jamk

Hajautetun verkon optimointi

Juha-Pekka Tiirikainen

Opinnäytetyö
Toukokuu 2022
Tietojenkäsittely ja tietoliikenne
Insinööri (AMK), tieto- ja viestintätekniikka

jamk | Jyväskylän ammattikorkeakoulu
University of Applied Sciences

jamk

Kuvailulehti

Tiirikainen, Juha-Pekka

Hajautetun verkon optimointi

Jyväskylä: Jyväskylän ammattikorkeakoulu. Toukokuu, 2022, 36 sivua

Tekniikan ala. Tieto- ja viestintätekniikan tutkinto-ohjelma. Opinnäytetyö AMK.

Julkaisun kieli: suomi

Julkaisulupa avoimessa verkossa: kyllä

Tiivistelmä

Tiedonvaihto on suuressa roolissa nykyajan sovellusten toimintaa. Jatkuvasti kasvavien datamäärien myötä tiedonsiirtojen osuus sovellusten kehittämisestä tulee kasvamaan entisestään.

Hajautetuilla verkoilla pyritään estämään verkkojen ongelmia ja pyrkiä tekemään verkoista vikasietoisempia. Hajautettujen verkoissa verkkojen vasteaikoja saadaan perinteisiä verkkoja pienemmiksi.

Tiedonsiirron rooli toimeksiantajan sovelluksessa on merkittävässä osassa ja on osa toimivaa kokonaisuutta. Tavoitteena oli löytää keinoja toiminnan tehostamiseksi ja pyrkiä omaksumaan toimeksiantajan kehittämän sovelluksen toiminnallisuutta osana tutkimustyötä.

Laadullisessa tutkimuksessa on hyödynnetty konstruktiivista tutkimustapaa, jonka tarkoituksena on löytää reaalimaailman ongelmiin ratkaisuja. Tutkimuksen lopputuloksena syntynyt konstruktio eroaa kaikesta jo olemassa olevasta. Kehittämisprosessin lopputuloksena löydetty kehitysidea oli työn aikana syntynyt konstruktio.

Avainsanat (asiasanat)

hajautettu verkko

Muut tiedot (salassa pidettävät liitteet)

jamk

Description

Tiirikainen, Juha-Pekka

Optimalization of decentralized network

Jyväskylä: JAMK University of Applied Sciences, May 2022, 36 pages

Bachelor's Degree Programme in Information and Communications Technology

Permission for open access publication: Yes

Language of publication: Finnish

Abstract

Data exchange is big part of modern day software engineering. Continous grow of data makes data exchange even bigger part of future software development.

With decentralized networks try to prevent problems in networks and aim to make the network more fault tolerant. With decentralized networks allows smaller response times than other networks.

Data exchange in the principals software is one of the main part and part of the working system. The goal was to find ways to optimize the performance of the network and to get use to the software in general as part of the study.

In the qualitative research utilized a constructive research approach aimed to solve realworld problems. As the result of the study, there was born a construction that is different from anything from the field has to offer. The end result of the development process founded idea of optimatization was the construction of this work.

Keywords/tags (subjects)

decentralized network

Miscellaneous (Confidential information)

Sisältö

Kuviot

1 Johdanto

Tiedonvaihto on prosessi, jossa dataa siirretään tietokoneen solmusta toiseen käyttämällä laskennallisia tekniikoita ja teknologioita. Data siirretään bittien ja tavujen muodossa digitaalisen tai analogisen välineen kautta. Tämä prosessi mahdollistaa digitaalisen tai analogisen viestinnän tai tiedon liikkumisen laitteiden välillä. Tiedonsiirto käyttää hyväkseen monia erilaisia tietoliikennemuotoja, joilla data siirretään verkon nodesta toiseen. Siirretty data voi olla mitä tahansa tyyppiä, kokoa tai laatua. Analogisessa tiedonsiirrossa data siirretään analogisina signaaleina, kun taas digitaalisessa datasiirrossa data muunnetaan digitaalisiksi bittivirroiksi. Tiedonsiirto etäpalvelimelta paikalliselle koneelle on esimerkki digitaalisesta tiedonsiirrosta. (Data transfer, 2013)

Statista (Statista Research Apartment, 2022) on tehnyt arvion että vuonna 2022 luodaan, tallennetaan, kopioidaan ja käytetään maailmanlaajuisesti 97 tsettatavua dataa ja tietoa. Statistan toisen arvion mukaan luodun datan ja tiedon määrä tulee kasvamaan 180 tsettatavuun vuodessa vuonna 2025 ja, että vain kaksi prosenttia edellisenä vuonna luodusta ja käytetystä datasta tallennetaan ja säilytetään seuraavalle vuodelle (Statista Research Apartment, 2022).

Työn toimeksiantajana toimii yritys X, jossa kirjoittaja työskentelee. Yritys X on oman alansa johtavia tekijöitä alallaan. Yrityksen osaamisalaa ovat graafiset tietokannat sekä niiden hyödyntämiseen liittyvien ohjelmistojen kehitys. Kehitettävien ohjelmistojen pääpiirteenä voidaan pitää sitä, että ne kaikki käyttävät toimintansa yhteydessä graafisia tietokantoja eri tavoin tarpeen mukaan. Esimerkkeinä ominaisuuksista, joita sovelluskehittämisessä käytetään hyväksi voidaan mainita semanttiset haut ja kontekstuaaliset palvelut. Yritys on kotimainen yksityinen ohjelmistoyritys, joka toimii kansainvälisillä markkinoilla. Työn kohteena on sovellus, jota on kehitetty jo useita vuosia.

Työn tavoitteena oli päästä toimeksiantajan kehittämän sovelluksen toimintaan ja toiminnallisuuksiin sisään, koska sovelluksen kehitys ja korjaaminen tulevat olemaan osa kirjoittajan tulevaisuuden työtehtäviä. Toiminnan ymmärtämisen ja muun tiedonkeruun jälkeen pyrittiin etsimään kehityskohteita sovelluksen toimintaan, niin pitkällä kuin lyhyelläkin aikavälillä.

Työn pienempänä tavoitteena oli päästä paremmin kiinni toimeksiantajan prosesseihin. Pääpaino niissä oli keskittyä tulevien tehtäviin liittyviin prosesseihin eli mitä tulee virheiden etsimiseen, niiden paikantamiseen sovelluksen koodissa ja sovelluksen kehittämiseen sekä virheiden korjaamiseen.

Opinnäytetyön aiheena olevan sovelluksen osan kehittäminen on osa toimivaa kokonaisuutta, mutta tarkoituksena on etsiä mahdollisia kehityskeinoja toiminnan tehostamiseksi. Työ on rajattu sovelluksen tiedonvaihdon toiminnan parantamiseen, ja tarkemmin sovelluksessa tapahtuvan tiedonvaihdon viestien tiedonkeräykseen, viestien lähettämiseen sekä viestien vastaanottamiseen. Rajaus on toteutettu näin, koska ne ovat tiedonvaihdon kolme kriittisintä toimintoa. Työ on tärkeä toimeksiantajalle, jotta sovelluksen nykyistä toimintaa voitaisiin parantaa. Tiedonvaihto on tärkeässä roolissa sovelluksen toiminnassa ja sen toimivuuden parantaminen toisi tehokkuutta sovelluksen toimintaan.

2 Tutkimusmetologia

2.1 Tutkimuskysymys

Kun tutkimuksen aihe on saatu päätettyä, on seuraava vaihe tutkimusongelman muodostaminen. Tutkimusongelma on osa jokaista tutkimusta ja siihen pyritään vastaamaan tutkimusmetodologioita sekä materiaaleja käyttäen. Tutkimuskysymyksen luominen tutkimusongelman pohjalta helpottaa työtä, koska vastaaminen kysymykseen on helpompaa kuin ongelman ratkaiseminen. Tutkijaa ja tutkimuksen etenemistä ohjaavat tutkimusongelma sekä tutkimuskysymys. Tutkimuskysymyksiä voi olla yksi tai enemmän, mutta oikeiden kysymysten avulla ongelmaan tullaan saamaan ratkaisu. (Kananen, J, 46-48)

Laadullisen tutkimuksen tutkijakysymyksissä korostuvat muun muassa kohteen esiintymisympäristöön ja taustaa, kohteen tarkoitukseen ja merkitykseen, ilmaisuun ja kieleen liittyvät näkökulmat. (Laadullinen tutkimus, 2021) Laadullisen tutkimuksen hyvä puoli on se, että se on luonteeltaan syvällistä. Tämä auttaa aiheen sekä ongelman yksityiskohtien ymmärtämisessä. Laadullisen tutkimuksen avulla voidaan muotoilla hypoteesi ja siten kerätä tiedot, joiden avulla voidaan paljastaa hypoteesin pätevyys tai virheellisyys.

Opinnäytetyössä vastataan seuraavaan päätutkimuskysymykseen:

> Kuinka saadaan optimoitua hajautetun verkon toimintaa?

Tämä tutkimuskysymys jakaantuu täsmentäviin alikysymyksiin:

> Kuinka saadaan optimoitua hajautetussa verkossa tapahtuvien tiedonsiirtojen tiedonkeräystä?
>
> Kuinka saadaan optimoitua hajautetussa verkossa tapahtuvien tiedonsiirtojen lähetystä?
>
> Kuinka saadaan optimoitua hajautetussa verkossa tapahtuvien tiedonsiirtojen vastaanottamista?

2.2 Esiolettamukset työn lopputuloksesta

Esiolettamuksena työn lopputuloksesta on selvitys, jossa kuvataan sovelluksen hajautetun verkon toiminnan jatkokehityksen ideoita lyhyellä ja pitkällä aikavälillä. Selvityksestä käy ilmi kehitysideoiden toiminnan tehostamiseen tarvittavan kehityksen määrä sekä kehitysidean muutoksen aikaan saama hyöty sovelluksen toiminnassa.

2.3 Tutkimusmenetelmä

Tämän tutkimuksen päätavoitteena on omaksua toimeksiantajan sovelluksen hajautetun verkon toiminta ja toiminnallisuus sekä löytää tiedon kerryttämisen jälkeen keinoja sen toiminnan tehostamiseksi. Kananen (2017) kirjoittaa, että kvantitatiiviseen tutkimukseen liittyvä yleistysten teko ei kuulu kvalitatiiviseen tutkimukseen ja että laadullisella tutkimuksella on tarkoitus saavuttaa ymmärrys tutkimuksen kohteena olevaan ilmiöön. Suurin kysymys, johon haetaan vastausta kuuluukin, että mistä ilmiössä on kyse. (32.) Tämän tutkimuksen ilmiönä käsitellään on tiedonsiirtoa sovellusten välillä, pyritään ymmärtämään mistä ilmiössä on kyse ja kehittelemään kehitysideoita sen tehostamiseksi.

Tutkimuksessa voisi käyttää hyväksi konstruktiivista tutkimusotetta. Konstruktiivisessa tutkimusotteessa metodologia, joka tuottaa innovatiivisia konstruktioita, joiden avulla yritetään ratkaista reaalimaailman ongelmia. Konstruktiivisen tutkimusotteen on tarkoitus tuottaa kontribuutiota sitä soveltavalle tieteenalalle. Konstruktio on konstruktiivisen tutkimusotteen ydinkäsite. Konstruktio on käsite, joka on abstrakti ja jolla voi olla toteutumia suuri, itseasiassa loputon määrä. Konstruktioita ovat kaikki ihmisen luomat artefaktit. Kaikille erilaisille konstruktioille tunnusomaista se että ne keksitään ja kehitetään, ei löydetä. Kaikesta jo olemassa olevasta poikkeavan konstruktion kehittäminen luo jotain uutta: uusi todellisuus kehittyy uudenlaisista konstruktioista.

Konstruktiivisessa tutkimusotteessa (Lukka n.d.)

- Keskitytään ratkaisemaan tarpeellisiksi koettuja tosielämän ongelmia käytännössä.
- On tarkoituksena ratkaista alkuperäinen tosielämän ongelma tuottamalla innovatiivinen konstruktio.
- Odotetaan tutkijan saavuttavan kokemuksellista oppimista, jota on tarkoitus saada kun tehdään tutkijan sekä käytännön edustajien kesken yhteistyötä.
- Kytketään tutkimusote olemassa olevaan teoreettiseen tietämykseen huolellisesti.

- Pyritään kiinnittämään teoriaan empiiriset löydökset reflektoimalla niitä.

Tämän laadullisen tutkimuksen kehittämisprosessin tutkimusotteeksi sopii parhaiten viimeisenä mainittu konstruktiivinen tutkimusote. Konstruktiivisen tutkimuksen lopputuloksena pyritään löytämään ratkaisu, joka on hyvä ja jota voidaan suositella. Tämä antaa ratkaisulle, johon konstruktiivisella tutkimusotteella on päädytty, normatiivisen piirteen. Ratkaisulla on tarkoituksena saavuttaa parantunut tilanne alkuperäiseen verrattuna. Tämän tutkimuksen tavoitteena oli saada tietoa toimeksiantajan sovelluksen hajautetun verkon toiminnasta ja pyrkiä tiedon ja toiminnallisuuden ymmärtämisen jälkeen löytää keinoja, joilla nykyistä toimintaa pystyisi parantamaan. Työssä luotu konstruktio on luotu teorian pohjalta, jota käytetään myös havaintojen jäsentelyssä. Aiheen parissa työskenteleviltä henkilöiltä voidaan käydä keskustelua aiheesta. Vasta tiedonkeräyksen ja konsultoinnin jälkeen aloitetaan uuden kehittäminen. (Virtanen 2006, 51.)

Objektiivisuuden puute on konstruktiivisen tutkimusotteen tunnettu heikkous. Oman ongelmansa tekevä tutkija on itse tekemässä uutta konstruktiota, joka voi rajata näkökulmaa. Ongelmaa on koitettu ehkäistä konsultoimalla käytännön edustajia työyhteisössä. Työyhteisöllä on siis ollut osansa työn kehityksessä.

2.4 Tutkimusetiikka

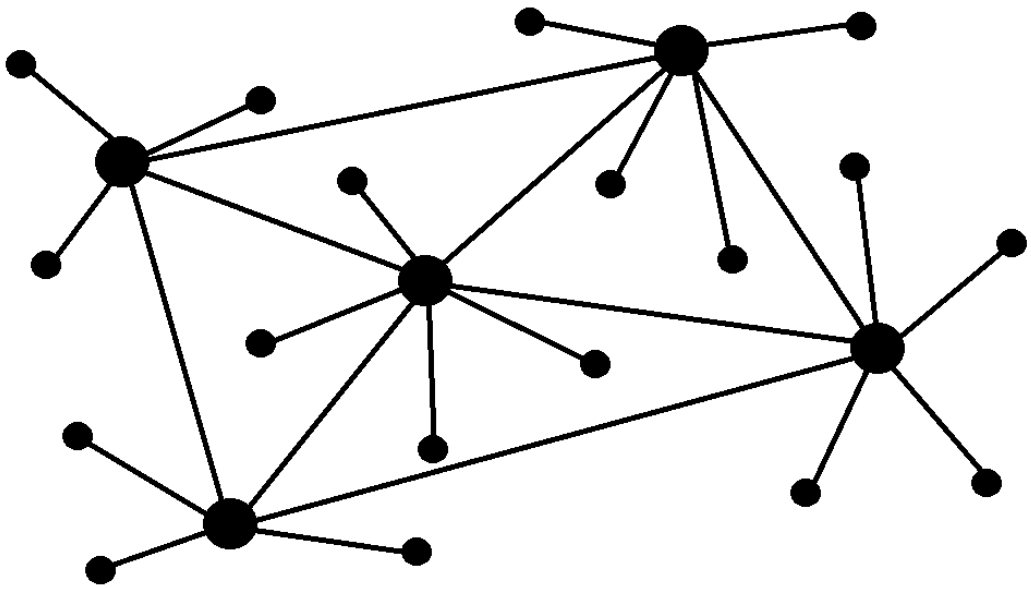
Tutkimuksessa on huomioitu hyvä tieteellinen käytäntö. Tähän tutkimukseen liittyvät keskeiset huomiot ovat seuraavat (Hyvä tieteellinen käytäntö, 2021):

- Tutkimuksen aikana noudatettiin rehellisyyttä, tarkkuutta sekä huolellisuutta kaikissa sen vaiheissa.
- Tieteellisen tutkimuksen mukaisten ja eettisesti kestävien tiedonhankinta-, tutkimus- ja arviointimenetelmiä sovelletaan tutkimukseen. Tieteellisen tiedon luonteeseen kuuluvan avoimuutta toteutetaan tutkimuksessa. Tuloksia julkaistaessa toteutetaan vastuullista tiedeviestintää.
- Muiden tutkijoiden työn ja saavutukset otetaan huomioon asianmukaisella tavalla kunnioittamalla ja viittaamalla niihin asianmukaisella tavalla. Oman tutkimuksensa ja sen tuloksien julkaistessa tutkijan tulee antaa muille heidän ansaitsemansa kuuluva arvo ja merkitys.
- Tarvittavat tutkimusluvat on hankittu.
- Tutkimusorganisaatiossa otettiin huomioon tietosuojaa koskevat kysymykset.

3 Hajautettu verkko

3.1 Mikä tarkoittaa hajautettu verkko?

Hajautettu verkko, tai hajautettu tietokanta, tarkoittaa verkkoa, jolla ei ole yksittäistä tiettyä serveriä, eli pääserveriä. Verkossa olevat koneet, eli nodet, jakavat resurssinsa sekä ohjelmiston keskenään verkossa olevien koneiden kanssa. Tällöin yhden koneen ei tarvitse huolehtia koko verkon toiminnasta, vaan esimerkiksi tietojen käsittelyä voidaan hajauttaa konekohtaisesti. (Ditributed databases, n.d). Hajautetun verkon rakennetta kuvattu kuviossa 1.



Kuvio 1. Hajautetun verkon rakenne

Kuviossa 1 isompi pallo kuvaa hajautetun verkon ensisijaisina palvelimina toimivia nodeja ja pienemmät niihin liittyneitä käyttäjiä.

Hajautettu verkko näyttäytyy käyttäjälle samanlaisena kuin perinteinen server-client-verkko, mutta periaatteessa se on joukko tietokantoja, jotka jakavat tietoa toisilleen tiedon muuttuessa ja pitävät toisensa tiedot ajan tasalla verkossa. Hajautetun verkon tietoa on mahdollista käyttää ja

muokata yhtäaikaisesti monesta eri pisteestä verkkoa. (Ditributed databases, n.d.). Verkon tapahtumien työmäärän jaolla saavutetaan monenlaisia hyötyjä, kuten parannettu luotettavuus, skaalautuvuus sekä yksityisyys (Centralized Networks vs De-centralized Networks, 2018).

3.1.1 Hajautetun verkon hyviä puolia

Hajautetun verkon hyviin puoliin lukeutuu vikasietoisuus, nopeus sekä skaalautuvuus. Hajautetut verkot ovat vikasietoisia, mutta niihin vaikuttavat osittain samat ongelmat ku in perinteisiin keskitettyihin verkoille. Verkon design auttaa parantamaan sen vikasietoisuutta, yhden tai useamman keskitetyn serverin kaatumisen jälkeen muut verkon serverit jatkavat tiedon tarjoamista verkossa. Toinen designin hyvistä puolista on, että yleensä tieto on käytettävissä nopeammin. Nopeutta voidaan parantaa tekemällä uusia nodeja paikkoihin, missä käyttäjien määrä voi olla suuri. (Touron, M., 2019).

Hajautettua verkkoa on mahdollista laajentaa tarpeen tullen helposti, jos verkkoon halutaan li sätä uusia käyttäjiä. Verkon helppo laajentaminen on etu perinteisiin client-server-verkkoihin verrattuna. Toinen etu perinteisiin client-server-verkkoihin on se, että client-server-verkon kasvaessa yleensä sen suorituskyky heikkenee. Hajautetuissa verkoissa tilanne on usein päinvastainen, uusien koneiden liittyessä verkkoon, verkon suorituskyky voi parantua. Suorituskyvyn parantuminen on sivuvaikutus siitä, että verkon jokainen kone voi toimia myös serverinä ja näin jakaa resursseja verkon käytettäväksi. Hajautettujen verkkojen yhdistäminen mahdollistaa yhteistyön erilaisten laitteiden välillä. Erilaisten laitteiden erilaiset resurssit voivat hyödyntää koko verkon toimintaa. (Indeed Editorial Team, 2021).

3.1.2 Hajautettujen verkkojen huonoja puolia tai haasteellisuuksia

Vaikka hajautetut verkot ovat hyvin vikasietoisempia kuin keskitetyt verkot, vaikuttavat niihin osittain samat ongelmat. Esimerkiksi serverien kaatuminen on edelleen osa hajautettujen verkkojen toimintaa, mutta hajautettujen verkkojen design mahdollistaa yhden tai useamman serverin kaatuessa tiedon tarjoamisen jatkamisen muiden verkossa olevien servereiden kautta käyttäj ille. Vaikka serverien kaatuminen ei riko verkkoa, niiden kaatuminen voi kuitenkin vaikuttaa verkon tehokkuuteen ja mahdollisesti rajoittaa pääsyä johonkin tietoon. (Touron, M., 2019).

3.2 Hajautetun verkon käyttötarkoituksia

Hajautettujen verkkojen käyttö on laajaa nykyaikaisessa bisnesympäristössä.

3.2.1 Kryptovaluutat

Kryptovaluutat käyttävät laajalti lohkoketjuja, joiden toiminta perustuu hajautettuihin verkkoihin. Kryptovaluuttojen taustalla toimiva teknologia hyväksikäyttää P2P-verkkojen tehoa ja mahdollistaa jaetun ja luotettavan tapahtumakirjan. Hajautettu tilikirjateknologia tallentaa tapahtumat muuttumattoman, aikaleimatun digitaalisen lohkon, joka osoittaa lähettäjiin ja vastaanottajiin. Mikään keskitetty taho ei käsittele lohkoketjujen verkkoja ja osallistujat voivat vahvistaa tapahtuman toistensa joukossa. Teknologia mahdollistaa ihmisten sekä instituutioiden luottamuksen lopputulemaan ilman luottamisen tarvetta osallistujiin. Tämä, uudenlainen hajautetun datan muoto ja käsittely toimii digitaalisena pääkirjana, jonne kirjataan kaikki tapahtumat sekä aktiviteetit julkisesti. (p2p in blockchain, 2021)

3.2.2 Suoratoistopalvelut

Hajautettuja verkkoja käytetään myös paljon suoratoistopalveluissa. Suoratoistopalveluiden luovat hajautetun verkon samaa sisältöä katselevien käyttäjien kanssa. Tämä mahdollistaa sen, että käyttäjät voivat jakaa katseltavan sisällön osia keskenään ilman että jokaisen erillisen katsojan tarvitsee hakea niitä erikseen serveriltä uudelleen ja uudelleen. Miksi ladata tiedostot, jonka joku toinen käyttäjä on ladannut maan toiselta puolelta, uudelleen erikseen itsellesi, kun vaihtoehtona on hakea samat tiedostot lähempää ja nopeammin. (Akamai's Peer-To-Peer Love Story, 2017)

3.2.3 Hakukoneet

Kuten YaCy (YaCy is free software for your own search engine) sivuillaan kysyy, mitä jos isojen ammattihakukoneoperaattorin omaan ohjelmiston luottamisen sijaan hakukonettasi pyörittäisi monet yksityiset tietokoneet, jotka eivät ole minkään yrityksen tai henkilön hallinnassa . Kuten hajautetuissa verkoissa, YaCy-hakukoneohjelma, joka tarjoaa tuloksia riippumattomien vertaisten koneiden verkosta keskusserverin sijaan (What is this?, n.d).

3.3 Data arkkitehtuuri

Data-arkkitehtuuri on tärkeä osa datan hallintaa. Hyvin suunnitellulla data-arkkitehtuurilla voidaan ennaltaehkäistä datan rikkoutumista. Hyvin suunniteltu data-arkkitehtuuri myös helpottaa dataa käyttävien sovellusten integrointia sekä helpottaa datan käsittelyä. Hyvin suunniteltu data-arkkitehtuuristrategia voi olla valttikortti kilpailijoihin verrattuna, sillä voidaan saada operaationallista tehokkuutta parannettua ja mahdollistaa parempaa päätöksen tekoa. Yrityksillä, joiden datan käsittely on hyvin hoidettu, voivat tulla ketterämmäksi ja voivat muuttua trendien mukaan sekä hyödyntää uusia tilaisuuksia bisneksen muuttuessa nopeammin. Hyvä datan käsittely voi myös ennaltaehkäistä tietovuotoja, datan yksityisyyden ongelmia sekä sääntöjen noudattamisen ongelmia, jotka voisivat vahingoittaa yrityksen nimeä ja mainetta, aiheuttaa suunnittelemattomia kuluja sekä estää mahdollisia laillisia vaaroja. Päällimmäisenä hyvä lähestymiskulma datan käsittelyyn voi tarjota parempaa liiketoiminnan suorituskykyä. (What is data management and why is it important?, 2019)

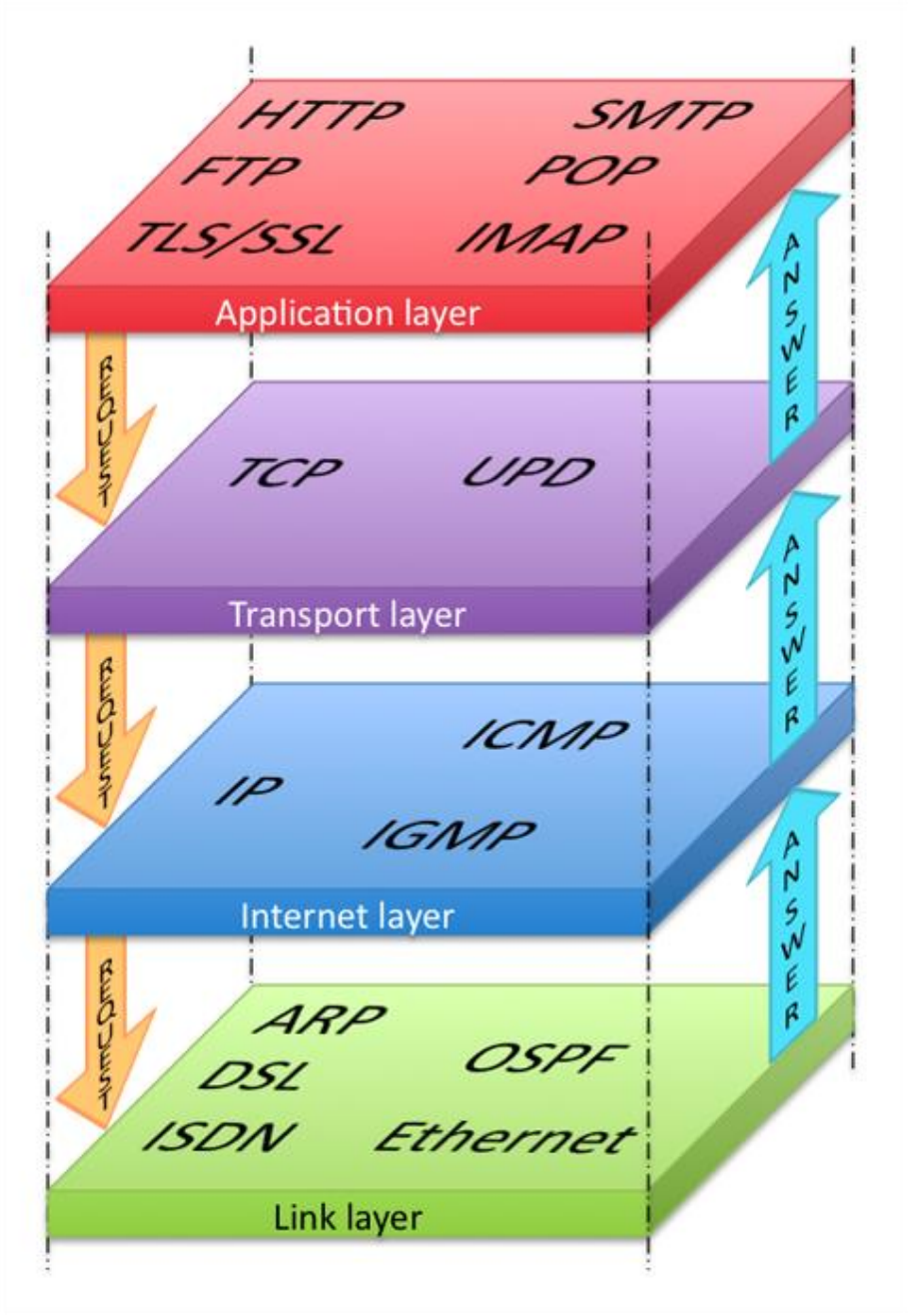
3.3.1 Hajautetun verkon arkkitehtuurin ongelmat

Hajautettujen verkkojen pääongelmat ovat globaalin kellon sekä jaetun muistin puute. Hajautetussa systeemissä on monia eri systeemeitä ja jokaisella on oma kellonsa. Jokaisen systeemin oma kello toimii eri nopeudella tai tarkkuudella, mikä johtaa siihen, että ne ovat asynkronisia. Käynnistyksen aikana kelloja säädellään, jotta aika olisi yhtenäinen koko verkossa, mutta jo yhden kellosyklin jälkeen kellot tulevat olemaan eriävissä ajoissa ja missään kelloissa ei ole tarkkaa aikaa. Aika tiedetään tietyllä tarkkuudella, koska ajan perusteella hajautetuissa järjestelmissä tehdään ajallisia tapahtumien järjestelyä, ajantasaisten tietojen keräämiseksi integroidusta järjestelmästä sekä prosessien ajoittamiseksi. Hajautetuilla systeemeillä ei myöskään ole jaettua fyysistä muistia, jokaisella koneella on oma, spesifinen fyysinen muistinsa. Koska muistia ei ole jaettu, ei verkossa oleva kone voi tietää globaalin hajautetun verkon tilaa. (ashushrma378, 2020)

3.4 TCP/IP

TCP/IP-protokolla on lyhenne englannin kielen sanoista Transmission Control Protocol/Internet Protocol. Se tarkoittaa joukkoa viestintäprotokollia, joilla verkkoon liitetyt laitteet yhdistyvät toisiinsa. Koko IP-pakettia, sääntöineen ja menettelyineen, kutsutaan yleisesti TCP/IP:ksi. TCP ja IP

ovat kaksi tärkeintä paketin protokollaa, pakettiin kuuluu myös muitakin protokollia. Protokollapaketti toimii abstraktikerroksena internet-sovelluksen ja reitityksen välillä. Protokollapino kuvattu kuviossa 2. TCP/IP-protokolla määrittelee, miten data vaihdetaan internetin yli mahdollistamalla päästä-päähän viestintää. Viestintämalli kertoo miten data tulisi pilkkoa paketteihin, osoittaa, siirtää, reitittää sekä ottaa vastaan vastaanottavassa päässä. TCP/IP tarvitsee erittäin vähän keskistä hallintoa ja on suunniteltu tekemään verkot luotettaviksi automaattisen toipumiskyvyn kanssa verkossa olevan laitteeseen tulevan vian sattuessa. (Shacklett, M.).



Kuvio 2. TCP/IP-protokollapino. (Bughunter, 2009)

3.5 Publish-subscribe-malli

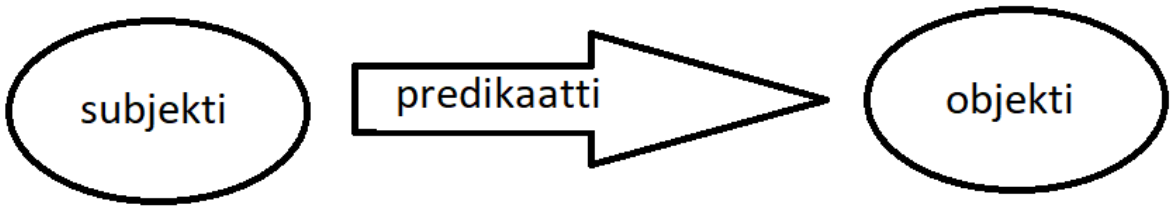
Publish-subscribe-malli on viestien asynkroninen palvelu-palvelu-lähettämisen muoto, jota käytetään serverittömissä sekä mikropalveluiden arkkitehtuureissa. Publish-subscribe-mallissa lähetetty viesti otetaan vastaan kaikkien kuuntelijoiden toimesta, mutta käsitellään vain niiden toimesta, jotka viestin ovat tilanneet. Mallia voidaan hyödyntää tapahtumalähtöisissä arkkitehtuureissa sekä applikaatioiden irti kytkemiseksi tehokkuuden, luotettavuuden ja skaalautuvuuden parantamiseksi. (What is Pub/Sub Messaging?)

3.5.1 Viestien lähettäminen ja vastaanotto

Viesti lähetetään antamalla kutsu julkaisijakomponentille, joka työntää viestin sitä vastaavaan otsakkeeseen. Viestien otsakkeet siirtävät viestejä ilman viivettä tai erittäin pienellä viiveellä ja julkaisevat viestit otsakkeiden tilaajille. Viestit lähetetään verkon jokaiseen koneeseen, oli viesti tilattu tai ei, mutta viestit käsitellään, jos viestin otsake on tilattu. Viestin vastaanottajat tekevät viestin niille määriteltyjä toimintoja. Viestin julkaisijan ei tarvitse tietää, kuka julkaistua viestiä eikä viestin vastaanottajan tarvitse tietää, mistä viesti tulee. (What is Pub/Sub Messaging?).

3.6 RDF-tietokanta

RDF-tietokanta on graafitietokanta, joka tallentaa tiedot verkostona ja käyttää päätelmiä löytääkseen uutta tietoa olemassa olevista suhteista. Sen joustava ja dynaaminen luonne mahdollistaa erilaisten tietojen linkittämisen, indeksoinnin semanttista hakua varten ja sen rikastaminen tekstianalyysin avulla suurien tietokaavioiden muodostamiseksi. (What is an RDF Triplestore?. n.d.) Triple on RDF-tietokannan tiedon tallentamisen muoto. Perusperiaate triplelle kuvattu kuviossa 2.



Kuvio 3. RDF-tietokannoissa käytettävän tietokantaobjektin pelkistetty kuvaus

RDF, joka tulee englannin kielen sanoista Resource Description Framework, on W3C:n standardoima malli tietojen julkaisemiseen ja vaihtoon verkossa. Graafitietokannan triplevarastot tallentavat dataa objektien verkostona, jonka välillä on materialisoituneita linkkejä. Tämä tekee RDF-tietokannoista ensisijaisen vaihtoehdon erittäin yhteen liitetyn tiedon hallintaan. Triplevarastot ovat joustavampia ja halvempia kuin esimerkiksi relaatiotietokanta. RDF-tietokanta, jota usein kutsutaan semanttiseksi graafitietokannaksi, pystyy myös käsittelemään tehokkaita sema nttisia kyselyitä ja käyttämään päätelmiä uuden tiedon paljastamiseen olemassa olevista suhteista. (What is an RDF Triplestore?. n.d.)

3.7 Semanttinen verkko

Semanttinen verkko on verkko, joka on muodostettu datasta. Semanttisten verkkojen teknologioilla mahdollistetaan ympäristö, jossa verkkoa käyttävä sovellus voi tehdä kyselyitä verkon dataan. Verkon muodostamiseen tarvitaan paljon dataa, joka tulisi olla saatavilla standardimuodossa, tavoitettavissa sekä hallittavissa semanttisten verkkojen työkaluilla. (Linked data, n.d).

Sekä W3C (n.d). että O'Brien (2020) toteavat, jotta datasta muodostetun verkon luominen olisi mahdollista tulisi dataan liitettyjen suhteiden olla myös saatavilla. Ilman datan suhteita verkon muodostaminen verkkoon tuodusta datasta ei olisi mahdollista.

3.8 Replikointi

Replikointi on jatkuvaa datan muutoksien kopiointia yhdestä tietokannasta toiseen. Replikoinnin idea on parantaa tietokantojen taakkaa sekä parantaa vikasietokykyä. Vikasietokykyä parantuu koska julkaisevan koneen ongelmatilanteissa tietoa vastaanottava kone voi ottaa julkaisevan koneen roolin verkossa. (Replication, 2011).

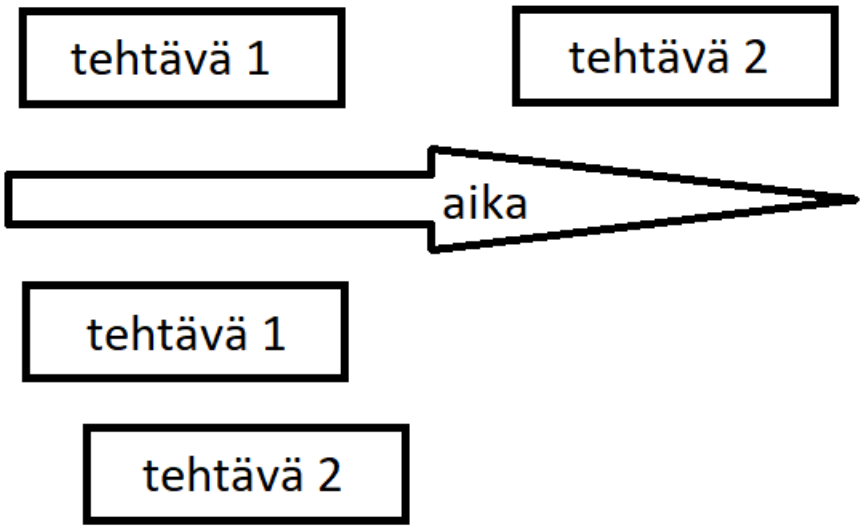
3.9 Asynkroninen tiedonvaihto

Toisin kuin synkronisessa tiedonsiirrossa, asynkronisessa tiedonsiirrossa tiedonsiirto ei ole jatkuvaa eikä tasaisin välein tapahtuvaa. Asynkronisessa tiedonsiirrossa lähettimellä ja vastaanottimella on eri kellotaajuudet. Tiedonsiirtoa lähetyksiä hallitaan käyttämällä aloitus- ja lopetusbittejä tiedonsiirtojen aloittamiseen ja lopettamiseen. Asynkronisessa tiedonsiirrossa tiedonsiirtojen ajoituksilla ei ole niin suurta merkitystä kuin synkronisessa tiedonsiirrossa. (Mikä on ero synkronisen ja asynkronisen tiedonsiirron välillä?, n.d.)

3.10 Säikeistys

Grafiikka- sekä moniydinprosessorien saatavuuden parantuessa sovelluksista on tulossa monimutkaisempia kehittäjien hyödyntäessä säikeitä sovellusten maksimaalisen tehokkuuden ja responsiivisuuden saavuttamiseksi. Monimutkaisuuden kasvaessa kasvaa myös vaikeusaste sovellusten koodin kirjoittamiseen, testaukseen ja sovellusten hallintaan. (Burns, B., 2022).

Burns (2022) ja JavaTPoint (n.d.) toteaa, säikeistäminen mahdollistaa monen prosessin suorittamisen yhdenaikaisesti. Tämä säästää aikaa, koska monen prosessin suorittaminen yhdenaikaisesti vie kokonaisuudessaan vähemmän aikaa, kuin niiden suorittaminen peräkkäin. Ajan säästyminen taas tehostaa sovelluksen toimintaa ja parantaa vasteaikoja. Peräkkäinen metodien suorittaminen sekä säikeistämisen toiminnan kuvaus kuviossa 3. Kuviossa on kuvattu karkeasti säikeistämisen tuoma hyöty samanmittaisten metodien suorittamisen tuomalle hyödylle.



Kuvio 4. Kuviossa kuvattu metodien peräkkäisen sekä säikeistyksen suoritustapojen aikojen vertailua suoritusajassa.

Säikeistäminen tuo kuitenkin monimutkaisuutta ja voi mahdollisesti luoda ongelmia, joita on vaikea korjata. Yhdenaikainen tiedonmuokkaus, resurssien riittämättömyys sekä säikeiden koordinoimattomuuden ongelmat voivat myöskin nousta esiin. (Burns, B. 2020).

3.10.1 Java Future

Java Future esittää asynkronisen laskennan tulosta. Asynkronisen tehtävän luonnin jälkeen palautetaan Future-objekti, jonka tuloksena saatu objekti toimii käsitteenä laskennan tulokselle. Laskennan valmistuttua tulokseen pääsee kiinni Future-objektin kautta, joka luotiin tehtävän luonnin aikana. (Jenkov, J., 2018). On vain otettava huomioon, että Future-objektin tulokseen ei pääse käsiksi ennen kuin laskenta on päätetty. Siksi on tärkeää, että muistetaan antaa laskennalle sen tarvitsema aika suorittamalla jotain muuta ennen tuloksen pyytämistä Future-objektilta.

3.11 Yourkit

YourKit-sovelluksella voidaan seurata ja nauhoittaa Java-sovelluksien käyttämän koneen resurssien käyttöä sekä sovelluksen metodien toimintaa. Sovellus kertoo tarkat tallennettavan sovelluksen

käyttämät metodit sekä suoritusajat, joiden perusteella voidaan päästä kiinni kohtiin koodissa, joiden toimintaa kannattaa tarkastella tarkemmin. YourKit-sovelluksen keräämästä datasta saadaan irti kaikkea muutakin, josta voi olla hyötyä sovelluksen toiminnan kehittämisessä.

3.12 WireShark

WireShark on työkalu, jolla päästään kiinni siihen liitetyn sovelluksen verkon toimintaan protokolla tasolla. Sitä kautta saadaan myös todennettua sovelluksen turvallisten tiedonsiirtojen käyttöä. WireShark mahdollistaa reaaliaikaisen sekä tallennettujen tiedostojen tarkastelun verkon käytöstä sovelluksen verkon protokollien käytöstä.

4 Tutkimus ja kehitys

4.1 Tutkimuksen tavoite ja toimenpiteet

Tutkimuksen tavoitteena oli toiminnan sisäistys ja tiedonhankinnan jälkeinen kehitysideointi. Tutkimus jaettiin kolmeen eri osaa alatutkimuskysymysten mukaan. Jokaiseen alatutkimuskohtaa pyrittiin löytämään ainakin yksi kehitysidea, jonka toiminnan tehostamista pyrittiin testaamaan käytännössä. Kaikkiin kolmeen alatutkimuskysymykseen pyrittiin tehdä työtä kolmessa eri vaiheessa: **tilannekartoitus** (sisältää sovelluksen tämän hetkisen toiminnan ymmärtäminen ja teoreettisen toiminnan tiedonhaku verkosta), **kehitys** (tehtiin tiedonhaun perusteella toimenpiteitä toiminnan parantamiseksi) ja **testaus** (pyrittiin todistamaan tehdyn korjauksen toiminnan parantaminen).

Tutkimus suoritettiin periaatteessa pienenä projektina osana toimeksiantajan tulevaisuuden kehitystyötä. Tarkoituksena oli tehdä selvitystyötä, jonka perusteella voitaisiin allokoida mahdollisia kehitysideoita tulevaisuutta ajatellen. Projekti pyrittiin suorittamaan kirjoittajan työtehtävien ohessa. Tämä taas osittain vaikutti kehitysideoiden vähyyteen ja osittain niiden testaukseen ajanpuutteen takia.

Tutkimuksen tarkoituksena oli kehittää hajautetun verkon nykyistä toimintaa paremmaksi. Lähtötilanteena oli osa toimivaa kokonaisuutta oleva sovelluksen osa ja tämä tilanne ei toiminnan osalta saisi muuttua. Toimeksiantajan sovellus on ollut kehityksessä jo useita vuosia, joten kehityskohteiden lista ei ole kovin laaja. Tilannekartoituksen aikana sovelluksen toiminnallisuutta tarkasteltiin ja pyrittiin löytämään kehityskohteita YourKit-sovelluksella kerätyn datan perusteella.

Toisaalta tutkimuksessa pyrittiin käyttämään toimeksiantajan prosesseja niiden omaksumiseen sekä toimeksiantajan käyttämiä työkaluja, joiden käyttöä kirjoittajan on tarkoitus myös tulevissa työtehtävissä.

Tilannekartoituksen jälkeen aloitettiin kehitys toiminnan tehostamiselle. Tutkija käytti hyväkseen toimeksiantajan projektiryhmää ja sen tietoutta kehitysideoiden ideoinnissa. Myös muualta haettua tietoa pyrittiin käyttämään kehityksien ideoinnissa. Ideointivaiheessa pyrittiin karsimaan

pois ideoita, jotka kyllä höydyttäsivät toimintaa, mutta olisivat liian suurien muutosten vaatimia parannuksia tai joiden implementoinnin riskinä saattaisi olla toiminnan hajoaminen.

4.2 Teknisen aineiston kerääminen ja analysointi

Opinnäytetyölle ei ollut olemassa alustavaa valmista aineistoa. Aineisto, johon kehitysideoiden tuloksia verrattiin, kerättiin sovelluksen toiminnan aikana ennen muutoksia. Kerätty data pyrittiin keräämään työn toimintaa optimoitavien metodien toiminnan aikana, mutta myöhempi tarkastelu osoitti, että se ei ollut mahdollista tutkijan odotusarvon mukaan. Niinpä päädyttiin keräämään vastaavalla tavalla kuin oli ajateltu, mutta myöhempään ja eri tarkoitukseen. Ensimmäisen datakeräystavan osoittauduttua hankalaksi, siirryttiin aikaperusteiseen datankeräystapaan. Tällä tavoin saatiin tarkempia tuloksia tiettyjen metodien suoritusajoista, ja joihin oli helppo verrata uusia arvoja.

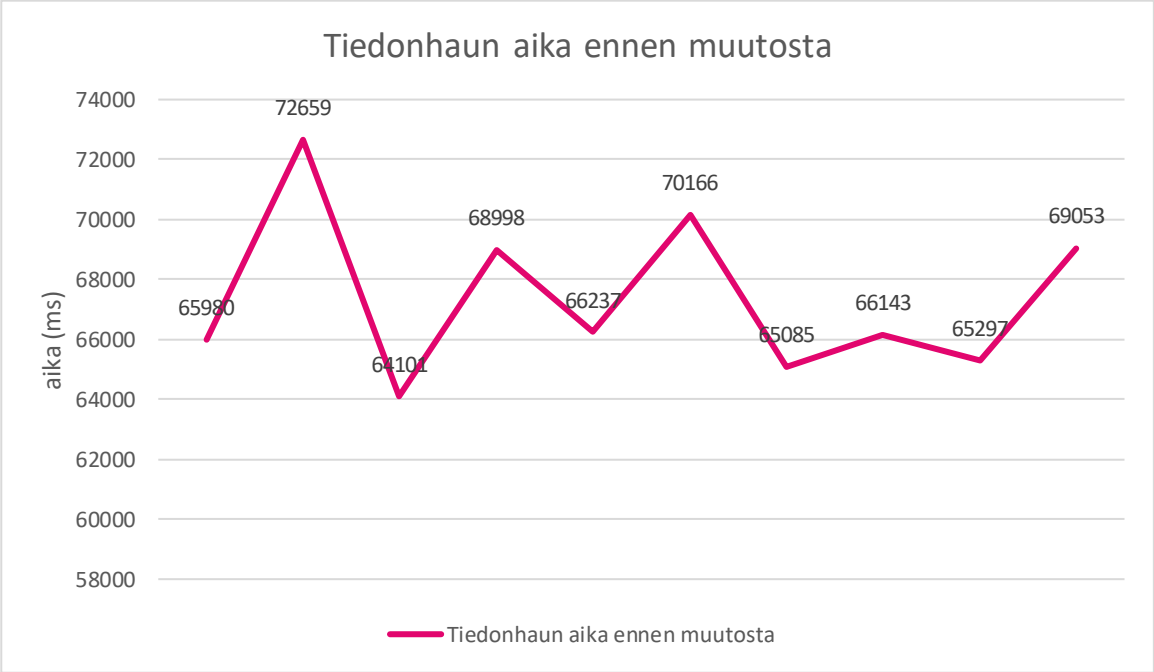
Opinnäytetyössä oli yksi päätutkimuskysymys ja siitä johdettua kolme alatutkimuskysymystä. Alatutkimuskysymyksistä kysymykseen yksi kerättiin eri dataa kuin kahteen viimeiseen. Ensimmäiseen kysymykseen liittyvän datan keräys suoritettiin osittain YourKit-ohjelmalla ja osittain keräämällä tietoa sovelluksen logeista. Kahteen seuraavaan alatutkimuskysymykseen oli kerättävä oma datansa, koska niissä käytettiin eri metodeja sovelluksen toiminnassa. Tarkoituksena oli kerätä sovelluksen suoritusaikoja lähetyksen ja vastaanoton suorittavien metodien käyttämästä ajasta, mutta niiden kerääminen osoittautui hankalaksi rajata sovelluksen kokonaissuoritusajoista.

4.3 Alkuperäisen aineiston keräys

Tarkoituksena oli kerätä dataa erillisellä ohjelmalla (YourKit) toimeksiantajan sovelluksen toiminnan aikana. Datasta pystyy katselemaan sovelluksen koodin eri metodien suoritusaikoja. Pitkien suoritusaikojen perusteella voi löytää kohtia koodissa, joiden toimintaa kannattaa alkaa tarkastelemaan tarkemmin. Toinen vastaavanlainen aineiston keräys mahdollistettiin lisäämällä koodin suoritusyhteyteen rivejä, jotka kertovat metodien suoritusajat sovelluksen sisäiseen logiin. Tämä keino osoittautui tarkemmaksi datankeräys menetelmäksi, koska YourKit-ohjelman kerätystä datasta on vaikea ajoittaa tietyn metodin suoritus ja näin ollen varsinainen suoritusajan aloitus ja lopetus ajan määrittäminen ei ole helppoa.

YourKit-ohjelmalla saadaan sovelluksen eri metodien suoritusaikoja, joiden perusteella voidaan alkaa tarkastelemaan jotakin koodin osaa tarkemmin. YourKit-ohjelma kertoo sovelluksen tarkkojen metodien nimet, joten tarkemman koodin tarkastelun aloittaminen on helppoa. Jos koodista löydetään jokin ongelma, pyritään sille keksimään toimiva ratkaisu.

Kerätty aineisto kuvasi tietyllä tietokannan koolla kuluneen tiedonhaun kesto millisekunteina seuraavaa tiedonsiirtoa varten. Tiedonhaun alkuperäisten arvojen testien aikana tehtävät toimenpiteet oli tarkoitus testata uudelleen kehitysideoiden testien aikana. Tiedonhakua varten sovelluksella suoritetaan tietokannan yksittäinen muutos, joka siirretään verkon nodesta toiseen. Kehitysidean testauksessa seurattiin sovelluksen tekemän laskennan kestoja tiedonsiirtoa varten ja kirjattiin ne ylös. Tuloksista saatiin vastaava taulukko (Taulukko 1).



Kuvio 5. Tiedonhaun kesto millisekunteina, sovelluksen tietokannassa olevan datan määrä 500 000 yksikköä

Sovelluksen toiminnan tarkastelun yhteydessä on tarkoitus tarkistaa, olisiko käytetyistä kolmannen osapuolen kirjastoista tarjolla uudempia versioita. Uusia versioita olisi ollut tarjolla, mutta osittain johtuen muuttuneiden kirjastojen lisensseistä, niiden käyttö ei olisi enää mahdollista sovelluksen tarpeiden perusteella.

Toista ja kolmatta alatutkimuskysymystä varten oli kerättävä eri data kuin ensimmäiseen. Ensimmäinen alatutkimuskysymyksen toiminnallisuus, eli tiedon keräys tiedonsiirtoa varten, on osa viestien lähettämistä, mutta toiminnallisuudet on kuitenkin eritelty tätä tutkimusta varten. Koska tiedonsiirtoa on kahdenlaista, tuli molemmille pyrkiä keräämään oma datansa. Toiminnallisuuksien suoritusajat ovat hyvin lyhyitä, joten alkuperäisen toiminnan suoritusaikojen kartoitus osoittautui mahdottomaksi.

4.4 Tiedon keräys lähetystä varten

YourKit-sovelluksella kerätystä datasta pystyttiin näkemään, että tiedonkeräys on varsin hidasta nykyisellä toteutuksella. Kerätyn datan perusteella pystyttiin kaivamaan pitkäkestoisten metodien nimiä ja aloitettiin niiden toiminnan selvitys. Sovelluksen koodia tarkastelemalla selvisi, että eri metodit, jotka keräävät dataa seuraavaa tiedonsiirtoa varten, suoritetaan peräkkäin. Varsinkin sovelluksen sisäisen tietokannan kasvaessa peräkkäiset haut tiedolle alkoivat viedä paljon aikaa.

Heräsi idea, että olisiko tietoja keräävien metodien suorittaminen asynkronisesti mahdollista. Aloitettiin kehitysidean hahmottelu konsultoimalla työyhteisöön kuuluvia henkilöitä. Heidän kanssaan todettiin metodien säikeistyksen olevan mahdollinen kehitysidea toiminnallisuuden tehostamiseksi. Tehostaminen kuitenkin vaatisi kirjoittajalle tuntemattoman Future-kirjaston käyttöä. Future edustaa asynkronisen laskennan tulosta Java-kielessä.

Tutkija aloitti Future-kirjastoon tutustumisen niin verkosta kuin sovelluksesta löytyvistä muista Future-kirjastoa käyttävistä luokista ja pyrki selvittämään sen tarkoitusta sekä kuinka sitä käytetään. Tiedonhaun tuloksena tutkijan oli mahdollista suunnitella Future-kirjastoa käyttävä kehitysehdotus toiminnan tehostamiseksi. Suunnitelmassa oli pyritty ottamaan huomioon erilaisten metodien toimintojen tarpeet ja mahdollinen erilainen implementointi. Onneksi sovelluksen tiedonkeräyksen toiminnallisuuksien implementoinnit vastasivat paria kohtaa lukuun ottamatta toisiaan ja eriävätkin kohteet oli mahdollista saada toimimaan Futurea käyttäen.

Alkuperäisen toiminnan sijaan, jossa kaikki tietoa hakevien metodien suoritukset tulivat toistensa perään, ne annettaisiin Future-luokalle suoritettavaksi taustalla asynkronisesti. Tiedonhaun suorittaminen taustalla vaatii kuitenkin valmisteluja koodin osalta. Käytettävissä olevien ytimien määrä tulee selvittää ja tehdä toiminnallisuutta, jotta käytetään oikeaa määrää olemassa olevien ytimien

mukaan. Käytettävissä olevien ytimien saaminen koodin kautta oli myös pieni tiedonetsinnän paikka. Tiedonetsinnän jälkeen tietokoneessa olevien ytimien hakeminen koodin kautta onnistui ja seuraavana oli niiden määrän määrittely, joka lasketaan käytettävissä olevien ytimien määrästä. Pääsääntönä on se, että ei käytetä liian montaa ydintä rinnakkaisten prosessien suorittamiseen ja ettei niitä käytetä liikaa ja ettei muu sovelluksen suorittaminen hidastu tai jopa pysähdy. Laskenta perustuu sovellusta pyörittävän koneen ytimien määrään ja on korkeintaan ytimien määrä miinus 1. Jos käytössä on enemmän ytimiä, annetaan sovellukselle tarvittaessa enemmän ytimiä käyttöön.

Tiedonkeräyksen jokaiselle metodille tehdään oma Future-objekti, joka suorittaa vanhan toiminnallisuuden taustalla. Jotta Future-objektit ehtivät suorittaa metodinsa, tulee Future-objektien kutsumisen jälkeen antaa niille aikaa suorittaa tehtävänsä suorittamalla jotakin muuta ennen kuin Future-objektilta voidaan alkaa kyselemään taustalla suoritetun prosessin tulosta. Jos Future-objekti ei ehdi suorittaa sille annettua metodia, se ei palauta mitään ja voi vaikuttaa sovelluksen toimintaan muualla. Tässä tilanteessa lähetykseen ei tulisi mukaan lähetettävää dataa ja vastaanottajalle ei tulisi sille kuuluva tieto lähetyksen mukana.

Taustalla suoritetun tiedonkeräyksen jälkeen Futurelta pyydetään sen keräämät tiedot talteen ja tehdään sille annetut toimenpiteet. Tähän tutkimukseen implementoitiin toiminnallisuus yksitellen tietojen keräys jokaiselta Future-objektilta erikseen. Tieto Futurelta kerätään ja asetetaan osaksi lähetettävää viestiä. Kun kaikki käytetyt Futuret on käyty läpi ja tiedot kerätty, palautetaan kerätty data toiselle metodille, jonka kautta tiedonsiirto hoidetaan.

Kesken ensimmäisen kehitysidean tekemisen huomattiin seuraava mahdollinen korjausidea. Tutkija huomasi, että on mahdollista että kahden peräkkäisen pitkäkestoisen haun suorittaminen voi alkaa hidastamaan sovelluksen toimintaa. Kyseessä olisi siis tilanne, jossa säikeistämisen idea häviää, koska suoritusaika ei lyhene niin paljon kuin olisi mahdollista. Syntyi toinen idea kehitykselle, jossa sovelluksen metodien suoritusjärjestystä muutettaisiin niin, että mahdollisuus tiedonhaun venymiselle pienenisi, eli siirretään tiedonhakuja suorittavat metodit sellaiseen suoritusjärjestykseen, että minimoidaan mahdollisuus kahdelle pitkälle metodisuoritukselle. Tarkoituksena oli siis siirtää pidempikestoiset metodit suoritettavaksi heti tiedonhaun alussa. Näin ollen

niiden peräkkäinen suoritus samassa säikeessä on vähemmän todennäköistä ja kokonaisuudessa tiedonkeräyksen ajallinen suoritusaika lyhentyisi.

Koska molemmat korjaukset osuvat samaan luokkaan, kannattaa ne testata samassa testissä. Kehitysidean implementoinnin jälkeen voitiin aloittaa vertailu alkuperäiseen kerättyyn dataan. Kehitysideoiden testaus suoritettiin vastaanvankaltaisena kuin alkuperäisen datan keräyksen aikana, suo-ritettiin sovelluksella toiminto, joka tekee muutoksen sovelluksen tietokantaan ja seurattiin sovelluksen tiedonsiirron viemää aikaa, kun haetaan tietoa lähetykseen sovelluksen logeista.

4.5 Viestien lähettäminen

Kerätystä datasta pystyttiin toteamaan, että viestien lähettämisessä on käytössä tehokas suoritustapa. Aika, joka lähetyksiin kuluu, on kohtuullinen ja perusteltu eikä lähetyksiin liittyvän koodin lähempi tarkastelu antanut perusteluja muutoksille. Lähetyksissä käytetyissä kirjastoissa ei ole paljoa sijaa muutoksille, publish-subscribe-malli on todella suoraviivainen toteuttaa ja johon ei ole tullut paljoa muutoksia kirjaston käyttöönoton jälkeen. TCP/IP-protokollissa on periaatteessa sama tilanne, mitään mullistavaa ei ole löydetty implementoinnin jälkeen. Molemmissa on uusia versioita tarjolla, mutta niiden tuomat lisenssimuutokset ja pienet pakkauksen muutokset eivät toisi sovellukselle suurta hyötyä.

Koodin tutkimisen jälkeen alkoi ideointi optimoinneille. Yksi mieleen tullut idea olisi lyhentää viestejä tai pyrkiä parantamaan viestien pakkausta. Toimeksiantajalla oli jo tiedossa viestien pituuksia lyhentävän kirjaston käyttöönotto, jossa tietokantojen tietokantaobjektien tunnisteita saataisiin lyhennettyä ilman suurempaa riskiä ristiriitaisesta datasta. Joten uusien ideoiden tulisi olla parempia kuin suunnitellun kirjaston mahdollistamat edut, ilman ristiriitaisen tiedon mahdollisuuden kasvamista. Tiedonetsintä aiheesta ei tuottanut jo tiedossa olevan muutoksen lupaamien parannusten tasoisia tuloksia, joten annettiin tämän kehitysidean jäädä.

Koska lähetykseen käytetään kahta eri mallia, publish-subscribe-mallia sekä TCP/IP-protokollia hyödyntävää tiedonvaihtoa, oli tarkasteltavia kirjastoja enemmän.

Publish-subscribe-mallin kirjasto ei ole vanha, mutta uusia versioita olisi muutama tarjolla, tarkasteluhetkellä samalla viikolla olisi tullut uusin versio. Uudessa versiossa olisi tarjolla uusia korjauksia ja hieman parempi pakkaus, jolla olisi mahdollista nopeutta tiedonsiirtoa. Nykyinen toiminnallisuus on kuitenkin niin hyvällä tasolla, että saatava parannus olisi kuitenkin pieni.

TCP/IP-protokolla mallissa viestien lähettämiseen käytettävä serialisointikirjasto on muutaman version vanhempi kuin uusin julkaisu. Uuden version implementointi ei olisi iso muutos, mutta saatavat hyödyt ovat minimalistiset.

4.6 Viestien vastaanottaminen

Alkuperäisen kerätyn aineiston perusteella aloitettiin tutkimaan sovelluksessa tapahtuvien tiedonvaihtojen vastaanottamista. Kerätyn datan perusteella ei löytynyt ainakaan suoraan ongelmakohtia sovelluksen toiminnassa tämän toiminnallisuuden kohdalla. Viestien käsittely vastaanoton yhteydessä kuluttama aika ei ollut hälyttävän suuri.

Koodin tarkastelu nosti yhden idean esille. Koska sovellus käyttää tietojen tallentamiseen asynkronista tallennusta, olisi mahdollista korottaa kerralla lähetettävän tietotyypin määriä. Tämä vaihtoehto saattaisi aiheuttaa kuitenkin pakettien katoamista tiedonsiirtojen aikana. TCP/IP-protokollissa on kyllä mukana toiminnallisuutta, joka tunnistaa pakettien jäämisen matkalle ja vaatii uudelleenlähetystä, jos paketteja jää matkalle. Joten pakettien matkalle jääminen ei pitäisi olla ongelma, toki uudelleenlähetys voi syödä aikaa muulta suoritukselta verkossa. Kehitysidean toiminnallisuutta ei olisi kovin helppo todentaa, sillä on lähes mahdotonta lähettää täsmälleen samanlaisia paketteja verkossa osapuolten kesken.

Viestien lähetykseen tulossa olevan viestejä lyhentävän kirjaston vaikutukset tulevat näkymään myös viestien vastaanottamisessa, viestien mukana tulevien lyhyempien objektien vertailu paikalliseen dataan vertailtaessa lyhentää objektien käsittelyaikoja ja lyhentää näin vastaanottamisenkin kokonaisaikaa.

Idea tietotyypin määrien noston per viesti takana oli, että vähempien viestien lähettäminen ja vastaanottaminen on nopeampaa kuin monen pienemmän viestin vastaava käsittely. Molemmilla

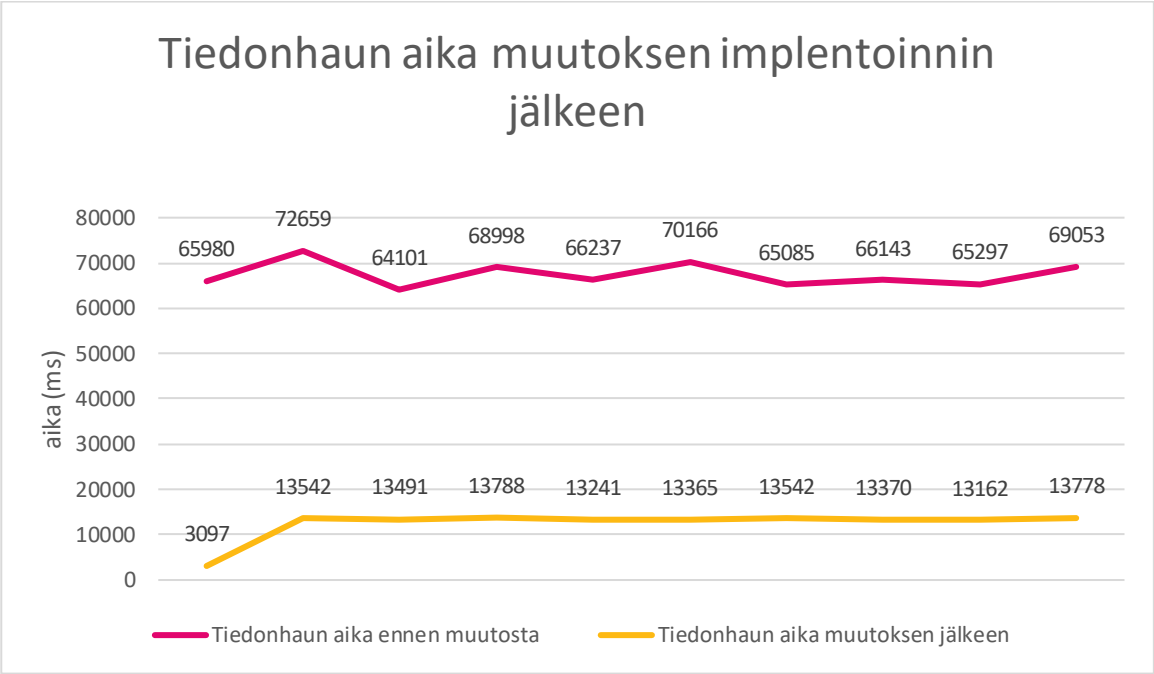
vaihtoehdoilla on kannattajansa ja molemmissa on hyvät ja huonot puolensa. Pienempien pakettien lähettäminen voi parantaa verkon latenssia, mutta voi huonontaa isojen kokonaisuuksien tiedonsiirtoa verkossa. Suuremmilla paketeilla taas saataisiin kaikki tieto tarpeen tullen kerralla siirrettyä, mutta suuren paketin lähettäminen ja vastaanottaminen voi rasittaa verkkoa enemmän.

Tämän kehitysidean implementointi ei ollut hankalaa, mutta testaaminen sitäkin hankalampaa. Tiedonsiirrossa liikkuvien pakettien koko ei voi määritellä tarkasti jokaiselle eri lähetettävälle käsittelyn osalle. Idea pakettien koon muuttamisesta oli hyvä idea, mutta kehitysidean testaaminen ja todentaminen onkin sitten eri juttu. Tuloksia yritettiin saada kerättyä niin verkon protokollien kautta kuin sovelluksen metodien suoritusaikojen kautta, mutta tuloksia ei saatu kunnolla eristettyä tietystä toiminnallisuudesta.

5 Tulokset

5.1 Tiedon keräys lähetystä varten

Tähän kysymykseen tehty kehitysidean testaus antoi erittäin hyviä tuloksia. (vrt Kuvio 1 ja Kuvio 2.)



Kuvio 6. Testitulokset kehitysidealle, tietokannan koko 500 000 yksikköä

Verrattuna alkuperäiseen toimintaan sovelluksen tiedonhaun toiminnallisuuden viemä aika näyttäisi parantuvan monella kymmenellä sekunnilla, noin neljäsosaan alkuperäisestä. Jos tämä kehitysidea haluttaisiin ottaa käyttöön, tulisi toiminnalle suorittaa paljon testausta, sillä tämä osan muutokset voivat vaikuttaa sovelluksen toimintaan toisaalla. Tässä testauksessa pyrittiin vain todentamaan kehitysideaa koskevan toiminnallisuuden parantuminen. Testauksessa tulisi ainakin varmentaa, että kaikki haluttava data tulee tiedonsiirron mukaan. Mutta sovelluksen toiminta kehitysidean toiminnan aikana näytti alustavasti toimivan alkuperäistä tilannetta vastaavalla tavalla.

5.2 Viestien lähettäminen

Kehitysideoiden hyödyt teoreettisella tasolla eivät antaneet osviittaa parantuvasta toiminnasta, joten tähän liitettyjen kehitysideoiden testaustulokset jäivät keräämättä.

5.3 Viestien vastaanottaminen

Tähän alatutkimuskysymykseen kohdistuneiden kehitysidean testauksesta ei saatu riittäviä todisteita, että yhden lähetyksen aikana siirrettävien pakettien koon nostamisella voitaisiin saada hyötyjä viestien vastaanottamisessa. Niin tiedonsiirtojen seurannan että metodien suoritusaikojen perusteella ei voitu varmaksi sanoa, että toiminta olisi tehostunut. Oletusarvo on että suuremmilla pakettien arvolla voitaisiin saada selkeämpiä tuloksia, mutta aikataulun tiukkuuden takia sitä ei ehditty testaamaan.

6 Yhteenveto

6.1 Vastaus tutkimuskysymykseen

Aikarajoitteiden ja muiden ongelmien jälkeen tutkimuskysymykseen, joka määriteltiin muotoon **hajautetun verkon optimointi,** löydettiin yksi uusi konstruktio. Kysymyksestä saatiin vielä muodostettua kolme alitutkimuskysymystä, jotka liittyivät hajautetun verkon toimintaan oleellisesti. Tämän työn tutkimuskysymykseen vastattiin löytämällä kehityskeino sovelluksen tiedonsiirtoihin liittyvän tiedonhaun kautta. Kehitystyön prosessi vastasi hyvin suunniteltua, jossa tiedonhaun kautta pyrittiin löytämään kehitysideoita toiminnalle. Lähtökohta työhön oli haastava, sillä sovellus on ollut kehityksessä niin pitkään ja toiminnallisuudet ovat niin huippuunsa hiottuja, että parannusten tekeminen vaatisi mitä luultavammin suuremman remontin toiminnallisuuden parantumiselle.

Konstruktiivinen tutkimusote näkyi hyvin työn suorittamisen aikana. Työn yhteydessä löydetty uusi konstruktio tuo toimintaan tehokkuutta, jota työssä alun perinkin haettiin: ratkaisua reaalimaailman ongelmaan. (Lukka, n.d.) Konstruktiivisen tutkimusprosessin ominaispiirre käytännön edustajien kanssa työskentelystä toteutui osana toimeksiantajan projektiryhmää. Lopputulokseen peilaten konstruktiivinen tutkimusote oli hyvä valinta laadullisen tutkimuksen toteuttamiseen.

Pelkän tutkijan näkökulma oli konstruktiivisen tutkimusotteen tiedetty heikkous. Toimeksiantajan projektiryhmän konsultointi auttoivat merkittävästi saadun lopputuloksen aikaan saamisessa. Heiltä saadut kommentit ja vinkit kehitysideoiden implementoinnista auttoivat saamaan näkökulmaa tutkimukselle.

6.2 Tulosten ja tietoperustan välinen suhde

Tietoperusta oli hieman huteralla pohjalla tässä työssä. Aiheen rajaus onnistui mielestäni hyvin mutta tietoperustan koonti aiheeseen liittyen ei ainakaan ollut onnistunut. Osaa tietoperustan teoriasta ei tullut käytettyä kehityksen aikana ollenkaan.

Säikeistys on keskeinen osa sovelluksien kehitystä, ja sillä saadaan useimmiten toimintaa parantanettua. Tässä tapauksessa säikeistyksellä saatava hyöty on suuri, sillä haun sovelluksen tietokannasta suurilla tietomäärillä alkaa merkittävästi hidastamaan toimintaa. Toki on pidettävä mielessä sovellusta suorittavien tietokoneiden tai laitteiden luomat rajoitteet säikeistyksiä tehdessä. Liian monen yhtäaikaisen suorituksen ajo voi hidastaa sovelluksen toimintaa.

Future on myös erittäin kätevä tapa saada sovelluksien toiminnan tehostamiseksi. Nykyajan grafiikka- ja moniydinprosessorit mahdollistavat monien samanaikaisten tehtävien suorittamisen ja sitä kannattaa pyrkiä hyödyntämään sovellusten kehittämisessä.

Hajautettujen verkkojen käyttö tulee varmasti kasvamaan tulevaisuudessa datamäärien kasvun myötä. Hajauttamalla verkkoja ja tekemällä nodeja verkkoon haluttavan tiedon vasteaikoja saadaan lyhennettyä suurestikin, varsinkin kun tiheissä kohdissa verkkoja tehdään uusia solmuja verkkoon.

6.3 Toimeksiantajan saamat hyödyt

Toimeksiantajan saamat hyödyt voisi luokitella hyviksi. Testitulosten perusteella toiminnallisuuden toimintaa voitaisiin alustavien testien perusteella parantaa merkittävästi muutoksella, joka tutkimuksen aikana kehitettiin ja testattiin. Kehitetty ehdotus ei mitä luultavammin ole lopullisessa muodossaan, mutta suuntaa antava lopulliselle sovellukseen tulevaisuudessa implementoitavalle toiminnallisuudelle.

Kahteen vastaamattomaan alatutkimuskysymykseen ei tässä tutkimuksessa löydetty vastauksia, mutta se ei ole koko totuus. Tutkimuksen aikataulun takia suljettiin pois suuret korjausehdotukset, joista voisi olla hyötyä myös näihin vastaamattomiin alatutkimuskysymyksiin.

Myös sovelluksen käyttöä suurilla datamäärillä, joilla sitä nyt testattiin, paljasti monta kohtaa sovelluksen käytöstä, mihin kannattaisi olla kiinnittämässä huomiota. Nämä löydetyt uudet ongelmat eivät osuneet tutkimukseen tehtyihin rajoihin, joten niiden selvittely ja sekä korjaus- että kehitystyö tulee tehdä tämän työn ulkopuolella.

6.4 Haasteet ja kehittämisehdotukset

Aikataulu muodostui suurimmaksi ongelmaksi tämän tutkimuksen suorittamiseksi. Aikataulun tiukkuuden vuoksi kehitysehdotuksia jouduttiin esimerkiksi pitkän kehitysajan takia rajaamaan pois. Myös testausajat jäivät varsin pieniksi ja testaaminen ylipäänsä aika suppeaksi ajan puutteen takia.

Kehittämisehdotuksia työlle voisi olla puuttuneiksi jääneiden kahden alatutkimuskysymyksen auki jääneiden kysymyksien vastaaminen tai uudelleentutkiminen. Kuten todettu, toimeksiantajan sovellus on ollut kehityksessä jo yli vuosikymmenen ajan ja siinä ajassa on ehditty tekemään paljon, että sovelluksen suorituskyky on tällä tasolla.

6.5 Mahdollisia jatkokehitysaiheita

Uskoisin, että sovelluksessa olevia toiminnallisuuksia olisi mahdollista käyttää tutkimuksessa löydettyä ratkaisua sekä edelleen säikeistää tiedonvaihdonkin toiminnan tehostamiseksi. Tähän tutkimukseen tehty kehitysehdotus ei ole lopullinen versio, miten hajautetun verkon toimintaa voitaisiin jatkokehittää. Mutta on pidettävä mielessä, että kaikkea ei voi eikä kannata edes yrittää optimoida.

Uskoisin myös, että sovelluksessa olisi mahdollista käyttää samaa lähestymistapaa muuallakin toiminnallisuudessa. Tutkimuksen kaltaisen toiminnallisuuden muuttaminen peräkkäisistä metodien suorittamisesta rinnakkain suoritettavaksi toisi tehokkuutta muissa monia pitkäkestoisia metodeja sisältävien toiminnallisuuksien suorituksessa.

Pidemmällä aikataululla voitaisiin harkita jopa suurempia muutoksia vaativia ideoiden kehitystä sekä testaamista. Käytössä olevien kirjastojen toteutukset ovat suoraviivaisia ja niiden optimoinneille ei ole montaa kohtaa mihin edes kohdistaa pieniä ongelmia. Uusien kirjastojen tuomisella sovellukseen voitaisiin saada toimintaa tehokkaammaksi.

Lähteet

ashushrma378. 2020. Limitation of Distributed System. Artikkeli. Viitattu 28.4.2022. https://www.geeksforgeeks.org/limitation-of-distributed-system/.

Bughunter. 2009. TCP/IP-protokollapino. Verkkosivu. Viitattu 7.5.2022. https://fi.wikipedia.org/wiki/TCP/IP#/media/Tiedosto:InternetProtocolStack.png.

Burns, B. 2020. What Is Multithreading: A Guide to Multithreaded Applications. Blogipostaus. Viitattu 6.5.2022. https://totalview.io/blog/multithreading-multithreaded-applications.

Data transfer. 2013. Verkkosivu. Viitattu 1.5.2022. https://www.techopedia.com/definition/18715/data-transfer.

Distributed Databases. n.d. Opaskirja. Viitattu 24.4.2022. https://docs.oracle.com/cd/A57673_01/DOC/server/doc/SCN73/ch21.htm

HSG. 2012. Business Rule Management System. Verkkosivu. Viitattu 28.4.2022. https://www.hartmannsoftware.com/Blog/Enterprise-Rule-Applications/brms.

Hyvä tieteellinen käytäntö (HTK). 2021. Verkkosivu. Viitattu 1.5.2022. https://tenk.fi/fi/tiedevilppi/hyva-tieteellinen-kaytanto-htk.

JavaTPoint. n.d. Multithreading in Java. Verkkosivu. Viitattu 6.5.2022. https://www.javatpoint.com/multithreading-in-java.

Jenkov, J. 2018. Java Future. Verkkosivu. Viivattu 6.5.2022. https://jenkov.com/tutorials/java-util-concurrent/java-future.html.

Kananen, J. 2015. Online research for preparing your thesis: a guide for conducting qualitative and quantitative research online. Jyväskylä: JAMK University of Applied Sciences. Viitattu 1.5.2022. https://janet.finna.fi/, Booky.fi.

Laadullisen tutkimuksen tekeminen. n.d. Verkkosivu. Viitattu 1.5.2022. https://fi.surveymokey.com/mp/conducting-qualitative-research/.

Laadullinen tutkimus. 2021. Verkkosivu. Viitattu 1.5.2022. https://koppa.jyu.fi/avoimet/hum/menetelmapolkuja/menetelmapolku/tutkimusstrategiat/laadullinen-tutkimus.

Lukka, K. N.d. Konstruktiivinen tutkimusote. Metodix - metoditietämystä kaikille -sivusto. Viitattu 1.5.2022. https://metodix.fi/2014/05/19/lukka-konstruktiivinentutkimusote/

Stedman,C & Vaughan, J. 2019. What is data management and why is it important? Verkkosivu. Viitattu 24.4.2022. https://www.techtarget.com/searchdatamanagement/definition/data-management

Mikä on ero synkronisen ja asynkronisen tiedonsiirron välillä? n.d. Verkkosivu. Viitattu 6.5.2022. https://fi.strephonsays.com/what-is-the-difference-between-synchronous-and-asynchronous-data-transfer.

Miller, E. 2017. Akamai's Peer-To-Peer Love Story. Blogiteksti. Viitattu 22.4.2022. https://blog.peer5.com/akamais-peer-to-peer-love-story/

N-able. 2018. Centralized Networks vs Decentralized Networks. Blogipostaus. Viitattu 18.4.2022. https://www.n-able.com/blog/centralized-vs-decentralized-network

Nummenmaa, L., Holopainen, M., Pulkkinen, P. (2014). Tilastollisten menetelmien perusteet. Sanoma Pro.

O'Brien, F. 2020. Developing For The Semantiv Web. Artikkeli. Viitattu 5.5.2022. https://www.smashingmagazine.com/2020/10/developing-semantic-web/.

Replication. 2011. Verkkosivu. Viitattu 7.5.2022. https://www.techopedia.com/definition/1236/replication.

Semanttinen Web. n.d. Verkkosivu. Viitattu 18.4. https://www.profium.com/fi/teknologiat/metadata/semanttinen-web/

Shacklett, M., Novotny, A & Gerwig, K. n.d. TCP/IP. Verkkosivu. Viitattu 28.4.2022. https://www.techtarget.com/searchnetworking/definition/TCP-IP.

TCP/IP protocols. 2022. Verkkosivu. Viitattu 18.4. https://www.ibm.com/docs/en/aix/7.2?topic=protocol-tcpip-protocols

Touron, M. 2019. Centralized vs Decentralized vs Distributed Systems. Blogiteksti. Viitattu 28.4.2022. https://berty.tech/blog/decentralized-distributed-centralized.

Vermaark, W. 2021. What Are Peer-to-Peer (P2P) Networks? Artikkeli. Viitattu 24.4.2022. https://coinmarketcap.com/alexandria/article/what-is-peer-to-peer-p2p

Indeed Editorial Team. 2021. What Is a Peer-to-Peer (P2P) Network?. Verkkosivu. Viitattu 28.4.2022. https://www.indeed.com/career-advice/career-development/what-is-a-peer-to-peer-network.

Virtanen, A. 2006. Ammattikasvatuksen aikakauskirja, Ajankohtaisia teemoja ammattikasvatuksesta. Artikkeli: Konstruktiivinen tutkimusote, Miten koulutus ja elinkeinoelämänodotukset kohtaavat ammattikorkea-koulun opinnäytetöissä 46–52. Viitattu 1.5.2022. https://journal.fi/akakk/article/view/114874/67807?acceptCookies=1.

W3C. n.d. Linked data. Verkkosivu. Viitattu 5.5.2022. https://www.w3.org/standards/semanticweb/data.

What is an RDF Triplestore?. n.d. Verkkosivu. Viitattu 28.4.2022. https://www.ontotext.com/knowledgehub/fundamentals/what-is-rdf-triplestore/.

What is Pub/Sub Messaging?. n.d. Nettisivu. Viitattu 18.4.2022. https://aws.amazon.com/pub-sub-messaging/

What is this? n.d. Verkkosivu. Viitattu 26.4.2022. https://github.com/yacy/yacy_search_server.

YaCy is free software for your own search engine. n.d. Verkkosivu. Viitattu 26.4.2022. https://yacy.net/.

Statista Research Department. 2022. Volume of data/information created, captured, copied, and consumed worldwide from 2010 to 2025. Verkkosivu. Viitattu 1.5.2022. https://www.statista.com/statistics/871513/worldwide-data-created/.