Tampereen yliopisto

Joonas Kristo

UNITY-PELIMOOTTORIN LÄHESTYTTÄVYYS PELINKEHITYKSESSÄ

Informaatioteknologian ja viestinnän tiedekunta
Kandidaattitutkielma
Tammikuu 2023

TIIVISTELMÄ

Joonas Kristo: Unity-pelimoottorin lähestyttävyys pelinkehityksessä
Kandidaattitutkielma
Tampereen yliopisto
Tieto- ja sähkötekniikan kandidaatin tutkinto-ohjelma, Tietotekniikka
Tammikuu 2023

Nykypäivän näyttävät videopelit ja niiden suuret markkinat luovat kiinnostavan pohjan pelimoottoreiden lähestyttävyyden tarkempaan tutkimiseen pelinkehityksessä. Tämä tutkielma käsittelee Unity-pelimoottorin lähestyttävyyttä pelinkehityksessä aloittelevan pelinkehittäjän näkökulmasta. Tutkielmassa tutustutaan myös muihin suosituimpiin pelimoottoreihin, joihin Unityä verrataan. Tutkielman tavoitteena on auttaa aloittelevia pelinkehittäjiä päättämään, onko Unity sopiva heille. Motivaationa työn tekemiseen on kirjoittajan oma kokemus videopeleistä sekä kiinnostus pelimoottoreita ja pelinkehitystä kohtaan.

Lähestyttävyydelle ei ole olemassa vain yhtä ja tiettyä määritelmää. Tässä tutkielmassa se on määritelty tämän tutkielman tarkoitusperien pohjalta käyttäen valmiita kehittäjä- ja käyttäjäkokemuksen mittareita. Tutkielmassa lähestyttävyydellä tarkoitetaan ohjelmointiympäristön käyttäjäkokemuksen mittausta arvioimalla Unityn houkuttelevuutta, helppokäyttöisyyttä, hyödyllisyyttä sekä käyttöastetta. Lähestyttävyyttä tutkitaan olemassa olevan kirjallisuuden sekä käytännön esimerkin pohjalta.

Tutkielma sisältää kirjallisuuskatsauksena luodun teoriaosuuden sekä käytännön esimerkin pelinkehityksestä. Teoriaosuus alkaa taustoittamalla aihetta pelinkehityksen prosessilla ja jatkuu tarkastelemalla syvemmin yleisempien pelimoottoreiden ominaisuuksia sekä lähestyttävyyttä. Pelimoottoreista Unity on valittu lähestyttävyyden tutkimisen keskiöön, sillä se on pelimoottoreista suosituin ja siitä on löydettävissä riittävästi tietoa kirjallisuuskatsauksen tekemiseen. Tutkielman lopussa käytännön esimerkissä luodaan pieni videopeli seuraamalla opetusvideoita, ja arvioidaan Unityn lähestyttävyyttä saatujen kokemusten perusteella.

Olemassa oleva kirjallisuus osoittaa, että Unityn lähestyttävyys on hyvällä tasolla ja se on lähestyttävyydeltään paras suosituimmista pelimoottoreista. Tähän vaikuttaa suurelta osin käyttäjäystävällinen käyttöympäristö sekä laaja dokumentaatio ja suuri yhteisö, joka auttaa käyttäjiä ongelmatilanteiden ratkomisessa. Unityn vaatima ohjelmointitaito vaikuttaa olevan ainoa tekijä, joka heikentää sen lähestyttävyyttä. Muun muassa Unrealissa käytettävä visuaalinen ohjelmointi ei väitteiden mukaan vaadi lainkaan ohjelmointitaitoa. Käytännön esimerkissä voidaan yhtyä olemassa olevan kirjallisuuden väitteisiin Unityn lähestyttävyydestä. Kokemusten perusteella Unity on helppokäyttöinen ja nopeasti opittava, mutta vaatii ohjelmointitaitoa käyttäjältään.

Avainsanat: Unity, lähestyttävyys, pelinkehitys, ohjelmointi, videopeli

SISÄLLYSLUETTELO

1 Johdanto

Videopelit kasvavat koko ajan tarinallisesti rikkaammiksi ja grafiikallisesti todentuntuisemmiksi kokonaisuuksiksi, mikä on herättänyt suuren yleisön huomion. Tämän seurauksena pelimoottorien suosio on kasvanut pelinkehittäjien määrän kasvaessa. Tässä työssä otetaan erityistarkasteluun Unity-pelimoottori sen ollessa suosituin kaikista markkinoilla saatavilla olevista pelimoottoreista [10]. Unityn valintaan vaikutti suuresti se, että siihen liittyen löytyi tarpeeksi tieteellistä kirjallisuutta kirjallisuuskatsauksen tekemiseen. Tutkielma käsittelee Unityn lähestyttävyyttä pelinkehityksessä, verraten sitä muihin suosituimpiin pelimoottoreihin. Muut pelimoottorit on otettu mukaan tutkielman teoriaosuuteen, jotta saadaan laajempi katsaus pelimoottoreiden ominaisuuksista sekä vertailukohtia Unitylle. Pelimoottoreista käydään läpi niiden keskeisimpiä ominaisuuksia ja tutkitaan niiden lähestyttävyyttä. Tässä työssä lähestyttävyyttä tarkastellaan aloittelevan pelinkehittäjän näkökulmasta, kenellä on hieman kokemusta ohjelmoinnista. Tämä tutkimus auttaa aloittelevia pelinkehittäjiä päättämään, mikä pelimoottori on sopiva heille.

Tieteellinen tutkimus Unityn käytöstä pelinkehityksessä ja sen lähestyttävyydestä on vähäistä, joten niiden tutkiminen on haastavaa. Toisaalta tämä työ tulee hieman paikkaamaan tilannetta vähäisen tiedon osalta. Motivaationa työn tekemiseen on kirjoittajan oma kiinnostus videopelejä, pelimoottoreita sekä pelinkehitystä kohtaan.

Tutkielma aloitetaan luvussa 2 kertomalla videopeleistä sekä pelinkehityksestä. Luvussa 3 käydään läpi pelimoottori yleisellä tasolla ja tutustutaan tarkemmin Unityyn sekä muutamaan tärkeimpään pelimoottoriin. Luvussa 4 käsitellään lähestyttävyyttä terminä sekä tutkitaan tarkemmin Unityn lähestyttävyyttä. Luvussa käydään läpi myös muiden potentiaalisten pelimoottoreiden lähestyttävyyttä pinnallisesti. Luvussa 5 tehdään käytännön esimerkki pienen pelin luomisesta Unitylla, minkä perusteella arvioidaan Unityn lähestyttävyyttä.

2 Videopelit ja pelinkehitys

Videopeleillä tarkoitetaan kaksi- tai kolmeulotteista virtuaalista maailmaa, jossa pelaaja hallitsee useimmiten pelihahmoa, ajoneuvoa tai eläintä [5]. Yleensä videopelien tarkoitus on viihdekäyttö, mutta niitä voidaan hyödyntää myös terveysalalla, opetuksessa, liiketoiminnassa, teollisuudessa sekä sotilaallisilla aloilla [9]. Videopelit syntyivät muistiin perustuvan laskennan alkuaikoina yksinkertaisina teknisinä demoina, joita opiskelijat saattoivat luoda yliopiston tietokoneluokissa keksiäkseen hauskan ja helpon tavan näyttää tietokoneen ominaisuuksia tekniikasta tietämättömille ihmisille. Kun tietokoneet tulivat paremmin koko kansan saataville, alkoivat teknikot ohjelmoida omia pelejään. He tekivät kaiken suunnittelun, grafiikat, tasot, tarinan ja ohjelmistot itse. Pelien kehittyessä yhä suuremmiksi, näyttävimmiksi ja tarinallisesti rikkaammiksi, alkoivat teknikot keskittyä itse koodin kirjoittamiseen. Rinnalle palkattiin muunmuassa graafisia taiteilijoita ja pelin suunnittelijoita. Pelinkehitystiimien koko kasvoi muutamasta henkilöstä kaksinumeroisiksi ryhmiksi. [2]

Cohenin ja Bustamanten [2] mukaan pelinkehitys on prosessi, joka jaetaan viiteen vaiheseen. Ensimmäinen vaihe on konseptivaihe (engl. Consept Phase), jossa kehitystiimi pohtii mitä peliltä halutaan, mitä sen pitää saavuttaa ja mitä vaaditaan, jotta se tavoittaa yleisönsä. Kehitystiimi suunnittelee peliä, antaa sille budjetin sekä sopii alustavan aikataulun pelin valmistumiselle. Toisena on tuotantoa edeltävä vaihe (engl. Pre-Production Phase), jonka alussa budjetin, aikataulun ja henkilöstön määrän perusteella päätetään, kuinka laaja peli luodaan. Seuraavaksi luodaan suunnitelmat itse pelille (engl. Game Design Doc), grafiikoille (engl. Art Design Doc), ohjelmistolle (engl. Tech Design Doc) sekä peliäänille. Tämän jälkeen luodaan prototyyppejä pelistä, jotta kehitystiimi, kumppanit sekä rahoittajat näkevät pelin idean olevan mahdollinen toteuttaa. Tämä myös luo uskoa projektin onnistumiseen ja näyttää kehityssuunnan. Kolmas vaihe eli tuotannon vaihe (engl. Production Phase) on vaiheista pisin, sillä se voi kestää kuukausista jopa vuosiin, riippuen asetetusta aikataulusta. Tässä vaiheessa luodaan itse peliä suunnitelmien pohjalta, minkä ohella myös muun muassa kirjoitetaan ja nauhoitetaan pelille dialogeja, sävelletään pelimusiikkia ja mainostetaan peliä. Neljäntenä on rutistusvaihe (engl. Crunch Mode), minkä aikana peli viimeistellään. Grafiikat, peliäänet, musiikki ja kerronta lisätään pelikoodiin. Kun pelin eri osat on koottu yhteen, kokonaisuutta testataan ja korjataan, kunnes se on valmis luovutettavaksi pelin julkaisijalle.

Viides eli viimeinen vaihe on tuotannon jälkeinen vaihe (engl. Post-Production phase), minkä aikana osa pelinkehittäjistä siirretään uusien projektien pariin. Suunnitelmista riippuen pelinkehittäjiä voi jäädä luomaan peliin ladattavaa lisämateriaalia (engl. downloadable content, DLC) tai korjaamaan pelin mahdollisia vikoja uusilla päivityksillä.

3 Pelimoottori

Tässä osassa käydään läpi pelimoottorien toimintaperiaatteita sekä sekä tutustutaan yleisimmin käytettyihin pelimoottoreihin. Kohdassa 3.1 tarkastellaan pelimoottorien teoriaa ja myös pohjustetaan aihetta pelimoottorien historialla. Kohdassa 3.2 tutustutaan syvällisemmin tämän tutkielman pääaiheeseen eli Unity-pelimoottoriin. Kohdassa 3.3 tarkastellaan yleisellä tasolla muita merkittäviä pelimoottoreita ja pohditaan miten ne eroavat Unitystä.

3.1 Pelimoottorien taustaa

Termi ”pelimoottori” syntyi 1990-luvun puolivälissä viitaten FPS-peleihin (engl. First Person Shooter), kuten suosittuun Doom-peliin. Doom-pelissä on eroteltu hyvin pelin ohjelmisto-osat sen muista osista. Pelin ohjelmisto-osia ovat esimerkiksi grafiikoiden renderöinti, törmäyksentunnistus sekä äänijärjestelmä, ja muita osia ovat esimerkiksi itse grafiikat sekä tasojen suunnittelu. Tämä jako vaikutti ratkaisevasti pelinkehitykseen, kun kehittäjät pystyivät hyödyntämään valmista ohjelmistoa muokkaamalla ainoastaan muun muassa grafiikoita, tasosuunnittelua, hahmoja ja ajoneuvoja. Tämä synnytti myös paljon pieniä itsenäisiä pelistudioita, jotka ainoastaan muokkasivat jo olemassa olevia pelejä. 90-luvun lopulla jotkut pelit suunniteltiinkin nimenomaan uudelleenkäyttöä ja muokkausta ajatellen. Tästä eteenpäin pelien moottoreita tehtiin tarkoituksella helposti muokattaviksi skriptikielellä, jonka seurauksena pelimoottorit ovat kehittyneet sellaisiksi, mitä ne tänä päivänä ovat. Pelinkehittäjät lisensoivat käyttämiään pelimoottoreita ja pystyvät näin videopelien tuottamisen ohella ansaitsemaan rahaa. Pelinkehittäjät yleensä hyödyntävät omaa pelimoottoriaan useissa peleissään. On huomattavasti halvempaa hyödyntää vanhojen pelien ohjelmistoja, kuin luoda jokainen peli tyhjästä. [2]

Nykyään pelimoottorit auttavat pelinkehityksessä antamalla valmiita visuaalisia mallipohjia ja resursseja uudelleenkäytettäväksi, joten ohjelmointikokemuksen tarve vähenee huomattavasti. Pelimoottorit toimivat perustana peliin liittyvien tehtävien, kuten grafiikan näyttämisen, datan tulkitsemisen, fysiikkaan liittyvän laskennan sekä muistinhallinan

toteuttamiselle. Tämä nopeuttaa pelinkehittäjien toimintaa ja antaa aikaa keskittyä pelin yksityiskohtien luomiseen. Pelimoottorin avulla sama peli voidaan helposti luoda monelle eri alustalle, kuten PC:lle, konsoleille ja mobiililaitteille. [15]

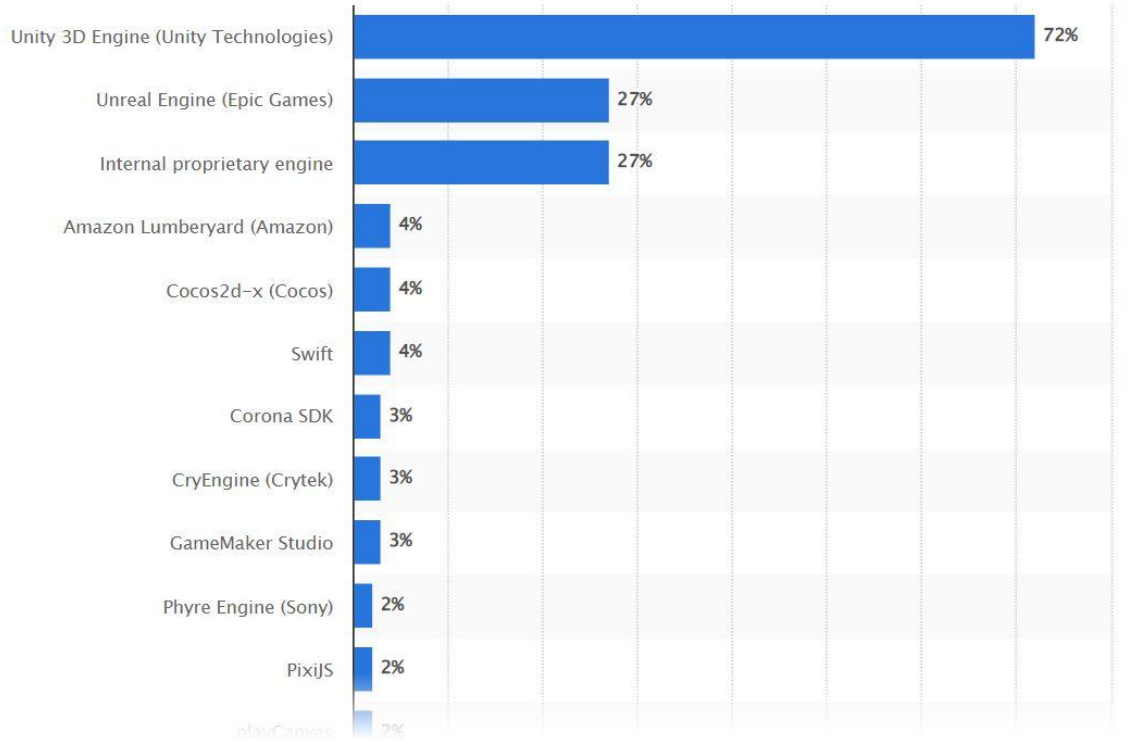
3.2 Unity

Unity (usein myös Unity3D) on Unity Technologiesin luoma ohjelma kaksi- ja kolmiulotteisten pelien luomiseen. Se ilmestyi kesäkuun 6. päivä vuonna 2005. Sen loivat David Helgason, Joachim Ante ja Nicholas Francis Tanskassa. Tavoitteena oli luoda edullinen pelimoottori ammattimaisilla työkaluilla harrasteleville pelinkehittäjille. Alun perin Unity oli saatavilla ainoastaan Mac OS X:lle, mutta version 1.1 myötä se tuli saataville myös Windowsille. [6] Unityllä voidaan luoda pelejä useille alustoille, kuten PC:lle, mobiililaitteille, konsoleille sekä virtuaaliseen todellisuuteen. Unity myös tarjoaa erillisen tilan kaksiulotteisten pelien luomiseen. Unityn ohjelmointikieliä ovat C#, JavaScript sekä Boo, ja siinä käytetään komponenttipohjaista modulaarista järjestelmää objektien luomiseen. Objektit luodaan komponenteista, jotka on rakennettu funktioista, mikä poikkeaa tavallisesta olio-ohjelmoinnissa käytettävästä perimisestä (engl. inheritance). Modulaarinen lähestymistapa lisää joustavuutta ohjelmointiin ja takaa sujuvan komponenttien yhteensovituksen. [1][15]

Barczakin ja Woźniakin [1] artikkelissa käsiteltävistä pelimoottoreista Unity on ainoa, joka ei tarjoa pelin logiikan luomiseen visuaalista ohjelmointia vaan tyytyy perinteiseen ohjelmointiin. Unityssä on Asset Store, josta käyttäjä voi ladata paljon erilaisia laajennusosia (engl. plug-in) sekä materiaaleja. Yksi mainitsemisen arvoinen laajennusosa on työkalu 3D-mallien muokkaamiseen, sillä Unityn perusversio ei itsessään tarjoa tällaista työkalua. 3D-mallien animointi on myös hyvin puutteellista verrattuna muihin pelimoottoreihin, eikä luurankoanimaation (engl. skeleton animation) luominen ole mahdollista. Artikkelin mukaan Unityn Asset Store päihittää sisältönsä laajuudella muiden pelimoottorien vastaavat kirjastot, vaikkakin useimmat sen tarjoamat tuotteet ovat maksullisia. Unity on sopiva pienemmille pelistudioille sekä harrastelijoille, sillä suurien pelien kohdalla Unityn pelimoottori alkaa menettämään tehonsa eivätkä Unityn työkalut ole riittävän kehittyneitä AAA-luokan pelien luomiseen. [1]

3.3 Muut pelimoottorit

Voheran ja muiden [15] mukaan parhaita pelimoottoreita esiteltäväksi ovat Unityn lisäksi Unreal Engine, Crytekin Cryengine sekä YOYOn GameMaker, koska ne ovat helpoiten ymmärrettäviä pelimoottoreita. Nämä kaikki pelimoottorit lukeutuvat kuvan 1 mukaisesti suosituimpien pelimoottorien joukkoon. Tutkimuksen mukaan Unity on suosituin pelimoottori Isossa-Britanniassa. Toiseksi suosituin on Epic Gamesin Unreal Engine, jolla on luotu muunmuassa suosittu selviytymispeli *Fortnite.* Unrealin kanssa samoissa lukemissa ovat pelinkehittäjien omat pelimoottorit (Internal proprietary engine), joita ei ole yleisesti saatavilla. Tällainen on esimerkiksi Rockstar Gamesin RAGE (Rockstar Advanced Game Engine), jolla on luotu muunmuassa suosittu videopeli *Grand Theft Auto V* [5]. Vaikka tutkimus käsittää ainoastaan Ison-Britannian, on se suuntaa antava koko maailman tilanteeseen.



Kuva 1: Eri pelimoottorien suosio Isossa-Britanniassa 2019 [10].

Unreal Engine julkaistiin vuonna 1998 ja sen uusin versio 5.0 ilmestyi huhtikuussa 2022. Sillä voi luoda pelejä useille alustoille, kuten PC:lle, mobiililaitteille, konsoleille sekä

virtuaaliseen todellisuuteen. Ohjelmointi tehdään C++-kielellä tai Unrealin omalla Blue-Print-ohjelmoinnilla. Unrealin merkittävimpiä hyötyjä ovat uudelleenkäytettävät koodikirjastot, olio-ohjelmointi sekä BluePrint-ohjelmointi. BluePrint on visuaalista ohjelmointia, jossa koko pelin logiikan voi luoda diagrammilla kirjoittamatta riviäkään koodia. [15] Unrealin työkalut näyttävien pelien luomiseen ovat paremmat kuin Unityllä. Uusimman version myötä Unreal tarjoaa Lumen-työkalun, jonka avulla peli saa todenmukaisen valotuksen, varjot sekä heijastukset automaattisesti. Toinen merkillepantava ominaisuus on uusi Nanite-työkalu, joka tarjoaa todenmukaiset peligrafiikat ilman huomattavaa pelin suorituskyvyn laskemista. [11] Unreal on ilmainen käyttää, kun peli on tuottanut alle miljoona dollaria.

GameMaker ilmestyi vuonna 2011 ja sen uusin versio GamerMaker 2 tuli markkinoille vuonna 2017. GameMaker on luotu C++- ja C# -kielillä ja tarjoaa työkalut järjestelmäriippumattomien kaksiulotteisten pelien luomiseen. Myös kolmiulotteisia ominaisuuksia on saatavilla rajoitetusti. Gamemaker ei vaadi aiempaa ohjelmointikokemusta visuaalisen ohjelmointikielen eli GameMaker Languagen (GML) ansiosta. GameMakerin hinnoittelu määräytyy sen mukaan, kuinka monelle alustalle haluaa pelin saataville. [15]

CryEnginen ensimmäinen versio ilmestyi vuonna 2002 ja alun perin sen oli tarkoitus olla vain tekninen demo Nvidialle. CryEngine on keskittynyt erityisesti FPS-pelien luomiseen pyötäkoneille, konsoleille sekä virtuaaliseen todellisuuteen. Ohjelmointi tehdään C++ ja Lua -kielillä. CryeEgine tarjoaa pelimoottorin lisäksi avoimen lähdekoodin, minkä ansiosta pelin kustomointi on laajempaa. CryEnginen vahvuuksia ovat hyvälaatuiset grafiikat sekä hyvä pelin suorityskyky. CryEngine on ilmainen käyttää, mutta vaatii tekijänpalkkiota pelin julkaisemisen jälkeen. [15][1]

4 Lähestyttävyys

Lähestyttävyys ei ole yleinen sana suomen kielessä, mutta se on sopivin sana kuvaamaan tämän työn luonnetta. Lienee tarpeellista määritellä lähestyttävyys, jotta pelimoottoreiden lähestyttävyyttä osataan arvioida oikein. Kohdassa 4.1 etsitään parhaiten kuvaava määritelmä lähestyttävyydelle. Kohdassa 4.2 arvioidaan Unityn lähestyttävyyttä olemassa olevan kirjallisuuden perusteella. Kohdassa 4.3 käydään läpi pinnallisesti muiden suosituimpien pelimoottorien lähestyttävyyttä. Kohdassa 4.4 kuvataan tapa, jolla Unityn lähestyttävyyttä mitataan käytännössä.

4.1 Lähestyttävyyden määritelmä

Fagerholmin ja Münchenin [4] luoma määrittely lähestyttävyydelle, josta he käyttävät lyhennettä DE eli ”developer experience”, on melko lähellä tutkielmaan ajateltua lähestyttävyyttä. Artikkelissa kehittäjäksi ajatellaan henkilöä, joka jollain tavalla on mukana ohjelman kehityksessä. Tässä tapauksessa tarkastellaan siis henkilöä, joka kehittää peliä käyttäen Unityä. Kokemuksella tarkoitetaan projektiin osallistumista, eikä kerrytettyä kokemusta esimerkiksi ohjelmoinnista tai pelimoottorien käytöstä. Artikkelin määritelmän mukaan kehittäjän kokemus koostuu useista osa-alueista, joista tutkielman kannalta merkittävä on ohjelmointiympäristön kokemus, mikä koostuu muun muassa käytetyistä työkaluista, ohjelmointikielistä, kirjastoista ja alustoista. [4]

Koska Unity on pitkälle kehitetty ohjelmisto, jonka työkalut ovat käyttövalmiita, voidaan sen lähestyttävyyttä mitata myös käyttäjäkokemuksella (engl. user experience eli UX). Rajanen ja muut [8] esittelevät useita määritelmiä käyttäjäkokemukselle, joista tämän työn lähestyttävyyden mittaamiselle osuvin määritelmä on Kujalan ja muiden [7] määrittelemä käyttäjäkokemuksen mittaus järjestelmälähtoisestä näkökulmasta. Määritelmän mukaan mitattavia osa-alueita ovat järjestelmän houkuttelevuus, helppokäyttöisyys, hyödyllisyys ja käyttöaste. Osa-alueiden merkityksiä avataan tekstissä myös tarkemmin. *Houkuttelevuudella* arvioidaan näyttääkö tuote houkuttelevalta ja onko se kiinnostava käyttäjän mielestä. *Helppokäyttöisyydellä* kuvataan tuotteen käytön helppoutta ja vaivattomuutta. *Hyödyllisyydellä* kuvataan, kuinka hyödyllinen tuote on käyttäjälle. *Käyttöasteella* tarkoitetaan tuotteen käyttöä ajan mittaan. Tämän tutkielman tapauksessa käyttöaste voisi tarkoittaa esimerkiksi sitä, kuinka paljon itse Unityä käytetään pelin logiikan ohjelmointiin verrattuna. Muunmuassa Unreal-pelimoottorissa ohjelmointi tapahtuu täysin pelimoottorin sisällä, joten siinä käyttöaste on todennäköisesti suurempi kuin Unityssä.

Näiden kahden määritelmän pohjalta Unityn lähestyttävyyttä mitataan tämän työn teoriaosuudessa ohjelmointiympäristön käyttäjäkokemuksena. Lähestyttävyys mielletään tuotteen käytön aloittamiseen, eikä niinkään pitkäaikaiseen käyttöön. Luvussa 5 Unityn käyttöä kokeiltaessa kokemus arvioidaan Kujalan ja muiden [7] määrittelemällä mittaustavalla, jossa käyttäjä merkitsee itsearviona kokemuksensa graafiin ajan suhteena.

4.2 Unityn lähestyttävyys

Dickson ja muut [3] vertailevat artikkelissaan Unityn ja Unrealin käyttökokemuksia kolmiulotteisen pelin kehityksessä. Heillä on testiryhmänä opiskelijoita, jotka aloittivat Unityn sekä Unrealin käytön ja arvioivat kokemuksiaan. Pelimoottoreista tutkittiin, onko niiden oppiminen nopeaa, kuinka suosittuja ne ovat, mitä ohjelmointikieliä on käytettävissä, toimiiko pelimoottori usealla eri alustalla, onko pelimoottori stabiili, millainen yhteisö ja dokumentaatio pelimoottorilla on, sekä kuinka halpa pelimoottori on käyttää.

Dickson ja muut [3] kertovat Unityn olevan suosittu pelimoottori, minkä vahvistaa myös kuva 1 [10]. Suosiosta kertoo myös Unityn aktiivinen yhteisö, joka vastasi opiskelijoiden kysymyksiin ja näin edesauttoi oppimista. Unityssä käytettävät JavaScript ja C# ovat tunnettuja ohjelmointikieliä, mikä parantaa Unityn lähestyttävyyttä. Artikkelissa myös kerrotaan Unityn olleen stabiilimpi kuin Unreal, vaikkakin pieniä ongelmia esiintyi molemmissa. Unity ei ole laskennallisesti niin raskas kuin Unreal, joten se soveltuu myös vanhemmille ja tehottomille tietokoneille. Myös Unityn maksuttomuus parantaa lähestyttävyyttä. Kerättyjen kokemusten mukaan Unityn käyttö on helpompaa ja sen oppiminen nopeampaa kuin Unrealin.

Barczak ja Woźniak [1] nostavat esiin Unityn dokumentaation, joka on paras heidän vertailemistaan pelimoottoreiden dokumentaatioista. Heidän mukaansa Unityssä käytettävä C#-kieli on helpompi ja mukavampi käyttää kuin C++-kieli. Myös valmiit pohjat ja esimerkit, käyttäjäystävällinen käyttöliittymä sekä suuri ja aktiivinen yhteisö parantavat käyttäjäkokemusta ja täten lähestyttävyyttä. Heidän vertailemistaan pelimoottoreista Unity on aloittelijoille paras, sillä se on kokonaisuudessaan helpoin oppia ja käyttää. Myös Vohera ja muut [15] toteavat Unityn olevan paras pelimoottori aloittelijoille, perustellen väitettään hyvin samankaltaisilla argumenteilla kuin Barczak ja Woźniak [1].

4.3 Muiden pelimoottorien lähestyttävyys

Sekä Vohera ja muut [15] että Barczak ja Woźniak [1] toteavat Unrealin soveltuvan paremmin kokeneemmille käyttäjille. Vohera ja muut [15] mainitsevat kuitenkin Unrealin oppimiskäyrän olevan jyrkempi kuin Unityn, sen visuaalisen skriptauksen sekä kehittyneen graafisen ympäristön ansiosta. Unrealin tarjoama BluePrint on etu, sillä ohjelmoin-

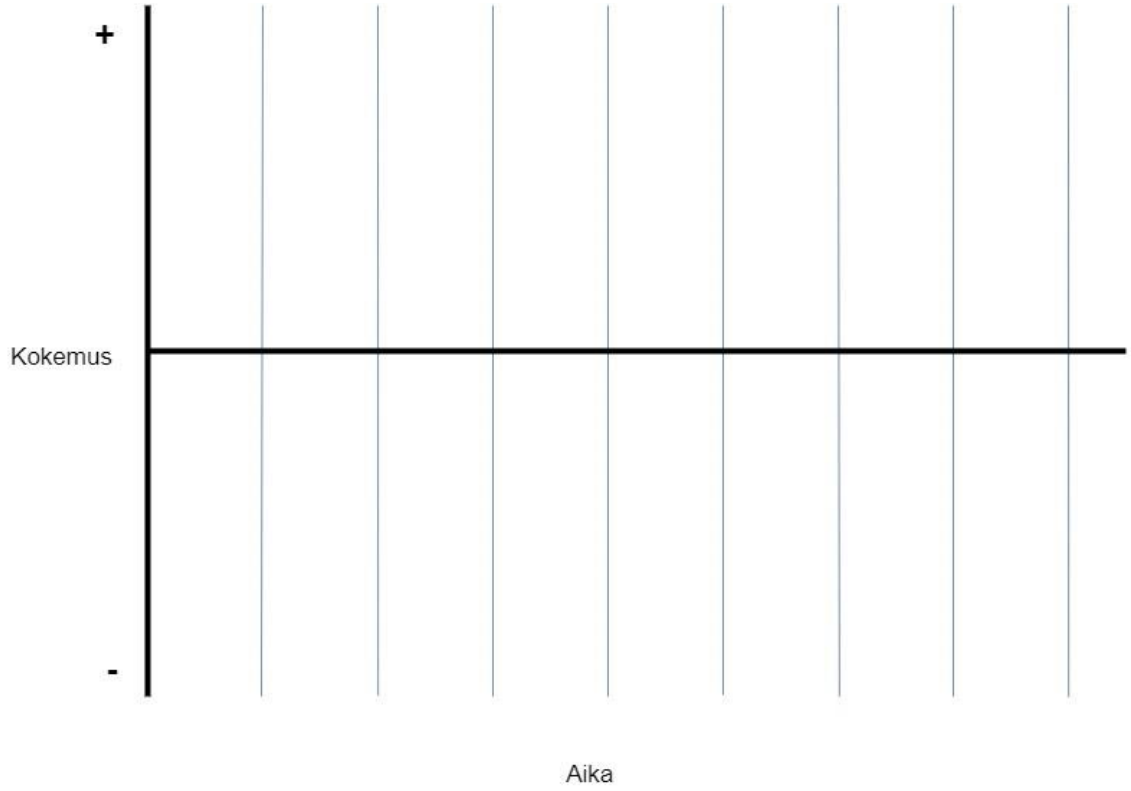
titaitoa ei näin ollen tarvitse, joten ohjelmointia taitamattomalle Unreal voisi olla sopivampi, kuin Unity. Pelin logiikan luominen BluePrintillä pitäisi onnistua jopa nopeammin, kuin perinteisellä ohjelmoinnilla.

Barczak ja Woźniak [1] eivät suosittele CryEngineä aloittelijoille, sillä sen dokumentaatio on huono, opetusvideoita (engl. tutorial) on vain vähän saatavilla ja yhteisö on pieni. Voheran ja muiden [15] mukaan GameMaker on päinvastoin erittäin helppo oppia, sillä se ei vaadi mitään ohjelmointitaitoa. Tällöin myös suunnittelijat ja taiteilijat voivat rakentaa projektiaan ilman ohjelmoijan apua, käyttämällä visuaalisen skriptauksen raahaa ja pudota (engl. drag-and-drop) -toimintoa. Gamemakerin huonoja puolia kuitenkin ovat tuki ainoastaan kaksiulotteisten pelien luomiseen sekä ohjelman maksullisuus.

4.4 Lähestyttävyyden arviointi

Unityn käytön esimerkissä luvussa 5 lähestyttävyyttä arvioidaan Kujalan ja muiden [7] kuvaamalla tavalla, missä kokemukset kuvataan graafiin. Tätä graafia kutsutaan käyttäjäkokemuksen käyräksi (engl. UX Curve). Tässä tavassa jokaisesta kohdassa 4.1 kuvatusta osa-alueesta (houkuttelevuus, helppokäyttöisyys, hyödyllisyys ja käyttöaste) luodaan kuvan 2 mukainen graafi, johon käyttäjä merkitsee kokemuksiaan kyseisestä osa-alueesta. Graafissa kokemusten positiivisuus esitetään ajan suhteena.

Kujala ja muut [7] kehittivät käyttäjäkokemuksen käyrän tutkiessaan ihmisten kokemuksia matkapuhelinten käytössä. Tutkimusaika oli 3–12 kuukautta, jonka aikana käyttäjä sai kirjata kokemuksiaan graafiin. Käyrän kerrotaan olevan parhaiten sopiva valmiin tuotteen pitkäaikaisen käytön testaamiseen. Luvun 5 käytännön esimerkissä tutkimusaika on huomattavasti lyhyempi, sillä lähestyttävyys mielletään tuotteen käytön aloittamiseen, eikä niinkään pitkäaikaiseen käyttöön. Käyrä kuitenkin vaikuttaa sopivalta myös lyhytaikaiseen lähestyttävyyden mittaukseen sen helppokäyttöisyyden ja ymmärrettävyyden ansiosta.



Kuva 2: Lähestyttävyyden arvioinnin graafipohja, mukaelma Kujalan ja muiden [7] graafista.

Luvun 5 käytännön esimerkissä Unityn lähestyttävyyttä arvioidaan siis ainoastaan tämän tutkielman kirjoittajan näkökulmasta saatujen kokemusten perusteella. Kirjoittajalla ei ole aiempaa kokemusta Unityn tai muiden pelimoottoreiden käytöstä.

5 Esimerkki Unityn käytöstä

5.1 Pelin luominen

Unityn asentaminen onnistuu lataamalla ensiksi Unityn omilta verkkosivuilta *Unity Hubin*, jonka kautta itse pelimoottorin laataminen ja asennus tapahtuu. Valitsemalla *Unity Personal* -tilauksen saa Unityn käyttöönsä ilmaiseksi. Tässä tutkielmassa käytetään Unityn versiota 2021.3.10f1, jota Unity Hub tarjosi ensimmäisenä asennettavaksi. Ensimmäistä projektia alottaessa Unity tarjoaa valmiita projektipohjia (engl. template), joiden päälle aloittelijan on helpompi alkaa rakentamaan peliä. Valittavia pohjia ovat muun muassa FPS-peli, kaksiulotteinen peli, Lego-peli sekä karting-peli. Tässä esimerkissä valmista pohjaa ei käytetä, vaan pelin luonti aloitettiin tyhjältä kolmiulotteiselta pohjalta,

sillä harva opetusvideo hyödyntää valmista pohjaa. Pelin luomisessa käytettiin apuna hyvin pitkälti erilaisia opetusvideoita, joita seuraamalla pelinkehitys oli sujuvaa. Videot antoivat hyvän pohjan jatkaa pelin toiminnallisuuksien kehittämistä itsenäisesti vielä pidemmälle. Pelinkehitys Unityllä tehdään Unity Editorin kautta, jossa luodaan ja muokataan peliobjekteja (engl. GameObject). Peliobjekteja voivat olla esimerkiksi seinät, viholliset, valot tai pelaajan ase [12]. Peliobjektien toiminnallisuudesta vastaavat komponentit (engl. Component), joita voidaan lisätä peliobjekteille haluttu määrä [13]. Unityn tarjoamia komponentteja on paljon, mutta ne eivät millään riitä luomaan kaikkia toiminnallisuuksia, mitä peliin halutaan. Tällöin käyttäjän täytyy luoda omia skriptejään (engl. script), jotka ovat yksi komponenttilaji. Skriptit ovat tavallisia kooditiedostoja, joita voidaan luoda käyttäen haluttua koodieditoria [14]. Tässä pelissä skriptit luotiin C#-kielellä käyttäen Microsoftin Visual Studiota. Pelin kehityksessä tutustuttiin myös Asset Storen käyttöön, josta on peräisin pelissä käytetty ase.

Kuva 3: Kuvakaappaus luodusta pelistä.

Luotu peli on yksinkertainen FPS-peli, jossa tavoitteena on tuhota kolme tietokoneen ohjaamaa vihollishahmoa ampumalla. Pelaaja aloittaa pelin aulasta, jossa on myöhemmässäkin vaiheessa mahdollista käydä keräämässä elinvoimapisteensä täyteen. Varsinainen pelikenttä löytyy liukuovien takaa, jotka voidaan avata ovien vieressä olevalla painikkeella. Pelikentältä löytyy kolme vihollishahmoa, jotka kiertävät pelikenttää ennaltamäärättyä reittiä pitkin. Vihollinen lähtee jahtaamaan pelaajan hahmoa, jos pelaaja tulee liian

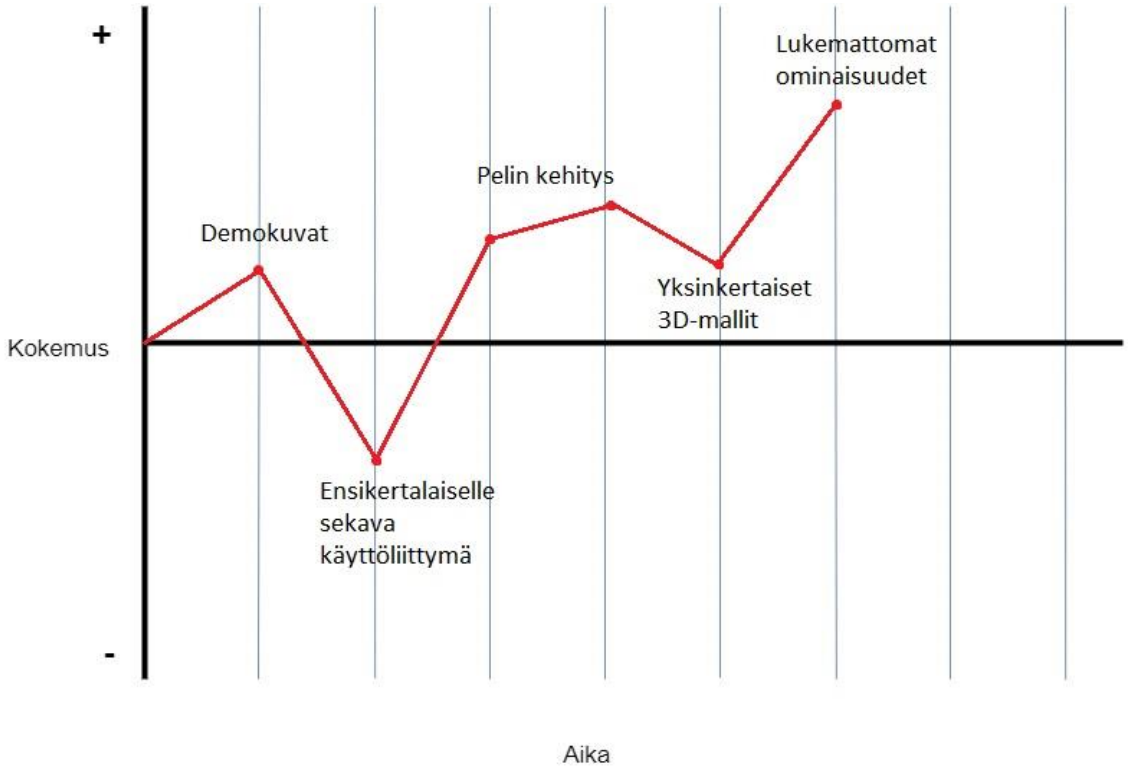
lähelle vihollista. Pelaajan paetessa riittävän kauas, vihollinen luovuttaa ajojahdin suhteen ja jatkaa pelikentän partiointia normaaliin tapaan. Jos pelaajan hahmon ja vihollisen välillä on kontakti, menettää pelaaja elinvoimapisteitään. Elinvoimapisteiden loppuessa peli päättyy. Peliin luotiin myös muita toiminnallisuuksia, kuten pelaajan mahdollisuus mennä kyykkyyn tai juosta nopeampaa.

5.2 Unityn lähestyttävyyden arviointi

Pelin kehitykseen käytettiin aikaa noin 10 tuntia. Tämän aikana kokemuksia kirjattiin tasaisin väliajoin kohdan 4.4 mukaisesti kuvan 2 lähestyttävyyden arvioinnin graafiin jokaisesta lähestyttävyyden mittauksen osa-alueesta. Kokemuksia myös avattiin sanallisesti arvioinnin yhteydessä.

5.2.1 Houkuttelevuus

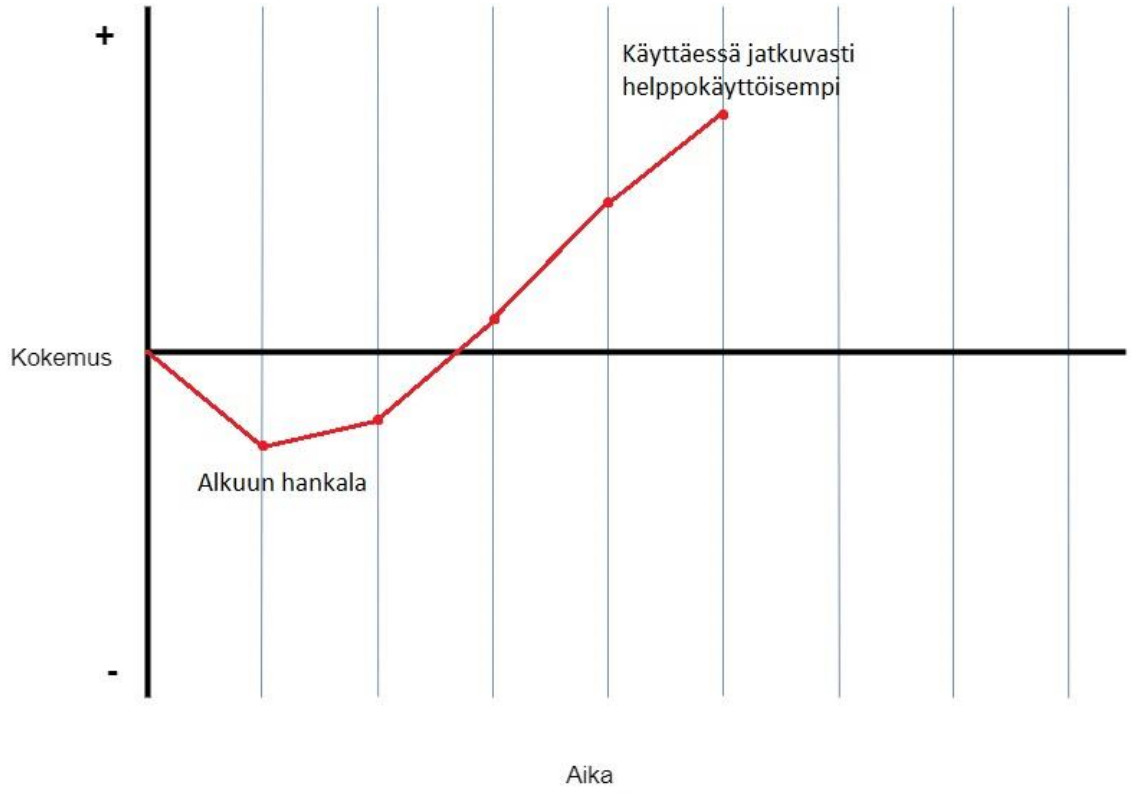
Kuten kuvasta 4 nähdään, Unityä ladattaessa oli sen houkuttelevuus hyvällä tasolla. Demovideot, mainoskuvat ja esimerkit olivat houkuttelevan näköisiä ja saivat innostumaan pelinkehityksestä Unityllä. Avatessa Unity Editorin ensimmäistä kertaa, sai työkalujen ja ominaisuuksien runsas määrä houkuttelevuutta hieman laskemaan, sillä niiden käytöstä ei vielä ollut tietoa ja ne näyttivät monimutkaisilta. Opetusvideoita seuratessa sekä peliä kehittäessä tuli työkalujen käyttö tutuksi ja tämä nosti houkuttelevuutta huomattavasti korkeammalle. Houkuttelevuus kuitenkin hieman laski, kun huomasi 3D-mallien yksinkertaisuuden ja niiden rajoittuneen muokkaamisen. Malleja oli saatavilla runsaasti Asset Storesta, mutta suurin osa näistä maksoi. Pelin ollessa valmis saivat pelinkehityksen lukemattomat mahdollisuudet houkuttelevuuden nousemaan aiempaa korkeammalle. Lisättävien ominaisuuksien kirjo on lähes rajaton.



Kuva 4: Unityn houkuttelevuuden arviointi peliä kehittäessä.

5.2.2 Helppokäyttöisyys

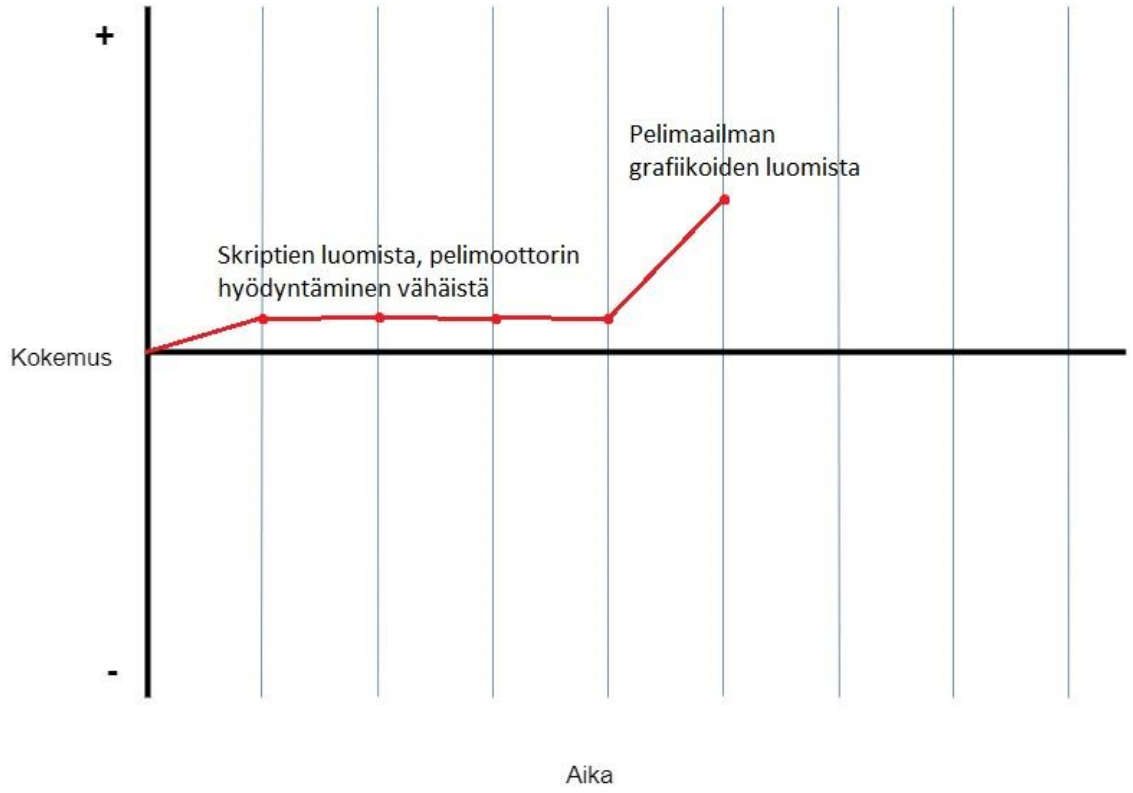
Kuvan 5 mukaisesti Unityn käyttö oli alussa haastavaa, sillä sen ominaisuuksista tai työkaluista ei ollut tietoa. Peliä kehittäessä ja opetusvideoita seuratessa, alkoi Unity olla jatkuvasti helppokäyttöisempi. Unityn oppiminen oli nopeaa ja palkitsevaa. Pelin logiikka kirjoitettiin C#-kielellä, joka ei suuresti poikennut Javasta tai C++-kielestä ja oli näin helppokäyttöinen. Unityn kattava dokumentaatio ja aktiivinen yhteisö varmistivat sen, että ongelmia kohdattaessa olivat ratkaisut nopeasti saatavilla.



Kuva 5: Unityn helppokäyttöisyyden arviointi peliä kehittäessä.

5.2.3 Käyttöaste

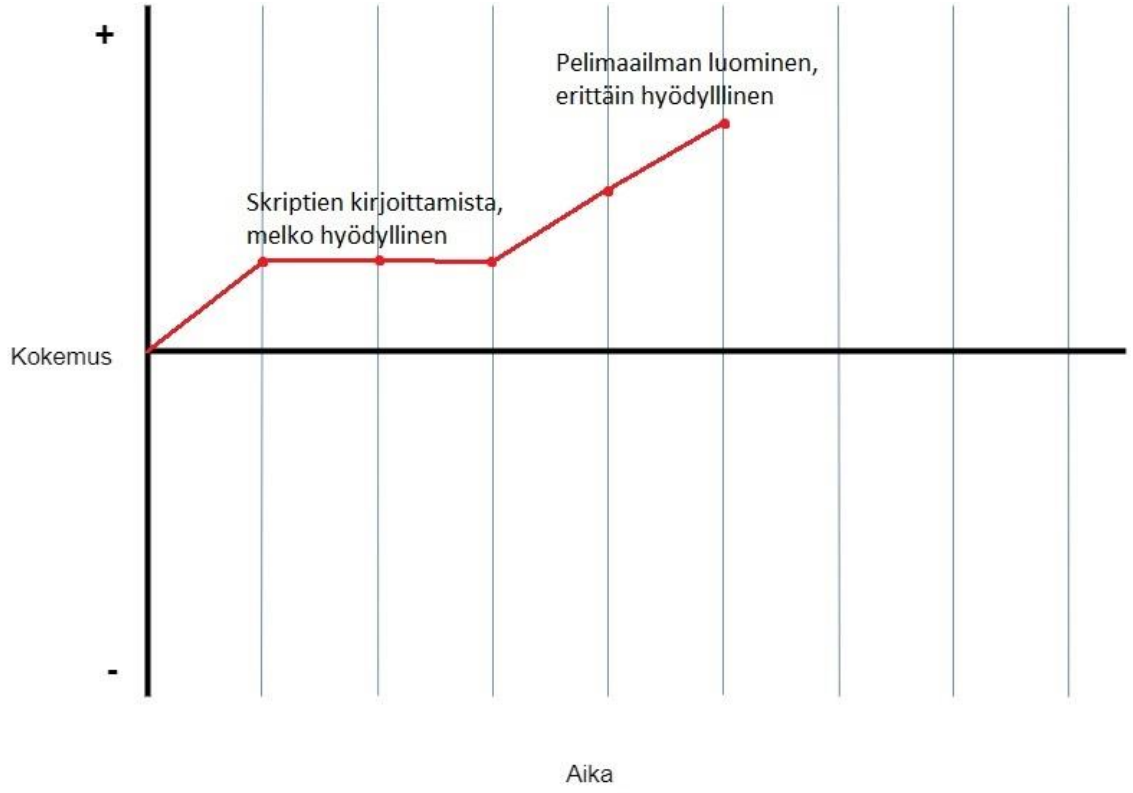
Unityn valmiit komponentit ovat hyvin rajallisia, joten omia skriptejä pitää kirjoittaa paljon. Lisäksi alussa on hyvä oppia perusasiat syvällisemmin, joten on jopa suositeltavaa kirjoittaa yksinkertaisetkin toiminnot itse. Kokeneemman käyttäjän on viisainta hyödyntää enemmän valmiita komponentteja ja jo olemassa olevia skriptejä. Kuten kuvasta 6 nähdään, pelin kehitys oli alussa suurelta osin skriptitiedostojen kirjoittamista Microsoftin Visual Studiota hyödyntäen, jolloin Unityn käyttöaste oli pientä. Vasta pelin logiikan ollessa lähes valmis, päästiin kunnolla hyödyntämään pelimoottorin ominaisuuksia pelimaailman grafiikoiden luomisessa ja käyttöaste nousi huomattavasti.



Kuva 6: Unityn käyttöasteen arviointi peliä kehittäessä.

5.2.4 Hyödyllisyys

Voidaan todeta, että Unityn käyttö pelinkehityksessä on hyödyllistä. Se helpottaa sekä nopeuttaa pelinkehitystä ja innostaa luomaan pelejä. Se tarjoaa valmiita materiaaleja ja antaa ideoita pelien sisältöön. Kuvan 7 mukaisesti Unityn hyödyllisyys alussa ei ollut verrattain korkealla, sillä pelin kehitys oli lähinnä skriptien kirjoittamista Microsoftin Visual Studiota hyödyntäen, mikä ei poikkea paljon pelinkehityksestä ilman pelimoottoria. Pelimaailman grafiikoiden luominen Unitylla on nopeaa ja mutkatonta, missä korostuu sen hyödyllisyys.



Kuva 7: Unityn hyödyllisyyden arviointi peliä kehittäessä.

5.3 Johtopäätökset

Kohdassa 5.2 esitettyjen graafien perusteella voidaan todeta, että Unityn lähestyttävyys parani, mitä pidemmälle pelin kehitys jatkui. Houkuttelevuutta haittasivat ainoastaan ensikertalaiselle sekavan näköinen käyttöliittymä ja 3D-mallien yksinkertaisuus. Käyttöliittymän kuitenkin oppi nopeasti, jolloin houkuttelevuus nousi. Helppokäyttöisyys kärsi alussa, sillä lukuisten työkalujen ja ominaisuuksien vuoksi Unityn käyttö oli monimutkaista. Helppokäyttöisyys kuitenkin parani, kun työkaluihin ja ominaisuuksiin tutustui

opetusvideoiden avulla. Unityn hyödyllisyys ja käyttöaste olivat alkuun alhaisella tasolla, sillä pelin kehitys oli lähinnä skriptien kirjoittamista koodieditoria käyttäen, jolloin Unityn käyttö jäi vähemmälle. Unity toi parhaat puolensa esiin vasta yhdistettäessä luotuja toiminnallisuuksia 3D-maailmaan. Tulosten perusteella voidaan Unityn lähestyttävyyden olevan hyvällä tasolla, sillä jokainen graafi päätyi lopulta positiivisen kokemuksen puolelle.

Käytännön esimerkin tulokset ovat samankaltaisia kohdassa 4.2 kirjallisuudesta saatujen tulosten kanssa. Tästä johtuen käytännön esimerkistä saatuja tuloksia voidaan pitää merkityksellisinä, vaikka ne ovat vain yhdeltä henkilöltä kerättyjä kokemuksia Unityn käytöstä.

6 Yhteenveto

Videopelien nopea kehitys viimeisten vuosien aikana on herättänyt suuren yleisön huomion. Tämän seurauksena pelimoottorien suosio on kasvanut pelinkehittäjien määrän kasvaessa. Pelimoottoreilla on suuri rooli pelien kehityksessä, sillä ne tarjoavat työkaluja, materiaaleja ja valmiita pohjia pelien sujuvaan kehitykseen. Kasvaneen suosion myötä yhä useampi harkitsee pelinkehitystä, ja pelimoottoreiden suurten markkinoiden ansiosta kynnys pelinkehityksen aloittamiseen on matala.

Tutkielman tarkoituksena oli selvittää tarkemmin Unity-pelimoottorin lähestyttävyyttä pelinkehityksessä. Työ myös auttaa aloittelevia pelinkehittäjiä päättämään, onko Unity sopiva heille. Unity valikoitui keskiöön, sillä se on pelimoottoreista suosituin ja siitä löytyvä tieteellinen materiaali on riittävää kirjallisuuskatsauksen tekemiseen. Lähestyttävyyttä arvioitiin teoriaosuudessa kirjallisuuskatsauksena olemassa olevan materiaalin pohjalta, sekä käytännön esimerkissä pelinkehityksessä saatujen kokemusten pohjalta.

Teoriaosuudessa Unityn lähestyttävyyttä ja ominaisuuksia verrattiin muihin suosituimpiin pelimoottoreihin, joiden perusteella Unity on pelimoottoreista lähestyttävyydeltään paras. Unity on sopivin pelimoottori aloittelijoille sen käyttäjäystävällisen käyttöliittymän sekä laajan dokumentaation ja yhteisön ansiosta. Käytännön esimerkissä vahvistettiin jo teoriaosuudessa havaitut asiat ja todettiin Unityn lähestyttävyyden olevan hyvällä tasolla. Itse pelikoodia Unityssä täytyy kirjoittaa skriptien avulla paljon, joten ohjelmoinnista tietämättömälle Unreal voisi olla sopivampi vaihtoehto. Opetusvideoita on runsaasti saatavilla Unityyn ja lähes kaikkiin ongelmiin löytyy ratkaisu dokumentaatiosta tai erilaisilta foorumeilta. Unityn suurin ongelma tulee esille vasta AAA-luokan pelejä luodessa, sillä niiden kehitykseen Unity ei ole sopiva.

Tietoa Unityn lähestyttävyydestä on vain vähän saatavilla, ja siksi tämä työ on merkityksellinen. Lähestyttävyyden mittauksessa ei ole vakiintunutta määritelmää, mikä luo haastetta tutkimustyön tekemiseen. Toisaalta tämä mahdollistaa lähestyttävyyden mittauksen usealla eri tavalla, jolloin kerätty tieto on antoisaa ja monipuolista. On tärkeää, että tutkimustyötä jatketaan ja lähestyttävyyttä mitataan usealta eri osa-alueelta useasta eri näkökulmasta.

Lähdeluettelo

[1] Barczak, A., & Woźniak, H. (2020). *Comparative Study on Game Engines*. Studia Informatica, 23, 5–24. *https://doi.org/10.34739/si.2019.23.01*

[2] Cohen, D. S., & Bustamante, S. A. (2009). *Producing games from business budgets to creativity and design* (1st edition). Focal Press

[3] Dickson, P. E., Block, J. E., Echevarria, G. N., & Keenan, K. C. (2017). *An Experience-based Comparison of Unity and Unreal for a Stand-alone 3D Game Development Course*. In Proceedings of the 2017 ACM Conference on Innovation and Technology in Computer Science Education. *https://doi.org/10.1145/3059009.3059013*

[4] Fagerholm, F., & Münch, J. (2012). *Developer experience: concept and definitio*n. In Proceedings of the 2012 International Conference on Software and System Process (ICSSP), 73–77. *https://doi.org/10.1109/ICSSP.2012.6225984*

[5] Gregory, J. (2019). *Game Engine Architecture, Third Edition*. A K Peters/CRC Press. *https://doi.org/10.1201/9781315267845*

[6] Haas, J. (2014). *A History of the Unity Game Engine*. Worcester: Worcester Polytechnic Institute.

[7] Kujala, S., Roto, V., Väänänen-Vainio-Mattila, K., Karapanos, E., & Sinnelä, A. (2011). *UX Curve: A method for evaluating long-term user experience*. Interacting with Computers, 23(5), 473–483. *https://doi.org/10.1016/j.intcom.2011.06.005*

[8] Rajanen, D., Clemmensen, T., Iivari, N., Inal, Y., Rızvanoğlu, K., Sivaji, A., & Roche, A. (2017). *UX Professionals' Definitions of Usability and UX – A Comparison Between Turkey, Finland, Denmark, France and Malaysia*. HUMAN-COMPUTER INTERACTION - INTERACT 2017, PT IV, 10516, 218–239. *https://doi.org/10.1007/978-3-319-68059-0_14*

[9] Roedavan, R., Pratondo, A., Utoro, R. K., & Sujana, A. P. (2020). *Zetcil: Game Mechanic Framework for Unity Game Engine*. IJAIT (International Journal of Applied Information Technology), 3(2), 965. *https:doi.org/10.25124/ijait.v3i02.2779*

[10] Statista Research Department, Inc. *Leading Game engines used my video game developers in the United Kingdom (UK) 2019*. Statista, 2022 (Viitattu 19.10.2022):

https://www.statista.com/statistics/321059/game-engines-used-by-video-game-developers-uk/

[11] Unreal Engine, *Unreal Engine 5.0 Release Notes*. Saatavilla: *https://docs.unrealengine.com/5.0/en-US/unreal-engine-5.0-release-notes/.* Viitattu 19.10.2022

[12] Unity Technologies (2021.). Unity User Manual (2021.3): *GameObjects*. Saatavilla: *https://docs.unity3d.com/Manual/GameObjects.html.* Viitattu 27.11.2022

[13] Unity Technologies (2022). Unity User Manual (2021.3): *Using Components*. Saatavilla: *https://docs.unity3d.com/Manual/UsingComponents.html.* Viitattu 27.11.2022

[14] Unity Technologies (2022). Unity User Manual (2021.3): *Creating and Using Scripts*. Saatavilla: *https://docs.unity3d.com/Manual/CreatingAndUsingScripts.html.* Viitattu 27.11.2022

[15] Vohera, C., Chheda, H., Chouhan, D., Desai, A., & Jain, V. (2021). *Game Engine Architecture and Comparative Study of Different Game Engines*. In Proceedings of the 2021 12th International Conference on Computing Communication and Networking Technologies (ICCCNT), 1–6. *https://doi.org/10.1109/ICCCNT51525.2021.9579618*