suurempia kuin nolla.

Objektin ollessa valittuna se voi mahdollisesti sisältää erilaisia valintoja. Nämä valinnat näkyvät päätietoalueella kuvina, jotka esittävät suoritettavaa toimin-

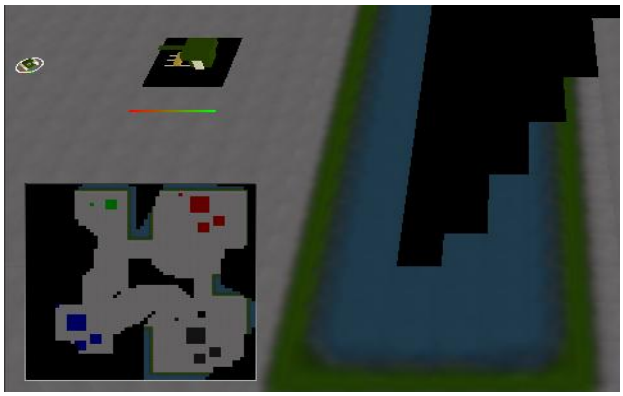
toa. Vietäessä hiiren kursori kuvien päälle näytetään toimintoa selittävä teksti. Seliteteksti koostuu yleisesti toiminnon kuvauksesta ja sen mahdollisesta arvosta.

Päätietoruudun vasemmassa alareunassa sijaitsee ilmaisin, joka kertoo pelaajan sen hetkisen varallisuuden ja maksimi krediittimäärän. Ilmaisin päivittyy välittömästi tapahtumien mukaan, jolloin pelaaja näkee aina oikean, sen hetkisen varallisuutensa.

Päätietoruudussa on myös toteutettu tietokonepelaajien häviöstä kertova infoteksti, joka ilmestyy hetkeksi näkyviin jonkin tietokonepelaajan tuhoutuessa kokonaan. Teksti piirretään kyseisen pelaajan värisenä ruudun oikeaan laitaan, ja siinä ilmaistaan tuhoutuneen pelaajan nimi sekä ilmoitetaan tämän tuhoutumisesta.

4.3.2 Minimap

Minimap on pienoismuodossa oleva kartta, johon piirretään joukkojen sen hetkinen sijainti ja pelattava kartta. Minimap-karttaa käytetään varsinkin strategiapeleissä hahmottamaan pelattavaa maailmaa ja koordinoimaan hyökkäyksiä minimapista näkyvien objektien sijaintien mukaan (Kuva 19).



Kuva 19. Esimerkki minimapin näkymästä.

Minimap ei ole pelissä oletuksena näkyvissä, mutta sen voi ottaa halutessaan käyttöön painamalla M-näppäintä. Kartassa näkyvät värit kuvaavat eri pelaajien joukkoja ja rakennuksia, joten eri pelaajille kuuluvat objektit erottaa toisistaan selvästi.

Minimap on toteutettu käyttämällä sen piirrossa vastaavaa toteutusta kuin maatason kartan piirrossa, mutta minimapin laatat piirretään pienempinä ja 2D-kuvina pelimaailman päälle osana hudia.

4.4 Soittolistat

Soittolistoja käytetään avustamaan pelissä käytettyjen äänitehoste- ja musiikkikokonaisuuksien yhteenliittymistä. Soittolistat koostuvat erilaisista äänitiedostoista, jotka lisätään listaan latausjärjestyksessä. Soittolistaa soitettaessa äänitiedostoja käydään läpi listassa olevan järjestyksen mukaan. Soittolistassa olevia äänitiedostoja voidaan soittaa myös satunnaisessa järjestyksessä asettamalla soittokäskyssä tämän mahdollistava arvo aktiiviseksi.

Pelissä käytetään kahteen eri tarkoitukseen soittolistoja: taustamusiikin soittamiseen sekä puheen toistamiseen. Taustamusiikkilistaa käytetään pelin aikana jatkuvasti soivan musiikin toistamiseen. Musiikkikappaleita soitetaan aina satunnaisessa järjestyksessä tiettyjen kappaleiden toiston välttämiseksi.

Joukon valinnassa käytettävä soittolista kasataan samoin tavoin kuin musiikkisoittolista. Se sisältää huudahduksia, joita käytetään ilmaisemaan joukkotyyppisen objektin valintaa. Soittolistasta toistetaan satunnaisesti valittu ääni aina, kun objekti valitaan, ja mitään valintaääntä ei soiteta samaan aikaan.

Objektin liikkeelle lähdössä käytetään myös soittolistaa toistamaan puheääniä. Liikkeelle lähdettäessä ääniä käytetään myös ilmaisemaan pelaajalle, että hänen suorittamansa hiirenpainallus on huomioitu. Ilman merkkiääntä pelaaja saattaa luulla, että painallusta ei ole huomioitu, jolloin lisäpainallukset kuormittaisivat turhaan pelissä käytettävää reitinhakua. Liikkeelle lähdettäessä soitettava ääni valitaan listalta satunnaisesti.

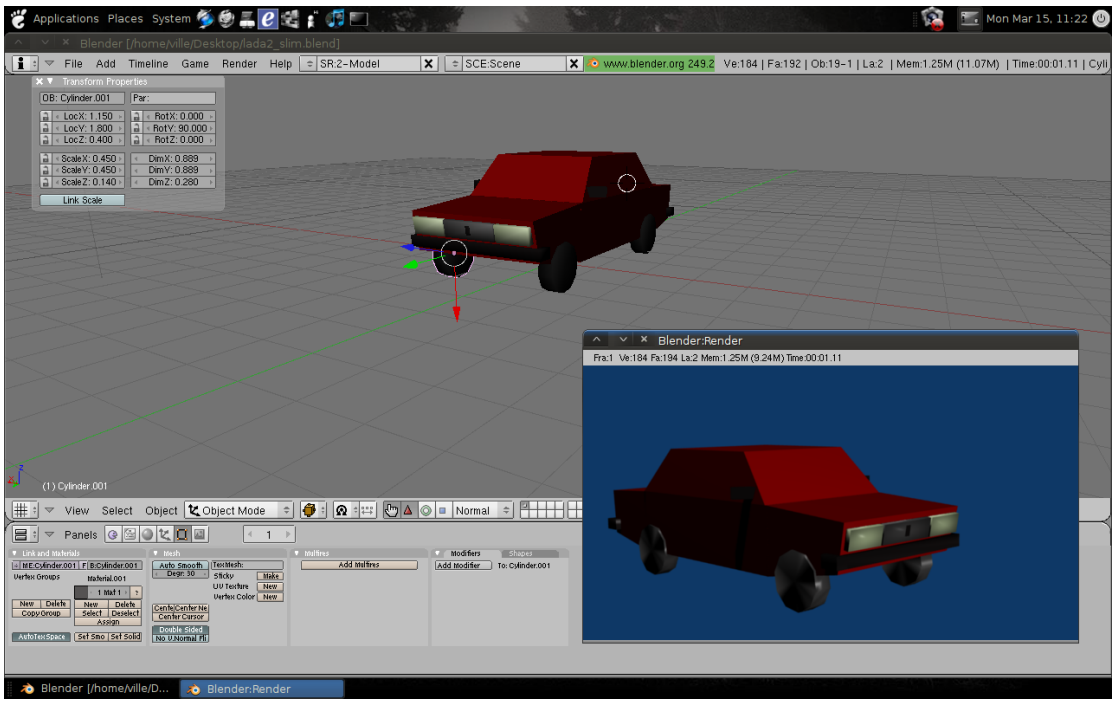
4.5 Objektit

Pelissä käytettävät objektit ovat ajoneuvoja tai rakennuksia. Objekteja käytetään muodostamaan pelin tapahtumat. Ne sisältävät tekoälyä, eli ne voivat suorittaa erilaisia toimintoja itsenäisesti ilman pelaajan käskyjä. Objektit mukautuvat sen hetkiseen pelitilanteeseen ja toimivat tilanteen vaatimalla tavalla.

Myös pelaajat ovat eräänlaisia objekteja, jotka pitävät listaa pelaajan hallitsemista rakennuksista ja joukoista. Pelaaja ei kuitenkaan voi sisältää toisia pelaajia. Pelaajia on olemassa kahdentyyppisiä: ihmisen ohjaamia pelaajia ja tekoälyn ohjaamia tietokonepelaajia.

4.5.1 Mallinnus

Objektien näkyvää osaa kutsutaan malliksi. Mallit on toteutettu peliin Blender 3D -mallinnusohjelmalla (Kuva 20). PPC:n objektien mallit on tehty yksinkertaisiksi, jotta niiden piirtäminen pelimaailmassa olisi kevyempää, jolloin malleja voidaan piirtää pelimaailmaan suurempi määrä samanaikaisesti.



Kuva 20. Mallit peliin on luotu käyttämällä Blender-mallinnusohjelmaa.

Mallit koostuvat yksinkertaisista muodoista, kuten kolmioista ja neliöistä. Näiden pisteet on sijoitettu 3D-maailmaan, jolloin kolmiot ja neliöt muodostavat yhtenäisen kokonaisuuden eli 3D-mallin. 3D-mallin luominen on aikaa vievä prosessi, koska objekteissa on useita eri muotoja, jotka täytyy saada toteutettua vain kolmioita ja neliöitä käyttäen.

Jotta tehtyjä malleja voitaisiin käyttää pelissä, ne täytyy muuntaa pelimoottorin tukemaan muotoon. Muunnosta varten peliin on kehitetty työkalu, joka lukee Blenderistä saadun tiedoston ja tallentaa sen pelimoottorin käyttämään muotoon.

Mallit ovat tärkeä osa pelikokonaisuutta, sillä mallit ovat objektien näkyvä osa ja ne muodostavat suuren osan pelin graafista ilmettä. Kuitenkin pelien mallien teossa tulee ottaa huomioon pelille asetetut teho vaatimukset. Tarpeettoman monimutkaiset 3D-mallit voivat hidastaa pelin toimintaa liiallisen paljon.

4.5.2 Pelaajat

Pelaajat on jaoteltu kahteen eri päätyyppiin: tietokonepelaajiin ja ihmispelaajiin. Molemmilla tyypeillä on samat perustoiminnot, mutta tietokonepelaaja suorittaa toimintonsa itsenäisesti. Tietokonepelaaja myös jatkuvasti tutkailee pelitapahtumia ja reagoi pelitapahtumien muutoksiin.

Pelaaja-objekti sisältää erilaisia toimintoja ja pitää tallessa pelaajalle tärkeitä tietoja, kuten krediitti- ja energiatiedot. Krediittitiedot sisältävät pelaajan sen hetkisen krediittimäärän sekä maksimi krediittimäärän, jonka pelaaja voi saavuttaa. Maksimikrediittimäärää voidaan kasvattaa rakentamalla siihen tarkoitettu rakennus. Energiatiedot puolestaan pitävät sisällään pelaajan energiankulutuksen, energiantuotannon ja näiden välisen suhteen. Ihmispelaajan energiasuhde näytetään pelin aikana hudin alueella.

Ihmispelaajia voi olla PPC:ssä kerrallaan vain yksi, mutta pelimoottori pystyy käsittelemään myös useampia ihmispelaajia. Ajomoodi on toteutettu vain ihmispelaajalle. Ajomoodin toimintaa voidaan muokata, sekä toiminnon tila voidaan selvittää objektissa toteutetuilla funktioilla. Näin ollen tietokonepelaaja ei voi hyödyntää ajomoodia.

Tietokonepelaaja rakentaa ja ohjaa omia joukkojaan itsenäisesti sekä lähettää joukkojaan hyökkäyksiin ihmispelaajan tukikohtaan. Tietokonepelaaja aloittaa pelaamisen rakentamalla tukikohtaansa. Tukikohdan rakentamisen tietokonepelaaja aloittaa barrack-rakennuksen rakentamisella.

Rakentamisessa pelaaja etsii rakennuksen kokoista vapaata paikkaa pelimaailmasta, johon rakentamisen voi aloittaa (Liite 1). Seuraava vaihe tukikohdan rakentamisessa on rakentaa factory-rakennus, joka turvaa tietokonepelaajan krediittien tuotannon. Tämän saavutettuaan pelaaja panostaa energiantuotantoon ja rakentaa tarvittavan määrän power-rakennuksia, jotta energiataso saavuttaa asetetun rajan. Tukikohdan ollessa lähes valmis, tietokonepelaaja suojaa tukikohtaansa rakentamalla pillbox-rakennuksia, jotka suojaavat tukikohtaa vihollisten ja ihmispelaajan hyökkäyksiltä.

Tietokonepelaajan rakennettua barrack-rakennuksen pelaaja alkaa tuottaa joukkoja. Joukkojen tuottamisessa otetaan huomioon pelaajan sen hetkinen krediittimäärä sekä viimeksi rakennetun joukon valmistumisaika. Tietokonepelaaja valmistaa vain tankkeja ja pyrkii tuottamaan niitä kappalemäärältään tasaisesti.

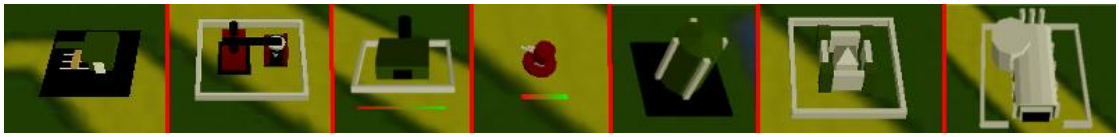
Tietokonepelaaja lähettää joukkojaan hyökkäyksiin ihmispelaajaa vastaan. Pelaaja valitsee hyökkäävät joukot listaltaan, kunnes asetettu hyökkäysjoukon koko täyttyy. Uusi hyökkäys suoritetaan satunnaisesti tietyn ajan jälkeen, jolloin tietokonepelaajien toiminta ei ole kiinteästi määriteltyä.

Tietokonepelaajia avustetaan pelissä antamalla niille ylimääräisiä krediittejä, jotta pelin vaikeusaste olisi hieman korkeampi. Myös pelin alkaessa tietokonepalaajilla on ihmispelaajaa enemmän krediittejä. Pelin aikana pelaaja saa kymmenen sekunnin välein ylimääräisiä krediittejä riippuen kentän pelatusta ajasta.

Pelaajia PPC:ssä voi olla vain neljä ihmispelaajan lisäksi. Pelaajien joukot ja rakennukset erottavat toisistaan kullekin pelaajalle asetetusta väristä. Värejä ovat vihreä, punainen, sininen, keltainen ja harmaa. Näistä väreistä ihmispelaaja on aina vihreän värinen.

4.5.3 Rakennukset

PPC:ssä on useita rakennuksia, joita käytetään erilaisiin tarkoituksiin (Kuva 21). Eri rakennuksilla on pelissä pituudeltaan erilaiset rakennusajat, ja niiden hinnat poikkeavat toisistaan. Eri rakennukset ovat myös toisistaan poikkeavan kokoisia, mutta rakennuksen koko ei välttämättä vaikuta rakennuksen kestävyyteen tai hintaan. Äkillisen rahantarpeen iskiessä kaikki rakennukset voi myös myydä, joista maksetaan rakennuksen tyypin ja kunnon mukainen hinta. Rakennuksien rakentamista on pelissä rajoitettu niin, että rakennuksien tulee olla tukikohdan läheisyydessä, ja niitä ei voida rakentaa veteen tai joukkojen päälle.



Kuva 21. Pelissä esiintyvät rakennukset: päärakennus, power, factory, pillbox, siilo, armory ja barrack.

Rakennuksen rakentuessa sen ympärille nousee niin sanottu rakennus suoja, joka ilmaisee rakennuksen valmistumistilan. Suojan laskeutuessa takaisin alas rakennuksen valmistus on valmis, ja rakennuksen tuomat toiminnot ovat käytettävissä. Nämä toiminnot poikkeavat eri rakennuksilla toisistaan, mutta kaikki rakennukset tuovat pelaajalle hyödyllisiä ominaisuuksia. Valitun rakennuksen mahdolliset toiminnot tulevat näkyviin hudin päätietoruutuun, josta halutut toiminnot voidaan suorittaa.

Pelin alkaessa ihmispelaajalla on ainakin yksi rakennus valmiiksi rakennettuna. Tämä päärakennus toimii tukikohdan arvokkaimpana rakennuksena, jota pyritään suojaamaan taistelun aikana. Päärakennusta ei voi myöskään ostaa mistään. Päärakennuksen avulla pelaajalla on mahdollisuus rakentaa muita rakennuksia, jos päärakennus tuhoutuu taistelun aikana, ei muita rakennuksia ole mahdollista enää rakentaa.

Armory-rakennuksessa voidaan parantaa pelimaailmassa olevien omien joukkojen tulivoimaa ja kuntoa. Jokainen toiminto kuitenkin maksaa usean pienemmän joukon rakentamisen verran, joten toimintojen käyttöä tulee harkita tarkoin. Toiminnot kasvattavat haluttua arvoa joukon sen hetkisen tulivoiman tai kunnon mukaan.

Barrack-rakennus on rakennuksista suurin. Barrack-rakennuksen ollessa valmis pelaaja voi rakentaa kaikkia rakennettavissa olevia joukkoja. Eri joukoilla on toisistaan eroavat rakennusajat ja hinnat. Pelaajan rakentaessa useita joukkoja samanaikaisesti, rakennettavat joukot lisätään jonotuslistaan, josta ne valmistuvat sinne lisätyssä järjestyksessä. Jonon pituuden ja sen hetkisen rakennustilanteen voi nähdä hudin päätietoruudusta rakennuksen ollessa valittuna. Joukon valmistuessa barrack-rakennus etsii pelimaailmasta vapaan paikan ja lähettää valmistuneen joukon kyseiseen vapaaseen ruutuun. Mikäli valmistunut joukko jää jostain syystä barrack-rakennuksen sisälle, se automaattisesti poistetaan ja pelaajalle palautetaan siitä maksettu krediittimäärä.

Siilo-rakennuksen käyttötarkoitus pelissä on varsin yksinkertainen. Sen tuoma varastointitila mahdollistaa suurempien krediittimäärien hallitsemisen. Jokainen siilo-rakennus lisää pelaajan maksimi krediittimäärää tuhannella, mutta siilon tuhoutuessa tai myytäessä maksimi krediittimäärä laskee saman verran.

Factory-rakennusta käytetään pelissä lisäämään pelaajan krediittituotantoa. Mitä useampi factory, sitä enemmän krediittejä pelaaja saa. Krediittien tuotantoon vaikuttaa sen hetkinen energiataso. Mikäli pelaajan energiataso on alhainen, myös krediittien tuotanto hidastuu.

Pillbox-rakennus toimii tukikohtaa suojaavana rakennuksena. Se tunnistaa vihollisjoukot ja rupeaa ampumaan heitä kohti, kun vihollisjoukot saapuvat pillboxin ampumaetäisyydelle. Pillbox voi kääntyä 360 astetta, joten sillä on erittäin laaja ampuma-alue ja soveltuu erittäin hyvin vartioimaan tukikohtaa.

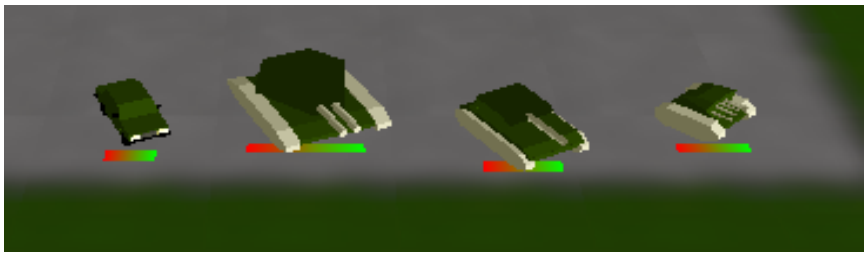
Power-rakennus on rakennus joka tuottaa kaiken tukikohdan tarvitseman energian. Yleisesti power-rakennuksia tulee olla useita, jotta vaadittava energiantarve täytetään. Power-rakennuksen luoma energia vaikuttaa usean eri rakennuksen toimintoihin, kuten rakennusten valmistumisaikaan, joukkojen valmistumisaikaan, sekä krediittien tuotantoon. Pelissä sen hetkinen energiataso näkyy hudin alueella.

Taulukko 1. PPC:n eri rakennusten ominaisuuksien yhteenveto.

Selite	Päärakennus	Barrack	Siilo	Factory	Power	Pillbox	Armory
Kunto	10000	1200	1200	1200	1200	500	1200
Energian kulutus	50	25	10	30	0	12	15
Rakennusaika	-	15s	12s	15s	10s	8s	20s
Hinta	-	2500cr	1500cr	2000cr	750cr	500cr	3500cr

4.5.4 Joukot

Pelissä esiintyy neljän tyyppisiä joukkoja: pikkutankkeja, normaalikokoisia tankkeja, suurikokoisia tankkeja, sekä lada-henkilöautoja (Kuva 22). Jokaisella joukolla on toisistaan poikkeavat ominaisuudet, kuten nopeus, tulivoima, kunto, valmistumisaika ja hinta. Jokainen joukko on suunniteltu niin, että niiden hinta ja tehokkuus ovat tasapainossa, sekä jokaiselle joukolla on omat hyvät ja huonot puolensa taistelussa.



Kuva 22. Pelissä esiintyvät joukot: lada-henkilöauto, suurikokoinen tankki, normaalikokoinen tankki ja pienikokoinen tankki.

Minitank on pikkutankki, joka on joukoista halvin ostaa. Pienellä hinnalla pelaaja saa käyttöönsä nopean, mutta kunnoltaan heikon tankin. Tankin ominaisuuksiin kuuluu lisäksi, että se lataa ammuksensa nopeasti, mutta niiden teho on pieni. Myös tankin ampumaetäisyys on pienempi kuin muilla joukoilla. Minitank on joukoista nopein rakentaa, jolloin niistä saadaan kasattua nopeasti suurikin armeija.

Normaalikokoinen tankki on pikkutankkia hieman kalliimpi, mutta tulivoimaltaan tehokkaampi. Tankin rakennus- ja latausajat ovat myös pikkutankkia suurempia sekä se liikkuu pikkutankkia hitaammin pelimaailmassa. Toisaalta normaalikokoisella tankilla on suurempi ampumaetäisyys, sekä se on pikkutankkia kestävämpi taisteluissa.

Suurikokoinen tankki on PPC:n tehokkain, mutta kallein tankki. Sen tulivoima on tankeista selvästi suurin ja se kestää tulitusta parhaiten kaikista joukoista. Huonoina puolina suurikokoisilla tankeilla on niiden hitaus, pitkät latausajat, sekä kauan kestävä rakennus. Suurikokoinen tankki kuitenkin loistaa muiden tankkien joukosta sen suurella ampumaetäisyydellä ja tulivoimallaan.

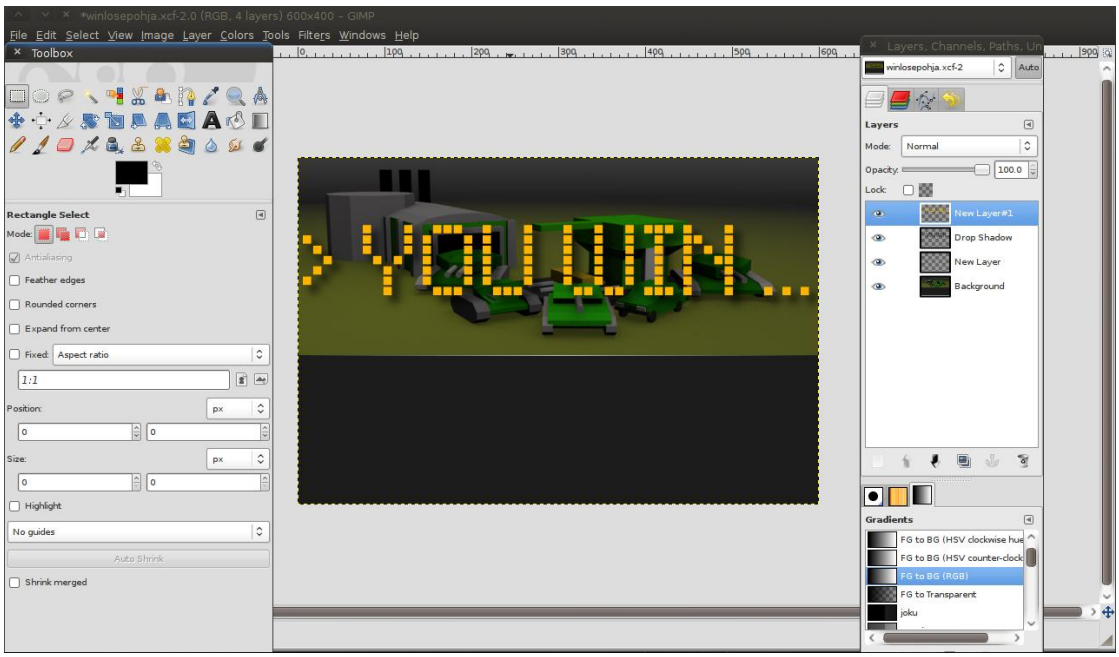
Ladat ovat PPC:n erikoisase. Ladat eivät ammu mihinkään, vaan niitä käytetään ajamalla ne käsiajomoodilla vihollisten rakennuksiin tai joukkoihin päin. Tämä aiheuttaa laajan räjähdyksen, jonka seurauksena on suuret tuhot räjähdyksen lähimaastossa. Ladan heikkoutena on sen huono kesto tulituksen alla, joten se hajoaa erittäin helposti. Lada on myös kaikista joukoista kallein ja hitain rakentaa.

Taulukko 2. PPC:n eri joukkojen ominaisuuksien yhteenveto.

Selite	Pikkutankki	Normaalikokoinen tankki	Suurikokoinen tankki	Lada-henkilöauto
Kunto	100	200	500	50
Tulivoima	2	17	50	0
Nopeus	70	40	20	100
Latausaika	0,2s	1s	2,5s	0
Ampumaetäisyys	3	4	6	0
Rakentumisaika	2s	4s	6s	10s
Hinta	100cr	250cr	500cr	3000cr

4.6 Peligrafiikka

Peligrafiikoiden tekoon pelissä on käytetty Gimp -kuvankäsittelyohjelmaa ja Blender -mallinnusohjelmaa. PPC:n grafiikat on pidetty yksinkertaisina ja siisteinä pelin yleistä linjaa noudattaen. Kuvankäsittelyyn pelin toteutuksessa valittiin Gimp, koska se on monipuolinen ja vapaassa levityksessä sekä tekijöille entuudestaan tuttu (Kuva 23). Gimp -kuvankäsittelyohjelma toimii myös usealla eri käyttöjärjestelmällä ja tukee useita eri kuvaformaatteja.



Kuva 23. Kuvankäsittelyyn käytetyllä Gimp-ohjelmalla on luotu suuri osa PPC:n grafiikoista.

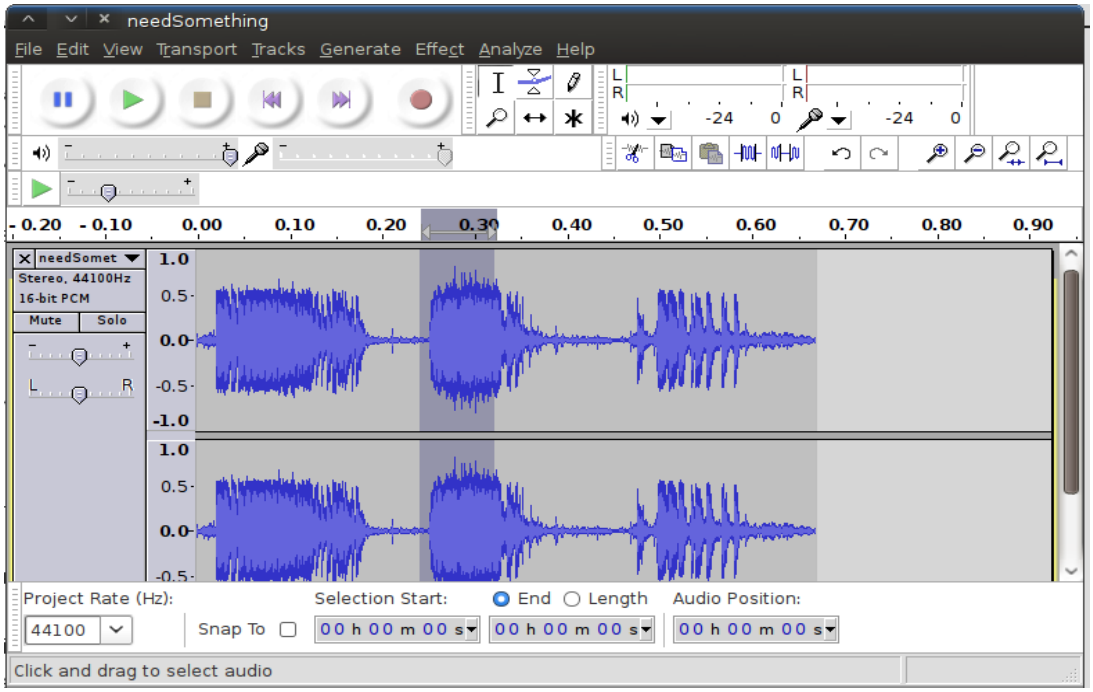
PPC:n grafiikoista on yritetty tehdä myös mahdollisimman kevyet, joten kuvaformaatiksi valikoitui PNG eli häviötön bittikarttagrafiikan tallennusformaatti, joka sisältää myös alfa-kanavan. PNG:n valintaan on vaikuttanut myös sen suuri väriavaruus ja sen vapaa käyttö.

PPC pelin yleistä linjaa noudattaen kaikki pelissä esiintyvät grafiikat on toteutettu täysin tekijöiden toimesta. Grafiikoiden teossa on käytetty molempia edellä mainittuja ohjelmia. Grafiikat ovat ensin 3D -mallinnusohjelmalla luotu 2D-muotoon, joka on tämän jälkeen avattu Gimp ohjelmaan jossa on suoritettu kuvan jatkokäsittely ja viimeistely. Kaikki grafiikat on tallennettu tämän jälkeen PNG-muotoon pelin kuvakansioon. Rajoituksena pelissä käytetyillä kuvilla on, että kaikkien kuvien koko tulee olla kahdella jaollinen.

Grafiikoista monia kuvia ei ole tehty pelkästään käyttämällä Gimp -kuvankäsittelyohjelmaa. Yksi poikkeus on pelissä käytetty tilemap-tiedosto, joka sisältää kaikkien karttojen teossa käytettyjen laattojen kuvat. Tilemap-kuvan teko on haastavaa, koska siinä tulee ottaa huomioon kaikkien eri laattojen mahdolliset liitokset toisiinsa. Tilemap-tiedosto sisältää elementtejä karttojen tekoon, kuten nurmea, hiekkaa, vettä ja asfalttia. Näitä yhdistelemällä voidaan tehdä laajoja karttakokoelmia.

4.7 Peliäänet

Peleissä käytetään ääniä luomaan tunnelmaa ja parantamaan pelikokemusta. PPC:n äänimaailma on pyritty pitämään yksinkertaisena, mutta toimivana luomaan mukava pelikokemus. Äänien editointiin on käytetty Audacity -äänenkäsittelyohjelmaa, joka on yksinkertainen käyttää, monipuolinen, ja vapaasti käytettävissä (Kuva 24). Audacity tarjoaa käyttäjälle monia hyödyllisiä ominaisuuksia järkevässä ja helppokäyttöisessä paketissa. Audacityn valintaan on vaikuttanut myös sen alustasta riippumattomuus ja sen laaja tuki useimmille ääniformaateille.



Kuva 24. PPC:n peliäänien muokkaamiseen käytettiin Audacity-äänieditoria.

PPC:n ääniformaatiksi valittiin Ogg Vorbis, joka on pakattu ääniformaatti jota käytetään esimerkiksi tietokonepelien ääniformaattina. Ogg Vorbis myös pienentää huomattavasti äänien ja musiikkien vaatimaa levytilaa, jolloin itse pelikin vie vähemmän tilaa tietokoneen kovalevyltä.

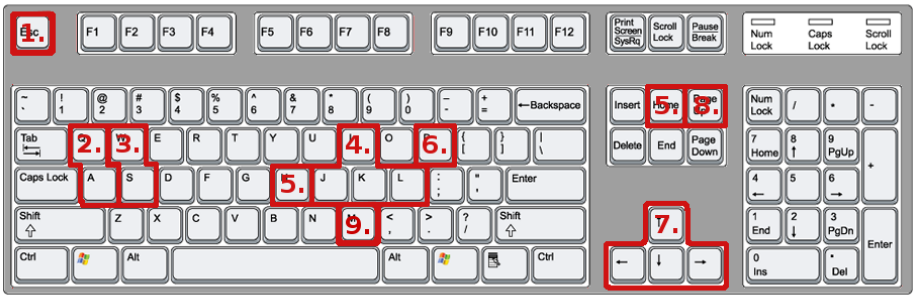
Pelissä kuultavat äänitehosteet ovat tehty pelin hengen mukaisesti itse nauhoittaen ja muokaten äänen editoimisohjelmaa käyttäen. Kaikki ajoneuvojen- ja ampumisäänet ovat nauhoitettu puolustusvoimien käyttämästä aidosta kalustosta sekä aidoista aseista. Nämä äänet ovat tämän jälkeen muokattu vastaamaan pelissä tarvittavaa käyttötarkoitusta.

Pelissä kuuluvat huudahdusäänet on nauhoitettu käyttäen mikrofonia ja muokattu kuulostamaan kuin puhe kuuluisi radiopuhelimesta. Muokkaamisessa on käytetty Audacityn tarjoamia toimintoja, joilla normaali puheääni saadaan kuulostamaan peliin sopivalta.

PPC:n musiikin ovat toteuttaneet Mika Liljeqvist ja Simo Andersen. Musiikin avulla pelin tunnelmasta saadaan elävämpi ja mielenkiintoisampi. Musiikki kappaleiden määrä on myös tasapainoisessa suhteessa pelin pituuden kanssa, jolloin pelin musiikkeihin ei pääsy kyllästymään.

4.8 Pelikontrollit

Pelissä käytetään erilaisia näppäimistökontrolleja ohjaamaan peliä ja siihen liittyviä toimintoja. Pelissä käytetään vain esiasetettuja näppäimiä, eikä niitä voi muuttaa (Kuva 25).



Kuva 25. Pelissä käytettävät kontrollit.

Painamalla Esc-näppäintä valikkotilassa, voidaan palata takaisin edellisiin valikoihin. Päävalikossa Esc-näppäin toimii sovelluksen sulkemispainikkeena, jolloin sovellus suljetaan ja asetuksiin tehdyt muutokset tallennetaan. Pelitilassa Esc-näppäintä painettaessa käyty taistelu automaattisesti luovutetaan, ja siirrytään häviötilaan.

Q- ja A-näppäimiä käytetään ainoastaan pelitilassa suoritettaviin toimintoihin. Molempia näppäimiä käytetään kiertämään kameraa X-akselin suuntaisesti pelimaailmassa, jolloin katselukulma vaihtuu. Katselukulma rajoittuu maksimissaan suoraan ylhäältä kuvattuun kuvakulmaan, ja minimissään lähes maantasolla olevaan kulmaan.

W- ja S-näppäimet vaikuttavat kameran zoomin toimintaan. W-näppäintä painettaessa kameraa liikutetaan lähemmäksi sen hetkistä katselupistettä. Vastaavasti S-näppäimellä kameraa liikutetaan kauemmaksi maantasosta. Zoomin toimintaa on rajoitettu niin, että kameran liike säilyy pelimaailman sisällä.

Kameraa liikutetaan siirtelemällä hiiren kursoria sovellusikkunan reunoilla. Sama toiminto voidaan suorittaa myös näppäinkomennoilla, joihin käytetään I, J, K ja L-näppäimiä. Kyseisiä näppäimiä käytetään nuolinäppäinten tavoin liikuttamaan kameran sijaintia pelimaailman sisällä.

Pelaajan tukikohtaan palaamisen helpottamiseksi pelissä on toteutettu toiminto, joka palauttaa kameran sijainnin tukikohtaan. Toiminnon suorittamiseen käytetään joko H- tai Home-näppäintä. Tukikohdan sijainti tallennetaan pelin käynnistyessä muistiin, jota toiminto hyödyntää kameran liikuttamiseen.

Pelin voi halutessaan pysäyttää. Toiminnon suorittamiseen käytetään P-näppäintä, jolloin pelin toiminnot jäädytetään. Painettaessa nappia uudelleen, toiminnot palautuvat ennalleen ja peli jatkuu normaalisti.

Peleissä käytetään yleensä objektien ohjaamiseen nuolinäppäimiä. Myös PPC:ssä käytetään nuolinäppäimiä vapaa-ajomoodin ollessa aktiivinen kontrolloimaan objektin liikkumista. Nuolinäppäimiä käytettäessä objekti liikkuu eteen- ja taaksepäin sekä kääntyy Y-akselin ympäri.

Taistelun aikana pelissä voi halutessaan vaihtaa taustalla soivan kappaleen toiseen. Tämä toiminto tapahtuu painamalla PageUp-näppäintä, jolloin soittolistasta valitaan satunnaisesti uusi kappale soitettavaksi.

Taistelun käynnistyessä minimap ei ole näkyvissä. Painamalla M-näppäintä minimap otetaan käyttöön ja se tulee näkyviin. Toinen painallus sulkee minimapin, jolloin minimap poistuu näkyvistä.

Näppäinkontrollien lisäksi peliä ohjataan hiirellä. Hiirtä käytetään liikuttamaan kameraa pelimaailmassa sekä valitsemaan ja liikuttamaan objekteja. Uusien joukkojen luonti tapahtuu myös hiirtä käyttäen, klikkailemalla hiirellä hudissa olevia valinta vaihtoehtoja.

Objekteja valitaan painamalla hiiren vasemman puoleista näppäintä, joko objektin kohdalta, tai pitämällä näppäintä pohjassa ja rajaamalla tietty alue, jonka sisältä kaikki pelaajan objektit valitaan. Joukkoja liikutetaan klikkaamalla hiiren oikean puolesta näppäintä kartalla, jolloin kaikki valittuna olevat joukot etsivät reitin kyseiseen pisteeseen ja aloittavat sinne siirtymisen.

Pelissä olevaa kameraa liikutetaan pelimaailmassa viemällä hiiren kursori sovellusikkunan laitoihin, jolloin kamera ryhtyy liikkumaan kyseiseen suuntaan. Kameran liike on kuitenkin rajoitettu niin ja kamera pysyy aina pelimaailman sisäpuolella.

4.9 Järjestelmävaatimukset

PPC:tä kehittäessä on pyritty mahdollisimman hyvään suorituskykyyn hitaammillakin tietokoneilla. Peli on suunniteltu niin, että sen muistinkäyttö on mahdollisimman vähäinen sekä grafiikanpiirto kuormittaa suoritinta vain tarvittaessa. Jotta peli toimisi mahdollisimman sulavasti, näytönohjaimen tulisi tukea OpenGL 3D-kiihdytystä, jolloin grafiikan piirto suoritetaan näytönohjaimella tietokoneen suorittimen sijaan.

Taulukko 3. PPC:n Järjestelmävaatimukset

Selite	Minimi	Suositus
Suoritin	800MHz	1000MHz
Muisti	64Mb	256Mb
Näytönohjain	On	3D-kiihdytys, OpenGL 2.1
Vapaa kovalevytila	72Mt	72Mt
Käyttöjärjestelmä	Windows 2000 tai uudempi, Linux 2.6 kernel tai uudempi.	Windows XP tai uudempi, Linux 2.6 kernel tai uudempi.

Pelimoottori toimisi myös mobiililaitteilla, mikäli se ei käyttäisi GLU -kirjastoa. GLU -kirjaston käyttöön on päädytty sen tarjoamien 3D-maailman näkymän perspektiivin asettamisfunktioiden vuoksi. Funktiot mahdollistavat yksinkertaisen näkymän siirtämisen, muokkaamisen ja kohdistamisen tiettyyn pisteeseen 3D-maailmassa.

4.10 Testaus

PPC:n testaus on opinnäytetyötä kirjottaessa ollut vielä kesken, sillä peli on saavuttanut vasta beeta-vaiheen. Beeta-vaiheessa peliä tullaan testaamaan mahdollisten ohjelmointivirheiden varalta ja peliä testataan myös muiden virhetilanteiden sekä väärin toimivien ominaisuuksien varalta.

Testaus tullaan suorittamaan blackbox testausmenetelmällä, jossa testaaja ei tiedä miten pelin tulisi toimia tietyissä tilanteissa, vaan testaaja raportoi kaikenlaisen mielestään poikkeavan käytöksen. Raportointi suoritetaan pelille tarkoitettuun virheidenraportointijärjestelmään, josta kehittäjät voivat nähdä testaajien löytämät virheet.

Pelin testaajina toimivat kehittäjien lähipiiri ja ystävät sekä ohjelmistoakatemian opiskelijat ja opettajat. Peliä tullaan mahdollisesti testaamaan myös ohjelmistoakatemialla järjestettävässä ohjelmistotestaus-kurssissa osana sen suoritusta.

5 INTERNET-SIVUSTO

Internet-sivusto on tärkeä osa pelin näkyvyyden ja tunnettuuden kannalta. Sivusto toimii eräänlaisena tukikohtana pelille, koska se sisältää uutisia pelin kehityksestä sekä lisämateriaalia pelistä. Sivusto sisältää myös linkit pelin lataamiseksi eri alustoille ja niiden eri versioita.

Ladattavaa lisämateriaalia sivustolla ovat esimerkiksi taustakuvat, pelistä otetut kuvankaappaukset ja pelissä esiintyvä musiikki. Sivustolta löytyy myös peliopas PPC:n pelaamisen avuksi, jossa esitetään kaikki pelin toiminnot sekä näppäinkomennot.

5.1 Suunnitelma

Sivusto suunniteltiin alusta lähtien toimimaan Zend Framework -luokkakirjaston päälle. Suunnittelussa otettiin huomioon sivujen helppo ja nopea päivitys, koska sivustolla tultaisiin julkaisemaan paljon erilaista tietoa, joita saatettaisiin päivittää jopa useita kertoja viikossa. Sivuston suunnittelussa päädyttiin ratkaisuun, jossa erillistä hallintopuolta ei sivustolle toteutettu, vaan sisältöä hallinnoidaan kokonaisuudessaan tietokantojen kautta. Näiden tietokantojen hallintaan käytetään phpMyAdmin php-pohjaista tietokantahallintajärjestelmää, joka on asennettuna palvelimelle.

Sivuston suunnittelussa asetettiin vaatimukseksi XHTML:n käyttäminen sen standardinmukaisuuden ja laajan selaintuen vuoksi. Sivustolla tultaisiin käyttämään CSS-tyylitiedostoja sen ulkonäön määrittämiseen. Vaatimukseksi sivustolle asetettiin, että sen täytyisi läpäistä W3C:n XHTML- ja CSS-validaattorit.

Sivuston sisältämien sivujen tiedot tallennetaan tietokantaan, josta ne voidaan hakea nopeasti erilaisia käyttötarkoituksia varten. Tiedot koostuvat sivun otsikosta, tekstistä sekä luonti- ja muokkausajasta. Tietokantaan kerätään myös tiedot mahdollisista ladattavista tiedostoista ja niiden latauksista.

Sivustolle suunniteltiin myös palautesivu, joka koostuu lähettäjän nimestä, sähköpostiosoitteesta sekä palautteesta. Täytetyt tiedot tultaisiin tarkistamaan Zend Frameworkin tarjoamalla tarkistustoiminnolla, jotta kaikki tarvittavat tiedot olisivat täytettyinä, eikä tietokantaan saapuisi virheellistä tietoa.

Sivustolle päätettiin suunnitella myös virheiden käsittelijä, joka kerää kaikki sivustolla mahdollisesti tapahtuvat virheet käsittelyä varten. Mahdollisen virheen sattuessa se kirjattaisiin tietokantaan sekä ohjattaisiin sivustolla vierailija sivulle, joka ilmaisee virheen tapahtuneen. Esimerkkinä voitaisiin pitää tilannetta, jossa haettua sivua tai tiedostoa ei löydy, ja käyttäjä ohjataan sivulle, joka kertoo virheen.

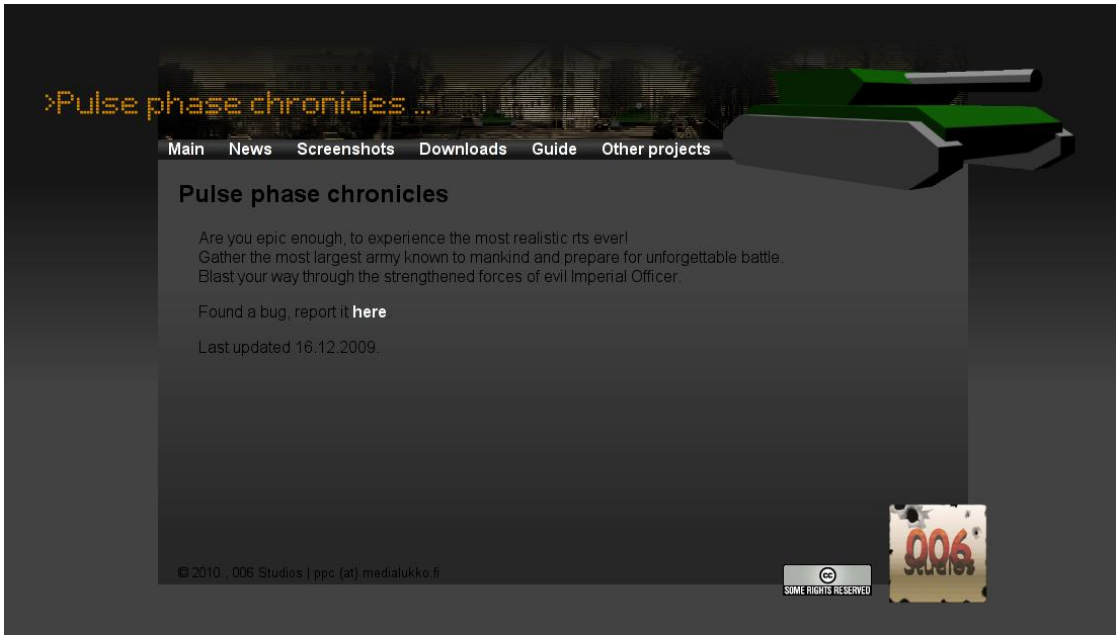
Sivustolle suunniteltiin toiminto, joka pitää yllä tiedostojen latauksesta kertovaa laskuria. Laskurin toiminta perustuu jokaisen latauksen kirjaamiseen tietokantaan josta tiedostokohtainen latausten määrä voidaan laskea. Jokaisesta latauksesta tietokantaan kirjattaisiin käyttäjän ip-osoite, latausaika, käytetty selain, sekä ladatun tiedoston nimi.

Sivustolle päätettiin suunnitella myös toiminto, joka kerää tietoa sivustolta ladattavista sivuista. Toiminto kirjaa tietokantaan jokaisen sivulatauksen, sekä tallentaa latauksesta vierailijan ip-osoitteen, latausajan, käytetyn selaimen, sekä sivun osoitteen. Toimintoa voidaan käyttää pitämään kirjaa suosituista sivuista sekä sivuston kävijämääristä.

Sivustolle haluttiin suunnitella myös käyttäjälle näkyvä laskuri, joka ilmaisee kyseisen sivun prosessointiin käytetyn ajan. Lukua voidaan käyttää osoittamaan palvelimen suorituskykyä kyseistä sivua ladattaessa.

5.2 Toteutus

Sivusto on toteutettu suunnitelman päälinjoja seuraten. Alkuperäisestä suunnitelmasta poiketen sivustolle ei toteutettu palauteosiota eikä sivujen prosessointiaika-laskuria. Sivuston ulkoasun hiomiseen käytettiin huomattavasti aikaa, siinä käytettyjä kuvia kuvattiin ympäri Kotkan kaupunkia ja yhdisteltiin toimivaksi ja hyvännäköiseksi kokonaisuudeksi. Sivuston ulkoasu toteutettiin pelin teemaan sopivaksi ja sivustolle lisättiin pelissäkin esiintyviä objekteja (Kuva 26).



Kuva 26. Sivuston ulkoasu toteutuksen ollessa valmis.

Sivuston pääsivulla kerrotaan pelin pelityyppi sekä pelisarjassa esiintyvää tarinaa. Pääsivulla mainitaan myös edellinen sivuston päivityskerta, ja se sisältää linkin virheiden raportointi sivulle. Sivustolle siirryttäessä saavutaan aina pääsivulle, josta voidaan navigoida eteenpäin.

Uutiset-sivulla raportoidaan pelinkehityksessä tapahtuneista edistyksistä sekä kehitystiimin peliin liittyvästä toiminnasta. Uutiset-sivulle voidaan lisätä myös linkkejä, esimerkiksi sivustolle lisätyistä uusista pelikuvista.

Kuvankaappaus-osio sisältää pelitilanteista ja pelikehityksestä otettuja kuvia, joilla osoitetaan pelin kehityksen edistymistä. Kuvia voidaan tarkastella niiden oikeassa koossa klikkaamalla haluttua kuvaa, jolloin se avautuu selaimeen suuremmassa koossa.

Lataukset-osiossa on listattuna pelin eri versiot sekä ladattava oheismateriaali. Oheismateriaalia ovat esimerkiksi mainosvideo, taustakuvat ja pelissä esiintyvä musiikki. Pelistä ladattavana ovat eri käyttöjärjestelmille tarkoitetut versiot ja niiden asennuspaketit sekä ilman asennusta toimivat versiot. Pelien lataukset kirjataan tietokantaan ja latausmäärät ilmoitetaan lataukset-sivulla.

Opas-osio on tarkoitettu olemaan apuna pelikontrollien ja pelitoimintojen käyttämiselle. Opas käsittelee kaikki pelin pelaamiseen vaadittavat toiminnot ja näppäimistökontrollit, joilla peliä voi ohjata hyödyntäen kaikkia pelin tarjoamia toimintoja. Oppaassa ohjataan käyttäjää kohta kohdalta tekemään toiminnon vaatimia toimenpiteitä. Opas sisältää ohjeistuksen valikon toimintaan, objektin valintaan, hudin käyttöön ja seuraamiseen, rakennuksien ja joukkojen hallintaan, rakentamiseen, liikkumiseen, hyökkäämiseen sekä näppäinkomentoihin.

Muut projektit-osio käsittää kehitystiimin muut projektit, jotka liittyvät kyseiseen peliin. Projektit ovat pelisarjan edelliset osat, jotka liittyvät oleellisesti kyseisen pelin tarinaan. Tulevaisuudessa mahdollisten jatko-osien sivustot tullaan julkaisemaan muut projektit-osiossa.

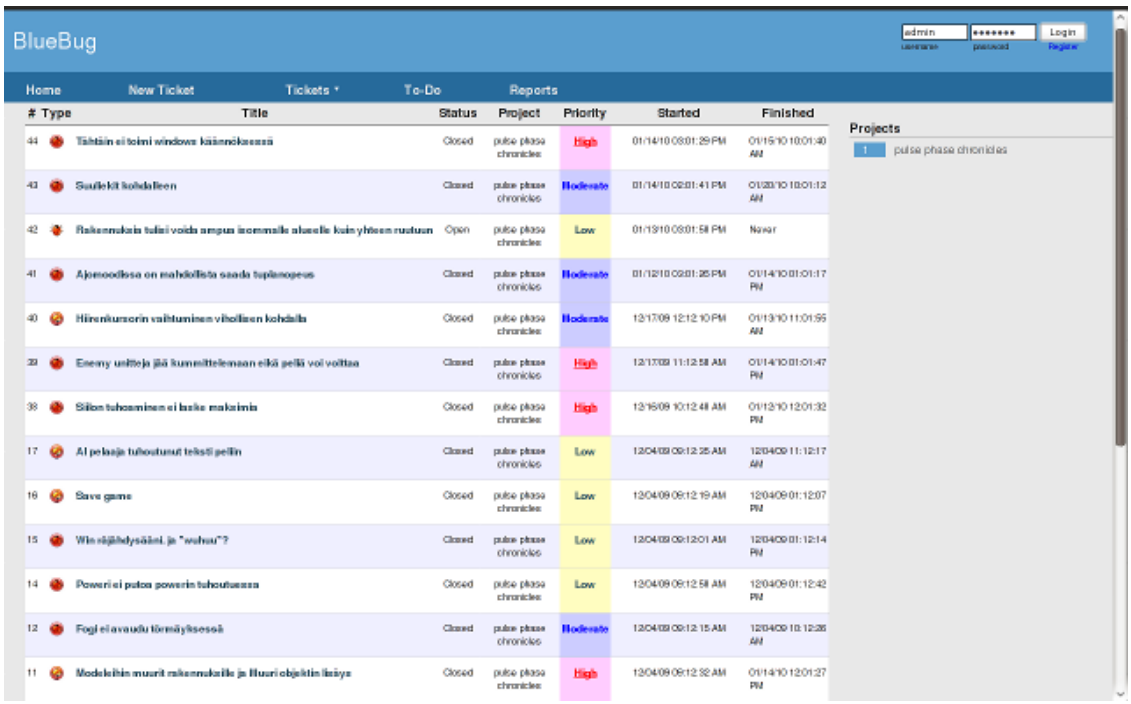
6 BUG TRACKING SYSTEM

Bug tracking system eli virheidenseuraamisjärjestelmä on ohjelma, jota käytetään ohjelmistokehityksessä kirjaamaan sovelluksessa ilmenneitä virheitä sekä ylläpitämään niiden tilaa ja vakavuutta. Virheidenseuraamisjärjestelmää käytetään, koska se helpottaa ylläpitämään ohjelmassa tapahtuneita virheitä, sekä helpottaa projektin hallintaa laajoissa töissä, joissa useat tekijät tekevät töitä samanaikaisesti.

Bug tracking system koostuu useista virheraporteista, joissa yleensä ilmaistaan ainakin virheen ilmoitusaika, sen vakavuus, projektin nimi, virheen tila, lyhyt kuvaus virheestä sekä selostus virheestä ja siitä, miten sen voi toistaa. Virheiden korjaamisen voi virheidenseuraamisjärjestelmässä osoittaa tietylle henkilölle, joka helpottaa projektinhallintaa ja sovelluksen virheiden kokonaisuuden hahmottamista.

Virheidenseuraamisjärjestelmässä virheiden vakavuutta voidaan muuttaa. Vakavuusastetta käytetään priorisoimaan virheiden korjaamisjärjestystä ja ilmaisemaan, voidaanko sovellusta käyttää virheestä huolimatta. Kun virhe todetaan korjatuksi, sen tila voidaan vaihtaa suljetuksi, joka ilmaisee muille, että virhe on korjattu. Joissain virheidenseuraamisjärjestelmissä voidaan käyttää myös useita muita tiloja ilmaisemaan virheen korjauksen edistymistä.

Opinnäytetyössä virheidenseuraamisjärjestelmäksi valikoitui BlueBug, joka on yksinkertainen ja selkeä web-pohjainen järjestelmä virheiden seuraamiseen (Kuva 27). BlueBug-järjestelmä sisältää vain oleellisimmat ominaisuudet virheidenseuraamisjärjestelmästä ja soveltuu erittäin hyvin yksinkertaista virheiden seurantaa vaativiin projekteihin.



Kuva 27. BlueBug virheidenseuraamisjärjestelmässä ilmoitetut virheet ja ominaisuudet ilmaistaan selkeästi ja yksinkertaisesti käyttäjälle.

BlueBug sisältää myös toiminnon, jolla virheiden kirjaamisen lisäksi voidaan pyytää tekijöitä lisäämään kehitettävään sovellukseen uusia ominaisuuksia. Järjestelmässä voidaan tarkastella myös raporttia, joka tekee yhteenvedon avonaisista ja suljetuista virheistä sekä uusista ominaisuuksista.

Uutta ilmoitusta luodessa sille annetaan otsikko, kuvaus, valitaan onko ilmoitus virhe vai ominaisuus, valitaan sen vakavuus sekä lisätään mahdollinen liitetiedosto. Uuden ilmoituksen voi luoda, kuka tahansa ilman kirjautumista järjestelmään. Näin ollen myös anonyymit ilmoitukset ovat mahdollisia. Järjestelmässä olevia ilmoituksia voi poistaa tai muokata vain kyseisen projektin hallitsija.

7 JATKOKEHITYS

PLEI-pelimoottori tulee laajentumaan tulevaisuudessa uusien projektien myötä. Pelimoottori on vielä kehitysvaiheessa, eli siihen tullaan lisäämään uusia ominaisuuksia sekä mahdollisesti muokkaamaan jo olemassa olevia. Uusien projektien tuomien vaatimusten myötä pelimoottori tulee laajentumaan ja kehittymään entistä monipuolisemmaksi pohjaksi erityyppisille projekteille.

PLEI-pelimoottoria tullaan optimoimaan ainakin grafiikanpiirron ja matemaattisten laskutoimitusten osalta. Siihen lisätään myös tuki säikeistykselle. Lisäksi pelimoottoriin lisätään myöhemmin tuki mobiililaitteille, koska pelisarjan suunnitteilla oleva seuraava osa tulee olemaan saatavilla myös mobiilisovelluksena.

Grafiikanpiirto siirretään mobiililaitteille mukautuvaan muotoon sekä siihen lisätään tuki tekstuurien käsittelyyn. Lisättävällä säikeistyksellä pelimoottori osaa hyödyntää järjestelmästä mahdollisesti löytyvät useat suoritinytimet, jolloin sovelluksen suorituskyky kasvaa, ja se osaa hyödyntää käytettävän tietokoneen kaikkia tehoja.

PPC:n valmistuessa beeta-vaiheesta peliä ei enää jatkokehitetä. Beeta-vaiheen jälkeen PPC:stä löydetyt ohjelmointivirheet korjataan ja peli saatetaan valmiiseen muotoonsa, minkä jälkeen se julkaistaan ASSEMBLY 2010 -messuilla.

8 LOPPUSANAT

Opinnäytetyön aiheenvalinta ei ollut laisinkaan hankalaa, sillä olemme koko opiskeluajan suunnitelleet ja kehittäneet pelejä ja peleihin liittyviä projekteja. Kehitimme Pulse Phase -pelisarjan, jota opinnäytetyönä tehty peli ja pelimoottori luontevasti jatkavat.

Päätimme tehdä pelisarjan seuraavaan osaan laajakäyttöisen pelimoottorin, jota tultaisiin mahdollisesti käyttämään jatkossakin vastaavanlaisiin peliprojekteihin. Pelimoottorin kehitystyö veikin opinnäytetyöhön käytetystä ajasta yli puolet, koska halusimme varmistaa sen toimivuuden jatkossa kehitettäviä pelejä varten. Pelimoottorin lopputulokseen olemme varsin tyytyväisiä, mutta tietysti aina parannettavaa löytyy. Pelimoottoria tullaan kuitenkin jatkokehittämään jo pelisarjan seuraavaan osaan, joka on suunnitteilla.

PPC:stä haluttiin haastava ja monipuolinen projekti toteuttaa. Sen peligenreksi valittiin reaaliaikastrategia, jonka tiedettiin olevan vaikea, mutta mielenkiintoinen genre meille molemmille. Lopputulos miellyttää meitä ja mielestämme pelin toteutus on varsin onnistunut ja vastaa odotuksia. Pelin tarjoama tunnelma ei myöskään ole niin sanotusti puuduttava, eli peliä jaksaa myös pelata, eikä siinä ilmene häiritseviä ominaisuuksia.

Opinnäytetyötä tehtäessä kohtasimme useita hidasteita, jotka olivat haastavuudeltaan toisistaan poikkeavia. Näihin hidasteisiin löydettiin kuitenkin aina toimiva ratkaisu, eikä näin ollen niillä ollut vaikutusta lopputulokseen. Haastavin osuus projektissa oli grafiikoiden toteutus, sillä kumpikaan meistä ei ole graafikko.

LÄHTEET

1. Stroustrup, B. 2000. C++-ohjelmointi. Suom. Veli-Pekka Ketola. Jyväskylä: Teknolit.

2. Johdanto olio-ohjelmointiin. VirtuaaliAMK. Saatavissa: http://www.pspt.fi/virtuaaliamk/etr/olio_ohjelmointi_vamk/materiaali1.html [viitattu: 5.2.2010].

3. Simple DirectMedia Layer. Saatavissa: http://www.libsdl.org [viitattu:5.2.2010].

4. OpenGL. The Industry's Standard for High Performance Graphics. Saatavissa: http://www.opengl.org [viitattu:5.2.2010].

5. OpenGL Safety Critical Profile. Saatavissa: http://www.khronos.org/opengles/sc/ [viitattu:8.2.2010].

6. OpenGL ES. Saatavissa: http://www.khronos.org/opengles/ [viitattu:8.2.2010].

7. PHP. Saatavissa: http://www.php.net [viitattu:8.2.2010].

8. PHP opas. Saatavissa: http://www.2kmediat.com/php/ [viitattu:8.2.2010].

9. Zend Framework. Saatavissa: http://framework.zend.com/ [viitattu:9.2.2010].

10. Käyttöliittymän ylläpidettävyys. Saatavissa: http://www.tol.oulu.fi/kurssit/ot3/Lectures/OT3_L3.html [viitattu:9.2.2010].

11. What is Subversion?. Saatavissa: http://svnbook.red-bean.com/en/1.4/svn.intro.whatis.html [viitattu:11.2.2010].

12. Code::Blocks. Saatavissa: http://www.codeblocks.org [viitattu:11.2.2010].

13. A* Pathfinding for Beginners. Saatavissa: http://www.policyalmanac.org/games/aStarTutorial.htm [viitattu:12.2.2010].

14. SDL_Image. Saatavissa: http://www.libsdl.org/projects/SDL_image/ [viitattu:16.2.2010].

15. Display Lists Tutorial. Saatavissa: http://www.lighthouse3d.com/opengl/displaylists/ [viitattu:22.2.2010].

```
int AIPlayer::searchPlace( Vector3D position, UnitBase* unit )
{
    ObjectBase* obj = static_cast<ObjectBase*>(unit);
    unsigned int mapWidth = m_cMapHandler->getWidth();
    unsigned int mapHeight = m_cMapHandler->getHeight();
    int tilesX, tilesY, colStart, colEnd, rowStart, rowEnd, curBlock;


    int row = abs(((int)position.z-Engine::levelMultiply)/Engine::levelMultiply)-1;
    int col = abs(((int)position.x-Engine::levelMultiply)/Engine::levelMultiply)-1;

    int i = 1;
    Block* mapBlocks = m_cMapHandler->getMapdata();
    for ( unsigned int j = row + 1; j <= row+m_cMapHandler->getHeight()/2; j++ )
    {
        colStart = (col - i > 0 )? col - i: 0;
        colEnd = ( col + i < mapWidth )? col + i: mapWidth;

        rowStart = ( row - i > 0 )? row - i: 0;
        rowEnd = ( row + i < mapHeight )? row + i: mapHeight;

        for ( int x = colStart; x <= colEnd; ++x )
        {
            if ( x > mapWidth ) continue;
            if ( rowStart > mapHeight ) continue;

            curBlock = rowStart*mapWidth+x;
            if ( isFree( x, rowStart, obj ) )
            {
                return (rowStart-1)*mapWidth+(x-1);
            }

            curBlock = rowEnd*mapWidth+x;
            if ( isFree( x, rowEnd, obj ) )
            {
                return (rowEnd-1)*mapWidth+(x-1);
            }
        }

        for ( int y = rowStart + 1; y <= rowEnd - 1; y++ )
        {
            if ( colStart > mapWidth - 1 ) continue;
            if ( y > mapHeight - 1 ) continue;

            curBlock = y*mapWidth+colStart;
            if ( isFree( colStart, y, obj ) )
            {
                return (y-1)*mapWidth+(colStart-1);
            }


            curBlock = y*mapWidth+colEnd;
            if ( isFree( colStart, y, obj ) )
            {
                return (y-1)*mapWidth+(colEnd-1);
            }
        }
        ++i;
    }

    return -1;
}
```

```
void MapHandler::load( const char* name, ObjectHandler *Objects )
{
   m_sCurrentLevel =  DATA_PATH "Data/Levels/";
   m_sCurrentLevel += name;
   m_sCurrentLevel += ".lvl";

   std::ifstream mapFile;
   mapFile.open( m_sCurrentLevel.c_str(), std::ifstream::in | std::ifstream::binary );
   if ( !mapFile.is_open() ) {
     throw ExceptionCritical(tostring("Map ") + tostring(name) + tostring(" not found."));
   }

   m_suiWidth = m_suiHeight = 0;
   m_cMapData = NULL;
   mapFile.seekg(0, std::ios::beg);
   mapFile.read(reinterpret_cast<char *>(&m_suiWidth) , sizeof(char));
   mapFile.read(reinterpret_cast<char *>(&m_suiHeight) , sizeof(char));

   unsigned int length = m_suiWidth * m_suiHeight;
   mapFile.seekg(2,std::ios::beg);
   m_cMapData = new unsigned char[length];
   mapFile.read((char*)m_cMapData, length);

   m_sBlocks = new Block[length];
   memset( m_sBlocks, 0, sizeof(Block)*length );
   for ( unsigned int i = 0; i < length; i++ )
      m_sBlocks[i].type = m_cMapData[i];

   unsigned char* m_cObjectData = new unsigned char[length];
   mapFile.read((char*)m_cObjectData, length);

   m_iMaxIndex = length;

   for ( unsigned int s = 0; s < m_suiHeight; s++ )
   {
      for ( unsigned int i = 0; i < m_suiWidth; i++ )
      {
         if ( (unsigned int)m_cObjectData[(s*m_suiWidth)+i] != 0 )
         {
            Objects->addObject( (unsigned int)m_cObjectData[(s*m_suiWidth)+i], levelMultiply*i, levelMultiply*s );
         }
      }
   }

   mapFile.close();

   delete[] m_cObjectData;
}
```