Ville Kivelä

# Pelitapahtumien analysointi pelisuunnittelun tukena

Insinööri (AMK)

Tieto- ja viestintätekniikka

Kevät 2023

KAMK • University of Applied Sciences

## Tiivistelmä

**Tekijä(t)**: Kivelä Ville

**Työn nimi:** Pelitapahtumien analysointi pelisuunnittelun tukena

**Tutkintonimike:** Insinööri (AMK), tieto- ja viestintätekniikka

**Asiasanat:** analysointi, analytiikka, telemetria, Jupyter, Unreal

Työn tavoitteena oli kehittää palvelu, jota Kajaanin ammattikorkeakoulun opiskelijat voivat käyttää peliprojekteissa pelitapahtumien analysointiin. Palvelua varten tarvittiin pelimoottorille lisäosa, jolla tapahtumia voidaan lähettää palvelimelle. Tarvetta oli myös selainpohjaisen analysaattorin kehittämiselle, jolla pelien tapahtumia pystyttiin tarkastelemaan ja analysoimaan. Työssä käytetyn telemetriapalvelimen oli toinen opiskelija kehittänyt jo ennen tämän työn aloittamista. Työn toimeksiantajana toimi Kajaanin Ammattikorkeakoulu Oy.

Analytiikkaa käytetään monella eri alalla sen hyödyllisyyden vuoksi. Sen avulla voidaan tehdä dataan perustuvia päätöksiä sekä saada hyödyllistä tietoa, joka muuten jäisi huomaamatta. Pelitapahtumien analysointia käytetään pelien suunnittelun apuna, uusien pelien kehittämisessä sekä peleihin että pelaajien käyttäytymiseen liittyvissä tutkimuksissa. Datan visualisoimisella tieto saadaan paremmin esitettävään muotoon, jota on helpompi ymmärtää.

Työssä kehitettiin pelimoottorille lisäosa, joka voidaan liittää mihin tahansa samalla pelimoottorilla kehitettyyn peliin. Pelitapahtumien analysointia varten kehitettiin selainpohjainen analysaattori. Analysaattoriin toteutettiin muutamia hyödyllisiä visualisointeja sekä esimerkkejä siitä, miten palvelimelta saadaan luettua tietoa ja miten sitä voidaan käsitellä. Pelimoottorilla toteutettiin myös testisovellus, jolla esiteltiin lisäosan toiminnallisuus ja jota voidaan käyttää palvelun kokonaisuuden testaamiseen.

Aikaansaannoksena syntyi toimiva pelimoottorin lisäosa, joka voidaan konfiguroida eri peleille tapahtumien lähettämiseksi palvelimelle. Työssä toteutettu analysaattori ei sellaisenaan toimi jokaiselle pelille, mutta sitä voidaan käyttää esimerkkinä siitä, kuinka analysaattoria on mahdollista käyttää eri pelitapahtumien analysointiin. Testisovellus havainnollistaa, kuinka lisäosaa voidaan käyttää omassa peliprojektissa, ja se on myös hyödyllinen työkalu, kun halutaan testata eri pelitapahtumien lähettämistä palvelimelle pelin suunnitteluvaiheessa.

## Abstract

**Author(s):** Kivelä Ville

**Title of the Publication:** Analysis of Gameplay Events as Assistance for Game Design

**Degree Title:** Bachelor of Engineering, Information and Communication Technology

**Keywords:** analysis, analytics, telemetry, Jupyter, Unreal

The goal of this thesis was to develop a service that Kajaani UAS students can use in their game projects to analyse gameplay events. For the service, a plugin for game engine that can be used to send events was needed. There was also a need to develop a browser-based analyser that could be used to view and analyse the events. The telemetry server used in the thesis had already been developed by another student before starting this thesis. This thesis was commissioned by Kajaani UAS Oy.

Analytics is used in many different fields because of its usefulness. It can be used to make data-based decisions and to obtain useful information that would otherwise go unnoticed. The analysis of gameplay events is used as assistance for game design and as a help to develop new games. It is also used in studies related to games and player behaviour. By visualising the data, the information is presented in a form that is easier to understand.

In this thesis, a plugin was developed for a game engine, which can be used in any game developed with the same engine. A browser-based analyser was developed to analyse the gameplay events. A few useful visualisations and examples how information can be read from the server and how it can be processed were implemented in the analyser. A test application was also implemented to present the functionality of the plugin and to test the entire service.

As a result of this thesis, a functional game engine plugin was developed, which can be configured for different games to send events to the server. The analyser that was implemented in this work will not work as such for every game, but it can be used as an example of how it is possible to analyse different game events. The test application illustrates how the plugin can be used in your own game project, and it is also a useful tool when you want to test sending different game events to the server during the games design phase.

Sisällys

1 Johdanto ..... 1

2 Pelianalytiikka ..... 2

2.1 Telemetria ja pelattavuusmittarit ..... 3

2.2 Analysoiminen pelisuunnittelun näkökulmasta ..... 4

2.3 Tiedon louhinta, tekoäly ja datan visualisoiminen ..... 8

3 Jupyter Notebook ja Unreal Engine ..... 9

4 Käytännön toteutus ..... 11

4.1 Unreal Engine -lisäosa ..... 11

4.2 Analysaattori ..... 16

4.3 Esimerkkisovellus ja palvelukokonaisuus ..... 20

5 Yhteenveto ..... 22

Lähteet ..... 23

Liitteet

# 1 Johdanto

Analytiikkaa hyödynnetään usealla eri alalla päätöksenteon apuna. Sen avulla voidaan esimerkiksi parantaa tehokkuutta, syventää asiakasymmärrystä, luoda kilpailuetua, lieventää riskejä ja pienentää kustannuksia. Analytiikkaa voidaan myös käyttää pelien kehittämisen työkaluna. Pelitapahtumien analysointi auttaa kehittäjiä ymmärtämään pelaajien käyttäytymistä, parantamaan pelikokemusta sekä suunnittelemaan parempia pelejä.

Työn toimeksiantaja on Kajaanin Ammattikorkeakoulu Oy. Tilaajalla oli tarvetta palvelukokonaisuudelle, jota opiskelijat voivat käyttää omien peliensä pelitapahtumien analysointiin. Erityinen tarve oli pelimoottorin lisäosalle, jota opiskelijat pystyvät käyttämään oman peliprojektinsa kanssa tapahtumien lähettämiseen palvelimelle sekä analysaattorille, jota voidaan käyttää tiedon hakemiseen palvelimelta ja tapahtumien analysoimiseen. Tämän lisäksi kysyntää oli testisovellukselle, jotta nähdään, kuinka lisäosa voidaan liittää ja konfiguroida omaan peliin. Kokonaisuudessa käytetty palvelin oli jo toteutettu ennen tämän työn aloittamista.

Tämän työn ensimmäinen tavoite on luoda pelimoottorille lisäosa, jota opiskelijat voivat käyttää peleissään tiedon lähettämiseen palvelimelle. Toinen tavoite on tehdä selainpohjainen analysaattori, jota opiskelijat voivat hyödyntää peliensä tapahtumien analysoinnissa. Kolmas tavoite on luoda testisovellus, joka käyttää pelimoottorille luotua lisäosaa ja jolla voidaan testata ja näyttää palvelun toimivuutta kokonaisuutena. Pelimoottoriksi valittiin Unreal Engine, jolle lisäosa ja testisovellus toteutettiin. Selainpohjaisen analysaattorin toteutustavaksi valikoitui Jupyter Notebook.

Työn teoriaosuudessa esitellään analytiikkaan liittyviä käsitteitä. Sen jälkeen tarkastellaan pelitapahtumien analysoinnin perusteita ja hyötyjä. Käytännön osassa käytössä olevat työkalut ja teknologiat esitellään lyhyesti. Käytännön osuudessa esitellään myös vaatimukset ja toteutus sekä tarkastellaan palvelua kokonaisuutena. Työ päättyy pohdintaan työn etenemisestä sekä palvelun onnistumisesta alkuperäiset tavoitteet huomioon ottaen.

# 2 Pelianalytiikka

Taloudellisesti kannattavan pelin tekeminen nykypäivän markkinoille on haastavaa. Vuosittain julkaistaan tuhansia pelejä monille eri alustoille ja ne kaikki kilpailevat pelaajien ajasta ja huomiosta. Peliala on omaksunut monia työkaluja käyttöönsä muilta IT-aloilta. Yksi näistä työkaluista on analytiikka, joka on vaikuttanut paljon pelialaan ja pelien kehittämiseen viime vuosina. [1, s. 14.]

Analytiikalla tarkoitetaan prosessia, jossa kerätään, prosessoidaan, analysoidaan ja tulkitaan dataa, jotta siitä saadaan johdettua hyödyllistä tietoa. Analytiikkaan kuuluu olennaisena osana statistiikka, tiedon louhinta, matematiikka, ohjelmointi ja datan visualisointi sekä johtopäätöksien kommunikointi asiaankuuluville sidosryhmille. [1, s. 14.]

Analytiikka voidaan jakaa neljään eri päätyyppiin [2]:

1. Kuvaava analytiikka: mitä tapahtui?
2. Diagnostinen analytiikka: miksi tapahtui?
3. Ennakoiva analytiikka: mitä tulee tapahtumaan lähitulevaisuudessa?
4. Ohjaileva analytiikka: mitä tehdä seuraavaksi?

Analytiikalla on lukemattomia eri alalajeja sekä käyttökohteita. On tärkeää huomata, että analytiikka ja analysointi eivät tarkoita samaa asiaa. Analysointi kuuluu olennaisena osana analytiikkaan.

Analytiikka on Business Intelligencen (BI) osajoukko. BI on laaja termi, jonka tavoitteena on tehdä raa'asta tiedosta hyödyllistä. BI auttaa päätöksiä ohjautumaan enemmän tiedon mukaan (eng. data-driven). [1, s. 14.]

Pelianalytiikalla tarkoitetaan pelaajan käyttäytymisen tutkimista statistiikan avulla. Pelianalytiikkaa voidaan käyttää kaupallistamisen ja markkinoinnin lisäksi myös pelaajien käyttäytymisen tutkimiseen itse pelin sisällä. [3.]

## 2.1 Telemetria ja pelattavuusmittarit

Telemetrialla tarkoitetaan tiedon keräämistä ja toimittamista välimatkan yli. Käytännössä se tarkoittaa, että asiakassovellus kerää ja lähettää dataa palvelimelle tietokantaan. [1, s. 16.]

Datan keräämisessä tulee ottaa huomioon käyttäjien yksityisyys. Datasta on mahdollista luoda käyttäjäprofiileja, joilla voi seurata tarkasti yksittäisen käyttäjän toimintaa [1, s. 30]. Data voi olla myös anonyymiä eli käyttäjästä ei tallenneta yksilöivää tietoa tietokantaan. Tiedon kerääminen telemetrian avulla on hyödyllisempää kuin pelaajilta suoran palautteen saaminen. Pelaajat saattavat liioitella, vähätellä tai valehdella kokemuksistaan [4].

**Mittarilla** tarkoitetaan mitattavaa muuttujaa, jota halutaan tarkastella [3]. Mittari voi olla yksittäinen muuttuja tai monesta muuttujasta johdettu kokonaisuus. Mittarin kanssa yleensä aina tallennetaan aika, jolloin mitattava asia tapahtui. [1, s. 18.]

Mittarit jakautuvat neljään eri pääkategoriaan: asiakas-, yhteisö-, suorituskyky- ja pelattavuusmittareihin [3]. Tässä työssä tarkasteltiin vain pelattavuusmittareita.

**Pelattavuusmittareilla** tarkoitetaan tapahtumia, jotka tapahtuvat pelin sisällä. Se yleensä vastaa kysymyksiin: mitä, missä, milloin ja kenelle se tapahtui? Mittarit ovat hyödyllisiä pelin kehittäjille. Niillä voidaan esimerkiksi seurata, pelaavatko pelaajat peliä kehittäjien tarkoittamalla tavalla. [1, s. 23.]

Mittarit ovat aina pelikohtaisia muutamaa poikkeusta lukuun ottamatta. Esimerkkinä pelisession pituus on tällainen poikkeus.

Pelattavuusmittarit voidaan jakaa vielä kolmeen eri alakategoriaan [1, s. 24]:

1. Pelitapahtumat. Kaikki pelin sisäiset tapahtumat, jotka tapahtuvat pelaajalle. Suurin osa tapahtumista kuuluu tähän kategoriaan. Yleensä mukana on spatiaalinen ja ajallinen tieto, missä ja milloin.

2. Käyttöliittymätapahtumat. Vuorovaikutus pelin valikoiden ja asetuksien kanssa.

3. Systeemitapahtumat. Pelimoottorin ja sen sisäisten systeemien tapahtumat. Esimerkiksi tekoälyn ohjaamien vihollisten tapahtumat.

Mahdollisia mittareita on melkeinpä ääretön määrä, eikä kaikkia pelin tapahtumia ole järkevää tallentaa tietokantaan. Pelin suunnitteluvaiheessa olisi hyvä miettiä, mitä mittareita haluaa mitata. Uusien mittareiden lisääminen myöhemmin on mahdollista päivityksellä.

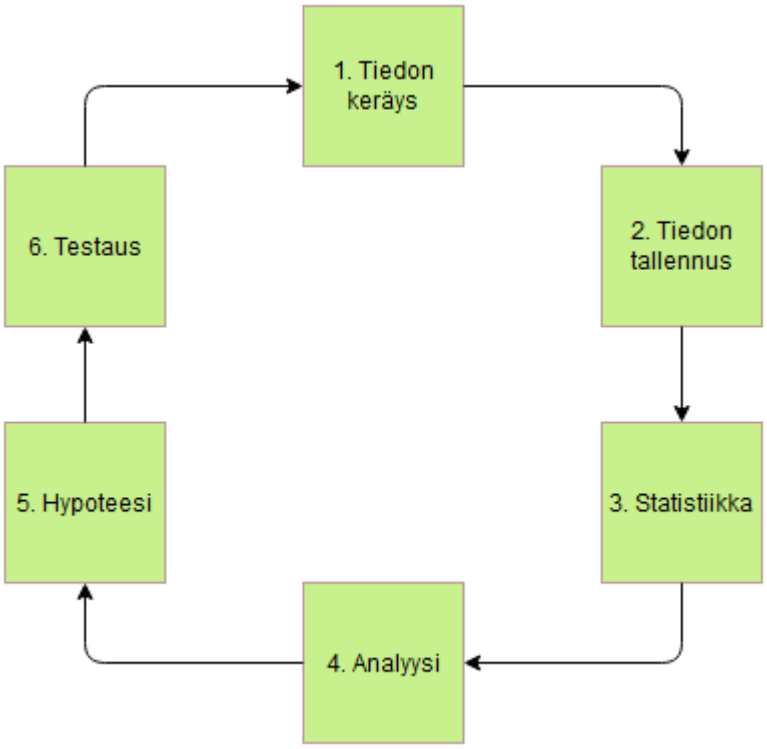
## 2.2 Analysoiminen pelisuunnittelun näkökulmasta

Pelitapahtumien seuraaminen ja analysoiminen on hyödyllistä kaiken kokoisille pelistudioille. Tällöin suunnittelijoiden ei tarvitse arvailla, miten pelaajat käyttäytyvät. Analysoinnin avulla voidaan tutkia, että pelin ydinpelattavuus on kunnossa. Pelitapahtumien analysoinnissa työskennellään vaikeasti ennustettavan ihmiskäyttäytymisen kanssa. [3.]

Analysointi myös paljastaa pelin mahdolliset pullonkaulat, kuten huonon perehdytyksen, tai jos pelaajat suosivat tiettyä asetta muiden sijaan. A/B-testaaminen on myös hyvä tapa, kun halutaan tutkia kahta eri vaihtoehtoa.

Kaikkien mahdollisten mittareiden tallentaminen telemetriapalvelun tietokantaan ei ole käytännöllistä. Se mitä halutaan mitata ja analysoida, tulisi päättää jo pelin suunnitteluvaiheessa. Jokaisella projektilla on erilaiset tarpeet. [3.]

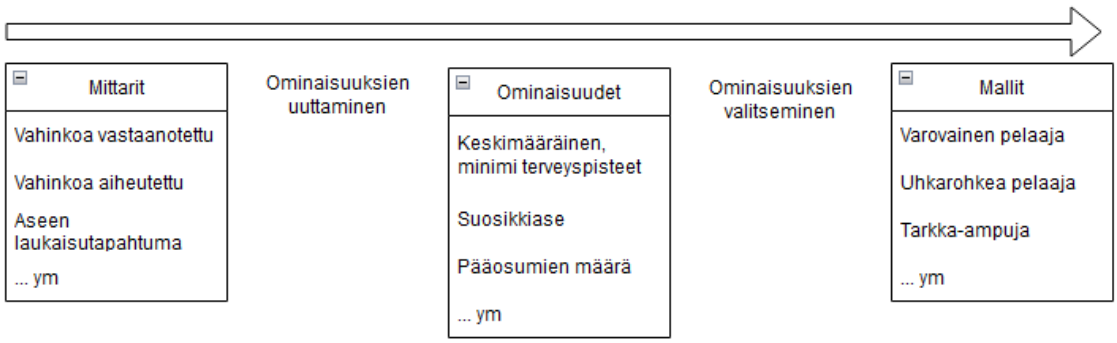
Hypoteeseja muodostamalla voidaan kysymyksen perusteella päätellä, mitä mittareita tarvitaan vastauksen luomiseksi. Esimerkiksi jos halutaan tietää pelaajan yleisin kuolinsyy, kehittäjät tietävät jo ennalta, mitä mittareita tulisi seurata ja analysoida [1, s. 258]. Kuvassa 1 esitetään iteratiivinen prosessi, jonka avulla hypoteesi voidaan muodostaa ja testata [4]. Prosessi voidaan toistaa, kunnes päästään toivottuun lopputulokseen.




Kuva 1. Iteratiivinen sykli, jolla testataan hypoteesia [4.]

Ensimmäisessä vaiheessa tieto kerätään, eli käytännössä peli lähettää sen telemetriapalvelimelle. Tieto tallentuu palvelimelle, jolloin toinen vaihe toteutuu. Kolmannessa vaiheessa tietokannasta haetaan tutkittavaan asiaan relevantti tieto. Neljännessä vaiheessa tieto analysoidaan. Viidennessä vaiheessa muodostetaan analyysin perusteella hypoteesi, jota halutaan testata. Lopuksi suoritetaan testaus, joka yleensä pelien tapauksessa tarkoittaa pelin päivittämistä. Tietyn aikavälin jälkeen prosessi voidaan toistaa, jolloin voidaan analyysin avulla nähdä, pitikö hypoteesi paikkaansa. Tarvittaessa voidaan muodostaa uusi hypoteesi ja testaus, kunnes päästään haluttuun lopputulokseen. [4.]

Tutkiva lähestymistapa pyrkii muodostamaan olemassa olevasta datasta malleja, jotka jakavat pelaajat erilaisiin ryhmiin. Esimerkiksi rauhallisesti pelaavat pelaajat voidaan eritellä enemmän suoran toiminnan pelaajista. Mallien muodostaminen ei ole triviaali tehtävä. Se on iteratiivinen prosessi, joka vaatii tutkimista ja algoritmien säätämistä [1, s. 259]. Mittareista uutetaan ominaisuuksia, joiden avulla luodaan malleja. Malleja voidaan muodostaa käsin tai tekoälyn avulla. Kuvassa 2 esitettävässä prosessissa näytetään, miten malleja voidaan muodostaa käsin.




Kuva 2. Mallien muodostamisprosessi [1, s. 265]

Vaihtoehtoinen tapa muodostaa malleja on tehdä kuvitteellisia persoonia, jotka määritellään tiettyjen ominaisuuksien perusteella. Valven vuonna 2008 julkaisemassa Left 4 Dead -pelissä pelaajien persoonat luodaan pisteyttämällä neljä eri kategoriaa. Ne ovat ominaisuuksia, jotka muodostuvat kategoriaan sopivien mittarien avulla. Pelaajat saavat pisteiksi joko plus tai miinus sen perusteella, yltävätkö pisteet ennalta määrätyn perusviivan yli vai jäävätkö ne sen alle. Pisteyttämällä vain kahdella arvolla prosessi yksinkertaistuu huomattavasti, vaikka se menettääkin vähän tarkkuutta. Taulukossa 1 nähdään, miten pelaaja, joka selviytyy ja tappaa keskimääräistä enemmän, mutta auttaa ja parantaa muita vähemmän, luokitellaan Ramboksi. [1, s. 267–276.]

Taulukossa 1 ei nähdä kaikkia mahdollisia kombinaatioita, mutta jos jokin kombinaatio myöhemmin nousee suosituksi, siitä voidaan tehdä uusi malli ja nimetä se sopivasti.

Taulukko 1. Left 4 Dead -pelin persoonat [1, s. 275]

| | Karkuri | Rontti (eng. grunt) | Samarialainen | Lääkäri | Rambo | Punainen risti | Ekspertti | Aloittelija |
|---|---|---|---|---|---|---|---|---|
| Selviytyä | + | - | - | - | + | - | + | - |
| Tappaa | - | + | - | - | + | - | + | - |
| Auttaa | - | - | + | - | - | + | + | - |
| Parantaa | - | - | - | + | - | + | + | - |

## 2.3 Tiedon louhinta, tekoäly ja datan visualisoiminen

Tiedon louhinnalla yritetään löytää arvokasta tietoa sekä toistuvia kaavoja isosta määrästä dataa [1, s. 207]. Tiedon louhinnan apuna voidaan käyttää tekoälyä, sillä se on erinomainen tunnistamaan malleja ja riippuvuuksia datasta.

Tiedon louhinnan ja tekoälyn yhdistäminen kokonaisuudeksi, jota voi oikeasti hyödyntää, vaatii valtavasti panostusta ja aikaa. Silti se ei välttämättä anna vastauksia kysymyksiin, sillä pelaajat käyttäytyvät tavalla, jota on vaikea ennustaa. [5.] Tässä työssä ei tutkittu tiedon louhinnan ja tekoälyn mahdollisuuksia käytännössä ja teoriassakin se mainitaan vain ohimennen.

**Datan visualisoimisella** tieto saatetaan helposti ymmärrettävään muotoon. Yleinen työnkulku on, että data kerätään, muutetaan, visualisoidaan, analysoidaan ja lopuksi esitetään ulkopuoliselle yleisölle [1, s. 408]. Erilaisia tapoja visualisoida on useita. Yleisimpiä ovat esimerkiksi erilaiset pylväät, kuvaajat, ympyrädiagrammit sekä taulukot.

Spatiaalisessa mittarissa on sijaintitieto mukana. Yleensä myös aika on tiedossa, mikä tekee siitä erittäin hyödyllisen. Esimerkiksi pelaajan liikkeistä on mahdollista muodostaa graafinen lämpökartta tai rata.

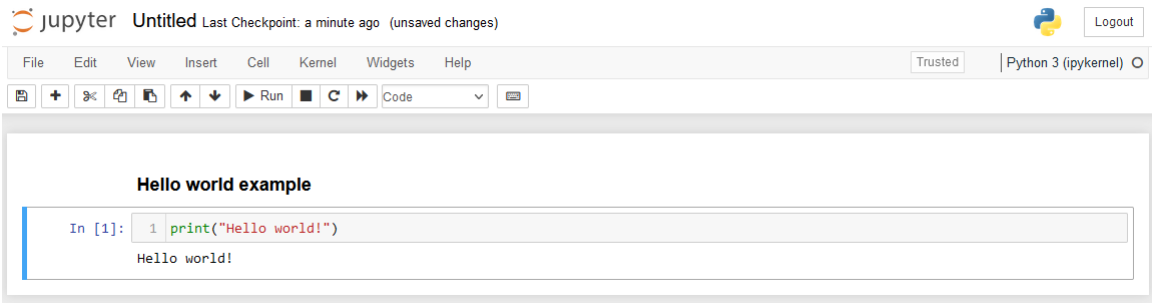
Spatiaalinen visualisointi on helposti ymmärrettävää kaikkien osapuolten näkökulmasta. Sen avulla voi löytää pelin sisäisiä pullonkauloja. Esimerkiksi jos pelin ensimmäisessä kentässä kestää pelata kauemmin kuin kehittäjät suunnittelivat, spatiaalinen visualisointi auttaa näyttämään pelaajien sijainnit tietyllä ajanhetkellä, mikä auttaa ratkaisemaan ongelman [1, s. 366]. Suurin osa pelien tapahtumista on spatiaalista ja ajallista, sillä asiat tapahtuvat pelimaailmassa tietyssä sijainnissa tiettyyn aikaan.

# 3 Jupyter Notebook ja Unreal Engine

Jupyter Notebook on avoimen lähdekoodin ohjelmisto, jota voi käyttää dokumenttien luomiseen ja jakamiseen. Dokumentit voivat sisältää koodia, yhtälöitä, visualisointeja ja tekstiä. Dokumentteja käytetään selaimella. [6.]

Dokumentin koodi ajetaan kernelissä, joka on niin sanottu ”laskentakone”. Oletuksena dokumentti käyttää Python-ohjelmointikieltä, ja se oli käytössä myös tässä työssä. Kerneliä ei tarvitse itse käynnistää, vaan se käynnistetään taustalle, kun dokumentti aukaistaan. [7.]

Jupyter Notebook -dokumentissa on käyttöliittymä, jolla voidaan hallita itse dokumenttia. Dokumentti voidaan pitää palvelimella, josta sen voi aukaista selaimella tai sen voi myös ajaa lokaalisti samalla laitteella selaimen kanssa. Kuvassa 3 näytetään esimerkki käyttöliittymästä, jossa on yksinkertainen pätkä koodia.




Kuva 3. Jupyter Notebookin käyttöliittymä selaimessa

Dokumentti koostuu soluista, jotka ovat yleensä joko ohjelmakoodi- tai Markdown-soluja. Markdown-solut on tarkoitettu muistiinpanojen, otsikoiden ja muun ei-ohjelmakoodin näyttämiseen. Ohjelmakoodisolut ovat koodia varten, ja ne voidaan ajaa yksitellen tai kaikki kerralla. [8.]

**Unreal Engine** on Epic Gamesin vuonna 1998 julkaisema pelimoottori. Sen viimeisin vakaa julkaistu versio on 5.1.1, jota käytettiin tässäkin työssä. Unrealin kanssa voi ohjelmoida sen omalla visuaalisella skriptauskielellä Blueprinteillä sekä C++-kielellä. Epic Games käyttää moottoria kehittämässään Fortnite-pelissä, minkä ansiosta moottorin uusia ominaisuuksia testataan miljoonien pelaajien voimin. Moottoria voi käyttää ilmaiseksi tiettyyn rajaan asti, minkä jälkeen maksetaan ehtojen mukainen prosenttiosuus myyntivoitoista. [9.] Tämän vuoksi moottori on suosittu myös indie-kehittäjien ja harrastelijoiden keskuudessa.

Tässä työssä käytettiin Unreal Engineä lisäosan kehittämiseen sekä palvelun testaamiseen. Lisäosat ovat kokoelma ohjelmakoodia ja dataa, jota kehittäjät voivat lisätä omiin projekteihinsa. Niillä voi laajentaa pelimoottorin ominaisuuksia tai ne voivat sisältää peleihin tarkoitettua toiminnallisuutta. Lisäosat koostuvat yhdestä tai useammasta moduulista. Moduuli voi olla joko pelkästään editorissa tai myös pelissä toimiva. [10.]

Tässä työssä toteutettiin lisäosa, joka koostuu kahdesta moduulista. Toinen on vain editorissa käytössä ja toinen on pelimoduuli.

# 4 Käytännön toteutus

Työn lähtökohtana oli jo valmiiksi toteutettu telemetriapalvelin. Palvelimelle pystyi lähettämään dataa tietokantaan sekä tekemään kyselyitä tietokannasta.

Työn tavoitteena oli toteuttaa palvelu, jota Kajaanin ammattikorkeakoulun pelialan opiskelijat voivat käyttää omissa projekteissaan pelien analysointiin.

Suunnitteluvaiheessa todettiin palvelun tarvitsevan kolme pääkomponenttia:

- Telemetriapalvelimen, jolle voi lähettää dataa sekä suorittaa kyselyitä peliä varten.
- Pelimoottorin lisäosan, jonka avulla voi lähettää dataa palvelimelle.
- Analysaattorin, jonka avulla voi tarkastella pelikohtaista dataa ja analysoida sitä.

Alla on kerrottu valituista tekniikoista sekä työn toteutuksesta.

**Telemetriapalvelin** oli jo toteutettu toisen opiskelijan toimesta ennen tämän työn aloittamista. Se on mikropalvelu, joka toimii Linux-pohjaisella palvelimella. Pelin käynnistyessä analysoitava peli ottaa yhteyttä palvelimeen ja uusi pelisessio aloitetaan. Pelin aikana peli lähettää tapahtumia, jotka tallennetaan palvelimen tietokantaan. Palvelimelle voi suorittaa kyselyitä, joissa pelikohtaisia tapahtumia voi lukea analysointia varten.

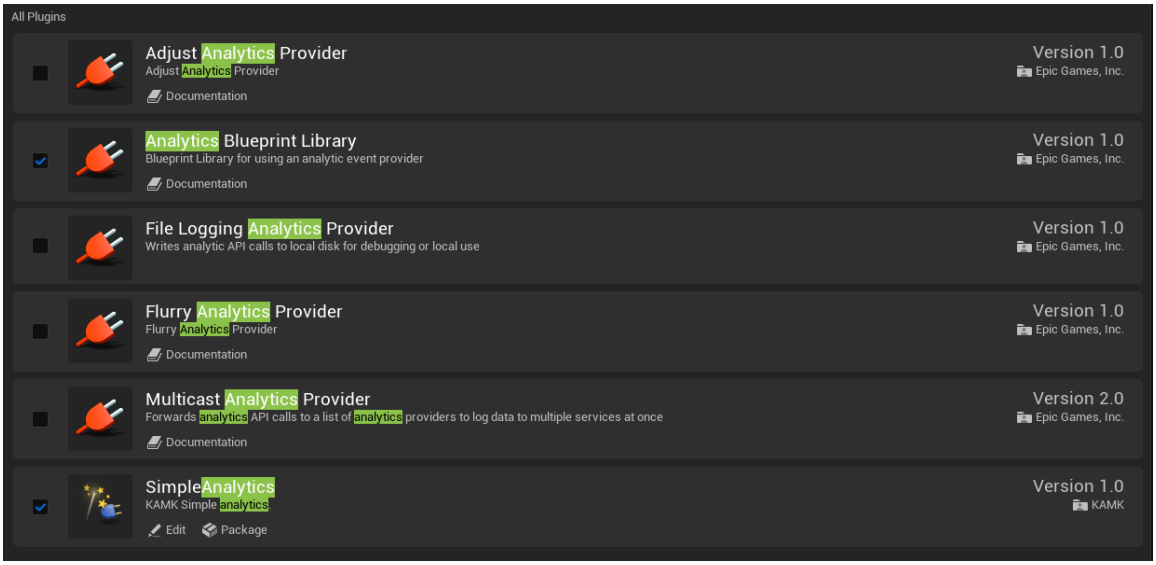
## 4.1 Unreal Engine -lisäosa

Työtä varten toteutettiin Unreal Enginelle (UE) lisäosa, jota voi käyttää tiedon lähettämiseen telemetriapalvelimelle. UE tarjoaa abstraktin luokan, jonka päälle lisäosa on rakennettu. Tämän ansiosta on mahdollista vaihtaa analytiikkapalvelun tarjoajaa ilman, että tarvitsee muuttaa pelin sisäistä koodia. Pelinkehittäjien tarvitsee vain valita oikea palveluntarjoaja. [11.]

Tässä työssä toteutettu lisäosa nimettiin Simple Analytics -nimellä. Se koostuu kahdesta eri moduulista. Toinen on pelin aikana käytettävä moduuli ja toinen on vain editorissa käytössä oleva moduuli, jonka avulla voi muokata lisäosan asetuksia. Kun peli paketoidaan omaksi sovellukseksi, asetukset tallennetaan peliin eikä niitä voi loppukäyttäjä muokata. Molemmat moduulit kehitettiin C++-kielellä.

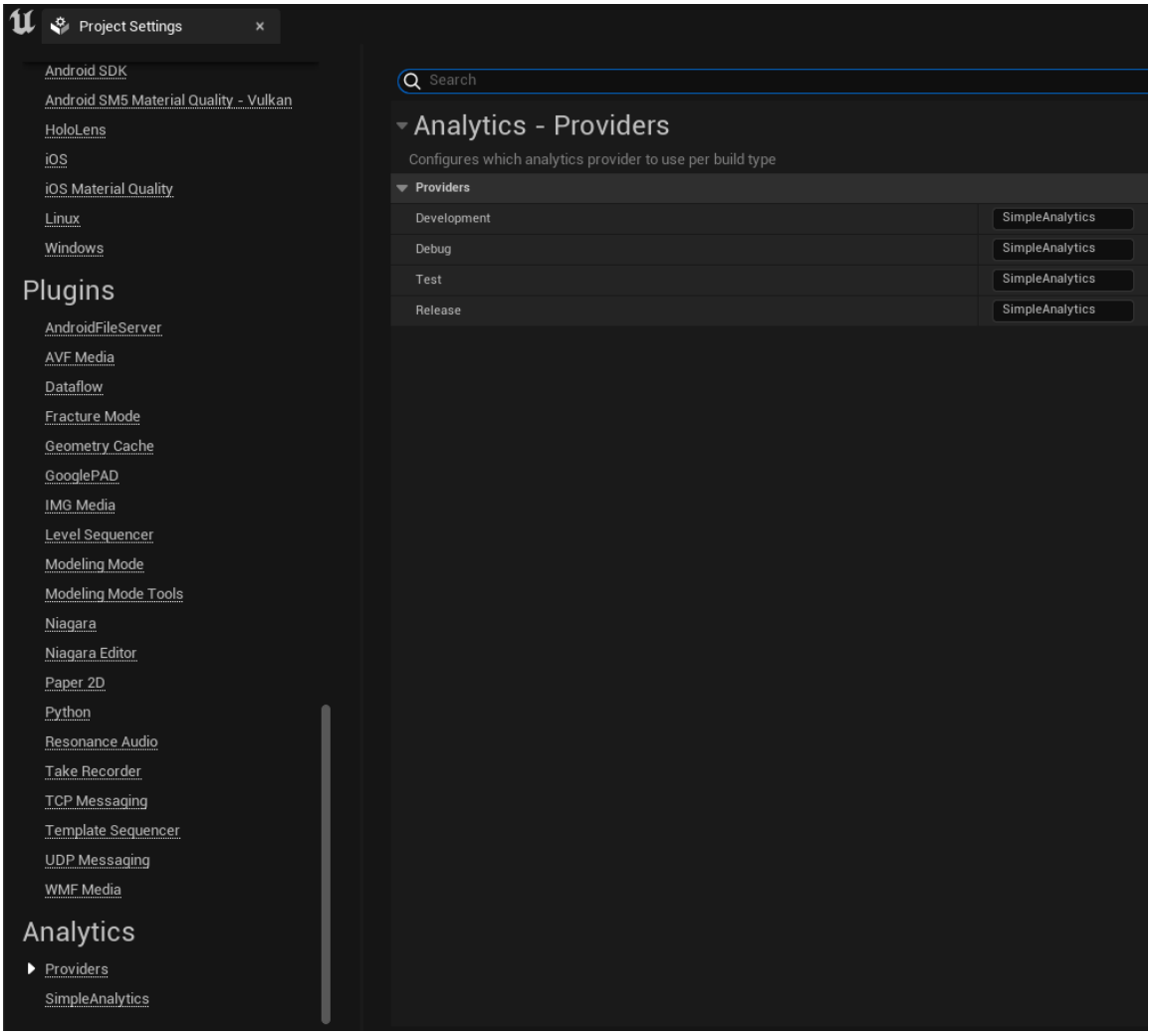
Pelinkehittäjä voi käyttää geneeristä Blueprint Analytics Library -lisäosaa, joka on Epic Gamesin kehittämä, ja se on tarkoitettu toimimaan siten, että ns. pellin alla olevasta toteutuksesta ei tarvitse tietää. Sen avulla voidaan aloittaa ja lopettaa istunto sekä lähettää tapahtumia. [12.]

Blueprint Analytics Library -lisäosaa voi myös käyttää C++-kielen kautta, joten se ei pakota kehittäjää pelkästään Unrealin Blueprint-ohjelmointikieleen. Kuvassa 4 näytetään tarvittavien lisäosien päälle laittaminen.



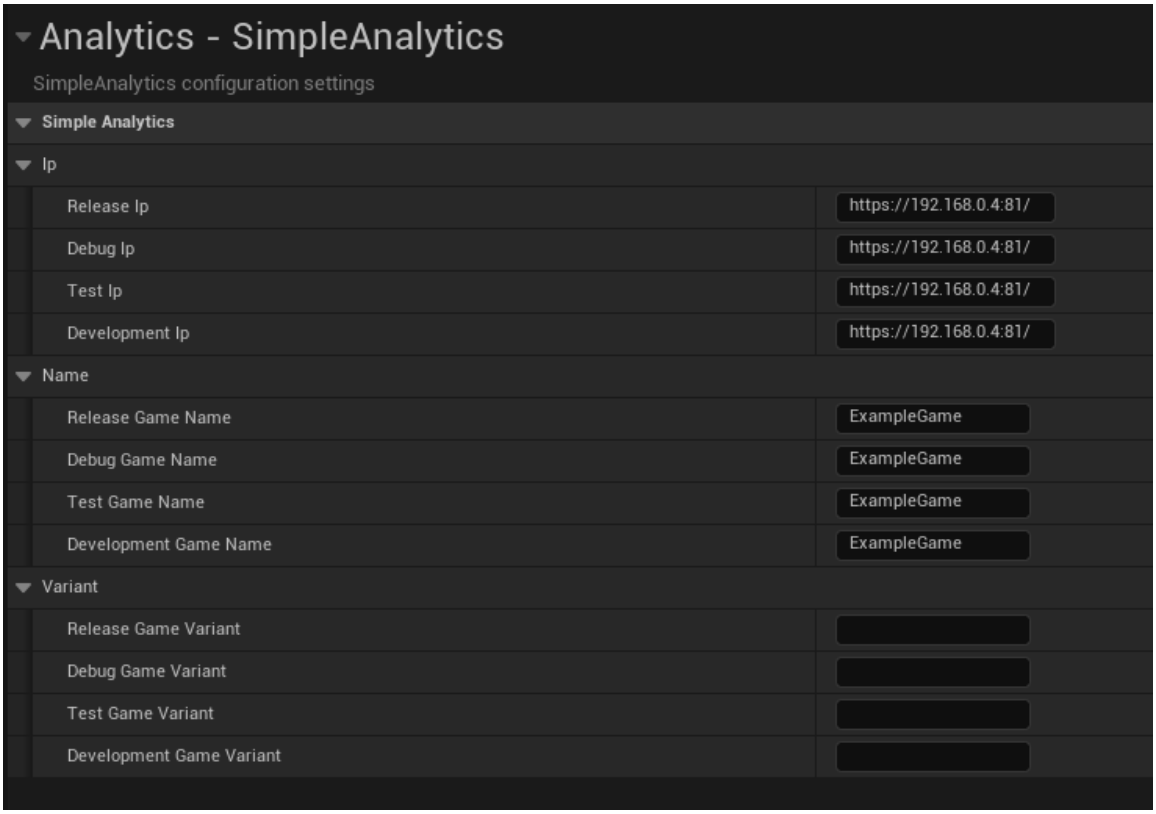

Kuva 4. Tarvittavien lisäosien päälle laittaminen Unrealissa

Peliprojektin asetuksista voidaan valita palveluntarjoaja kirjoittamalla halutun lisäosan nimi sopivaan kenttään. Kuvassa 5 havainnollistetaan, kuinka valitaan työssä toteutettu palveluntarjoaja.



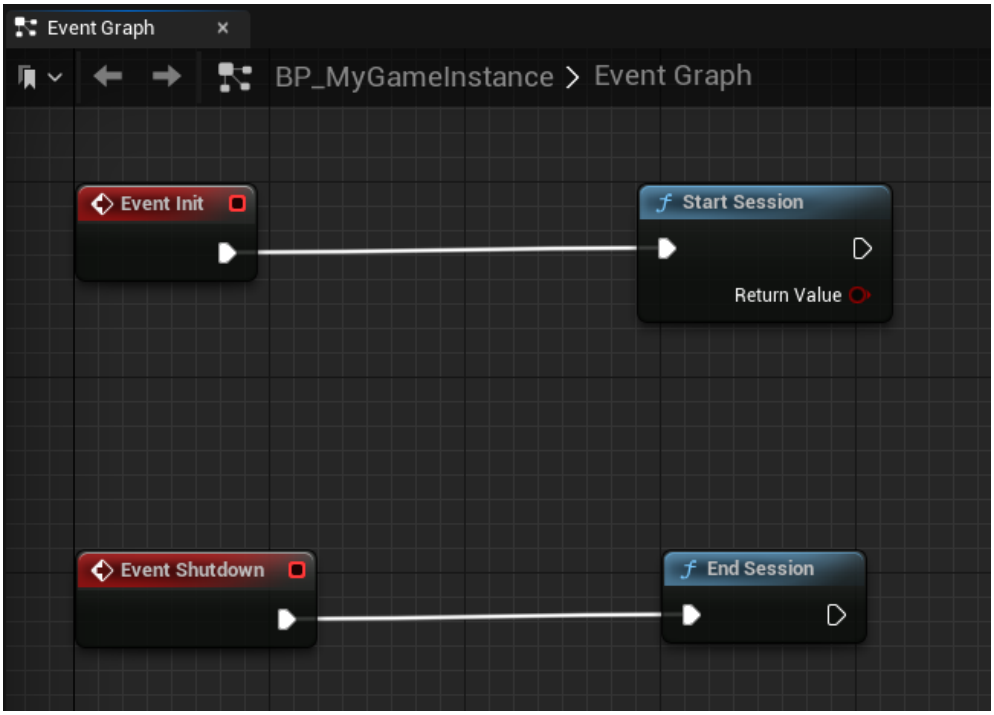

Kuva 5. Analytiikan palveluntarjoajan valitseminen Unrealin projektin asetuksissa

Lisäosan asetuksista voidaan määrittää asetuksia erikseen eri ympäristöille. Variant-muuttujan avulla pelillä voidaan pitää sama nimi, mutta analysoitaessa voidaan suodattaa tapahtumia Variant-muuttujan mukaan. Tämä helpottaa A/B-testaamista, jossa halutaan testata eri variaatioita esimerkiksi samasta kentästä, mutta eri määrällä vihollisia. Kuvassa 6 esitetään lisäosan muokattavat asetukset.



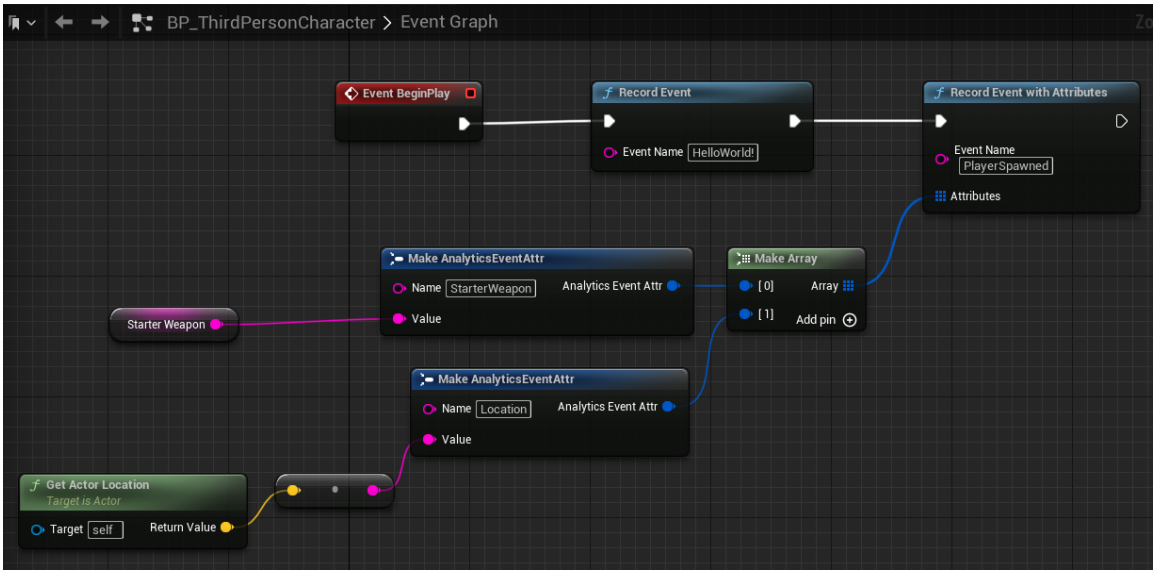

Kuva 6. Simple Analytics -lisäosan asetuksien määrittäminen Unreal-editorissa

Yleensä istunto aloitetaan käynnistäessä peli ja lopetetaan suljettaessa peli. Kentän aloittamiselle ja lopettamiselle kannattaa olla omat tapahtumat, jotta istuntoja ei aloiteta ja lopeteta jatkuvasti. Se myös helpottaa analysointia, sillä tapahtumat voidaan suodattaa paremmin. Kuvassa 7 havainnollistetaan, kuinka istunto voidaan aloittaa ja lopettaa.




Kuva 7. Istunnon aloittaminen ja lopettaminen Unrealin Blueprint-graafissa

Tapahtumilla voi olla pelkkä nimi tai ne voivat sisältää yhden tai useamman parametrin. Kuvassa 8 näytetään, kuinka tapahtuma voidaan lähettää. Parametrien määrää tai esitystapaa ei ole ennalta määrätty, vaan pelinkehittäjien tulee suunnitella ne itse omaan peliin sopiviksi. Palveluntarjoajasta riippumatta kannattaa aina varmistaa, mikä on maksimi parametrien määrälle. Työssä käytetty telemetriapalvelin antaa tapahtumille aikaleimat niiden saapuessa, joten niitä ei tarvitse erikseen lähettää.

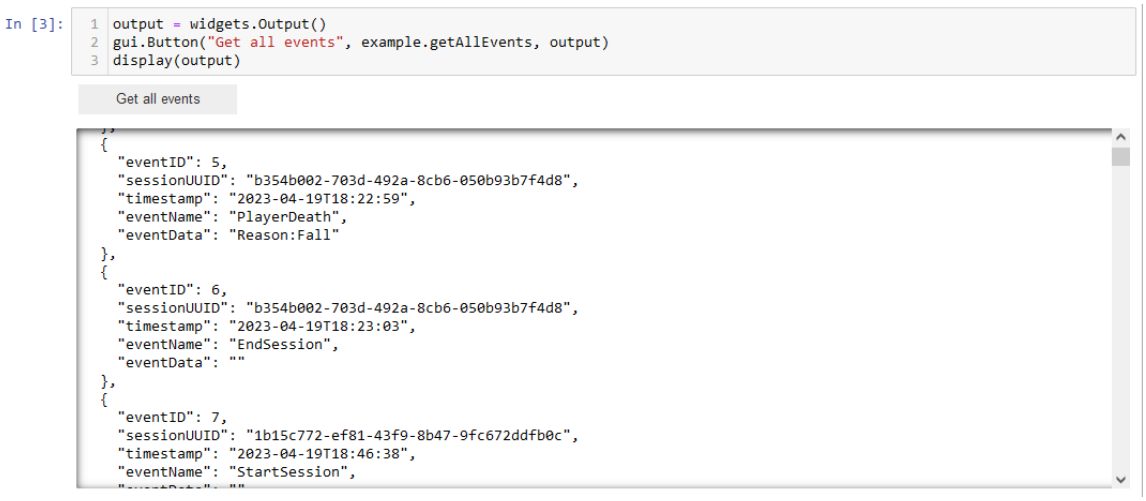



Kuva 8. Tapahtuman lähettäminen. Ensimmäinen tapahtuma on ilman parametreja ja toisessa on kaksi parametria.

## 4.2 Analysaattori

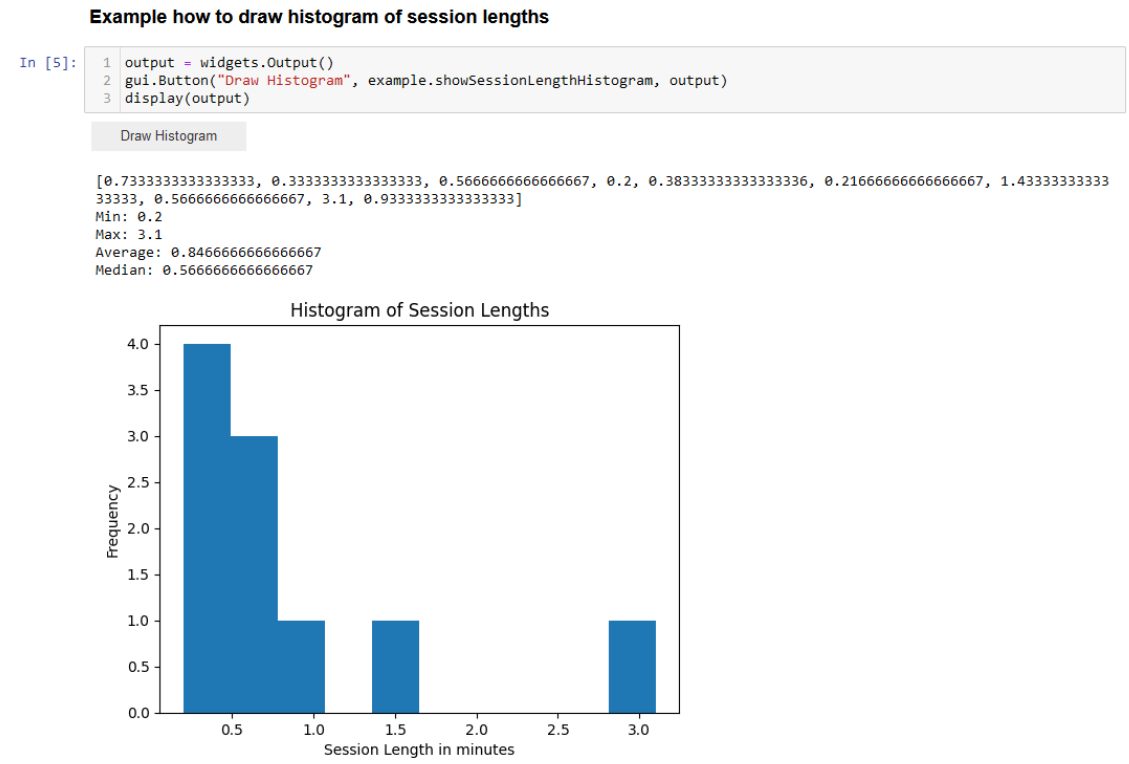
Tiedon hakuun palvelimelta ja sen analysointiin käytettiin Jupyter Notebookia. Analysaattori olisi mahdollista pystyttää palvelimelle, mutta tässä työssä sitä ei tehty. Analysaattoria on helpompi ja nopeampi kehittää, kun sitä voidaan ajaa ja muokata samalla laitteella.

Analysaattorilla on mahdollista hakea kaikki pelin tapahtumat kerralla tai suodattaa ne tietylle istunnolle. Tämä helpottaa analysaattorin kehittämistä, kun tapahtumia voi tarkastella ensin tekstimuodossa. Kuvassa 9 näytetään kaikkien tapahtumien haku.



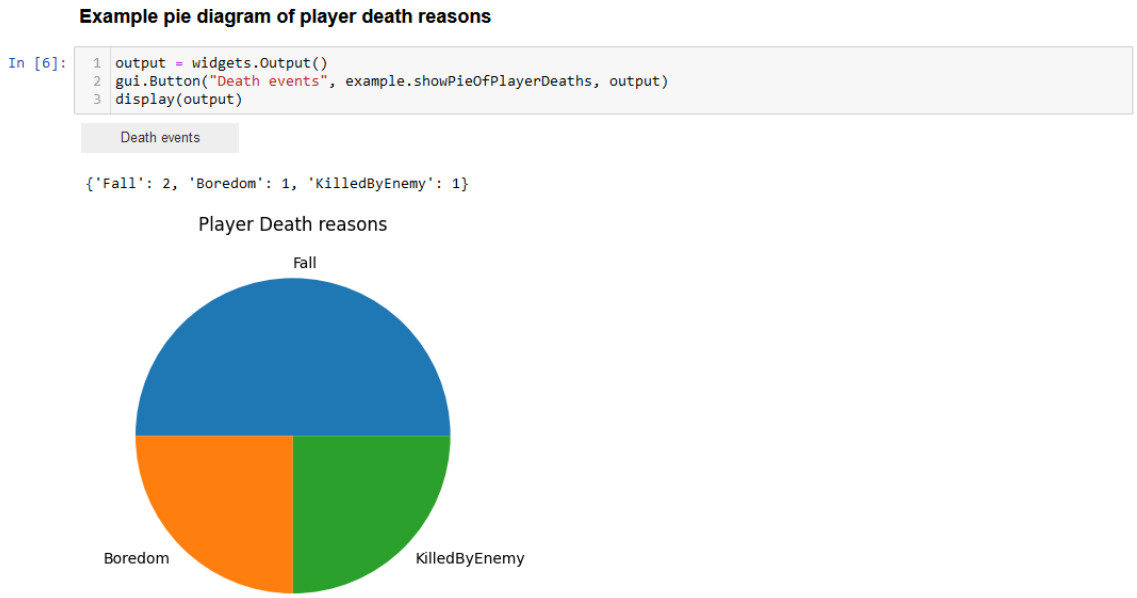

Kuva 9. Kaikkien pelin tapahtumien haku

Analysaattorille toteutettiin muutama yleinen visualisointi. Esimerkiksi peli-istuntojen pituus on yleinen mittari, jota tutkitaan jokaisessa pelissä. Kuvassa 10 näytetään yksi tapa esittää istunnot visuaalisesti.



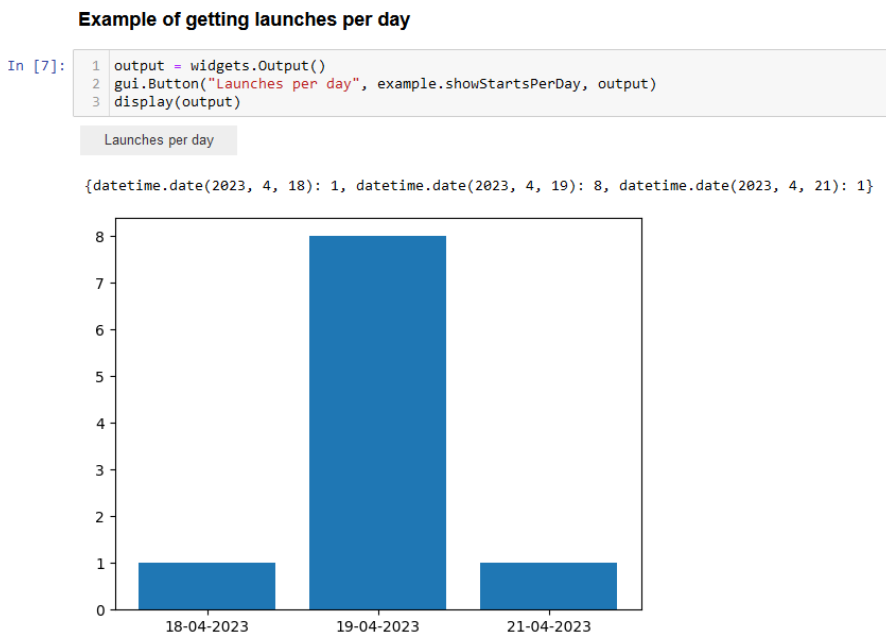

Kuva 10. Peli-istuntojen pituuden esittäminen graafisesti

Erilaiset ympyrädiagrammit ovat myös hyödyllisiä, sillä niiden avulla esitettävä asia hahmottuu nopeasti ja selkeästi. Tässä harjoitustyössä näytettiin, kuinka pelaajan kuoleman syy voidaan piirtää ympyrädiagrammiin. Tästä on helppo heti huomata, jos jokin syy esiintyy toisia huomattavasti enemmän. Kuvassa 11 esitetään ympyrädiagrammi, jossa tarkastellaan pelaajan kuolemaan johtaneita syitä.



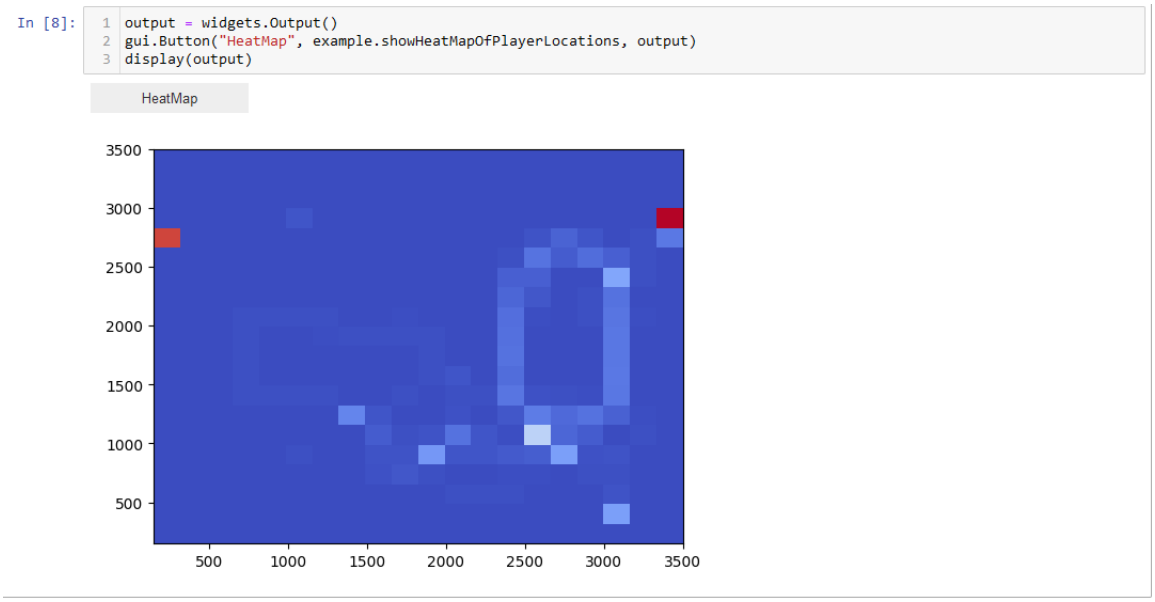

Kuva 11. Ympyrädiagrammi, jossa näkyvät pelaajan kuoleman syyt

Pelin käynnistykset päiväkohtaisesti ovat hyödyllinen tieto jokaiselle pelille, sillä niiden avulla näkee, pelaako peliä edes kukaan. Tässä työssä se toteutettiin yksinkertaisesti, mutta oikea peli vaatinee suodatuksen kuukausien perusteella, etteivät kaikki päivät tule samaan graafiin. Kuvassa 12 näytetään pylväskaavio, jossa käynnistykset on visualisoitu.




Kuva 12. Pelin käynnistämismäärät päiväkohtaisesti

Työssä toteutettiin yksinkertainen lämpökartta pelaajan liikkeistä. Kartta on helppo skaalata oman pelin kentän rajojen mukaan, joten siitä on vaivatonta hahmottaa, missä pelaajat liikkuvat pelin aikana. Jos pelaaja jää paikalleen pitkäksi aikaa, se vääristää kartan värejä. Olisikin hyvä suodattaa paikkatietoa poikkeustilanteiden poistamiseksi. Kuvassa 13 esitellään työssä käytetty lämpökartta, joka näyttää pelaajien liikkeet pelaamisen aikana. Lämpökartta on kuvattu pelimaailmassa ylhäältäpäin siten, että y-akselilla on pelimaailman suunta eteenpäin ja x-akselilla oikealle. Numerot akseleilla kertovat pelimaailman koordinaatit x- ja y-akseleilla.




Kuva 13. Pelaajan liikkeet visualisoiva lämpökartta

## 4.3 Esimerkkisovellus ja palvelukokonaisuus

Esimerkkisovellus on Unreal Enginellä luotu sovellus, jonka käyttöliittymästä voi lähettää tapahtumia telemetriapalvelimelle. Sen tarkoituksena oli olla esimerkkinä siitä, kuinka lisäosaa voidaan käyttää pelien kanssa ja kuinka lisäosa voidaan liittää omaan peliprojektiin. Sillä on myös helppo testata eri tapahtumia nopeasti, kun analysaattoria kehitetään. Tapahtuma voi sisältää useamman parametrin tai olla parametriton.

Sovellus toteutettiin kokonaan Unrealin Blueprint-ohjelmointikielellä, jotta se on helposti tutkittavissa eikä se vaadi ohjelmakoodin kääntämiseen tarvittavia työkaluja. Simple Analytics -lisäosa käännettiin kehitysvaiheessa binäärimuotoon, minkä ansiosta se voitiin lisätä sovellukseen ilman, että sitä tarvitsee kääntää uudestaan. Tämä on tärkeää siksi, että osa kehittäjistä käyttää vain visuaalista ohjelmointikieltä eikä heillä välttämättä ole edes työkaluja C++-koodin kääntämiseen. Kuvassa 14 näytetään esimerkkisovelluksen käyttöliittymä.


Kuva 14. Esimerkkisovellus, jolla voi lähettää tapahtumia palvelimelle

Palvelua oli helppo testata pystyttämällä palvelin virtuaalikoneeseen ja aukaisemalla analysaattori selaimessa sekä ajamalla testisovellus pelimoottorissa. Analysaattoria on helppo kehittää tällä tavalla, koska pelitapahtumat päivittyvät palvelimelle reaaliajassa.

Kokonaisuuden tavoitteena oli luoda palvelu, joka on helppo lisätä ja käyttää opiskelijoiden projekteissa. Telemetriapalvelin ja Unreal Engine -lisäosa eivät ota kantaa siihen, minkälaisia tapahtumia peli haluaa lähettää ja analysoida, vaan ne ovat niin sanotusti geneerisiä. Esimerkkisovellus ja Jupyter Notebook -analysaattori kehitettiin rinta rinnan, koska analysaattorin täytyy tietää, kuinka käsitellä tietyn pelin tapahtumia. Itse analysaattori ei siis ole geneerinen, jotta se toimisi sellaisenaan jokaisessa erilaisessa pelissä. Se on enemmänkin yksinkertainen esimerkki siitä, kuinka kehittää analysaattori oman pelin tarpeisiin.

# 5 Yhteenveto

Työn tavoitteena oli luoda palvelukokonaisuus, jota pelialan opiskelijat voivat käyttää omissa projekteissaan pelitapahtumien analysointiin. Kokonaisuuteen kuului pelimoottorin lisäosa, selaimella toimiva analysaattori sekä niiden toiminnan yhdistäminen palvelimeen, joka toimii tietokantana. Pelimoottorilla tehtiin yksinkertainen testisovellus, jolla lisäosan toimintaa voidaan testata sekä demonstroida sen liittäminen omaan projektiin.

Työn toteutus eteni alkuun varsin hyvin, mutta hidastui loppua kohden. Teorian tutkimiseen käytettiin paljon aikaa, mutta silti itse analysaattori jäi varsin yksinkertaiseksi. Toteutus aloitettiin telemetriapalvelimen toiminnan testaamisella sekä pelimoottorin lisäosan kehittämisellä. Seuraavaksi toteutettiin pelimoottoriin testisovellus, jolla oli helppo lähettää tietoa palvelimelle ja todeta lisäosan toimivuus, kun se liitetään toiseen projektiin. Lopuksi kehitettiin analysaattori, jolla voidaan hakea ja käsitellä tietoa palvelimelta. Analysaattorille keksittiin tapahtumia, joiden avulla voidaan havainnollistaa sen toimintaa.

Työn loppuvaiheessa palvelukokonaisuutta testattiin pidemmillä peli-istunnoilla ja lähettämällä useita tapahtumia sekunnissa palvelimelle. Suorituskyky- tai vakausongelmia ei havaittu. Palvelua ei testattu oikeassa peliprojektissa, joten testaaminen oli hieman puutteellista. Pelimoottorin lisäosan liittämistä toisiin projekteihin testattiin, eikä siinä havaittu puutteita.

Nykyisessä muodossaan palvelukokonaisuus tarjoaa hyvän pohjan, jota voidaan hyödyntää eri peliprojekteissa. Pelimoottorin lisäosa voidaan asetuksia muuttamalla saada toimimaan eri peleissä. Analysaattori ei sellaisenaan toimi jokaisessa pelissä, mutta se toimii yleishyödyllisenä esimerkkinä, jota voi jatkokehittää oman pelin tarpeisiin. Palvelun käyttäminen pidempiaikaisessa projektissa saattaa paljastaa puutteita tai jatkokehitysideoita, jotka tässä työssä jäivät huomaamatta.

Lähteet


1. M. Seif El-Nasr. Game Analytics, Maximizing the Value of Player Data. Springer-Verlag London 2013.

2. J. Frankenfield. Data Analytics: What It Is, How It's Used, and 4 Basic Techniques. [Internet]. 2023 [viitattu 28.4.2023]. Saatavilla: https://www.investopedia.com/terms/d/data-analytics.asp

3. N. Lovato. Game Analytics from A game Designer's Perspective. [Internet]. 2015 [viitattu 9.4.2023]. Saatavilla: https://gameanalytics.com/blog/gameanalytics-game-designers-perspective/

4. D. Kennerly. Better Game Design Through Data Mining. [Internet]. 2003 [viitattu 28.4.2023]. Saatavilla: https://www.gamedeveloper.com/design/better-game-design-through-data-mining

5. D. Nozhnin. Predicting Churn: Data-Mining Your Game. [Internet]. 2012 [viitattu 10.4.2023]. Saatavilla: https://www.gamedeveloper.com/design/predicting-churn-data-mining-your-game

6. M. Driscoll. Jupyter Notebook: An Introduction. [Internet]. 2023 [viitattu 11.4.2023]. Saatavilla: https://realpython.com/jupyter-notebook-introduction/

7. A. Ingargiola. What is the Jupyter Notebook? [Internet]. 2015 [viitattu 20.4.2023]. Saatavilla: https://jupyter-notebook-beginner-guide.readthedocs.io/en/latest/what_is_jupyter.html

8. Jupyter Team. The Jupyter Notebook. [Internet]. 2015 [viitattu 20.4.2023]. Saatavilla: https://jupyter-notebook.readthedocs.io/en/stable/notebook.html

9. Wikipedia. Unreal Engine. [Internet]. 2023 [viitattu 21.4.2023]. Saatavilla: https://en.wikipedia.org/wiki/Unreal_Engine

10. Epic Games. Plugins. [Internet]. 2023 [viitattu 21.4.2023]. Saatavilla: https://docs.unrealengine.com/4.27/en-US/ProductionPipelines/Plugins/

11. Epic Games. Instrumenting Your Game. [Internet]. 2023 [viitattu 11.4.2023]. Saatavilla https://docs.unrealengine.com/4.27/en-US/TestingAndOptimization/Analytics/Instrumenting/

12. Epic Games. Blueprint Analytics Plugin. [Internet]. 2023 [viitattu 21.4.2023]. Saatavilla: https://docs.unrealengine.com/4.27/en-US/TestingAndOptimization/Analytics/Blueprints/