Jani Rissanen

TELEMETRINEN TIEDONKERUUJÄRJESTELMÄ RUST0 GAMES OY:LLE

Opinnäytetyö
Kajaanin ammattikorkeakoulu
Tradenomi
Tietojenkäsittelyn koulutusohjelma
Kevät 2013

KAJAANIN AMMATTIKORKEAKOULU
UNIVERSITY OF APPLIED SCIENCES

OPINNÄYTETYÖ
TIIVISTELMÄ

Koulutusala Luonnontieteiden ala	Koulutusohjelma Tietojenkäsittelyn koulutusohjelma
Tekijä(t) Jani Rissanen	
Työn nimi Telemetrinen tiedonkeruujärjestelmä Rust0 Games Oy:lle	
Vaihtoehtoiset ammattiopinnot Peliohjelmointi	Ohjaaja(t) Matti Härkönen
	Toimeksiantaja Rust0 Games Oy
Aika 10.12.2012	Sivumäärä ja liitteet 20+7

Opinnäytetyn tavoitteena oli tuottaa Rust0 Games Oy:lle järjestelmä, jonka avulla yritys voi seurata ohjelmiston käytön tunnuslukuja. Tärkeimpänä ominaisuutena oli mahdollistaa iteratiivinen pelinkehittäminen, ohjelmiston avulla kerättävän metriikan kautta.

Työssä kerrotaan aluksi lyhyesti mitä telemetriikka on, sekä mihin ja miten sitä nykyään käytetään. Opinnäytetyössä käydään myöhemmin läpi perinteisen tiedonhankinnan menetelmät, sekä niiden hyvät ja huonot puolet. Etsin ja tutkin eri Java-pohjaisten tietokantojen suorituskykymittauksia, myöhemmin tehtävää projektiosuutta varten. Kerron myös erilaisten kuvaajien tyypit, ja missä tilanteessa mitäkin erilaista kuvaajatyyppiä yleensä käytetään.

Projektin tuotteeksi tuli melkein valmis ohjelmisto, jonka avulla Rust0 Games pystyy keräämään anonyymisti kvantitatiivista tilastoa käyttäjien toiminnasta. Lopullinen tuote pystyi hyödyntämään toisen tradenomiopiskelijan Juha Lampénin opinnäytetyöksi tekemää verkkorajapintaa. Lopulliseen tuotteeseen kuuluu tiedonsiirto, -tallennus ja -esittäminen. Opinnäytetyössä esitellään lopullista tuotetta, ja sen tuottamia kuvaajia. Lopullinen tuote on edelleen jatkokehityskelpoinen, koska yhden henkilön tekemänä ja suunnittelemana aika ei riittänyt kuin ainoastaan välttämättömien ominaisuuksien tekemiseen.

Kieli	Suomi
Asiasanat	Testaus, tiedonkeruu, telemetria, analyysi
Säilytyspaikka	☐ Verkkokirjasto Theseus ☐ Kajaanin ammattikorkeakoulun kirjasto

KAJAANIN AMMATTIKORKEAKOULU
UNIVERSITY OF APPLIED SCIENCES

THESIS
ABSTRACT

School Natural Sciences	Degree Programme Data Processing
Author(s) Jani Rissanen	
Title Telemetric datagathering system for Rust0 Games	
Optional Professional Studies Game programming	Instructor(s) Matti Härkönen
	Commissioned by Rust0 Games Oy
Date 10.12.2012	Total Number of Pages and Appendices 20+7

The goal of this thesis was to produce a system which Rust0 Games could use to gather and follow software usage figures. In addition, the most important feature of the system is to provide a method for iterative game development by means of gathering and analyzing metrics.

This work shortly explains what telemetrics is and for what and how it is used these days. Later in this thesis, traditional ways of data collection are explained, as well as the pros and cons for these methods are described. Different Java-based database performance measures were searched and and researched for the future project. Also, different types of charts were described, and for what and where these chart types are typically used.

The final product of this thesis was almost fully functional software which Rust0 Games now uses to gather anonymous quantitative statistics about the behaviour of their users. This software utilized another bachelor of business student's, Juha Lampén's thesis of a network interface. This product consists of data transfer, storage and visualization. In this thesis the final product is introduced, and a few charts made with this software are demonstrated. The end product of the thesis was left a bit underdeveloped because it was made by one person, and the planned timetable was enough for only the necessary features.

Language of Thesis	Finnish
Keywords	Testing, datamining, telemetry, analysis
Deposited at	☐ Electronic library Theseus ☐ Library of Kajaani University of Applied Sciences

SYMBOLILUETTELO

SQL	*Structured Query Language.* Tietokantakieli; useat tietokanta-ratkaisut ovat SQL-pohjaisia.
AB-testi	Testausmenetelmä, jolla arvioidaan jonkin ominaisuuden toimivuutta tarjoamalla toiselle ryhmälle testattavaa ominaisuutta ja toiselle ei.
Unity 3D	Pelimoottori, joka mahdollistaan nopean ja ketterän pelinkehityksen.
JDK	*Java Development Kit.* Oracle Corporation:n ylläpitämä ympäristö Java-koodien kääntämiseen.
JRE	*Java Runtime Environment.* Käännettyjen Java koodien suorittamiseen vaadittava ohjelmisto.

SISÄLLYS

1 JOHDANTO

Opinnäytetyöni toimeksiantaja on Kajaanilainen pelialan yritys Rust0 Games Oy. Yrityksen on perustanut kaksi aikaisemmin Kajaanin ammattikorkeakoulussa opiskellutta henkilöä. Yritys tarvitsi jonkinlaisen järjestelmän, jonka avulla voidaan tehdä kehitteillä olevista peleistä parempia, sekä voidaan muodostaa erilaisia avainlukuja pelien tilasta.

Opinnäytetyöni tarkoituksena on antaa Rust0 Games:lle työkalut tilastojen automaattiseen keräämiseen, sekä kerätyn aineiston tulkintaan. Opinnäytetyön aikana olen tutustunut erilaisiin tiedonhankintamenetelmiin, ja siihen millaista tietoa on mahdollista kerätä automaattisesti.

Opinnäytetyön tavoitteena on tutustua ja selvittää yleisesti käytettyjä menetelmiä tunnuslukujen hankintaan. Tarkoituksena on myös tutkia, kuinka iteratiivinen pelinkehitys tapahtuu käytännössä.

2 TELEMETRIIKKA JA TIEDON TALLENTAMINEN

Telemetriikka juontaa juurensa kreikan kielen sanoista ”tele” ja ”metron”, jotka tarkoittavat suomennettuna ”etäistä” ja ”mittausta”. Yleensä telemetria mielletään nimenomaan langattomaksi tiedonsiirroksi (radioaallot, infrapuna, jne.), mutta pelkästään sitä se ei kuitenkaan ole, vaan myöskin kaikki fyysistä tietä kulkeva tieto lukeutuu telemetriseen tiedonsiirtoon. (Wikipedia, 2012)

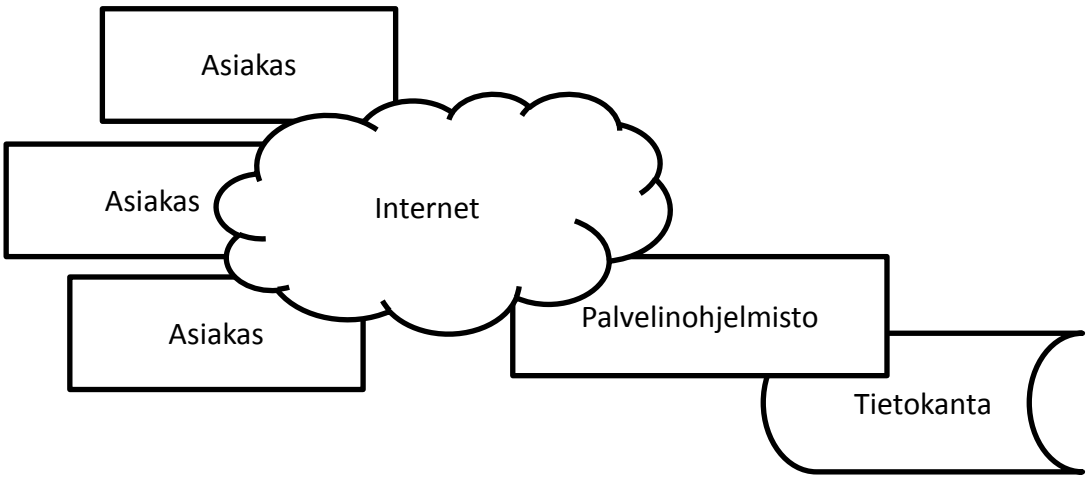
Telemetristä tiedonhankinnan menetelmää sovelletaan nykyään useassa eri käytännön sovelluksessa. Esimerkiksi sairaaloissa potilaiden tilaa voidaan tarkkailla etänä. Jos potilaalla on jokin ongelma tai hätä, voidaan se huomata ilman, että tarvitsee paikan päällä käydä tarkastamassa potilaan tila. Muita telemetriikkaa hyödyntäviä merkittäviä sovelluksia on:

- Meteorologia
- Öljynporaus
- Moottoriurheilu
- Maanviljely

Pelinkehityksen maailmassa telemetriikka on kuitenkin vielä lapsenkengissään, mutta sen suosio on valtavalla vauhdilla kasvamassa. Alkuaikoina telemetriikkaa käytettiin apuna luomaan mahdollisimman ihmismäisesti käyttäytyviä tekoälyjä. Myöhemmin sen on huomattu olevan hyödyllinen myös muissa käytännön ongelmissa. Tietoa kerätään lähinnä tuotoista, laitteistosta, sekä kehitysprosesseista. Tärkeimmäksi on kuitenkin noussut pelaajien käyttäytymisen seuranta, joka on myöhemmin osoittautunut yhdeksi tärkeimmistä pelinkehitysprosessin tekijöistä. Nykyään tätä menetelmää käytetäänkin mallintamaan ja selvittämään sitä kuinka pelaajat pelaavat pelejä. (Bauckhage C, Kersting K, Sifa R, Thurau C, Drachen A & Canossa A, 2012)

Kerättyä aineistoa voidaan myös hyödyntää toiseenkiin suuntaan; annetaan pelaajan itsensä, sekä muiden pelaajien nähtäväksi, mitä tietoa heistä on kerätty pelaamiseen liittyen. Tätä menetelmää jo hyödynnetäänkin nykyään esimerkiksi peleissä ansaittavien saavutusten muodos-

sa. Näitä kerättyjä tietoja yhdistetään isommiksi kokonaisuuksiksi ja käytetään pelaajien välisessä vertailussa. Saavutusten lisäksi näissä esilläolevissa tilastoissa voidaan näyttää parhaiden pelaajien listoja, pelin sisäisiä tapahtumia, ryhmiin kuulumisia, jne. Pelaajien pelikokemukseen voidaan siis vaikuttaa tarjoamalla tällaisia tilastoja pelaajien saataville. (Medler, 2012)



Kuvio 1. Opinnäytetyön telemetrisen tiedonsiirron ja –tallennuksen kuvaus

Opinnäytetyöni viitekehystä esittää Kuvio 1, josta käy ilmi kuinka järjestelmän kokonaisuuden on tarkoitus muodostua. Tietoa siirretään telemetrisin keinoin mobiileista päätelaiteista palvelinlaitteiston ohjelmistolle Internetin välityksellä, joka tämän jälkeen tallentaa kaiken kerätyn tiedon tulevia tutkimuksia varten. Tiedonsiirtoväylänä tälle järjestelmälle toimii Internet. Tiedontallennusvälineisiin lukeutuu sekä itse päätelaitteet, että palvelimella sijaitseva tietokanta, johon tieto lopulta päätyy.

Tarkoituksena on kerätä pelin aikana tapahtuvista asioista haluttuja tietoja, jolloin voidaan tehdä johtopäätöksiä pelin toiminnasta. Tiedon tallentamiseen käytetään jonkinlaista tietokantasovellusta. Koska palvelinalustan käyttöjärjestelmänä toimii linux -pohjainen ratkaisu ja palvelimen ohjelmointiin käytetään Javaa, tulee tietokannan käyttöön valita sellainen ohjelmisto, joka toimii nopeasti ja luotettavasti edellä mainitulla kokoonpanolla.

Kun Javalla ollaan muodostamassa millaista tahansa tietokantayhteyttä, sen hoitaa JDBC- (Java Database Connectivity) ajuri, joka mahdollistaa keskustelun Javasta käsin yhdistettävään kantaan. Tietokantavalinnaksi oli muodostunut HSQLDB, joka on kevyt ja nopea, täy-

sin Java pohjainen tietokantasovellus. Kyseisen kannan toiminnasta on myös tehty tehokkuustutkimuksia PolePoint -nimisellä työkalulla. Näistä testiraporteista käy ilmi, että HSQLDB voi olla äärimmäisen nopea, varsinkin kun käytetään muistinvaraisia (cached) tauluja, joista tieto viedään kovalevylle pysyvään muistiin. Liitteestä 2 voidaan havannoida suorituskykyjen eroja muihin JDBC:lla toteutettuihin ratkaisuihin. Kuvioissa pienempi luku tarkoittaa parempaa. (HSQLDB, 2012)

Näiden kuvioiden nopeudet eivät kuitenkaan ole täysin vertailukelpoisia, koska testit on suoritettu niin, että HSQLDB on muista tietokantasovelluksista poiketen, käyttänyt muistinvaraisia taulurakenteita. Tämä tarkoittaa sitä, että HSQLDB:n ei ole tarvinnut tehdä kiintolevylle luku- ja kirjoitusoperaatioita. Tuotantokäytössä kuitenkin käytetään näitä nopeita muistinvaraisia tauluja. (HSQLDB, 2012)

HSQLDB:n suorituskykyä on kuitenkin testattu PolePoint:lla myös niin, että HSQLDB ei ole käyttänyt muistinvaraisia tauluja, mitkä ovat hitaampia. Liitteessä 3 nähdään vertailukelpoisemmat testitulokset. Kuvioissa mitataan suoritettujen käskyjen lukumäärää ennaltamääritetyssä ajassa, mikä merkitsee sitä, että kuvioissa suurempi luku tarkoittaa parempaa. Kuitenkin näitä tuloksia tulkittaessa tulee ottaa huomioon, että muistinvaraisten taulujen nopeustesteissä on käytetty viimeisintä versiota HSQLDB:sta (HSQLDB-2.2.6), kun taas ei-muistinvaraisten taulujen nopeustestissä on käytetty huomattavasti vanhempaa versiota HSQLDB:sta (HSQLDB-1.8.1). (HSQLDB, 2012)

3 AINEISTON HANKINTA JA VISUALISOINTI

Tehtävänä oli tehdä tiedonkeruujärjestelmä, jota voitaisiin hyödyntää tiedon keräämisessä, sekä tiedon analysoimisessa. Seuraavissa kappaleissa kerrotaan tiedon keräämisestä, sekä siitä kuinka erilaisia tietoja on hyvä visualisoida, jotta johtopäätökset olisivat mahdollisimman helppo tehdä.

3.1 Aineiston hankintamenetelmät

Aineiston hankinnan menetelmiin kuuluu lukuisia eri tapoja toteuttaa kyseinen toimenpide. Se, että millaista aineistoa halutaan, kvantitiivista tai kvalitatiivista, on valittavissa erilaisia menetelmiä aineiston keräämiseen. Opinnäytetyössäni kuitenkin toteutan automaattisen järjestelmän jonka avulla voidaan helposti hankkia kvantitiivista aineistoa, eli määrällistä aineistoa. Kerron eri tiedonhankintatavoista ja niiden hyödyistä, sekä siitä milloin niitä tulisi käyttää.

3.1.1 Kysely- ja monivalintalomakkeet

Yleisesti tällaisia lomakkeita käytetään, kun halutaan saada kattava määrä tietoa ihmisten mielipiteistä. Kysely- ja monivalintalomakkeiden hyötyjä on, että täyttäjä voi tehdä lomakkeen anonyymisti. Se myös on kyselyn laatijalle halpaa, ja tuloksia on helppo verrata keskenään. Lomakkeita on vaivatonta jakaa useille vastaajille, ja niiden avulla voidaan saada valtava määrä tietoa. (McNamara, 2012 & Teach-ICT a, 2012)

Täydellinen tällainen lomakekysely ei kuitenkaan ole. Kyselyn tuloksia voi vääristää esimerkiksi kysymysten huono muotoilu, missä vastaaja ei välttämättä täysin ymmärrä kysymystä, ja vastaa silti. Vastaaja voi myös olla huolimaton kirjoittaessaan tai rastittaessaan vastauksia, jolloin vastauksissa voi olla harhaa. Nämä eivät myöskään tarjoa mahdollisuutta kertoa koko totuutta asiasta. (McNamara, 2012 & Teach-ICT a, 2012)

3.1.2 Haastattelut

Haastattelut tehdään silloin, kun halutaan saada yksityiskohtainen kuva siitä, miten asiakas näkee tuotteen, tai mitkä ovat henkilökohtaiset mielipiteet tuotteesta tai sovelluksesta. Haastattelut ovat hyödyllisiä, koska niistä saadaan laaja kuva siitä, mikä on pielessä ja mikä on kohdallaan. Haastattelemalla saadaan myös yhteys asiakkaaseen ja hänen asiakkuuteen. (McNamara 2012, Whorton, 2009)

Haastattelujen tekemisen huonoihin puoliin kuuluu kuitenkin se, että koska haastattelut suoritetaan kasvotusten asiakkaan kanssa, voi asiakas koettaa miellyttää haastattelijaa tai haastattelija voi tarkoituksettomasti johdatella haastateltavaa. Haastateltavien hankinta voi olla kallista riippuen siitä, kuinka pitkiä haastatteluja ollaan tekemässä ja kuinka paljon. Haastattelujen suorittaminen voi olla myös hyvin aikaa vievää, koska ensin tulee tehdä suunnitelma haastattelusta, ja sen jälkeen haastatella haastateltavaa, jonka jälkeen haastattelua analysoidaan. (McNamara, 2012 & Whorton, 2009)

3.1.3 Tarkkailu

Tarkkailua tehdään kun halutaan nähdä kuinka jokin oikeasti toimii. Tarkkailun etuihin kuuluu, että tuloksia saadaan välittömästi samalla kun tarkkailua suoritetaan. Tarkkailua käytetään yleensä prosessien tutkimisessa. Tarkkailun etuihin lukeutuu myös se, että ongelmatilanteisiin voidaan reagoida reaaliajassa. (McNamara, 2012 & Teach-ICT b, 2012)

Tarkkailemalla saadut tulokset voi olla kuitenkin hankalia kategorisoida. Ja nähdyt tulokset tai asiat voivat olla hankalia muuntaa käsiteltävään muotoon. Tarkkailutilaisuuksien järjestäminen voi olla kallista. Tarkkailijan läsnäolo voi myös tarkoituksettomasti vaikuttaa osanottajan suoriutumiseen tai ohjelmistonkäyttöön. (McNamara, 2012 & Teach-ICT b, 2012)

3.1.4 Fokusryhmät

Fokusryhmillä voidaan saada ongelmallisiin kohtiin perusteelliset vastaukset ryhmäkeskustelun avulla. Henkilöt osaavat ja uskaltavat kertoa omat ajatuksensa, kun ryhmän muut jäsenet ovat tukena. Ryhmältä voidaan saada kattava analyysi, kun ryhmältä kysytään esimerkiksi reaktioista testattavaan kohteeseen tai ehdotukseen. Nämä ryhmät ovat oleellisesti erinomaisia tulkitsemisessa ja markkinoinnin tukemisessa. Fokusyhmiä järjestämällä voidaan nopeasti saada yleinen mielipide tutkittavasta asiasta. Fokusrymät ovat tehokas tapa saada laaja kuva ja syvällinen tieto lyhyessä ajassa, ja voi johtaa ratkaiseviin päätöksiin testattavissa asioissa. (McNamara, 2012 & Students Speak Toolkit, 2012)

Fokusryhmien keskustelut voivat edetä vauhdilla ilman kunnollista moderointia, jolloin tulokset voivat olla hankalia analysoida. Fokusryhmään osallistuvat henkilöt toimivat anonyymisti osallistuessaan tällaiseen tilaisuuteen, joten he tarvitsevat hyvät ja turvalliset tilat. Lisäksi myös kuudesta kahdeksaan toisilleen tuntemattomien ihmisten hankkiminen samaan paikkaan ja samaan aikaan voi olla hankala järjestää. (McNamara, 2012 & Students Speak Toolkit, 2012)

3.1.5 Tapaustutkimus

Kun halutaan täysin ymmärtää tai kuvata asiakkaan kokemukset, voidaan silloin lähteä ratkaisemaan ongelmaa tapaustutkimuksen avulla. Tapaustutkimuksen avulla saadaan muodostettua kattava tutkimus tekemällä erilaisia tutkimuksia eri tekijöistä. Tapaustutkimuksen etuina on, että saadaan lähes täydellinen kuva siitä mitä on koettu ja mitä on saavutettu. (McNamara, 2012 & Soy, 1996)

Tapaustutkimusten tekeminen vie yleensä paljon aikaa, koska hyvän tapaustutkimuksen tekeminen vaatii laajaa tiedonhankintaa, -järjestelyä ja -kuvausta. Tapaustutkimuksella ei myöskään välttämättä saavuteta uusia ratkaisuja ongelmiin, vaan pikemminkin voidaan kuvata nykyisiä ongelmia tarkemmin. (McNamara, 2012 & Soy, 1996)

3.1.6 A/B-testaaminen

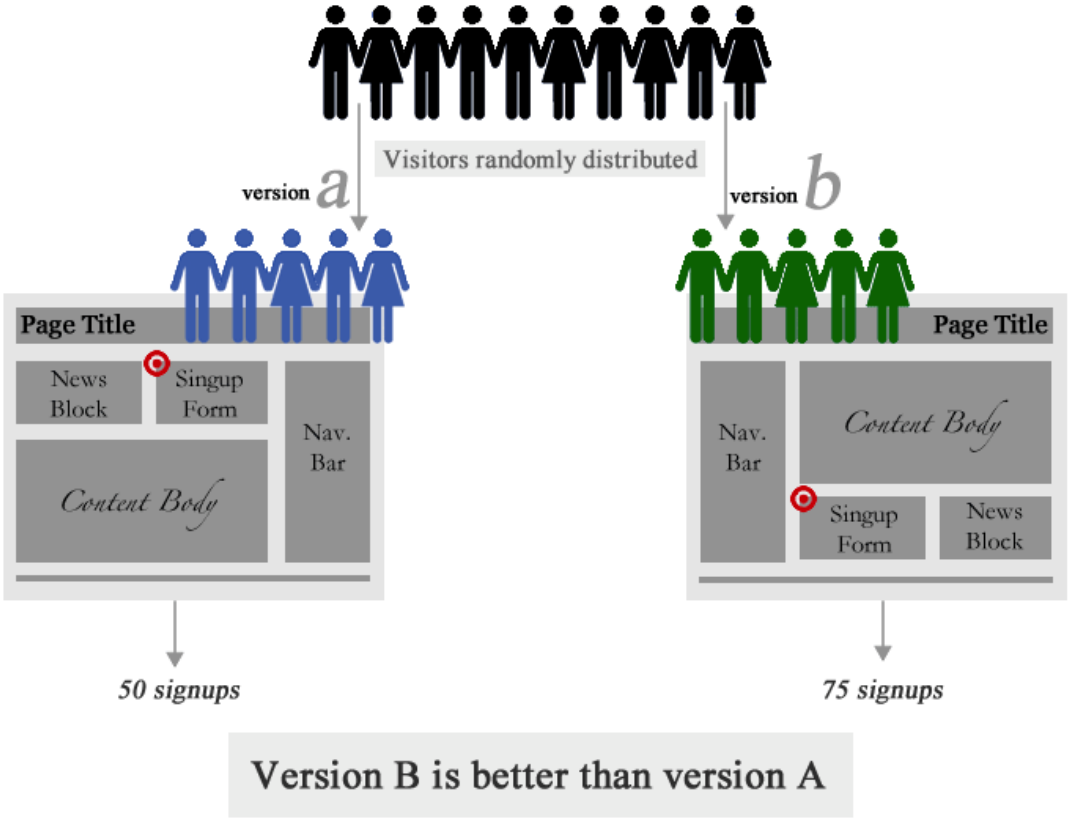
Muun aineiston keräämiseen yhteydessä kannattaa aina kun on mahdollista toteuttaa samalla A/B testausta. A/B testaus tarkoittaa sitä, että jaetaan testaajat kahteen ryhmään, ja samoja asioita testattaessa muutetaan joitain yksittäisiä asioita, jotka vaikuttavat testin kulkuun. Esimerkiksi voidaan vaihtaa vaikkapa jonkin painikkeen väri. Tällainen testi mahdollistaa varsinkin uusien asioiden paremmuuden testaamisen. (Chopra, 2010)

Jos testattavia vaihtoehtoja on useampia, tai jos halutaan testata erilaisten muutosten yhdistelmiä, voidaan A/B testaus laajentaa A/B/N testaukseen, missä N tarkoittaa haluttua määrää erilaisia testivaihtoehtoja. (Chopra, 2010)

Kun A/B testausta suoritetaan, tulee eri vaihtoehdot testata aina samanaikaisesti jotta saataisiin täsmällinen kuva siitä, mikä vaihtoehto on todellakin paras. Jos testaus suoritetaan eri versioille peräkkäisinä viikkoina voi tuloksien laatu ja määrä vaihdella eri syiden seurauksena. Eli testattaessa tulee aina jakaa samanaikaiset testaajat kahteen tai useampaan samankokoiseen ryhmään. (Chopra, 2010)

Ominaisuuksien testaaminen tällä tavalla olisi hyvä suorittaa ainoastaan uusilla asiakkailla. Vanhat asiakkaat saattavat yllättyä tai säikähtää uudesta asiasta, joka ei välttämättä edes tule lopulliseen tuotteeseen. Jos testataan vanhoillakin asiakkuuksilla, niin testattavien asioiden pitäisi olla niin pienimuotoisia, etteivät ne häiritse vanhoja asiakkaita. Uusille asiakkaille tulisi myös näyttää aina sama variaatio testattavasta asiasta. Jos käyttäjälle näytetään ensin yhdenlainen näkymä, ja seuraavana päivänä sama näkymä onkin toisenlainen, voidaan siinä menettää asiakas, joka on hämmentynyt jatkuvasti muuttuvan käyttöliittymän takia.
(Chopra, 2010)

A/B-testejä kannattaa suorittaa myös ilmiselville asioille. Tulokset voivat hyvinkin yllättää. Jokin asia voi näyttää kehittäjän silmään erittäin huonolta ominaisuudelta, mutta tuhannet asiakkaat saattavat kokea tämän ominaisuuden ohjelmiston myyntivaltiksi. Oma mielipide ei siis saisi, eikä saa vaikuttaa mitattuihin tuloksiin. Kuvio 2 tiivistää A/B testauksen perusperiaatteen Paras Chopran mukaan. (Chopra, 2010)



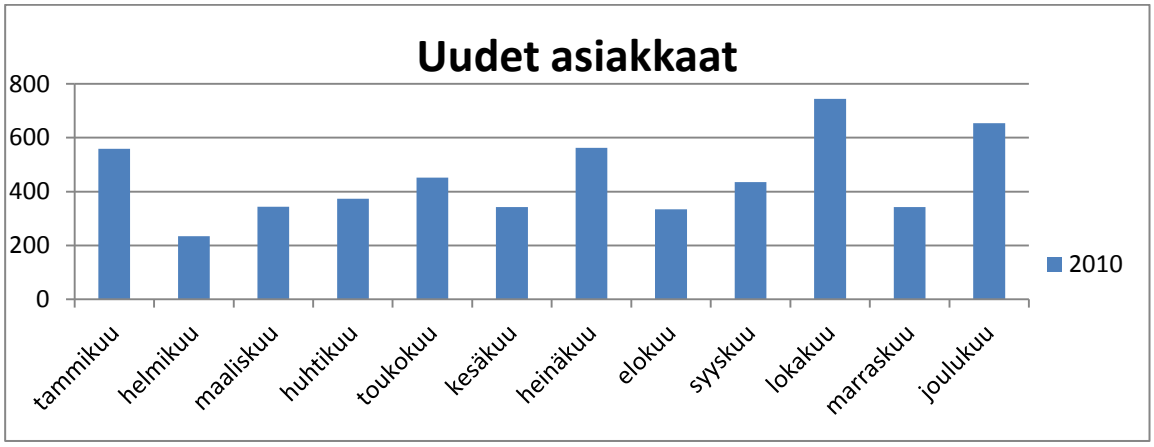
Kuvio 2. A/B testauksen perusperiaate

3.2 Kerätyn tiedon visualisointi

Jotta kerättyä tietoa voisi jotenkin analysoida, kannattaa se visualisoida jollain tavalla. Selkein tapa esittää tilastoja, on muodostaa niistä erilaisia havainnollistavia kuvia. Niistä käy hyvin ilmi, että miten asiat oikeasti ovat. Kerätyn aineiston luonteesta riippuen on olemassa erilaisia kuvaajia, jotka sopivat eri asioiden näyttämiseen.

3.2.1 Pylväskuvaaja

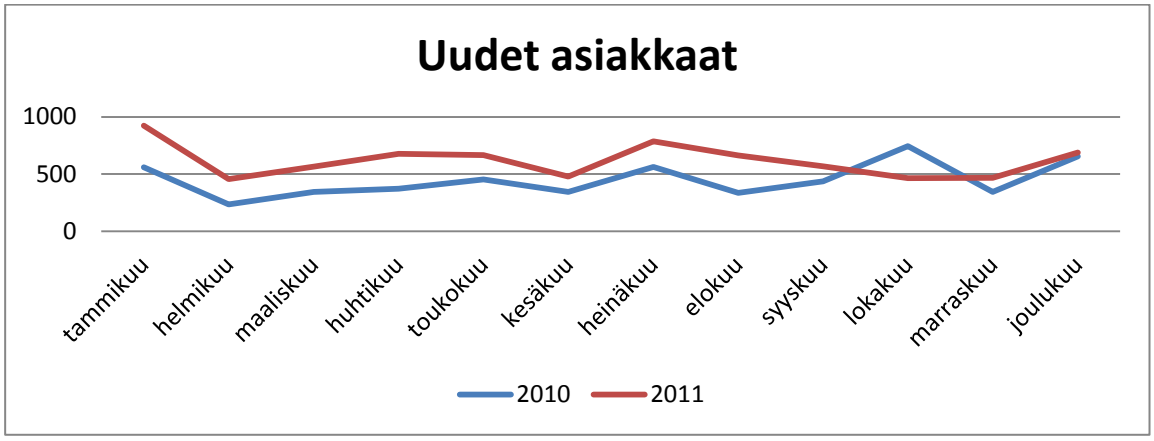
Pylväskuvaajan muodostaminen on yksi tapa näyttää tilastoja graafisesti. Näiden avulla voidaan erinomaisesti verrata vaikkapa jonkin asian eri kertojen tuloksia. Tai voidaan esittää yhden ominaisuuden hyvyys verrattuna muihin ominaisuuksiin. Pylväskuvaajalla voidaan myös näyttää jatkuvaa jakaumaa. Tällaista kuviota kutsutaan histogrammiksi. Histogrammiin ei jätetä väliä pylväiden välille. Kuvio 3 havainnollistaa miltä pylväskuvio näyttää. (Laininen, 2004 & Nadhani, 2009)



Kuvio 3. Esimerkki pylväskuvaajasta

3.2.2 Viivakuvaaja

Tällaisia kuvaajia käytetään usein aikaan sidotuissa tilastoissa. Näillä kuvaajilla voidaan esimerkiksi ilmaista vuoden tuloja verrattuna edellisen vuoden tuloihin, kahdella eri viivalla samassa kuvaajassa. Viivakuvaajasta myös näkee selkeämmin, onko suunta ollut nouseva vai laskeva, varsinkin kun muutokset ovat verrattaen pieniä. Kuvio 4 havainnollistaa miltä viivakuvaaja näyttää. (Laininen, 2004 & Nadhani, 2009)



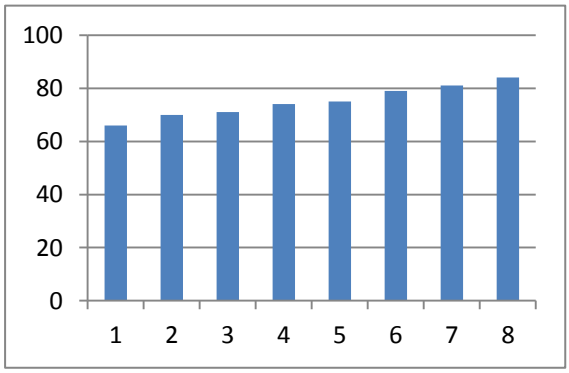
Kuvio 4. Esimerkki viivakuvaajasta.

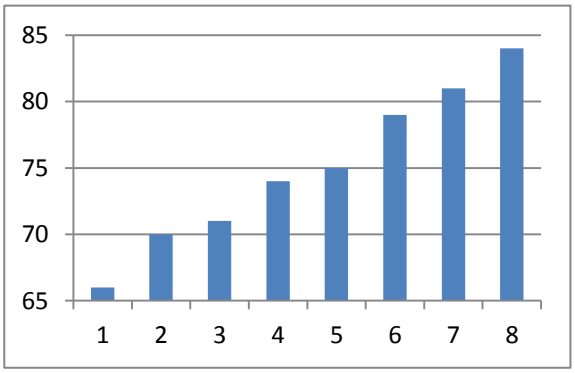
3.2.3 Sektorikuvaaja

Sektorikuvaajat eli yleisemmin piirakkakuviot ovat erinomaisia näyttämään, kuinka jokin kokonaisuus muodostuu pienemmistä asioista. Tällaisesta kuvaajasta näkee nopeasti, jos jokin osa-alue on selkeästi suurempi kuin toinen. Sekä myös päinvastoin jos jokin osa-alue on erittäin pieni tai heikko. (Laininen, 2004 & Nadhani, 2009)

3.2.4 Kuvaajien väärintulkinta

Esitetyt kuvaajat voivat myös kuitenkin vääristää totuutta. Kun tarkemmin katsotaan kuvaajassa käytettyä arvoaluetta, voidaan hyvin lähellä toisiaan olevien arvojen erotus todeta suuremmaksi, kuin mitä ne todellisuudessa ovat. Kuvio 5 havainnollistaa tällaisen vääristymän. Sekä vasemman-, että oikeanpuoleisessa kuvaajassa on käytetty samoja arvoja, Kuvio 5. Kak-





si pylväskuvaajaa joissa samat arvot

mutta oikeanpuoleinen kuvaaja antaa ymmärtää, että ensimmäisen pylvään arvo ovat moninkertaisesti paljon pienempi kuin viimeisen pylvään arvo, vaikka todellisuudessa arvot ovat hyvinkin lähellä toisiaan. Tällä tavalla vääristyneet kuvaajat voivat johtaa turhiin, huonoihin tai jopa vääriin päätöksentekoihin. (Laininen, 2004 & Nadhani, 2009)

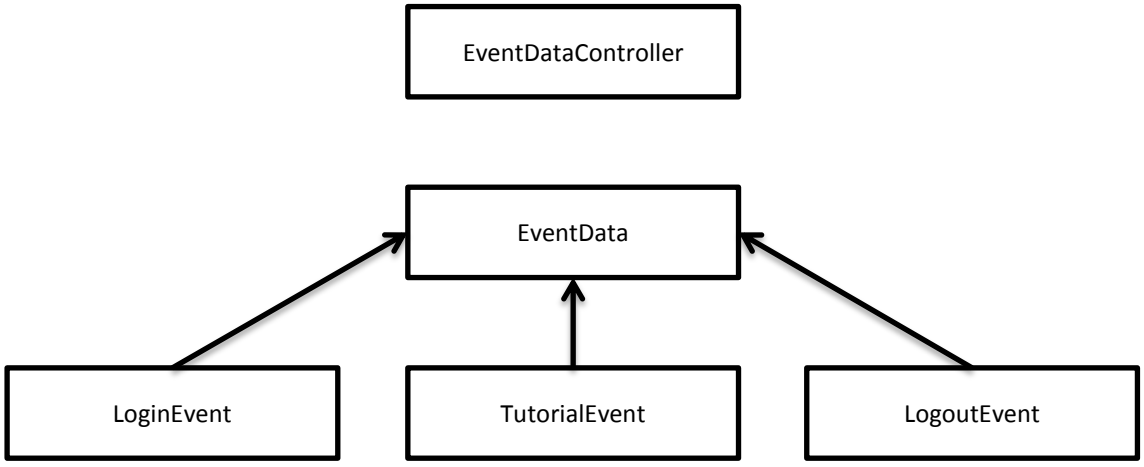
4 TELEMETRINEN TIEDONKERUUJÄRJESTELMÄ

Järjestelmä jonka suunnittelin ja toteutin, toimii Unity3D pelimoottorin päällä asiakassovelluksessa. Asiakaspään koodit on kirjoitettu C#-kielellä. Palvelinsovellus on kirjoitettu Java 6 -pohjaisella kielellä. Unity3D on komponentti- ja oliopohjainen pelimoottori. Tämä tarkoittaa sitä, että jokaisella peliobjektilla on yksi tai useampi komponentti, jotka ohjaavat kyseisen peliobjektin toimintaa komponenttiin ohjelmoidulla tavalla.

4.1 Asiakassovelluksen arkkitehtuuri

Asiakassovelluksen kokonaisuus käy ilmi Kuviosta 6. Tiedonhankintajärjestelmän ytimenä toimii EventDataController, joka hoitaa kerätyn tiedon tallentamisen ja lähettämisen. Jos Internet-yhteyttä ei ole saatavilla, tallentaa tämä luokka kaiken kerätyn tiedon kohdelaitteeseen. EventDataController on lisätty pysyvään peliobjektiin, jonka ainoana tehtävänä on olla olemassa, jotta pysyvät komponentit, kuten juuri tämä voivat toimia.

EventData -luokka on kerättävän tiedon perusluokka, joka automaattisesti tallentaa kaikille kerätyille tapahtumille yhteiset tiedot, kuten tapahtuman aikaleiman, sekä tapahtuman tyypin. Tästä luokasta voidaan periä kaikki yksilölliset tapahtumat, kuten esimerkiksi vaikka LoginEvent, TutorialEvent ja LogoutEvent.



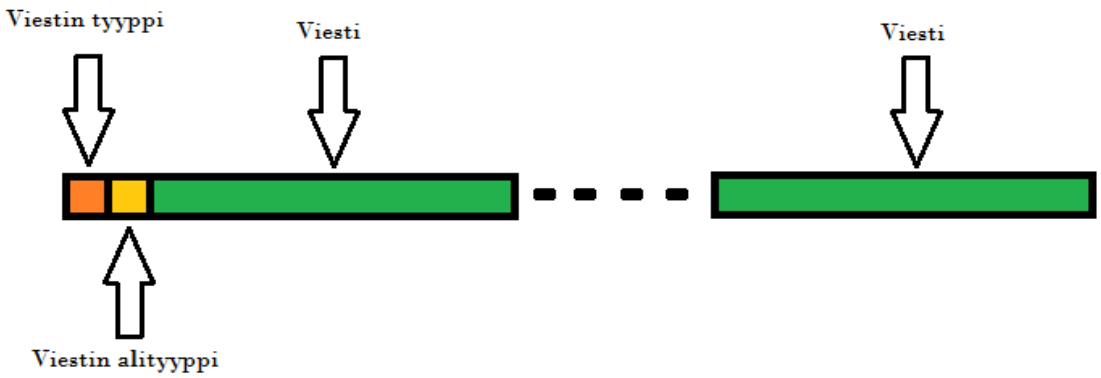
Kuvio 6. Asiakassovelluksen luokat ja niiden periytymiset.

4.2 Palvelinsovelluksen arkkitehtuuri

Palvelimella oleva sovellus on kirjoitettu Javalla, koska palvelinalustana toimii linux – pohjainen käyttöjärjestelmä, sekä Java -kielen samankaltaisuus C# -kielen kanssa. Javan virtuaalikone toimii alustariippumattomasti, ja näin ollen oli järkevä valinta. Palvelin on yhteydessä HSQLDB tietokantaan jatkuvasti, jolloin tarvittavat tietokantaoperaatiot voidaan suorittaa nopeasti.

ConnectThread-luokka huolehtii saapuvien yhteyksien kommunikoinnista. Yhteyden auettua ConnectThread-luokka lukee datavirran alusta ensimmäisen tavun, joka kertoo kuinka seuraavia tavuja tulee tulkita, toisin sanoen ensimmäinen tavu kertoo viestin tyypin. Tämän jälkeen kyseinen luokka valitsee oikean käsittelijän kyseiselle viestityypille.

Viestin ollessa ”Event” tyyppinen, se antaa datavirran EventDataHandler-luokalle luettavaksi, joka sitten lukee virrasta seuraavan tavun. Tämä tavu ilmaisee viestin alityypin, eli minkälainen tapahtumaviesti virrasta voidaan lukea. Kuviossa 7 on visuaalinen esitys tämäntyyppisestä viestistä.



Kuvio 7. Datavirran rakenne

Sitten kun viestin tyyppi ja alityyppi tiedetään, voidaan virta ohjata kyseisiä viestejä lukeville funktioille. Kun virrasta on luettu alityypin mukaisesti oikea määrä tavuja, tutkitaan onko virrassa vielä tavuja jäljellä. Jos virrasta löytyy vielä lukemattomia tavuja, luetaan taas yksi tavu, josta saadaan selville virrassa jäljellä olevan viestin alityyppi, jonka jälkeen taas voidaan lukea oikea määrä tavuja ja tallentaa kyseenomainen viesti tietokantaan. Tätä prosessia jatketaan niin kauan, kuin virrassa on tavuja, joita ei ole luettu. Tämä suoritetaan aina yhteyskoh-

taisesti omassa säikeessä. Tämä tarkoittaa sitä, että vaikka datavirran lukemisessa tapahtuisi virhe, palvelinsovellus ei sulkeudu, vaan ainoastaan kyseenomainen säie sammuu.

4.3 Tietoliikenne

Järjestelmän avulla tietoa voidaan aiheuttaa siirtoliikennettä mahdollisesti hyvinkin paljon, joten siirrettävää tietoa on rajoitettu ja pakattu niin paljon kuin mahdollista. Yhden yksittäisen tapahtuman HTTP- viestin kuormassa kulkeva datamäärä on kuitenkin aina vähintään 10 tavua; ensimmäinen tavu kertoo tulevan viestin tyypin, toinen tavu kertoo viestin alityypin ja seuraavat kahdeksan tavua on EventData -luokan ilmaisemalle aikaleimalle. Viestissä olevan tyypin ja alityypin mukaan luetaan kyseenomaisen viestin mukaisesti oikeassa järjestyksessä kaikki viestin mukana kulkeutuneet parametrit.

Verkkoliikenteeseen tulevien pakettien koko voi kuitenkin kasvaa eri syistä johtuen; seurattavasta tiedosta riippuen yhden paketin koko voi kasvaa hyvinkin suureksi, ja jos laite on ollut Internetin ulottumattomissa kun tapahtumia on tallennettu, lähetetään ne seuraavalla kerralla kun laite saa Internetiin yhteyden.

4.4 Kerätystä datasta saatavat perustiedot

Rust0 Games halusi, että järjestelmän avulla voitaisiin mitata muun datan lisäksi myös käytön laajuutta mittaavia tärkeitä tunnuslukuja:

- DAU
    - Daily Active Users
    - Ilmaisee kuinka monta yksittäistä käyttäjää kunakin päivänä on peliä pelannut.
- WAU
    - Weekly Active Users

    - Ilmaisee saman kuin DAU, mutta kutakin viikkoa kohti.

- MAU
    - Monthly Active Users
    - Ilmaisee saman kuin DAU ja WAU, mutta kutakin kuukautta kohti.

- Retention
    - Retentio
    - Ilmaisee paluuta sovelluksen ääreen, kuinka suuri osa käyttäjistä palaa sovelluksen pariin n. päivänä.

- Running retention
    - Juokseva retentio
    - Ilmaisee n:nen päivän paluuta sidottuna nykyaikaan.

4.5 Perustietojen visualisointi

Suunnitellessani käyttöliittymää näiden tietojen visualisointia varten, päädyin käyttämään pääasiallisesti viivakuvaajia kahdesta syystä:

- Viivakuvaajia käytetään pääasiassa aikaan verrattaviin arvoihin.
- Viivakuvaajien piirtäminen on moninkerroin yksinkertaisempaa, kuin esimerkiksi sektorikuvaajien.

Kuviosta 7 näkee käyttöliittymän eri valintatyökalut kuvaajien piirtämiseen. Alasvetovalikosta voidaan valita piirrettävän kuvaajan tyyppi. Riippuen kuvaajan tyypistä seuraavat kaksi päivämäärän valinta ruutua asettavat päivämäärä-alueen. Kaikissa muissa kuvaajatyypeissä paitsi ”Retention” niillä asetetaan kuvaajaan aikajana vaaka-akselille. Jäljelle jäävässä ”Retention” kuvaajatyypissä niillä asetetaan ensimmäisen kerran pelanneiden aika-alue. Valintalaatikko päivämäärävalitsimien oikealla puolella, täsmentää tarvittaessa kuvaajan luomisessa käytettä-

vää hakua, kuvaajan tyypistä riippuen. Tätä tarvitsee käyttää toistaiseksi vain ”Retention” ja ”Running retention” kuvaajatyypeissä. Näissä paluuta visualisoivissa kuvaajissa edellämainitulla valintalaatikolla määritetään n. päivä.

Kuvio 8. Kuvaajien luomiseen käytetty käyttöliittymä

Seuraavat kuviot ovat tekemäni käyttöliittymän muodostamia kuvia Rust0 Games:n Rival Rumble -pelistä, joka on kirjoitushetkellä julkaistu Google Play palvelussa sekä Kongregate portaalissa.

Liitteen 1 ensimmäinen kuvio ilmaisee pelin ”DAU” -arvoja lokakuussa. Kuvaajasta voidaan tehdä esimerksi johtopäätös, että peliä pelataan yleensä enemmän viikonloppuisin kuin arkena.

Liitteen 1 toinen kuvio ilmaisee lähes samaa kuin Liitteen 1 ensimmäinenkin kuvio. Siinä missä aiemmassa kuviossa nähdään yksittäisten käyttäjien määrä päivittäin, tässä kuviossa nähdään yksittäisten käyttäjien määrä viikottaisella tasolla. Liitteen 1 kolmannessa kuviossa ilmaistaan yksittäisten käyttäjien määrä kuukausittaisella tasolla.

Liitteen 1 neljäs kuvio ilmaisee paluuprosentin seitsemälle ensimmäiselle päivälle, lokakuussa aloittaneista pelaajista. Kuviosta voidaan päätellä, että ainoastaan noin kolmannes pelaajista palaa pelin pariin seuraavana päivänä.

Liitteen 1 viides kuvio ilmaisee toisen päivän paluuprosenttia kunakin päivänä. Kuviosta voidaan päätellä, että lauantaina ensimmäisen kerran pelanneet pelaajat, palaavat pelin ääreen eniten seuraavana päivänä.

5 POHDINTA

Opinnäytetyö kokonaisuutena onnistui hyvin, koska sain tehtyä Rust0 Games:lle tuotteen, jota se pystyy nyt käyttämään sekä nykyisten, että tulevien pelien iteratiivisessa kehittämisessä. Vaikka työ tulikin tehtyä sovellettua vesiputousmallilla noudattaen, tuli lopputuotteesta silti toimiva ja suhteellisen helppokäyttöinen. Ohjelmiston rungon tein ensin ilman varsinaista suunnitelmaa, koska Rust0 Games:llä oli kiire tällaisen ohjelmiston käyttöön. Jälkikäteen toiveiden mukaan täydensin ja korjailin ohjelmiston rakennetta, toiminnallisuutta ja ulkonäköä, sen mukaan miten se oli enää mahdollista.

Aikataulu oli alusta pitäen hieman hämärä, minkä vuoksi ajankäyttö venyi hyvinkin pitkäksi. Paremmalla suunnittelulla ja ajankäytöllä tästä työstä olisikin saanut enemmän irti.

Opinnäytetyössä kuvaajien piirtotyökalu jäi valitettavasti pienemmälle huomiolle, koska tiedon siirron ja tallennuksen toteutuksessa meni odotettua enemmän aikaa. Kuitenkin suhteellisen lyhyellä ajankäytöllä, sain toteutettua peruskäytön kuvaajien muodostamisen.

Työn tekeminen oli suhteellisen hankalaa, koska metriikan keräämisestä on hyvin vähän julkista tietoa. Luultavasti tällaiset järjestelmät ja niiden dokumentaatiot ovat yrityksille ikäänkuin markkinaetu, koska tiedot ja taidot pidetään yrityksen sisällä. Tästä syystä myöskään lähteitä ei ollut kovinkaan paljoa.

Näitä järjestelmiä ja teorioita tulisi useammankin alalla toimivan henkilön opiskella, koska nykyään lähes kaikki ohjelmistot ja sovellukset keräävät käyttäjistään tietoa. Olipa se tieto sitten hyvin yleismalkaista tai äärimmäisen yksityiskohtaista; tilastoja kuitenkin kerätään. Näillä kerätyillä tiedoilla ohjelmistojenkehittäjät voivat antaa meille entistä parempia käyttäjäkokemuksia, ja peleissä tasapainoisemman, sekä monipuolisemman pelikokemuksen. Miksi siis emme keräisi ja hyödyntäisi kerättyä tietoa?

LÄHTEET

Schreiber I, 2010 Game Balance Concepts – A continued experiment in game design and teaching
Saatavilla:
http://gamebalanceconcepts.wordpress.com/2010/08/25/level-8-metrics-and-statistics/
Luettu 11.8.2012

Chopra P, 2010, The Ultimate Guide To A/B Testing
Saatavilla:
http://www.smashingmagazine.com/2010/06/24/the-ultimate-guide-to-a-b-testing/
Luettu 13.8.2012

McNamara, 2012, Overview of Basic Methods to Collect Information
Saatavilla:
http://managementhelp.org/businessresearch/methods.htm
Luettu 15.8.2012

Wikipedia, 2012, Telemetry
Saatavilla:
http://en.wikipedia.org/wiki/Telemetry
Luettu 10.8.2012

HSQLDB, 2012, HyperSQL 100% Java database
Saatavilla:
http://hsqldb.org/
Luettu 10.8.2012

Whorton K, 2009, Qualitative Interviews: The Pros and Cons Marketing Insights
Saatavilla:
http://www.asaecenter.org/Resources/ENewsletterMarketingInsights.cfm?ItemNumber=38637
Luettu 18.12.2012

Teach-ICT a, 2012, Questionnaires - pros and cons
Saatavilla:
http://www.teach-ict.com/as_a2_ict_new/ocr/A2_G063/331_systems_cycle/investigation_methods/miniweb/pg4.htm
Luettu 18.12.2012

Teach-ICT b, 2012, Observations
Saatavilla:
http://www.teach-ict.com/as_a2_ict_new/ocr/A2_G063/331_systems_cycle/investigation_methods/miniweb/pg10.htm
Luettu 18.12.2012

Turn Up the VOLUME: the Students Speak Toolkit, 2012, What are the pros and const of focus groups
Saatavilla:
http://www.robertsandkay.com/tutv/iii_b_2.html
Luettu 18.12.2012

Soy S, 1996, The Case Study as a Research Method
Saatavilla:
http://www.gslis.utexas.edu/~ssoy/usesusers/l391d1b.htm
Luettu 18.12.2012

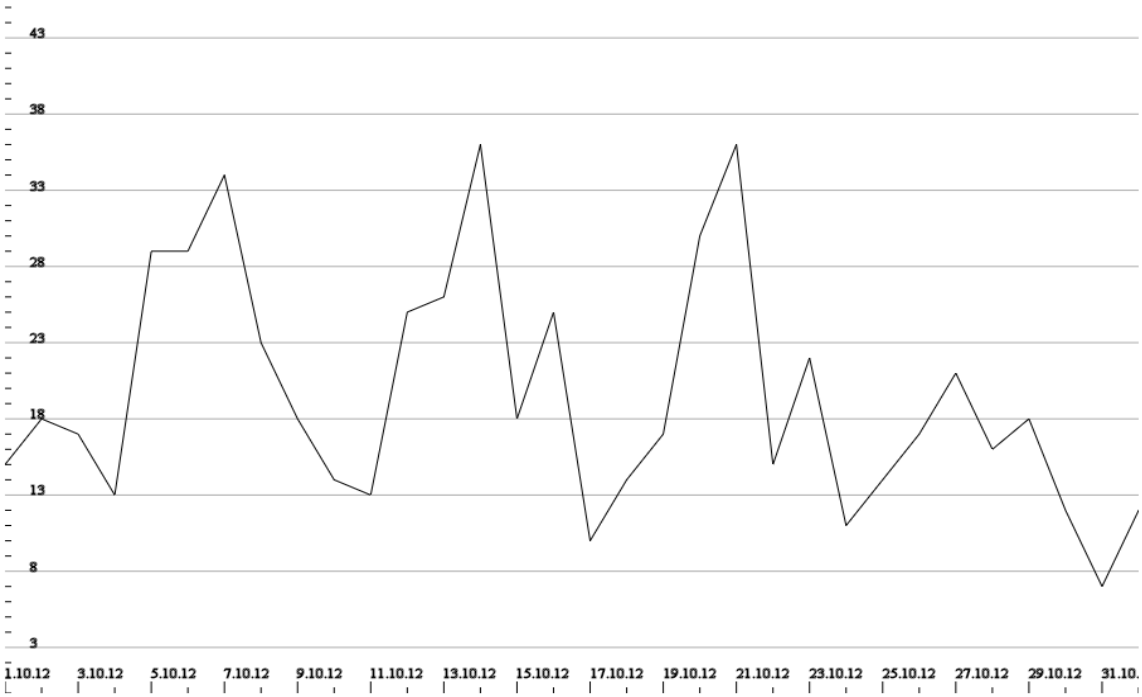
Nadhani S, 2009, Selecting the Right Chart Type for your Data
Saatavilla:
http://www.tutorial9.net/tutorials/web-tutorials/selecting-the-right-chart-type-for-your-data/
Luettu 18.12.2012

Medler B, 2012, Game Data Piracy: How Players Abduct Data in Order to Transform their Gameplay Experiences
Saatavilla:
http://meaningfulplay.msu.edu/proceedings2012/abstract.php?paperid=152
Luettu 4.1.2013

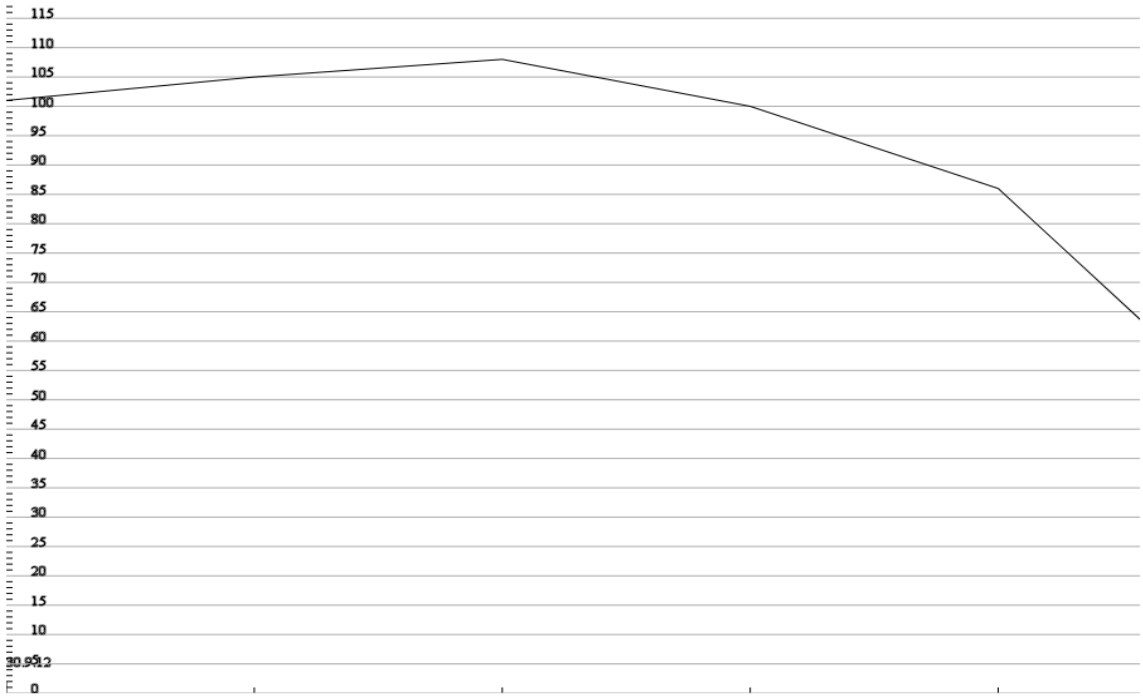
Bauckhage C, Kersting K, Sifa R, Thurau C, Drachen A & Canossa A, 2012, How Players Lose Interest in Playing a Game: An Empirical Study Based on Distributions of Total Playing Times

Laininen P, 2004, Tilastollisen analyysin perusteet, Otatieto, 2004

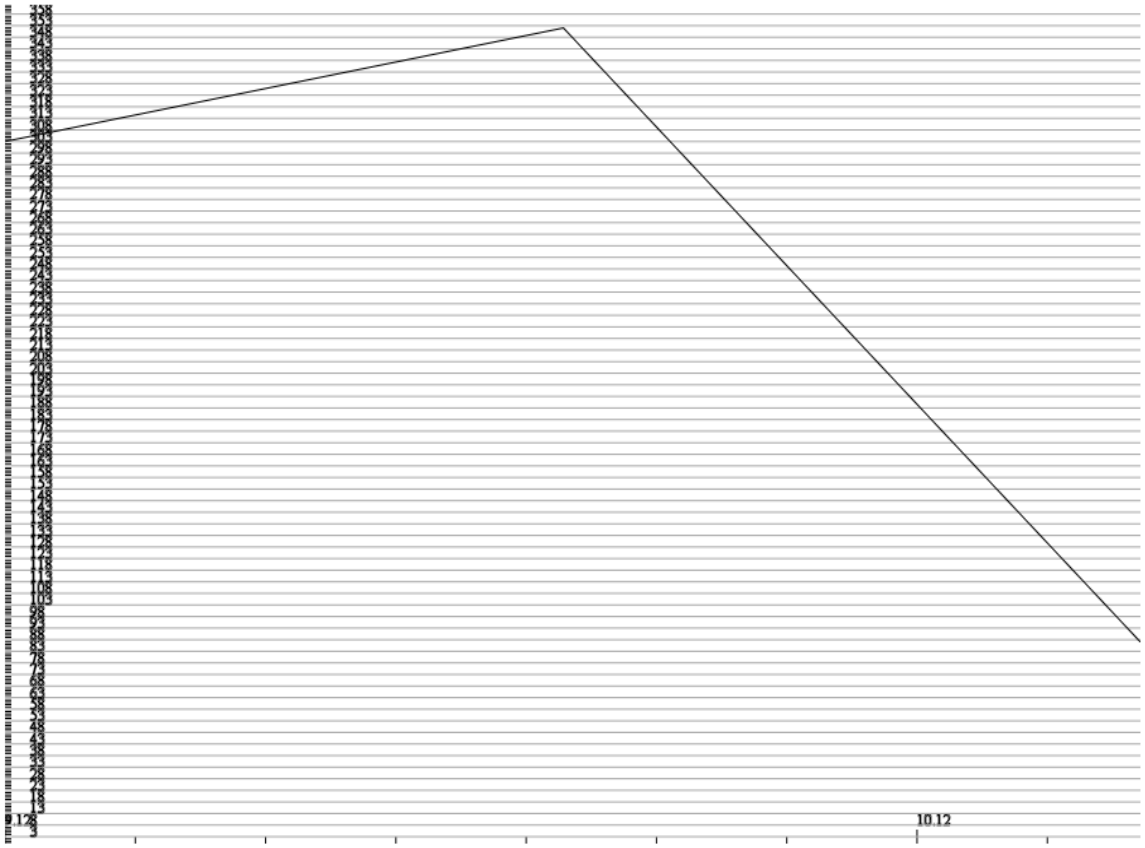
Valmiin ohjelmat tuottamat kuvaajat



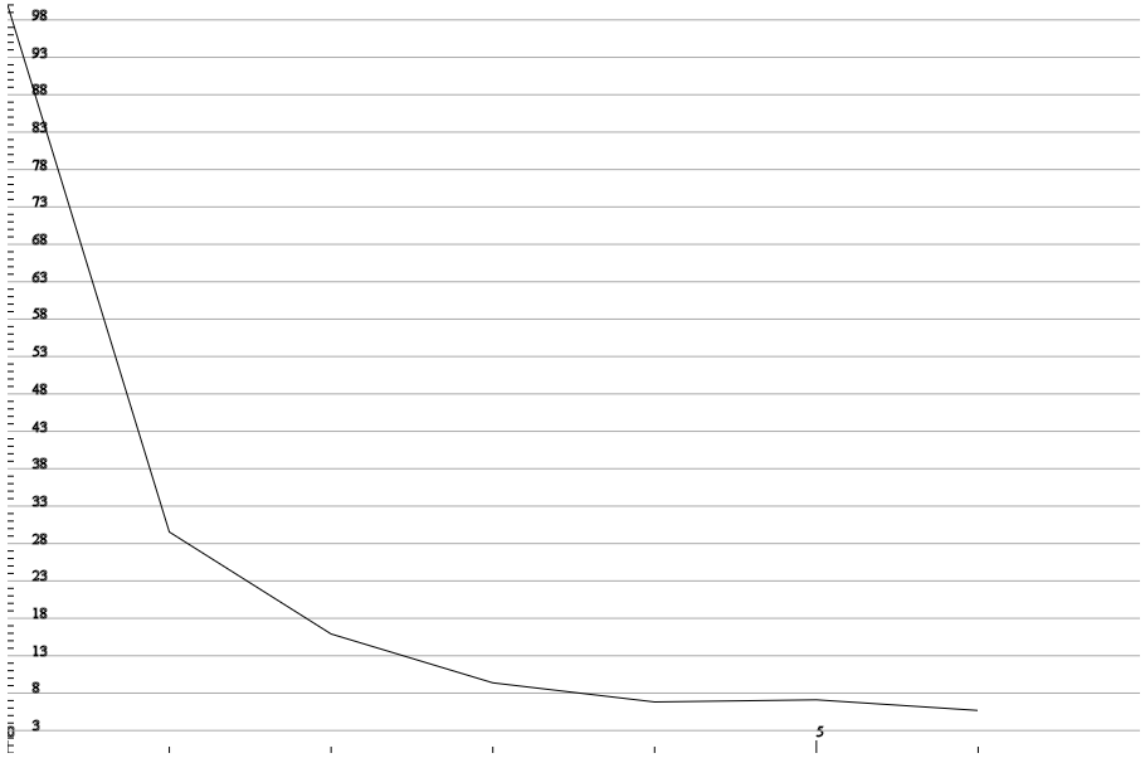
Kuvaaja pelin ”DAU” –arvoista



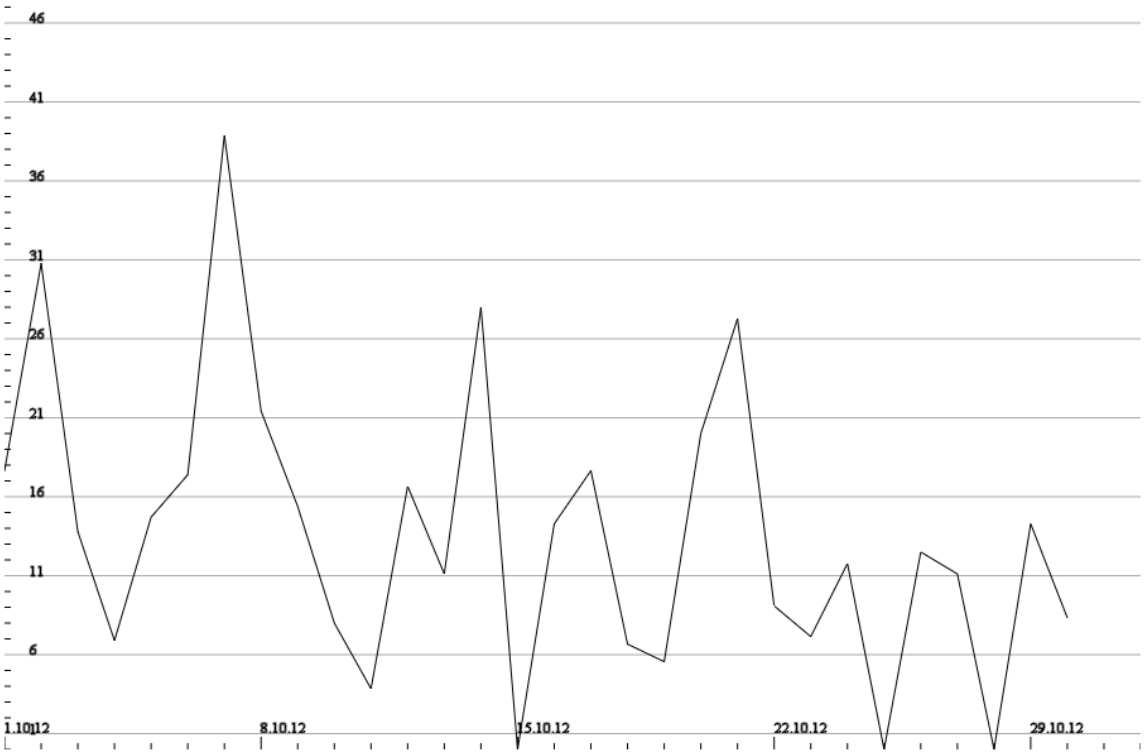
Kuvaaja pelin ”WAU” –arvoista



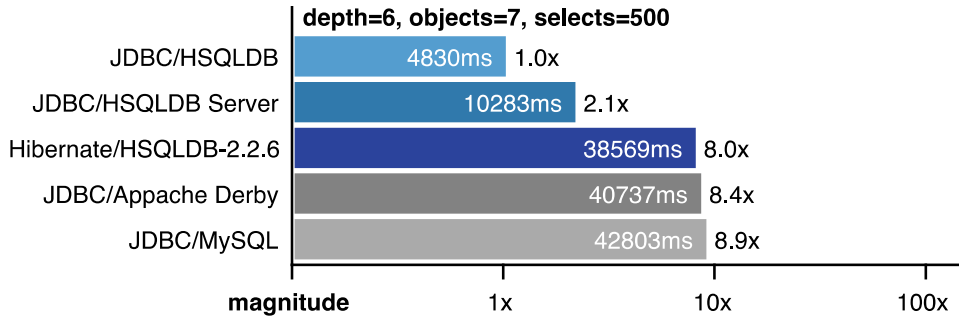
Kuvaaja pelin ”MAU” –arvoista



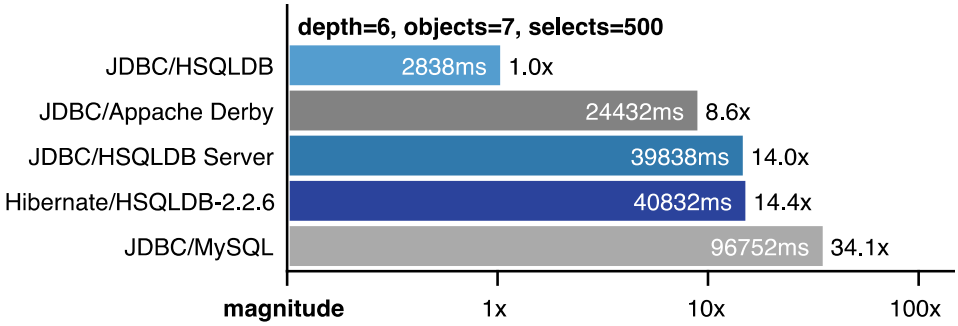
Seitsemän päivän retentio



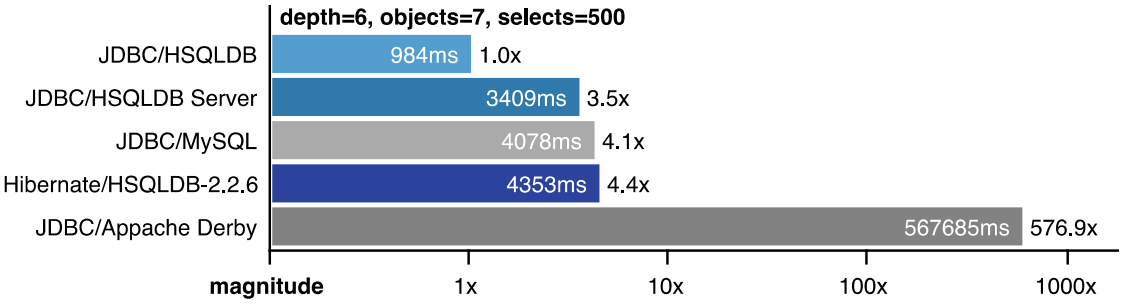
Toisen päivän juokseva retentio



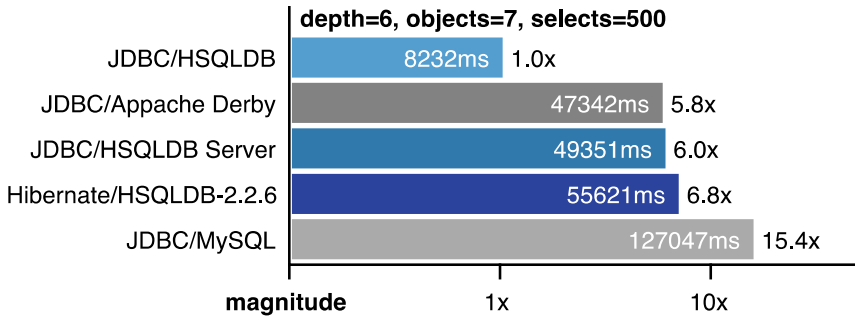
HSQLDB kirjoitusnopeuden vertailu (write)



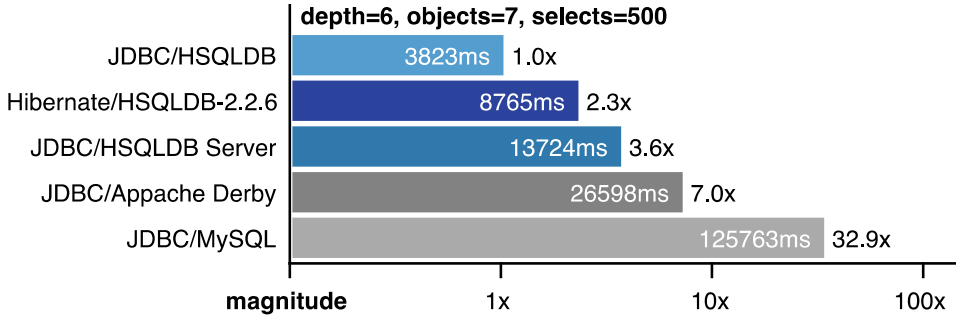
HSQLDB lukunopeuden vertailu (read)



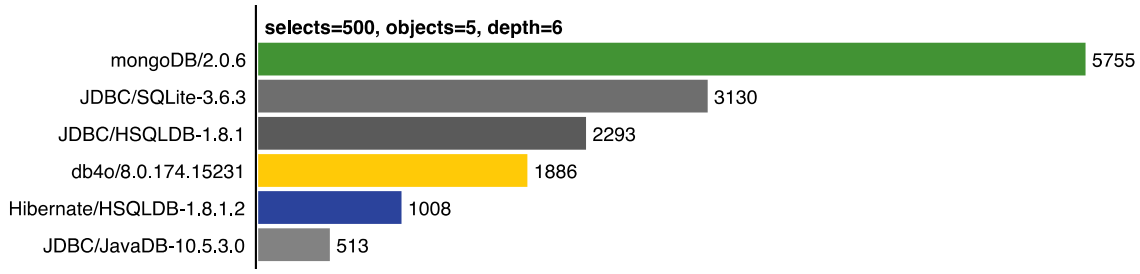
HSQLDB kyselynopeuden vertailu (query)



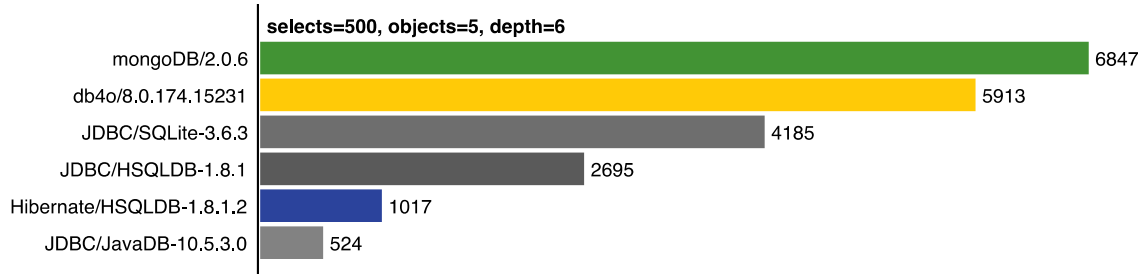
HSQLDB päivitysnopeuden vertailu (update)



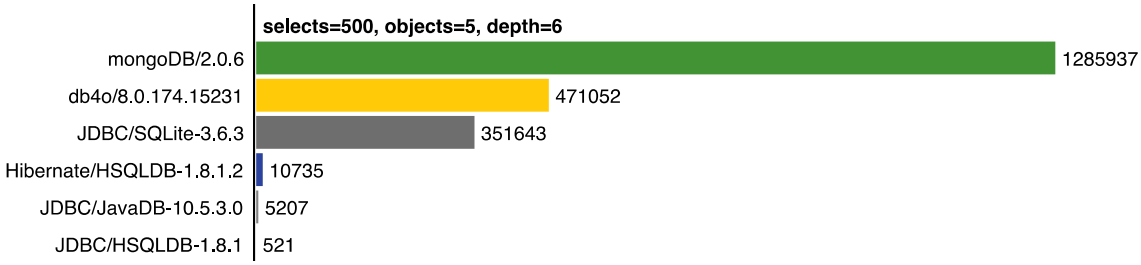
HSQLDB poistonopeuden vertailu (delete)



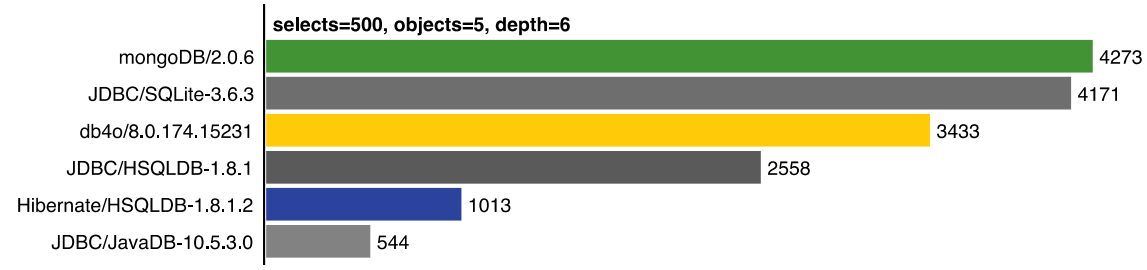
HSQLDB kirjoitusnopeuden vertailu (write)



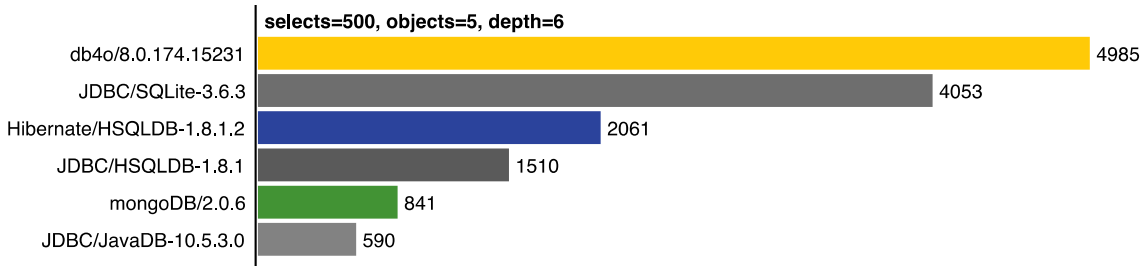
HSQLDB lukunopeuden vertailu (read)



HSQLDB kyselynopeuden vertailu (query)



HSQLDB päivitysnopeuden vertailu (update)



HSQLDB poistonopeuden vertailu (delete)