Tasosuunnittelu Source Engine -pelimoottorilla

Viestintä
3D-visualisointi
Opinnäytetyö
31.5.2009

Arttu Mäki

Metropolia

Koulutusohjelma Viestintä	**Suuntautumisvaihtoehto** 3D-visualisointi
Tekijä Arttu Mäki	
Työn nimi Tasosuunnittelu Source Engine -pelimoottorilla	
Työn ohjaaja/ohjaajat Kristian Simolin	

Työn laji	**Aika**	**Numeroidut sivut + liitteiden sivut**
Opinnäytetyö	31.5.2009	31

TIIVISTELMÄ

Opinnäytetyön tutkimuksen kohteena selvitettiin ja ratkaistiin yleisiä ongelmia ja haasteita liittyen Valve Softwaren kehittämään, Source Engine –pelimoottorilla toimivaan tasosuunnitteluohjelmaan ja sen käyttöön. Työ käy läpi tärkeimmät suunnitteluun liittyvät työtavat geometrian rakentamisesta valaistuksen määrittämiseen. Työnä esitellään Valve Softwarelle 2008 keväällä myyty projekti ”Fastlane”, josta tuli yksi virallisista kartoista Team Fortress 2 –moninpeliin.

Tasosuunnittelulla tarkoitetaan kentän rakentamista alkuperäisten suunnitelmien perusteella aina toimivaksi pelikentäksi asti. Kenttään rakennetaan pelimekaaniset elementit, valaistus, mallit ja äänet. Opinnäytetyössä on tarkasteltu pelisuunnittelun historiaa, yritysten taustaa, sekä käyty läpi käytettävän ohjelmiston työkalut ja toiminnot.

Työssä on käytetty apuna laajaa valikoimaa eri lähdemateriaaleja koskien taso- ja pelisuunnittelua ja käyty läpi tapauskohtaisesti se, miten voitaisiin selvitä prosessista mahdollisimman tehokkaasti hyödyntäen Valve Softwaren tarjoamia monipuolisia työkaluja.

Teos/Esitys/Produktio
Säilytyspaikka Metropolia Ammattikorkeakoulu
Avainsanat tasosuunnittelu, pelisuunnittelu, Source Engine, Valve Software, Valve Hammer Editor, Team Fortress 2

Metropolia

Culture

Degree Programme in	Specialisation
Applied Sciences	3D-visualisation

Author
Arttu Mäki

Title
Level Design with Source Engine

Tutor(s)
Kristian Simolin

Type of Work	Date	Number of pages + Appendices
Bachelor's Thesis	31 May 2009	31

ABSTRACT

This thesis addresses the design and evaluation of level design process using Valve Software's Source Engine executables and programs. It concerns the usage of the said tools, known problems and troubleshooting the appropriate methods for developing a great level for Team Fortress 2 videogame. This thesis describes the project "Fastlane", which was sold to Valve Software during the summer of 2008.

The first case study is to examine and find solutions for designers to work and use the tools more efficiently. The thesis also reveals the facts from the eye of the designer itself. This also includes a two-phase study on the history of level design and the companies behind the culture we call the games industry. The second case is a study to determine efficient ways to use the tools. The thesis explains the basic toolset from geometry modification to setting up lighting and entities. This thesis includes a wide range of sources concerning the methods behind level- and game design processes and how you can take the tasks and targets much further in your own project.

Thesis results are shown between the cases to introduce each chapter each as its own case to the reader. Many questions were asked in the formation of this study. Game development is a growing business and needs more interested people to try and feel the art of game design. Fastlane level is a perfect case to determine the low and high points of multiplayer level design process and the thoughts behind it and Source Engine gave a powerful platform to work on.

Growing business, rapidly improving tools and technology is giving a hard time to developers around the world to keep everyone in the loop. It's recommended to keep working on the defined skill set as often as possible.

Work / Performance /Project

Place of Storage
Metropolia Library / Tikkurila Unit

Keywords
level design, game design, source engine, valve software, valve hammer, team fortress 2

Sisällys

1 JOHDANTO

Peliala kehittyy ja kasvaa vuosi vuodelta ja alalla tarvittavan henkilöstön määrä ei vastaa tarvetta. Suomessa useat yritykset palkkaavat työvoimaa ulkomailta, jotta nopeasti kasvavan toiminnan tarpeet saataisiin täytettyä. Pelisuunnittelu on yksi näistä alueista, johon lisätyövoimaa tarvittaisiin.

Pelitilojen suunnittelu ei ole uusi ilmiö. Lapset tekevät leikeissään sitä päivittäin rakentamalla monimutkaisia pelitiloja, joissa säännöt voivat muuttua epäröimättä hetkessä. Tietokonepelisuunnittelu sen sijaan on ollut läsnä keskuudessamme vasta hieman yli 30 vuotta ja tänä aikana se onkin kasvanut ja kehittynyt dramaattisesti. Tasosuunnittelu oli aikoinaan pelkkä osa pelisuunnittelua. Yksi henkilö pystyi tekemään pelin kokonaisuudessaan täysin yksin hyödyntäen omia taitojaan. Nykyään pelisuunnitteluprosessi on kuitenkin muuttunut niin monimutkaiseksi, että tasosuunnittelijoiden osuus pelin kehityksessä on pieni, mutta erittäin tärkeä osa.

Opinnäytetyön päätavoitteena on tutkia Valve Software -pelistudion kehittämän Source Engine -pelimoottorilla toimivan tasosuunnitteluohjelman käyttöä ja suunnittelusta koituvia ongelmia ja esteitä niin suunnittelijan kuin pelaajankin näkökulmasta. Opinnäytetyössä läpikäyty projekti on Valve Softwarelle etätyönä tehty, Team Fortress 2 -videopeliin suunniteltu taso ”Fastlane”. (Kuva 1)

Kuva 1: Team Fortress 2 –kenttä ”Fastlane”.

Opinnäytetyö koostuu viidestä osa-alueesta. Ensimmäisessä osassa käydään läpi opinnäytetyön tavoitteet, rajaukset sekä keskeisimmät määritelmät. Opinnäytetyön toisessa osassa esitellään tarkemmin työn taustalla oleva Valve Software ja Team Fortress 2 -peli. Työn kolmannessa osassa tutustutaan tasosuunnitteluun yleisesti ja käydään läpi pelialan taustalla olevia muita yrityksiä. Osassa käydään läpi myös pelisuunnittelussa käytettävien pelimoottorien historiaa.

Neljännessä osassa tutustutaan tasosuunnittelun eri vaiheisiin työkalujen esittelystä viimeistellyn kentän testaukseen, valaistukseen ja teksturointiin. Lopuksi osassa viisi tarkastellaan tasosuunnittelun tulevaisuutta ja sen asettamia vaatimuksia kehitykselle.

2 VALVE SOFTWARE

Ehkä keskeisin ja arvostetuin yritys pelialalla on amerikkalainen Valve Software, jonka portfolioon kuuluu miljoonia myynyt Half-Life-pelisarja sekä laajalti menestynyt ja räjähdysmäisesti kasvava Steam-asiakasohjelmisto, jonka avulla sadat pelikehittäjät voivat myydä pelejä kasvaville asiakasryhmille.

Valve Softwaren portfolioon kuuluu myös Counter-Strike-, Day of Defeat- ja Team Fortress -moninpelit. Tämän portfolion ansiosta Valve on kasvanut alan suurimmaksi ja kannattavimmaksi yritykseksi yli 20 miljoonan maailmanlaajuisesti myydyn pelin kappalemäärällä. Valve hallitsee myös 80%:aa prosenttia Internetissä pelattavista moninpeleistä.

Valve työllistää yli 160 artistia, suunnittelijaa, ohjelmoijaa ja kirjoittajaa. 12 vuoden aikana Valve Softwaresta on tullut pelikehittäjien suunnannäyttäjä ja alan kasvavin ja tehokkain yritys.

2.1 Historia

Valve Softwaren ensimmäiset askeleet nähtiin Kirklandissa, Washingtonin osavaltiossa. Vuonna 2003 yritys muutti Bellevuen kaupunginosaan, aivan tuottajayritys Sierran läheisyyteen. Half-Life -pelisarjan johdattama suosio antoi vankan pohjan Valvelle jatkaa

alalla. Pelaajat saivat käyttöönsä työkalut, joilla kuka vaan pystyi rakentamaan omia kenttiä ja peli-ideoita Valven pelisovelluksien päälle. (Valve Software, 2009)

Valve on tunnettu sen jatkuvasta tuesta kehitystyökalujen julkaisun suhteen. Tämän ansiosta kehittyi nykyään Valven portfolioon kuuluvia pelejä kuten Counter-Strike, Day of Defeat ja Team Fortress. (Valve Software, 2009)

Tammikuussa 2008 Valve julkisti ostavansa Turtle Rock Studios -kehitysstudion ja heidän kehittämän teknologian. Tämän ansiosta Valven uusin tuotos, Left 4 Dead, kehittyi entisestään ja näki päivänvalon syksyllä 2008. (Valve Software, 2009)

2.2 Team Fortress 2

Kuva 2: Team Fortress 2.

Team Fortress 2 sisältyy Valve Softwaren uusimpiin tuotoksiin ja oli kehityksessä eri vaiheiden kera yli 10 vuotta. Team Fortress 2 (TF2) (Kuva 2) on luokkapohjainen verkossa pelattava moninpeli jonka alkuperäiset suunnittelijat ovat Robin Walker, John Cook ja Ian Caughley (Team Fortress, Wikipedia).

Pelin tarkoituksena on kahden eri joukkueen välinen taistelu. Pelaaja voi valita yhdeksästä toisistaan eroavista hahmoluokista omaan pelityyliin sopivan. Jokaisella hahmoluokalla on uniikit aseistukset ja ominaisuudet sekä taidot, joita hyödyntäen voidaan lyödä vastustajajoukkue. Pelin ideana onkin yhteistyö joten jokaisella hahmoluokalla on jotain annettavaa joukkuetoverille. Team Fortress 2 sisältää kolme eri pelimuotoa, joista suosituin on alueiden valtaus ja hallitseminen. Team Fortress -pelisarja tulikin tunnetuksi alun perin joukkuepelaamista kannustavasta lipunryöstö-pelimuodosta.

Tässä opinnäytetyössä suunniteltu kenttä käyttää ”Control Point” -pelimuotoa, jossa molempien joukkueiden tarkoitus on vallata ja hallita alueita. Jos jompikumpi joukkueista valtaa kaikki alueet, peli päättyy.

3 TASOSUUNNITTELU YLEISESTI

Tasosuunnittelu käsitteenä on laaja asia. Parhaat tasosuunnittelijat palaavat alkuperäisen suunnitelman pariin vielä projektin jälkeen ja arvioivat mahdollista prosessin parantamista, sekä käyvät läpi projektiin liittyvät haasteet läpikotaisesti. Oppinut suunnittelija ei karsasta ajatusta jättää vanhaa suunnitelmaa taakseen ja luoda jotain uutta ja virkistävää tilalle. Tämä ei kuitenkaan aina ole paras lähtökohta projektin eteenpäin viemiselle, mutta joskus suunnitelman muuttaminen voi olla hyväksi koko projektin kannalta. Suunnittelijana on myös tärkeää huomata, milloin omat rajat tulevat vastaan. Väsyykö suunnittelija näkemäänsä projektiin ja riittääkö into jatkaa projektin parantamista loppuun asti ilman, että heittää hyvää työtä roskakorin pohjalle. Suunnittelijalle on tärkeää myös kritiikin vastaanotto ja sen hyväksyminen.

Tason suunnittelu moninpelille eroaa täysin yksinpelikenttien suunnittelutavoista. Moninpelissä luodaan ns. ”leikkikenttä” oikeille ihmisille, kun taas yksinpelissä pelikokemus määräytyy yhden pelaajan ja tietokoneen ohjaamien elementtien yhteisymmärryksestä. On kuitenkin tärkeää muistaa, että sekä yksin- että moninpeleissä vallitsevat samat arvot pelikokemuksen ympärillä. Siksi on tärkeää, että suunnittelija tietää, mikä on hyvä tapa suunnitella kenttä jommallekummalle pelitavalle.

Pelikokemusta määritellään yleensä sanalla ”gameflow”, joka tarkoittaa pelin teknistä ja teoreettista kulkua pelaajan näkökulmasta. Tärkein osa tämän termin merkitystä on seuraavat arvot:

1. Annetaan pelaajalle mielikuva siitä, että hän voi mennä mihin tahansa.
2. Vaikka edellinen ei olekaan täysin mahdollista, ohjataan pelaajaa tämän tietämättä eteenpäin erilaisilla visuaalisilla vihjeillä.
3. Nämä arvot voidaan mahdollistaa käyttämällä suunnittelussa aina kahta tai kolmea eri reittiä paikasta A paikkaan B. Vältetään lineaarisia linjoja tason suunnittelussa.
4. Tason tempo on tärkeä määrittää jo suunnittelun alkuvaiheessa. Tason tempolla tarkoitetaan sitä matkaa, jonka pelaaja joutuu kävelemään saavuttaakseen pisteen B pisteestä A.
5. Mitä intressejä pelaajalla on valita reitti X tai Y pisteen A ja B välillä.

Hyvä tasosuunnittelu koostuu siis pelimekaanisista asioista, hyvästä visuaalisesta silmästä, itsekritiikistä ja kritiikin vastaanottamisen tärkeydestä. Tärkeintä on kuitenkin

yhteystyö oman itsensä kanssa. Näin saavutetaan toimiva kokonaisuus mahdollisimman pienellä resurssimäärällä.

3.1 Tasosuunnittelun kehitys

Tasosuunnittelun ensimmäiset askeleet tietokonepelien 3D-ympäristöissä otettiin vuonna 1974 kun Battlezone–niminen peli julkaistiin kotitietokoneille (PC, Mac). Peli oli jäljitelmä suositusta kolikkopelistä, jossa ohjattiin tankkia erilaisissa maastoissa samalla kun tarkoitus oli väistellä ohjuksia ja vihollisen hyökkäystä. Battlezone -pelin tasosuunnittelu olikin yksinkertaista ja se sisälsi lähinnä viivoista luotuja vuoristoja ja esteitä. (Gamasutra, 2006.)

Ensimmäinen ns. oikeana pidetty 3D-ympäristöön tehty peli oli ID Softwaren vuonna 1992 julkaisema Wolfenstein 3D, joka nosti riman aivan uudelle korkeudelle. Pelissä käytettiin yksinkertaista yhden sivun graafista ilmettä, mutta maailma oli muuten pinnoitettu eri materiaaleilla luoden illuusion kolmiulotteisista pinnoista. Wolfenstein -peliin rakennetut kentät luotiin TED5 –nimisellä ohjelmistolla, joka mahdollisti tehokkaan tavan luoda kenttiä pienellä aikataululla. Suunnitteluohjelmistossa luotiin kentät ylhäältä päin kuvatusta näkymästä (vrt. nykypäivän 4 eri suunnan ja perspektiivinäkymän puutteet). (Gamasutra, 2006.)

Vuosi 1993 muutti kaiken. ID Software julkisti aivan uuden version pelimoottoristaan, joka oli pakattu täyteen uusia tehokkaita ominaisuuksia, niin pelaajalle kuin suunnittelijallekin. Doom oli syntynyt. On selvää, että muut kehitysstudiot olivat kiinnostuneita teknologiasta ja täten syntyikin monia Doom –klooneja, jotka yrittivät parhaalla mahdollisella tavalla saada aikaiseksi samanlaista elämystä kuin mitä Doom joskus pelaajille tarjosi. (Gamasutra, 2006.)

Kuva 3: ID Softwaren Doom 1 ja Doom 2.

Doom (Kuva 3) oli ensimmäinen 3D -ympäristöön sijoittuva peli, jossa tuotiin peleihin uutena elementtinä mm. käytettävät hissit ja erilaisista ”triggereistä” käynnistyvät pelimekaaniset tapahtumat. Doom oli myös ensimmäinen peli, jossa käytettiin hyväksi tasosuunnittelun monimutkaisuutta. Pelaaja voitiin nostaa erilaisille tasoille, kun ennen tämä ei olisi ollut edes mahdollista. Uusi pelimoottori mahdollisti myös seinien monimutkaisemmat muodot. Aikaisempi rajoitus oli 90 asteen kulmissa. Uusi moottori poisti tämän rajoituksen ja avasi suunnittelijoille aivan uusia mahdollisuuksia tasosuunnittelun kannalta. Doom -pelimoottori mahdollisti myös erilaisten valojen käytön ympäristöissä. Aikaisemmin kaikissa peleissä valaistus oli sama ja täten etäisyyksien hahmottaminen oli joskus vaikeaa. Uusi tekniikka mahdollisti mm. eriväriset ja eri kirkkauksilla toimivat valot. (Gamasutra, 2006)

3.2 Pelimoottorit

Teknologian kehityksen perässä on joskus mahdotonta pysyä ja sama koskee myös vuosien mittaan nopeasti tapahtunutta pelimoottorien kehitystä.

Pelimoottorilla tarkoitetaan pelin alla toimivaa luurankoa, joka hallitsee lähes kaikkia pelissä tapahtuvia toimintoja kuten grafiikkaa, ääniä, tekoälyä, verkkosovelluksia jne.

3.2.1 Source Engine

Source Engine on Valve Softwaren kehittämä 3D-pelimoottori, jota on käytetty suurissa maailmanluokan peleissä kuten Half-Life 2 (Episode 1 ja 2), Counter-Strike, Day of Defeat , Team Fortress 2 ja kolmannen osapuolen peleissä kuten Dark Messiah of Might and Magic. Source -pelimoottori kehitettiin modulaariseksi, minkä takia se vuosien varrella kehittynyt teknisesti muita pelimoottoreita nopeammin – vaikkakin ei ehkä täysin yllä samalle tekniselle tasolle muiden kanssa. (Valve Software, VDC 2008)

Source Engine on ominaisuuksiltaan nopea, luotettava ja joustava pelimoottori vaativien tasojen rakentamista varten. Shader-pohjainen renderöintiteknologia auttaa pelikehittäjiä tuottamaan tehokkaasti ja nopeasti monimutkaisia, näyttäviä ja nopeatempoisia skenaarioita.

Source Engine -moottori hyödyntää kehittynyttä prosessoriteknologiaa kuten moniydinprosessoreita ja SIMD-teknologiaa. Moottori kykenee hyödyntämään myös uusimmat ominaisuudet uusista näytönohjaimista käyttäen DirectX -rajapintaa, tuoden visiosi eloon elinvoimaisen yksityiskohtaisesti. (VDC, Valve Software 2008.)

Kehittynyt shader–teknologia

- Tuki HLSL shader –teknologialle. Direct3D –pohjainen SM3.0 rajapinta.
- LOD-mallit maailmassa ja malleissa. ”Level of Detail” –teknologia mahdollistaa suorituskyvyn parantamiseen tehokkaat työkalut.

Dynaamiset valot ja varjot

- Radiositeettivalaistus. Valaistus säilytetään valokartoissa joka takaa optimaalisen suorituskyvyn monimutkaisenkin valaistuksen käytössä.
- HDR-valaistus. ”High Dynamic Range” –teknologia emuloi ihmissilmän reaktiota valon vaihteluun.
- Rim-valaistus. Teknologia joka mahdollistaa yksityiskohtaisen valaistusarvojen käytön mm. pelaajamalleissa ja objekteissa.

Erikoisefektit

- Partikkelit, volumetrinen savu, sade ja sumu.

- Partikkelieditori. Muokkaa ja luo omia partikkeleita tehokkaasti.
- Motion Blur. Tehokas tapa emuloida liikkeen tuntua teknisesti.
- Vesimateriaalit.

3.2.2 Muita pelimoottoreita

Unreal Engine -pelimoottori on yksi alan suosituimmista pelimoottoreista. Se on lisensoitu sadoille yrityksille eri puolille maailmaa, eri tarkoituksiin. Moottorin ensimmäinen versio tuli julki vuonna 1998 ja sen kehitti Epic Games. Unreal-moottorista on saatavilla jo kolme eri versiota ja ne sisältävät mm. tuen UnrealScript-toiminnoille, jotka muistuttavat osittain C++- tai Java-ohjelmointikieliä.

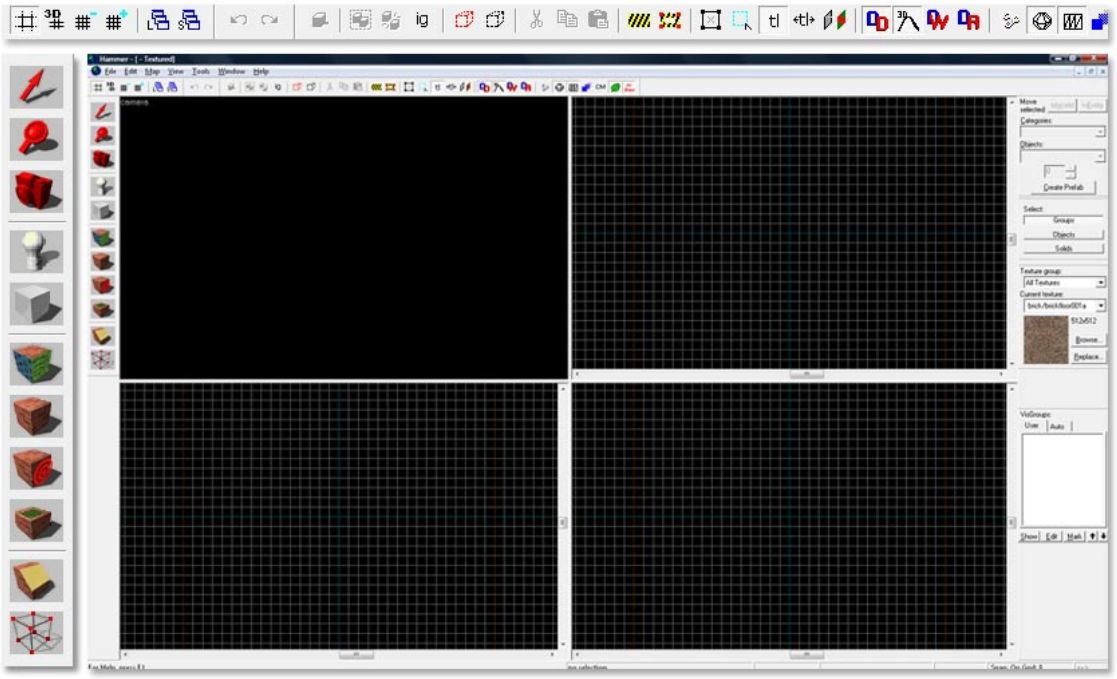
Unreal Engine 3 -versio on kehitetty käyttämään Microsoft DirectX 9 -teknologiaa 32- ja 64-bittiselle Windows-käyttöjärjestelmälle ja XBOX360 -alustalle. DirectX 10- ja DirectX 11 -rajapinnat ovat käytössä 32- ja 64-bittisellä Windows Vista -käyttöjärjestelmällä.

ID Tech on ID Softwaren kehittämä pelimoottoriperhe vuosien takaa. Doom oli ensimmäinen ID Tech -moottorilla luotu peli. Yrityksen uusin peli, Rage, on kehityksessä moottorin uusimmalla versiolla, ID Tech 5:lla. Tunnetuin ID Tech -sarjan moottoreista on ensimmäisen version ohella neljäs versio, jonka avulla on tuotettu mm. miljoonia myynyt Doom 3 ja sen lisäosat.

Muita mainittavia pelimoottoreita ovat suosittu Renderware, Jade Engine, CryEngine ja Lithtech.

4 TASOSUUNNITTELUN ERI VAIHEET

Source Engine -pelimoottorille tasoja työstäessä käytetään Valve Softwaren tarjoamia työkaluja. Nämä suunnittelutyökalut ovat ilmaisia ja kaikkien saatavilla, jos omistaa lisenssin yhteen yrityksen julkaisemasta tuotteesta. Samalla työkalupaketilla voi työstää kenttiä ja omaa sisältöä kaikkiin Valve Softwaren peleihin.



Kuva 4: Valve Hammer Editor, Valve Software

Kaikki työkalut ovat saatavilla Steam -asiakasohjelmiston kautta. Käyttäjä asentaa asiakasohjelmiston ja asentaa Source SDK -työkaluohjelmiston. Kyseisen ohjelmiston mukana tulee valmiudet luoda suunnitella oma peli alusta lähtien, omia karttoja, ääniä, malleja sekä suuri määrä täysin ilmaisessa jakelussa olevaa dokumentaatiota. Ohjelmistopaketissa tulee mukana myös mahdollisuus katsoa ja tutkia valmiita malleja ja luoda koreokraafisia luonnoksia valmiista hahmomalleista.

Tasosuunnittelussa käytetään Valve Hammer Editor –nimistä (Kuva 4) ohjelmistoa, jonka avulla tuotetaan lähes kaikki kenttiin tarvittava materiaali ääniä ja valmiita kolmiulotteisia malleja lukuun ottamatta.

Valve Hammer Editor on nopealla silmäyksellä hyvin yksinkertaisen näköinen ja voikin hämätä uusia käyttäjiä. Ohjelmiston käyttöliittymä on yksiselitteinen ja muistuttaa useita mallinnus- tai arkkitehtiohjelmistoja. Ohjelmiston pääikkuna on jaettu neljään näkymään: camera (3d), top (2d), front (2d) ja side (2d). Ikkunat ovat täysin muokattavissa halutuiksi helpon muokkausvalikon kautta. Vasempaan reunaan on sijoitettu painikkeita useimmista ohjelmiston tärkeimmistä toiminnoista. Oikeasta reunasta löytyy materiaaliasetukset (material settings), näkymäasetukset (visgroups) ja kategorian muokkausominaisuudet (category settings). Ylärivin valikko sisältää muun muassa näkymäverkon (grid), koon muokkausasetukset, kartan rakennusvalikot sekä näkymäasetuksia.

4.2 BSP-geometria

Valve Software käyttää peleissään vielä Quake-ajoilta tunnettua BSP-teknologiaa. BSP on lyhenne sanoista ”Binary Space Partition”. BSP:tä käytetään pääosin kenttien rakentamisessa ja se on yksi niistä suunnittelutavoista, jolla voidaan hallita kentän optimointia, luoda kolmiulotteisia tiloja ja pelaajalle määritettävien rajoitteiden suunnittelua. (VDC, Valve Software)

Valve Hammer Editor -ohjelmistossa kaikki pinnat tehdään BSP-teknologialla ja siten tämän omaisuuden hallitseminen on erittäin tärkeää osata. Nämä tasot ja pinnat tunnetaan myös nimellä ”brush”. BSP-geometria rakennetaan tarkalle neliskulmaiselle verkolle. Kyseinen verkko auttaa eri tasojen yhdistämistä ja rakenteen selkeyden kannalta siisteyden ylläpitoa. Tasojen pinnoille voi määritellä materiaaleja ja koko BSP-tasolle voi määrittää tarvittaessa erilaisia ominaisuuksia. Jotkut BSP-tasolle määritetyt tekstuurit määrittävät jo kyseiselle pinnalle/tasolle erilaisia ominaisuuksia kuten ”triggerit” ja alueet, jotka laukaisevat erilaisia tapahtumia.

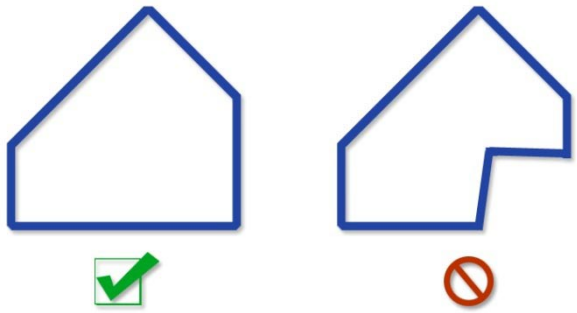
Kuva 5: Vasemmalla: Pelkkä BSP – Oikealla: BSP + mallit ja maasto. Ei valaistusta.

Kuva 5 -kuvasta näkyy, kuinka ”Fastlane” -kartassa suurin osa näkyvistä pinnoista on täysin rakennettu hyödyntäen BSP-geometriaa. Näin luotiin talojen muodot, tasanteet, katot ja näkyvät sekä mielenkiintoa tuovat rakennusten korkeuserot.

BSP-geometriaa rakentaessa on myös hyvä pitää mielessä jatkuvasti se, mikä on lopullinen kentän navigaatiorakenne, jotta pelimekaaniset elementit eivät kärsisi pelkästä visuaalisesta ilmeestä. Fastlane-kenttää rakentaessa hyödynnettiin aluksi täysin BSP-geometriaan pohjautuvaa pohjakuvaa. Tämän pohjakuvan avulla tehtiin muutokset

navigaatiokarttaan. Kun BSP-geometrian rakentaa huolellisesti alusta lähtien, on kartan seuraavien vaiheiden hallinnointi huomattavasti helpompaa.

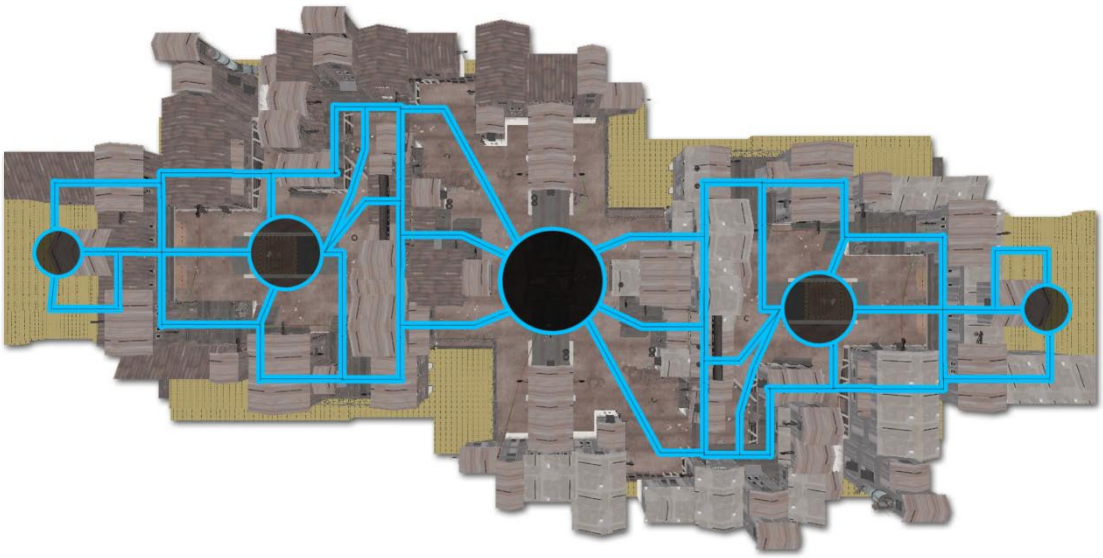
BSP-geometrian ainoa huono puoli on se, että päällekkäin ja toisiaan koskettavat pinnat leikkaavat tasoja, joka vaikuttaa mm. optimointi- ja suorituskykyongelmiin. Tämän ongelman voi kiertää käyttämällä BSP-pohjaisia entiteettejä. Entiteetit eivät vaikuta tasoja laskettaessa BSP:n hajontaan ja leikkaukseen. BSP-pohjaisessa geometriassa ei myöskään saa käyttää pintoja, jotka leikkaavat yhden ”brushin” sisällä toisiaan. (Kuva 6)



Kuva 6: Geometrian linjat eivät saa leikata saman ”brushin” sisällä toisiaan.

BSP-geometrian rakentaminen kenttää suunnitellessa on ehkä prosessin helpoin osuus, ei tosin haasteeton. Fastlane-kenttää rakentaessa huomattiin, että geometrian täsmällisyys ja järjestelmällisyys tulisi aina olla etusijalla. Rakennusurakkaa helpotti myös se, että kenttä kaksi puolta olivat toistensa peilikuvia. Käytännössä siis riitti, että toisen puolen geometria rakennettiin ensin valmiiksi ja tämän jälkeen siirrettiin puolikkaan peilikuvaksi vastakkaiselle puolelle. Tämä ei aina ole paras mahdollinen lähestymistapa kenttää rakentaessa, ja siten voikin tuottaa ongelmia kentän tasapainottamisessa ja optimoinnissa.

Fastlane-kartta sisältää yhdellä puolella kolme päänavigaatioaluetta, jotka yhdistää eri määrä pieniä reittejä rakennelmien sisällä tai ulkopuolella. Tasapainotuksen kannalta oli tärkeää, että jokaiselle alueelle pääsee ainakin kahdesta tai kolmesta eri suunnasta. (Kuva 7)

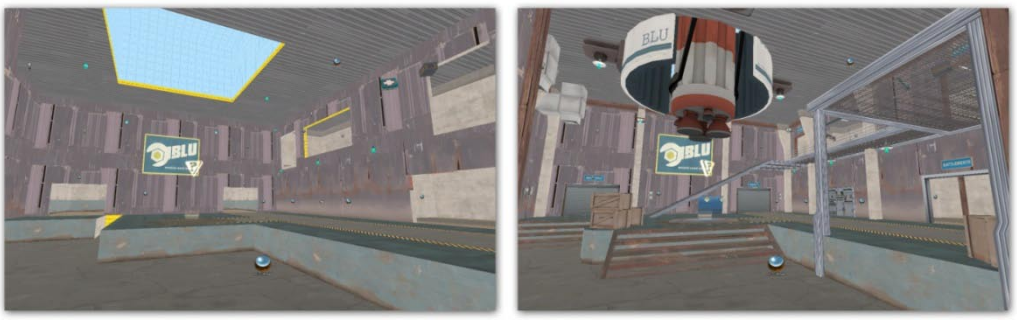


Kuva 7: BSP-geometrian muodoista syntyvä kartan ulkoasu. Pääreitit on merkitty sinisellä.

4.3 3D-mallien käyttö

Pelimoottorien kehittyessä 3D-mallien käyttö peleissä on yleistynyt räjähdysmäisesti. Jotkut pelimoottorit keskittyvät ainoastaan 3D-mallien (mesh, prop jne.) käyttöön. Vaikka uusimmissakin moottoreissa on käytössä erilaisia BSP-tekniikoita, jää niiden käyttö kuitenkin paljon vähäisemmälle. 3D-mallit antavat artisteille ja suunnittelijoille paljon vapaammat kädet, sillä monipuolisilla malllinnusohjelmilla saa aikaiseksi paljon yksityiskohtaisempia malleja kuin tasosuunnitteluohjelmalla.

3D-mallit ovatkin tässä tapauksessa kartan viimeinen silaus, jotta työstä saataisiin mahdollisimman näyttävä ja rikas. Fastlane-kentässä on käytetty yhteensä noin 1900 3D-mallia rikastuttamaan ympäristöä. Mallien joukosta löytyy mm. ikkunoita, rappusia, rakennuselementtejä ja pieniä yksityiskohtia kuten lamppuja, johtoja, kaiteita jne. (Kuva 8)

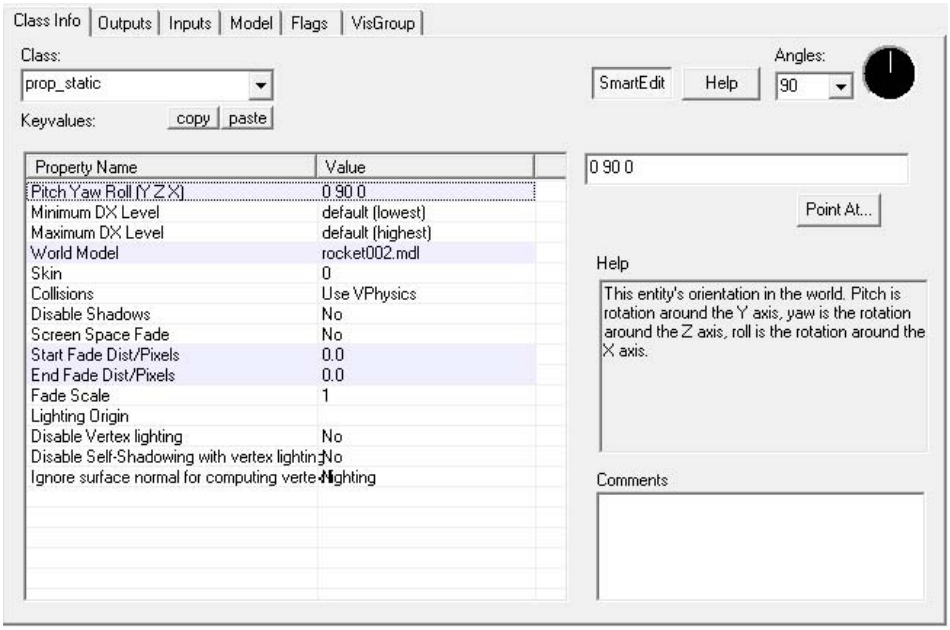


Kuva 8: Tila ilman 3D-malleja. Oikealla: Tila 3D-mallien kanssa. Ei valaistusta.

3D-malleja voi tehdä millä vaan mallinnussovelluksella. Näistä kätevin on kuitenkin XSI-sovellus, joka sisältää helppokäyttöiset työkalut mallin siirtämiseen suoraan pelimoottorin käytettäväksi.

Source Enginessä on tuki kolmenlaisille 3D-malleille: Staattiset, Dynaamiset ja fysiikanmallinnusta sisältävät mallit. Staattiset mallit ovat maailmaan asetettuina liikkumattomia malleja eikä niitä voi manipuloida pelistä käsin. Dynaamiset mallit mahdollistavat niiden liikuttamisen tai animoinnin. Fyysisesti rakennetut mallit mahdollistavat tehokkaan fysiikkamoottorin käytön mallin kanssa. Mallia voi manipuloida, siirrellä ja heitellä pelimoottorin sisällä.

Mallien käyttö pelissä on tehty helpoksi. Voit siirrellä, pyörittää ja asettaa mallin suoraan suunnitteluohjelmassa paikalleen. Malleilla on myös monia ominaisuuksia, joita on mahdollista muokata suoraan tasosuunnitteluohjelmasta käsin. (Kuva 9)



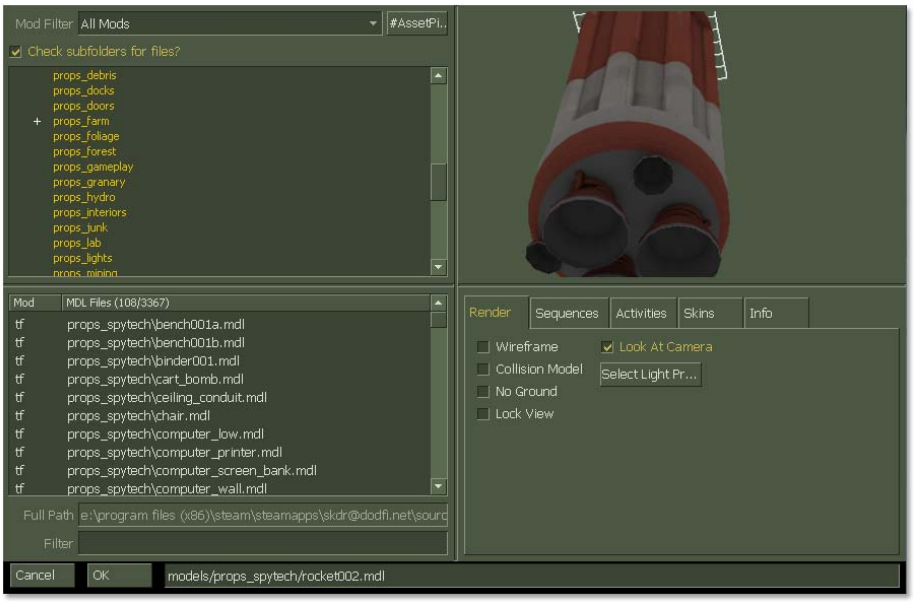
Kuva 9: 3D-mallin asetusikkuna Valve Hammer Editorissa.

Mallin sijaintia, pyöritystä jne. voi säädellä numeerisesti, jos halutaan tarkat arvot. Tässä tapauksessa rakettimoottorin alaosa on asetettu karttaan ja sille on määritelty z-akselilla pyöritysarvoksi 90 astetta. ”Skin” -asetuksella voidaan määritellä toissijainen tekstuuri mallille sillä ehdolla, että se on myös asetettu mallia tehtäessä mallin teknisten tietojen sekaan.

Mallien mukana tulee myös ”collision box”, joka määrittelee käytännössä raja,t johon pelaaja pelin sisällä törmää. Asetuksen avulla voidaan määrittää mallista rajat kokonaan pois käytöstä, jos siihen on tarve. Normaalisti mallit tuottavat myös valojen kohdistuksen ansiosta varjoja ja tämän asetuksen voi poistaa kohdasta ”Disable Shadows”.

Tärkein osa mallin asetuksia on kuitenkin ”Screen Fade”, jonka avulla määritellään se etäisyys, jolloin malli poistuu näkyvistä. Asetuksen avulla voidaan tehokkaasti määritellä mallin näkyvyyden rajat.

Source Enginen mukana tulee myös tehokas mallien tarkastelusovellus. Tämän sovelluksen avulla on mahdollista tarkastella kaikkia saatavilla olevia malleja reaaliajassa oikeassa valaistuksessa ja se näyttää myös mallien pintamateriaalien oikeat ominaisuudet. Samaisella sovelluksella voi myös tarkastella malleja ”wireframe” –tilassa ilman valaistusta, eri tekstuureilla ja malleille määriteltyjä animaatioita. ”Wireframe” –tila näyttää mallin polygoniverkoston ja rakenteen selkeämmällä tavalla. Ohjelmisto pystyy myös määrittelemään, minkä modifikaation malleja tarkastellaan kätevässä kansiorakennenäkymässä. (Kuva 10)



Kuva 10: Valve Hammer Editorin mallien tarkasteluohjelma.

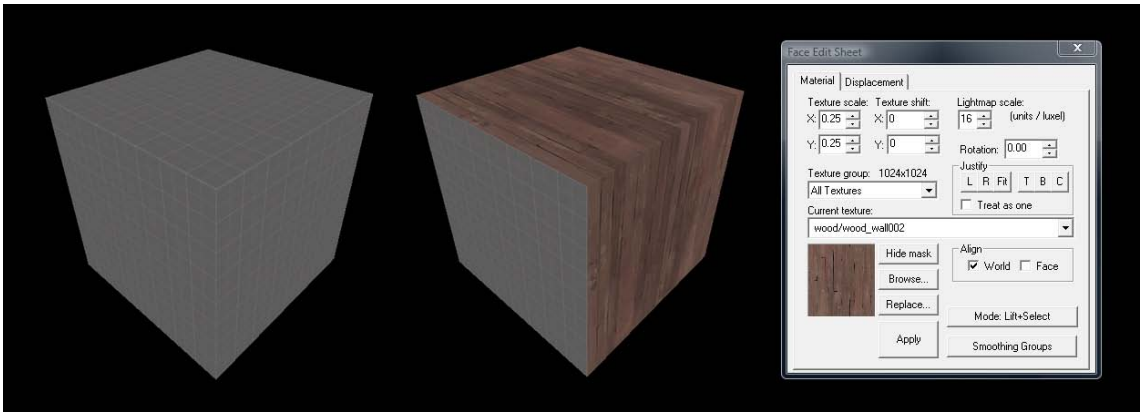
Fastlane-kentässä mallien käyttö oli jo alkuvaiheessa täysin mukana. On tärkeää, että heti kehityksen alkaessa on tiedossa, minkälaista visuaalista ilmettä suunnittelutyöstä haetaan. Valve Softwaren tarjoamat valmiit mallit ovat jaettu yleensä teeman mukaan samoihin kansiorakenteisiin, joten samankaltaisten mallien löytäminen on tehty helpoksi. Kehitysvaiheen alkuvaiheilla on hyvä asettaa malleille myös ”Screen Fade” -

ominaisuudet jotta optimointi suunnittelun loppuvaiheessa olisi siltä osin valmista. Fastlane-kenttää rakentaessa ”Screen Fade” -asetusten määrittäminen jäi yhdeksi viimeisistä prosesseista ja täten hidasti hieman suunnitelman valmistumista.

4.4 Pintamateriaalit

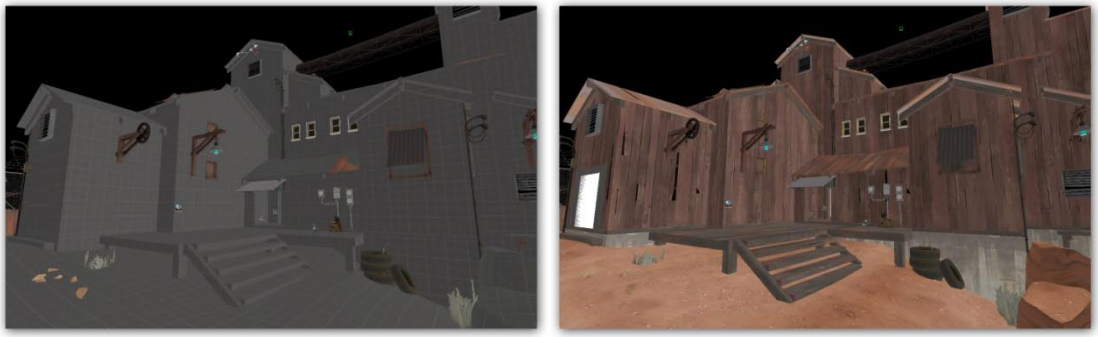
Pintamateriaali määrittää kaksiulotteisen tason tekstuuritiedot. Tämä pinta sisältää tiedon mm. pinnan luovasta tekstuurista (kaksiuloitteisesta kuvasta), fyysisistä ominaisuuksista ja valon vaikutuksesta. Pelimoottorin sisällä pintamateriaalit antavat eri pinnoille visuaalisen ilmeen ja on tärkeää, että työstettävässä kartassa on käytetty samoja piirteitä omaavia pintamateriaaleja. Pintamateriaali on siis kaksiulotteinen kuva pinnasta, jolla visualisoidaan mm. rakenteen materiaali.

Source Enginessä on monipuoliset pintamateriaalityökalut. Materiaalia voi suurentaa, venyttää, pyörittää, siirtää ja lukita paikalleen jonkun neljän reunan mukaan. ”Fastlane” -kartassa käytetyt materiaalit ovat peräisin samasta materiaalipaketista, jota on käytetty mm. Valve Softwaren virallisessa ”Badlands” tasossa. Paketti sisältää kaksi eri kokoelmaa tekstuureja värimäärittelyineen molemmille joukkueille.



Kuva 11: Valve Hammer Editorin pintamateriaalien muokkaus.

Materiaaleja voi määritellä niin, että jokaiselle ”brushin” pinnalle voi asettaa täysin oman materiaalin. Tällä ominaisuudella ei ole teknisiä rajoitteita. Ainoa rajoite on vain pintojen kokonaismäärässä. (Kuva 11)



Kuva 12: Vas.: ”Fastlane” –kartta ilman tekstuureja – Oik.: Lopullisilla tekstuureilla

”Team Fortress 2” -pelissä on käytetty suurimmilta osin käsinmaalattuja pintamateriaaleja. (Kuva 12) Tämä antaa pelille uniikin ilmeen ja näin erottuu massasta. Molempien joukkueiden graafinen ulkoasu on suunniteltu selkeästi ja niin, että kaksi eri joukkuetta ovat täysin erotettavissa toisistaan. Pelissä on kaksi joukkuetta: punaiset ja siniset. Sinisen joukkueen pintamateriaalit ovat teollisuushenkisiä, sinisen kylmiä, kun taas punaisen joukkueen materiaalit muistuttavat farmeja, puutaloja ja lämpimiä värisävyjä (Kuva 13). Sinisen puolen rakennusten pinnat ovat myös yleensä horisontaalisti vaakatasossa, kun taas punaisen joukkueen rakennusten muoto on yleensä esitetty vinoilla pinnoilla.

Kuva 13: Värimäärittelyt, graafinen ilme ja tyyli. (NPAR07, Illustrative Rendering, Valve Software)

Valve Softwaren työkaluissa materiaali tarkoittaa samanaikaisesti montaa eri asiaa. Pelimoottorissa käytettävä materiaalitiedosto (VMT, Valve Material Type) sisältää määritykset viidelle erilaiselle materiaalitiedostolle:

1. Tekstuurin nimi
2. Fyysiset materiaalitiedot
3. ”Shader” -tekniikkaan perustuvat tiedot
4. ”Fallback” -materiaalitiedot
5. ”Proxy” -materiaalitiedot.

4.5 Valaistus

Source Engine käyttää radiositeettiarvoihin pohjautuvaa valaistusjärjestelmää. Tämän järjestelmän avulla on helppo luoda ympäristöön mukautuva ”ambient” -valaistus sekä tarkkoja ”valokarttoihin” perustuvia valaistustasoja. Source Engine tukee myös täysin dynaamisia valonlähteitä sekä partikkeleihin perustuvia lähteitä. (Advanced Lighting, Valve Software, VDC 2008)

Dynaaminen valonlähde

Näkymätön lähde, joka tuottaa maailmassa vaihtuvan sekä liikkuvan valon. Sen sijaintia voi muuttaa ja sen voi sijoittaa osoittamaan liikkuvia kohteita päin, luoden dynaamisen varjon kohteesta riippumatta. Dynaamisen valon arvoja lasketaan jatkuvasti ja täten se tekee siitä raskaan käyttää. Dynaamista valoa suositellaan käytettävän vain ahtaimmissa sisätiloissa, joissa pelimoottorin optimointimenetelmät tulevat paremmin tarpeeseen.

Hohtavat tekstuurit

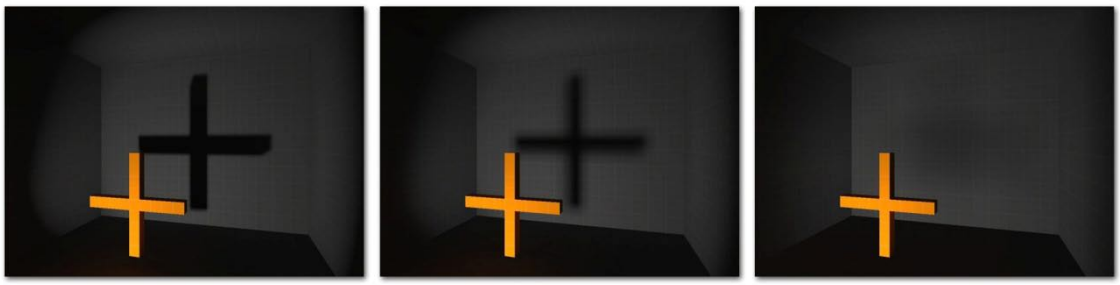
Suunnittelija voi luoda tekstuureja pinnoille tai malleille, jotka hohtavat valoa luoden illuusion kirkkaasta valosta. Tämän valonlähteen käyttö vaatii muokkausta Valven VMT-tiedostojen arvomäärityksiin. Tämä valonlähde on optimoinnin kannalta tehokkain, sillä se ei vie yhtään resursseja entiteettien käsittelyssä.

Partikkelivalo

Partikkelivalo on samassa luokassa normaalin valon kanssa, mutta tässä tapauksessa valaisee vain ”env_smokestack” entiteetin luomia partikkeleita.

Valokartat

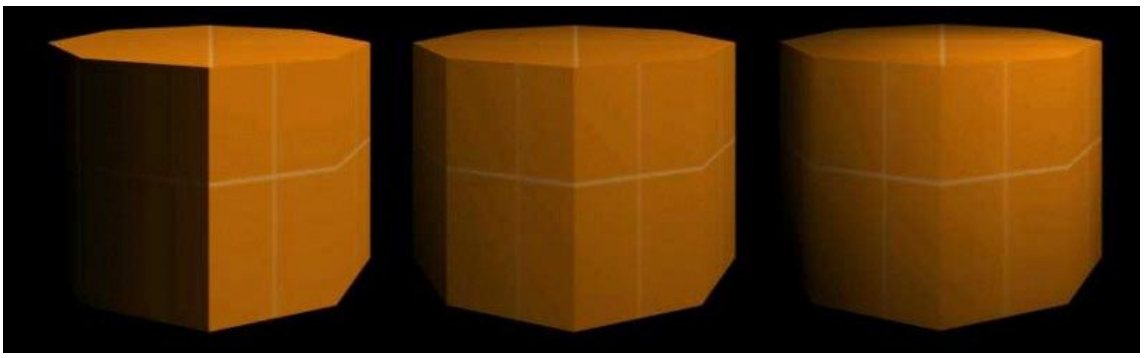
Valokartat määrittelevät sen, kuinka tiheä on pinnalle piirrettävä verkosto, joka määrittää varjojen tarkkuuden. Mitä pienempi verkko, sitä korkealaatuisempi varjo (Kuva 14).



Kuva 14: Valokarttojen arvot: 4, 16 ja 64 (units) [VDC, Valve Software]

Pintojen tasaisuusarvot (Smoothing Groups)

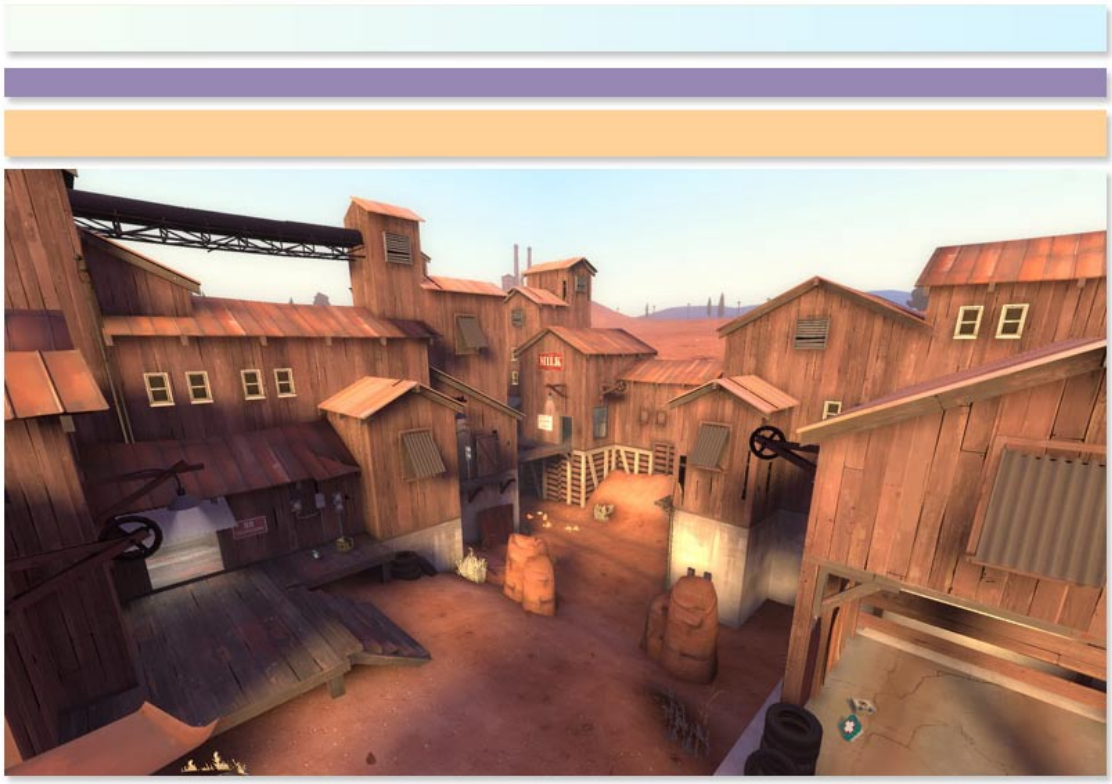
Source Enginessä on mahdollisuus muokata BSP ja 3D-mallien pintojen tasaisuusarvoja. Tämän arvon määrittely auttaa yksinkertaisten pyöreiden pintojen valaistuksen pehmentämisessä. Haluttavat pyöristettävät pinnat valitaan ja määritellään jokaiselle eri pintaryhmälle oma hallintanumero. Tämän jälkeen pelimoottori valaisee pinnan tasaisesti määrittelyjen mukaisesti (Kuva 15).



Kuva 15: Tasaisuusarvojen asettaminen tuottaa huomattavan muutoksen. (VDC, Valve Software)

”Fastlane” -kentässä käytettiin neljää erityyppistä valaistustekniikkaa. Auringonvalo määrittyy ”light_environment” entiteetin avulla, johon määritellään valon väri, kirkkaus, varjojen väri ja varjojen tummuus. Muihin määrityksiin sisältyvät mm. auringon suunta ja korkeus sekä auringon valon levittäytyminen eri pinnoille. Source Engine tukee myös HDR-tekniikkaa (High Dynamic Range), joka mahdollistaa valon ”exposure”- sekä dynaamisen valonvaihtelun arvojen määrittämisen.

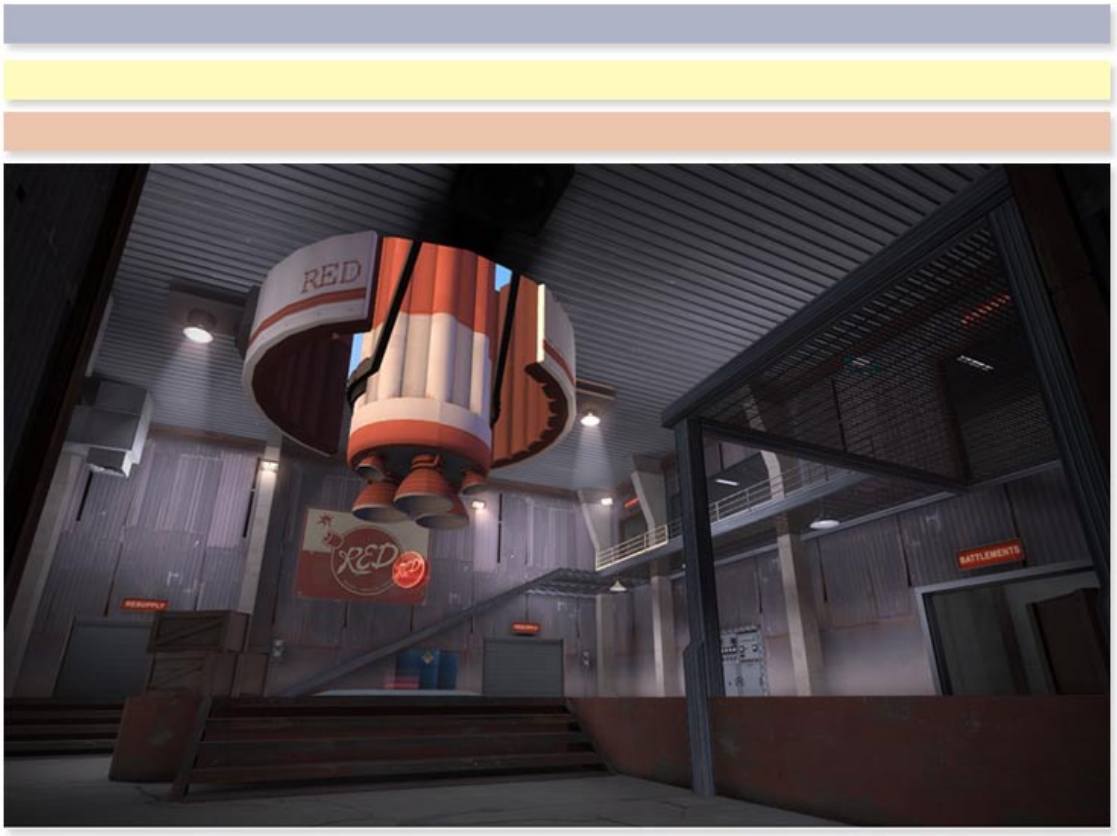
”Fastlane” -kentän valaistusarvot kävivät läpi monta eri testausvaihetta, kunnes olin tyytyväinen lopputulokseen. Valon suunta on -25 astetta (-90 astetta = suoraan ylhäältä alas), valo määritettiin 30 asteen kulmaan origosta ja valon värinä käytettiin hillittyä oranssin sävyä. Varjon väriksi valittiin haalea violetti sävy. Nämä arvot tasapainotettiin taivaan hennolla sinisellä vastavärillä. Karttaan lisättiin viimeiseksi vielä valaistuarvojen mukainen sumuefekti, joka tuo tilan syvyysmäärittelyt paremmin esille. Myös taustalla näkyvät, tilan illusioota luovat maisemat mukailevat värisävymäärittelyitä (Kuva 16).



Kuva 16: Kartan valaistusarvot tasapainottavat toisiaan.

”Fastlane” -kentän sisätiloissa käytettiin selvästi suurempaa kontrastia valaistuarvojen välillä. Vaikka auringonvaloa saattoikin päästä sisätiloihin, oli tärkeää, että valojen ja varjojen välille syntyisi hieman kevyempi rajaus. Sisätiloissa hyödynnettiin materiaalivalojen lisäksi kohdistettuja (spotlight) valoja, sekä normaaleja, himmeitä valoentiteette-

jä. Sisätiloissa on myös tärkeää pitää silmällä sitä, kuinka valon arvot vaikuttavat tilojen kaukaisimpiin nurkkiin. Sisätilojen valaistus onkin yksi suunnitteluvaiheen haastavammista tehtävistä. (Kuva 17)



Kuva 17: Kartan sisätilojen valaistuarvot. Näkyvien värien kirkkaus vaihtelee kohteen mukaan.

4.6 Entiteetit

Entiteeteillä tarkoitetaan maailmaan sijoitettavia pisteitä/alueita, jotka määrittävät jonkun pinnan, ulkoasullisen tai pelimekaanisen ominaisuuden. Kaikki tasoon aseteltavat 3D-mallit, valaistus- ja pelimekaaniset arvot ovat entiteettejä. Source Engine käyttää kolmea erilaista entiteettiä:

1. ”Brush”-entiteetti
2. ”Point”-entiteetti
3. ”Node”-entiteetti

”Brush”-entiteetti määrittelee kolmiulotteisessa tilassa kolmiulotteiselle kappaleelle jonkin arvon, joka vaikuttaa kartan toimintaan lukuisilla eri tavoilla. Suunnittelija voi mm.

asettaa tasoon ”brush”-entiteetin, joka estää pääsyn tietyille alueille tai entiteetin, joka määrittää valitun tapahtumaketjun aktivoinnin tietyn 3D-kappaleen sisällä (kutsutaan nimellä ”trigger”). Suurin osa optimoinnin kannalta tärkeistä entiteeteistä ovat ”brush”-entiteettejä, joiden avulla pystytään mm. määrittämään tilojen optimointialueita.

”Point”-entiteetit ovat maailmaan asetettavia pisteen kokoisia entiteettejä, joilla ei ole itsessään alueellista vaikutusta. Nämä entiteetit ovat yleensä näkymättömiä ja määrittelevät mm. pelin tekoälyn toimintaa, valaistusta, 3D-malleja ja tasojen pelimekaanisia arvoja.

”Node”-entiteetit ovat myös maailmaan asetettavia pisteen kokoisia entiteettejä, jotka linkittyvät toisiinsa luoden verkon, jota kutsutaan nimellä ”nodegraph”. Yleensä ”nodet” määrittävät tekoälyn liikettä tason sisällä. Moninpelikentissä ”node”-entiteettejä ei käytetä, koska kyseinen pelimekaniikka ei tarvitse tekoälytoimintaa suoriutuakseen tehtävästä. ”Node”-entiteettejä ovat mm. seuraavat entiteetit:

1. Info_node – Määrittelee alueet, joilla tekoäly voi liikkua tason sisällä.
2. Info_node_hint – Määrittelee alueen, jonka sisällä tekoäly voi kiivetä eri tasoille.
3. Info_node_link_controller – Ohjaa node-linkkien välistä toimintaa.

4.7 Optimointi

Source Enginessä optimointi on tärkeä osa suunnitteluvaihetta ja se on otettava huomioon heti projektin alkuvaiheessa ja on avain sulavaan pelin kulkuun. Source Enginessä optimointi voidaan jakaa viiteen eri osa-alueeseen seuraavasti:

1. Vuodot (”leaks”)
2. Näkyvyys
3. Fysiikat
4. Materiaalit
5. Valaistus

Vuodot (”leaks”)

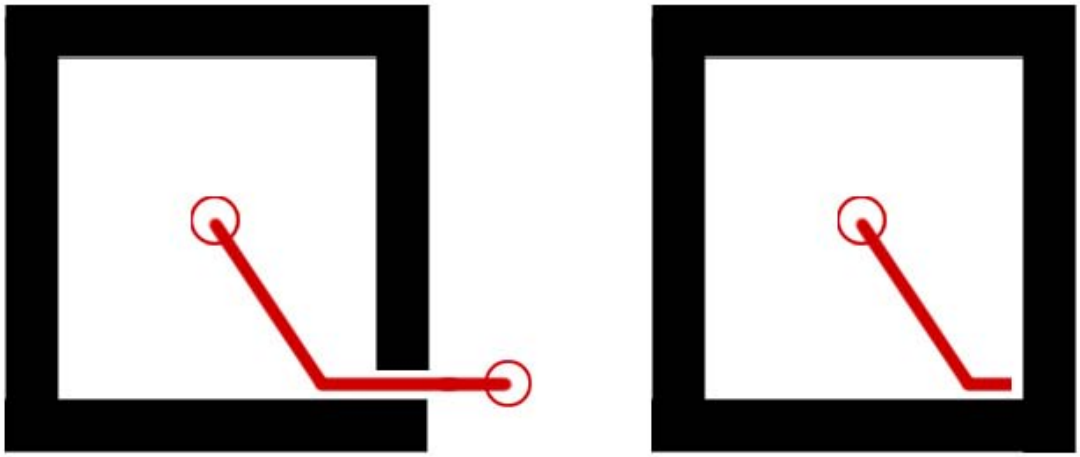
Kaikkien Source Enginellä rakennettujen karttojen tulee olla suljettuja. Vuodoilla tarkoitetaan tasosta löytyviä aukkoja, joista valo karkaa tyhjyyteen. Vuodot aiheuttavat kar-

tan rakentamisen yhteydessä paljon ongelmia, jotka voivat estää suunnitteluprosessin edistymistä.

Kartasta löytyvä vuoto voi luoda seuraavia ongelmia:

1. VBPS-prosessi ilmoittaa vuodosta eikä VVIS-prosessi laske näkyvyysarvoja.
2. Kun VVIS-prosessi ei laske näkyvyysarvoja, niin VRAD-prosessi ei myöskään laske valaistusarvoja.
3. Kun VVIS-näkyvyysarvoja ei lasketa, luulee pelimoottori, että koko kartta on piirrettävä kerrallaan. Tämä tuottaa suuria suorituskykyongelmia.

Vuodon löytää helpoiten lataamalla ”pointfile”-tiedoston, joka sisältää koordinaatit sille alueelle, mistä vuoto on alun perin lähtöisin. Myös kartan ulkopuolella olevat objektit, kuten ”point”-entiteetit, voivat aiheuttaa vuodon. (Kuva 18)



Kuva 18: ”Pointfile”-tiedosto ilmoittaa vuodon alku- ja loppupään helpottaen vuodon korjausta.

Näkyvyyden rajoittaminen on tehokkain optimointikeino

Näkyvyyden rajoittaminen on paras tapa optimoida kartan toimivuutta ja suorituskykyä. Valitettavasti jokaisen objektin omakohtainen näkyvyyden tarkastelu pelimoottorin silmistä on raskaampaa suorituskyvylle kuin niiden kaikkien piirtäminen samanaikaisesti, joten tässä tapauksessa on tehtävä kompromisseja.

”VIS” on ohjelmisto, joka laskee tason näkyvyysarvot. Laskenta-aika on yleensä muutaman minuutin, vaikka kyseessä olisikin monimutkainen kenttä. Jos laskenta-aika on enemmän kuin normaalisti, kannattaa huomioida seuraavia asioita: (Visibility Optimization, VDC)

1. On tärkeää käyttää ”detail brusheja”, jotka eivät vaikuta laskentaprosessiin
2. Kaikkien tasojen on oltava verkon (grid) sisällä.
3. On tärkeää käyttää vain yksinkertaisia ”brush”-tasoja.
4. Isojen tilojen luomista vältetään.

Fysiikat

Fyysisiä tietoja sisältävät mallit ovat nykypäivää pelimoottoreissa ja onkin harvinaista, että niitä ei olisi asetettu lainkaan. Moninpelikentissä fysiikkamallinnetut kappaleet voidaan asettaa käyttämään niille tarkoitettu entiteettiä nimeltä ”props_physics_multiplayer”. Tämän entiteetin avulla voidaan myös määritellä, mitkä fyysiset ominaisuudet lasketaan pelaajan ja palvelimen näkökulmasta.

Materiaalit

On selvää, että kaikkien käyttäjien koneet eivät pyöritä tasoja samalla tarkkuudella kuin muiden ja siten optimoinnissa on otettava huomioon laaja määrä eri materiaalioptimointitapoja. Materiaalien ”shader”-arvot ovat näistä kaikkein arvokkaimpia suorituskyvyn kannalta. Valven resursseissa kaikille materiaaleille on määritelty ”fallback”-arvot, jotka ovat tietokoneen kokoonpanon mukaan säädettävissä ilman, että käyttäjä huomaa eroa. Jos materiaaleja tekee itse, on ”fallback”-arvot määriteltävä erikseen. ”Fallback”-arvot ovat suoraan verrannollisia DirectX-rajapinnan eri versionumeroihin.

On olemassa myös erikoismateriaaleja, joiden käyttöä suositellaan suunnitteluvaiheen alkuvaiheesta lähtien. (Valve Software, VDC 2008). ”Nodraw” -materiaalin tarkoitus on luoda pintoja, jotka eivät piirry pelimoottorin sisällä ollenkaan. Käytännössä kaikki pinnat, jotka eivät ole pelaajan näköpiirissä, kannattaa luoda ”nodraw”-materiaalilla. ”Fastlane” -kentän kaikki ”brush” -tasot luotiin aluksi ”nodraw” -materiaalilla ja vasta tämän jälkeen teksturoitiin oikeilla materiaaleilla.

Valaistus

Source Enginen staattiset valaistusarvot lasketaan valmiiksi käyttöä varten ”valokartoille”. Tämä mahdollistaa käytännössä rajattoman valojen määrän karttaa rakennettaessa. Valaistusarvojen säätäminen on hyvä tapa optimoida karttaa entisestään.

1. Valokarttojen verkon tiheyttä voidaan säätää. Mitä pienempi arvo on, sitä tiheämpi verkko. Siksi suositellaankin, että laajoille pinnoille asetetaan isompi valokartan tiheysarvo kuin pienille. Sisätiloissa voidaan käyttää huoletta pienempää arvoa, koska näkyvyysarvot on laskettu tehokkaammin.
2. Valojen nimeämistä kannattaa välttää, koska laskentaohjelmistot luulevat yleensä, että nimetyt valot on tarkoitettu otettavaksi pois käytöstä jossain vaiheessa ja siten laskee valolle arvon kaksinkertaisesti.

Dynaamiset valot ovat suorituskyvyn kannalta kaikkein raskaimpia. On tärkeää, että dynaamisten valojen käyttöä tarkkaillaan jo alusta alkaen. On myös tärkeää tarkastella, mille alueelle dynaaminen valo osoittaa. Mitä monimutkaisempi geometria ja mitä enemmän 3D-malleja on valon osoittamalla alueella, sitä raskaampaa valon käytöstä tulee. Dynaamisia valoja voidaan käyttää ns. ”low-point” -alueilla, joissa yleensä riittää budjetti hieman monimutkaisemmallekin valaistukselle.

4.8 Testaus

Kartan testauksella tarkoitetaan prosessia, joka käydään läpi jo varhaisessa vaiheessa suunnittelua. Testauksen tarkoituksena on pelata suunniteltua kenttää noin kahdenkymmenen hengen voimin ja etsiä mahdollisia ongelmia ja parannuksia. Testaajia kehotetaan antamaan kritiikkiä ja parannusehdotuksia kartan toimivuutta ajatellen. Tason aikaisessa vaiheessa testaaminen helpottaa myös muutosten tekemistä, sillä yleensä aikaisessa vaiheessa kartan visuaalinen ilme ei ole täysin vielä valmis ja suunnittelijalla on varaa tehdä muutoksia ilman, että työtä heitetään turhaan hukkaan.

”Fastlane” -kentän suunnitteluvaiheessa testattiin pelimekaanisia elementtejä jo hyvissä ajoin, jotta voitiin saavuttaa mahdollisimman tasapainottunut ja miellyttävä pelikokemus. Kävi ilmi, että jokaisen hahmoluokan on oltava tasapainossa kaikkien eri reittien välillä. Tähän vaikuttaa mm. ”control point” -pisteiden asettaminen kartalla, tilojen

koko ja käytävien rakenteet. Tason kehitysvaiheessa tehtiinkin suuria muutoksia keskialueen geometriaan, jotka vaikuttivat suuresti kartan lopulliseen toimivuuteen.

5 TASOSUUNNITTELUN TULEVAISUUS

CryEngine 2 -pelimoottorilla luotu ympäristö.

Peliteknologia ja pelisuunnittelu jatkavat kehittymistään ja tasosuunnittelijoiden rooli tämän muutoksen keskellä on suuri. Tasosuunnittelijoiden yhteistyö muiden osa-alueiden osaajien kanssa yritysmaailmassa tulee olemaan keskeisessä asemassa entistä enemmän. Ohjelmoijat tulevat määrittelemään suurimmilta osin sen, mitä suunnittelija voi saatavilla olevilla työkaluilla tehdä. (C. Bleszinski, 2000.)

Suunnitteluohjelmat tulevat siirtymään monimutkaisiin ratkaisuihin ja muistuttavatkin jatkossa monipuolisten mallinnusohjelmien tapaan kokonaista ohjelmistopakettia. Monet kehittäjät ovatkin jo luovuttaneet omien suunnitteluohjelmien luomisen suhteen ja siirtyneet laajojen mallinnusohjelmien pariin. Näihin ohjelmiin on sitten lisätty suunnitteluun tarvittavat lisäosat. On myös huomattava, että yhä suurempi ja suurempi osa suunnitteluohjelmista painottaa tason muokkauksen reaaliaikaisuuden tärkeyttä. Siten suunnittelijoille annetaan entistä vapaammat kädet ja nopeammat työtavat tuottaa

sisältöä projektiin kuin projektiin. Monimutkaisen mallinnusohjelman hallitsemista tullaankin vaatimaan myös tasosuunnittelijoilta jatkossa. (C. Bleszinski, 2000.)

Ilman muiden osa-alueiden osaajia ei tasosuunnittelijan työstä tehdä sen helpompaa. On siis tärkeää muistaa, että pelisuunnittelu yrityksessä ei ole yhden miehen työ. Mukaan tarvitaan suuri määrä taitavia artisteja ja ohjelmoijia. Ennen tasosuunnittelija teki kaiken, mutta nykyään alati muuttuvat roolijaot painottavat sitä, että vaikka suunnittelijan rooli onkin kapeampi. Tämä on silti päivä päivältä yhä tärkeämpi asia, jotta voitaisiin saavuttaa tavoitteet ja luoda jotain sellaista mistä heijastuu kaikkien ahkeran työn tulos. (C. Bleszinski, 2000.)

6 YHTEENVETO

Opinnäytetyön tarkoituksena oli tutkia tasosuunnittelun tekniseen toteutukseen liittyviä haasteita, ratkaista ongelmia ja samalla perehtyä pelimoottorin teknisiin ominaisuuksiin. Näin luotiin kokonaisuus, jonka lukemisesta hyötyvät kaikki alasta kiinnostuneet. Projekti osoittautui haastavaksi, mutta lopulta onnistuneeksi ja palkitsevaksi. Projektin prosessi käytiin läpi selkeästi eri vaiheiden mukaisesti.

Mielenkiintoisin osuus koko projektissa on selvästi geometria ja sen valaiseminen, ja tämä onnistuikin ongelmitta. Projektia suunniteltiin aluksi yksinkertaisilla piirroksilla paperille ja tämän pohjalta rakennettiin yksinkertainen geometria käyttäen BSP-tasoja. Näkymä valaistiin yksinkertaisilla, väliaikaisilla valaistusarvoilla, jotta saatiin luotua syvyyttä tasojen välille. Suunnittelutyön lopussa kartta valaistiin käyttäen oikeita arvoja ja tekniikoita.

Projekti oli kokonaisuudessaan todella mielenkiintoinen, sillä aikaisempi kokemus Valven työkaluilla edesauttoi projektin kulkua huomattavasti. Toimivan moninpelikentän rakentaminen vaatii ideoiden iteroimista ja prosessin tutkimista tarkemmin kuin monet luulevatkaan. Lisähaasteen projektiin toi testaajien antama palaute, jonka avulla parannettiin kartan pelattavuuden hiomista entistäkin parempaan suuntaan. Lopullisesta kentän toimivuudesta ja pelattavuudesta ei voitu olla varmoja ennen kuin kentän kaikki ominaisuudet ja sisältö oli saatu rakennettua.

Kokonaisuudessaan projektiin kului yhteensä noin kaksi kuukautta ja teki projektista ajallisesti tehokkaimman projektin suunnitteluhistoriassani.

Fastlane-tason suunnittelu alkoi alkukesästä vuonna 2008, jolloin olin saanut juuri valmiiksi edellisen projektini. Edellisen projektin siivittämänä oli helppo siirtyä kehittämään uusia ideoita vanhojen tilalle. Source Engine oli jo ennestään siis tuttu alusta suunnittelun ohessa. Kaiken kaikkiaan projektista nousi esille seuraavat tekijät, jotka edesauttoivat projektin toimivuutta suunnitelmien kannalta:

1. Yhteistyö Valve Softwaren kanssa
2. Omaan työhön vaikuttamisen mahdollisuus
3. Sitoutuminen projektiin
4. Tavoitteet
5. Työn prioriteettijärjestykset

Source Engine pelimoottorina on oiva tapa päästä käsiksi tasosuunnitteluun. Se tarjoaa tehokkaat työkalut, hyvän dokumentaation, ison käyttäjäryhmän ja mikä parasta, paljon potentiaalisia pelaajia testaamaan suunniteltua tasoa. Projekti oli prosessina mieluinen eikä sen tekemisessä kohdattu juurikaan vastoinkäymisiä. Se antoi kokonaisuudessaan uusia kokemuksia ja tavoitteita seuraavien projektien suunnittelua ajatellen.

LÄHTEET

General Level Design Documentation. Valve Developer Community. [verkkodokumentti]
<http://developer.valvesoftware.com/wiki/Category:Level_Design>
(luettu 28.1.2009)

Team Fortress 2 Level Creation. Valve Developer Community. [verkkodokumentti]
<http://developer.valvesoftware.com/wiki/Team_Fortress_2_Level_Creation>
(luettu 28.1.2009)

Basic Map Construction. Valve Developer Community. [verkkodokumentti]
<http://developer.valvesoftware.com/wiki/TF2/Basic_Map_Construction>
(luettu 28.1.2009)

Team Fortress 2 Design Theory. Valve Developer Community. [verkkodokumentti]
<http://developer.valvesoftware.com/wiki/TF2_Design_Theory>
(luettu 28.1.2009)

Team Fortress 2 Designer's Reference. VDC. [verkkodokumentti]
<http://developer.valvesoftware.com/wiki/TF2/Team_Fortress_2_Mapper%27s_Reference>
(luettu 28.1.2009)

Entity Inputs & Outputs. Valve Developer Community. [verkkodokumentti]
<http://developer.valvesoftware.com/wiki/Inputs_and_Outputs>
(luettu 28.1.2009)

Optimization (Level Design). Valve Developer Community. [verkkodokumentti]
<http://developer.valvesoftware.com/wiki/Optimization_(level_design)>
(luettu 28.1.2009)

Gamasutra's History of Level Design. Gamasutra. [verkkodokumentti].
<http://www.gamasutra.com/view/feature/2674/educational_feature_a_history_and_.php>
(luettu 28.1.2009)

Wikipedia – Level Design. Wikipedia Organization. [verkkodokumentti]
<http://en.wikipedia.org/wiki/Level_design>
(luettu 28.1.2009)

The Art and Science of Level Design by Cliff Bleszinski. Cliff Bleszinski. [verkkodokumentti]
<http://www.cliffyb.com/rants/art-sci-ld.shtml>
(luettu 28.1.2009)

NPAR07 Illustrative Rendering in Team Fortress 2, Valve Software. [verkkodokumentti]
< http://valvesoftware.com/publications/2007/NPAR07_IllustrativeRenderingInTeamFortress2.pdf>
(luettu 23.2.2009)

LIITTEET

Liite 1: (DVD-levy) Fastlane-kartan esittelyvideo MOV-tiedostoformaatissa