SAVONIA

OPINNÄYTETYÖ - **AMMATTIKORKEAKOULUTUTKINTO**
TEKNIIKAN JA LIIKENTEEN ALA

TIETOJÄRJESTELMÄN SUORITUSKYVYN VALVONTA

TEKIJÄ: Joonas Karjalainen

SAVONIA-AMMATTIKORKEAKOULU

OPINNÄYTETYÖ
Tiivistelmä

Koulutusala Tekniikan ja liikenteen ala	
Koulutusohjelma/Tutkinto-ohjelma Sähkötekniikan koulutusohjelma	
Työn tekijä Joonas Karjalainen	
Työn nimi Tietojärjestelmän suorituskyvyn valvonta	
Päiväys 17.05.2018	Sivumäärä/Liitteet 45+0
Ohjaajat Pekka Granroth, lehtori, Pasi Liimatainen, lehtori	
Toimeksiantaja/Yhteistyökumppani GE Healthcare Finland Oy, Aki Tirkkonen	

Tiivistelmä

Työn aiheena oli CHA System Performance Monitoring Dashboard-työkalun kehittäminen. Kyseinen työkalu on GE Healthcaren tarjoama ratkaisu sairaalaverkon sisällä toimivien tietojärjestelmien suorituskyvyn valvontaan. Opinnäytetyön tavoitteena oli perehtyä sairaalaverkon tietojärjestelmien datan keräämiseen ja visualisointiin, jotta tietojärjestelmien ylläpitäjät voisivat helposti monitoroida järjestelmien suorituskykyä.

Työ aloitettiin teoriaosiolla, jossa ensimmäisenä perehdyttiin visualisoinnin eri menetelmiin. Tämän jälkeen työssä siirryttiin monitoroinnin teorian tutkimiseen, jonka yhteydessä selvitettiin, mitä osa-alueita tietojärjestelmistä tulisi monitoroida ja miten. Teoriaosion jälkeen siirryttiin toteutusvaiheeseen, joka aloitettiin oikeaa sairaalaympäristöä mallintavan testiympäristön konfiguroimisella. Konfiguroinnin jälkeen alettiin tekemään muutoksia CHA System Performance Monitoring Dashboard-työkaluun ja sen esittämiin visualisointeihin.

Opinnäytetyön lopputuloksena syntyi päivitetty versio CHA System Performance Monitoring Dashboard-työkalusta, joka tarjoaa asiakkailleen kattavan kokonaisuuden tietojärjestelmiensä kokonaisvaltaiseen monitorointiin proaktiivisella tasolla. Työkalun päivittämisen yhteydessä sovellettiin työn teoriaosion yhteydessä opittuja menetelmiä.

Avainsanat
suorituskyky, monitorointi, tietojärjestelmä

SAVONIA UNIVERSITY OF APPLIED SCIENCES

THESIS
Abstract

Field of Study Technology, Communication and Transport			
Degree Programme Degree Programme in Electrical Engineering			
Author Mr. Joonas Karjalainen			
Title of Thesis System Performance Monitoring			
Date	17.05.2018	Pages/Appendices	45+0
Supervisors Mr. Pekka Granroth, Principal Lecturer, Mr. Pasi Liimatainen, Principal Lecturer			
Client Organisation /Partner GE Healthcare Finland Oy, Mr. Aki Tirkkonen			

Abstract

The subject of this thesis was to improve the CHA System Performance Monitoring Dashboard-tool. The tool is GE Healthcare's solution to monitor the performance of hospital information systems. The goal of this thesis was to study the gathering and visualization of the data that these hospital information systems offer, so that the local administrators at different hospitals could easily monitor the state of their information systems and its infrastructure.

The thesis was started with a theoretical component regarding the different possibilities and methods of data visualization. After that, the work continued by studying the theories involved around system performance monitoring. During this time, it was also investigated that which different components of the information system should be monitored and how.

After studying the different theories involved around the subject, it was time to move on to the implementation phase of the thesis. This section was started by configuring a demo environment, which represented an actual hospital environment. After the demo environment was configured, the improving and developing of the CHA System Performance Monitoring Dashboard-tool was started.

As a result, a new upgraded version of the CHA System Performance Monitoring Dashboard-tool was created. The tool offers its end users a comprehensive way to monitor their information systems on a proactive level. The different methodologies learned during the theoretical section of this thesis were also applied while developing the tool.

Keywords
performance, monitoring, information system

SISÄLTÖ

LYHENTEET JA KÄSITTEET

LYHENTEET JA KÄSITTEET

Aikasarjatietokanta (Time Series Database) = Tietokantamalli, jossa data tallennetaan ja indeksoidaan aikaleiman mukaan.

HDD (Hard Disk Drive) = Massamuisti, jossa tieto tallennetaan yhden tai useamman metalli- tai lasikiekon pinnalla sijaitsevaan materiaaliin ferromagnetismin avulla.

JSON (JavaScript Object Notation) = Avoimen standardin tiedostomuoto tiedonsiirtoon.

I/O (Input / Output) = Tiedon siirtäminen tai signaloiminen tietokoneen komponenttien välillä.

IOPS (Input / Output Operations Per Second) = Kirjoitus- ja lukuoperaatioiden määrä per sekunti.

Kojelauta (Dashboard) = Grafanan tarjoama malli visualisointien toteuttamiseen, joka tallennetaan JSON-muodossa.

MTA (Mail Transfer Agent) = Ohjelmisto, joka välittää sähköpostiviestejä päätteiden välillä.

Paneeli (Panel) = Grafanan tarjoama komponentti, jonka avulla visualisoidaan dataa kojelaudoille.

RAM (Random Access Memory) = Tietokoneen keskusmuisti, jonne suoritettavat ohjelmat ja käyttöjärjestelmä ladataan.

SMART (Self-Monitoring, Analysis, and Reporting Technology) = Kiintolevyjen kunnon seurantajärjestelmä.

SSD (Solid State Drive) = Massamuisti, joka ei sisällä liikkuvia osia ja jossa tieto säilyy ilman virtaa.

1 JOHDANTO

Tietojärjestelmän seurannalla eli monitoroinnilla tarkoitetaan tietojärjestelmän suorituskyvyn valvontaa. Sen avulla on mahdollista tarkastella erinäisiä tietoja järjestelmään liittyen, joita ovat esimerkiksi reaaliaikaiset tiedot käytetyistä laitteistoresursseista (suoritin, levy, verkko ja muisti). Näitä resursseja ja muita laitteiston suorituskykyyn liittyviä muuttujia tarkastelemalla on mahdollista saada kattava kokonaiskuva tietojärjestelmän toimivuudesta.

Opinnäytetyön tavoitteena on perehtyä tietojärjestelmien datan keräämiseen, jäsentämiseen ja visualisointiin, jotta CHA System Performance Monitoring Dashboard-työkalun avulla voitaisiin tunnistaa niin sanotut kriittiset pisteet ja mahdollisesti ennakoida esimerkiksi järjestelmän liiallinen kuormittuminen, sekä ennaltaehkäistä mahdolliset vikatilanteet ja suorituskyvyn heikkeneminen.

CHA System Performance Monitoring Dashboard-työkalu on GE Healthcaren tarjoama monitorointityökalu heidän tarjoamansa Centricity High Acuity-potilastietojärjestelmän suorituskyvyn seurantaan.

2 OPINNÄYTETYÖN TAVOITE, KUVAUS JA RAJAUS

Nykypäivänä erilaisilla tietojärjestelmillä on yhä suurempi rooli eri aloilla toimivien yritysten jokapäiväisessä toiminnassa. Mikäli esimerkiksi sairaalaverkon tietojärjestelmä pettäisi, voisi se pahimmillaan estää oikeanlaisen hoidon antamisen sitä tarvitseville potilaille. Marraskuussa 2017 Helsingin yliopistollisen sairaalan tietojärjestelmässä ilmeni ongelmia, jonka myötä sairaalahenkilökunnan työnteko vaikeutui ja potilaita koskevia tietoja jouduttiin katsomaan papereista, joita saatiin tulostettua, tai joita potilailla oli mahdollisesti mukanaan (Salminen ja STT, 2017). Vastaavanlaisten tilanteiden ehkäisemiseksi tulee tietojärjestelmiä toimittavien yritysten kiinnittää erityistä huomiota toimittamiensa tuotteiden laatuun ja toimivuuteen, jonka myötä tietojärjestelmien seuranta ja tarkkailu on tärkeää.

Opinnäytetyön toimeksiantajana toimii GE Healthcare Finland Oy. Kyseinen yritys on kehittänyt Suomessa potilasmonitoreita, anestesian antoon liittyviä ratkaisuja ja terveydenhuollon tietojärjestelmiä jo yli 45 vuoden ajan. Suomessa yrityksellä on kaksi toimipistettä, jotka sijaitsevat Helsingissä ja Kuopiossa. Yritys työllistää Suomessa noin 800 ihmistä (GE Healthcare, 2018).

Työn aiheena on CHA System Performance Monitoring Dashboard-työkalun kehittäminen. Kyseinen työkalu on GE Healthcaren tarjoama ratkaisu sairaalaverkon sisällä toimivien tietojärjestelmien suorituskykyyn liittyvän datan keräämiseen ja visualisointiin, jonka avulla sairaaloiden tietojärjestelmien ylläpitäjät voivat helposti monitoroida järjestelmän toimintaa. Työkalu itsessään koostuu useammasta avoimeen lähdekoodiin perustuvasta ohjelmistosta.

Työn tavoitteena on perehtyä tietojärjestelmien datan keräämiseen, jäsentämiseen ja visualisointiin, jotta työkalun avulla voitaisiin tunnistaa niin sanotut kriittiset pisteet ja mahdollisesti ennakoida esimerkiksi järjestelmän liiallinen kuormittuminen, sekä ennaltaehkäistä mahdolliset vikatilanteet ja suorituskyvyn heikkeneminen.

Työ aloitetaan perehtymällä aiheeseen liittyvään teoriaan eli visualisointiprosessin hahmottamiseen, monitoroinnin taustoihin ja työn kohteena olevaan tietojärjestelmään. Lopuksi siirrytään monitoroinnissa käytettyjen tekniikoiden käsittelyyn ja työkalun päivittämiseen. Työkalun päivittämistä havainnollistetaan muodostamalla esimerkkivisualisointi tietojärjestelmän keskeisimmistä monitoroitavista kohteista.

On huomioitavaa, että työn tavoitteisiin ei sisälly monitorointityökalussa käytettyjen tekniikoiden vaihtaminen tai vastaavien tekniikoiden vertailu. Opinnäytetyön aikana käytetyt työkalut tullaan kuitenkin päivittämään uusimpiin versioihinsa.

3 VISUALISOINTI

3.1 Data, informaatio ja tieto

Visualisoinnin tehtävänä on välittää katselijalle informaatiota. Tämä informaatio on tietoa, joka koostuu visualisoinnissa käytetystä datasta. Teoksessaan "*Tietovarantojen hyödyntäminen ja demokratia*" Kuronen määrittelee datan seuraavasti: "Data on vaihteleva kokoelma koodeja, merkkejä, numeroita, pulsseja ja signaaleja. Dataan ei välttämättä liity mitään merkitystä; se ei ehkä ole millään keinolla tulkittavissa informaatioksi ja omaksuttavissa tiedoksi." (Kuronen 1998, 8). Dataksi luetaan siis esimerkiksi bitit, painetun tekstin kirjaimet ja puheen äänteet.

Informaatio voidaan mieltää jalostettuna datana. Siihen liittyy tai siihen on liitettävissä jonkinlainen merkitys tai tulkinta. Hyvä esimerkki informaatiosta on laitteiden sisäiset signaalit, jotka ohjaavat laitteiden toimintaa halutulla tavalla. Informaatiota ei tule kuitenkaan sekoittaa keskenään tietoon. Tiedolla tarkoitetaan yleisesti informaatiota, jonka ihminen oppinut ja ymmärtänyt. Kokemuksen karttuessa, ihmisen omaksumat tiedot jalostuvat ymmärrykseksi (Kuronen 1998, 8).

3.2 Visuaalinen esitys

Visualisoinnilla tarkoitetaan jonkin asian esittämistä niin, että se voidaan havaita näköaistilla. Ominaisia keinoja visualisoinnin luomiselle ovat muun muassa kuvat, taulukot ja animaatiot. Esimerkiksi kartta toimii maaston ja paikkatiedon visualisoijana. Kosaran (2008-07-24) mukaan visualisointi voidaan määritellä seuraavalla tavalla:

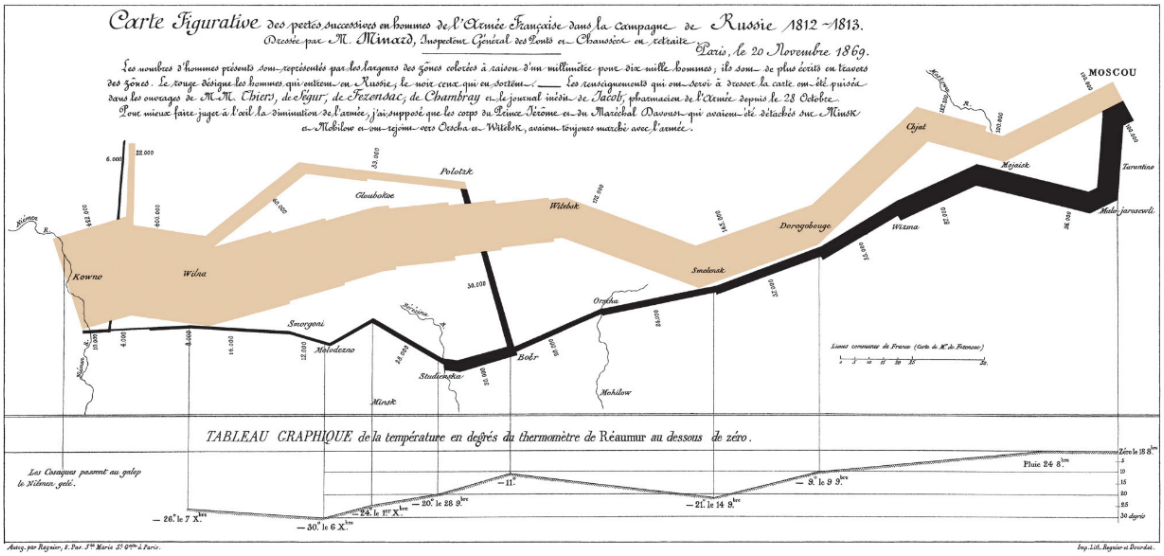
- Visualisointi perustuu dataan, joka ei ole näkyvissä ja josta ei voi tehdä suoria havaintoja. Visualisoinnin tarkoitus on muuntaa näkymätön data näkyväksi.
- Visualisoinnin tärkein tehtävä on luoda kuvallinen ilmaisu. Kuvallisen ilmaisun ohella voidaan käyttää myös tekstimuotoista tarkennusta visualisoitavasta datasta.
- Visualisoinnin tulee välittää tieto katsojalle tunnistettavalla ja ymmärrettävällä tavalla.

Visualisoinnin hyödyllisyys on täysin riippuvainen siitä, miten se esitetään. Kyseinen lause voi tuntua tyhmältä, mutta sitä se ei ole. Visualisoinnin välittämää tietoa voidaan käyttää hyödyksi vasta silloin, kun asianosaiset voivat tulkata ja ymmärtää, mitä se tarkoittaa. Jotkin visualisoinnit voivat olla hyvinkin hankalasti tulkittavissa. Ei siis riitä, että pelkästään visualisoinnin laatija tai hänen välittömään läheisyyteensä kuuluvat henkilöt osaavat tulkata sen. Tärkeintä on, että mahdollisesti jopa täysin ulkopuolinen henkilö kykenee tulkitsemaan ja ymmärtämään visualisoinnin välittämän tiedon merkityksen. Näin ollen visualisointi ei välitä katselijoilleen vääristynyttä tietoa, jonka ansiosta katselija saattaisi tehdä vääriä päätöksiä ymmärtämänsä tiedon perusteella.

Visualisoinnin edelläkävijä Edward Tufte määrittelee teoksessaan "*The Visual Display of Quantitative Information*" yhdeksän laatukriteeriä visualisoinnille (Tufte 2001, 13). Kaikkia Tuften esittämiä kriteerejä ei välttämättä pystytä soveltamaan kaikkiin visualisointeihin, mutta visualisointeja laadittaessa on syytä noudattaa niitä mahdollisimman tarkasti. Näiden laatukriteerien mukaan onnistuneen visualisoinnin tulee

- esittää data
- johtaa katselija ajattelemaan substanssia esitystavan sijaan
- välttää datan vääristämistä
- esittää suuri määrä tietoa pienellä alueella
- tehdä suuret datajoukot johdonmukaisiksi
- rohkaista katselijaa vertailemaan datajoukon eri osia
- esittää data usealla tasolla, sekä suurpiirteisesti että yksityiskohtaisesti
- olla tarkoituksenmukainen
- nitoutua yhteen datajoukon tilastollisten ja sanallisten kuvausten kanssa.

Kirjassaan Tufte esittelee ranskalaisen Charles Minardin vuonna 1896 laatiman visualisoinnin (kuva 1), joka kuvastaa Napoleonin armeijan etenemistä Venäjällä vuosina 1812-1813. Visualisoinnissa on kuusi eri muuttujaa; joukkojen määrä, joukkojen etenemissuunta, joukkojen sijainti tietyllä ajanhetkellä, kuljettu matka, maantieteelliset koordinaatit sekä lämpötila. Kyseistä visualisointia pidetään yhä yhtenä kaikkien aikojen parhaimmista, ja Tufte itse kommentoi visualisointia kirjassaan sanoin "Se voi hyvinkin olla paras koskaan piirretty tilastollinen grafiikka." (Tufte 2001, 40)



Kuva 1 - Charles Minardin visualisointi vuodelta 1869 (Tufte 2001, 41).

Visualisoinnin laatija voi käyttää eräänlaista tarkistuslistaa varmistamaan visualisoinnin onnistumisen. Mikäli kaikki tarkistuslistan kohdat on otettu huomioon visualisoinnin laadinnassa, on lopputuloksena hyvin todennäköisesti onnistunut visualisointi. Ennen visualisoinnin laatimisen aloittamista tulee varmistaa seuraavat asiat:

1. Kohderyhmä. Visualisointi täytyy suunnata tietylle kohderyhmälle. On ensisijaisen tärkeää havainnollistaa kohderyhmän kyky ymmärtää esitystä, eikä sitä tule ali- tai yliarvioida.
2. Kuvion rooli. Kuvio voi esimerkiksi havainnollistaa tai tutkia käsiteltävää kohdetta, tuoda esiin monimutkaisia riippuvuuksia tai keventää muuten ehkä liian raskaslukuista kokonaisuutta.
3. Kuviotyyppi. Esitettävien tietojen luonne, esitystilanne ja kohdeyleisö vaikuttavat valittavaan kuviotyyppiin.
4. Missä ja miten kuvio esitetään. Esitysväline ja -tilanne vaikuttavat siihen, millainen kuvio pitäisi tehdä. Esimerkiksi kaksiväriseen tutkimusraporttiin ei välttämättä sovi värillisessä esityksessä käytetty esitysmuoto.
5. Kuvion koko. Kuvion fyysinen koko tulee suhteuttaa kuviossa käytetyn datan ja sen yksityiskohtien määrään.
6. Vaihtoehtoiset esitysmuodot. On syytä tarkastella, sopisiko esimerkiksi taulukko tai tekstimuotoinen esitys paremmin tiedon välittäjäksi kyseiselle visualisoinnille.
7. Työvälineet. Erilaiset esitysmenetelmät ja -tekniikat vaikuttavat oleellisesti visualisoinnin lopulliseen toteutukseen. Esimerkiksi tietokoneen avulla luotavaan visualisointiin vaikuttavat käytettävissä olevat ohjelmistot ja tulostinlaitteet (Vesala 2000, 200).

Valmistuneelle visualisoinnille on omanlainen tarkistuslista. Kaikkia tarkistuslistan kohtia on syytä pohtia visualisoinnin kohderyhmän näkökulmasta, ja se koostuu seuraavista asioista:

1. Kuvion välittämä tieto ja totuudenmukaisuus. Kuvion tarkoituksena on välittää tietoa jostain asiasta. Mikäli laadittu kuvio ei onnistu välittämään kyseistä tietoa tai sen välittämä tieto on vääristynyttä, on sen perimmäinen tarkoitus hukattu.
2. Helppolukuisuus. Laaditun kuvion tulee välittää tieto katselijalle nopeasti ja selkeästi.
3. Kuvion koko ja muoto. Jossain tilanteissa on mahdollista vaikuttaa käytetyn kuvion kokoon, muotoon ja tilaan, kun taas toisissa tilanteissa kuvion koko, muoto ja tila voivat olla ennalta määrättyjä. Mikäli esitykseen sisältyy muita samantapaisia kuvioita, voi olla tarpeen muokata kaikki niistä kuvioista samankokoisiksi ja -muotoisiksi.
4. Kuvion sijainti. Mikäli kuvio on osa esimerkiksi raporttia tai artikkelia, tulisi sen sijaita mahdollisimman lähellä siihen liittyvää tekstin osaa.
5. Kuvion testaus. Ihmiset ovat monesti sokeita omien teelmiensä virheille. Näin ollen on erittäin tervetullutta pyytää esimerkiksi kollegaa arvioimaan kuvio ja sen onnistuminen tiedon välittäjänä.
6. Kokonaisuuden tarkastelu. Vaikka visualisointi olisi teknisesti moitteeton, voi se olla sisällöllisesti käsittämätön tai harhaanjohtava. Lopuksi on syytä tarkastella visualisointia laajemmasta perspektiivistä ja arvioida visualisoinnin välittämän tiedon tarkoitusta osana kokonaisuutta (Vesala 2000, 201).

3.3 Visualisoinnin muotoja

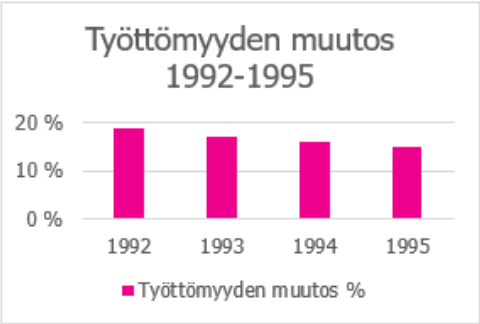
Tietoa voidaan yleisesti esittää usealla eri tavalla ja monessa eri muodossa. Näitä ovat esimerkiksi teksti, taulukko tai kaavio. Nämä eri tavat käyttäytyvät hieman eri tavoin ensinnäkin siksi, että niillä voidaan esittää erilaisia asioita ja toiseksi siksi, että niiden välittämä tieto saavuttaa katselijan eri tavoin. Näin ollen visualisointeja suunniteltaessa valinnat käytetyistä visualisoinnin muodoista tulee tehdä tiedostetusti eri esitysvälineiden välillä.

Teksti antaa visualisoinnin laatijalle vapauden tuoda esiin mittavia suhteita eri asioiden välillä. Kirjoitettu teksti ei anna lukijalleen mahdollisuutta arvioida tekstin välittämää tietoa, vaan se pohjautuu ajatukseen, jossa lukijaa vaaditaan uskomaan tekstin kirjoittajaa. Tekstin avulla visualisoinnin laatija voi ilmaista välitettävän tiedon monella eri tavalla, esimerkiksi käyttäen erilaisia tunnepitoisia ilmaisuja tai adjektiiveja apuna tiedon välityksessä.

Taulukkoa voidaan mieltää kaikista kattavimpana tiedon esittämisen muotona. Sen avulla on mahdollista esittää paljon tarkkaa tietoa, esimerkiksi numeerisia arvoja. Päällimmäisimpänä havaintona taulukkoa luettaessa ovatkin yksittäiset tietoalkiot, eikä suinkaan taulukon esittämien arvojen heijastama kokonaisuus tai yksittäisten tietoalkioiden väliset suhteet. Taulukko onkin siinä mielessä huono esittämään säännönmukaisuuksia ja riippuvuuksia, eikä siitä ole helppo havaita esimerkiksi trendiä. Taulukkoa tutkiskellessa katselija joutuukin usein tekemään erinäisiä päässälaskuja hahmottaakseen eri tietoalkioiden väliset suhteet.

Kaavioita käytettäessä tieto esitetään katselijalle usein erilaisten symbolien, esimerkiksi pylväiden avulla. Käytettyjen symbolien suuruussuhteet herättävät katselijan silmissä mielleyhtymiä, jotka voidaan tulkita määrinä. Näin ollen kaavio välittää tiedon katselijan mieleen nopeasti ja tehokkaasti. On kuitenkin huomioitava, että väärin ymmärrettynä kaavio voi herättää katselijalle vääristyneitä mielleyhtymiä kaavion välittämästä tiedosta (Vesala 2000, 9-14).

Kuviossa 1 näiden kolmen eri esitysmuodon eroavaisuuksia on havainnollistettu käyttämällä kuvitteellista esimerkkiä työttömyyden muutoksista vuosien 1992 ja 1995 välillä.



Työttömyyden muutos 1992-1995

1992	19 %
1993	17 %
1994	16 %
1995	15 %

Työttömyys on laskenut jyrkästi vuodesta 1992 vuoteen 1995.

Kuvio 1 – Esimerkki visualisointimuotojen eroavaisuuksista.

3.4 Data-mustesuhde

Visualisoinnin tärkeimpänä tehtävänä on kiinnittää katselijan huomio visualisoinnissa käytetyn datan sisältöön ja sen merkitykseen, eikä muihin merkityksettömiin asioihin. Tätä merkityksellisten ja merkityksettömien asioiden välistä suhdetta voidaan kuvastaa data-mustesuhteen avulla, jossa suhteutetaan informaatiota esittävän musteen määrä visualisoinnin kokonaismusteen määrään:

$$data - mustesuhde = \frac{datamusteen\ määrä}{kokonaismusteen\ määrä}$$

Mitä suurempi kyseisestä data-mustesuhteesta muodostuu, sitä onnistuneempana visualisointia voidaan mieltää (Tufte 2001, 93). Tällöin suurin osa jäljelle jäävästä musteesta kuvastaa itse tietoa eikä mitään muuta, joka voisi vääristää tai haitata tiedon välittämistä. Näin ollen visualisointeihin sisällytettyä koristeellisuutta tulee välttää, sillä se ei tuo visualisoinnille lainkaan lisäarvoa eikä se välitä katselijalle informaatiota itse datasta. Tietoa välittämättömät visualisoinnin elementit mielletään usein kuvioroinaksi.

3.5 Kuvioroina

Visualisointiin sisällytetty koristeellisuus muodostaa paljon turhaa mustetta, jota voidaan mieltää visuaaliseksi rihkamaksi tai kuvioroinaksi. Koristeellisuuden tarkoitus vaihtelee – sen avulla voidaan esimerkiksi tehdä visualisoinnista tieteellisemmän ja tarkemman näköinen, elävöittää visualisointia tai antaa visualisoinnin suunnittelijalle mahdollisuus kehittää taiteellisia taitojaan. Kuvioroinan kolme yleisintä tyyppiä ovat:

- Tahaton optinen taide: Tietynlaiset toistuvat kuviot voivat saada aikaan moiré-ilmiön. Kyseessä on interferenssikuvio, jonka ansiosta kuvaan muodostuu häiritsevä illuusio väreilystä.
- Levottomuutta herättävä ruudutus: Taulukoissa usein apuna käytetty ruudutus ei saa sekoittua itse tietoa esittävän sisällön kanssa, ja ruudutus tulee pitää niin maltillisena kuin vain mahdollista.
- Itsetarkoituksellinen grafiikan korostaminen: Mikäli visualisoinnin graafinen näyttävyys asetetaan etusijalle, voi visualisoinnin välittämä informaatio jäädä taustalle taiteellisuuden kustannuksella (Tufte 2001, 107-121).

On myös henkilöitä, jotka puhuvat kuvioroinan puolesta ja uskovat sen olevan joissain tilanteissa paikallaan. Amerikkalaisessa Time-uutislehdessä pitkään grafiikkaohjaajana työskennelleen Nigel Holmesin mukaan tietoa esittävän grafiikan tulisi houkutella lukijan mielenkiinto visuaalisella tavalla tai olla joissain tilanteissa jopa humoristista. Holmes toteaa, että esimerkiksi tilastografiikassa huumorin käyttö voi auttaa lukijaa muistamaan esityksen välittämän ydinsanoman paremmin (Cairo 2013, 69).

4 TIETOJÄRJESTELMÄN SUORITUSKYKY

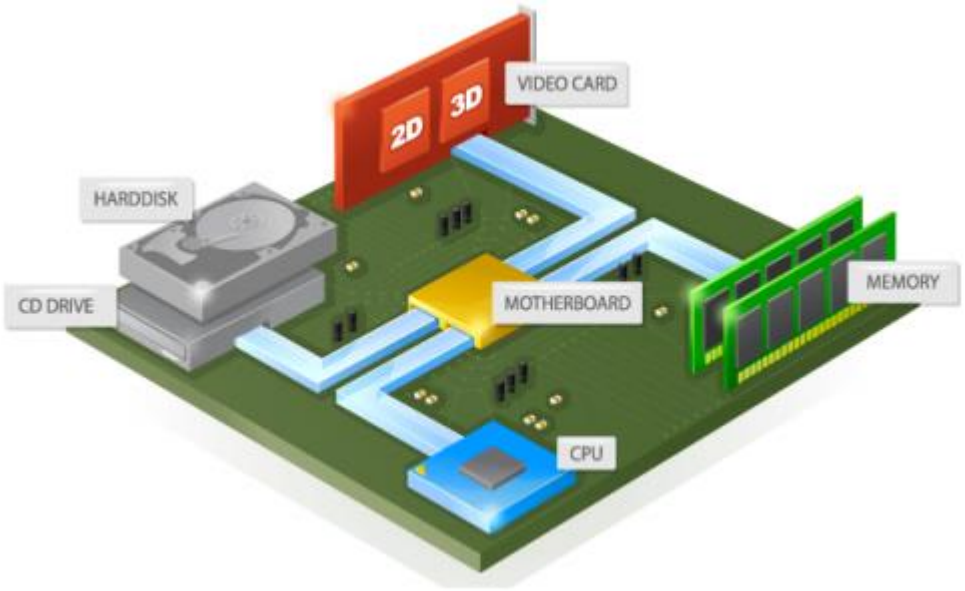
Ennen perehtymistä opinnäytetyön kohteena olevaan monitoroitavaan tietojärjestelmään ja sen monitoroinnissa käytettäviin tekniikoihin, on syytä käsitellä tietojärjestelmien suorituskykyä omana kokonaisuutenaan paremman kokonaiskuvan hahmottamiseksi.

Yleisesti ottaen tietojärjestelmän suorituskyvyllä tarkoitetaan tietojärjestelmän kykyä suorittaa sille annettuja tehtäviä (Allen 1994, 1). Korkeaan tietojärjestelmän suorituskykyyn liittyy kontekstista riippuen erilaisia asioita, kuten esimerkiksi lyhyet vasteajat, resurssien matala käyttöaste, korkea suoritusteho sekä tietojärjestelmän korkea saatavuus. Tietojärjestelmän käyttämistä tai tarjoamista ohjelmistoista puhuttaessa tärkeäksi elementiksi nousevat ohjelmistojen vasteajat, jotka toimivat keskeisinä järjestelmän käyttökokemuksen mittareina. Kyseiset vasteajat voivat mitata esimerkiksi aikaa, joka kuluu jonkin toisen näyttöikkunan avaamiseen tai jonkin palvelupyynnön suorittamiseen.

Suorituskyvyn valvonnalla eli monitoroinnilla tarkoitetaan tietojärjestelmän käytössä olevien resurssien valvontaa. Näitä resursseja ja muita laitteiston suorituskykyyn liittyviä muuttujia tarkastelemalla on mahdollista saada kattava kokonaiskuva tietojärjestelmän yleiskunnosta ja toiminnasta. Monitorointi tarjoaa kuitenkin myös paljon muuta kuin vain pelkkää teknistä tietoa. James Turnbull (2016, 1.) toteaa, että monitorointi tarjoaa käännöksen yrityksen liikearvon sekä järjestelmien ja sovellusten tuottamien tietojen välillä. Tietojärjestelmän monitorointiratkaisu muuntaa näiden järjestelmien ja sovellusten tuottaman tiedon mitattavaksi käyttäjäkokemukseksi. Tämä käyttäjäkokemus puolestaan antaa yritykselle palautetta siitä, vastaavatko sen tarjoamat tuotteet ja palvelut asiakkaiden toiveita ja tarpeita. Samainen käyttäjäkokemus tarjoaa myös yrityksen IT-osastolle tietoa siitä, mitkä osat järjestelmässä toimivat ja mitkä eivät toimi odotetulla tavalla. Voidaan siis todeta, että monitorointi palvelee kahta asiakasta: yritystoimintaa ja yrityksen IT-osastoa (Turnbull 2016, 1.).

4.1 Pääasialliset resurssit

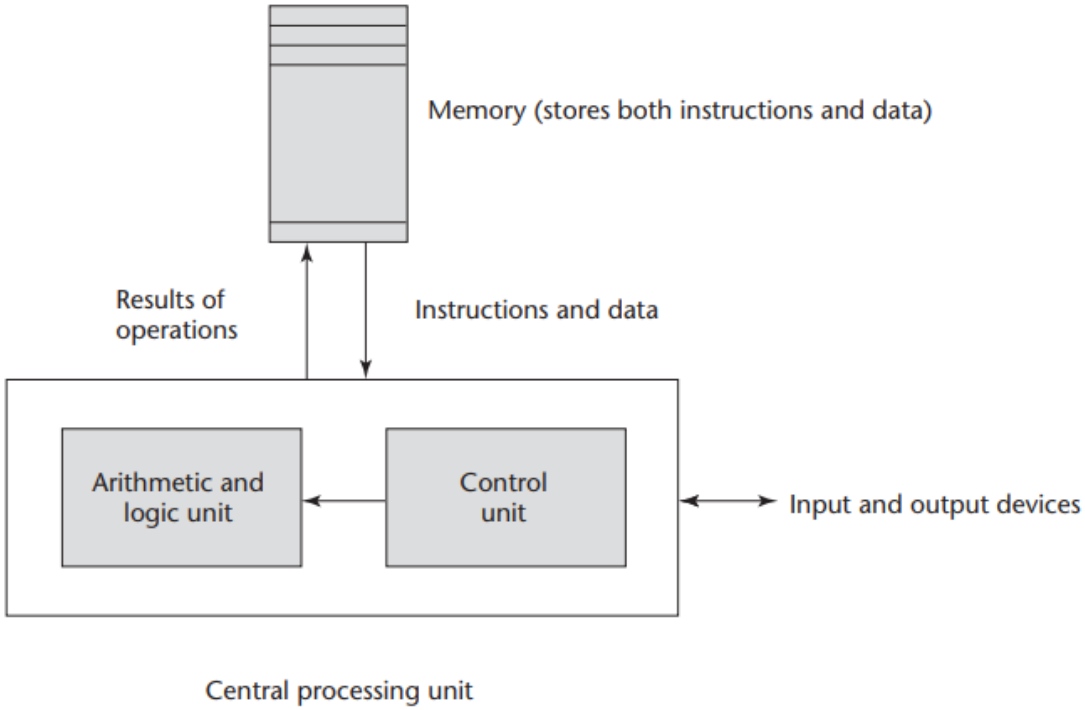
Suorituskykyyn vaikuttavat oleellisesti tietojärjestelmän käytössä olevat resurssit, jotka voidaan jakaa neljään pääkategoriaan prosessorin, muistin, massamuistin sekä verkkoyhteyden välillä. Tietokoneisiin on yleensä liitetty muitakin erilaisia komponentteja ja laitteita, kuten esimerkiksi virtalähde ja näppäimistö, mutta niillä ei ole varsinaista roolia tietokoneen suorituskyvyn kannalta. Esimerkiksi virtalähde nimensä mukaisesti tarjoaa virran koneeseen, eikä varsinaisesti osallistu ohjelmien tai annettujen tehtävien suorittamiseen. Edellä mainittuihin kategorioihin kuuluvia komponentteja ja niiden suorituskyvyn mittareita tarkastelemalla on mahdollista saada kattava kokonaiskuva tietojärjestelmän tilasta.



Kuva 2 - Havainnekuva tietokoneen keskeisimmistä komponenteista.

Optimaalisen suorituskyvyn takaamiseksi järjestelmän komponenttien välille ei saisi syntyä niin sanottua pullonkaulaa. Pullonkaula on ilmiö, jossa jokin tietojärjestelmän komponentti tai osa-alue estää järjestelmää toimimasta täydellä kapasiteetilla. Käsitteen nimi viittaa nesteen säilyttämiseen tarkoitettuun pulloon, jonka muoto kapenee kaulaa kohden ylöspäin aina pullon suuhun asti. Pullon suu määrittelee nesteen virtausnopeuden, mikäli pullossa oleva neste kaadettaisiin pois. Yleisiä esimerkkejä pullonkauloista tietotekniikassa ovat nopean suorittimen ja hitaan näytönohjaimen yhdistelmät graafisesti vaativissa sovelluksissa, joissa suorittimen laskentateho ylittää näytönohjaimen tarjoaman laskentatehon. Kyseisessä tilanteessa suoritin kykenee suoriutumaan sille annetuista tehtävistä nopeasti, kun taas näytönohjain hidastaa komponenttien yhteistyötä, jolloin järjestelmä ei kykene toimimaan täydellä kapasiteetilla. Toisena esimerkkinä suuria tietomääriä käsiteltäessä toimii moderni järjestelmä, jossa on vanha ja hidas kiintolevy, joka ei kykene vastaamaan laitteiston muiden komponenttien nopeuteen. Pullonkauloja voi siis ilmetä lähes kaikissa tietojärjestelmissä käytetyissä komponenteissa.

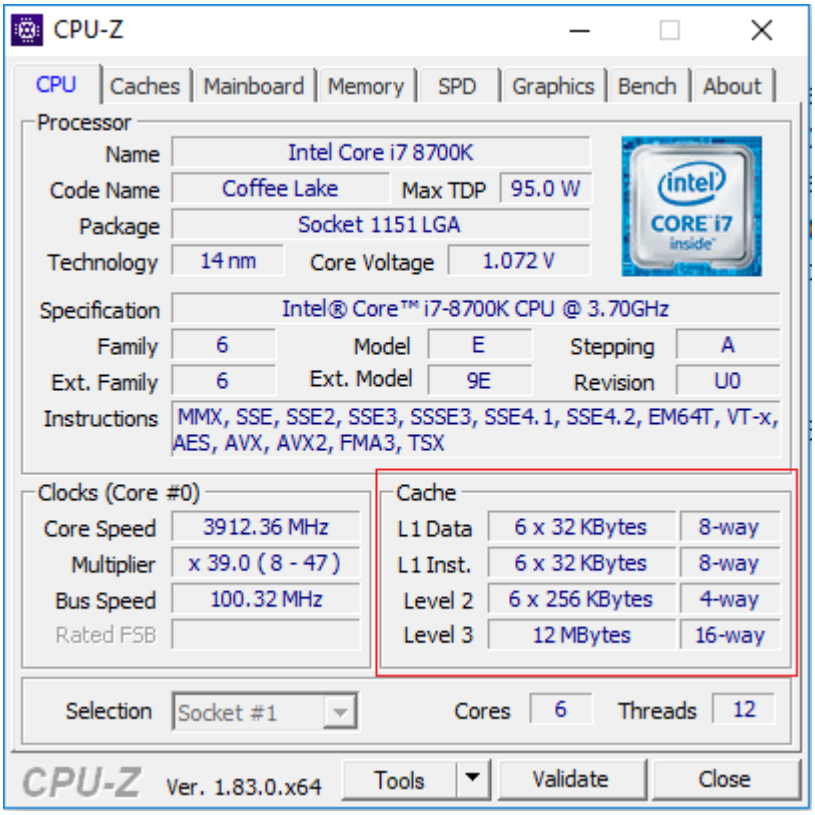
Yksi tunnetuimmista pullonkauloista tietojärjestelmissä on niin sanottu Von Neumannin pullonkaula. Kyseisessä pullonkaulassa on kyse suorittimen ja muistin välisestä tiedonsiirtoväylästä, joka rajoittaa suorittimen ja muistin välisen tiedonsiirtonopeuden verrattuna muistin määrään, ja se on ollut yksi tärkeimmistä motivaattoreista rinnakkaislaskennan tutkimuksessa ja kehityksessä (Sebesta 2012, 27). Pullonkaula pohjautuu von Neumannin arkkitehtuuriin, joka on nimetty yhden sen keksijän, John von Neumannin mukaan. Von Neumannin arkkitehtuuriin perustuvassa tietokoneessa sekä data että ajettavat ohjelmat sijaitsevat samassa muistiavaruudessa. Tietokoneen suoritin on eristetty muistista, jolloin käskyt ja data tulee siirtää väylien avulla muistista suorittimelle ja päinvastoin. Lähestulkoon kaikki 1940-luvun jälkeen rakennetut tietokoneet ja tietojärjestelmät perustuvat von Neumannin arkkitehtuuriin (Sebesta 2012, 18).



Kuvio 2 – Von Neumannin arkkitehtuuri (Sebesta 2012, 19).

Von Neumannin pullonkaulan lieventämiseksi on olemassa useita keinoja, joista kenties keskeisimmässä roolissa on ollut välimuistin lisääminen suorittimen ja muistin välille. Lähes kaikissa nykyajan suorittimissa on välimuistia usealla tasolla, jotta usein käytettyihin tietoihin päästään nopeammin käsiksi. Lähimpänä suorittimen ydintä oleva välimuisti (L1) on kaikista nopein, mutta samalla pienin kooltaan ja kauimmaisena suorittimen ytimestä sijaitseva välimuisti (L3) on kaikista hitain, mutta samalla suurin kooltaan.

Kuvassa 3 on esitelty Intelin i7 8700K-suorittimen perustietoja, jossa on nähtävillä kyseisen suorittimen välimuistin hierarkia. Kyseisen suorittimen välimuisti on jaettu tasoihin L1, L2 ja L3. Kuten monissa muissakin nykypäivän suorittimissa, L1-tason välimuisti on jaettu keskenään omiin sektioihinsa L1d (data) sekä L1i (instructions). L1d-tason välimuisti on datalle omistettu välimuisti, kun taas L1i-tason välimuisti on omistettu suoritettaville käskyille. Suoritin sisältää kuusi ydintä, joilla jokaisella on oma L1- ja L2-tason muisti. Matalimman tason L3-muisti on jaettu kaikkien ytimien kesken.



Kuva 3 – Intel i7 8700K-suorittimen perustiedot.

Apica AB:n perustajajäsen Sven Hammar listaa APMDigestin julkaisemassa artikkelissa viisi yleisintä syytä pullonkaulailmiön syntymiselle. Apica on yksi markkinoiden johtavista sovellusten suorituskyvyn testaamiseen kohdistuvista yrityksistä, joka tarjoaa asiakkailleen erilaisia ratkaisuja sovellustensa suorituskyvyn testaamiseen (Apica, 2018).

1. Suorittimen käyttö. Suorittimeen kohdistuvat pullonkaulat syntyvät silloin, kun suoritin on niin kiireinen, ettei se kykene suoriutumaan tehtävistään ajallaan.
2. Keskusmuistin käyttö. Keskusmuistiin kohdistuvat pullonkaulat viittaavat joko liian alhaiseen keskusmuistin määrään tai nopeuteen. Mikäli järjestelmän tarvitsema keskusmuistin määrä ylittää käytettävissä olevan keskusmuistin kapasiteetin, alkaa järjestelmä purkaa keskusmuistin sisältöä huomattavasti hitaammalle kiintolevylle, jolloin järjestelmän käyttö hidastuu takuulla.
3. Verkkoyhteyden käyttö. Pullonkaulat verkkoyhteydessä muodostuvat useimmiten silloin, kun verkkoyhteyden kaistanleveys ei riitä jouhevaan verkkoliikenteeseen useamman laitteen kanssa. Tällöin jokin laite joutuu väistämättä odottamaan vuoroaan, esimerkiksi verkkokaupan palvelimen ruuhkautuessa alennusmyyntien aikaan.
4. Ohjelmistoon liittyvä rajoite. Huonosti optimoidut ohjelmistot voivat itsessään muodostaa pullonkaulan, esimerkiksi lataamalla ja/tai käyttämällä tietokannan tai järjestelmän resursseja ei-optimaalisella tavalla.
5. Levyn käyttö. Nykypäivänä ylivoimaisesti hitain komponentti tietokoneissa on kiintolevy (HDD tai SSD). Kiintolevyillä on omat fyysiset rajoitteet sille, kuinka nopeasti tietoa voidaan lukea tai tallentaa, joka voi helposti muodostaa pullonkaulan (Hammar ja APMDigest, 2017).

Tarkastellaan seuraavaksi kategorioittain tietojärjestelmän eri osa-alueita ja niiden vaikutuksia tietojärjestelmän toimintaan. Käydään samalla läpi kyseisten osa-alueiden tyypillisimpiä pullonkauloja sekä suorituskyvyn valvonnan mittareita. Kyseisen osion jälkeen meillä on hyvä perusta monitoroinnin aloittamiselle, mutta ennen sitä on syytä tutustua monitoroinnin teoriaan omana kokonaisuutenaan.

4.1.1 Prosessori

Prosessori (CPU) on tietokoneen ydin, joka suorittaa tietojärjestelmän antamia konekielisiä käskyjä. Nämä käskyt voivat olla esimerkiksi laskutoimituksia tai ohjaus- tai I/O-operaatioita. Prosessoria itseään ei tule sekoittaa grafiikkaprosessoriin (GPU). Grafiikkaprosessori on erillinen mikroprosessori, jonka pääasiallisena tehtävänä on suorittaa 2- ja 3-ulotteisen grafiikan renderointia. On kuitenkin olemassa mikropiirejä, joissa sekä prosessori ja grafiikkaprosessori on yhdistetty samalle mikropiirille. Näihin lukeutuvat esimerkiksi AMD:n APU-sarjan prosessorit sekä Intelillä prosessorit, jotka on varustettu Intel HD Graphics-mikropiirillä. Grafiikkaprosessoria ei tulla käsittelemään tässä työssä sen laajemmin, sillä monitoroitavassa tietojärjestelmässä ei ole suurta graafista laskentatehoa vaativia osa-alueita.

Yksi yleisimmistä prosessorin suorituskyvyn valvonnan mittareista on suoritinaika. Suoritinajalla tarkoitetaan prosessorin käyttämää aikaa jonkin toiminnon suorittamiseen, ja se useimmiten ilmaistaan prosenttilukuna prosessorin kapasiteettiin verraten. Mikäli suoritinaika on korkea, esimerkiksi yli 70 %, voi tietokone toimia hitaasti sillä prosessori ei kykene suoriutumaan sille annetuista tehtävistä tarpeeksi nopeasti.

Useat erilaiset laitteet, kuten esimerkiksi kiintolevyt, verkkoliitännän ohjaimet ja oheislaitteet lähettävät prosessorille keskeytyssignaaleja, kun ne tarvitsevat prosessorin suoritinaikaa. Tällöin prosessorin senhetkinen ohjelman suoritus keskeytyy ja prosessori suorittaa keskeytyssignaalin lähettäneen laitteen pyynnön, jonka jälkeen se jatkaa aiemmin keskeyttämänsä ohjelman suorittamista. Windows-ympäristöissä on käytössä myös niin kutsutut DPC-keskeytykset, jotka mahdollistavat matalamman prioriteetin omaavien tehtävien suorituksen lykkäämisen (Microsoft TechNet, 2018). Keskeytykset ovat siis täysin normaaleja asioita tietokoneiden keskuudessa, mutta jos suuri osa prosessorin suoritinajasta kuluu erilaisten keskeytysten käsittelyyn, voi se viitata ongelmaan tietokoneen laitteistossa tai laiteajureissa. Suoritinajan tapaan, keskeytysaika ilmaistaan useimmiten prosenttilukuna.

Rinnakkain suoritettaville ohjelmille on olennaista, että prosessori kykenee tallentamaan suoritettavan ohjelman tilan muistiin, jotta sen suorittamista voidaan jatkaa myöhemmin. Kontekstin vaihdolla tarkoitetaan tilannetta, jossa prosessori vaihtaa suoritettavaa säiettä, samalla tallentaen edeltävän suorituksen tilan myöhempää suoritusta varten. Kontekstin vaihto on mahdollista suorittaa esimerkiksi edellä mainittujen keskeytyssignaalien avulla. Seuraamalla kontekstin vaihtojen määrää voi päätellä, käyttääkö tietokone prosessoria tasaisesti kaikkien tehtävien suorittamiseen. Suuri kontekstin vaihtojen määrä viittaa usein siihen, että liian moni säie kilpailee käytettävästä suoritinajasta, jolloin järjestelmä kuormittuu (Microsoft TechNet, 2008).

4.1.2 Muisti

Tietokoneen muistilla viitataan tietokoneen fyysiseen keskusmuistiin (RAM). Keskusmuisti toimii tietokoneen lyhytkestoisena työmuistina, jonne säilötään kaikki suoritettavat ohjelmat ja niiden tarvitsemat tiedot. Keskusmuistin jatkeena käytetään usein virtuaalista näennäismuistia, joka koostuu sivutustiedostosta tietokoneen pitkäaikaisessa muistissa, esimerkiksi kiintolevyllä. Näennäismuisti mahdollistaa sivutuksen, joka tarkoittaa keskusmuistissa sijaitsevien tietojen siirtämistä pitkäaikaiseen muistiin, vapauttaen tilaa muille keskusmuistin vaatimille tiedoille. Sivutuksen tarkoituksena on sivuttaa sellaista tietoa, mikä ei ole kyseisellä hetkellä järjestelmän käytössä. Mikäli järjestelmä tarvitsee jotain sen aikaisemmin sivuttamaa tietoa, tulee sen noutaa tieto takaisin sivutustiedostosta keskusmuistiin näennäismuistin mahdollistamien virtuaaliosoitteiden avulla, jotka tietokoneen prosessori kykenee muuntamaan fyysisiksi osoitteiksi.

Microsoft kutsuu vastaavanlaisia tilanteita sivuvirheiksi (Microsoft, 2018). Sivuvirheet ovat normaali tapahtuma näennäismuistia hyödyntävässä järjestelmässä, mutta liiallinen määrä niitä voi olla haitallista järjestelmän suorituskyvylle. Useimmiten tämä näkyy tietojärjestelmän kiintolevyjen tai SSD-levyjen I/O-tapahtumien kasvuna. Sivuvirheiden määrä sekunnissa onkin yksi muistin suorituskyvyn valvonnan mittareista.

Yleisin suorituskyvyn valvonnan mittari muistille on käytettävissä olevan keskusmuistin määrä. Mikäli kaikki käytössä oleva keskusmuisti on käytetty eikä keskusmuisti riitä kaikille ajettavilla ohjelmille, alkaa tietokone turvautua niin sanottuun heittovaihtoon (engl. swapping). Heittovaihdossa järjestelmään kohdistuu suuri määrä keskeytyksiä johtuen virtuaalimuistijärjestelmästä. Nämä keskeytykset käynnistävät muistin siirron pitkäaikaisesta muistista sijaitsevasta sivutustiedostosta keskusmuistiin ja päinvastoin, ruuhkauttaen koko järjestelmän (Haikala ja Järvinen, 4.).

Heittovaihdon voi aiheuttaa myös muistivuoto, jossa suoritettava ohjelma ei vapauta varaamaansa muistia silloin, kun sitä ei enää tarvita. Mikäli ohjelmaa tai sen muistivuotoa sisältämää osaa suoritetaan tarpeeksi kauan, voi käytettävissä oleva keskusmuisti loppua. Vastaavia tilanteita voidaan estää roskankeruulla (engl. garbage collection), jossa pyritään poistamaan muistista tietoja joihin ei tulla enää viittaamaan (Jones, Hosking ja Moss 2012, 1-2).

4.1.3 Massamuisti

Massamuistilla tarkoitetaan pääsääntöistä tallennustilaa, johon ohjelmat ja tiedostot tallennetaan. Nykypäivänä tietokoneissa käytetään massamuistina joko perinteistä kiintolevyä (HDD) tai SSD-levyä (SSD). Perinteisten kiintolevyjen toiminta perustuu ferromagneetismiin, jossa tieto tallennetaan kiintolevyllä sijaitseviin pyöriviin metalli- tai lasikiekkoihin, jotka on päällystetty ferromagneettisella aineella. SSD-levyillä tieto puolestaan tallennetaan levyllä sijaitseviin puolijohdepiireihin, esimerkiksi flash-muistipiireille.

Massamuisti on ylivoimaisesti tietokoneen hitain komponentti, ja se voi olla yksi syy järjestelmän hidastumiselle esimerkiksi aikaisemmin mainitun heittovaihdon tapahtuessa. Tarkasteltaessa koko tietokoneen muistihierarkiaa, on massamuisti pinon pohjimmaisena. Kappaleessa 4.1 mainitsin Von Neumannin pullonkaulan, jota on yritetty lieventää eri tasoisten välimuistien käytöllä tietokoneen suorittimen ja keskusmuistin välillä. Keskusmuistin toiminta voidaan samalla tavalla mieltää välimuistina suorittimen ja massamuistilaitteen välillä. Massamuisti sijaitsee kauimmaisena tietokoneen suorittimesta, jolloin se on kaikista tietokoneen muisteista hitain mutta suurin. Tutkijat havaitsivat tämän massa- ja välimuisteja koskevan ongelman jo vuonna 1946. Ideaalisessa tilanteessa tietokoneen muisti olisi kapasiteetiltaan rajattoman suuri ja nopea, jotta mikä tahansa tieto olisi hetkessä suorittimen käytettävissä. Tämä ei ole kuitenkaan fyysisesti mahdollista, jonka myötä meidän tulee hyväksyä muistihierarkia, jossa jokaisella hierarkian tasolla on suurempi kapasiteetti kuin edeltäjällään, mutta jotka ovat hitaammin käsiteltävissä (Burks, Goldstine ja Neumann 1946, 7).

Yleisin mittari tietokoneen massamuistin suorituskyvyn mittaamiselle on käytettävissä oleva tallennustilan määrä. Esimerkiksi tietokantapalvelimet tarvitsevat paljon tilaa, sillä ne voivat kasvaa hyvinkin nopeasti käyttötarkoituksesta riippuen. Mikäli tietokantapalvelimen tallennustila loppuisi, voisi se pahimmillaan johtaa tietokannan korruptoitumiseen. Muita yleisiä mittareita massamuistin suorituskyvyn mittaamiselle ovat luku- ja kirjoitusoperaatioiden määrä sekunneissa (IOPS) sekä hakuaika. Mitä korkeampi luku- ja kirjoitusoperaatioiden määrä sekunneissa on, sitä nopeammin laite kykenee siirtämään tietoa edestakaisin muiden laitteiden välillä. Hakuajalla tarkoitetaan massamuistilaitteen kuluttamaa aikaa tarvittavan tiedon löytämiseen.

Kiinto- ja SSD-levyjen sisään on rakennettu SMART-seurantajärjestelmä. Kyseinen järjestelmä mahdollistaa levyn tarkempien tietojen ohjelmallisen seurannan. SMART-seurantajärjestelmä mittaa muun muassa levyn lämpötilaa, uudelleenosoitettujen sektoreiden määrää, käynnistys- ja sammutuskertoja sekä käyttöikää. Koska kyseiset levyt poikkeavat toimintaperiaatteiltaan ja tekniikoiltaan toisistaan, on SMART-seurantajärjestelmän tarjoama luettelo levyjen tiedoista erilainen poiketen levyn tyypistä.

4.1.4 Verkkoyhteys

Laajemmista tietojärjestelmistä puhuttaessa oleelliseksi osaksi kokonaisuutta tulee myös verkko, johon järjestelmään sisällytetyt laitteet on kytketty. Esimerkiksi päätelaitteet ovat usein yhteydessä johonkin sovellus- tai tietokantapalvelimeen ja vikatilanteita voi muodostua, jos päätelaitteet eivät kykene kommunikoimaan sovelluspalvelimen kanssa. Myöskin huonot tai hitaat verkkoyhteydet näkyvät suoraan päätelaitteiden hitautena, mikäli kommunikointi esimerkiksi kyseisen sovellus- tai tietokantapalvelimen kanssa on hidasta tai verkkaista. Käyttäjät voivat näin ollen mieltää hitauden yksinomaan käyttämänsä päätelaitteen viaksi, vaikka vian ydin olisikin todellisuudessa huonossa verkkoyhteydessä.

Yleisimpiä verkon suorituskyvyn seurannan mittareita ovat kaistanleveys, läpisyöttö, latenssi ja sen vaihtelu sekä kadonneiden pakettien määrä. Kaistanleveys kertoo yhteyden teoreettisesti suurimman mahdollisen tiedonsiirtonopeuden, kun läpisyöttö on varsinainen tiedonsiirtonopeus. Kumpikin mittaus ilmoitetaan useimmiten bitteinä per sekunti. Latenssilla tarkoitetaan aikaa, mikä kuluu datapaketin edestakaiseen lähetykseen lähettäjän ja vastaanottajan välillä. Latenssin kasvaessa verkkoyhteyden responsiivisuus heikkenee, mikä heijastuu käyttäjälle yleisesti palveluiden hitautena. Latenssia tarkkailtaessa on hyvä huomioida myös latenssin vaihteluväli eli minimi- ja maksimiarvojen poikkeamat. Mikäli vaihteluväli on suuri, voi se aiheuttaa katkoksia esimerkiksi verkon yli käytävissä verkkopuheluissa. Kadonneiden pakettien määrä ilmoitetaan useimmiten prosentteina, ja se kertoo nimensä mukaisesti tiedonsiirron aikana kadonneiden pakettien määrän. Esimerkiksi verkon yli käytävissä puhekeskusteluissa satunnainen pakettien katoaminen ei vaikuta suuresti keskustelun laatuun. Sen sijaan häviöt, jotka ovat 5-10 % koko pakettivirrasta, vaikuttavat huomattavasti keskustelun laatuun (Antonakos ja Mansfield 2010, 501).

Verkossa olevien laitteiden välistä kommunikointia voidaan priorisoida QoS:n (Quality of Service) avulla. Priorisoinnin avulla tietoliikennettä voidaan hallinnoida niin, että esimerkiksi verkon tosiaikainen liikenne omaa korkeamman prioriteetin kuin esimerkiksi verkkoselaaminen. Näin ollen esimerkiksi tietojärjestelmän järjestelmänvalvojat voivat varmistaa jonkin verkkoon kytketyn laitteen ensisijaisuuden ja toiminnan. Quality of Serviceä ei tule kuitenkaan sekoittaa suoranaisesti palvelun laadun mittariksi, vaan sitä kuvaa paremmin palvelutasosopimus eli SLA (Service-level agreement). Kyseessä on asiakkaan ja palveluntarjoajan välinen sopimus, joka määrittelee vaatimustasot palvelun toiminnalle. Taulukossa 1 on esitelty esimerkkejä erilaisten reaaliaikaisten verkkosovellusten palvelutasojen vaatimustasoista.

Taulukko 1 – Reaaliaikaisten verkkosovellusten palvelutasosopimuksen vaatimustasoja

Liikenteen tyyppi	Maks. Kadonneet paketit	Maks. Yhdensuuntainen latenssi	Maks. Latenssin vaihtelu
VoIP (IP-puhe)	1 %	200 ms	30 ms
Videoneuvottelu	1 %	200 ms	30 ms
IPTV-video	0,001 %	250 ms	50 ms

4.2 Monitoroinnin eri tasot

Vuonna 2014 James Turnbull käynnisti kyselyn, jonka avulla hän pyrki selvittämään, miten eri IT-alan yritykset hyödyntävät monitorointia päivittäisessä toiminnassaan. Kysely koostui 12 kysymyksestä ja siihen vastasi 1016 osallistujaa. Analysointiin Turnbull huomioi vain kaikkiin kysymyksiin vastanneiden osallistujien vastaukset, joita oli 866.

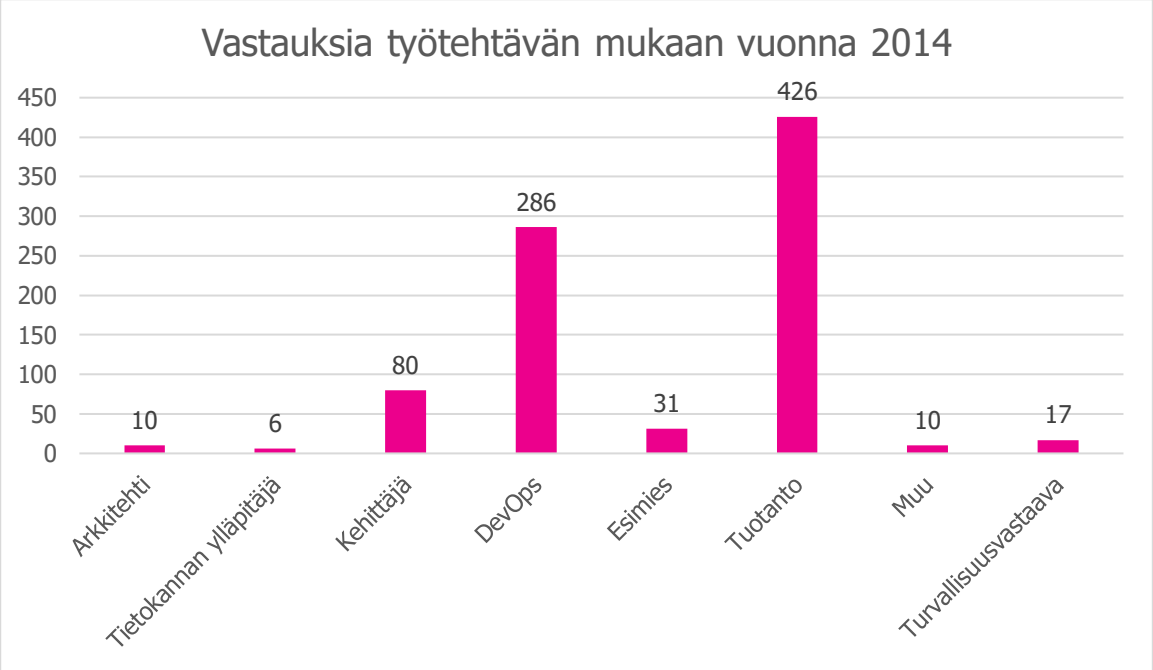
Vuonna 2015 Turnbull järjesti vastaavanlaisen kyselyn uudelleen selvittääkseen, onko yritysten tavoissa hyödyntää monitorointia tapahtunut muutoksia verrattuna edellisen kyselyn tuottamiin tuloksiin. Kysely koostui pääosin samoista kysymyksistä kuin edellisenä vuonna, mutta joukkoon oli liitetty muutama tarkentava kysymys liittyen monitoroitavan datan keräämiseen ja visualisoimiseen. Kyselyyn vastasi 1116 osallistujaa, joista 884 vastasi kaikkiin kysymyksiin. Kumpaakin kyselyä mainostettiin Turnbullin omassa blogissa, Twitterissä ja erinäisillä postituslistoilla (Turnbull, 2014).

Vuoden 2014 kyselyn kysymykset:

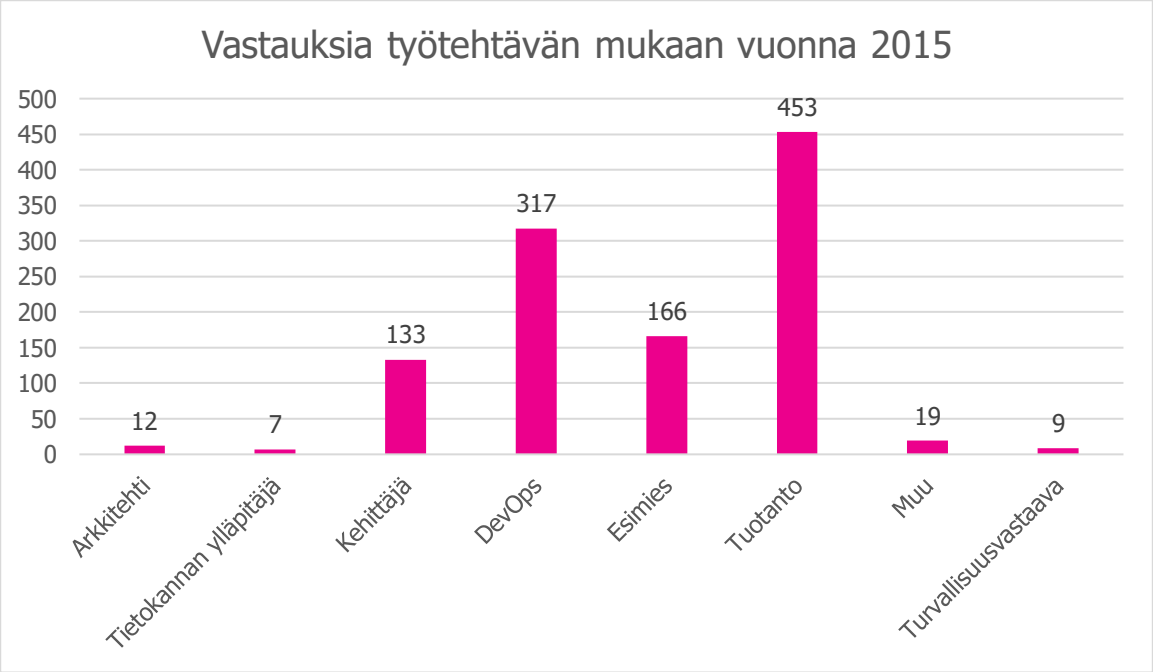
1. Mikä seuraavista kuvaa parhaiten työtehtävääsi?
2. Kuinka suuri yrityksesi on?
3. Oletko vastuussa yrityksesi IT-monitoroinnista?
4. Jos et ole vastuussa monitoroinnista, kuka on?
5. Mikä on ensisijainen valvontatyökalusi?
6. Keräättekö dataa infrastruktuurista ja sovelluksista?
7. Jos keräätte dataa, mihin käytätte sitä?
8. Mitä osa-alueita ympäristöstänne monitoroitte?
9. Milloin yleisimmin lisäätte valvontamittareita tai graafeja ympäristöönne?
10. Onko teillä koskaan vastaamattomia hälytyksiä valvontaympäristössänne?
11. Kuinka usein jokin asia menee vikaan, jota monitorointijärjestelmänne EI huomaa?
12. Käytättekö mitään hallintatyökalua monitoroinnin infrastruktuurin hallitsemiseksi?

Vuoden 2015 kyselyn lisätyt kysymykset:

1. Mitä työkaluja käytätte monitoroitavan datan keräämiseen?
2. Mitä työkaluja käytätte monitoroitavan datan säilyttämiseen?
3. Mitä työkaluja käytätte monitoroitavan datan visualisointiin?

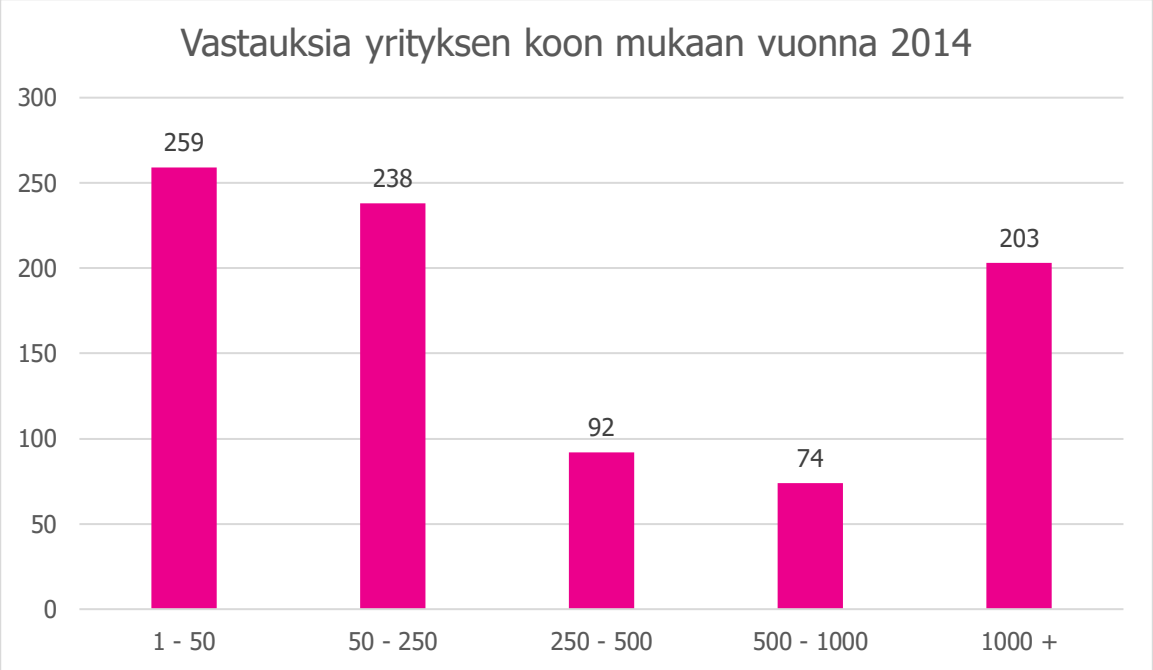


Kuvio 3 – Vastauksia työtehtävän mukaan vuonna 2014 (Turnbull, 2014).

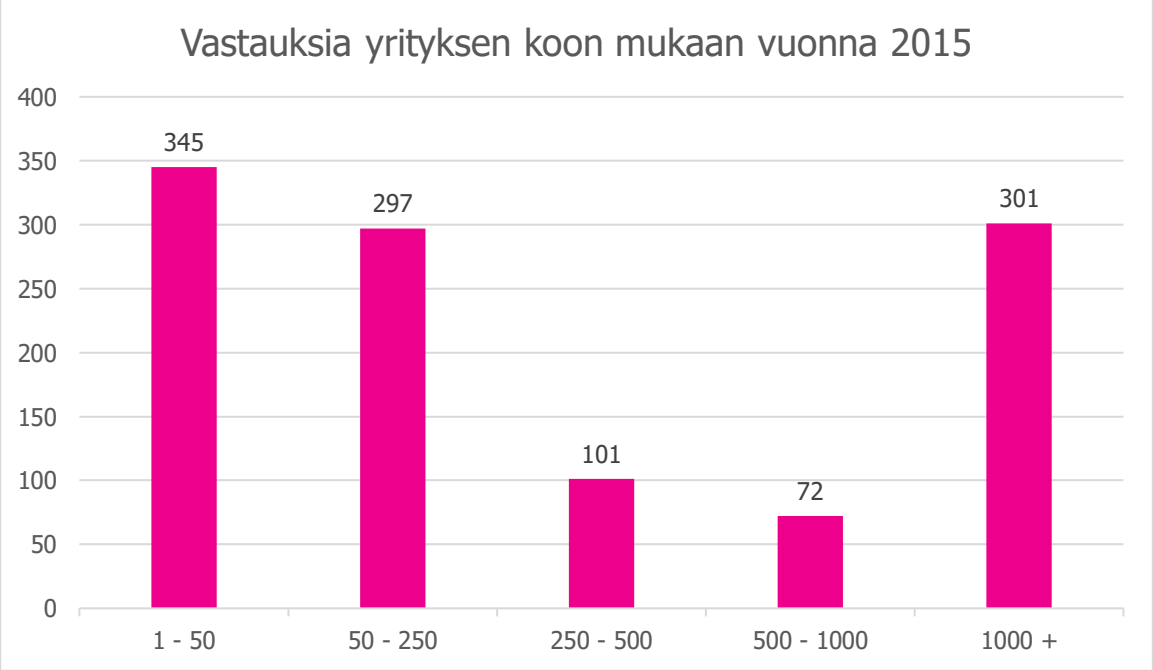


Kuvio 4 – Vastauksia työtehtävän mukaan vuonna 2015 (Turnbull, 2015).

Kyselyihin vastasi yhteensä 1982 osallistujaa. Kumpanakin vuonna suurin osa kyselyyn vastanneista koostui yrityksen operatiivisissa tehtävissä työskentelevistä henkilöistä, joiden osuus kaikista kyselyihin vastanneista oli noin 44 %. Toiseksi suurin ryhmä vastanneista koostui DevOps-tehtäviin sijoittuneista henkilöistä noin 30 % osuudella. Kolmanneksi suurin ryhmä oli poikkeava kyselyiden välillä. Vuonna 2014 kyseinen ryhmä koostui ohjelmistokehittäjistä noin 9 % osuudella, kun taas vuonna 2015 kyseinen ryhmä koostui erilaisissa esimiestehtävissä työskentelevistä henkilöistä noin 15 % osuudella.

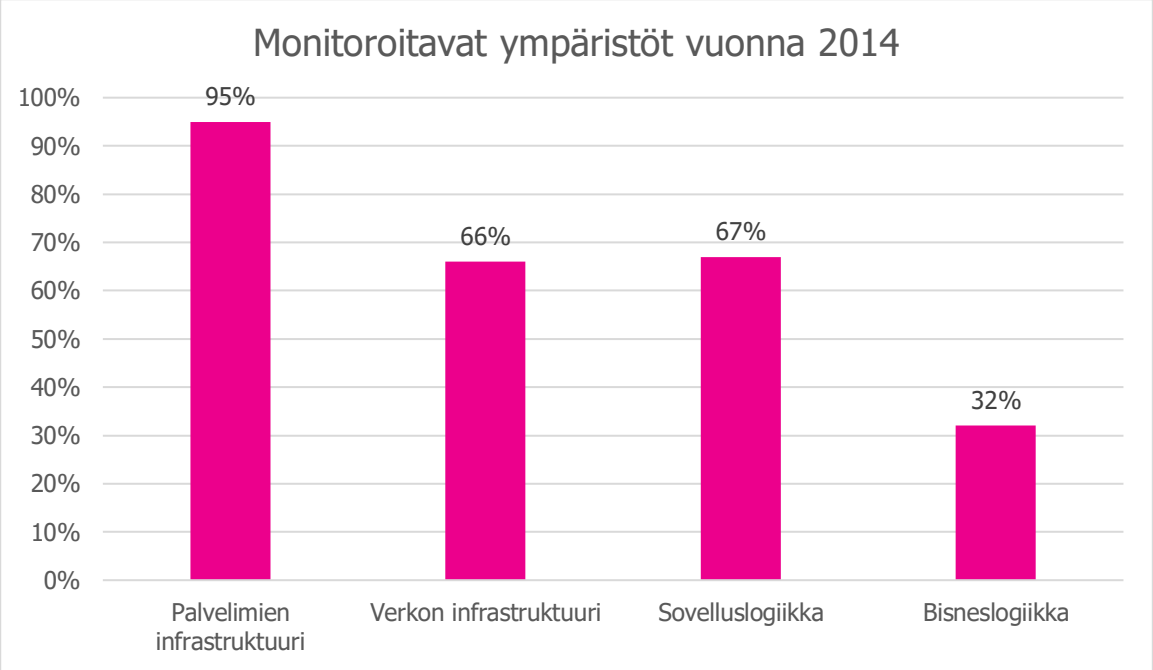


Kuvio 5 - Vuoden 2014 kyselyyn vastanneiden henkilöiden edustamien yritysten koko (Turnbull, 2014).

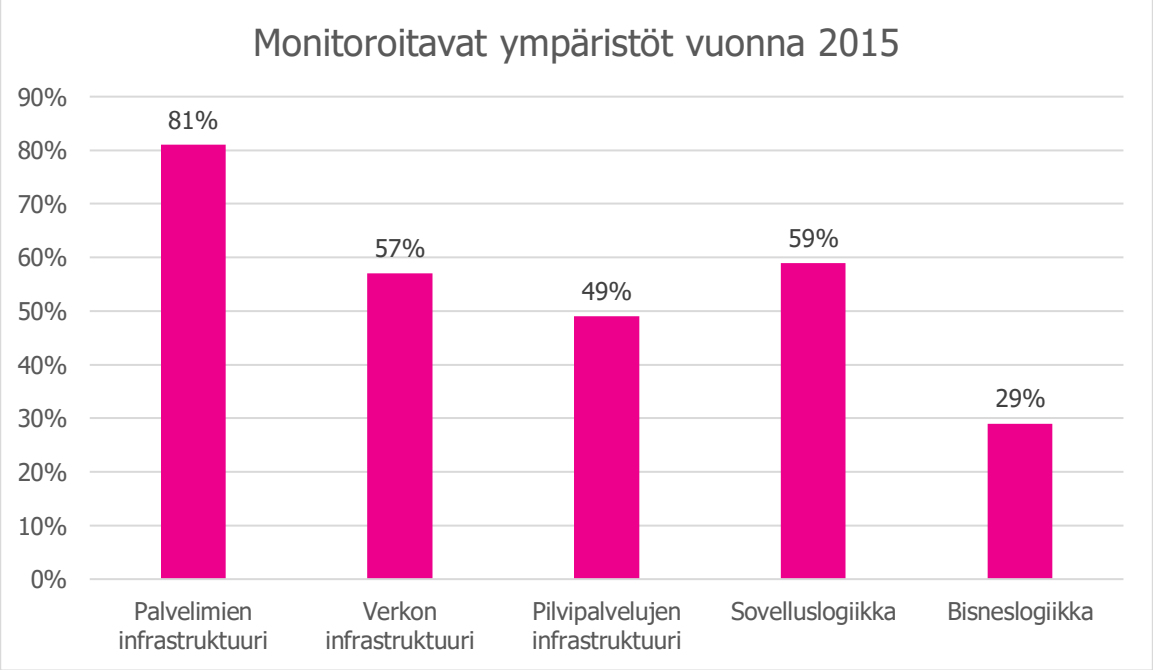


Kuvio 6 - Vuoden 2015 kyselyyn vastanneiden henkilöiden edustamien yritysten koko (Turnbull, 2015).

Kumpanakin vuonna suurin osa eli noin 30 % kaikista kyselyihin vastanneista henkilöistä edusti pieniä yrityksiä, joiden henkilöstömäärä sijoittui 1-50 välille. Toiseksi suurin ryhmä oli henkilöstömäärältään 50-250 sijoittuvat yritykset noin 27 % osuudella kaikista kyselyihin vastanneista. Kolmanneksi suurin ryhmä noin 25 % osuudella kaikista kyselyihin vastanneista koostui yrityksistä, joiden henkilöstömäärä oli yli 1000 henkilöä.



Kuvio 7 - Monitoroitavat ympäristöt vuonna 2014

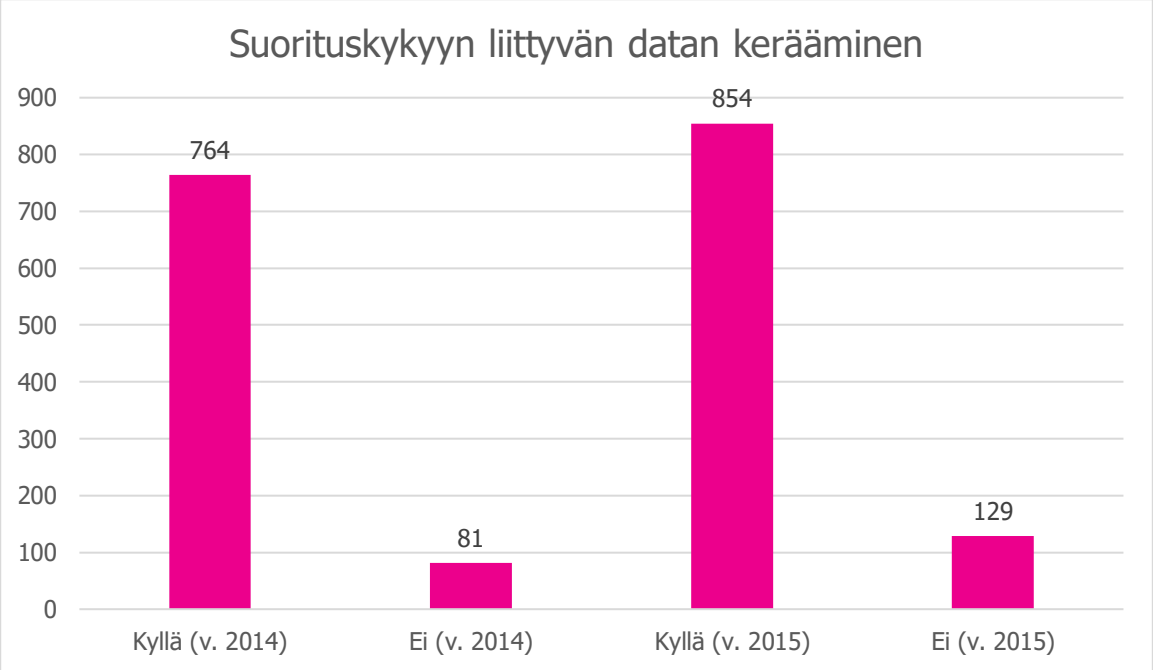


Kuvio 8 - Monitoroitavat ympäristöt vuonna 2015

Kumpanakin vuonna yritysten pääasiallinen monitoroinnin kohde oli palvelininfrastruktuuri, jota monitoroi 95 % vuoden 2014 ja 81 % vuoden 2015 kyselyyn vastanneista yrityksistä. Toisiksi suurin monitoroinnin kohde yrityksillä oli sovelluslogiikka, jota vuonna 2014 monitoroi 67 % ja vuonna 2015 59 % yrityksistä. Kumpanakin vuonna matalin monitoroitava ympäristö oli liiketoimintalogiikka, jota vuonna 2014 monitoroi 32 % ja vuonna 2015 vain 29 % yrityksistä.

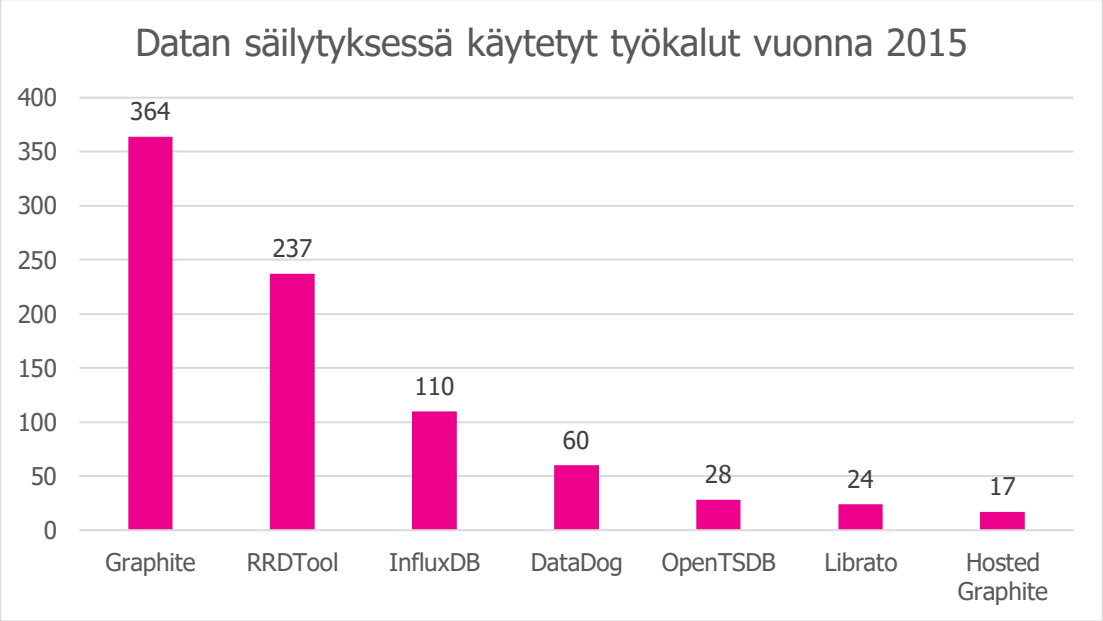
Uutena tulokkaana vuoden 2015 kyselyyn Turnbull lisäsi pilvi-infrastruktuurin, joka oli monitoroinnin alaisuudessa 49 % yrityksistä. Kyselyyn oli myös lisätty ”Muu ympäristö, mikä?”-tyyppinen vaihtoehto, johon osa henkilöistä oli vastannut monitoroitavaksi ympäristöksi muun muassa tietokannat, työasemat ja ulkoiset palvelut.

Kumpanakin vuonna kyselyssä kysyttiin, keräävätkö yritykset suorituskykyyn liittyvää dataa. Mikäli kysymykseen vastasi kyllä, kysyttiin heiltä mihin he käyttävät tätä keräämäänsä dataa. Vuoden 2015 kyselyyn oli lisätty kolme kysymystä tarkentamaan yrityksen käyttämiä työkaluja datan keräämiseen, säilöntään ja visualisointiin liittyen.



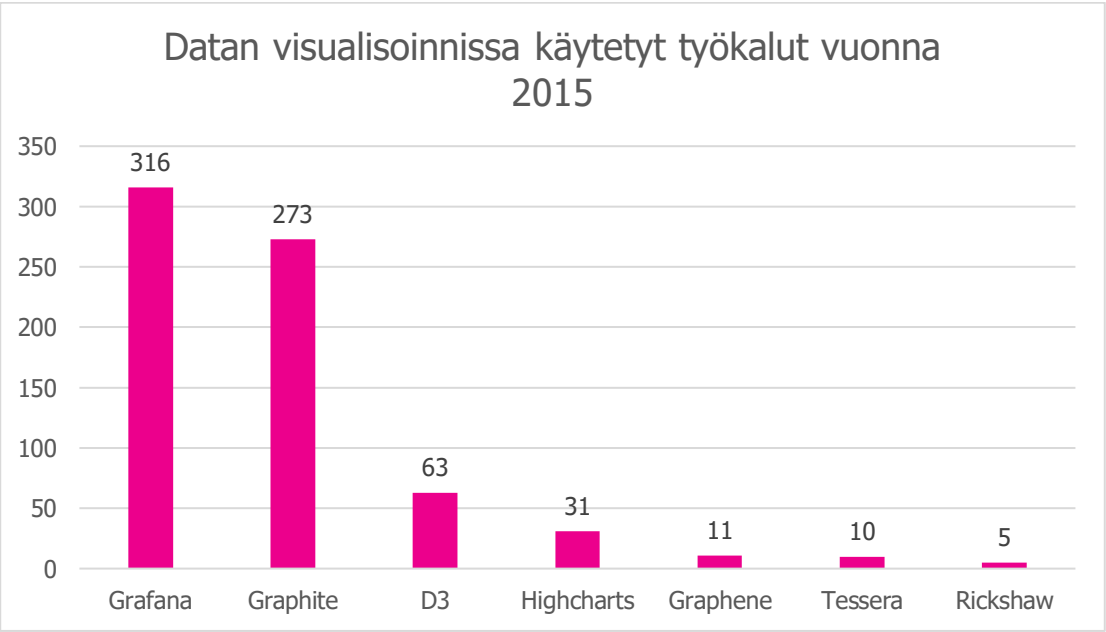
Kuvio 9 - Datan kerääminen vuosina 2014 ja 2015 (Turnbull, 2014 ja 2015).

Kuviosta 9 on nähtävissä, että noin 90 % vuoden 2014 kyselyyn vastanneista keräsivät suorituskykyyn liittyvää dataa, kun taas vuonna 2015 dataa keräsivät noin 88 % kyselyyn vastanneista. On siis päivänselvää, että data näyttelee keskeistä roolia tietojärjestelmien suorituskyvyn valvonnassa. Kerättyä dataa käytettiin pääasiallisesti tietojärjestelmän suorituskyvyn analysointiin sekä vikojen ja trendien havaitsemiseen (Turnbull, 2014 ja 2015).



Kuvio 10 - Datan säilönnän työkalut vuonna 2015 (Turnbull, 2015).

Tietojärjestelmän suorituskykyyn liittyvän datan säilömiseen käytettiin enimmäkseen Graphitea noin 43 % osuudella, RRDToolia noin 33 % osuudella ja InfluxDB:tä noin 13 % osuudella. Graphiten suosiota selittää varmasti se, että se yhdistää datan säilömiseen ja visualisointiin tarvittavat työkalut yhdeksi kokonaisuudeksi.



Kuvio 11 - Datan visualisoinnin työkalut vuonna 2015 (Turnbull, 2015).

Tietojärjestelmän suorituskykyyn liittyvän datan visualisointiin käytettiin enimmäkseen Grafanaa noin 45 % osuudella, Graphitea noin 39 % osuudella ja D3:a noin 9 % osuudella. Grafanan suosiota selittää sen monipuolisuus ja muokattavuus verrattuna kilpaileviin työkaluihin, kuten esimerkiksi Graphiteen. Grafanan avulla on mahdollista tehdä monipuolisia, jopa hyvinkin komplekseja visualisointeja hyödyntäen sen tarjoamia erilaisia visualisoinnin muotoja.

Kyselyjen ja saamiensa vastausten perusteella Turnbull kehitti kolmijakoisen mallin kuvaamaan monitoroinnin eri tasoja. Tasojen tarkoituksena on kuvata, minkälaista monitorointia sitä suorittavissa yrityksissä on havaittavissa. Nämä kolme tasoa ovat:

- Manuaalinen monitorointi
- Reaktiivinen monitorointi
- Proaktiivinen monitorointi

4.2.1 Manuaalinen monitorointi

Matalimmalla tasolla monitorointi on pääosin manuaalista eli se on joko käyttäjän aloitteesta tapahtuvaa tai sitä ei suoriteta lainkaan. Mikäli monitorointia suoritetaan, se hallitaan pääosin joko tarkistuslistojen tai muiden ei-automatisoitujen prosessien avulla. Useimmiten monitoroinnista tulee lastikulttitieteen tyylistä, jolloin vain aikaisemmin järjestelmässä rikkoutuneet tai toimimattomat osa-alueet päätyvät monitoroinnin alaisuuteen. Kyseisissä osa-alueissa ilmeneviä vikoja korjataan seuraamalla tietyntyyppisiä mekaanisia vaiheita, joiden avulla viat näissä osa-alueissa on aikaisemminkin saatu korjattua (Turnbull, 2016).

Matalimman tason monitorointi keskittää huomionsa seisokkien minimoimiseen sekä omaisuuden hallintaan, eikä se tuota juurikaan arvoa laadun tai palvelun mittaamiseksi. Manuaalista monitorointia harjoittavat tyypillisesti pienet yritykset, joiden IT-osasto on pieni tai sitä ei ole lainkaan.

4.2.2 Reaktiivinen monitorointi

Seuraavalla tasolla monitorointi on pääosin automaattista, mutta siitä löytyy joitain manuaalisen monitoroinnin piirteitä, tai joitain osa-alueita ei monitoroida lainkaan. Monitoroinnin automatisoinnin tukena on usein käytetty erilaisia työkaluja, joiden avulla tarkistetaan säännöllisesti erilaisia perustietoja järjestelmän käytettävissä oleviin laitteistoresursseihin liittyen. Monitoroinnin tukena voidaan käyttää erilaisia hälytyksiä, jotka hälyttävät tiettyjen tarkasteltavien suureiden raja-arvojen ylityksistä järjestelmänvalvojalle esimerkiksi sähköpostiviestien avulla (Turnbull, 2016).

Reaktiivisen monitoroinnin huomio keskittyy järjestelmän saatavuuden mittaamiseen ja monitoroinnista saatua tietoa käytetään osittain käyttäjäkokemuksen mittaamiseen. Suorituskykyyn liittyvää dataa visualisoidaan yhden tai useamman keskitetyn päätteen avulla, ja käytössä voi olla erilaisiin käyttötarkoituksiin kohdennettuja visualisointialustoja. Jotain dataa voidaan säilyttää myöhempää jatkokäyttöä varten.

Reaktiivista monitorointia harjoittavat yritykset ovat kooltaan tyypillisesti pieniä tai keskikokoisia yrityksiä, ja vastaavanlainen monitorointi on tyypillistä myös suurempien yritysten IT-osastojen keskuudessa. Monitorointijärjestelmää ylläpidetään monitorointitason nimen mukaan reaktiivisesti, jolloin tyypillisimmät päivitykset sen toimintaan tapahtuvat vastauksena erilaisiin tapahtumiin.

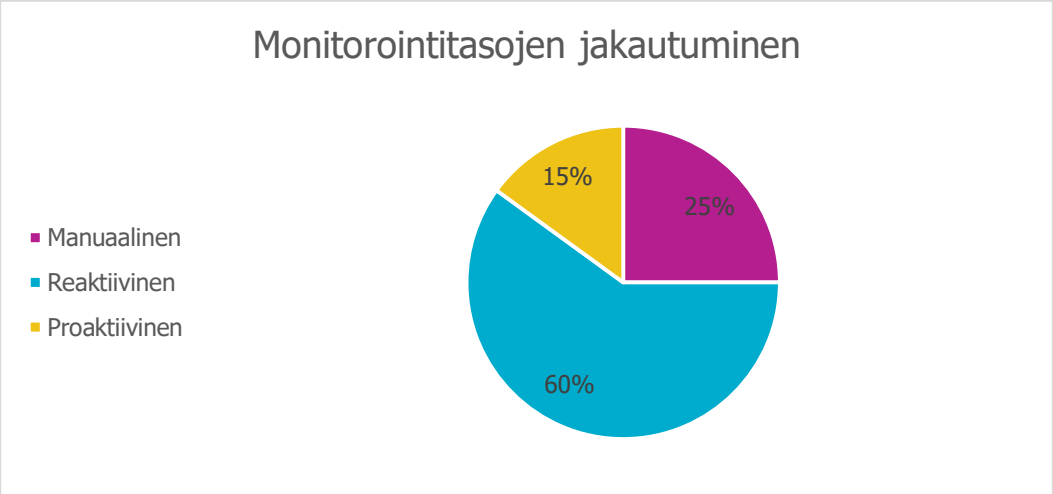
4.2.3 Proaktiivinen monitorointi

Korkeimmalla tasolla monitorointia pidetään yritystoiminnan sekä yrityksen infrastruktuurin ytimenä. Valvonta on automatisoitua ja sen tukena käytetään laajalti erilaisia työkaluja, joiden avulla mittausten tuloksista saadaan irti suurin mahdollinen hyöty. Mittaukset painottuvat koko järjestelmän suorituskyvyn mittaamiseen, eivätkä suinkaan vain yksittäisten laitteistoresurssien tarkisteluun. Suorituskykyyn liittyvää dataa kerätään ja sitä hyödynnetään analysoinnissa ja vianmäärityksessä. Monitorointia tuetaan hallitusti erilaisilla hälytyksillä ja automaattisilla vastauksilla (Turnbull, 2016).

Proaktiivisen monitoroinnin huomio keskittyy palvelun laadun sekä käyttäjäkokemuksen mittaamiseen. Suorituskykyyn liittyvää dataa visualisoidaan kohdennetusti käyttötarpeen mukaan erilaisia visualisointialustoja hyödyntäen. Monitorointijärjestelmän ylläpito on niin ikään proaktiivista, jolloin sitä päivitetään jatkuvasti ja yritystoiminta tukeutuu sen olemassaoloon.

Proaktiivinen monitorointi on tyypillisintä webkehitykseen keskittyvissä yrityksissä sekä erilaisissa varttuneissa startup-yrityksissä. Vastaavanlainen lähestymistapa on myös yleisesti omaksuttu käyttöön yrityksissä, jotka tukeutuvat DevOps-toimintamalliin. Kyseisen toimintamallin tarkoituksena on edistää kehitys- ja tuotantotoimintojen välistä toimintaa yrityksen sisällä. Kehitystoiminnon tyypillisiin tehtäviin kuuluvat uusien toiminnallisuuksien laatimiseen sekä vikojen korjaamiseen liittyvät tehtävät aina suunnittelusta ja ohjelmoinnista testaukseen ja dokumentointiin. Tuotantotoiminnon tehtäviin puolestaan lukeutuu palveluiden vaatiman infrastruktuurin rakentaminen ja ylläpito.

4.2.4 Monitorointitasojen jakautuminen



Kuvio 12 – Monitorointitasojen jakautuminen (Turnbull, 2016).

Turnbullin laatimat monitoroinnin eri tasot jakautuivat kyselyihin vastanneiden yritysten mukaan kuvion 12 mukaisesti. Suurin osa yrityksistä noin 60 % osuudella sijoittui reaktiivisen monitoroinnin piiriin, mikä ei tule juurikaan yllätyksenä. Reaktiivinen monitoroinnin taso on suhteellisen helppoa saavuttaa, ja yleensä se kattaa yritysten perustarpeet monitoroinnin suhteen. Proaktiivisen tason tavoitteleminen on suositeltavaa, mutta sen saavuttaminen edellyttää sen nimensä mukaisesti proaktiivista käyttäytymistä monitoroitavien kohteiden suhteen.

Proaktiivisen monitorointitason saavuttamiseksi fokus monitoroinnissa kohdistuu erilaisiin tapahtumiin, metriikoihin sekä lokeihin, eikä suinkaan vain yksittäisten palvelujen tilan testaamiseen yksinkertaisilla tarkistuksilla. Turnbull mainitsee kirjassaan, että yhdistämällä erilaisia tapahtumia, metriikoita ja lokeja oikein ne toimivat totuuden lähteenä sekä monitoroitavan ympäristön tilasta sekä sen suorituskyvystä. Kirjassaan hän myös toteaa, että mikäli jokin metriikka visualisoituu oikein, niin silloin sen taustalla sijaitseva palvelu on luultavasti saatavilla. Jos metriikka ei visualisoidu, niin silloin palvelu ei välttämättä ole enää saatavilla (Turnbull, 2016).

5 TIETOJÄRJESTELMÄ

Tässä opinnäytetyössä tietojärjestelmällä viitataan sairaalaverkon sisällä toimivaan potilastietojärjestelmään, joka koostuu GE Healthcaren Centricity High Acuity-tuoteperheen ohjelmistoista. Paremman ymmärryksen saamiseksi on tarpeellista kertoa hieman kyseisten ohjelmistojen käyttötarkoituksista.

5.1 Centricity High Acuity

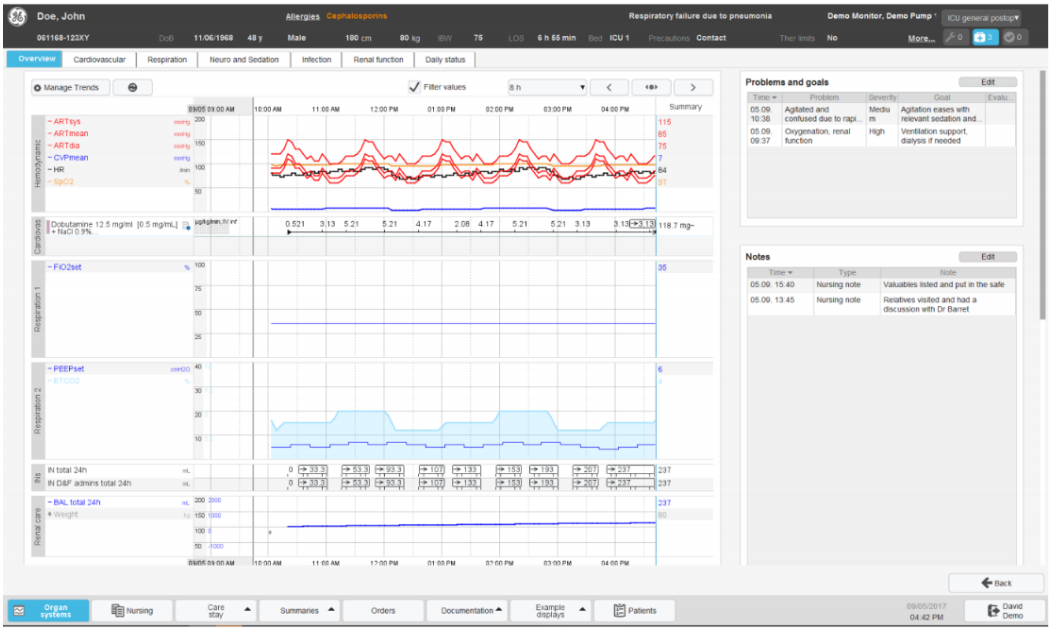
Centricity High Acuity on kliiniseen dokumentaatioon kehitetty järjestelmä, jonka avulla lääkärit, hoitajat ja muut hoitoalan ammattilaiset pääsevät käsiksi tarvittaviin potilastietoihin joko paikan päällä tai etänä missä ja milloin tahansa. Järjestelmä tarjoaa hoitohenkilökunnalle tarvittavia dokumenttimalleja koko hoitojakson ajan (GE Healthcare, 2018).

5.1.1 Anesthesia

Centricity High Acuity Anesthesia on kliininen tietojärjestelmä, jonka tarkoitus on tukea kliinistä dokumentointia sekä päätöksentekoa koko perioperatiivisen prosessin ajan (preoperatiivinen arviointi, intraoperatiivinen vaihe, toipumisvaihe ja potilaan postoperatiivinen arviointi). Ohjelmiston avulla koulutetut kliiniset käyttäjät voivat muun muassa hakea, kirjata, tallentaa ja katsella potilastietoja sekä suunnitella hoitoa tarpeen mukaan (GE Healthcare, 2018).

5.1.2 Critical Care

Centricity High Acuity Critical Care on kliininen tietojärjestelmä, jonka tarkoitus on tukea kliinistä dokumentointia sekä päätöksentekoa hoitoyksiköissä, joissa hoidetaan tehohoitoa tarvitsevia potilaita. Ohjelmiston avulla koulutetut kliiniset käyttäjät voivat muun muassa hakea, kirjata, tallentaa ja katsella potilastietoja sekä suunnitella hoitoa tarpeen mukaan (GE Healthcare, 2018).



Kuva 4 – Kuvankaappaus Centricity High Acuity Critical Care-ohjelmistosta (GE Healthcare, 2018a).

6 KÄYTETYT TEKNIIKAT

6.1 InfluxDB

InfluxDB on InfluxDatan kehittämä, avoimeen lähdekoodiin perustuva aikasarjatietokanta. Se on kirjoitettu Googlen kehittämällä Go-ohjelmointikielellä ja se on optimoitu käsittelemään suuria kirjoitus- ja lukutapahtumia. InfluxDB on tarkoitettu käytettäväksi mihin tahansa käyttötarkoitukseen, johon liittyy suuria määriä aikaleimattua dataa. Esimerkkejä vastaavista käyttötarkoituksista ovat muun muassa DevOps-seuranta, Internet of Things-anturidata sekä reaaliaikainen analyysi (InfluxData, 2018a).

Aikasarjatietokanta on nimensä mukaisesti aikaleimatun datan säilöntään tarkoitettu tietokanta, jossa data tallennetaan ja indeksoidaan aikaleiman mukaan. InfluxDB:ssä data tallennetaan tietokantoihin (database), jotka sisältävät mittauksia (measurement). Näille tietokannoille on mahdollista määrittää säilytysohjeet datan säilyttämistä varten (retention policy). Säilytysohjeet kertovat tietokannalle, kuinka pitkältä ajanjaksolta se säilyttää dataa ja kuinka monta kopiota kyseisestä datasta säilytetään klusterissa. Tietokantojen sisältämiä mittauksia voi verrata SQL-pohjaisten tietokantojen tauluihin, sillä käsitteellisellä tasolla ne ovat keskenään hyvinkin samankaltaisia.

Tietokannan tietojen käsittely tapahtuu käyttämällä InfluxQL-kyselykieltä. Kielen syntaksi on tarkoituksenmukaisesti hyvin samankaltainen SQL-kyselykielen kanssa, jotta käyttäjät joilla on kokemusta muista SQL-kyselykielistä, voisivat helposti omaksua sen käytön. Haut tietokantaan tapahtuvat SELECT-lausetta käyttämällä, ja sen rinnalla on mahdollista käyttää muita tarkentavia ehtoja, kuten esimerkiksi GROUP BY-, INTO- tai WHERE-ehtoja. Kieli sisältää myös muita hyödyllisiä funktioita tiedon hakemiseen.

Yksi tietokanta koostuu vähintään kahdesta pakollisesta kolumista: aikaleimasta (time) sekä kentistä (Field). Aikaleima tallennetaan time-kolumniin jokaiselle datapistelle formaatissa RFC3339 ja niitä voi olla vain yksi jokaista datapistettä kohden. Kenttä koostuu avain-arvo-parista, jossa kolumnin nimi toimii avaimena (Field Key) ja sitä vastaavat datapisteet arvoina (Field Value). Avaimet tallennetaan merkkijonoina, kun taas arvot voidaan tallentaa joko merkkijonoina, totuusarvomuuttujina tai liuku- tai kokonaislukuina (InfluxData, 2018a).

Tietokanta voi myös sisältää tunnisteita (Tag). Tunnisteet koostuvat kenttien mukaisesti avain-arvo-parista, jossa kolumnin nimi toimii avaimena (Tag Key) ja sitä vastaavat datapisteet arvoina (Tag Value). Kaikki tunnisteet tallennetaan merkkijonoina. Tunnisteet eivät kuitenkaan ole pakollisia, mutta niiden käyttö on suotavaa lähinnä siksi, että kenttien tavoin kyseiset kolumnit eivät ole indeksoituja. Näin ollen tunnisteille tehtävät tietokantahaut ovat nopeampia ja ne ovatkin ideaalisia usein haetun datan säilyttämiseen (InfluxData, 2018a).

Nämä kaikki edellä mainitut muodostavat yhdessä pisteen (point). Piste on siis käytännössä yksi rivi samassa sarjassa samalla aikaleimalla.

6.2 Grafana

Grafana on Grafana Labsin kehittämä, avoimeen lähdekoodiin perustuva yleiskäyttöinen visualisointityökalu. Sen avulla on mahdollista luoda erilaisia kojelautoja (dashboard), jotka mahdollistavat aikaleimatun datan visualisoimisen. Grafana toimii websovelluksena ja se tukee virallisesti kahdeksaa eri tietolähdettä, joihin kuuluvat muun muassa Elasticsearch ja InfluxDB. Versiosta 3.0 lähtien Grafanaan on voinut asentaa myös muita tietolähteitä liitännäisinä (plugin), mutta ne eivät ole virallisesti tuettuja (Grafana Labs, 2018).

Grafanan keskeisimmässä roolissa ovat kojelaudat. Yksittäinen kojelauta on käytännössä järjestetty kokoelma erilaisia paneeleja (panel), jotka yhdessä muodostavat visuaalisen esityksen kerätystä datasta. Kojelautoja voi olla yksi tai useampia, ja niitä on mahdollista tuoda tai viedä pois järjestelmästä JSON-muotoisina dokumentteina.

Paneelit ovat yksittäisiä komponentteja jotka visualisoivat dataa halutusta lähteestä, esimerkiksi graafin muodossa. Jokainen paneeli sisältää kyselyeditorin (Query Editor), jonka toiminnallisuus on hieman poikkeava käytetystä tietolähteestä riippuen. Kyselyeditori helpottaa datan hakemista käytetystä tietolähteestä, ja kaikki sen avulla tehdyt muutokset paneelin kyselyyn (Query) heijastuvat välittömästi. Kojelautojen tapaan myös paneeleita voi tuoda tai viedä pois järjestelmästä JSON-muotoisina dokumentteina.

Kuva 5 - Esimerkki Grafanan kojelaudasta (Grafana Labs, 2018).

6.3 Postfix

Postfix on avoimeen lähdekoodiin perustuva MTA-ohjelmisto (Mail Transfer Agent), joka mahdollistaa sähköpostiviestin välittämisen ja vastaanottamisen, sekä omalla domain-päätteellä varustettujen sähköpostiosoitteiden luomisen eri käyttäjille. Postfixin on alun perin luonut hollantilainen fyysikko ja ohjelmoija Wietse Venema vuonna 1997. Ohjelmistoa kehitetään edelleen itse alkuperäisen luojan Wietse Veneman sekä useiden muiden tahojen avustuksella (Postfix, 2018).

Postfix ei tue IMAP- tai POP3-protokollien mukaista autentikointia viestien välitykseen, jonka myötä sen rinnalle on asennettava protokollia tukeva palvelin. Sähköpostiohjelmistot käyttävät kyseisiä protokollia sähköpostiviestien noutamiseen sähköpostipalvelimelta.

6.4 Courier

Courier on avoimeen lähdekoodiin perustuva MTA-ohjelmisto, joka toimii muun muassa IMAP- ja POP3-palvelimena. Kyseinen ohjelmisto on parhaiten tunnettu sen toimivuudesta juuri kyseisiä protokollia hyödyntävänä palvelimena, sillä siitä on mahdollista asentaa vain kyseiset palvelimet sisältävät komponentit. Näin ollen sen voi yhdistää kätevästi muihin sähköpostipalvelimiin, kuten esimerkiksi Postfixiin, joka ei itsessään tue kyseisten protokollien mukaista autentikointia viestien välityksessä (Double Precision Inc., 2015).

6.5 Telegraf

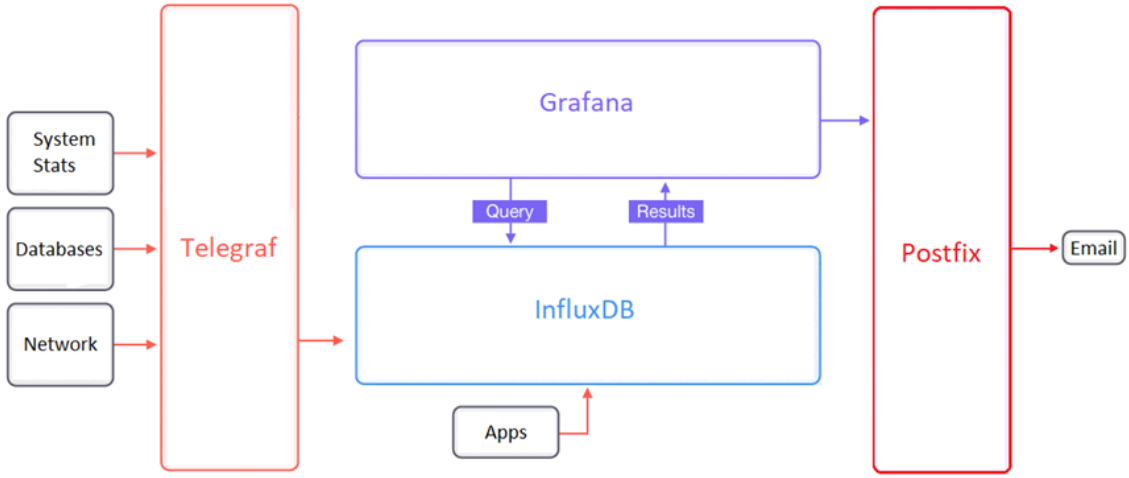
Telegraf on InfluxDatan kehittämä, avoimeen lähdekoodiin perustuva datan keräämiseen ja lähettämiseen tarkoitettu keräysagentti. Sen toiminta perustuu erilaisten liitännäisten hyödyntämiseen. Liitännäispohjaisen arkkitehtuurinsa ansiosta Telegrafin toimintoihin voi vaikuttaa teoriassa kuka tahansa, sillä kuka tahansa voi kirjoittaa sille oman liitännäisen. Vuodesta 2015 lähtien Telegrafin liitännäisten määrä on kasvanut jo yli 150:een, joista suurin osa on yhteisön laatimia (InfluxData, 2018b).

Yleisimmät Telegrafin kanssa käytettävät liitännäiset ovat joko syöte- tai tulosteliitännäisiä (input/output). Syöteliitännäisten tarkoitus on nimensä mukaisesti kerätä dataa jostain lähteestä, kun taas tulosteliitännäiset lähettävät dataa eteenpäin. Vastaavanlaisia liitännäisiä ovat esimerkiksi *postgresql*-liitännäinen, joka mahdollistaa suorituskykyyn liittyvän datan keräämisen Postgres-asennuksesta sekä *influxdb*-liitännäinen, joka kirjoittaa dataa InfluxDB-tietokantaan joko HTTP- tai UDP-protokollan avulla (InfluxData, 2018b).

7 MONITOROINNIN TOTEUTUS JA TESTAUS

7.1 Monitorointipalvelimen asennus

Monitoroinnin toteutusta varten pystytettiin erillinen monitorointipalvelin, jonka tarkoituksena oli toteuttaa tietojärjestelmän monitorointi. Lopuksi monitorointipalvelimelle yhdistettiin yksi testikäytössä ollut tietojärjestelmä. Yhtenä opinnäytetyön tavoitteena oli päivittää monitorointipalvelimella käytetyt työkalut uusimpiin versioihin, joten kaikista kyseisistä ohjelmistoista asennettiin niiden uusimmat versiot.



Kuva 6 - Monitorointipalvelimen arkkitehtuuri.

Monitorointipalvelimelle asennettiin käyttöjärjestelmäksi Ubuntu 16.04 LTS erillisen dokumentaation mukaisesti. Asennuksen yhteydessä järjestelmälle määritettiin kiinteä IP-osoite, jotta testikäytössä ollut tietojärjestelmä voitaisiin onnistuneesti yhdistää monitorointipalvelimelle.

Monitorointipalvelimen aikasarjatietokantana toimiva InfluxDB asennettiin InfluxDatan laatiman dokumentaation mukaisesti (InfluxData, 2018a). Käytetty versio InfluxDB:stä oli versio 1.6.2. Ensin palvelimelle oli lisättävä InfluxDatan pakettivarastot (repository) terminaalin kautta seuraavilla komennoilla:

curl -sL https://repos.influxdata.com/influxdb.key | sudo apt-key add -
source /etc/lsb-release
echo "deb https://repos.influxdata.com/${DISTRIB_ID,,} ${DISTRIB_CODENAME} stable" | sudo tee /etc/apt/sources.list.d/influxdb.list

Tämän jälkeen itse InfluxDB:n palvelu asennettiin seuraavilla komennoilla:

sudo apt-get update && sudo apt-get install influxdb
sudo service influxdb start

InfluxDB:n asentamisen jälkeen vuorossa oli Grafanan uusimman version 5.1.0:n asennus. Asennus suoritettiin Grafana Labsin laatiman dokumentaation mukaisesti (Grafana Labs, 2018). Ensin Grafanan pakettivarastot tuli lisätä palvelimelle seuraavalla komennolla:

echo "deb https://packagecloud.io/grafana/stable/debian/ stretch main" | sudo tee /etc/apt/sources.list

Tämän jälkeen Grafanan paketti ladattiin ja asennettiin seuraavilla komennoilla:

wget https://s3-us-west-2.amazonaws.com/grafana-releases/release/grafana_5.1.2_amd64.deb
sudo apt-get install -y adduser libfontconfig
sudo dpkg -i grafana_5.1.2_amd64.deb
sudo apt-get update
sudo apt-get install grafana

Hälytyksiä varten monitorointipalvelimelle valittiin Postfix, jonka tarkoituksena on toimittaa Grafanan välittämät hälytykset erikseen konfiguroituihin sähköpostiosoitteisiin. Postifxin tuorein versio 3.3.0 asennettiin monitorointipalvelimelle seuraavien komentojen avulla:

sudo apt-get install postfix

Lopuksi palvelimelle ladattiin uusimmat versiot (IMAP versio 4.18.2 ja POP3 versio 0.78) Courierin tarjoamista IMAP- ja POP3-palvelimista, jotta sähköpostiviestejä voitaisiin välittää käyttäen kyseisiä protokollia:

sudo apt-get install courier-imap
sudo apt-get install courier-pop

Tietojärjestelmän virkaa toimittaneelle palvelimelle asennettiin Telegrafin versio 1.5.2, jonka tarkoituksena on kerätä ja välittää suorituskykyyn liittyvää dataa kyseiseltä palvelimelta. Telegrafin asennuspaketti ladattiin InfluxDatan sivuilta, ja asennettiin Windowsin komentoriviltä seuraavilla komennoilla:

telegraf.exe -service install -config telegraf.conf

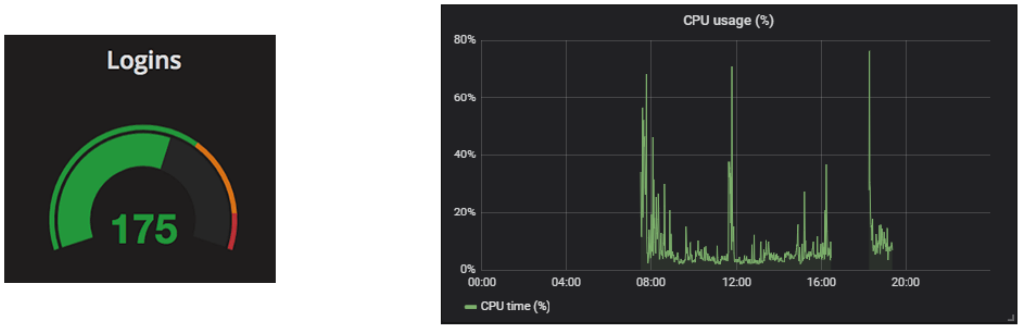
Telegrafin asennuksen yhteydessä palvelulle annettiin parametrinä konfiguraatiotiedoston sijainti, jonka mukaan Telegraf määritetään toimivaksi. Konfiguraatioon lisättiin SQL Server-liitännäinen, jonka avulla Telegraf välittää suorituskykyyn liittyviä tietoja tietokantapalvelimesta. Tämän lisäksi konfiguraatiotiedostoon lisättiin muutamia Windowsin suorituskykylaskureita, joihin lukeutui muun muassa metriikoita levyn sekä tietoliikenteen käytöstä. Lopuksi kaikki työkalut konfiguroitiin toimimaan CHA System Performance Dashboard-työkalun mukana tulleiden ohjeiden mukaisesti.

7.2 Kojelautojen muokkaus

Kojelautojen muokkauksen tavoitteena oli luoda visualisointeja, joiden avulla niiden katselijat voisivat helposti ja nopeasti ymmärtää monitoroitavan tietojärjestelmän tilan. Esimerkkiä varten luotiin yksi kojelauta, jonka tarkoituksena on antaa katselijoilleen korkean tason yleiskuva monitoroitavasta tietojärjestelmästä. Kyseinen kojelauta nimettiin Performance Dashboardiksi, ja sen visualisoimiin metriikoihin lukeutuu muun muassa:

- Suoritinaika prosentteina
- Levyn käyttö prosentteina
- Keskimääräinen aika kirjoitus- tai lukutapahtuman suorittamiseen levyllä
- Käytettävissä olevan keskusmuistin määrä
- Virtuaalikoneen (JVM) käytettävissä oleva muistin määrä
- Sivutustiedoston käyttö
- Tietokantojen terveydentila
- Tietokantojen lokitiedostojen koko prosentteina
- Tietoverkon kaistan käyttö prosentteina
- Tietoverkon lähetetyt ja vastaanotetut tavut
- Tietoverkon jonossa olevat paketit

Kojelaudan pääasiallisena visualisointimenetelmänä toimii Grafanan graafiset paneelit (Graph Panel). Niiden avulla on helpompi saada yleiskuva tietojärjestelmän tilasta kuin esimerkiksi Grafanan yksittäisten lukuarvopaneeleiden (Singlestat) avulla, sillä graafien avulla erilaisten trendien havaitseminen ja tulkitseminen on helpompaa. Yhdistelemällä graafeja visuaalisesti tarkoituksenmukaisella tavalla, kokonaisuuden hahmottamisesta tulee nopeaa ja intuitiivista.



Kuva 7 - Singlestat ja Graph Panel.

Edellä mainittujen metriikoiden lisäksi kojelaudalla visualisoidaan perustietoja tietojärjestelmän konfiguraatioon liittyen. Konfiguraatiosta viestivät muun muassa sovellusversio, tietokantojen versio sekä tietojärjestelmän käytettävyysaika.

Kuva 8 – Performance Dashboard-kojelauta.

Kuvassa 8 on esitelty Performance Dashboard-kojelauta. Kojelaudan esittämät metriikat jaettiin kahteen osioon: tietojärjestelmän yleiset tiedot (System information) sekä tietojärjestelmän suorituskyky (System performance). Jakamalla metriikat omiin osa-alueisiinsa trendien ja metriikoiden välisten suhteiden havaitseminen on helpompaa. Kojelaudan muodostamisen yhteydessä sovellettiin kappaleessa 3.2 mainittua visualisoinnin tarkistuslistaa, ja samalla pyrittiin välttämään visuaalisen rihkaman muodostumista.

Singlestat-paneelien avulla esitetään pääosin keskimääräisiä arvoja erinäisistä tietojärjestelmän perusmetriikoista, joita ovat esimerkiksi keskimääräinen suoritinkäyttö, levynkäyttö ja vapaa levytila. Singlestat-paneelien etuna on se, että niiden avulla on helppo havaita keskiarvot kyseisistä mittauksista muokkaamalla kojelaudalle valittua aikaskaalaa. Kuvassa 10 kojelaudan aikaskaalaksi on asetettu 30 päivää, jolloin Singlestat-paneelit visualisoivat keskiarvoja niiden esittämistä metriikoista kyseisellä aikavälillä.

Vastakohtana keskiarvojen esittämiselle, Singlestat-paneelien avulla erinäisten trendien ja yksittäisten piikkien havaitseminen mitatussa datassa on vaikeaa. Tämän myötä niiden tueksi otettiin käyttöön graafisia paneeleja, jotka myös visualisoivat erinäisiä perusmetriikoita tietojärjestelmästä. Graafiset paneelit reagoivat myös kojelaudalla käytetyn aikaskaalan muutoksiin, ja niiden avulla trendien ja yksittäisten piikkien havaitseminen datan keskeltä on helpompaa. Kyseiset paneelit tarjoavat myös mahdollisuuden luoda erilaisia hälytyksiä.

7.3 Hälytykset ja raja-arvot

Grafana tarjoaa mahdollisuuden lisätä hälytyksiä sekä erilaisia raja-arvoja kojelaudoilla käytettyihin paneeleihin. Uusin versio (versio 5.1) Grafanasta mahdollistaa hälytysten lisäämisen vain graafisille paneeleille, mutta Grafana Labs on ilmoittanut tekevänsä hälytykset mahdolliseksi myös muille paneeleille tulevissa julkaisuissa (Grafana Labs, 2018).

Hälytysten avulla graafisille paneeleille on mahdollista lisätä tietyin väliajoin suoritettavia hälytyssääntöjä. Mikäli hälytyssäännön ehdot täyttyvät, hälytyssääntö lähettää ilmoituksen hälytyksen aktivoitumisesta ennalta määritettyihin sähköpostiosoitteisiin. Ilmoituksen yhteyteen on mahdollista lisätä tekstiä tarkentamaan hälytystä. Tämän opinnäytetyön kirjoitushetkellä Grafana mahdollistaa hälytysten ehtojen muodostamisen vain kyselyjen (Query) avulla. Lisäksi hälytyksissä käytetyt kyselyt eivät saa sisältää Grafanan ympäristömuuttujia (Template Variable).

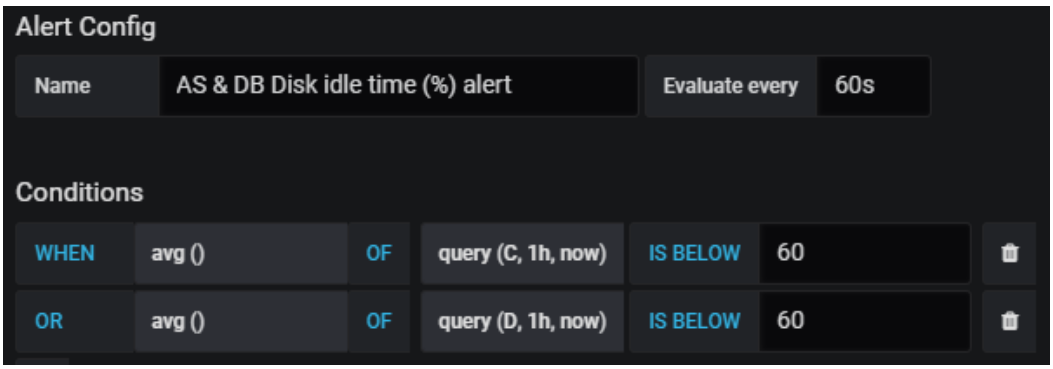
```
avg() OF query(A, 5m, now) IS BELOW 14
```

Kuva 9 - Esimerkki hälytyssäännön ehdosta.

Kuvassa 9 on esitelty esimerkki erään hälytyssäännön ehdoista. Ehto alkaa funktiolla, joka esimerkissä on *avg()*. Funktio määrää kyselyn tuottamien arvojen käsittelytavan. *Avg()*-funktio keskiarvoistaa jokaisen kyselyn tuottaman arvon. Muita funktioita ovat esimerkiksi *min()*, *max()*, *sum()* ja *count()*. Seuraavaksi ehdolle määritellään suoritettava kysely ja käsiteltävä aikahaarukka. Esimerkissä käytetty kysely on kysely A, ja aikahaarukka on viimeiset viisi minuuttia. Lopuksi ehdolle määritetään haluttu raja-arvo, joka kyseisessä esimerkissä on kaikki alle 14 jäävät arvot. Raja-arvon käsittelylle on mahdollista määritellä, tuleeko arvon olla määritellyn rajan alapuolella, yläpuolella, sisäpuolella, ulkopuolella. Ehdon voi myös määritellä täyttyväksi mikäli kysely ei palauta lainkaan tuloksia. Hälytyksen tilan voi myös määritellä muuttumaan haluttuun tilaan sen mukaan, palauttaako kysely lainkaan tuloksia tai ovatko kaikki sen palauttamat tulokset tyhjiä. Samoja sääntöjä voi soveltaa myös silloin, mikäli hälytyksen suorittamisessa tapahtuu virhe.

Hälytyksiä lisättiin Performance Dashboard-kojelaudalle seuraavien perusmetriikoiden yhteyteen:

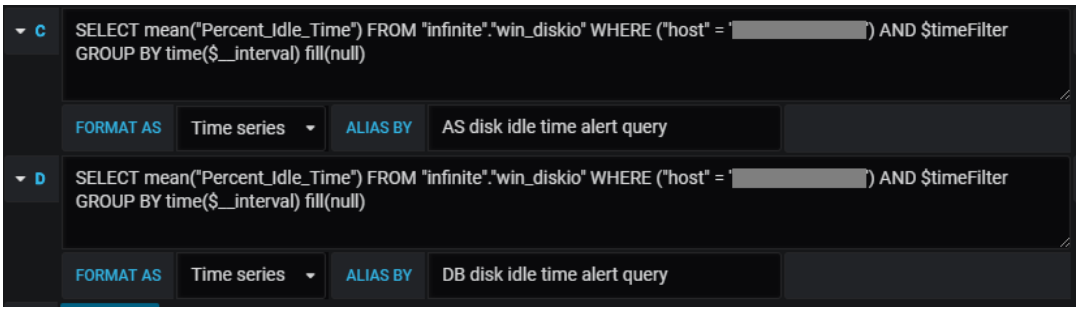
- Suoritinaika prosentteina
- Levyn käyttö prosentteina
- Keskimääräinen aika kirjoitus- tai lukutapahtuman suorittamiseen levyllä
- Tietoverkon kaistan käyttö prosentteina



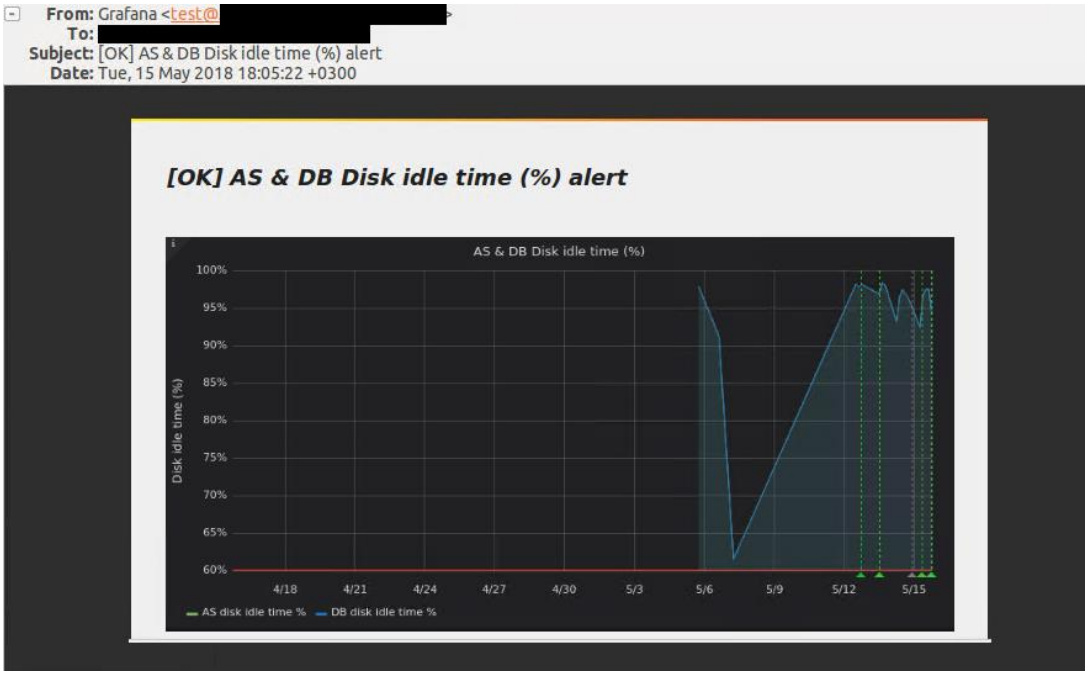
Kuva 10 - Esimerkki lisätystä hälytyksestä.

Kuvassa 10 on esitelty esimerkki kojelaudalle lisätystä hälytyksestä liittyen sovellus- ja tietokantapalvelimen levyn joutokäyntiaikaan prosentteina. Hälytystä varten kirjoitettiin kyselyt C ja D, jotka hakevat keskimääräisen levyn joutokäyntiajan prosentteina sovellus- ja tietokantapalvelimilta.

Ehtojen aikahaarukka on yksi tunti taaksepäin nykyhetkestä, ja kummankaan kyselyn tuottama arvo ei saa olla alle 60 %. Ehto suoritetaan minuutin välein. Mikäli hälytyssäännön ehto täyttyy, Grafana ilmoittaa hälytyksen aktivoitumisesta sähköpostitse ennalta määritettyyn sähköpostiosoitteeseen.

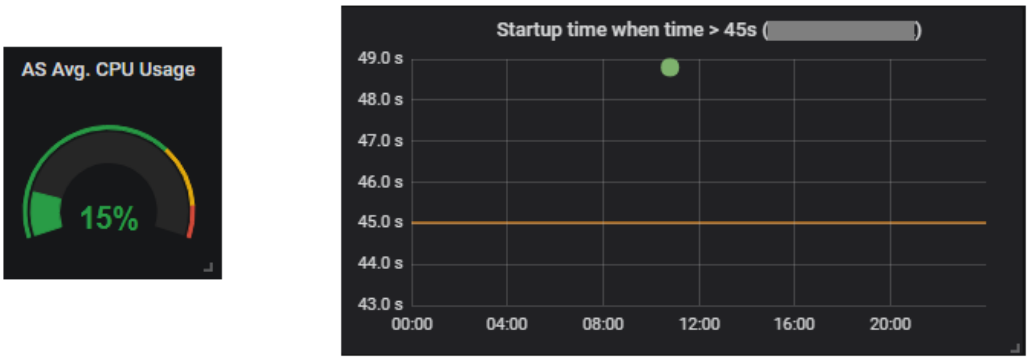


Kuva 11 - Hälytyksessä käytetyt kyselyt C ja D.



Kuva 12 - Esimerkki hälytyksen lähettämästä ilmoituksesta.

Kaikkia raja-arvojen ylityksiä ei suinkaan tarvitse hoitaa pelkkien hälytyksien avulla, vaan niiden lisäksi esimerkiksi Singlestat- ja Graph-paneeleille on mahdollista määritellä erilaisia raja-arvoja auttamaan visualisoinnin tulkitsemisessa.



Kuva 13 - Raja-arvot Singlestat- ja Graph-paneeleilla.

Kuvassa 13 on esitetty kaksi esimerkkiä raja-arvojen määrityksistä. Singlestat-paneeleille raja-arvojen määritys tapahtuu määrittelemällä paneelille kaksi pilkulla eriteltyä arvoa, joiden mukaan rajat määräytyvät. Kuvan esimerkissä raja-arvoiksi on asetettu 70 % ja 90 %, jolloin kaikki alle 70 % jäävät arvot sijoittuvat vihreälle sektorille, 70 % ja 90 % väliin sijoittuvat arvot oranssille sektorille ja yli 90 % sijoittuvat arvot punaiselle sektorille. Kaikki raja-arvojen värit ovat vapaasti muokattavissa.

Kuvan 13 graafiselle paneelille raja-arvoksi on asetettu 45 sekuntia, jolloin paneelille piirtyy viiva kyseiselle kohdalle graafille määritettyjen akselien mukaisesti. Raja-arvoja voi lisätä useampia, ja myös niiden väritykset ovat vapaasti muokattavissa.

Paneeleille voidaan siis määrittää raja-arvoja indikoimaan esimerkiksi suositeltua maksimiarvoa kyselyn palauttamille tuloksille ilman, että kaikista muutoksista tarvitsee ilmoittaa sähköpostitse esimerkiksi järjestelmänvalvojalle. Näin ollen voidaan välttyä yhdeltä monitoroinnin kompastuskiveltä liittyen liiallisiin hälytyksiin. Mikäli hälytyksiä syntyy liikaa, voi niitä vastaanottava henkilö uupua liiallisesta hälyttämisestä, jonka seurauksena voi olla kriittisten ongelmien huomiotta jättäminen. Vastaavanlaisen tilanteen syntyessä hälytyssäännöt tulee arvioida uudelleen.

8 POHDINTA JA JATKOTOIMENPITEET

Digitaalisen tietotekniikan yleistyessä erilaisten tietojärjestelmien sekä niitä ympäröivien infrastruktuurien tilan ymmärtäminen on noussut avainasemaan palveluiden luotettavuuden ja vakauden takaamisen saralla. Järjestelmien terveydentilasta saatu tieto auttaa niistä vastaavia henkilöitä toimimaan oikein tilanteissa, joissa järjestelmän suorituskyky on heikentynyt. Tämän lisäksi tieto voi auttaa vastuuhenkilöitä järjestelmiin kohdistuvien muutosten tekemisessä, olipa kyse sitten esimerkiksi järjestelmän skaalautuvuudesta tai komponentteihin kohdistuvista muutoksista.

8.1 Pohdinta

Opinnäytetyön tavoitteena oli perehtyä sairaalaverkon tietojärjestelmien datan keräämiseen, jäsentämiseen ja visualisointiin. Työn toteutus alkoi visualisointiin sekä monitorointiin liittyvän teorian tutkimisella, jonka jälkeen oli aika soveltaa opittuja tietoja CHA System Performance Dashboard-työkalun kehityksessä. Työn aikana päivitettiin monitorointityökalussa käytetyt tekniikat uusimpiin versioihin sekä tutustuttiin työkalun tarjoamiin mahdollisuuksiin monitorointisovelluksena.

Yleisesti ottaen System Performance Monitoring Dashboard-työkalun tarjoamat visualisoinnit olivat jo valmiiksi hyvällä tasolla. Pääpaino kojelautoihin tehdyissä muutoksissa kohdistuikin kojelautojen kykyyn luoda katselijalleen korkean tason yleiskuva monitoroitavasta kohteesta. Kun yleiskuva tietojärjestelmän tilasta on saatu muodostettua erilaisin visualisoinnin keinoin, on sitä helppo tukea muilla tarkempaa tietoa välittävillä kojelaudoilla ja visualisoinneilla. CHA System Performance Dashboard-työkalu tarjoaa asiakkailleen kattavan kokonaisuuden tietojärjestelmiensä kokonaisvaltaiseen monitorointiin proaktiivisella tasolla.

Kojelautojen muokkaamisen yhteydessä sovellettiin myös kappaleessa 3.2 mainittua visualisoinnin tarkistuslistaa, joka toimi luotettavana ohjenuorana visualisointien laatimiselle. Onnistuneen visualisoinnin edellytyksenä on, että kyseisen tarkistuslistan kriteerit täyttyvät ennen ja jälkeen visualisoinnin laatimisen.

8.2 Jatkotoimenpiteet

Opinnäytetyön jatkotoimenpiteinä on päivitetyn monitorointityökalun testaaminen laajemmassa testiympäristössä. Työ tarjoaa myös ohjenuoran yrityksen sisäisen monitoroinnin päivittämiselle laajalla teoriaosuudellaan.

Yksi varteenotettava vaihtoehto jatkotoimenpiteille on perehtyminen Docker-virtualisointitekniikkaan ja sen tuomiin mahdollisuuksiin monitorointityökalun jakelussa. Dockerin avulla on mahdollista luoda erilaisia sovelluskontteja, jonka myötä monitorointityökalu käyttöjärjestelmineen voitaisiin pakata valmiiseen sovelluskonttiin. Näin ollen asiakkaiden ei enää tarvitsisi asentaa kaikkia monitorointisovelluksessa käytettyjä työkaluja erikseen sillä kaikki löytyisi yhdestä paketista.

LÄHTEET

SALMINEN, R. ja STT. 2017. Laaja tietoliikennekatkos häiritsee Husin tietokoneita – Sairaaloiden toiminta pyritään pitämään normaalina. Yleisradio Oy. [viitattu 2018 – 1 – 28]. Saatavissa: https://yle.fi/uutiset/3-9920602

GE Healthcare Oy. 2018. Kotisivut. [viitattu 2018 – 1 -31]. Saatavissa: http://ww.gehealthcare.fi/fi-FI

KURONEN, Timo. 1998. Tietovarantojen hyödyntäminen ja demokratia. Helsinki: Sitra. Saatavissa: https://media.sitra.fi/2017/02/27173509/sitra174-2.pdf

KOSARA, Robert 2008-07-24. What is Visualization? [viitattu 2018 – 2 – 12]. Saatavissa: https://eagereyes.org/criticism/definition-of-visualization

TUFTE, Edward R. 2001. The Visual Display of Quantitative Information. Connecticut: Graphics Press.

TURNBULL, James. 2016. The Art of Monitoring. New York: Turnbull Press.

TURNBULL, James 2014-10-27. Monitoring Survey. [viitattu 2018 – 2 – 14]. Saatavissa: https://www.kartar.net/2014/10/monitoring-survey/

TURNBULL, James 2015-06-16. Monitoring Survey 2015. [viitattu 2018 – 2 – 14]. Saatavissa: https://www.kartar.net/2015/06/monitoring-survey-2015/

KUUSELA, Vesa. 2000. Tilastografiikan perusteet. Helsinki: Edita.

CAIRO, Alberto. 2013. The Functional Art. Berkley: New Riders.

ALLEN, Arnold. 1994. Computer Performance Analysis with Mathematica. Kalifornia: Academic Press.

Apica AB. 2018. Kotisivut. [viitattu 2018 – 2 – 26]. Saatavissa: https://www.apicasystems.com/

HAMMAR, Sven. 2017. The 5 Most Common Application Bottlenecks. [viitattu 2018 – 2 – 26]. Saatavissa: http://www.apmdigest.com/the-5-most-common-application-bottlenecks

Microsoft TechNet. 2018. Introduction to DPC Objects. [viitattu 2018 – 2 – 28]. Saatavissa: https://docs.microsoft.com/en-us/windows-hardware/drivers/kernel/introduction-to-dpc-objects

Microsoft TechNet. 2008. Monitoring Context Switches. [viitattu 2018 – 2 – 28]. Saatavissa: https://technet.microsoft.com/en-us/library/cc938606.aspx

HAIKALA, Ilkka ja JÄRVINEN, Hannu-Matti 2003. Käyttöjärjestelmät. Helsinki: Talentum.

SEBESTA, Robert 2012. Concepts of Programming Languages. Boston: Addison-Wesley.

JONES, Richard, HOSKING, Antony ja MOSS, Eliot 2012. The Garbage Collection Handbook: The Art of Automatic Memory Management. Florida: CRC Press.

BURKS, Arthur, GOLDSTINE, Herman ja NEUMANN, John 1946. Preliminary Discussion of the Logical Design of an Electronic Computing Instrument. New Jersey: Institute for Advanced Study.

ANTONAKOS, James ja MANSFIELD, Kenneth 2010. Computer Networking from LANs to WANs: Hardware, Software, and Security. Boston: Course Technology.

GE Healthcare Oy. 2018a. Centricity High Acuity Critical Care User Manual. (sisäinen dokumentaatio)

InfluxData. 2018a. InfluxDB 1.5 documentation. [viitattu 2018 – 5 – 5]. Saatavissa: https://docs.influxdata.com/influxdb/v1.5

InfluxData. 2018b. Telegraf 1.6 documentation. [viitattu 2018 – 5 – 5]. Saatavissa: https://docs.influxdata.com/telegraf/v1.6/

Grafana Labs. 2018. Grafana Documentation. [viitattu 2018 – 5 – 5]. Saatavissa: http://docs.grafana.org/

Postfix. 2018. The Postfix Home Page. [viitattu 2018 – 5 – 5]. Saatavissa: http://www.postfix.org/

Double Precision Inc. 2018. Courier Mail Server. [viitattu 2018 – 5 – 5]. Saatavissa: http://www.courier-mta.org/