Ohjelmoinnin opettelu pelillistämisen ja Minecraftin kautta

Turun yliopisto
Tietotekniikan laitos
TkK-tutkielma
Tietotekniikka
Syyskuu 2024
Lauri Heinonen

TURUN YLIOPISTO
Tietotekniikan laitos

Lauri Heinonen: Ohjelmoinnin opettelu pelillistämisen ja Minecraftin kautta

TkK-tutkielma, 22 s.
Tietotekniikka
Syyskuu 2024

Pelillistäminen (engl. *gamification*) on viime vuosina noussut keskeiseksi työkaluksi erityisesti opetuksessa. Työssä tarkastellaan, miten pelillistämisen elementit voivat lisätä oppilaiden motivaatiota ja sitoutumista ohjelmoinnin oppimiseen. Pelillistämistä on kuitekin mahdotonta tarkastella ilman viitekehyksiä, joiden avulla ymmärrämme pelillistämistä tarkemmin. MDA-viitekehyksen (engl. *mechanics, dynamics, aesthetics, MDA*) avulla pelillistämistä voidaan analysoida kolmella tasolla: mekaniikka, dynamiikka ja estetiikka. Mekaniikka viittaa pelin sääntöihin ja toimintoihin, dynamiikka kuvaa pelaajan ja pelin välistä vuorovaikutusta ja estetiikka käsittelee pelaajan kokemia tuntemuksia ja kokemuksia. Oktalyysi-viitekehys (engl. *octalysis*) puolestaan keskittyy ihmisen sisäisiin motivaatiotekijöihin ja siihen, miten pelillistämisen elementtejä voidaan hyödyntää motivaation lisäämiseksi. Oktalyysi-viitekehystä kuvataan kahdeksankulmiolla, jonka jokainen sivu edustaa yhtä motivaatiotekijää.

Tutkielmassa analysoidaan Minecraftin pelillistämistä molempien viitekehysten avulla. Peliä on käytetty jo vaikkapa matematiikan, ohjelmoinnin ja luonnontieteiden opetuksessa pelillistettynä oppimisympäristönä. Minecraft mahdollistaa esimerkiksi sähköisten piirien sekä logiikkaporttien rakentamisen pelin sisäisen punakiven (engl. *redstone*) avulla. Tutkielmassa tarkastellaan myös Minecraftin eri versioiden ominaisuuksia ja soveltuvuutta ohjelmoinnin opetukseen. Pelistä on kehitetty opetukseen suunnattu versio Minecraft Education, joka mahdollistaa ohjelmoinnin korkealla abstraktiotasolla, käyttäen hiirellä liikuteltavia koodipalikoita. Peliin voi myös asentaa lisäosia (engl. *mod*), jotka mahdollista oikeiden ohjelmointikielien kirjoittamisen pelin sisällä.

Tutkielma korostaa, että pelillistäminen voi olla tehokas työkalu ohjelmoinnin opetuksessa, kunhan se suunnitellaan huolellisesti ja integroituu saumattomasti oppimissisältöön. Minecraft on monipuolinen peli, joka tarjoaa laajan kirjon mahdollisuuksia pelillistämisen hyödyntämiseen. Minecraftin eri versiot ja modit voivat auttaa tekemään ohjelmoinnista helpommin lähestyttävää, mielenkiintoisempaa ja motivoivampaa kaikenikäisille oppijoille.

Asiasanat: pelillistäminen, Minecraft, ohjelmointi, oppiminen, mod

Sisällys

1 Johdanto

Tutkielmassa käsitellään pelillistämistä ja sen soveltamista. Lisäksi tutkitaan myös pelillistämisen viitekehyksiä, kuten oktalyysi (engl. *Octalysis*) sekä *MDA*. Oktalyysin avulla voidaan tarkastella miten hyvin jokin asia vetoaa ihmisen sisäisiin motivaatiotekijöihin. MDA puolestaan keskittyy siihen, miten pelin mekaniikat luovat pelaajaan mieluisia kokemuksia. Tutkielmassa tarkastellaan, miten Minecraft on pelillistetty, ja miten sitä hyödynnetään ohjelmoinnin oppimisessa. Pelillistämisen ansiosta kiinnostuin itse nuorena ohjelmoinnista. Tutkielma keskittyy pitkälti Minecraftiin, koska se on ohjelmoitu Javalla, joka on yksi maailman yleisimmistä ohjelmointikielistä.

Luvussa kaksi käydään läpi, mitä pelillistäminen tarkoittaa. Alaluvuissa keskitytään pelillistämisen viitekehyksiin, joilla voidaan kehittää pelejä tai tarkastella miten hyvin jokin asia on pelillistetty. Luvussa kolme siirrytään Minecraftin ja sen pelillistämisen pariin. Ensimmäisessä alaluvussa keskitytään pelin perusversioon ja sen tarjoamiin elementteihin. Seuraavassa alaluvussa tarkastellaan pelistä julkaistua opetusversiota, joka tarjoaa perusversiota laajemmat työkalut ohjelmoinnin oppimiseen. Viimeinen alaluku keskittyy pelaajan muokkaamaan versioon, jossa on mahdollista ohjelmoida tietokoneita oikeilla ohjelmointikielillä. Lisäksi tarkastellaan, miten pelaaja voi ohjelmoimalla muokata peliä haluamakseen.

1.1 Tutkimuskysymykset

TK1: Mitä pelillistäminen on?

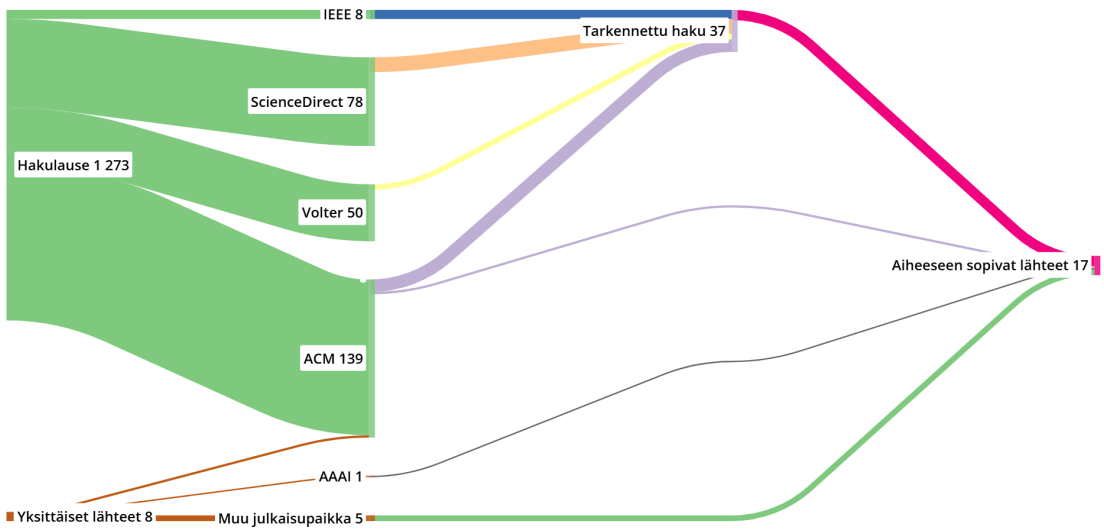
Pelillistäminen on nykyään suosittua useilla eri aloilla lupaavien tulostensa vuoksi, mutta mitä se käytännössä on? Miten pelillistäminen voidaan määritellä, mitä sillä saavutetaan ja miten sitä voidaan hyödyntää? Onko olemassa joitakin elementtejä, jotka eivät ole tyypillisiä peleille, mutta jotka silti tukevat pelillistämistä?

TK2: Miten Minecraft voidaan pelillistää?

Minecraft on yleisesti tunnettu sekä suosittu videopeli. Mikäli sen pelielementit olisi toteutettu huonosti, se ei todennäköisesti olisi saavuttanut nykyistä suosiota. Mitä pelielementtejä Minecraftissa siis on? Miten nämä elementit vaikuttavat pelaajien mielenkiintoon peliä kohtaan? Miten peliä muokkaamalla voidaan pelillistää ohjelmoinnin opetusta?

1.2 Menetelmät ja tiedonhaku

Suurin osa kirjallisuudesta on haettu käyttämällä ensin hakulausetta ”gamification AND minecraft AND programming” useissa akateemisesti luotettavissa tietokannoissa. Tämän jälkeen tietokannan hakutulosten määrästä riippuen hakulausetta on tarkennettu tietokantakohtaisesti, jotta saatujen hakutulosten määrää on kyetty rajaamaan riittävästi. Aineistoa hakiessa tekoälyä (Google Gemini) on käytetty luomaan tiivistelmiä tieteellisistä julkaisuista. Tiivistelmien perusteella aineistoa on karsittu ja valittu ne julkaisut, joita on käytetty tämän työn lähteenä. Kaikki tarkennetun haun tulokset on käyty läpi manuaalisesti valitsemalla aiheeseen sopivat lähteet, joita tällä menetelmällä kertyi 10. Loput lähteet on kerätty varmistaen, että löydetty kirjallisuus on akateemisesti luotettavin, jota kyseisestä aiheesta löytyy. Kuvassa 1.1 on esitetty Sankey-diagrammi tiedonhakuprosessin vaiheista.



Kuva 1.1: Sankey-diagrammi tiedonhaun etenemisestä

2 Pelillistäminen ja sen viitekehykset

Pelillistäminen on pelielementtien ja mekaniikkojen soveltaminen pelien ulkopuolelle. Kyseinen mekaniikka on saanut alkunsa nimensä mukaisesti peleistä, joiden tavoitteena on antaa pelaajalle mielekästä tekemistä motivaation ja sitoutumisen parantamiseksi. Haluttu vaikutus pelaajaan saadaan aikaan soveltamalla pelillisiä elementtejä, kuten pisteitä, tasoja tai palkintoja [1]. Näitä elementtejä voidaan soveltaa pelien lisäksi erilaisiin konteksteihin, kuten oppimiseen, työntekoon ja hyvinvoinnin ylläpitämiseen. Erityisesti oppimisympäristöissä pelillistämistä käytetään parantamaan opiskelijoiden motivaatiota, lisäten heidän sitoutumistaan opittavaan aiheeseen [2].

Pelillistämisen avulla voidaan tehdä abstrakteista ja haastavista asioista, kuten ohjelmoinnista, konkreettisempia ja helpommin lähestyttäviä. Esimerkiksi ohjelmoinnin opettamisessa pelillistäminen voi auttaa pilkkomaan oppimisprosessia pienempiin tavoitteisiin, joista oppija saa välitöntä palautetta ja palkintoja [3]. Pelillistäminen voi ilmetä esimerkiksi kilpailun, pisteytyksen, tasojen tai tulostaulukoiden muodossa. Pisteytyksen avulla voidaan antaa konkreettista palautetta edistymisestä, ja kilpailullinen asetelma kannustaa haastamaan itseään muiden voittamiseksi. Onnistuneista suorituksista voidaan antaa myös saavutuksia ja palkintoja, jotka symboloivat oppimisen edistymistä. On tärkeää huomata, että pelillistäminen ei tarkoita pelkästään pelien lisäämistä oppimisympäristöön. Jotta pelillistäminen olisi tehokasta, edellämainittujen elementtien tulee olla mielekkäällä tavalla integroi-

tuja oppimiskokemukseen siten, että se tukee oppimistavoitteita. [4]

On haastavaa määrittää tarkasti, milloin pelillistämisessä ollaan onnistuttu, sillä siihen vaikuttavat henkilökohtaiset mielipiteet. Pelillistämisen teoria pyritäänkin usein konkretisoimaan teoreettisiksi viitekehyksiksi. Viitekehysten avulla voidaan arvioida, kuinka pelillistämisen eri elementit toimivat sekä kuinka ne voidaan integroida johonkin ulkopuoliseen kontekstiin. Näitä voidaan pitää eräänlaisena näkökulmana, joka ohjaa ajattelua [4]. Ne toimivat myös työkaluna, jolla voidaan arvioida, kuinka hyvin tietty järjestelmä tai prosessi on pelillistetty ja missä määrin se motivoi käyttäjiään. Seuraavassa alaluvussa esitellään pelillistämisen viitekehys, jonka avulla voidaan luoda pelaajalle mielenkiintoisia sekä mukaansatempaavia kokemuksia.

2.1 MDA-viitekehys

Mekaniikkaa, dynamiikkaa sekä estetiikkaa tarkoittava MDA (engl. *Mechanics, Dynamics, Aesthetics*) viitekehys on luotu vuosina 2001-2004 Kaliforniassa järjestetyssä *Game Developers Conference*ssa. Kyseessä on yksi vanhimpia pelillistämiseen liittyviä malleja. Tämän viitekehyksen tavoitteena on ollut helpottaa pelien suunnittelua ja kehitystä kuluttajalle mielekkäämmän ja koukuttavamman lopputuloksen saavuttamiseksi. MDA:n avulla peliä voidaan tarkastella sekä suunnittelijan että pelaajan näkökulmasta. Suunnittelijoiden näkökulma on pilkottu viitekehyksen nimen mukaisesti kolmeen osaan: mekaniikkaan, dynamiikkaan sekä estetiikkaan. Mekaniikkojen avulla voidaan luoda dynaamisia järjestelmiä, jotka vuorostaan luovat esteettisiä kokemuksia pelaajalle. Pelaajan näkökulmasta viitekehyksen kolme vastaavaa osaa ovat säännöt, järjestelmä sekä hauskuus. [5]

Mekaniikka

Mekaniikka tarkoittaa pelin sääntöjä, toimintoja ja pelaajan mahdollisuutta hallita dataa. Mekaniikan avulla muodostetaan pohja pelin dynamiikalle sekä estetiikalle. Muokkaamalla pelin mekaniikkaa harkitusti voidaan hioa lopullista pelikokemusta haluttuun suuntaan. Esimerkiksi Monopolyssa mekaniikkaan kuuluvat pelaajan liikkuminen pelilaudalla nopan avulla, hotellien ostaminen ja muu rahaliikenne. Pelikokemusta voidaan muokata mekaniikan avulla esimerkiksi antamalla heikossa asemassa oleville tukea ja perimällä johtoasemassa olevilta pelaajilta veroja. Näin peli pysyy kaikille osapuolille jännittävänä pidempään. Mekaniikan hienosäätö on iteratiivinen prosessi, jossa on tärkeää ottaa huomioon pelitestaus ja pelaajien kokemukset. Esimerkiksi kevyessä ajanvietteessä mekaniikat voivat olla yksinkertaisempia, kun taas strategisissa peleissä mekaniikan on tuettava monikerroksista ja tarkoin suunniteltua pelikokemusta. [5]

Dynamiikka

Dynamiikka kuvaa mekaniikkojen ja pelaajan toiminnan välistä vuorovaikutusta. Toimiva dynamiikka luo pelaajalle mielekkäitä pelikokemuksia. Esimerkiksi haastetta voidaan luoda aikarajoitteilla ja painostamalla pelaajaa nopeisiin päätöksiin. Yhteisöllisyyttä voidaan vahvistaa jakamalla tietoja pelaajien kesken tai asettamalla voittoehtoja, joita on hankala saavuttaa yksin. Dynamiikan ja sen merkityksen ymmärtäminen auttaa suunnittelijoita luomaan pelejä, jotka tarjoavat mielekkäitä kokemuksia, tai parantamaan huonosti toimivia dynamiikkoja. Esimerkki huonosti toimivasta dynamiikasta on Monopoly, jossa hyvin menestyvät pelaajat rikastuvat entisestään ja huonosti menestyvät pelaajat joutuvat usein lopettamaan pelin kesken. Kun pelaajien välille muodostuu liiallisia eroja, huonosti pärjäävät pelaajat menettävät halunsa pelata peliä. Analysoimalla dynamiikkaa voidaan tunnistaa tällaiset ongelmat ja sunnitella ratkaisuja, jotka parantavat pelikokemusta. [5]

Estetiikka

MDA-viitekehyksessä estetiikalla ei tarkoiteta pelien grafiikkaa tai äänimaailmaa. Peli voi olla visuaalisesti hyvinkin kaunis, mutta pelaajan kokema hauskuus määrittelee usein, haluaako hän käyttää aikaansa peliin. Lähdemateriaalissa korostetaan, että pelisuunnittelussa tulisi pyrkiä siirtymään epämääräisten termien, kuten ”hauskuus” ja ”pelattavuus”, käytöstä kohti tarkempaa sanastoa pelikokemusten kuvaamiseen. Taulukossa 2.1 esitellään lähdemateriaalin mukainen taksonomia pelien estetiikan kuvaamiseksi.

Taulukko 2.1: Erilaisten pelikokemusten käsitteet englanniksi [5] sekä suomeksi käännettynä.

Englanti	**Suomi**
Sensation	peli aistinautintona
Fantasy	peli mielikuvituksena
Narrative	peli draamana
Challenge	peli esteratana
Fellowship	peli sosiaalisena kehyksenä
Discovery	peli tutkimattomana alueena
Expression	peli itsensä ilmaisuna
Submission	peli ajanvietteenä

Tämän taksonomian avulla voidaan jakaa tietynlaisia pelejä omiin kategorioihinsa ja helpommin havaita mahdolliset puutteet eri estetiikan osa-alueilla. Jokainen peli sisältää useampia taulukossa listattuja ominaisuuksia, joita voidaan parantaa pelaajan kannalta houkuttelevammaksi. Esimerkiksi The Sims -pelisarjassa yhdistyvät löytämisen, fantasian, ilmaisun ja draaman elementit, kun taas Quake-peli keskittyy haasteeseen, kilpailuun ja fantasiaan. Vaikka ei ole olemassa kaavaa, joka määrittäisi ”hauskuuden” ainesosat, tämä luokittelu auttaa kuvaamaan pelejä ja selventämään, miksi eri pelit vetoavat eri pelaajiin tai samaan pelaajaan eri aikoina. Esteettisten mallien avulla voidaan määritellä pelattavuuden malleja, jotka kuvaavat pelin dynamiikkaa ja mekaniikkoja tarkemmin. Esimerkiksi sekä Alias että Quake ovat kilpailuhenkisiä pelejä, joiden onnistuminen riippuu pelaajien tunnesitoutumi-

sesta toistensa voittamiseen. Tämä edellyttää toisiaan vastaan kilpailevia pelaajia ja selkeää voittoehtoa. Kilpailullisten pelien suunnittelussa on tärkeää tukea vastakkainasettelua ja antaa selkeää palautetta siitä, kuka on voitolla. Esteettisten tavoitteiden ymmärtäminen on tärkeää pelin suunnittelussa ja hienosäädössä. Esimerkiksi jos Monopolyyn lisättäisiin monimutkaisia laskutoimituksia vaativia veroja, se voisi heikentää pelaajien sitoutumista peliin, sillä se vaikeuttaisi rahamäärien, edistyksen ja kilpailutilanteen seuraamista. On siis tärkeää muistaa, että MDA:n komponentit eivät ole irrallisia kokonaisuuksia, vaan ne vaikuttavat toisiinsa ja lopulta pelaajakokemukseen. [5]

MDA siis keskittyy pelin rakenteeseen ja siihen, miten pelilliset elementit vaikuttavat pelikokemukseen. MDA:n avulla voidaan analysoida peliä eri näkökulmista ja ymmärtää, miten pelilliset valinnat vaikuttavat pelikokemukseen. Se on hyödyllinen työkalu sekä pelisuunnittelijoille että pelikriitikoille. Seuraava viitekehys puolestaan keskittyy ihmisen motivaatioon ja siihen, miten pelillistämisen elementtejä voidaan käyttää motivaation lisäämiseksi ja sitoutumisen parantamiseksi. [5]

2.2 Octalysis-viitekehys

Oktalyysi (engl. *Octalysis*) on Yu-kai Choun kehittämä hieman MDA:ta modernimpi pelillistämisen viitekehys. Se perustuu laajaan 17 vuoden tutkimukseen pelaajien tarpeista ja auttaa ymmärtämään ja soveltamaan pelielementtejä arkielämän tilanteisiin. Pelillistettyjen järjestelmien suunnittelussa korostetaan ihmiskeskeistä lähestymistapaa perinteisen toimintokeskeisen mallin sijasta. Useat prosessit on suunniteltu halutun toiminnon saavuttamiseksi, eikä niinkään käyttäjän sitoutumista ja motivointia ajatellen. Kun järjestelmä suunnitellaan ihmiskeskeisen mallin mukaan, käyttäjän mielenkiintoa saadaan kasvatettua, ja haluttu toiminto voidaan saavuttaa suuremmalla todennäköisyydellä. Oktalyysin viitekehystä kuvataan kahdeksankulmiolla, jonka jokainen sivu edustaa yhtä ihmisen sisäistä motivaatiotekijää. Nämä

tekijät ovat esitetty taulukossa 2.2. [6]

Erilaiset motivaatiotekijät voidaan luokitella eri kategorioihin useilla eri tavoilla, jotta saataisiin kokonaisvaltainen käsitys miten ne vaikuttavat käyttäytymiseen. Eräs keskeinen luokittelutapa on valkohattu vastaan mustahattu, jossa kahdeksankulmion yläosassa olevat positiiviset tekijät (taulukossa ⇑-merkityt tekijät 1, 2 ja 3) esittävät tekniikoita, jotka vetoavat positiivisiin yksilön tarpeisiin. Tämä tarkoittaa, että pelaaja motivoituu toimimaan, koska se antaa hänelle mahdollisuuden ilmaista luovuuttaan, kehittää taitojaan ja kokea onnistumisen tunteita tai tuntea olevansa osa jotain itseään suurempaa. Kahdeksankulmion alaosassa olevat negatiiviset tekijät, (taulukossa ⇓-merkityt tekijät 6, 7 ja 8) jotka vetoavat negatiivisiin yksilön tarpeisiin. Tällaisia motivaattoreita ovat esimerkiksi ennakoimattomuus ja uteliaisuus, pelko jostain negatiivisen tapahtumisesta tai halu saada jotain, jota ei voi saada. Vaikka mustahattu-tekniikat voivat olla erittäin tehokkaita pelaajan motivoinnissa, ne voivat myös jättää pelaajalle negatiivisen tunteen, jos niitä käytetään väärin. On kuitenkin tärkeää huomata, että mustahattu-luokittelu ei tarkoita, että tekniikka olisi automaattisesti huono. Mustahattutekniikoita voidaan käyttää myös positiivisten ja terveellisten tulosten saavuttamiseen, esimerkiksi motivoimalla ihmisiä liikkumaan enemmän tai syömään terveellisemmin. [6]

Toinen keskeinen luokittelutapa jakaa ydintarpeet symbolisesti vasempaan ja oikeaan aivopuoliskoon. Vaikka aivojen puoliskot toimivat todellisuudessa yhdessä, tämä metaforinen jaottelu auttaa hahmottamaan eri motivaatiotekijöiden välisiä suhteita eri näkökulmasta. Viitekehyksessä oikean aivopuoliskon (Taulukossa ⇒-merkityt) motivaatiotekijät liittyvät luovuuteen, itseilmaisuun ja sosiaalisuuteen, kun taas vasemman puoliskon (Taulukossa ⇐-merkityt) motivaatiotekijät liittyvät logiikkaan, laskelmiin ja omistajuuteen. Mielenkiintoista on, että vasemman aivopuoliskon ydintarpeet ovat ulkoisia motivaattoreita, eli ne motivoivat käyttäjää, koska hän haluaa saavuttaa jotain, kuten tavoitteen, hyödykkeen tai jonkin muun

hankalasti saavutettan asian. Oikean aivopuoliskon ydintarpeet ovat puolestaan sisäisiä motivaattoreita, jolloin käyttäjä ei tarvitse tavoitetta tai palkintoa voidakseen käyttää luovuuttaan, viettää aikaa ystävien kanssa tai tuntea ennakoimattomuuden jännitystä, vaan kokee itse toiminnan palkitsevana. [6]

Taulukko 2.2: Oktalyysin motivaatiotekijät suomennettuna

Motivaatiotekijä	**Kuvaus**
1. Eeppinen merkitys ⇑	Käyttäjä uskoo tekevänsä jotain hyvin tärkeää ja itseään suurempaa.
2. Saavutus ⇐ ⇑	Sisäinen halu edistyä, kehittää taitoja ja lopulta voittaa haasteita.
3. Voimaannuttaminen ⇑ ⇒	Käyttäjät voivat ilmaista itseään luovasti ja nähdä työnsä tulokset.
4. Omistajuus ⇐	Käyttäjän motivaatio syntyy omistamisen tunteesta jotakin asiaa kohtaan.
5. Sosiaalinen vaikutusvalta ⇒	Sosiaaliset elementit, kuten mentorointi, hyväksyntä ja kilpailu motivoivat käyttäjiä.
6. Niukkuus ⇐ ⇓	Sisäinen halu saada jotain, mitä ei voi saada heti, tai on hankalasti ansaittavissa.
7. Arvaamattomuus ⇓ ⇒	Sisäinen halu saada selville, mitä seuraavaksi tapahtuu.
8. Välttäminen ⇓	Käyttäjän motivaatio syntyy negatiivisten asioiden, kuten rangaistusten välttämisestä.

Oktalyysi voidaan jakaa myös tasoihin, riippuen siitä kuinka syvällisesti valittu peli halutaan pelillistää. Oktalyysin käyttö alkaa aina tasosta yksi, jossa analysoidaan ja listataan, mitkä pelimekaniikat vetoavat kuhunkin ydintarpeeseen. Nämä pelimekaniikat liitetään kahdeksankulmion sivuille oikeisiin kohtiinsa, ja merkitään kunkin ydintarpeen voimakkuus. Tarpeiden voimakkuus riippuu siitä, kuinka tehokkaasti pelimekaniikat siihen vetoavat. Erittäin voimakkaat ydintarpeet merkitään kahdeksankulmiossa pitkillä sivuilla, kun taas heikot ydintarpeet merkitään lyhyillä sivuilla. Jos sivu supistuu niin paljon, että se ylittää sisäisen kahdeksankulmion, se on liian heikosti edustettu ja sitä on parannettava. Tasolla kaksi optimoidaan pelaajan kokemus matkan kaikissa neljässä vaiheessa: löytäminen, perehdytys, eteneminen ja loppupeli. Löytäminen tarkoittaa sitä, miksi pelaajien kiinnostus herää,

ja mitä odotuksia pelikokemuksesta herää. Markkinointi ja viestintä ovat avainasemassa tässä vaiheessa. Perehdytysvaiheessa pelaajalle opetetaan pelin säännöt ja työkalut, ja sen tulisi olla sujuvaa ja mukaansatempaavaa, jotta pelaajat eivät turhaudu ja lopeta pelaamista heti alkuunsa. Mielekkäät tutoriaalit ovat avainasemassa tässä vaiheessa. Etenemisvaihe on pelin ydin, jossa pelaajat usein toistavat tiettyjä toimia saavuttaakseen tavoitteensa. Pelaajille on tärkeää tarjota sopivia haasteita ja palkita heitä edistymisestä. Loppupeli kuvastaa sitä vaihetta, miten pidempiaikaiset pelaajat pidetään mukana uuden sisällön loppuessa. Tässä vaiheessa tulisi tarjota pelaajille uusia haasteita ja tavoitteita, jotka estävät kyllästymisen peliin. Oikein toteutettuna tason kaksi avulla luodaan pelikokemus, joka on motivoiva ja palkitseva alusta loppuun. Tasolla kolme otetaan huomioon erilaiset pelaajatyypit ja se, miten he motivoituvat pelikokemuksen eri vaiheissa. Jos kyseessä on esimerkiksi toimintapeli, siihen saatetaan lisätä tarinankerronnalisia elementtejä, jotta peli on houkuttelevampi laajemmalle yleisölle. [6]

Oktalyysi keskittyy siis ihmisen motivaatioon ja siihen, miten pelillistämisen elementtejä voidaan käyttää motivaation lisäämiseksi ja sitoutumisen parantamiseksi. Se perustuu kahdeksaan ydinviettiin, jotka ohjaavat ihmisen käyttäytymistä. Nämä viettivät on järjestetty kahdeksankulmion muotoon, josta viitekehyksen nimi on peräisin. Yhdistämällä MDA:n ja Oktalyysin voidaan luoda syvällisempi ymmärrys pelien ja pelillisten järjestelmien toiminnasta. MDA:n avulla voidaan analysoida pelien rakenteellisia elementtejä ja niiden vaikutusta pelikokemukseen, kun taas Oktalyysi tarjoaa työkalut ihmisen motivaation ymmärtämiseen ja hyödyntämiseen. Esimerkiksi pelisuunnittelija voi käyttää MDA:ta pelin mekaniikkojen suunnitteluun ja Oktalyysia varmistaakseen, että pelin dynamiikka aktivoi pelaajissa haluttuja ydinviettejä. Tämä voi johtaa pelikokemukseen, joka on sekä hauska että motivoiva. On kuitenkin tärkeää muistaa, että sekä MDA että Oktalyysi ovat työkaluja, jotka auttavat ymmärtämään ja analysoimaan pelejä. Ne eivät tarjoa valmiita vastauksia tai

kaavoja, vaan ne edellyttävät kriittistä ajattelua ja luovuutta. Seuraavassa luvussa tarkastellaan, miten näitä viitekehyksiä voidaan soveltaa yksittäistä peliä tarkastellessä.

3 Minecraft

Minecraft on avoimen maailman hiekkalaatikkopeli, joka on kaikkien aikojen myydyin videopeli. Minecraftissa pelaajat voivat tutkia satunnaisesti generoitua maailmaa, kerätä resursseja, luoda rakenteita ja taistella vihollisia vastaan. Pelin keskeinen piirre on sen avoimuus ja pelaajan vapaus tehdä mitä haluaa. Peliä käytetään yhä enemmän myös pelillistettynä opetusvälineenä joustavuutensa ansiosta [7]. Minecraftin menestyksen myötä on julkaistu useita muita versioita, jotka vaihtelevat ominaisuuksien sekä käytetyn ohjelmointikielen perusteella. Tekstissä Minecraftin perusversiolla tarkoitetaan Minecraft: Java Editionia, josta julkaistiin ensimmäinen versio jo vuonna 2009. Perusversion lisäksi seuraavat alaluvut käsittelevät koulutukseen suunnattua Minecraft Education-versiota ja muunneltua perusversiota (engl. *modded*). Alaluvuissa tarkastellaan myös miten Minecraft on pelillistetty, sekä eri versioiden välisiä eroavaisuuksia.

Minecraft on ohjelmoitu Javalla, joka on yksi maailman käytetyimmistä ohjelmointikielistä. Tämän vuoksi pelin toiminnallisuutta on helppo muokata omiin tarpeisiinsa sopivaksi asentamalla yhteisön luomia lisäosia tai ohjelmoimalla omia lisäosia (engl. *modding*), josta suomeksi käytetään usein termiä modaus. Vuodesta 2018 lähtien pelin kehittäjät ovat asteittain avanneet pelin lähdekoodin kirjastoja kaikille avoimiksi, helpottaen pelin toiminnallisuuden muokkausta entisestään [8]. Pelin muokattavuus ja modifikaatiot ovat vaikuttaneet myös pelin suosioon positiivisesti [9].

3.1 Perusversio

Edellisen luvun alussa tarkastelimme MDA -viitekehystä, joka analysoi pelejä kolmella eri tasolla: mekaniikka, dynamiikka ja estetiikka. Tarkastellaan, miten nämä tasot ilmenevät Minecraftissa.

Mekaniikkaa ovat pelin perussäännöt ja toiminnot, jotka määrittelevät pelin kulun. Minecraftissa mekaniikkoja ovat esimerkiksi resurssien kerääminen, esineiden valmistaminen, rakentaminen, taistelu, sekä maailman generointi satunnaisesti. Resurssien kerääminen on perusta selviytymiselle ja edistymiselle. Pelaajan on kaivettava, hakattava ja kerättävä materiaaleja ympäristöstä voidakseen valmistaa työkaluja, rakentaa suojaa ja edetä pelissä. Tämä mekaniikka on olennainen osa lähes kaikkea mitä pelaaja tekee Minecraftissa. Esineiden valmistaminen lisää pelin syvyyttä ja monimutkaisuutta. Yhdistämällä erilaisia materiaaleja pelaaja voi luoda uusia ja hyödyllisiä esineitä. Rakennusmekaniikan ansiosta pelaajalla on vapaus asettaa keräämiään kuutioita vapaasti ympäri pelimaailmaa. Vapaavalintainen taistelumekaniikka tarjoaa pelaajalle haastetta ja jännitystä. Viholliset ilmestyvät yöllä ja voivat olla vaarallisia. Taistelussa pelaaja voi käyttää erilaisia aseita ja strategioita.

Dynamiikka syntyy, kun pelaajat hyödyntävät pelin mekaniikkoja ja ovat vuorovaikutuksessa pelin kanssa. Minecraftissa dynamiikkaa ovat esimerkiksi selviytyminen, tutkiminen, yhteistyö ja ongelmanratkaisu. Minecraftissa selviytyminen on keskeisessä roolissa, ja tämä pakottaa pelaajat hyödyntämään edellisessä kappaleessa mainittuja mekaniikkoja luovasti ja strategisesti. Minecraftin valtavat ja satunnaisesti generoidut maailmat kannustavat pelaajia tutkimaan ja löytämään uusia asioita. Pelaaja voi löytää esimerkiksi rakennelmia, joiden tutkimisesta seuraa usein palkinto. Tämä tutkimisen dynamiikka ruokkii pelaajan uteliaisuutta ja tarjoaa jatkuvasti uusia haasteita ja palkintoja. Moninpeliominaisuudet avaavat oven yhteistyölle ja vuorovaikutukselle muiden pelaajien kanssa.

Estetiikkaa ovat pelaajan kokemat tuntemukset ja kokemukset pelistä. Minec-

raftissa estetiikaksi voidaan laskea esimerkiksi rauhallisuus ja rentoutuminen, saavutuksen tunne, löytämisen ilo ja yhteisöllisyyden tunne. Pelissä ei ole kiirehtimisen pakkoa, ja pelaaja voi keskittyä omiin projekteihinsa. Minecraftin rauhallinen luonne on osa sen käyttäytymistä, joka antaa pelaajalle mahdollisuuden rentoutua ja nauttia pelistä. Pelaajat voivat kokea saavutuksen tunnetta rakentamalla monimutkaisia rakenteita, keräämällä resursseja ja selviytymällä pelin haasteista. Pelaaja voi asettaa itselleen tavoitteita ja työskennellä niiden saavuttamiseksi, mikä tuo tyydytystä ja ylpeyden tunnetta. Minecraftin laaja ja avoin maailma tarjoaa pelaajille loputtomasti mahdollisuuksia tutkia uusia alueita ja löytää uusia asioita. Minecraft on myös suosittu peli, jota pelataan usein yhdessä ystävien tai muiden pelaajien kanssa. Yhdessä pelaaminen voi luoda vahvan yhteisöllisyyden tunteen, kun pelaajat auttavat toisiaan, rakentavat yhdessä ja jakavat kokemuksiaan.

On tärkeää huomata, että jokainen pelaaja kokee esteettiset ominaisuudet eri tavalla. Se, mikä toiselle pelaajalle on rentouttavaa, voi toiselle olla tylsää. Yhdelle pelaajalle haastava kokemus voi toiselle olla turhauttava. Minecraftin estetiikka onkin vahvasti sidoksissa pelaajan omiin mieltymyksiin ja pelityyliin. Seuraavaksi tarkastellaan, miten Minecraft on pelillistetty oktalyysin avulla. Keskitytään ensin oktalyysin kahdeksaan ydintekijään:

1. Eeppinen merkitys ja kutsumus: Vaikka Minecraftissa ei ole selkeää juonta tai ennalta määrättyä tehtävää, pelaajat voivat löytää pelille oman merkityksensä ja kokea olevansa osa jotain suurempaa. Esimerkiksi pelaaja voi omistautua laajojen ja yksityiskohtaisten maailmojen rakentamiseen, joita muut pelaajat voivat ihailla. Tämä puolestaan voi herättää tunteen siitä, että pelaaja jättää oman jälkensä pelimaailmaan ja luo jotain arvokasta.

2. Kehittyminen ja saavutus: Minecraft tarjoaa lukuisia mahdollisuuksia kehittyä ja kokea saavutuksen tunnetta. Pelaaja oppii uusia asioita pelimekaniikoista, rakennustekniikoista ja resurssien hankkimisesta. Näiden haasteiden voittaminen tuo

mukanaan tyydytystä ja ylpeyden tunnetta.

3. Luovuus ja voimaannuttaminen: Minecraft on avoin hiekkalaatikkopeli, joka antaa pelaajille rajattoman vapauden luovuudelle. Pelaajat voivat rakentaa mitä tahansa, kokeilla erilaisia materiaaleja ja tyylejä sekä luoda omia sääntöjään.

4. Omistajuus ja hallinta: Minecraft korostaa omistajuuden tunnetta antamalla pelaajille täyden kontrollin luomistaan maailmoista. Pelaajat voivat kerätä resursseja ja rakentaa tai muokata ympäristöään haluamallaan tavalla. Tämä omistajuuden tunne lisää pelaajan sitoutumista peliin ja motivoi jatkamaan pelaamista.

5. Sosiaalinen vaikutus ja yhteenkuuluvuus: Minecraft on suosittu myös moninpelinä, ja pelaaminen ystävien tai muiden pelaajien kanssa voi luoda vahvan yhteisöllisyyden tunteen. Pelaajat voivat tehdä yhteistyötä rakentaakseen uskomattomia projekteja, jakaa resurssejaan ja auttaa toisiaan selviytymään pelin haasteista. Yhdessä pelaaminen lisää hauskuutta ja motivaatiota, ja se voi luoda pysyviä ystävyyssuhteita.

6. Niukkuus ja kärsimättömyys: Minecraft hyödyntää niukkuuden ja kärsimättömyyden elementtejä rajoittamalla tiettyjen resurssien saatavuutta ja asettamalla aikarajoitteita joidenkin toimintojen suorittamiselle. Esimerkiksi harvinaisten mineraalien löytäminen voi vaatia pitkiä ja vaarallisia tutkimusretkiä, ja tiettyjen kasvien kasvaminen vie aikaa. Nämä rajoitukset tekevät pelistä jännittävämmän ja motivoivat pelaajia työskentelemään ahkerammin tavoitteidensa saavuttamiseksi.

7. Arvaamattomuus ja uteliaisuus: Minecraftin maailmat generoidaan satunnaisesti, ja pelaajat voivat aina odottaa löytävänsä jotain uutta ja jännittävää. Tutkimattomat luolat, piilotetut aarteet ja arvaamattomat tapahtumat pitävät pelaajat valppaina ja innostuneina jatkamaan pelaamista. Uteliaisuuden tunne ja halu löytää uusia asioita motivoivat pelaajia tutkimaan maailmaa ja ottamaan riskejä.

8. Menetys ja välttäminen: Minecraft sisältää myös menetyksen ja välttämisen elementtejä. Pelaajat voivat menettää keräämiään resursseja ja rakentamiaan raken-

teita, jos he esimerkiksi kuolevat tai heidän maailmansa joutuu hyökkäyksen kohteeksi. Tämä menetyksen pelko motivoi pelaajia olemaan varovaisia ja suojelemaan omaisuuttaan. Se lisää myös jännitystä ja panoksia peliin.

Minecraftin pelillisyys on siis monipuolinen ja se vetoaa pelaajiin useilla eri tasoilla. Se onnistuu yhdistämään sekä vasemman että oikean aivopuoliskon ydintekijöitä ja tarjoaa sekä sisäistä että ulkoista motivaatiota. Tämä selittää osaltaan pelin kestävää suosiota ja kykyä viihdyttää pelaajia tuntikausia. On kuitenkin tärkeää huomata, että oktalyysin soveltaminen on subjektiivista ja riippuu yksittäisen pelaajan kokemuksesta. Kaikki pelaajat eivät koe samoja ydintekijöitä yhtä vahvasti.

Tarkastellaan seuraavaksi, miten Minecraftin avulla voi oppia ohjelmointia. Vaikka perusversiossa ei voi kirjoittaa yleisesti käytetyillä ohjelmointikielillä, peli tarjoaa kuitenkin useita tapoja opetella komentoja ja loogista ajattelua. Esimerkiksi kaikki loogiset portit on mahdollista rakentaa pelin tarjoaman punakiven (*redstone*) avulla. Punakivi on materiaali, joka toimii sähkönjohtimen tavoin. Sitä voidaan käyttää monimutkaisten piirien rakentamiseen, jotka ohjaavat erilaisia mekanismeja pelissä. Esimerkiksi yksinkertainen JA-portti voidaan rakentaa käyttämällä kahta punakivipolkua, jotka johtavat yhteen punakivisoihtuun. Soihtu syttyy vain, mikäli molemmat polut ovat aktiivisia samanaikaisesti. Vaikka punakivi ei noudata täysin samoja sääntöjä kuin oikeat sähköjohtimet, erilaisia punakivipiirejä yhdistemällä on mahdollista rakentaa jopa täysin toimiva tietokone, jolla voi suorittaa aritmeettisia operaatioita, tai jopa pelata yksinkertaistettua Minecraftia pelin sisällä.

Peli pitää sisällään myös komentokuutioita, joiden avulla pelaaja voi suorittaa komentoja pelimaailmassa. Komentokuutioita voidaan käyttää monimutkaisten automaatiojärjestelmien ja pelimekaniikkojen luomiseen. Niiden avulla voi myös oppia ohjelmoinnin periaatteita, kuten ehtolauseita ja silmukoita. Komentokuutioita ei kuitenkaan voi saada pelin selviytymistilassa, joten niiden käyttö keskittyy vain luovan tilan maailmoihin.

3.2 Minecraft Education

Oppimisprosessin pelillistäminen parantaa oppilaiden mielenkiintoa ja opppimistuloksia [10]. Juuri siksi Minecraftista onkin julkaistu etenkin peruskouluopetusta varten Minecraft Education, joka sisältää joitakin myös aikuisille suunnattuja tehtäviä [2] [11]. Pelin ominaisuuksia on muokattu opetukseen soveltuvammaksi esimerkiksi lisäämällä peliin sisäänrakennettuja oppitunteja ja tehtäviä, joita opettajat voivat käyttää oppituntien suunnittelussa [12]. Opettajille on luotu myös työkaluja, joiden avulla he voivat seurata oppilaiden edistymistä ja antaa palautetta [11]. Opettajat voivat myös rajoittaa oppilaiden pääsyä tiettyihin pelin ominaisuuksiin tai estää tiettyjen esineiden käytön. Opetusversio on jo itsessään hieman rajoitetumpi, sillä siitä on karsittu joitain peruspelin ominaisuuksia. Minecraft on nykyään myös osa Microsoftia, ja siksi se sisältyy myös joidenkin koulujen Microsoft 365 ohjelmistopaketteihin ilmaiseksi [13].

Minecraft Education pitää sisällään myös suoraan ohjelmointia koskevia oppitunteja. Opetus on suunniteltu siten, että ensimmäisillä oppitunneilla ei tarvita mitään pohjatietoa, ja tehtävät muuttuvat yksi kerrallaan haastavemmiksi [14]. Helpommat oppitunnit käyttävät usein pseudokoodia, kun taas haastavemmissa tehtävissä käytetään oikeita ohjelmointikieliä, kuten Python ja JavaScript [7] [15]. Microsoft on kehittänyt myös MakeCode-editorin, jossa on mahdollista luoda muun muassa muutospaketteja eli modeja (engl. *mod*) Minecraftiin raahaamalla valmiita toimintokuutioita haluamaansa järjestykseen [1]. Kuutiot vastaavat JavaScript-koodia, jota on myös mahdollista muokata suoraan palikoiden siirtelyn sijaan [15]. Minecraft Educationin muunteleminen on siis helppoa, eikä vaadi aiempaa osaamista ohjelmoinnista, tai kolmannen osapuolen työkaluja. MakeCodella tehdyt modit eivät ole kuitenkaan yhteensopivia perusversion kanssa, sillä Education-versio on ohjelmoitu C++ kielellä.

3.3 Muunneltu Minecraft

Mikäli peruspeliä halutaan muunnella, tarvitaan valmiin modin Java-arkisto eli JAR, sekä tapa injektoida se osaksi pelin koodia [9]. Valmiita modeja voi ladata ilmaiseksi useista internet-arkistoista (engl. *repository*) kuten CurseForge tai Modrinth. Modien injektointia varten on luotu yhteisön kehittämiä modausalustoja (engl. *modloader*) kuten Forge ja Fabric, joilla on omat vahvuutensa [9]. Tässä työssä ei tarkastella miten valmiiden modien injektointi peliin käytännössä toimii, sillä modausalustat hoitavat sen automaattisesti, eikä käyttäjän toimia tarvita. Modi sijoitetaan pelihakemiston mods-kansioon, jonka jälkeen käynnistetään Minecraft valitun modausalustan avulla, jolloin modi on injektoitu osaksi pelin koodia.

Omien modien luominen vaatii ohjelmointitaitoja, yleensä Javalla [9]. Ohjelmoijan tulee myös ymmärtää pelin mekaniikan, esineiden, lohkojen ja entiteettien toimintaperiaatteet, jotta ohjelmoitu modi ei toimi väärällä tavalla. Aiemmin mainitut modausalustat tarjoavat laajat ohjelmointirajapinnat sekä dokumentaatiot, joiden avulla on mahdolista päästä helpommin muokkaamaan pelimekaniikkoja tai lisäämään uusia ominaisuuksia. Dokumentaatioista löytyy myös modipohjia ja tutoriaaleja ensimmäisen modin luomiselle, jonka ansiosta prosessia ei tarvitse aloittaa tyhjästä. Useiden modien samanaikainen käyttö voi aiheuttaa ongelmia, jonka vuoksi modausalustojen sisäänrakennetut yhteensopivuusratkaisut vähentävät modien välisiä ristiriitoja, kuten ID-konflikteja. Mikäli ohjelmoija noudattaa valitsemansa modausalustan ohjelmointirajapintojen dokumentaatiota, alusta hoitaa mahdolliset konfliktit.

Pelin koodin muokkaaminen mahdollistaa sen, että mitä tahansa voidaan lisätä, poistaa tai muokata. Pienet modit voivat lisätä esimerkiksi uuden esineen, kun taas suuret modipaketit voivat muuttaa tapaa, miten pelimaailma luodaan, tai lisätä kokonaan uusia pelimekaniikkoja [9]. MDA-viitekehyksen mukaisesti pelimekaniikkoja muuttamalla voidaan vaikuttaa siihen, miten Minecraft on pelillistetty. Jos

pelaajat esimerkiksi kaipaavat enemmän seikkailua, he voivat asentaa haluamansa modin. Vastaavasti teknologiasta kiinnostuneet pelaajat voivat ladata teknologiaan suunnatun modipaketin, joka tarjoaa pelaajille mieluisia esteettisiä kokemuksia.

Ohjelmoinnin oppimisen kannalta esimerkiksi ComputerCraft on erityisen hyödyllinen modi [16]. ComputerCraft käyttää Lua-ohjelmointikieltä, joka on laajasti käytössä oleva kieli etenkin pelien muokkauksessa [17]. Vaikka pelkkä ohjelmoinnin mahdollisuus pelin sisällä ei tee siitä pelillistettyä, ComputerCraft tarjoaa useita ominaisuuksia, jotka tekevät ohjelmoinnin oppimisesta hauskempaa ja motivoivampaa Minecraft-ympäristössä. Merkittävin ominaisuus on se, että ohjelmointi on suoraan sidoksissa Minecraft-maailmaan. Modi sisältää ohjelmoitavia tietokoneita, modeemeja verkkojen muodostamiseen sekä kilpikonnia, jotka on ohjelmoitavissa liikkumaan, kaivamaan, rakentamaan tai taistelemaan [17]. Pelaajat voivat automatisoida tehtäviä, rakentaa monimutkaisia rakennelmia ja jopa luoda omia pelimekaniikkoja [16]. Tämä tekee ohjelmoinnista merkityksellisempää ja motivoivampaa kuin perinteisessä ohjelmointiympäristössä, jossa esimerkit voivat usein tuntua irrallisilta ja keinotekoisilta. Esimerkiksi pelaaja voi asettaa tavoitteeksi rakentaa automaattisen kaivosjärjestelmän tai luoda robotin, joka suojelee häntä vihollisilta. Tämä motivoi oppimaan ohjelmointia, koska pelaaja näkee konkreettisia tuloksia työstänsä. Myös virheet on helppo huomata tarkastellessa ohjelman toimintaa käytännössä. Pelaajat voivat kokeilla erilaisia ohjelmointiratkaisuja ja nähdä niiden vaikutukset reaaliajassa, tehden oppimisesta hauskaa ja innostavaa ilman epäonnistumisen pelkoa. Modista on myös luotu matalamman lähtötason versio ComputerCraftEdu, jossa pelaajat voivat aloittaa graafisilla koodilaatikoilla tai yksinkertaisilla komennoilla ja vähitellen edetä monimutkaisempaan ohjelmointiin. Tämä tekee aloittamisesta entistä helpompaa ja vähentää turhautumista. Näin pelaaja saadaan oppimaan ohjelmointia pelillistetyn ohjelmointiympäristön avulla.

4 Pohdinta ja yhteenveto

TK1: Mitä pelillistäminen on?

Pelillistäminen on tekniikka, jossa peleistä lähtöisin olevia elementtejä hyödyntämällä lisätään motivaatiota ja sitoutumista. Sitä voidaan hyödyntää alasta riippumatta, vaikka se on ollut erityisen suosittua opetuksessa. Pelillistämisessä hyödynnetään usein pelielementtejä, kuten pisteytystä, tasoja, haasteita ja palkintoja. Nämä elementit voivat motivoida ihmisiä toimimaan ja oppimaan uusia asioita tehokkaammin. On myös olemassa elementtejä, jotka eivät ole alun perin lähtöisin peleistä, mutta jotka silti tukevat pelillistämistä.

Pelillistämistä voidaan konkretisoida tarkastelemalla sitä erilaisten viitekehysten avulla. Mekaniikkaa, dynamiikkaa sekä estetiikkaa kuvaava MDA-viitekehys tutkii pelin eri kerrosten välisiä vuorovaikutuksia. Toinen viitekehys tarkastelee pelillistämistä ihmisille luonnollisten kahdeksan ydintarpeen kautta. Samat ydintarpeet eivät vetoa jokaiseen ihmiseen, jonka vuoksi järjestelmää pelillistäessä onkin tärkeää ottaa huomioon kohderyhmään vetoavat tarpeet. On myös tärkeää muistaa, että pelillistäminen ei automaattisesti takaa onnistumista. Jotta pelillistäminen olisi tehokasta, se on suunniteltava huolellisesti kohderyhmä sekä tavoitteet huomioiden. Lisäksi on tärkeää varmistaa, että pelillistämisen elementit integroituvat saumattomasti itse oppimissisältöön.

TK2: Miten Minecraft voidaan pelillistää?

Minecraft pitää sisällään useita pelillistämisen elementtejä. Pelissä on paljon

MDA:n mukaisia siteitä mekaniikan, dynamiikan ja estetiikan välillä, jotka kannustavat pelaajaa jatkuvaan kehitykseen. Minecraftissa on siis hyödynnetty MDA:lle ominaisia piirteitä pelaajalle mielenkiintoisten kokemuksien aikaansaamiseksi. Pelistä voidaan havaita myös runsaasti ydintarpeisiin vetoavia toimintamalleja, jotka ovat oktalyysista tuttuja. Oktalyysin motivaatiotekijöitä tarkastellessa huomataan, että Minecraft vetoaa melko tasaisesti jokaiseen kahdeksaan tekijään. Tärkeämpänä tekijänä Minecraftin suosion kannalta saattaakin olla se, että pelaajalla on vapaus hyödyntää itseensä parhaiten vetoavia tekijöitä. Monet pelit rajoittavat pelaajan vapautta, mutta Minecraftin vapaan ja muokattavan tyylin ansiosta se vetoaa useampiin pelaajiin.

Minecraft Education vie pelillistämisen astetta pidemmälle, hyödyntäen peruspelin elementtejä opetustarkoituksiin. Opetusversiossa on sisäänrakennettuja oppitunteja oikeilla ohjelmointikielillä, ja kyseistä versiota on mahdollista muokata pelin sisäisen MakeCode-editorin avulla luoduilla modeilla. Opetusversio kuitenkin rajoittaa pelikokemusta, sillä kyseessä on eri ohjelmointikielellä toteutettu ja joiltakin ominaisuuksiltaan karsittu versio. Lisää haastetta tai vapautta halutessaan pelaaja voi ryhtyä modaamaan pelin perusversiota joko yhteisön luomilla modeilla, tai jopa ohjelmoida omia modeja esimerkiksi Javalla. ComputerCraft on esimerkki modista, joka sisällyttää ohjelmointiympäristön saumattomasti osaksi pelin kulkua. Pelaaja voi oma-aloitteisesti helpottaa edistymistään pelissä esimerkiksi automatisoimalla yksinkertaisia tehtäviä itse ohjelmoidun kilpikonnan avulla.

Yhteenvetona voidaan todeta, että pelillistäminen on tehokas työkalu motivaation ja sitoutumisen lisäämiseksi. Minecraft on erinomainen esimerkki pelillistetystä pelistä, joka tarjoaa pelaajille runsaasti mahdollisuuksia oppia ja kehittyä. Minecraft Education ja modit, kuten ComputerCraft, laajentavat pelin käyttömahdollisuuksia entisestään.

Lähdeluettelo

[1] J. G. López Solórzano, C. J. Ángel Rueda ja O. O. Vergara Villegas, "Measuring Undergraduates' Motivation Levels When Learning to Program in Virtual Worlds", en, *Computers*, vol. 13, nro 8, s. 188, elokuu 2024, Number: 8 Publisher: Multidisciplinary Digital Publishing Institute, ISSN: 2073-431X. DOI: 10.3390/computers13080188. url: https://www.mdpi.com/2073-431X/13/8/188 (viitattu 05. 10. 2024).

[2] A. Bile, "Development of intellectual and scientific abilities through game-programming in Minecraft", en, *Education and Information Technologies*, vol. 27, nro 5, s. 7241–7256, kesäkuu 2022, ISSN: 1573-7608. DOI: 10.1007/s10639-022-10894-z. url: https://doi.org/10.1007/s10639-022-10894-z (viitattu 05. 10. 2024).

[3] K. M. Kapp, *The Gamification of Learning and Instruction: Game-based Methods and Strategies for Training and Education*, 1st. Pfeiffer & Company, huhtikuu 2012, ISBN: 978-1-118-09634-5.

[4] B. P. Sanders, "Could Minecraft Be a School?", en, teoksessa *Game-based Learning Across the Disciplines*, C. Aprea ja D. Ifenthaler, toim., Cham: Springer International Publishing, 2021, s. 383–393, ISBN: 978-3-030-75142-5. DOI: 10.1007/978-3-030-75142-5_17. url: https://doi.org/10.1007/978-3-030-75142-5_17 (viitattu 05. 10. 2024).

[5] R. Hunicke, M. LeBlanc ja R. Zubek, ”MDA: A Formal Approach to Game Design and Game Research”, en,

[6] *The Octalysis Framework for Gamification & Behavioral Design*, en-US, Section: Gamification Examples, syyskuu 2023. url: https://yukaichou.com/gamification-examples/octalysis-complete-gamification-framework/ (viitattu 05. 10. 2024).

[7] N. Klimová, J. Šajben ja G. Lovászová, ”Online Game-Based Learning through Minecraft: Education Edition Programming Contest”, teoksessa *2021 IEEE Global Engineering Education Conference (EDUCON)*, ISSN: 2165-9567, huhtikuu 2021, s. 1660–1668. DOI: 10.1109/EDUCON46332.2021.9453953. url: https://ieeexplore.ieee.org/document/9453953 (viitattu 05. 10. 2024).

[8] *Programmers: Play with Minecraft's Inner Workings!*, en-US, lokakuu 2018. url: https://www.minecraft.net/en-us/article/programmers-play-minecrafts-inner-workings (viitattu 05. 10. 2024).

[9] J. Koene, *Sams Teach Yourself Mod Development for Minecraft® in 24 Hours*, en. marraskuu 2015, ISBN: 978-0672337635.

[10] E. J. Slattery, D. Butler, K. Marshall et al., ”Effectiveness of a minecraft education intervention for improving spatial thinking in primary school children: A mixed methods two-level cluster randomised trial”, *Learning and Instruction*, vol. 94, s. 102 003, joulukuu 2024, ISSN: 0959-4752. DOI: 10.1016/j.learninstruc.2024.102003. url: https://www.sciencedirect.com/science/article/pii/S0959475224001300 (viitattu 05. 10. 2024).

[11] T. Zambon, E. Amadori, P. Bernardelli ja C. Prandi, ”GameOn! Residency: Promoting Peace, Inclusivity, and Sustainability Through Serious Games”, teoksessa *Proceedings of the 2024 International Conference on Information Technology for Social Good*, sarja GoodIT ’24, New York, NY, USA: Associa-

tion for Computing Machinery, syyskuu 2024, s. 462–466, ISBN: 9798400710940. DOI: 10.1145/3677525.3678697. url: https://doi.org/10.1145/3677525.3678697 (viitattu 05. 10. 2024).

[12] S. Bourdeau, T. Coulon ja M.-C. Petit, "Simulation-Based Training via a "Readymade" Virtual World Platform: Teaching and Learning With Minecraft Education", *IT Professional*, vol. 23, nro 2, s. 33–39, maaliskuu 2021, Conference Name: IT Professional, ISSN: 1941-045X. DOI: 10.1109/MITP.2021.3062935. url: https://ieeexplore.ieee.org/document/9390428 (viitattu 05. 10. 2024).

[13] *Lessons*, en-US. url: https://education.minecraft.net/en-us/lessons/hour-of-code-generation-ai.FI (viitattu 05. 10. 2024).

[14] P. Voštinár ja R. Dobrota, "Minecraft as a Tool for Teaching Online Programming", teoksessa *2022 45th Jubilee International Convention on Information, Communication and Electronic Technology (MIPRO)*, ISSN: 2623-8764, toukokuu 2022, s. 648–653. DOI: 10.23919/MIPRO55190.2022.9803384. url: https://ieeexplore.ieee.org/document/9803384 (viitattu 05. 10. 2024).

[15] E. Kutay ja D. Oner, "Coding with Minecraft: The Development of Middle School Students' Computational Thinking", *ACM Trans. Comput. Educ.*, vol. 22, nro 2, helmikuu 2022. DOI: 10.1145/3471573. url: https://dl.acm.org/doi/10.1145/3471573.

[16] D. Saito, H. Washizaki ja Y. Fukazawa, "Influence of the Programming Environment on Programming Education", en, teoksessa *Proceedings of the 2016 ACM Conference on Innovation and Technology in Computer Science Education*, Arequipa Peru: ACM, heinäkuu 2016, s. 354–354, ISBN: 978-1-4503-4231-5. DOI: 10.1145/2899415.2925477. url: https://dl.acm.org/doi/10.1145/2899415.2925477 (viitattu 05. 10. 2024).

[17] *CC: Tweaked*. url: `https://tweaked.cc/` (viitattu 05. 10. 2024).