TEKNILLINEN TIEDEKUNTA

MURTOSOKASTA SPOF-ILMIÖÖN

Mikko Paavola

KONETEKNIIKAN TUTKINTO-OHJELMA

Kandidaatintyö 2018

TIIVISTELMÄ

Murtosokasta SPoF-ilmiöön

Mikko Paavola

Oulun yliopisto, Konetekniikan tutkinto-ohjelma

Kandidaatintyö 2018, 26 s.

Työn ohjaaja: DI Jouko Heikkala

Yllättävät koneen tai muun teknisen järjestelmän vikauttavat ongelmat ovat tapauskohtaisia, mutta niiden ratkaisua on mahdollista ideoida tekniikan eri alojen käytännöistä. Tässä työssä tarkastellaan analogian käyttöä ja mahdollisuuksia tutkimusmenetelmänä kahden tekniikan eri alan ongelman ratkaisuissa.

Esimerkkinä opinnäytetyössä käytetään perämoottorin murtosokkaa, joka estää ulkopuolisen häiriön leviämisen moottoriin. Tiedonsiirto-ongelmissa yllättäviä, joskus jopa koko järjestelmän lamauttavia, yhdessä kohdassa esiintyviä virheitä kutsutaan SPOF-ongelmaksi.

Opinnäytetyössä otetaan kirjallisuudesta esimerkkejä sekä perämoottorin murtosokan että SPOF-ilmiön ominaisuuksista. Lähtökohtana on, että murtosokka on ratkaisu ongelmaan, SPOF-ilmiö puolestaan on ongelma, jonka ratkaisemisessa voitaisiin käyttää murtosokan toiminnallisia ominaisuuksia eli niin sanottuja "ohjelmallisia murtosokkia".

Asiasanat: analogia, murtosokka, SPOF, tiedonsiirto, voimansiirto

ABSTRACT

From Shear Pin to the SPoF phenomenon

Mikko Paavola

University of Oulu, Degree Programme of Mechanical Engineering

Bacherlor’s thesis 2018, 26 p.

Supervisor: M.Sc. (Tech.) Jouko Heikkala

The shear pin of the outboard motor is very useful protection against greater damages, if something hits the propeller. In other words shear pin is preventing and isolating these errors spreading to the rest of the system. Acronym SPOF (Single Point of Failure) describes the situation, when the single point fails and according to that the entire system will be broken at least awhile.

In this thesis the analogical reasoning is used to find out, if it possible to support conclusion that SPOF problems of the data communication in the computer networks are possible to prevent by the "programmable shear pin".

Few similarities are highlighted between power transmission and data communication. These similarities are used to justificate the use of Intel Management Engine platform to provide location for these "programmable shear pins".

Keywords: analogic, communication, shear pin, SPOF, transmission

ALKUSANAT

Pitkän työurani aikana ICT-alalla olen joutunut tarkastelemaan rakennettavia tietojärjestelmiä myös konetekniikan silmälasien läpi. Hyvän koneen tai ohjelman määritelmä ei mielestäni ole vain se, että se toimii, vaan että se myös rikkoutuessaan rikkoutuu ennakoitavalla ja turvallisella tavalla. Kandidaatintyössäni pyrin soveltamaan tätä ideaa tiedonsiirtotekniikkaan ja siinä yleiseen SPOF-ongelmaan analogian keinoin.

Tietotekniikkaan, erityisesti tiedonsiirron ja tiedonhallinnan kysymyksiin ajauduin konetekniikan ydinalueesta nimenomaan Oulun yliopiston silloisesta konetekniikan osastosta käsin. Osastolle oli hankittu aikanaan edistyksellinen CAD/CAM -laitteisto. Sain sen käyttöön varsin vapaat oikeudet ja eräin ajoin vastasinkin sen virheettömästä toiminnasta.

Nyt opiskelumaailman paluumuuttajana kiitän työni ohjaajaa Jouko Heikkalaa siitä, että hän on mahdollistanut työurani aikana hankkimani tietotaidon yhdistämisen konetekniikan opiskeluun.

Oulu, 22.1.2018

Mikko Paavola

SISÄLLYSLUETTELO

MERKINNÄT JA LYHENTEET

A	poikkipinta-ala
d	halkaisija
F	leikkausvoima
i	leikkauspintojen lukumäärä
π	ympyrän kehän suhde ympyrän halkaisijaan eli pii
τ	leikkausmurtolujuus
AMT	Intel® Active Management Technology
CSMA/CA	Carrier Sense Multiple Access With Collision Avoidance
CSMA/CD	Carrier Sense Multiple Access With Collision Detection
Ethernet	pakettipohjainen lähiverkkoratkaisu
HTTP	Hypertext Transfer Protocol
HTTPS	Secured Hypertext Transfer Protocol
IoT	Internet of Things
JeOS	Just Enough Operating System
ME	Intel Management Engine
MTBF	Mean Time Between Failures
OOB	Out-of-band Management
rootkit	computer program running under system
RSA	julkisen avaimen salausalgoritmi
SPF	Single Point Failure - Nasan käyttämä lyhenne
SPOF	Single Point of Failure
TCP	Transmission Control Protocol

1 JOHDANTO

Tekniikan eri osa-alueilla ennakoimattomasti ilmenevä, eskaloituva ja siksi toiminnan kannalta fataali vika tai häiriö on ongelma, jota on pyritty ratkaisemaan eri tavoin. Koneiden ja laitteiden osalta ongelmanratkaisut riippuvat sekä koneiden ominaisuuksista että niiden käyttökonteksteista. Yksinkertainen mutta tehokas ratkaisu useissa pienkoneissa on voimansiirtohäiriön laajenemisen estävä murtosokka, murtotappi tai katkopultti.

Tietotekniikassa ja yleensä korkean riskin teknologioissa kyseisen ongelman ratkaisuja on niin ikään monenlaisia ja teholtaan vaihtelevia (Perrow 1999, s. 353). Etenkin laajoissa tietojärjestelmissä, nimenomaan tiedonsiirtoon liittyvissä ongelmissa ratkaisut ovat valitettavan usein olleet tehottomia. Esimerkiksi julkishallinnon ja -talouden tiedonsiirtojärjestelmässä ilmenevä äkillinen ja ennustamaton vika voi aiheuttaa suurta haittaa levitessään hallitsemattomasti, kuten viime aikojen uutiset toistuvasti kertovat. Yksittäiset tietotekniset laitteet tulevat koko ajan luotettavammiksi. Pitkäaikaisen ICT-alan kokemukseni perusteella olen huomannut, että sen sijaan väärin suunnitelluista ja toteutetuista ohjelmista, protokollista ja ulkopuolisesta häirinnästä aiheutuvat ennustamattomat viat korostuvat fyysisten laitevikojen sijaan. Ne aiheuttavat sen, että järjestelmien ylläpitämä tiedonsiirto tai järjestelmäintegrointi vaikeutuu ja vikojen paikallistaminen ja korjaaminen hankaloituu.

Tämän kandidaatintyön toinen tarkasteltava elementti, murtosokka erityisesti perämoottoreissa, on nimenomaan voimansiirtoon liittyvän ongelman ratkaisu. Toinen elementti, SPOF (single point of failure), on tietoteknisissä järjestelmissä esiintyvä yksittäinen piste tai kohta, jonka virheellinen toiminta tai toimimattomuus aiheuttaa koko siihen liittyneen järjestelmän toimimattomuuden. Tämä on edelleen ongelma vailla omaa tehokasta vian eskaloitumisen estävää ratkaisuaan, ”murtosokkaansa”. Pyrkimyksenä onkin tarkastella analogia viitekehyksenä, onko murtosokassa sen kaltaisia ominaisuuksia, joiden periaatteita on sovellettu tai voitaisiin laajemmin soveltaa SPOF-ilmiön ratkaisuissa.

Analogiapäättelyllä ei kuitenkaan ensisijaisesti pyritä varsinaisesti ratkaisuun. Sen avulla voidaan vain saada uusia näkökulmia, generoida uusia ideoita ja ajatuksia. Tekniikkaan liittyvissä tutkimuksissa sitä ei yleensä ilmaista eksplisiittisesti, vaikka sen voi arvioida usein olleen taustalla innovaatioissa: tutusta ja tiedetystä ominaisuudesta tai ilmiöstä on johdettu uutta aivan toisenlaisillakin toiminta-alueilla.

Analogiassa on paljolti kyse termeistä, käsitteistä, rinnastuksista yksinkertaisista monimutkaisiin, tutummista tuntemattomiin ja päinvastoin. Analogian käyttö on aina valikoivaa. Ei tarkastella kaikkea, vaan vain osaa ominaisuuksista.

Perimmältään on kyse päätöksentekomekanismista tilanteissa, joissa ei tiedetä etukäteen kaikkea, vaan joudutaan etenemään tehtyjen analogioiden kautta, suppeammin ilmaistuna esimerkiksi eri tieteenalojen osaamisen pohjalta tehtyjen simulaatioiden avulla. *"Tällaisen päätöksentekokyvyn saavuttaminen edellyttää hyvin laaja-alaista osaamista muun muassa matematiikassa, mekaniikassa, materiaalitekniikassa ja tietotekniikassa"* (Niemi 2018).

2 ANALOGIA TEOREETTISENA VIITEKEHYKSENÄ

Analogia tarkoittaa muun muassa asioiden, objektien, ilmiöiden ja niiden suhteiden samankaltaisuutta (Nurmi et al. 2004, s. 28 ; Niiniluoto 1983, s. 29). Tieteenteoriassa analogiapäättelyllä pyritään ensin hahmottamaan olioiden x ja y yhteiset ominaisuudet. Näistä voidaan tietyin edellytyksin päätellä, että mikäli x:llä on ominaisuus B, niin y:lläkin täytyy olla ominaisuus B. (Haaparanta & Niiniluoto 2016, s. 103 - 104) Samankaltaisuuksiin liittyvien havaintojen ymmärtämisessä käytetään usein malleja. Analogiamallit ovatkin hyvin yleisiä tekniikassa, vaikka kyseisen termin sijaan käytetään muun muassa termejä "simulaatiomalli" tai "pienoismalli". (Niiniluoto 1984, s. 206) Pienois- tai simulaatiomalleja käyttäen saadaan tuntumaa monimutkaistenkin ongelmien hahmottamiseen, muistaen kuitenkin, että mitään ehdottomia vastauksia ei välttämättä saada (Niiniluoto 1984, s. 206 - 207 ; Kauppinen & Pietarinen 1989, s. 373). Usein käytännön ongelmia ratkaistaessa onkin tärkeää saada edes summittainen käsitys asioiden ja ominaisuuksien välisistä vuorovaikutussuhteista.

Tässä opinnäytetyössä analogiaa käytetäänkin, ei todistamaan luotettavasti tiettyä vastaavuutta, vaan antamaan uusia näkökulmia tietynlaisen ongelman heuristiseen lähestymis- ja ratkaisutapaan. Tämä tapa on vastakkainen algoritmiselle ongelman lähestymistavalle, jossa tarkasti määritellyillä ja matemaattisesti todistetuilla askelilla päädytään todistamaan vastaavuus alku- ja lopputilojen välille (Knuth 2000, passim).

Tässä kandidaatintyössä pyritään käyttämään analogiaa työkaluna pragmaattisesta näkökulmasta. Tällöin analogia puhtaana tieteenteoriana jää taustalle, eikä oteta suoraan kantaa vertailussa siihen, onko kyseessä heikko tai vahva analogia. On ainoastaan pyritty valitsemaan vertailtavista kohteista tavalla tai toisella merkittäviä ja ymmärrystä lisääviä ominaisuuksia. Analogian, kuten yleensä vertailun yhteydessä on ongelmana, että hypoteesi huomaamatta muuttuu faktaksi. Tätä on tietoisesti pyritty välttämään.

3 MURTOSOKKA

3.1 Murtosokasta yleisesti

Murtosokka koneissa ja laitteissa on pieni ja huomaamaton kone-elin, itsestäänselvyys, jonka täsmällistä määritelmää on vaikea löytää. Sen esinebiografia on puutteellisesti dokumentoitu. Ei esimerkiksi tiedetä tarkasti milloin, missä ja minkä tyyppisessä koneessa tai laiteessa sitä on käytetty ensimmäisen kerran. Murtosokasta voidaan käyttää myös nimitystä murtotappi tai katkotappi (Puolimoottorit 2017, 42 - 45; Rider 2017, 37-39). Murtosokkaa käytetään koneissa sallimaan tai estämään tietty liike, mikäli ennalta määrätty voima kohdistuu murtosokkaan se murtaen.

Murtosokalle ominaista on selkeä ongelmakohdan paikallistaminen, ongelman eskaloitumisen estäminen sekä halutun toiminnan nopea, yksinkertainen ja helppo palauttaminen. Edellytyksenä murtosokan tehokkaalle, nopealle ja yksinkertaiselle vaihtamiselle on se, että itse kone on suunniteltu myös tätä toimintaa silmällä pitäen.

Tekniikan käsikirjassa yksinkertaisimpana kone-elimenä mainitaan liitoselimien osalta niittiliitos. Murtosokkaa (kuva 3) voidaan tarkastella yksinkertaisena niittinä, joka mitoitetaan ainoastaan leikkausmurtolujuuden suhteen seuraavasti.

Tappiliitoksessa yhden tai kahden pinnan leikkausvoima (Leinonen 1972, s.315) saadaan kaavasta 1,

$$F \geq \tau \mathrm{A} \qquad (1)$$

missä τ on leikkausmurtolujuus, A on materiaalin leikkauspinta-ala ja F on murtumisen aiheuttava ulkoinen voima.

Koeolosuhteissa leikkausmurtolujuus voidaan laskea kaavasta 2,

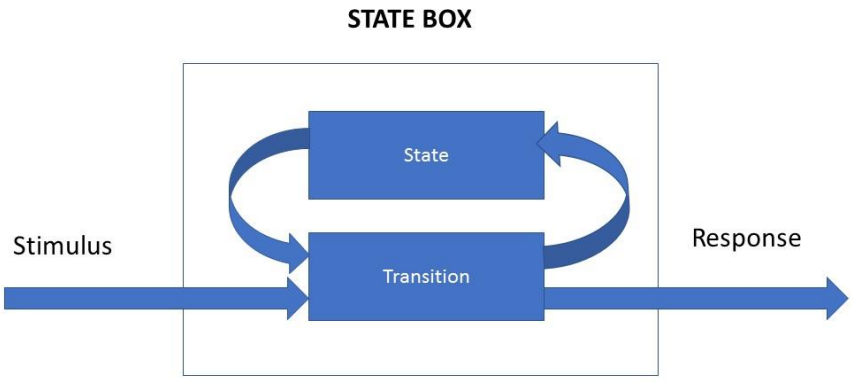
$$\tau = \frac{4F}{\pi d^2 i} \qquad (2)$$

missä leikkausmurtolujuus on τ, F on leikkausvoima N, d on sokan halkaisija ja i on rakenteesta johtuva arvo eli moneenko leikkauspintaan leikkausvoima kohdistuu yhtä-aikaisesti.

Leikkausvoima, jonka murtosokka kestää ennen murtumista, on siis :

$$F = \frac{\tau \pi d^2 i}{4} \qquad (3)$$

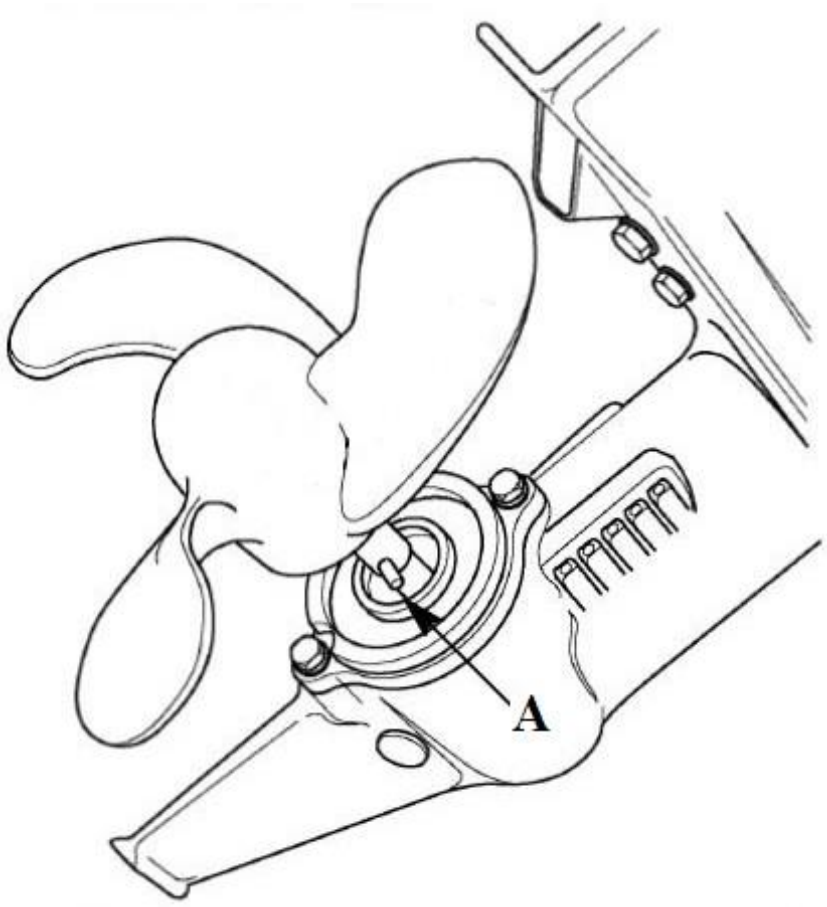
Kaavion (kuva 1.) laatikossa (State Box) sisääntuleva voima F (Stimulus) siirtyy ulos tai häviää kokonaan (Response). Toimintoa ohjaa tilan (State) raja-arvo x. Jos raja-arvo x ei ylity, niin tila ei muutu. Jos raja-arvo x ylittyy, tilaksi (State) tulee havaittava virhe ja viesti tai tieto siitä. Transition kuvaa sitä toimintoa, jonka murtosokka suorittaa murtuessaan.



Kuva 1. Murtosokan toiminnallinen kaavio (mukaillen Pressman 2005, s. 835; Prowell et al. 1999, s. 10; Stavely 1999, s. 210).

3.2 Murtosokka perämoottoreissa

Perämoottorissa potkuriin kohdistuva ulkopuolisen voiman välittyminen perämoottorin muihin voimansiirtoelimiin halutaan estää niitä suojellen murtosokan avulla. Murtosokka voi toimia myös toisin päin: moottorista tuleva voima on liian suuri ja murtosokka murtuu jo pelkästä moottorin aiheuttamasta liiallisesta vääntömomentista. Murtosokka suojelee moottoria myös silloin, kun itse moottori ei ole toiminnassa.



Kuva 2. Murtosokan (A) sijainti perämoottorissa (mukaillen, Shear Pin 2018).

3.3 Murtosokan ominaisuuksia

Analogiatarkastelua varten murtosokasta on valittu neljä ominaisuutta, joita sovelletaan tiedonsiirtoon liittyvän SPOF:in ratkaisumahdollisuuksiin.

Yksinkertaisuus

Kuva 3. Yamaha 20 hv perämoottorin murtosokkia, kooltaan 35 mm x 4,96 mm.

Murtosokka on hyvin yksinkertainen ja selkeä toiminnaltaan (kuva 2). Voidaan ajatella, ettei se osallistu itse voimansiirron funktioon, muulloin kuin vasta murtumisen aikana. Vaikka murtosokka on periaatteessa yksinkertainen ja helposti ymmärrettävä, ovat vaatimukset hyvin toimivalle murtosokalle moninaiset. Korroosiokestävyys, muokkauslujittuminen, haurasmurtuma, väsymislujuus, venymän minimointi, myötövanheneminen, kimmoinen ja plastinen muodonmuutos, muodonmuutoskyky, iskusitkeysominaisuudet, viruminen, lämpötilaneutraalius ovat ominaisuuksia, joiden kesken pitää löytää riittävät ja keskenään toimivat kriteerit (Salokangas 1973, s. 215-226).

Riippumattomuus

Potkurin ja vetoakselin geometrian lisäksi murtosokka ei aiheuta mitään vaatimuksia muulle voimansiirrolle. Edellä mainittujen hyvän murtosokan ominaisuusvaatimuksista huolimatta murtosokka voidaan hätätilanteessa tilapäisesti korvata vaikka rautanaulan pätkällä tai hammastikuilla. Lähes mikä tahansa materiaali käy tuolloin korvaamaan käytetyn aidon murtosokan. Ainoastaan raja-arvona käytettävä, sivuilla 10 - 11 esitetty leikkausmurtolujuus muuttuu.

Huollettavuus

Murtosokan murtumisen havaitseminen ja sokan vaihtaminen ei vaadi teknistä koulutusta tai osaamista. Itse sokan murtuminen on selkeä todiste leikkausmurtolujuuden ylittymisestä. Murtosokan vaihtaminen onnistuu ilman tarvittavia erikoistyökaluja. Murtosokan kehittäminen itsenäisesti vastaamaan paremmin käyttöolosuhteita voi jatkua tuotteen koko kehityskaaren ajan. Vuosihuollon yhteydessä murtosokka voidaan vaihtaa helposti uuteen parempaan versioon täysin riippumatta käytetyn perämoottorin versiosta.

Monistettavuus

Murtosokka esineenä on helposti monistettavissa ja varakappaleet mukana kuljetettavissa. Monistettavuudessa ei tässä tarkastella kuitenkaan itse murtosokan monistettavuutta, vaan voimansiirron monistettavuutta: murtosokan toiminnallinen ympäristö, yhden perämoottorin voimansiirto, ei ole monistettavissa. Periaatteessa voitaisiin tietysti käyttää varaperämoottoria, mutta tämän kaltainen varautuminen on nyt tehtävän tarkastelun ulkopuolella.

4 SPOF

4.1 Yleistä

SPOF:illa tarkoitetaan teknisen järjestelmän haavoittuvuutta yhden pisteen suhteen (Wikipedia b 2017). SPOF-ilmiö on laajasti tekniikassa käytetty termi, mutta tässä työssä se rajataan koskemaan vain tiedonsiirrossa tapahtuvia virheitä. Tietokonetekniikassa yllättävää ja odottamatonta virhettä pyritään estämään muun muussa monistamalla järjestelmän komponentteja, luomalla rinnakkaisia järjestelmiä ja reitittämällä verkkoja useampaan kertaan (Technet 2017). Tälle onkin perusteensa laitteiden fyysisen rikkoutumisen takia. Rikkoutunut laite pitää havaita ja vaihtaa uuteen. Vaihdossa on arvioitava esimerkiksi kokonaistaloudellinen vaikutus rikkoutumisen ja vaihtokustannusten välillä koko tuotannolle. Laitteiden ja varsinkin tietokonelaitteiden MTBF (MTBF 2016) eli keskimääräinen vikaantumisaika on pidentynyt radikaalisti ensimmäisistä tietokoneista lähtien. Ohjelmalliset ja varsinkin tiedonsiirtoviat eivät ole samalla tahdilla vähentyneet, kuten uutisista voidaan todeta.

Ohjelmoinnissa on johtavana ajatuksena oletus, että kaikki oman ohjelmakoodin ympärillä toimii hallitusti ja oikein. Oletetaan esimerkiksi kunkin systeemialiohjelman tekevän dokumentaation mukaan: aliohjelma suorittaa, mitä sen on määrä suorittaa tai se palauttaa määritellyn virhekoodin funktion arvona tai keskeytyksenä. Siis yleensä ei oleteta aliohjelman tekevän mitään odottamatonta ja ennalta arvaamatonta. Jo tietojärjestelmän suunnitteluvaiheessa otetaan mukaan vain ne asiat, jotka voidaan ennakoida. Joskus niistäkin otetaan vain ne, jotka lopulta katsotaan etukäteen merkityksellisiksi. Premissinä on piilossa olevan järjestelmän toimivuus ja sen toimimattomuus jätetään huomioimatta ja manuaalisen käsittelyn alaiseksi.

Vastaava tilanne perämoottoreissa olisi: ”Anna mennä vaan, ei siinä kartan mukaan mitään kiviä ole”. Vaikka vettä olisikin tarpeeksi, ei karttaan ole uppotukkeja kuitenkaan uiton jäljiltä merkitty.

4.2 Alustana Intel Management Engine

Perämoottorissa murtosokalle pitää olla oma selkeä suunniteltu paikka, mihin se asennetaan vaikuttamalla mahdollisimman vähän muutoksia muuhun olemassa olevaan rakenteeseen. Tässä rajataan tarkastelu tiedonsiirron puolella Intelin prosessoreihin, joissa jo viidentoista vuoden ajan on ollut paikka, jossa voidaan ajaa "ohjelmallisia murtosokkia" ilman vaikutusta tai riippuvuutta jo olemassa oleviin suojauksista johtuviin rajoituksiin, käyttöjärjestelmiin tai ohjelmiin. Tämä paikka on Intel Management Engine (ME).

Intelin ME on oikeastaan itsenäinen keskusyksikkö CPU, jossa toimii oma JeOS eli kevyt mikrokernel (Intel a 2017). Se on täysin itsenäinen alisysteemi, joka pyörii monissa eri kokoisissa Intelin prosessoreissa jo valmiina (Intel b 2017). Voidaan puhua jopa omasta itsenäisestä käyttöjärjestelmästä, joka toimii varsinaisen käyttäjälle näkyvän käyttöjärjestelmän esimerkiksi Linuxin ja Windowsin alla (Intel c 2017; Zammit 2016). ME latautuu käyttövalmiiksi ennen varsinaista käyttöjärjestelmää, valvoo vapaasti systeemimuistia, näyttöä, näppäimistöä, hiirtä ja kaikkia fyysisesti tietokoneeseen liitettyjä oheislaiteliityntöjä (Portnoy & Eckersley 2017). Tärkeimpänä tämän työn kannalta on, että se myös hallitsee verkkoliikennettä (Gillespie 2012). ME:llä on oma TCP/IP pino (Cimpanu 2016). Intelin käyttämä käyttöjärjestelmä on Tanenbaumin (Tanenbaum 2018) mukaan Minix 3. Minix 3 on mikrokernelpohjainen pieni ja nimenomaan itsekorjautuvuuteen pohjautuva käyttöjärjestelmä (Herder et al. 2006, s. 80 - 88). Kyseisen käyttöjärjestelmän alkuperäisen tekijän lisenssiehdot mahdollistavat rajattomat mahdollisuudet sovittaa kyseistä lähdekoodia miltei mihin tarkoitukseen tahansa (BSD License 2005). Melkeinpä ainoa kriteeri on, että alkuperäisen tekijän nimi täytyy käydä ilmi levitettävästä suoritettavasta ohjelmakoodista, vaikka kryptattunakin.

Kaikki edellä esitetty mahdollistaa ja tekee ideaaliseksi "ohjelmallisen murtosokan" sijoittamisen Intel ME:n alaisuuteen.

4.3 SPOF ja IoT

IoT eli Internet of Things lisää radikaalisti itsenäisten keskenään kommunikoivien tietokoneiden määrää tietoverkossa ja varsinkin keskenään viestivien tietokoneiden määrää (IoT 2017). Samalla tiedonsiirtoviat lisääntyvät, koska Ethernet perustuu CSMA/CD-periaatteelle, joka teoriassa ja myös käytännössä hukkaa viestejä mahdollistaessaan monen toimijan yhtäaikaisen verkon käytön tiedonsiirtoon. Pakettien törmäysten havaitsemisen jälkeen tapahtuu vain uudelleenlähetys satunnaisesti valitun ajan kuluttua. (CSMA/CD 2013) Viestipakettien yhteentörmäysten välttämiseksi voidaan toki valita suunnitteluperiaatteeksi CSMA/CA -menetelmä (CSMA/CA 2013). Tässä menetelmässä voi virhe tapahtua kanavan varauksen tai sen purun yhteydessä. Olipa kyseessä mikä tahansa menetelmä tai prosessi, niin aina jokin osa siitä voi tapahtua tai olla tapahtumatta. Siksi jos prosessissa on mahdollisimman vähän askelia, joiden täytyy seurata toisiaan kokonaisuuden takia, niin sitä vähemmän siinä on myös virheherkkiä kohtia.

Esimerkiksi asynkroninen tiedonsiirto ei odota saavansa mitään vastausta vastaanottavalta taholta, eli se ei jää odottamaan. Siksi on syytä kyseenalaistaa Coulourisin, Dollimoren ja Kingbergin väite, että asynkronisessa tiedonsiirrossa vastaanottajan synkronisesti toimivalla tiedonsiirrolla ei olisi negatiivisia vaikutuksia itse tiedonsiirtoon. Ainoana kriteerinä asynkroniselle vastaanotolle he esittävät itse ohjelman teon monimutkaisuuden ja sen takia sitä ei ole otettu käyttöön nykyisissä tiedonsiirtojärjestelmissä. (Coulouris et al. 2005, s. 134) Käyttäjät voivat itse todeta tämän vastaanottopään virhetilanteen surffatessaan netissä. Tällöin selain näyttää jumiutuneen, vaikka selain ei tee mitään muuta kuin odottaa seuraava prosessin osaa tapahtuvaksi. Yleisesti ottaen asynkronisessa tiedonsiirrossa olisi selkeämpää tarkoittaa sekä lähettävän että vastaanottavan osapuolen ei-blokkaavaa eli asynkronista käyttäytymistä. Tämä sama asia tulee esille Coulourisin, Dollimoren ja Kingbergin kirjoituksessa vikasietoisista palveluista, jossa jätetään vikasietoisuuden tarkastelun ulkopuolelle itse tiedonsiirron virheet (Coulouris et al. 2005, s. 615). Voidaan tietysti väittää, että tiedonsiirtovirheet eivät kuulu palvelulle itselleen. Mutta jos palvelua ei voida käyttää asynkronisella tavalla, vaan ainoastaan synkronisella tavalla, niin se lienee palvelun vika.

4.4 Tarkasteluun valitut ominaisuudet

Yksinkertaisuus

"Ohjelmallinen murtosokka" on helppo käsittää ohjelmamoduuliksi, joka katkaisisi yksinkertaisesti koko tiedonsiirron laittamalla talteen kyseisen virhetilanteen ja siirrettävän tiedon tulevaa kehitystyötä varten (Hunt et al 2011, s. 189). "Ohjelmallinen murtosokka" täyttäisi toiminnallisesti kuvan 1 vaatimukset, kun vaihdetaan voiman F (Stimulus) tilalle tiedonsiirrossa liikkuva viesti ja tilan (State) tilalle todettu virhetilanne ja itse viestin sisältö. Tiedonsiirto katkaistaisiin pysäyttämällä kyseinen tiedonsiirrosta vastaava ohjelmaosa ja korvaamalla se uudelleen käynnistetyllä osalla. "Ohjelmallisen murtosokan" tehtävä on määritelty yksinkertaisesti ja selkeästi. Murtosokassa kuten tässäkin, itse "ohjelmallisen murtosokan" sisäiseen rakenteeseen ei oteta kantaa.

Riippumattomuus

SPOF-ilmiön estävän "ohjelmallisen murtosokan" ominaisuuksien tarkastelua varten valitussa murtosokan paikassa eli Intel ME-ympäristössä se on täysin riippumaton eri käyttöjärjestelmien versioista, toimivuudesta ja niiden käyttöoikeuksista. Toisaalta mikään virhe tai oikeakaan toiminta käyttöjärjestelmässä tai sen alaisissa ohjelmissa ei voi vaikuttaa "ohjelmalliseen murtosokkaan" suoraan. Yhdessä "ohjelmallisessa murtosokassa" mitataan vain yhtä tiettyä parametria, jolla on aseteltavissa oleva raja-arvo. ME:n rakenteen vuoksi näitä murtosokkia voi olla useita erilaisia. Itse ohjelman sisäinen rakenne ei ole välttämättä yksinkertainen, ainoastaan sen tehtävä on yksinkertainen ja suoraviivainen. Tapauskohtaisesti sen tehtävä on poistaa raja-arvon vaihtuessa tai ylittyessä virheen tai virhekäyttäytymisen tapauskohtaisesti aiheuttama säie, prosessi, aliohjelma, kirjasto, laiteajuri tai koko käyttöjärjestelmä (Siberschatz 1998, passim). Poistettu osa voidaan ladata muistiin ja käynnistää uudestaan itse laitteella tai verkossa olevasta muistivälineestä.

Huollettavuus

"Ohjelmallista murtosokkaa" ei tarvitsisi vaihtaa, ellei sitten halutaisi vaihtaa sitä paremmin ajoympäristöön sopivaan versioon itse murtosokasta. Tämänkin vaihtamisen ME voi suorittaa, vaikka itse tietokoneen käyttöjärjestelmä ei olisi kunnossa (Cimpanu 2016). Sisäinen ME-käyttöjärjestelmä tekee tarvittavat eri tasoiset "ohjelmallisten

murtosokkien" versioiden vaihto-operaatiot täysin itsenäisesti. ME-käyttöjärjestelmä vaihtaa sisäisesti vikaantuneen ohjelmallisen murtosokkamoduulin automaattisesti pysäyttämällä sen ja käynnistämällä sen uudestaan. Toimenpiteitä varten tietokoneen ei tarvitse olla päällä, ainoastaan kytkettynä verkkoon ja virtalähteeseen. Kaikki tarvittavat toimenpiteet toimintojen uudistamiseen ja vaihtamiseen voidaan tehdä ulkopuolelta ilman käyttäjän omia toimenpiteitä tai edes kykyä estää sitä.

Monistettavuus

Päinvastoin kuin voimansiirrossa tiedonsiirrossa tieto voi olla useammassa paikassa samanaikaisesti. Siirrettävä tieto A:lta poistuu vasta, kun tieto on varmasti B:llä. Tähän päästään ainoastaan varmistamalla asia. Tiedonsiirtoketju voi myös olla paljon pitempi ja monimutkaisempi ja usein käytännössä onkin. Tällöin tieto säilyy kussakin aktiivisessa tiedonsiirtopisteessä erikseen ja poistuu vasta, kun yksimielisyys pisteiden kesken tiedonsiirron onnistumisesta on syntynyt. Tämä on protokolla-asia eikä siten suoranaisesti kuulu tämän työn piiriin. Tämä tiedonsiirron ominaisuus helpottaa ohjelmallisen murtosokan tehtävää, koska se voi huoletta poistaa virhekohdan ja olla samalla kuitenkin varma, ettei kyseinen viesti katoa sen ollessa kopioituna useammalle tietokoneelle. Uudelleenlähetys on tarpeeton, koska kokonaisjärjestelmällä on tarvittava tieto jo olemassa ja tiedonsiirto voi jatkua heti, kun "ohjelmallinen murtosokka" on tehnyt tehtävänsä.

5 MURTOSOKKA JA SPOF ANALOGIAN NÄKÖKULMASTA

5.1 Perustelut vastaavuuksille

Tarkoituksena oli löytää ominaisuuksia sekä perämoottorin murtosokasta että tiedonsiirtoon liittyvien SPOF-ilmiön korjausmahdollisuuksista, jotka vastaisivat toisiaan tietyllä korkealla abstraktiotasolla ilman teknisiä yksityiskohtia. Näissä molemmissa on lukematon määrä ominaisuuksia, jotka olisi voitu valita tarkastelukohteiksi. Valintaan on vaikuttanut esioletus ja omakohtainen kokemus aihepiiristä; näitä kahta on hyvin vaikea perustellusti erotella toisistaan. Esioletus perustuu siihen, että nämä valitut ominaisuudet ovat niitä, joita minkä tahansa järjestelmän suunnittelussa kannattaa tavoitella, jotta ennakoimattomat virheet hallittaisiin mahdollisimman sujuvasti.

5.2 Valittujen ominaisuuksien tarkastelu

Molemmissa tarkastelukohteissa eli murtosokassa ja "ohjelmallisessa murtosokassa" toteutuivat yksinkertaisen ratkaisun vaatimukset: uudelleentestausta ei tarvita, vaikka ympäristö muuttuisikin. Molemmilla ratkaisun sijaintipaikka mahdollistaa suuren riippumattomuuden muusta järjestelmästä. Kuitenkin sijainti myös mahdollistaa annetun tehtävän suorittamisen siinä laajuudessa, missä se on tarpeen. Pelkästään tiedonsiirto-ongelmia varten ei tarvita erikoishenkilöstä vikoja korjaamaan, kuten ei myöskään perämoottorin murtosokan murtuessa.

Monistettavuus on lähinnä esimerkkinä ominaisuudesta, joka ei vertaudu siirtojärjestelmien välillä. Monistettavuuden osalta tarkastelukohteet siis eroavat toisistaan fyysisten ominaisuuksiensa perusteella. Voimaa ei voi monistaa useampaan kohtaan samanaikaisesti ja käyttää sitten uudelleen tarvittaessa. Monistettavuus ja sen helppous antaa puolestaan suuren edun tiedonsiirron varmistamiseen, mutta tämä on valitun protokollan asia, eikä "ohjelmallisen murtosokan" asia.

6 JOHTOPÄÄTÖKSET

Periaatteessa analogia voi antaa uusia näkökulmia niihin tekijöihin, jotka ovat oleellisia jopa keskenään erilaisissa teknisissä järjestelmissä, vaikkakaan eivät suoraan verrannollisia keskenään. Analogiapäättelyllä, jossa tutusta ja tiedetystä edetään uuteen hypoteesiin, tietoon tai ratkaisuun, voidaan hyödyntää toisella tieteenalalla hyväksi koettuja suunnittelutapoja ja menetelmiä. Tässä työssä tuo tuttu ja tiedetty on perämoottorin murtosokka. Se on hyväksi koeteltu, joskaan ei ainoa keino estää yllättävän ulkopuolisen häiriön aiheuttaman vian laajeneminen koko moottoriin. Murtosokan hyvät ominaisuudet on tässä työssä pyritty siirtämään tiedonsiirto-ongelmien (SPOF) ratkaisuihin.

On mahdollista pohtia, voitaisiinko näitä kahden tarkastelukohteen vertailua jatkaa edelleen ja niin tavoittaa uusia näkökulmia tiedonsiirto-ongelmien ratkaisuihin. Analogia ei tässä työssä ja tarkastelluilla ominaisuuksilla ole antanut tarkkaa tietoa ongelmanratkaisuun, ainoastaan ideoita ja suuntaviivoja.

Työssä ongelmallista on ollut se, että Intelin Management Enginestä ei ole ollut tarkkoja tietoja saatavissa, ainoastaan vahvistamattomia ja toisen käden tietoja tarvittavista ominaisuuksista. Lähdekritiikkiin on siis jossain määrin aihetta. Toisaalta tällä hetkellä ei ole mitään täsmällisempiä keinoja tiedonsaantiin; Intelillä lienee tähän sekä kaupallisia että poliittisia syitä.

7 YHTEENVETO

Konetekniikassa yksinkertainen perämoottorin murtosokka ratkaisee yhden voimansiirron ongelman, joka laajetessaan voisi aiheuttaa suuria ja kalliita vikaantumisia. Kun ongelma pysäytetään mahdollisimman aikaisessa vaiheessa, tällöin myös kustannukset saadaan pysymään pieninä. Se millainen yllättäviin ja eskaloituviin tiedonsiirto-ongelmiin (SPOF) käytettävä murtosokkaa vastaava "ohjelmallinen murtosokka" voisi olla, on pyritty tässä työssä hahmottelemaan analogian keinoin.

Tarkastellut, jossain määrin yhteismitalliset ominaisuudet näille kahdelle ongelmanratkaisutavalle ovat: yksinkertaisuus, riippumattomuus ja vaivaton huollettavuus. Lisäksi tässä työssä on tarkasteltu vähemmän yhteistä ominaisuutta eli monistettavuutta. Voimansiirron ja tiedonsiirron yhteisten ominaisuuksien avulla on siis pyritty tuomaan esille uusia näkökulmia tiedonsiirrossa koko ajan kasvavan ongelman, SPOF-ilmiön ratkaisuun. Analogiapäättelyn avulla ei varsinaisesti ole löydetty uusia ongelmanratkaisumenetelmiä, ainoastaan uusia näkökulmia niihin. Tiedonsiirron onnistumisen yksinkertainen vaatimus voidaan pukea lauseeseen: lähettävän ja vastaanottavan pään täytyy olla yksimielisiä tiedonsiirrosta ja molempien täytyy tietää toisen tietävän sen. Jos käytetään "ohjelmallista murtosokkaa", tietojärjestelmät selviävät omista tiedonsiirto-ongelmistaan itsenäisesti ja nopeasti ilman käyttäjän työpanosta tai valvontaa.

Tutkimusta tarvitaan lisää, koska "ohjelmalliset murtosokat" ja puhtaasti asynkroninen tiedonsiirto eivät ole yleistyneet eduistaan huolimatta ainakin osaksi nykyisten järjestelmien ylläpitäjien työvoimapoliittisista syistä.

"Ohjelmallinen murtosokka" ei pysty yksin ratkaisemaan kaikkia tiedonsiirron ongelmia, vaan tarvitaan viestintäprotokollamuutoksia, jotka toimivat yhdessä "ohjelmallisen murtosokan" kanssa.

LÄHDELUETTELO

BSD License, 2005. BSD License Definition [verkkodokumentti]. The Linux Information Project. Saatavissa: http://www.linfo.org/bsdlicense.html [viitattu 21.1.2018]

Cimpanu, C., 2016. Intel x86 CPUs Come with a Secret Backboor That Nobody Can Touch or Disable [verkkodokumentti]. Softpedia News. Saatavissa: http://news.softpedia.com/news/intel-x86-cpus-come-with-a-secret-backdoor-that-nobody-can-touch-or-disable-505347.shtml [viitattu 5.1.2018]

Coulouris, G., Dollimore, J. & Kindberg, T., 2005. Distributed Systems – Concepts and Design. 4. painos. United States of America: Addison-Wesley, 927 s. ISBN 0-321-26354-5

CSMA/CA, 2013. CSMA/CA [verkkodokumentti]. Wikipedia. Saatavissa: https://fi.wikipedia.org/wiki/CSMA/CA [viitattu 6.1.2018]

CSMA/CD, 2013. CSMA/CD [verkkodokumentti]. Wikipedia. Saatavissa: https://fi.wikipedia.org/wiki/CSMA/CD [viitattu 6.1.2018]

Gillespie M., 2012. Remote Configuration for Intel® AMT [verkkodokumentti]. USA: Intel. Saatavissa: https://software.intel.com/en-us/articles/remote-configuration-for-intel-amt [viitattu 5.1.2018]

Haaparanta, L. & Niiniluoto, I., 2016. Johdatus tieteelliseen ajatteluun. Helsinki: Gaudeamus Oy, 175 s. ISBN 978-952-495-397-9

Herder J. N., Bos H., Gras B., Homburg P. & Tanenbaum A. S., 2006. MINIX 3: A Highly Reliable, Self-Repairing Operating System. ACM SIGOPS Operating Systems Review. Volume 40, Issue 3, July 2006. s. 80 - 89.

Hunt, A. & Thomas, D., 2011. The Pragmatic Programmer – From Journeyman to Master. United States of America: Addison-Wesley, 321 s. ISBN 0-201-61622-X

Intel a, 2017. What is Intel® Management Engine? [verkkodokumentti]. USA: Intel. Saatavissa: https://www.intel.com/content/www/us/en/support/articles/000008927/software/chipset-software.html [viitattu 5.1.2018]

Intel b, 2017. Intel® Management Engine Critical Firmware Update (Intel-SA-00086) [verkkodokumentti]. USA: Intel. Saatavissa: https://www.intel.com/content/www/us/en/support/articles/000025619/software.html [viitattu 5.1.2018]

Intel c, 2017. Intel-SA-00086 Detection Tool [verkkodokumentti]. USA: Intel. Saatavissa: https://downloadcenter.intel.com/download/27150?v=t [viitattu 5.1.2018]

IoT, 2017. Esineiden internet [verkkodokumentti]. Wikipedia. Saatavissa: https://fi.wikipedia.org/wiki/Esineiden_internet [viitattu 6.1.2018]

Kauppinen M. & Pietarinen J., 1989. Tieteellinen päättely ja tieteentutkimus. Turku: Turun yliopisto 129 s. ISBN 951-880-233-5

Knuth, D. E., 2000. The Art of Computer Programming – Volume 1 / Fundamental Algorithms. 7. painos. United States of America: Addison-Wesley, 650 s. ISBN 0-201-89683-4

Leinonen, T. E., 1972. Kone-elimet. Teoksessa: Leskinen, J. (toim.) Tekniikan käsikirja 7 – Koneensunnitteluoppi. 8. painos. Jyväskylä: Gummerus Oy, s. 313-527. ISBN 951-20-0057-1

MTBF, 2016. MTBF [verkkodokumentti]. Wikipedia. Saatavissa: https://fi.wikipedia.org/ wiki/MTBF [viitattu 6.1.2018]

Niemi, A. H., 2018. Teknillinen mekaniikka tukee vientiä. Kaleva, 20.1.2018, s.57

Niiniluoto, I., 1983. Tieteellinen päättely ja selittäminen. Helsinki: Otava, 416 s. ISBN 951-1-73379-6

Niiniluoto, I., 1984. Johdatus tieteenfilosofiaan - Käsitteen- ja teorianmuodostus. Helsinki: Otava, 314 s. 2. painos. ISBN 951-1-05435-X

Nurmi, T., Rekiaro, I., Rekiaro, P. & Sorjanen, T., 2004. Gummeruksen suuri sivistyssanakirja. 6. painos. Jyväskylä: Gummerus Kirjapaino Oy, 629 s. ISBN 951-20-5863-4

Perrow, C., 1999. Normal Accidents – Living with High-Risk Technologies. Princeton : Princeton University Press, 451 s. ISBN 0-691-00412-9

Portnoy, E. & Eckersley, P., 2017. Intel's Management Engine is a security hazard, and users need a way to disable it [verkkodokumentti]. USA: Electronic Frontier Foundation. Saatavissa: https://www.eff.org/deeplinks/2017/05/intels-management-engine-security-hazard-and-users-need-way-disable-it [viitattu 5.1.2018]

Pressman, R. S., 2005. Software Engineering – A Practitioner's Approach. 6. painos. New York: McGraw-Hill, 912 s. ISBN 007-123840-9

Prowell S. J., Trammel C. J., Linger R. C. & Poore J. H., 1999. Cleanroom Software Engineering - Technology and Process. United States of America: Addison-Wesley, 390 s. ISBN 0-201-85480-5

Puolimoottorit, 2017. Oy Promotor AB varaosaluettelo - Murray -varaosat [verkkodokumentti] Saatavissa: http://www.promotor.fi/tuotteet/kaikki.pdf, 47 s. [viitattu 22.1.2018]

Rider, 2017. Rider 11/13 Rider 11 Bio/13 Bio - Käyttöohje [verkkodokumentti] Saatavissa: http://www.husqvarna.com/ddoc/HUSO/HUSO2000_FIfi/HUSO2000_FIfi__1019158-11.pdf, 47 s. [viitattu 22.1.2018]

Salokangas, J., 1973. Metallien aineenkoetus. Teoksessa: Leskinen, J. (toim.) Tekniikan käsikirja 8 – Koneensunnitteluoppi. 8. painos. Jyväskylä: Gummerus Oy, s. 213-262. ISBN 951-20-0193-4

Shear Pin, 2018. Just Answer Boat - Have Boat Questions? Ask a Boat Repair Expert. [verkkodokumentti] Saatavissa: https://www.justanswer.com/boat/5egml-hi-2001-mercury-3-3hp-outbaord-spins-propeller.html [viitattu 21.1.2018]

Siberschatz A. & Galvin P. B., 1998. Operating System Concepts. 5. painos. USA : Addison-Wesley, 888 s. ISBN 0-201-54262-5

Stavely, A. M., 1999. Toward Zero-Defect Programming. United States of America: Addison-Wesley, 240 s. ISBN 0-201-38595-3

Tanenbaum, A. S., 2017. An Open Letter to Intel [verkkodokumentti]. Vrije Universiteit Amsterdam, Faculty of Science, Computer Sciences. Saatavissa http://www.cs.vu.nl/~ast/intel/ [viitattu 20.1.2018]

Technet, 2017. Avoiding Single Points of Failure [verkkodokumentti]. USA: Technet Microsoft Saatavissa: https://technet.microsoft.com/en-us/library/cc938489.aspx [viitattu 22.1.2018]

Wikipedia b, 2017. Single point of Failure [verkkodokumentti]. Wikipedia. Saatavissa https://en.wikipedia.org/wiki/Single_point_of_failure [viitattu 6.1.2018]

Zammit, D., 2016. Intel x86s hide another CPU that can take over your machine (you can't audit it) [verkkodokumentti]. Boingboing.net Saatavissa: https://boingboing.net/2016/06/15/intel-x86-processors-ship-with.html [viitattu 5.1.2018]