OULUN SEUDUN
AMMATTIKORKEAKOULU

Sami-Jukka Piippo

Mobiilipelin versiointi ja manuaalinen regressiotestaus

Case Hill Climb Racing

Mobiilipelin versiointi ja manuaalinen regressiotestaus

Case Hill Climb Racing

Sami-Jukka Piippo
Opinnäytetyö
Syksy 2013
Tietojenkäsittelyn Koulutusohjelma
Oulun seudun ammattikorkeakoulu

TIIVISTELMÄ

Oulun seudun ammattikorkeakoulu
Tietojenkäsittelyn koulutusohjelma

Tekijä: Sami-Jukka Piippo
Opinnäytetyön nimi: Mobiilipelin versiointi ja manuaalinen regressiotestaus Case Hill Climb Racing
Työn ohjaaja: Matti Viitala
Työn valmistumislukukausi ja -vuosi: kevät 2014 Sivumäärä: 27

Tässä opinnäytetyössä on tutkittu mobiilipelin versiointia ja manuaalista regressiotestausta. Käytännönosuutena oli pelitestausdokumentin luominen, sillä toimeksiantajalla oli tarve kyseiselle dokumentille. Toimeksiantajana työssä on ollut Fingersoft Oy, joka on vuonna 2012 perustettu oululainen pelialan yritys.

Pelin versioinnilla tarkoitetaan uuden peliversion luomista. Versioinnin toimenpiteitä ovat uusien ominaisuuksien suunnitteleminen, toteuttaminen, version merkitseminen, versioiden välisten erojen tunnistaminen ja version tallentaminen. Asiakkaille versiointi näkyy uusina ohjelmistopäivityksinä. Manuaalisella regressiotestauksella tarkoitetaan ihmisen tekemää testausta, joka tehdään aina ennen uuden peliversion julkaisua. Testaaja tutkii, onko työn alla olevassa ohjelmistossa virheitä ja kirjaa virheraportit tietokantaan, josta ohjelmoijat näkevät senhetkiset virhetoiminnot.

Ohjelmisto- ja pelitestausta käsittelevää kirjallisuutta oli tarjolla paljon. Tietoperustaa löytyi hyvin sekä kirjoista että Internetistä.

Opinnäytetyössä esitellään Hill Climb Racing -pelin taulukkomuotoinen testaussuunnitelma. Testaussuunnitelmasta löytyvät kaikki testattavat laitteet ja testitapaukset. Testaajan tehdessä tietyllä laitteella testin, jos testi onnistuu, hän tekee taulukkoon merkinnän laitteen kohdalle. Hill Climb-pelin testaamisessa on käytössä myös toinen dokumentti, jossa on listattu testit, niiden ominaisuudet ja se miten ominaisuuksien tulisi käyttäytyä.

Pelin on oltava mahdollisimman bugivapaa, jotta sitä olisi mielekästä pelata. Samalla on huomioitava pelin muu käytettävyys: jos siinä esiintyy puutteita, ei peli välttämättä vedä pelaajia puoleensa. Muiden käytettävyyden osa-alueiden testaaminen voisikin olla toisen opinnäytetyön aihe.

Fingersoftilla jatketaan opinnäytetyön yhteydessä tehdyn pelitestausdokumentin kehitystä. Tämä tulee olemaan yksi tulevaisuuden työtehtävistäni.

Asiasanat: testaus, mobiilipelit, testitapaukset, testausmenetelmät

ABSTRACT

Oulu University of Applied Sciences
Degree programme in Information Technology

Author: Sami-Jukka Piippo
Title of thesis: Mobile game versioning and manual regression testing Case Hill Climb Racing
Supervisor: Matti Viitala
Term and year when the thesis was submitted: Spring 2014
Number of pages: 27

The thesis studies mobile game versioning and manual regression testing. The goal of this thesis was to create a game testing documents, which the employer needed. The employer of the thesis is Fingersoft Ltd. a game company from Oulu which was established in 2012.

Game versioning refers to the creation of a new version of the game. The phases of versioning are: designing and creating new features, creating a version number, recognizing the differences between the versions and saving the new version. To the user versioning means new updates for the application. Manual regression testing refers to testing that a person spends time on, and is done before each new version of the game. The tester investigates the work in progress to discover any bugs and records them in the bug database. Developers can find the bug reports from the database.

There is a lot of literature available about software and game testing. The basis of knowledge was easy to find from literature and Internet.

The thesis presents a tabular test plan for Hill Climb Racing. From the test plan you can find every device and test case used. When the tester has done a test on a certain device and it succeeds he will mark it on the test plan is succeeded. Hill Climb Racing uses another document, which explains the tests and their features in detail.

The game should be as bug free as possible so that it is entertaining. At the same time you have to consider the game's usability: if there are any sorts of problems, the game might not interest a lot of gamers. Testing the other parts of usability could be another thesis all together.

Fingersoft will continue to develop the game testing document created during the thesis. This will be one of my future work assignments.

Keywords: testing, mobile games, test cases, testing methods

SISÄLLYS

1 JOHDANTO

Opinnäytetyön aiheena on mobiilipelin versiointi ja manuaalinen regressiotestaus. Valitsin aiheen, koska olen työskennellyt aiheen parissa. Videopelit ovat myös vapaa-ajanharrastukseni, ja olen toiminut vapaaehtoisena pelitestaajana usein. Teen opinnäytetyöni Fingersoft Oy:lle, joka on vuonna 2012 perustettu oululainen pelialan yritys.

Fingersoftin 2012 lokakuussa julkaisema peli Hill Climb Racing ylitti 100 miljoonan latauksen rajan syksyllä syksyllä 2013. Tätä kirjoitettaessa peliä lataa noin 300 000 uutta ihmistä joka päivä. Päivittäin pelillä on 4-6 miljoonaa pelaajaa.

Opinnäytetyöni on tärkeä Fingersoftille, koska testaus on olennainen osa pelien kehitysprosessia. Pelitestaus varmistaa, että peliin uusiin versioihin kehitetyt uudet ominaisuudet toimivat kuten on suunniteltu, eivätkä ne aiheuta ongelmia toiminnallisuuden kannalta. Siksi pelitestaus onkin yksi pelikehityksen tärkeimpiä vaiheita.

Testaus on osa pelin laadunvarmistusta ja se on elintärkeä vaihe pelin kuin pelin luomisessa. Testauksella pyritään löytämään ohjelmoinnin yhteydessä huomaamatta jääneet virheet (bugit). Virheet kirjataan ylös ja raportoidaan ohjelmoijille. Testaajan rooli on myös olla ensimmäinen pelin käyttäjä. Testaaja kommunikoi muulle tiimille siitä, mikä toimii ja mikä vaatii vielä hiomista.

Laadunvarmistuksella huolehditaan myös siitä, että lopputuotteeseen ei jää pelikokemusta heikentäviä ominaisuuksia. Esimerkiksi uuden kentän liian nopeasti kasvava vaikeusaste tai uuden ajoneuvon tarkoitukseton huijausmahdollisuus ovat esimerkkejä Hill Climb Racingista testaamisen avulla löydetyistä ominaisuuksista, joita ei ole tarkoitettu peliin ja jotka on korjattu pelin seuraavaan versioon.

Pelitestaaja työskentelee yhdessä ohjelmoijien ja artistien kanssa ennen kuin peli julkaistaan. Kun peli julkaistaan, jatkuu testaajan työ yhteistyössä asiakastuen kanssa. Joskus pelitestaajat voivat jopa tarkistaa ja kääntää pelin ohjeita ja oppaita kansainvälisille markkinoille. Yleensä ohjeiden laatiminen ei kuitenkaan kuulu testaajan työhön ja käännöstyöt on ulkoistettu käännöstoimistoille.

Pelitestaus tarkoittaa käytännössä sitä, että henkilö pelaa peliä, kuten pelaisi mitä tahansa peliä. Löytäessään virheen (bugin) pelistä, ilmoittaa testaaja siitä ohjelmoijille, jotka korjaavat sen. Julkaisun jälkeen testaaja voi joutua testaamaan erinäisiä asiakkaitten ilmoituksiin perustuvia asioita. Jos jokin ei asiakkaan mielestä toimi oikein, testaa testaaja onko kyseisessä tilanteessa mahdollisesti korjattavaa.

Opinnäytetyöni tavoitteena on luoda Fingersoftille toimiva pelitestausdokumentti, jonka avulla voidaan varmistaa, että uudessa ohjelmistoversiossa kaikki edellisen version ominaisuudet toimivat. Pelitestausdokumentin tarkoituksena on helpottaa ja suoraviivaistaa testausvaihetta ja nopeuttaa asiakkaiden tukipyyntöihin liittyviä toimenpiteitä.

Opinnäytetyötä on rajattu regressiotestaukseen ja versiointiin. Tutkimalla automatisoitua ja manuaalista regressiotestausta selvitetään se, olisiko automatisoitu testaus joissakin tilanteissa mahdollista. Opinnäytetyöhön kuuluu myös pelitestausdokumentin luominen. Testausdokumentti mahdollistaa nopean ja suoraviivaisen testaamisen. Dokumentissa tulee olemaan tieto siitä, mitä testataan ja miten kyseisen ominaisuuden oletetaan toimivan. Ellei toiminto läpäise testiä, siihen on kiinnitettävä vielä huomiota.

2 YLEISTÄ PELITESTAUKSESTA

Pelitestaus on työnä hyvin epäkiitollista, koska testauksen onnistuessa kukaan ei näe hyvin tehtyä työtä, mutta puutteellinen tai huolimattomasti tehty testaustyö näkyy käyttäjälle pelin huonona toimivuutena ja bugeina (Kasurinen 2013, 16). Usein käyttäjät valittavat keskeneräisenä markkinoille tuoduista peleistä ja sovelluksista. Käyttäjien kritiikki on monesti perusteltua. Yksi viimeisimpiä suuren yleisön mieliin jääneitä suuren luokan ongelmia uuden sovelluksen käyttöön otossa lienee VR:n lippujärjestelmän uusiminen vuonna 2011.

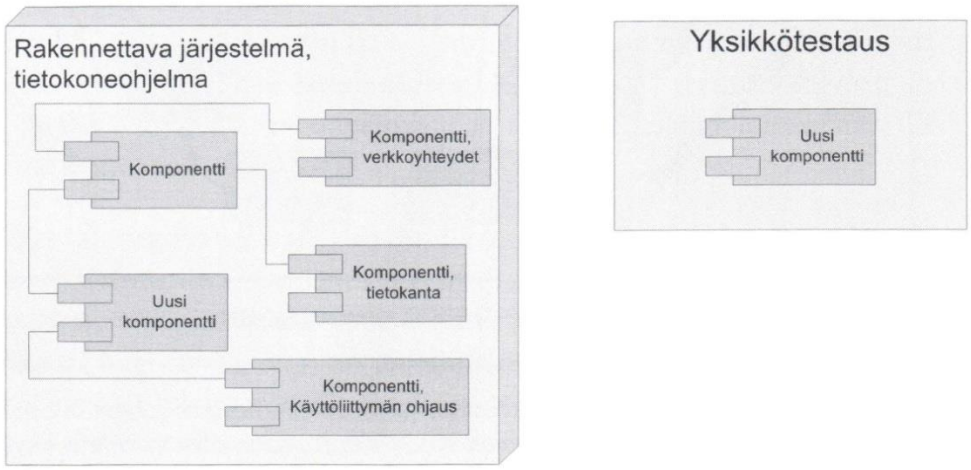
Uusien ohjelmien ja sovellusten käyttöönottovaiheessa ilmenevien ongelmien takia testaus saa huonosti tehdyn maineen. Käytännön ohjelmistoprojekteissa testaus on monesti projektin loppuvaiheessa tehtävä työ, jolle tulee kiire ja josta rahat loppuvat kesken. (Kasurinen 2013, 16.)

Ohjelmistotestaus tarkoittaa työtä, jota tehdään sen varmistamiseksi, että toteutettavasta ohjelmistotuotteesta tulee toivotun kaltainen, ja että kaikki siihen valmiiksi saadut ominaisuudet toimivat niin kuin oli tarkoitus. Käytännössä testausta voidaan pitää myös jatkuvana vertailutehtävänä: Testauksen avulla tarkastetaan, että se, mitä on saatu aikaiseksi vastaa sitä, mitä on ollut tarkoituksena tehdä, sekä tunnistaa ne kohdat, joissa tuotos poikkeaa suunnitelmista. (Kasurinen 2013, 10.)

Tyypillisiä testauksen tasoja ovat:

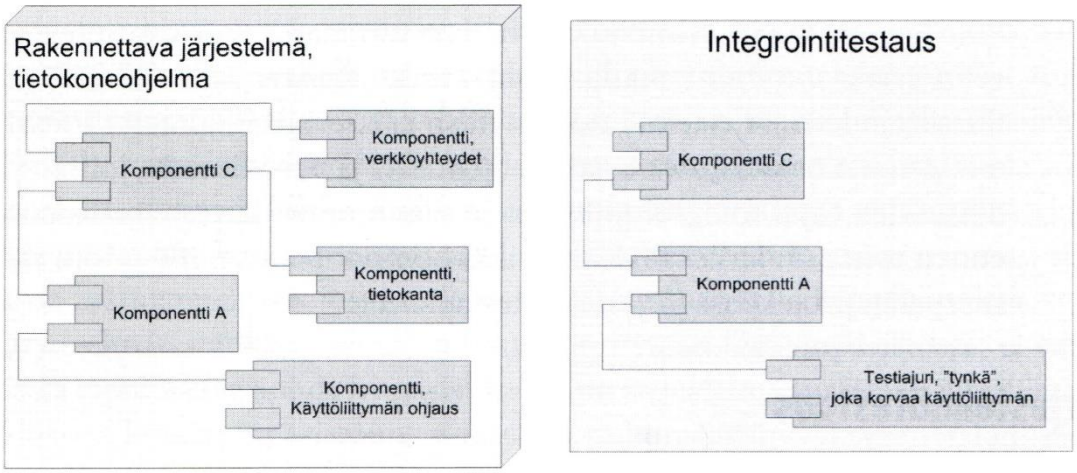
- Yksikkötestaus (Unit testing), jolla testataan yksittäisten sovelluskomponenttien kuten luokkien, metodien, funktioiden yms. toimintaa.
- Integraatiotestaus (Integration testing), jossa joukko erillisiä pienempiä komponentteja yhdistetään ja niiden yhteistoimintaa testataan. Integraatiotestaus tehdään yleensä yksikkötestauksen jälkeen.
- Järjestelmätestaus (system testing), joka testaa kokonaista ohjelmaa.
- Hyväksymistestaus (acceptance testing), jossa asiakas tarkistaa, täyttääkö sovellus asetetut vaatimukset. (Nuutila E. 2014, viitattu 12.4.2014.)

Kasurinen selventää yksikkötestausta seuraavasti: Yksikkötestaus tarkoittaa testaustyötä, jossa yhden yksittäisen moduulin, funktion tai olion toimintaa tarkastellaan välittömästi toteutuksen yhteydessä, useimmiten itse ohjelmoijan tai kehittäjän toimesta. Uutta komponenttia testataan muista osista erillään. Yksikkötestauksen tarkoituksena on varmistaa, että toteutettu toiminto tai muutos olemassa olevaan järjestelmään toimii ainakin periaatteessa. (Kasurinen 2013, 51.)



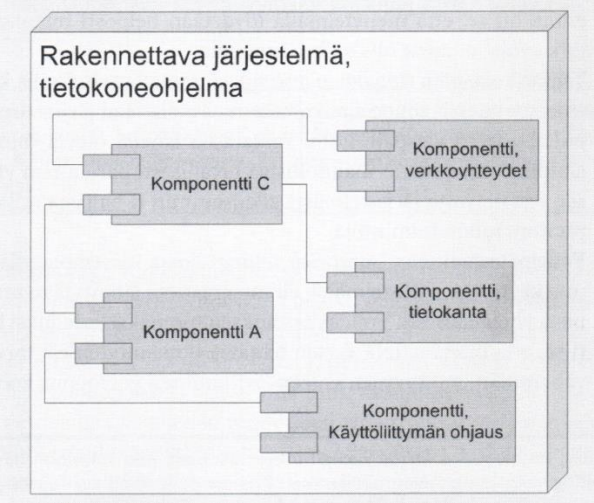
KUVIO 1. Yksikkötestaus (Kasurinen 2013, 52)

Integrointitestaus on yksikkötestauksen jälkeen suoritettava työvaihe. Siinä järjestelmän eri osia sovitetaan yhteen ja tarkoituksena on saada järjestelmä toimimaan yhtenä kokonaisuutena. Integrointitestauksen tärkein tavoite on todistaa, että järjestelmän osat toimivat myös yhdessä. (Kasurinen 2013, 54.)



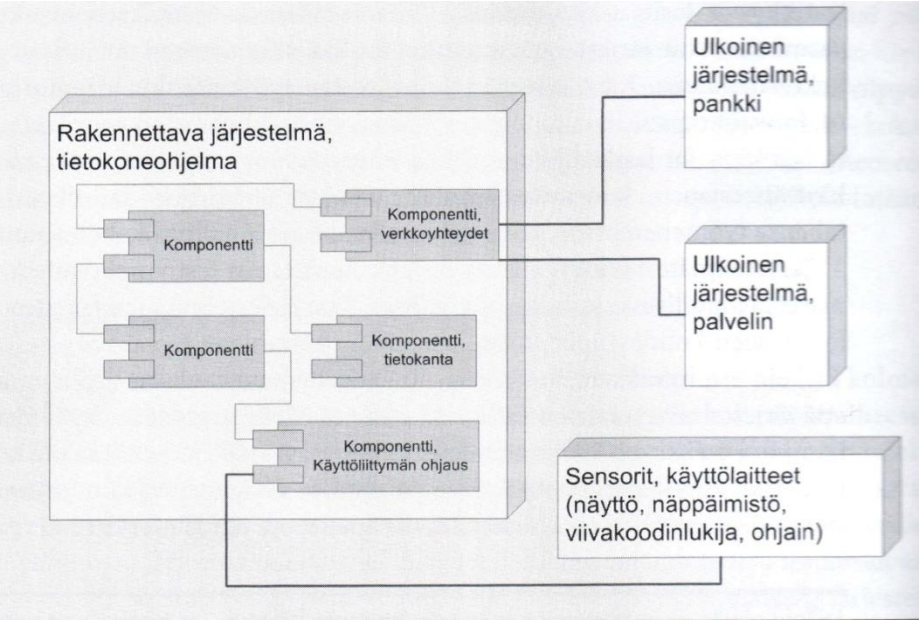
KUVIO 2. Integrointitestaus (Kasurinen 2013, 55)

Järjestelmätestaus ei viittaa mihinkään testaustapaan, vaan on yleisnimike kaikelle testaukselle, joka tehdään järjestelmälle. Järjestelmätestaus tekee sen, mihin nimi jo vihjaakin. Se on testaustyötä, jossa rakennettavaa järjestelmää testataan toiminnallisena kokonaisuutena. Kun yksikkötestaus ja integrointitestaus on tehty ja testit ovat onnistuneet, suoritetaan järjestelmätestaus. (Kasurinen 2013, 56.)



KUVIO 3. Järjestelmätestaus (Kasurinen 2013, 56)

Hyväksymistestaus on työvaihe, jolla on tarkoitus todistaa, että ohjelma on vaatimusmäärittelyn mukainen ja se pystyy luvattuihin toimintoihin. Tavallisesti hyväksymistestauksessa ohjelmaa käytetään sen kohdeympäristössä. Hyväksymistestauksessa tarkastetaan virallisesti, että tuote täyttää sille asetetut vaatimukset. Asiakkaan hyväksyttyä testaustulokset ohjelmisto siirtyy lakiteknisesti asiakkaan omaisuudeksi ja kehityksenaikainen huolto- ja korjausvelvoite lakkaa olemasta voimassa. (Kasurinen 2013, 57.)



KUVIO 4. Hyväksymistestaus (Kasurinen 2013, 58)

Eräässä vuoden 2006 pelikehittäjien konferenssissa pidetyssä luennossa käsiteltiin testauslähtöistä pelikehitystä. Testauslähtöinen pelikehitys tarjoaa välitöntä palautetta koodista, turvallisen mahdollisuuden viimehetken lisäyksiin ja muutoksiin. Lisäksi sen avulla saadaan parempaa koodirakennetta ja koodin dokumentointia. Peliä kehitetään siten, että koodia kirjoitetaan pienissä osissa, jotka testataan välittömästi. Testit on pidettävä helppoina ja todella nopeina, jotta ne eivät häiritse työn etenemistä. (Llopis N. 2006, viitattu 12.4.2014.)

Pelitestaajan on pidettävä silmällä muutamia asioita: Onko peli helppokäyttöinen? Onko siinä järkeä? Toimiiko se? Testaajan työssä on toisaalta kyse bugien etsimisestä ja toisaalta pelattavuuden testaamisesta. Batesin mukaan pelitestaajan tulee ajatella kuin viisi tuhatta eri pelaajaa, joista jokainen pelaa peliä eri tavalla. Keskeistä testauksessa on, että testaaja tekee asioita joka kerralla eri tavalla. Kun testaaja löytää bugin, on todella tärkeää kuvata virheellinen toiminto tarkasti ja ymmärrettävästi. Toistettavuus on bugiraportin tärkein ominaisuus. Kun bugi pystytään toistamaan, se helpottaa ohjelmoijan työtä bugin korjaamiseksi. (Bates 2001, 177-178.)

Käytännössä yhden pelitestaajan on mahdotonta testata peliä kuin viisi tuhatta eri pelaajaa. Jokaisella pelaajalla on oma tapansa pelata ja pelaajan aikaisemmat pelikokemukset ja mieltymykset vaikuttavat asiaan. Manuaalisella testauksella on omat rajoituksensa ja testitapausten laajaa variointia varten on kehitetty testausautomaatio.

Aina bugin toistaminen ei onnistu. Myös Fingersoftille tulee asiakkailta aika ajoin bugiraportteja, joissa kuvattuja bugeja ei yrityksistä huolimatta saada toistettua, vaikka laite ja peliversio ovat samat kuin asiakkaalla. Näissä tapauksissa bugi kirjataan tietokantaan ja asiaan palataan, jos bugi toistuu uudestaan.

Kansainvälinen ohjelmistotestauksen laadun varmistamisen hallitus (International Software Testing Qualifications Board) on sivuillaan kirjannut seitsemän testausperiaatetta. Ne ovat seuraavat:

1) **Testaus paljastaa vikojen olemassaolon:** Testaus voi näyttää, että sovelluksessa on vikoja, mutta ei voi todistaa ettei niitä ole. Testauksen jälkeenkään sovelluksesta ei voida sanoa sen olevan 100 %:n viaton. Testaus kuitenkin vähentää aina todentamattomien vikojen määrää.

2) **Tyhjentävä testaus on mahdotonta:** Kaiken testaaminen, mukaan lukien kaikki yhdistelmät syöttötietoja ja ennakkoehdotuksia on mahdotonta. Tyhjentävän testauksen sijasta pyritään hallitsemaan riskejä ja keskittämään testaukseen käytettävä työpanos priorisoinnin perusteella. Esimerkiksi: Sovelluksessa on yhdellä näytöllä 15 syöttökenttää, joista jokaiselle voi antaa 5 erilaista arvoa. Jos haluttaisiin testata kaikki mahdolliset yhdistelmät, tarvittaisiin 30 517 578 125 (5^{15}) testiä. On hyvin epätodennäköistä, että minkään projektin aikataulu sallisi näin monen testin tekemisen.

3) **Aikainen testaus:** Ohjelmistokehityksen elinkaaressa testauksen pitäisi alkaa niin aikaisin, kuin mahdollista ja sen tulisi keskittyä ennalta määritettyihin tavoitteisiin.

4) **Vikojen klusterointi:** Pieni osa moduuleita sisältää suurimman osan vikoja, jotka löydetään ennen julkaisua tapahtuvassa testauksessa.

5) **Hyönteismyrkky paradoksi:** Jos samankaltaisia testejä toistetaan uudelleen ja uudelleen, jossain vaiheessa samat testitapaukset eivät enää löydä uusia bugeja. Jotta ”hyönteismyrkky paradoksista” ei aiheudu ongelmia, on tärkeää arvioida testilistaa säännöllisesti. Uusien ja erilaisten testien kirjoittaminen täytyy tehdä eri osiin sovellusta tai järjestelmää, näin voidaan lisätä mahdollisuuksia uusien vikojen löytymiseen.

6) **Testaus riippuu testattavan toiminnon luonteesta:** Asiayhteys määrittelee testausmenettelyt. Esimerkiksi erilaisia nettisivuja testaan eritavoin. Sivustoja, joilla korkea turvallisuus vaatimus, testataan eri tavalla kuin verkkokauppasivustoja.

7) **Vikojen puuttumisen harhaluulo:** Jos kehitetty sovellus ei täytä käyttäjän tarpeita ja oletuksia, vikojen löytäminen ja korjaaminen ei auta.
(ISTQB 2014, viitattu 25.1.2014).

Yllämainitut periaatteet istuvat myös pelitestaukseen varsin hyvin. Testauksella voidaan paljastaa bugeja pelistä, mutta 100 %:n virheettömyyteen ei ole mahdollista päästä. Kaikkia mahdollisia testitapauksia on mahdotonta testata aikataulun asettamien rajoitusten ja mahdollisten testitapausten suuren määrän vuoksi. Ohjelmoijat testaavat koodin osia sitä mukaa, kun he saavat pelin koodia valmiiksi. Tällä tavalla varmistetaan, että ohjelmoitu osa toimii, kuten on suunniteltu. Uuden peliversion viat keskittyvät yleensä uuteen sisältöön ja pääosa niistä löydetään testauksessa ja saadaan korjattua ennen peliversion julkaisua.

Hyönteismyrkkyparadoksin torjumiseksi testitapauksia päivitetään ja uusia testitapauksia luodaan tarpeen mukaan. Pelitestauksessa testit on priorisoitu. Tämä tarkoittaa sitä, että tietyt testit on tehtävä jokaisen uuden peliversion yhteydessä, kun taas toisia testejä ei ole välttämätöntä tehdä aina. Testattavan toiminnon luonne vaikuttaa esimerkiksi siten, että verkkopankkiohjelmiston testaami-

sessa turvallisuuteen liittyvien ominaisuuksien testaaminen on keskeistä. Peliä testattaessa toiminnallisuus on tärkeintä. Toisaalta pelialalla pelkkä ohjelman toimivuus, ts. vikojen puuttuminen sovelluksesta ei riitä. Mikäli peli ei viihdytä eikä houkuta pelaamaan, se ei menesty kaupallisesti.

Testaaminen voidaan määritellä listaksi tehtäviä, jotka täytyy tehdä, jotta voidaan arvioida sovelluksen toimivuutta. Testaamista voidaan kuvailla myös prosessina, jolla pyritään paljastamaan virheitä ja osoittamaan, että sovelluksella on tietyn laatuisia ominaisuuksia. Virheitten etsiminen ja poistaminen alkaa testaamisen jälkeen, kun testaaja ilmoittaa sovelluksen virheellisestä toiminnasta. (Burnstein 2003, 7.)

3 TESTAUSSUUNNITELMA

Testaussuunnitelmassa selitetään, mitä testataan. Testitapauksella tarkoitetaan yksityiskohtaista toimenpidettä, joka testaa sovelluksen ominaisuuden tai sen osan kokonaan. Testitapaus selittää, kuinka testi tulee tehdä. Testitapauksessa kuvataan testin tarkoitus, kerrotaan millä laitteella testi tehdään sekä annetaan ohjeet testin tekemiseen ja ilmoitetaan onnistumiskriteerit tai testitulos. (Microsoft 2003, viitattu 15.2.2014.)

Testausta voidaan tehdä ja dokumentoida usealla eri tavalla. Voidaan esimerkiksi kehittää tarkkoja, reseptinkaltaisia testejä tai vaihtoehtoisesti laatia yleisselosteita testitapauksille. Tarkassa dokumentoinnissa testitapauksen vaiheet selitetään tarkasti. Kuvailevissa testitapauksissa testaaja päättää testejä tehdessään, kuinka toteuttaa testit ja mitä dataa käyttää. (Microsoft 2003, viitattu 15.2.2014.)

Useimmiten suositaan yksityiskohtaisia testitapauksia, sillä niissä hyväksymis- ja epäonnistumiskriteerit ovat yleensä helpommin määritettävissä. Yksityiskohtaiset testitapaukset ovat myös helpommin toistettavissa ja helpommin automatisoitavissa kuin kuvailevat testitapaukset. Tämä on tärkeä ominaisuus, jos on esimerkiksi tarkoituksena optimoida asetuksia vertaamalla eri aikoina tehtyjä testejä toisiinsa. Yksityiskohtaiset testit ovat työläämpiä, niiden kehittäminen ja ylläpitäminen vaatii enemmän aikaa. Kuvailevat testitapaukset, joissa menettely on vapaasti tulkittavissa, ovat vaikeita toistaa ja virhettä joudutaan etsimään. Tämä vie aikaa, joka olisi voitu käyttää itse testaukseen. (Microsoft 2003, viitattu 15.2.2014.)

4 REGRESSIOTESTAUS

Regressiotestaus on yleistermi, joka tarkoittaa suunnilleen samaa kuin uudelleentestaaminen. Kun mitä tahansa toimivan järjestelmän osaa muutetaan, ja muutoksen jälkeen halutaan varmentaa, että järjestelmä toimii edelleen oikein, testausta nimitetään regressiotestaukseksi. Regressiotestauksesta voidaan puhua myös silloin, kun kehitettävästä järjestelmästä on saavutettu osatavoite (milestone), ja kehitysversion kaikkien toimintojen oikeellisuus halutaan varmentaa; myös niiden toimintojen, jotka oli korjattu jo aikaisemmassa versiossa. Regressiotestauksen tärkein ominaisuus on todentaa, että jo kertaalleen korjatut ongelmat eivät esiinny osakoodiin tehtyjen muutosten jälkeen, eikä uusia virhetapauksia ilmene. Regressiotestien avulla pystytään myös todentamaan esimerkiksi versionhallinnassa tehtyjen kehityshaarojen yhdistämisen jälkeen se, että kaikki aiemmista osista korjatut virheet ovat poistuneet yhdistystä versiosta. Regressiotestaus ei siis ole mikään yksittäinen testauksen työtaso, vaan yleisnimi kaikelle testaustyölle, jolla varmistetaan, että uusi versio toimii oikein. (Kasurinen 2013, 68-69.)

Regressiotestaus ja siinä suoritettavat perustestitapaukset mahdollistavat testausautomaation käyttämisen. Jos regressiotestit tehdään esimerkiksi jokaiselle pelin pääversiolle tai testit tehdään projektin osatavoitteiden täyttyessä, toistetaan testitapauksia projektin aikana useaan otteeseen. Toisto on eräs testausautomaation keskeisimpiä vaatimuksia. (Kasurinen 2013, 70.)

4.1 Manuaalinen regressiotestaus

Manuaalinen regressiotestaus tarkoittaa ihmisen tekemää testausta, jossa testaaja käy järjestelmällisesti läpi ennalta suunniteltuja testitapauksia. Testaaja tutkii, onko työn alla olevassa ohjelmistossa virheitä, jotka kehitystiimi on kertaalleen korjannut. Testaaja kirjaa virheraportit tietokantaan, josta ohjelmoijat näkevät senhetkiset virhetoiminnot. (Schultz & Bryant 2012, 139.)

Esa Vaarala kuvailee manuaalista testausta seuraavasti: Ihmisen suorittama testaus on ohittamatonta, kun jäljitellään loppukäyttäjän käyttäytymistä sovelluksessa. Aito ihmiskäyttäjä käyttää sovellusta oikein, väärin ja yllättävästi. Mikä tärkeintä, ihmiskäyttäjä kokee tuotteen jollain tavalla ja se herättää hänessä tunteita. Ihmisen käyttäytymistä voi jäljitellä ja testata vain ihminen. Näitä testituloksia käyttämällä tuotteesta on mahdollista kehittää maailmanluokan tuote, joka miellyttää suurta joukkoa ihmisiä. (Vaarala 2012, 6.)

4.2 Automatisoitu regressiotestaus

Testausautomaatio tarkoittaa testaustoiminnan muotoa, jossa testien tekemistä varten rakennetaan automaattityövälineitä. Testausta automatisoitaessa otetaan joukko toistuvasti tehtäviä testitapauksia, ja kehitetään niiden nopeaa tarkastamista varten erillinen menetelmä. Testauksen automatisoinnin tavoitteena on vapauttaa testaajat muihin tehtäviin. Jos tuotteesta tehdään esimerkiksi joka yö uusi käännös (daily build), ei testaajien ajankäytön kannalta ole järkevää käyttää joka päivä aikaa samojen perustestien tekemiseen. Automaattitestit voivat myös olla tarkastuksia, jotka tietokone jätetään yön aikana tekemään, ja kehittäjät voivat aloittaa työpäivänsä läpikäymällä tarkastuksen tulokset ja korjaamalla moduulit joissa havaittiin ongelmia. (Kasurinen 2013, 76.)

Internetin keskustelufoorumilla, jossa pelinkehittäjät kokoontuvat kertomaan kokemuksistaan automatisoidun testauksen hyödyllisyydestä on listattu seuraavanlaisia asioita: automatisoidut testit ovat parantaneet pelin toimivuutta ja kasvattanut kehittäjien ja ohjelmoijien tuottavuutta. Se on myös parantanut koodin laatua ja vähentänyt ylityötunteja. Eräällä toisella foorumin käyttäjällä pikainen savutestaus (smoketest) oli auttanut 100 hengen ohjelmoijatiimiä löytämään pelin rikkovat bugit helposti vaikkakaan testit eivät olleet toimineet niin hyvin monninpelitilanteissa. Kuitenkin henkilön mielestä testit olivat olleet kokonaisuudessaan hyödyksi. (Stack Exchange 2014 viitattu 20.4.2014.)

5 PELIEN VERSIOINTI

Pelikehityksessä versioinnilla tarkoitetaan uuden peliversion luomista. Uuden version mukana tulee usein ominaisuuksia joita aiemmassa versiossa ei ole ollut. Versiointi on kriittinen osa pelinkehitystä. Versioinnin toimenpiteitä ovat uusien ominaisuuksien suunnitteleminen, toteuttaminen, version merkitseminen, versioiden välisten erojen tunnistaminen ja version tallentaminen.

Asiakkaille versiointi näkyy uusina ohjelmistopäivityksinä. Ohjelmistopäivityksiä ei tehdä vain sen takia, että pelin kehittäjä olisi kiirehtinyt tuotteensa kanssa saadakseen sen valmiiksi ja kansan saataville mahdollisimman nopeasti, vaan nykyään laitekanta on niin suuri, ettei ohjelmointi vaiheessa ole yksinkertaisesti mahdollista ottaa kaikkea variantteja huomioon. (Bates 2001, 218.)

Versioinnin yhteydessä korjataan myös bugeja. Asiakkaan lähettäessä palautetta tietystä bugista, johon hän on törmännyt, työskentelee peliyritys yhteistyössä asiakkaan kanssa ongelman ratkaisemiseksi. (Bates 2001, 218.)

Fingersoftilla asiakaspalautteet tulevat pelin käyttötuen tikettijärjestelmään. Käytössä oleva Zendesk-tikettijärjestelmä nopeuttaa tukipyyntöihin vastaamista ja sen avulla on helppo pitää kirjaa asiakkaan kanssa käydystä keskustelusta. Jos tukihenkilö ei pysty ratkaisemaan ongelmaa, hän kirjaa sen JIRA-tehtävienhallintatyökalulla bugiraporttitietokantaan. JIRA-ohjelmiston avulla havaitut bugit pysyvät järjestyksessä. Ongelmille löytyneistä ratkaisuista ja bugien korjaamisesta jää talteen tieto, joka löytyy helposti, jos samantapainen tilanne toistuu.

Pelin versioinnilla pyritään selventämään kehitystä ja helpottamaan käyttäjien käsitystä siitä, mitä versiota he sillä hetkellä käyttävät. Versioinnilla voidaan myös välittää tiedot muutoksista jotka astuvat voimaan version julkaisussa. (Padre 2011, Viitattu 12.4.2014)

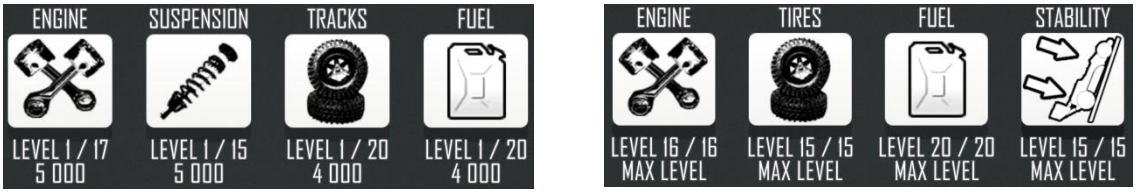
6 HILL CLIMB RACING

Hill Climb Racing -pelissä päähenkilönä on Newton Bill, nuori ja kunnianhimoinen mäkiajokuski. Hän on lähdössä matkalle, joka vie hänet paikkoihin, joihin kukaan ei ole autolla ennen uskaltautunut. Fysiikan lakeja uhmaten Newton Bill ei anna periksi ennen kuin kuun korkeinkin mäki on valloitettu.

Pelissä kilpaillaan yhdeksällätoista haastavalla ja ainutlaatuisella mäkiajoradalla. Pelaaja voi hankkia bonuksia uskaliailla tempuilla ja kerätä kolikoita auton virittämiseen, jotta pelaaja saavuttaisi vieläkin huimempia korkeuksia. Pelaajan kannattaa muistaa kuitenkin olla varuillaan, sillä Billin ajoneuvot eivät ole ihan nuoruuden voimissaan. Ajoneuvot kuluttavat bensan hetkessä ja matka loppuu siihen. Tämän voi kuitenkin välttää keräämällä kentän aikana vastaan tulevia bensakanistereita.

Pelissä on kirjoitushetkellä 19 ajoneuvoa, joissa kaikissa on mahdollista kehittää neljää erilaista ajoneuvon ominaisuutta esim. moottoria, jousitusta, renkaita, nelivetoa. Jokaisella pelin ajoneuvolla on omat uniikit ominaisuutensa.

KUVIO 5. Hill Climb Racing -pelin ajoneuvoja



KUVIO 6. Hill Climb Racing -pelin moottorikelkan ja traktorin ominaisuuksien valikot

Lisäksi ajoneuvot käyttäytyvät eri tavoin ja tarjoavat täten haasteen pelaajalle. Pelin grafiikat ovat värikkäät ja eloisat. Peli on optimoitu näyttämään hyvältä niin matalan kuin korkeankin tason laitteissa. Pelikerran jälkeen voit jakaa oman tuloksesi kavereille.

Hill Climb Racing on mahdollista ladata useista eri kauppapaikoista. Suurimalla osalla Android-laitteista pelin saa joko Google Play -kauppapaikasta tai Samsung Apps -kauppapaikasta. Amazonin e-kirjojen lukulaitteille pelin voi ladata Amazonin tarjoamasta sovelluskaupasta. Applen laitteille pelin saa ladattua Apple Appstoresta. Peli on täysin ilmainen, mutta malttamattomien on mahdollista myös ostaa kolikoita, joilla on mahdollista edetä pelissä nopeampaa.

Pelin ollessa valikoissa näkyy ruudun alalaidassa mainoksia, joista peliyritys saa tuloja. Kun pelaaja ostaa kolikkopaketin, mainokset poistuvat pelistä kokonaan. Kolikkopaketteja on erihintaisia ja mainokset poistuvat jokaisella niistä.

7 KÄYTÄNNÖN OSUUS

Tässä osuudessa esitellään Hill Climb Racing-pelin taulukkomuotoinen testaussuunnitelma. Taulukosta löytyvät kaikki testattavat laitteet ja testitapaukset. Käytännössä ei ole mahdollista hankkia testattavaksi kaikkia olemassa olevia laitteita. Testattavien laitteiden valinnassa otetaan huomioon, onko puhelin uutta vai vanhaa teknologiaa ja puhelinmallin suosio. Testilaitekannassa on eri-ikäisiä laitteita, jotka poikkeavat toisistaan mm. käyttöjärjestelmäversion, näytön koon ja grafiikkapiirin osalta. Jos testaaja saa paljon raportteja tietystä laitemallista, harkitaan laitteen hankintaa, jotta pelin toimivuutta pystytään testaamaan laitteella.

Testausta varten on laadittu taulukko, johon on listattu testitapaukset ja testattavat laitteet. Testaaja testaa yhden laitteen kerrallaan. Testin onnistuessa hän tekee taulukkoon merkinnän kyseisen laitteen ja testitapauksen kohdalle. Jos testi ei onnistu, taulukkoon ei tehdä merkintää. Lopuksi testaajan on helppo poimia ne ominaisuudet, jotka eivät toimineet ehtojen mukaisesti.

Hill Climb-pelin testaamisessa on käytössä myös toinen dokumentti, jossa on listattu testit, niiden ominaisuudet ja se miten ominaisuuksien tulisi käyttäytyä. Dokumentista näkee myös testattavan ominaisuuden prioriteetin. Korkean prioriteetin testit tulee tehdä ennen jokaisen uuden peliversion käyttöönottoa. Opinnäytetyössä ei esitellä tätä dokumenttia, koska se on liikesalaisuus.

Kun aloitin työni pelitestaajana, loin Hill Climb Racingiin vain yksityiskohtaisia testitapauksia. Jossain vaiheessa huomasin, että testitapaukset keskittyivät tiettyyn osaan peliä ja loin niistä testijoukkoja. Testijoukot keskittyvät esimerkiksi pelin toimintoihin, yleiseen toimivuuteen, valikoihin, äänentoistoon, peliennätysten jakamiseen sosiaalisessa mediassa jne. Esittelen seuraavaksi niistä muutamia.

7.1 Toiminnot

Tässä testijoukossa testataan, että kaikki pelin ajoneuvot ovat oikeassa järjestyksessä, eikä kehitysvaiheessa olevia ajoneuvoja näy valikossa. Ajoneuvot tulee myös voida valita. Kenttävalikon tulee toimia siten, että kaikki kentät ovat valittavissa, eikä valikossa ei näy kehitysvaiheessa olevia kenttiä. Tässä testijoukossa myös testataan, että ”Lisää”-sivu ja uusinta toiminto toimii ja ettei ”Lisää”-sivulla ole ylimääräisiä kohtia.

		Verified on:												
ID	Summary	iPhone 3gs (iOS 4.3.2)	iPhone 4 (iOS7)	iPhone 4S(iOS 5.01)	iPhone 5 (iOS 6.0)	iPad 1 (iOS 5.1)	iPad 2(iOS 5.1)	iPad 3 (iOS 6.0.1)	iPad Mini (iOS 6.1.3)	Nexus 7 (Jelly Bean 4.3)	Samsung Galaxy Mini (2.3.4)	Samsung Galaxy S2 (2.3.4)	Samsung Galaxy Y (2.3.6)	HTC Desire X (4.0)
1.1.1.	Vehicles are present													
1.1.2	Stages are present													
1.1.3	More page is present													
1.1.4	Disabling everyplay													

KUVIO 7. Toimintoihin keskittyvä testijoukko

7.2 Yleinen toimivuus

Tämä testijoukko varmistaa sen, että sovellus suoriutuu eri toiminnoista oikein. Testauksella varmistetaan, että ostetut ajoneuvot ja kentät säilyvät päivityksestä huolimatta. Muita testattavia asioita ovat mm. pelin nopeus, pelin sulkeutuminen ja virtuaalirahan ostaminen. Pelin nopeus ei saa heikentyä, pelin tulee sulkeutua oikein, virtuaalirahan oston pitää toimia, kuva ei saa vääristyä näyttöä kiertämällä ja kuvan tulee olla tarpeeksi terävää. Lisäksi testijoukossa on erilaisia testitapauksia, joilla testataan pelin toimivuutta, kun puhelin vastaanottaa puheluita ja viestejä, tai jos pelin aikana tapahtuu muu hälytys.

		Verified on:												
ID	Summary	iPhone 3gs (iOS 4.3.2)	iPhone 4 (iOS7)	iPhone 4S(iOS 5.01)	iPhone 5 (iOS 6.0)	iPad 1 (iOS 5.1)	iPad 2(iOS 5.1)	iPad 3 (iOS 6.0.1)	iPad Mini (iOS 6.1.3)	Nexus 7 (Jelly Bean 4.3)	Samsung Galaxy Mini (2.3.4)	Samsung Galaxy S2 (2.3.4)	Samsung Galaxy Y (2.3.6)	HTC Desire X (4.0)
1.2.1	Vehicle data persistance													
1.2.2	Stage data persistance													
1.2.3	Upgrade data persistance													
1.2.4	Speed of the game													
1.2.5	Exit Function													
1.2.6	Purchase coins fuctionality													
1.2.7	Display orientation													
1.2.8	Display resolution													
1.2.9	Incoming phone call													
1.2.10	Recieving SMS													
1.2.11	Clock/Calendar notifications													

KUVIO 8. Yleiseen toimivuuteen keskittyvä testijoukko

7.3 Äänet

Testijoukko testaa äänten toimivuuden ja varmistaa, ettei kuulokkeiden käyttö sekoita peliä.

		Verified on:												
ID	Summary	iPhone 3gs (iOS 4.3.2)	iPhone 4 (iOS7)	iPhone 4S(iOS 5.01)	iPhone 5 (iOS 6.0)	iPad 1 (iOS 5.1)	iPad 2(iOS 5.1)	iPad 3 (iOS 6.0.1)	iPad Mini (iOS 6.1.3)	Nexus 7 (Jelly Bean 4.3)	Samsung Galaxy Mini (2.3.4)	Samsung Galaxy S2 (2.3.4)	Samsung Galaxy Y (2.3.6)	HTC Desire X (4.0)
1.3.1	Audio options													
1.3.2	Headset use													

KUVIO 9. Ääniin keskittyvä testijoukko

7.4 Sosiaalisen median toiminnot

Testijoukko varmistaa, että sosiaalisen median toiminnot (jakaminen) toimii ja erinäiset valinnat vievät käyttäjän oikeaan paikkaan mahdollistaen sosiaalisen median toiminnot.

		Verified on:												
ID	Summary	iPhone 3gs (iOS 4.3.2)	iPhone 4 (iOS7)	iPhone 4S(iOS 5.01)	iPhone 5 (iOS 6.0)	iPad 1 (iOS 5.1)	iPad 2(iOS 5.1)	iPad 3 (iOS 6.0.1)	iPad Mini (iOS 6.1.3)	Nexus 7 (Jelly Bean 4.3)	Samsung Galaxy Mini (2.3.4)	Samsung Galaxy S2 (2.3.4)	Samsung Galaxy Y (2.3.6)	HTC Desire X (4.0)
1.4.1	Everyplay share													
1.4.2	Facebook like													
1.4.3	Twitter Follow													
1.4.4	Google+ Follow													

KUVIO 10. Sosiaalisen median toimintoihin keskittyvä testijoukko

8 POHDINTA

Oppinäytetyön tavoitteena oli luoda Fingersoftille toimiva testausdokumentti Hill Climb Racingin -peliin. Hill Climb Racing -pelin testaussuunnittelun tarkoituksena oli kehittää ja ohjeistaa testausmenettelyt niin, että jonain päivänä testaukseen perehtymätön henkilökin pystyisi suorittamaan testauksen. Kehittämistehtävä käsitti kahden testausdokumentin luomisen. Toinen kehitetyistä dokumenteissa esitellään opinnäytetyössä.

Olin todella onnekas, että sain kesätöitä Fingersoftilta ja sitä kautta opinnäytetyöni aiheeksi näin mielenkiintoisen työn. Olin jo aikaisemmin toiminut vapaaehtoisena pelitestaajana. Työssäni kiinnostuin koko ajan enemmän testauksesta ja sen menetelmistä, ja sain sitä myötä uutta potkua myös opiskelujen loppuun saattamiseen.

Yritykseltä löytyi jo eräässä vanhassa projektissa käytetty testausdokumentti, jota käytin pohjana, kun aloin työstämään Hill Climb Racing -peliin sopivaa testausdokumenttia. Aluksi testitapausten keksiminen oli hankalaa, sillä suuri osa testattavista ominaisuuksista oli niin itsestään selviä, etten mielestäni saanut rakennettua niille hyviä testejä. Yhdeksi haasteeksi muodostui myös ehtojen rakentaminen siten, että testattava ominaisuus tuli testattua oikein. Saatuani vanhemmilta kollegoilta ohjeita ja neuvoja testitapausten muodostaminen selkiytyi.

Syksyllä 2013 laatimani opinnäytetyön aikataulu osoittautui liian tiukaksi. Tasapainottelu työelämän ja opinnäytetyön välillä oli todella rankkaa. Loppuvaiheessa opinnäytetyön kirjoittaminen tuntui raskaalta, enkä meinannut uskoa saavani sitä valmiiksi. Onneksi perheeni ja työkaverini tukivat minua. Sain heiltä motivaatiota saattaa työ loppuun. Työni eteni lyhyin pyrähdyksin. Se venytti alkuperäistä aikatauluani.

Tavoitteeseen päästiin, ja opinnäytetyön kehittämistehtävä onnistui. Kehitetyt testausdokumentit ovat käytössä Fingersoftilla ja niihin on oltu tyytyväisiä. Ne

helpottavat uusien peliversioiden kehittämistä. Aikaisemmin yrityksellä ei ollut järjestelmällisen testauksen käytänteitä. Luodut käytänteet ja menettelyt nopeuttavat ja tehostavat versiokehitystä sekä parantavat tuotteen laatua.

Opinnäytetyötä tehdessäni huomasin, että bugien etsintä ja käytettävyyden testaaminen sivuavat toisiaan monelta osin. Bugien etsintä on osa käytettävyyden testausta. Pelin on oltava mahdollisimman bugivapaa, jotta sitä olisi mielekästä pelata. Samalla on huomioitava pelin muu käytettävyys: jos siinä esiintyy puutteita, ei peli välttämättä vedä pelaajia puoleensa. Muiden käytettävyyden osa-alueiden testaaminen voisikin olla toisen opinnäytetyön aihe.

Tein opinnäytetyöni Fingersoft Oy:lle palkattuna työntekijänä. Sain työn loppuvaiheessa yrityksestä vakituisen työpaikan, mistä olen erityisen tyytyväinen. Fingersoftilla jatketaan opinnäytetyön yhteydessä tehdyn pelitestausdokumentin kehitystä. Se tulee olemaan yksi tulevista työtehtävistäni.

LÄHTEET

Bates, B. 2001.Game Design: The art & business of creating games. Prima publishing.

Burnstein, I. 2003. Practical Software Testing. New York: Springer-Verlag, Inc.

ISTQB International Software Testing Qualifications Board. 2014. What are the principles of testing?. Viitattu 25.1.2014 http://istqbexamcertification.com/what-are-the-principles-of-testing/.

Kasurinen, J. P. 2013. Ohjelmistotestauksen käsikirja. Jyväskylä. Docendo.

Llopis, N. 2006. Backwards is Forward: Making Better games with Test-Driven Development. Viitattu 12.4.2014, http://gamesfromwithin.com/backwards-is-forward-making-better-games-with-test-driven-development

Microsoft. 2003. Designing Test Cases. Viitattu 15.2.2014, http://technet.microsoft.com/en-us/library/cc782852(WS.10).aspx.

Nuutila, E. 2014 Ohjelmoinnin peruskurssi Y2 Luento 2 Testaus. Viitattu 12.4.2014 https://noppa.aalto.fi/noppa/kurssi/cse-a1121/luennot/CSE-A1121_luentokalvot_2.pdf

Padre, J. 2011 Game developer tip 5: Using proper versioning. Viitattu 12.4.2014 http://developer.sonymobile.com/2011/10/06/game-developer-tip-5-using-proper-versioning/

Schultz, C. P. & Bryant, R. D. 2012. Game Testing All In One. Dulles, Virginia. Mercury Learning and Information

Stack Exchange. 2014. Automated testing of games [closed]. Viitattu 20.4.2014 http://gamedev.stackexchange.com/questions/574/automated-testing-of-games

Vaarala, E. Testiautomaatio on kuormitustestauksen kaveri. Laatu ja testaus 2/2012, 6 Testauksen osaamisyhteisö. Viitattu 12.4.2014 http://testausosy.fi/wp-content/uploads/2012/11/LT-Vol1Ed2.pdf