jamk.fi

Tukiaseman poistoprosessin automatisointi radioverkossa

Matkapuhelinverkot, DNA Oyj

Viljami Kilkkilä

Opinnäytetyö
Marraskuu 2020
Tekniikan ala
Insinööri (AMK), Tieto- ja Viestintätekniikan tutkinto-ohjelma

Jyväskylän ammattikorkeakoulu
JAMK University of Applied Sciences

jamk.fi

Kuvailulehti

Tekijä(t)	Julkaisun laji	Päivämäärä
Kilkkilä, Viljami	Opinnäytetyö, AMK	Marraskuu 2020
	Sivumäärä 43	Julkaisun kieli Suomi
		Verkkojulkaisulupa myönnetty: x

Työn nimi
Tukiaseman poistoprosessin automatisointi radioverkossa
Matkapuhelinverkot DNA Oyj

Tutkinto-ohjelma
Insinööri (AMK), Tieto- ja Viestintätekniikan tutkinto-ohjelma

Työn ohjaaja(t)
Mika Rantonen, Jani Immonen

Toimeksiantaja(t)
Jyrki Ollonqvist, DNA Oyj

Tiivistelmä

5G-teknologian yleistyminen on tuonut mahdollisuuksia uudistaa operaattoreiden radioverkkoteknologiaa. DNA:lla tämä nähtiin tukiasemateknologian päivittämisenä, mikä toi tullessaan paljon tehtävää uuden integroimiseen ja vanhan laitteen poistamiseen. Toimeksiantajan halu oli tuottaa DNA:lle tukiaseman poistoa helpottava työkalu, jonka tarkoituksena oli säästää aikaa toistuvilta yksinkertaisilta prosesseilta. Tukiasemien määrän ollessa lukuisissa tuhansissa, tiedettiin muutoksen vievän aikaa useiden vuosien ajan. Näin nähtiin mahdollisimman hyvin tehdyn prosessin automatisoinnin vaikuttavan muutosta hallitsevan ryhmän säästettyyn aikaan.

Soveltava tutkimus toteutettiin kehittämällä työkalua askel askeleelta ja testaamalla sen toimintaa. Näin saatiin korjattua virheitä sitä mukaa, kun niitä syntyi ja kehitys oli suhteellisen nopeaa. Tietoa kerättiin suurilta osin verkosta, mutta myös työpaikan kollegat tarjosivat hyviä menetelmiä prosessien hoitamiselle.

Tutkimuksen tuloksena saatiin radioverkkoa operoivan tiimin käyttöön työkalu, mikä vähensi tukiaseman vaihtoprosessiin käytettyä aikaa neljännekseen siitä, mitä se oli ennen työkalun olemassaoloa. Yhtenä työpäivänä säästetty aika voi parhaimmillaan ylittää itse työpäivän pituuden työkalun ansiosta.

Radioverkosta elementtejä ja konfiguraatioita poistava työkalun kehitys vaatii toiminnalta paljon huolellisuutta ja suunnittelua. Tehtyjen toimintojen tallentaminen lokiin on ehdotonta, jos verkosta havaitaan puutteita prosessin jälkeen. Hyvin suunniteltuna työkalun toiminta kaikin puolin helpottaa sitä käyttävän operoijan työpäivää, kun prosessin toimintaan on täysi luotto.

Avainsanat (asiasanat)
Radioverkko, Automatisointi, 5G

Muut tiedot (salassa pidettävät liitteet)

jamk.fi

Description

Author(s)	Type of publication	Date
Kilkkilä, Viljami	Bachelor's thesis	November 2020
		Language of publication:
	Number of pages 43	Permission for web publication: x

Title of publication
Hardware Manufacturers transfer process
Radio Access Networks, DNA Oyj

Degree programme
Engineer, Information and Communication Technology Degree

Supervisor(s)
Mika Rantonen, Jani Immonen

Assigned by
Jyrki Ollonqvist, DNA Oyj

Abstract

The generalization of 5G technology has brought opportunities to modernize operators radio network technology. With DNA Oyj, this opportunity was taken to action with update of base station equipment, which brought a lot of work, like to integrate the new and remove the previous technology. DNA Oyj had a desire for a tool to facilitate base station removal, which purpose was to save time from repetitive simple processes. With base stations quantity being significant in multiple thousands, it was known to take multiple years to complete this project. Thus, the automation of the swap process was seen as mandatory thing to implement and ease the workload of the team operating radio network.

Applied research is carried out by developing the tool step by step and testing its operation. In this way, errors were corrected as they came, and development proceeded relatively rapid. Information was collected from a large part of the network, but also workplace colleagues provided good methods for managing the processes.

As a result of the study, a tool was made available to the team operating the radio network, which reduced the time spent on the base station swap process by a quarter of what it was before the tool existed. The time saved in one working day can, at best, exceed the length of the working day itself using the tool.

The development of a tool that removes elements and configurations from a radio network requires a lot of care and planning. It is essential to record the activities performed to the log if deficiencies are detected in the network after the process. When well designed, the operation of the tool in all respects facilitates the working day of the operator using it, when the operation of the process is fully credited.

Keywords/tags (subjects)
Radio Network, Automation, 5G

Miscellaneous (Confidential information)

Sisältö

Kuviot

Lyhenteet ja termit

Abis	Rajapinta BSC:ltä BTS:lle radioliikennettä varten
ANR	Automatic Neighbor Relation, 4G-sukupolvessa kehittynyt automaatti sesti luotu naapuruusrelaatio toisen eNodeB:n kanssa
BSC	Base Station Controller,
BTS	2G-Tukiasema
eNodeB	4G-Tukiasema
GSM	Global System for Mobile Communicatios, viitataan usein 2G:hen
Handover	Mobiililaitteen siirtyessä toisen solun läheisyyteen hallitseva solu luovuttaa/suorittaa handoverin vahvemman signaalin tarjoavalle solulle
IuB	Rajapinta RNC:ltä NodeB:lle radioliikennettä varten
LAC	Location Area Code, määrittää kohteen sijainnin 16-bittisenä numerona 2G:ssä ja 3G:ssä
LTE	Long Term Evolution, viitataan usein 4G:hen
MGW	Media Gateway muuntaa median eri teknologioiden välillä, esimerkiksi 2G/3G
MSC	Mobile Switching Center, hallitsee puheluiden yhdistämistä ja toimintaa radioverkossa
NodeB	3G-Tukiasema

RNC	Radio Networki Controller, 3G-liikenteen ohjain
SCP	Secure Copy Protocol, datan siirtäminen tietoverkossa turvallisesti
SGSN	Serving GPRS Support Node, palvelee päätelaitetta julkiseen verkkoon pääsyssä 2G ja 3G tekniikoissa
SS7	Signaling System 7, sarja matkapuhelin protokollia hallitsemaan puheluita
TAC	Type Allocation Code, määrittää kohteen sijainnin 16-bittisenä nume rona 4G:ssä
UMTS	Universal Mobile Telecommunications System, käytetään viittauksena usein 3G:hen
VoLTE	Voice over LTE, mahdollistaa puhelun LTE-verkossa
VoWifi	Voice over Wifi, mahdollistaa puhelun Wifi-verkossa

1 Työn lähtökohdat

1.1 Taustaa työstä

Kohteena työssä oli DNA:n matkapuhelinverkoissa tapahtuva tukiasemien uusimisprosessi. Osasyynä laajempaa radioverkkoteknologian uusimista oli uuden 5G-teknologian käyttöönotto DNA:lla, mikä nähtiin tilaisuutena resurssien säästämisessä. Uuden sukupolven radioverkkoteknologian käyttöönotto tarkoittaa useamman tuhannen tukiaseman vaihtumista uusiin erilaisiin verkkolaitteisiin. Uuden tullessa vanhan tilalle, halutaan vanhasta verkkolaitteesta ja sen tiedoista päästä eroon, jotta säästytään konflikteilta verkkoliikenteessä. Laitevaihtojen määrän ollessa tuhansissa, halusi toimeksiantaja ratkaisuja joilla kyseinen prosessi säästäisi kalliita työtunteja. Kun nähdään tällainen kuva tilanteesta, missä kyseisiä prosesseja tulee jatkumaan useampia vuosia, on hyvä lähteä kehittämään nopeampia menetelmiä ja työkaluja. Näillä asiantuntijat säästävät aikaa yksinkertaisten operaatioiden suorittamiselta ja voivat keskittyä analyyttisempiin ja vaativampiin työtehtäviin.

1.2 Työn tavoitteet

Työssä lähtökohtana oli tuottaa työkalu nopeuttaakseen prosessia, jossa verkosta vaihtuu tukiasema uudempaan teknologiaan. Vanha verkkolaite verkossa tuottaisi ristiriitoja uuden korvaavan kanssa, sekä aiheuttaisi myös hämmennystä itse radioverkkoa ylläpitävien operoijien parissa. Työkalu kehitetään jatkoa varten omaksi moduuliksi, joka on osana koko tukiasemateknologian uusimista. Tätä varten työkalun koodin kommentointi ja rakenne piti tehdä järkeväksi, jotta sitä voidaan hyödyntää eri ympäristöissä helposti. Eri sukupolven matkapuhelinverkko-teknologioihin liittyy erilaisia piirteitä. Niiden rakenne on erilainen ja käyttöliittymä millä niitä ohjataan, voi myös vaihdella. Esimerkkinä 4G/5G-teknologiat eivät pohjaudu lähes laisinkaan ohjaimiin (RNC/BSC/MSC), vaan suurin osa poistoprosessissa näiden osalla kohdistuu itse tukiasemiin. 2G/3G:ssä taas kaikki kohdistuu ohjaimiin, näiden ollessa vanhempaan keskitettyyn teknologiaan

pohjautuvaa tekniikkaa. Jälkimmäisenä mainituissa sukupolvissa prosessin osalta onkin enemmän työnsarkaa ja tavoitteena suurin painoarvo asettuu näiden tekniikoiden hoitamiseen pois radioverkosta.

Työkalun toimiakseen, tarvitsisi siihen syöttää vain tukiaseman tunniste ja tämän jälkeen tarvittavien varmistusten kautta se lähtee tyhjentämään kyseiseen laitteeseen liittyviä parametreja ja tietoja verkosta. Lähtökohtana työkalu tehtäisiin Pythonilla, koska laitevalmistajalla on sillä ohjelmointikielellä omaa kirjastoa verkon hallitsemiseen ja konfiguroimiseen. Työkalu rakennettiin laitevalmistajan palvelinalustan päälle ja tekstipohjaiseksi eli ilman omaa graafista käyttöliittymää. Skriptin kulkua voi seurata reaaliajassa kun työkalu käy läpi osaprosesseja ja tulostaa tehdyt toimenpiteet käyttäjän terminaaliin.

Lopputuloksena pitäisi olla työkalu vaihtoprosessia varten, jota käyttämällä käyttäjällä kuluisi vähemmän aikaa yksinkertaisiin komentojen ja toimintojen syöttämiseen. Sen tulisi joko suoraan tuhota vanhat tiedot verkosta tai avittaa tekijää siinä määrin, ettei tietoja itse tarvitse hakea ja syöttää. Käyttäjällä on varmuus siitä, että työkalu tekee tulosta ja sitä ei tarvitse vahtia, vaan antaa lopuksi palautteen prosessin tärkeiksi havaittavista kohdista. Tarvittavat jatkotoimenpiteet ja virheet prosessissa tallentuisi tiedostoon, mistä ajoitettu skripti puhdistaisi niitä määritetyin väliajoin. Työkalun koodi on hyvin kommentoituna jolloin sen hyödyntäminen tulevissa toimeksiantajan prosesseissa olisi helppoa muidenkin kuin itse kehittäjän osalta.

1.3 Toimeksiantaja

Toimeksiantajana työlle toimii DNA Oyj, joka on Suomen suurimpia matkapuhelinverkon ylläpitäjiä jo vuodesta 2001. DNA:n 4G verkko kattaa 99% suomalaisista ja valokuitupohjaiseen kaapeliverkkoon kuuluu noin 620 000 kotitaloutta. (DNA yrityksenä, n.d.) Opinnäytetyö on tehty DNA:n tekniikan yksikön radioverkko-osastossa. Osaston tehtävänä on kehittää, valvoa, suunnitella ja rakennuttaa DNA:n matkapuhelinverkon toimintaa Suomessa.

2 Tutkimusasetelma

2.1 Tutkimuskysymykset

Opinnäytetyön tutkimusongelmaksi voidaan kuvailla toimeksiantajalla tapahtuva tukiasematekniikan uudistumista, mihin ei ole kehitetty vielä mitään ratkaisua mikä nopeuttaisi vanhan verkkolaitteen poistamista radioverkosta. Toimeksiantajalla oli tarve kehittää tätä prosessia. Sen ensimmäisiä ja olennaisimpia vaiheita oli saada työkalu, joka hoitaisi vaihtuvan tukiaseman poistotyön asiantuntijan puolesta.

Opinnäytetyön tutkimusongelmasta saatiin kehitettyä seuraavat opinnäytetyön tutkimuskysymykset:

1. Miten tuottaa tukiaseman poistotyökalu, jonka käyttäminen on selkeää ja yksinkertaista operointitiimin jäsenille?
2. Kuinka tehdä kyseisen työkalun koodista selkeä ja helposti muokattava jatkokehityksessä? Tämä myös helpottaisi muiden työntekijöiden mahdollisuutta skriptin korjaamiseen ja muokkaamiseen.

2.2 Tutkimusmenetelmä

Tutkimusmenetelmät jaetaan pääsääntöisesti kahteen erilaiseen menetelmään, kvantitatiiviseen ja kvalitatiiviseen. Edeltävällä haetaan pääsääntöisesti laajemmalla tutkimustiedolla tuloksia ja jälkimmäisellä enemmän laadullisia tuloksia. Kvantitatiivisissa tutkimuksissa ollaan kiinnostuneita luokittelusta, syy- ja seuraussuhteista, vertailusta ja numeerisiin tuloksiin pohjautuvasta ilmiön tulkitsemisesta. Kvantitatiiviseen tutkimukseen sisältyy laajalti erilaisia laskennallisia ja tilastollisia analyysimetodeja. (Määrällinen tutkimus, 2015.) Kvalitatiivisessa tutkimuksessa pohjaudutaan erilaisiin näkökulmiin toteutuksessa. Yhteisiä piirteitä nousee esille kohteen esiintymisympäristöön ja taustaan, kohteen tarkoitukseen ja merkitykseen, ilmaisuun ja kieleen kuuluvat näkökulmat (Laadullinen tutkimus, 2015).

Työssä lähdetään tutkimaan miten prosessia, jossa laitevalmistaja vaihtuu, voitaisiin nopeuttaa sisäisten työkalujen ja skriptien kautta. Näin ollen ei valita määrällistä tai

laadullista tutkimusmenetelmää, vaan opinnäytetyössä käytetään soveltavaa tutkimusta, jolloin ei pohjauduta vaan kahden pääsääntöisen menetelmän lopputulokseen eli toteumaan. Soveltavalla tutkimuksella tarkoitetaan sellaista tapaa tiedon keräämiseksi missä haetaan lopputulokseksi jotain käytännön sovellutusta. Tässä opinnäytetyössä otettiin käytännön ongelma tutkintaan ja haettiin sille ratkaisuja. (Soveltavasta tutkimuksesta, n.d.)

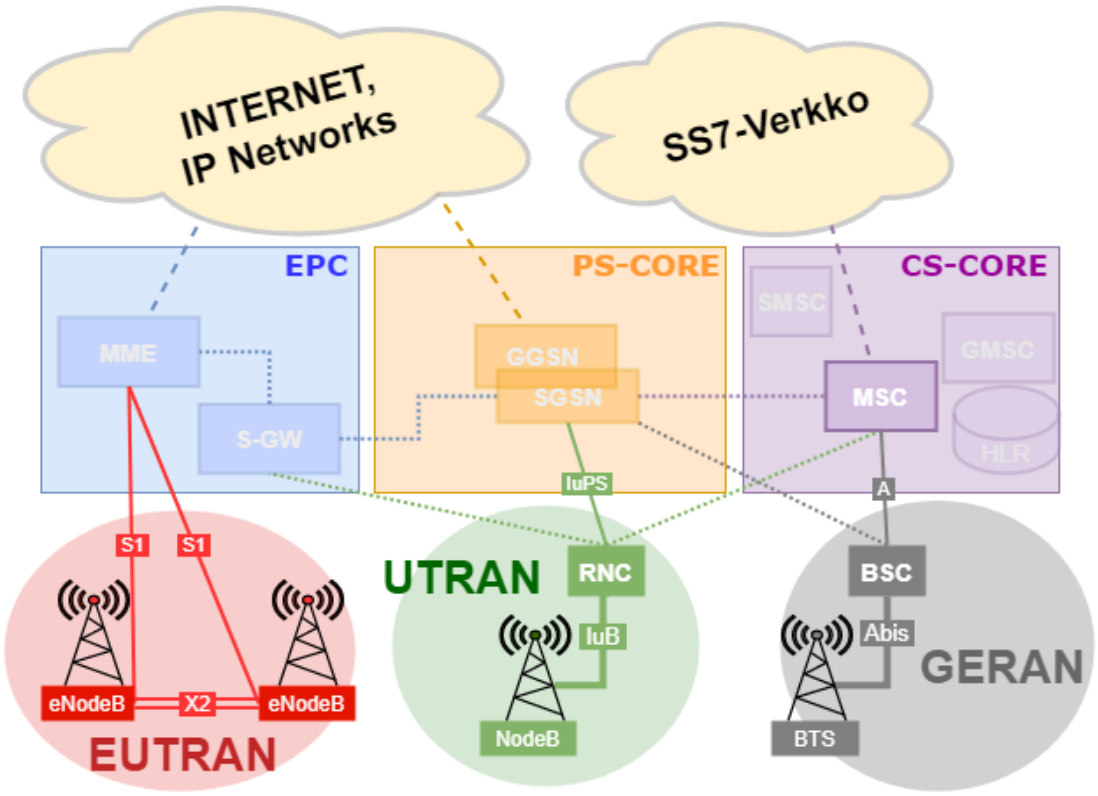
2.3 Soveltavan tutkimuksen toimeenpano

Soveltava tutkimus tässä työssä jaetaan neljään vaiheeseen. Vaiheessa 1. luodaan prototyyppejä, toisessa vaiheessa lasketaan ja valitaan paras, kolmannessa käytetään ratkaisua testattavana ja neljäntenä lopullinen tuotos. Tutkimusta lähettiin tekemään luomalla prototyyppejä, joilla haettiin ratkaisua eri kohtiin poistoprosessia. Testattiin prototyypit ja vertailtiin lähinnä aikaa, mikä niillä meni tietyn prosessin hoitamiseen. Näistä valittiin aina parhaimmat integroitavaksi työkaluun. Seuraavaksi oli tärkeää testauttaa työkalua usealla tekijällä. Erilaiset käyttöoikeus-, järjestelmä- ja yhteensopivuusongelmat eivät ilmene yhden henkilön testauksilla. Lisäksi käyttäjän väärinymmärrykset toiminnallisuudesta nousevat ylös ja myös kehitysideoita voi tulla lisää. Palautetta käyttökokemuksista ja lisääntyneistä ideoista prosessiin siirsi ratkaisun kyseisen toiminnallisuuden osalta valmiiksi tai takaisin ykkösvaiheeseen. Jokainen kehitysaskel opinnäytetyössä seurasi tätä toimenpidettä ja näin saatiin toimiva ja mahdollisimman aukoton työkalu tehokkaaseen käyttöön.

3 Radioverkkoteknologia

3.1 Radioverkon infrastruktuuri

Radioverkkojen toiminnan voi pääpiirteittäin jakaa kahteen alueeseen, tukiasemat ja ohjaimet. Vanhemmissa teknologioissa toiminnot ja resurssien jaot painottuvat tukiaseman alueen ohjaimelle. 4G:n ja 5G:n myötä liikenteen kasvaessa enemmän datan kuin puheen suuntaan, sekä laitteiden tehojen nousun tuloksena on kaikki toiminnallisuus lähes kokonaan tukiaseman päässä. Tässä työssä prosessin poistoalueisiin kuuluu BSC, MSC, RNC, MMTEL, eNodeB, NodeB ja BTS (ks. kuvio 1).



Kuvio 1 Matkapuhelinverkkojen topologia, muokattu. 6.9.2020. https://braindrain.expert/index.php/2019/03/06/summary-of-common-gsm-3g-lte-ims-protocol-interfaces-and-the-underlying-protocols/

3.2 Ohjaimet ja palvelimet

On kolmenlaisia palvelimia/ohjaimia matkapuhelinverkkoteknologiassa mitä käsitellään tässä työssä. Mobile Switching Center (MSC), Radio Network Controller (RNC), Base Station Controller (BSC) ja Mobile Telephony Application Server (MTAS). MSC:n tehtävä on reitittää puhelut oikeisiin osoitteisiin ja BSC/RNC:n tehtävä on jakaa ja määrittää resursseja niiden hallinnassa oleville tukiasemille. MTAS:n tehtävä on tuottaa puhelintoimintaa ja multimedia toimintoja mobiililaitteille (What is a mobile telephony application server (mobile TAS) / MMTel-AS? n.d.)

3.2.1 BSC (Base Station Controller)

BSC:n (ks. kuvio 2) tehtävä matkapuhelinverkoissa on hoitaa yhden tai useamman BTS:n (2G-tukiaseman) kontrollointia. Se toimii fyysisenä linkkinä tukiaseman ja MSC:n välillä. BSC varastoi verkon parametrejä ja dataa, joita muun muassa kantoaallon taajuus, taajuushyppely lista ja tehotasot. Hoitamalla tukiasemaan liittyvät toimenpiteet ja ohjaamiset, jättää BSC MSC:lle vain sille tärkeät tehtävät eli puheliikenteen ja tietokannan hallinnan. (Base Station Controller. n.d.)

Kuvio 2 Base station controller kabinetti. 6.9.2020. https://www.yunpantel-ecom.com/zxg10-ibsc-base-station-controller-cabinet-ip-system

3.2.2 MSC-S (Mobile Switching Center Server)

MSC (ks. kuvio 3) toimii ydinosana matkapuhelinverkkoa sen yhdistäessä tilaajien puheluita 2G ja 3G verkoissa, sekä hoitaa esimerkiksi myös SMS-viestit ja faksin. Kaikki mobiililiikenne GSM-verkosta reitittyy MSC:n läpi. Tapauksissa missä tilaajan puhelin on BSC:n solujen rajalla, MSC suorittaa luovutuksen solulle, jonka se hakee käymällä läpi listan läheisistä BSC:stä. (Mobile Switching Centre. n.d.)

Kuvio 3 Mobile switching center server. 6.9.2020. https://www.ericsson.com/en/portfolio/digital-services/cloud-communication/mobile-switching/mobile-switching-center-server

3.2.3 RNC (Radio Network Controller)

RNC (ks. kuvio 4) on hallitseva elementti 3G:n liityntäverkossa ja on vastuussa NodeB:den kontrolloinnista, jotka kuuluvat sen alaisuuteen. Se hoitaa alueen radioverkon resurssien hallinnan, osittain mobiilisuutta ja salaa datan ennen kuin se lähetetään tai vastaanotetaan puhelimesta. RNC yhdistyy piirikytkentäisessä runkoverkossa MGW:n läpi SGSN:n pakettikytkentäiseen runkoverkkoon. (Radio Network Controller, n.d.)

Kuvio 4 Radio network controller. 6.9.2020. https://vdocuments.mx/nokia-radio-network-controller-radio-network-controller-contact-us-tommiriihimakilemcon-asiacom.html

3.2.4 MTAS

Mobile TAS:n pääfunktioita matkapuhelinverkon toiminnassa on tuottaa hallinta äänelle ja videolle puheluissa, sisältäen tuen GSMA standardeille IR.92 ja IR94 VoLTE:lle ja IR.51 VoWiFi:lle. GSM:n täydentäviä palveluita kuten puhelun edelleen lähetystä, ja neuvottelua sekä pitämään yllä yhdenvertaisuutta TDM:n ja IP verkon infrastruktuurien kanssa. Piirikytkentäisten ja pakettivälitteisten välisen yhteyden pitäminen verkkojen välillä. Tarvittavat rakenteet uusien palveluiden ja sovellusten käyttöönottoon ja avoimeen kehitysalustaan, kannustaen kolmannen osapuolen kehityksiä. (What is a mobile telephony application server (mobile TAS) / MMTel-AS, n.d.) Matkapuhelinverkoissa jokainen hallitseva solu lisätään MTAS-palvelimelle solun yksilöivän tunnuksen, maakuntakoodin ja LAC:n mukaan.

3.3 Tukiasemakalustus

3.3.1 Keskusyksikkö

Tukiasemien keskusyksikkö eli BaseBand toimii kyseisen kohteen radioverkon hallintayksikkönä. BaseBandiin konfiguroidaan haluttavat parametrit ja toimii täten älyllisenä osana tukiasematoteutusta. Se tarvitsee radioyksiköitä ja antenneja vastaanottamaan ja lähettämään dataliikennettä langattomille päätelaitteille ja BaseBandista käyttäjän data siirtyy käytetyn sukupolven tekniikan runkoverkkoon. Tukiasemien

keskusyksiköt muodostuvat usein joko modulaarisesta piiristä (ks. kuvio 5), mihin voi lisätä kapasiteettia eri tekniikoista tai ei-modulaarisia (ks. kuvio 6), mihin ei voi jälkikäteen lisätä eri tekniikoitten kortteja. Esimerkkinä edellä mainitussa yksikössä käyttöönottopäivänä olisi piireinä otettu käyttöön vain 2G,3G ja 4G, mutta mahdollisuus 5G:lle samaan yksikköön onnistuisi myöhemmin. Jälkimmäisessä olisi ollut joko yksikkö, missä 5G on myös olemassa, mutta sitä ei oteta käyttöön tai muuten 5G:tä haluttaessa, olisi hankittava uusi yksikkö.

Kuvio 5 Huawein modulaarinen BaseBand https://www.alibaba.com/product-detail/Huawei-bbu3900-bbu3910-bbu-3900_60838249747.html

Kuvio 6 Ericssonin ei-modulaarinen BaseBand https://www.tempesttelecom.com/products/ericsson-baseband-6630-kdu137848-11/

3.3.2 Radiot

Tukiaseman radioyksiköt eli RRU:t (Remote Radio Unit) mahdollistavat tukiaseman nopean dataliikenteen ja vähäisen latenssin päätelaitteelle. RRU (ks. kuvio 7) vahvistaa RF signaaleja ja on yhteydessä BaseBand-yksikköön. Radioita löytyy erilaisiin ratkaisuihin ja jokaiselle löytyy jokin käyttö. 5G:n myötä radiot ovat integroituina antennin kanssa, joka näyttää olevan tulevaisuuden korkean datanopeuksien valinta. Näitä

ovat esimerkiksi Massive MIMO (Multiple Input Multiple Output) toteutus, jossa antennien määrä on korkea ja jolloin päätelaitteille saadaan enemmän kohdistettua radioliikennettä. (Ericsson Radio System / Radio, n.d..) Toinen integroitu radioyksikkö on mmWave, jossa tähdätään korkeisiin taajuuksiin aina 24GHz:stä 39GHz:iin (Ericsson Radio System / mmWave, n.d).

Kuvio 7 Ericsson Remote radio head W-CDMA LTE. Viitattu 27.9.2020. https://www.pngegg.com/en/png-xobsx

3.3.3 Antennit

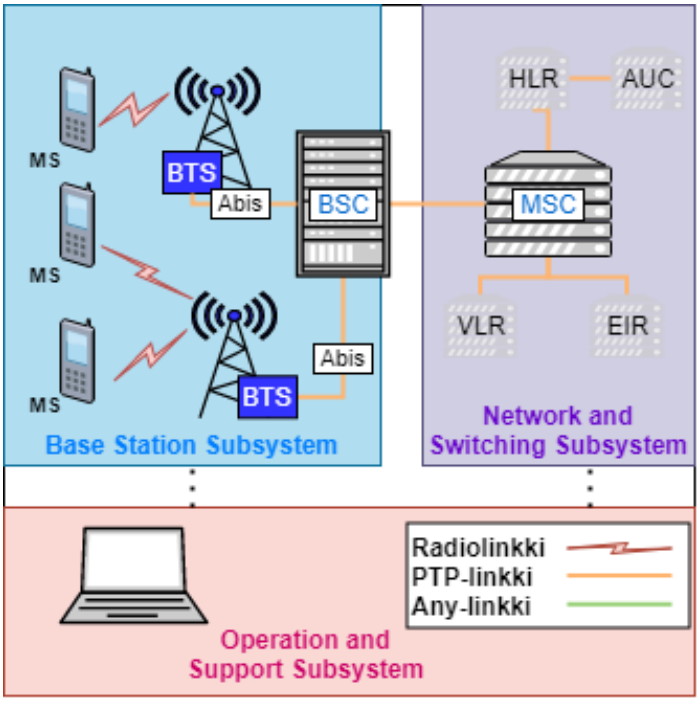
Antennien tarkoitus radioverkossa on lähettää ja vastaanottaa päätelaitteiden kanssa käytyä dataliikennettä. Antennit matkapuhelinverkoissa voidaan jakaa kolmeen kategoriaan. Passiiviset antennit, aktiiviset antennit ja antennin linjalaitteisiin. Aktiivisen ja passiivisen antennin ainoa ero käytännössä on, että aktiivisessa antennissa on vahvistin signaalin lähetystä varten. Antennin linjalaitteet pitävät huolen liikenteessä olevien signaalien suodattamisesta, radioiden liittämisestä antenneihin ja sisältää valikoiman täydellisten ja optimoitujen antennijärjestelmien luomiseen. (Ericsson Antenna System, n.d.) Uusimmat 5G-verkon antennit ovat integroitu radioihin ja ovat samalla aktiivisia antenneja. Samoissa toteutuksissa nähdään myös Massive MIMO -toteutus (ks. kuvio 8), jossa antennien määrä voi yhdessä yksikössä olla 64–128 antennia.

Kuvio 8 Massive MIMO for 5G explored by METIS. Viitattu 27.9.2020. https://www.ericsson.com/en/blog/2017/9/massive-mimo-for-5g-explored-by-metis

3.4 Matkapuhelinverkkojen sukupolvet

3.4.1 GSM

GSM eli Global System for Mobile Communication rakennettiin kattamaan 2G-yhteyksiä mobiilipuhelimiin. Se jaetaan neljään osaan, Base Station Subsystem (BSS), Operation and Support Subsystem (OSS), Network and Switching Subsystem (NSS) ja Mobile Station (MS) (ks. kuvio 9).

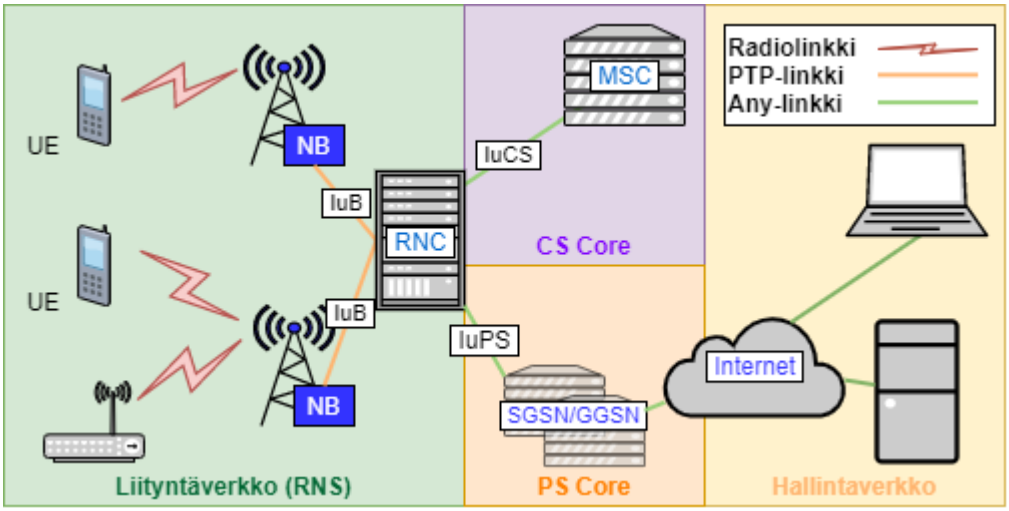


Kuvio 9 GSM-verkon toimintaperiaate. 6.9.2020. https://nickvsnetworking.com/gsm-with-osmocom-part-4-the-base-station-controller-bsc/

BSS pitää sisällään BTS:n ja BSC:n. Edeltävä on itse tukiasema maastossa joka radiolähettimineen ja antenneineen lähettää ja vastaanottaa mobiililaitteiden kanssa. BTS ja BSC kommunikoivat Abis-linkin kautta toisilleen. **MS** viittaa mobiililaitteeseen, joka sisältää SIM-kortin (Subscriber Identification Module), jolla tunnistetaan tilaaja. Mobile Station sisältää prosessorin, vastaanottimen ja näyttöyksikön. **BTS** on vastuussa yhteyden luomiseen Mobile Stationin ja verkon välille kattamalla soluillaan ympäröivää aluettaan kuvion 9 vasemmalla. **NSS,** jota usein kutsutaan myös core-verkoksi, on dataverkko pitäen sisällään erilaisia yksiköitä, jotka tuottavat ohjauksen ja liitynnän koko mobiiliverkossa. NSS:n pääelementtejä on MSC, HLR, VLR, EIR, AuC, GMSC ja SMS-G. **OSS** on komponentti NSS:ää ja BSC:tä. Sitä käytetään GSM-verkon kontrollointiin ja monitorointiin sekä liikennekuorman ohjaamiseen pois BSS:stä. 2G GSM verkko on looginen ja yksinkertainen tapa operoida verkkoa verrattuna uudempien sukupolvien ohjelmistopohjaiseen operointiin. (GSM Network Architecture, n.d.)

3.4.2 UMTS

UMTS eli Universal Mobile Telecommunications System on GSM:n seuraajana kolmas matkapuhelinteknologian sukupolvi. Kehitys tälle alkoi jo siinä vaiheessa, kun GSM otettiin käyttöön. Radioliityntäverkko muuttui paljon, kun uusi radioliityntä, joka perustui CDMA:han oli käytössä. Puheen lisäksi oli mahdollisuus tällä myös dataliikenteelle. UMTS-verkko sisältää kolme pääkomponenttia, User Equipment (UE), Radio Network Subsystem (RNS) ja Core Network (ks. kuvio 10).



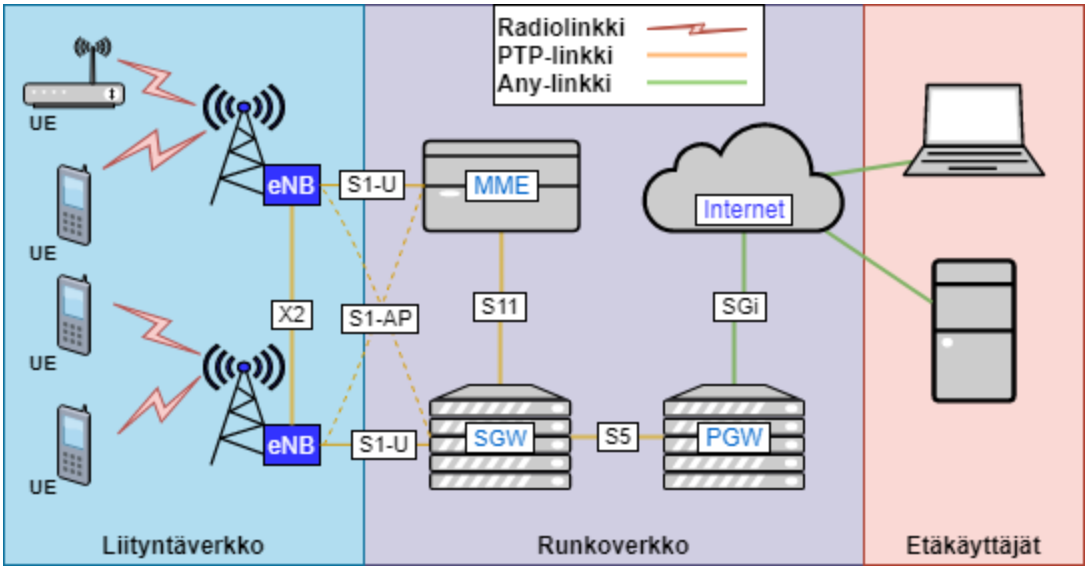
Kuvio 10 UMTS-verkon toimintaperiaate, muokattu. 6.9.2020. https://www.researchgate.net/figure/Simulation-topology_fig1_221289239

UE on merkittävä komponentti 3G UMTS-verkko arkkitehtuuria. Se muodostaa käyttäjän päädyssä lopullisen liitynnän. Kuviossa 10 vasemmassa reunassa. Osat UE:ssa koostuvat UE RF piiristä, kantataajuuden prosessoinnista, patterista ja USIM:stä. RF piiri käsittelee kaikki elementit signaalissa, sekä vastaanottimen että lähettimen. Kantataajuuden signaalin prosessointi sisältää lähinnä digitaalista piiriä. Molempien edellä mainittujen osien pääpiirteitä ovat virrankulutuksen minimointi ja käytön optimointi näiden käyttäessä edeltävään sukupolveen monimutkaisempia ratkaisuja. Patterit puhelimessa/laitteessa ovat myös kehittyneet kestävimmiksi. USIM (Universal Subscriber Identity Module), joka on vain kehittyneempi versio SIM-kortista pitää sisällään IMSI:n ja MSISDN:n (3G UMTS Network Architecture, n.d). RNS on osuus 3G UMTS-verkkoa, jonka liitynnässä on molemmat sekä UE että Core-verkko. Tähän kuuluu RNC:t ja NodeB:t (UMTS UTRA / UTRAN: radio network subsystem RNS, n.d).

3.4.3 LTE

LTE:ssä eli 4G-verkossa on toiminta kehittynyt enemmän sovelluspainotteiseksi ja ”äly” on siirtynyt runkoverkolta tukiasemalle kasvavin määrin, vaikka 4G:n runkoverkko on myöskin erittäin kehittynyt edeltäviin sukupolviin nähden. Kokonaisuudessaan voidaan 4G-verkko jakaa kahteen osaan, liityntä- ja runkoverkkoon. Liityntäverkossa löytyy elementteinä päätelaite ja eNodeB, mikä kuvastaa 4G-tukiasemaa. Kahden 4G-tukiaseman välillä on X2-rajapinta (ks. kuvio 12), mikä auttaa siirtämään tilaa-

jan liikennettä toiselle tukiasemalle, jakaa kuormaa verkon ollessa ruuhkainen ja toimii suorana yhteytenä tukiasemien välillä radioverkon kautta (MAPS™ LTE for X2 Interface Emulator. 28.9.2020). Liityntäverkko yhdistyy runkoverkkoon S1-rajapintojen kautta, joita on kahdenlaisia. Yksi MME:lle, joka on S1-AP ja toinen S-GW:lle, joka on S1-U (ks. kuvio 11). Runkoverkon MME (Mobility Management Entity) hallinnoi päätelaitteen mobiiliutta ja liityntäverkkoa, sekä perustaa kantajan polun päätelaitteelle. MME hallitsee myös seuraavia turvallisuusmenetelmiä, käyttäjän autentikointi, salaus ja eheyden suojaaminen. (Samaoui, El Bouabidi, Obaidat, Zarai & Mansouri 2015, 3-32.) SGW:n eli Serving Gatewayn päätoiminta perustuu reititykseen ja datan uudelleen lähetykseen. Se on myös vastuussa 4G-tukiasemien sisäisistä luovutuksista ja luo mobiilisuutta LTE:n ja muiden sukupolvien verkkoteknologioiden kanssa. PGW:n eli PDN Gatewayn toimintaan kuuluu päätelaitteen ja ulkoisen verkon linkin luonti. Se toimii dataliikenteen lähtöpisteenä päätelaitteelle. PGW:n perustoimintoihin kuuluu esimerkiksi käytännön täytäntöönpano, pakettien suodatus ja pakettiseulonta. (System Architecture Evolution (SAE) and the Evolved Packet Core (EPC), n.d.)



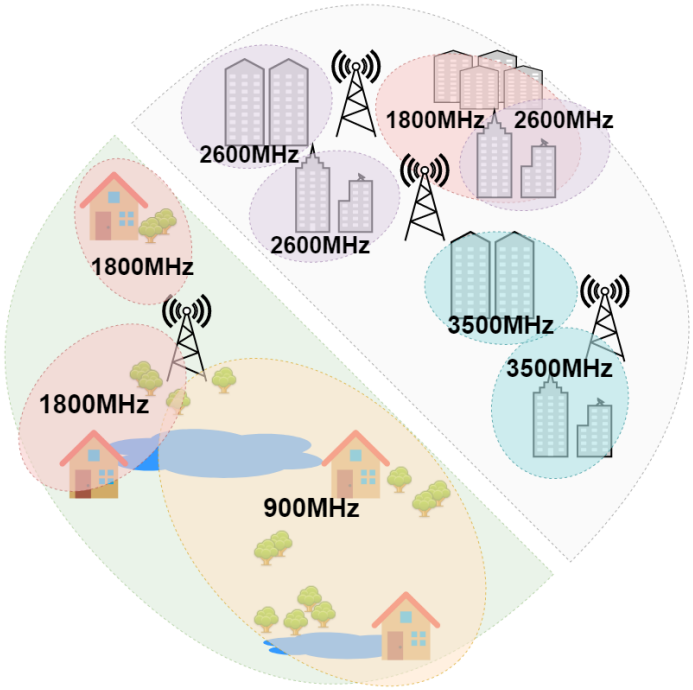
Kuvio 11 LTE-verkon toimintaperiaate, muokattu. 6.9.2020.
https://www.nsnam.org/docs/models/html/lte-design.html

3.5 Radioverkon parametrit

3.5.1 Solut tukiasemissa ja ohjaimissa

Mikä on solu?

Tukiaseman yhteyspisteet eli solut mahdollistavat päätelaitteiden langattoman kommunikoinnin ulkoverkkoon. Solut kattavat aluetta ympärillään sadoista metreistä kymmeniin kilometreihin. Solun kantavuus riippuu paljon siitä, minkä taajuista radiota lähetykseen ja vastaanottamiseen käytetään. Matalalla taajuudella katetaan laaja käyttäjäpiiri haja-asutusalueilla, mutta negatiivisena puolena nopeudet käyttäjillä on myös matalammat. Korkeammalla taajuudella vastakkaisesti katetaan kaupunkialuetta, missä käyttäjiä on enemmän ja langattomaan verkkoinfraan sijoittaminen mahdollistaa täten paremmat nopeudet päätelaitteille (ks. kuvio 12).



Kuvio 12 Solun kantama taajuuksittain

Solun olennaisimpiin parametreihin kuuluu solu-Id, solunimi, käytössä oleva taajuus ja sen sijaintia yksilöivät tunnukset kuten LAC ja TAC. Tehon solulle määrittää sitä kantava radio ja radioverkko-operoijan määrittämät parametrit tukiasemaan tai ohjaimeen.

GERAN

2G-teknologian solut löytyvät BSC-ohjaimelta. Luovutuksen tapahtuessa 3G:stä 2G:hen tarvitsee se kuitenkin ulkoisen solumäärityksen myös RNC:lle.

UTRAN

UMTS:n solut löytyvät myös ohjaimelta, mutta 3G-teknologiaa ylläpitävältä RNC:ltä. Ulkoisia UTRAN-soluja löytyy BSC:ltä ja 4G-tukiasemista mahdollistamassa päätelaitteen luovutuksen molempiin suuntiin.

E-UTRAN

LTE:n solut löytyvät vain itse tukiasemasta. Solut eivät itsessään ole uniikilla ID:llä varustettuja, vaan yksilöinti pohjautuu eNodeB:n tunnukseen eli eNBID:iin. Tämän takia LTE:ssä käytetään soluille nimitystä lokaali solu-Id, koska nimitys on jokaisen tukiaseman kohdalla yleensä sama. Näin jokainen solu voidaan yksilöidä käyttämällä eNBID:iä etuliitteenä.

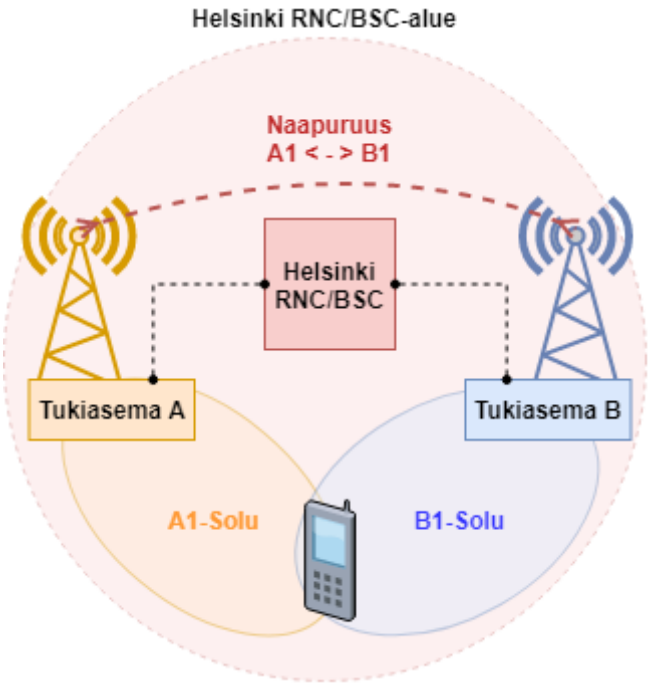
3.5.2 Naapuruusrelaatiot tukiasemien välillä

Mikä on naapuruusrelaatio?

Solun alla oleva päätelaite keskustelee ulkoverkon kanssa normaalisti käyttäjän ollessa paikallaan. Todellisuutta kuitenkin on, että päätelaitteen käyttäjä saattaa liikkua pienessä ajassa monen solualueen läpi ja täten palveleva tukiasema vaihtuu siinä samalla. Jotta yhteys päätelaitteen ja ulkoverkon välillä ei katkeaisi, tarvitsee tukiaseman luovuttaa päätelaite toisen solun/tukiaseman alle. Jotta luovutus tapahtuu katkoksetta, määritetään naapuruuden molempien päiden soluihin naapuruusrelaatio osoittamaan toisiaan. Relaation lisäksi pitää päätelaitteen kuunnella radiosignaalivoimakkuuksia ja ottaa käyttöön vahvimman tai priorisoidun taajuuden solu. Tällöin esimerkiksi autolla liikkuessa puhelut eivät katkea tai pätki kun vaihdetaan päätelaitetta palvelevaa solua useaan otteeseen.

Sisäinen relaatio

Sisäisellä relaatiolla tarkoitetaan solujen naapuruussuhdetta ohjaimen sisällä (ks. kuvio 13). Tällaisten naapuruuksien hallitseminen ja luonti on helppoa, kun kaikki tapahtuu yhdessä ohjaimessa.

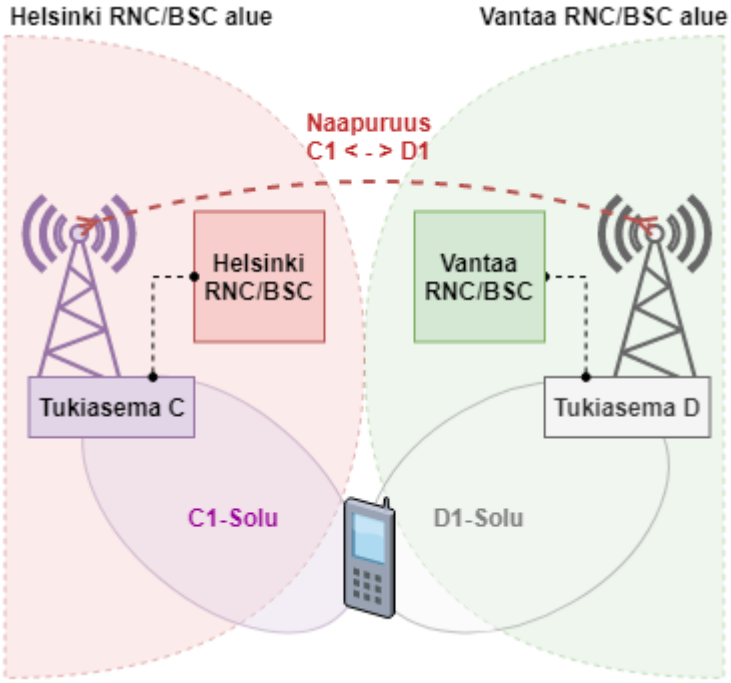


Kuvio 13 Sisäinen naapuruusrelaatio

Ulkoinen relaatio

Ulkoisella relaatiolla tarkoitetaan solujen naapuruussuhdetta kahden ohjaimen välillä

(ks. kuvio 14). Ulkoiset relaatiot tapahtuvat tilanteissa, kun tukiasema sijaitsee ohjaimen reuna-alueella. Nämä vaativat ulkoisen solun, joilla viitataan kohdeohjaimeen.



Kuvio 14 Ulkoinen relaatio

ANR

ANR eli *Automatic Neighbour Relation* on 4G-sukupolvessa kehittynyt automaattinen relaation luonti kahden tukiaseman välillä. ANR luotiin helpottamaan operaattoreiden naapuruuksien hallintaa tukiasemien määrän kasvaessa suurenevin määrin 4G-teknologian yhteydessä.

Frekvenssirelaatio

Frekvenssirelaatiossa luovutus tapahtuu määritettyyn taajuusalueeseen. Esimerkiksi jos käyttäjä on kiinni 3G:ssä ja siirtyy 4G:lle tapahtuu se frekvenssirelaation kautta. Utran soluun on määritetty taajuus, esimerkiksi 800MHz mihin siirto tapahtuu. Päätelaitteen saadessa parempia tasoja kyseisestä 800MHz:n 4G-solusta toteutuu luovutus 3G:ltä 4G:lle.

4 Alusta ja ohjelmointikielet

Alusta

Opinnäytetyössä työkalujen ohjelmointikielinä käytettiin Pythonia ja Expectiä. Alustana toimi Linuxin Red Hat Enterprise pohjainen palvelin. Palvelimelle ei voi asentaa omia ohjelmia ja sovelluksia, joten se alkujaan jo rajoittaa mahdollisuuksia tietyille ratkaisuille. Työkalut ja sen tarpeelliset pohjatiedostot on luotu oman käyttäjän kotihakemiston alapuolelle, mille on myös asetettu vaadittavat käyttöoikeudet. Linux alusta mahdollistaa myös ajoitettujen toimintojen ja skriptien ajamisen Crontab:lla. Crontab:iin voidaan määrittää pyörimään joku ajoitettu toiminto, joka toimii esimerkiksi minuutin välein. Tämä osoittautui hyödylliseksi toiminnoksi tukiaseman poistoprosessissa silloin, kun verkkolaitteisiin ei ollut toteutus hetkellä yhteyttä ja prosesseja voitiin jättää myöhemmille ajankohdille ajastetun skriptin hoideltavaksi.

Python

Python on matalan tason ohjelmointikieli, mikä nojaa laajaan kirjastokantaan. Kirjastoilla mahdollistetaan monimutkaisempia sovellutuksia ja koodin kirjoittamisesta tulee helpompaa. Opinnäytetyössä käytettiin Pythonin 2.7 versiota, sen ollessa ainoa mitä käytetty alusta tarjosi. (Applications for Python, n.d.)

OS-kirjastolla (lyh. Operating System) toteutetaan Pythonissa käyttöjärjestelmän toimintoja, missä skriptiä juostaan. Työssä kirjastoa käytettiin esimerkiksi polkujen määrittämiseen ja hakemiseen.

SYS-kirjastolla vaikutetaan Pythonin tulkitsijan toimintaan. Opinnäytetyössä tätä käytettiin ohjaamaan muun muassa työkalun luomaa tulostetta sekä lisätäkseen PATH:iin muita polkuja.

Shutil:lla saadaan toteutettua tiedostojen siirtämistä, muokkaamista ja käyttöoikeuksien siirtämistä. Työssä sillä saatiin kopioitua ja siirrettyä työkalun tuottamia tulosteita ja pohjatiedostoja.

Getpass tarjoaa kaksi funktiota, joilla se käsittelee skriptin käyttäjän käyttäjänimeä tai salasanaa. Tämän kirjaston avulla saatiin opinnäytetyössä käyttäjän kotihakemistoon luotua tiedostoja ja kansioita, koska käyttäjätunnus on osa kotihakemistoa.

Time-kirjasto tarjoaa erilaisia aikaan liittyviä funktioita. Sillä voidaan ottaa talteen nykyinen kellonaika erilaisissa muodoissa, pysäyttää prosessin toiminta määritetyksi ajaksi tai muokata aikaan liittyviä ominaisuuksia.

Subprocess-kirjastolla voidaan luoda uusia aliprosesseja, joista saadaan syöte, tuloste ja virheviestit. Työssä tällä voitiin esimerkiksi tarkistaa yhteys verkkolaitteeseen.

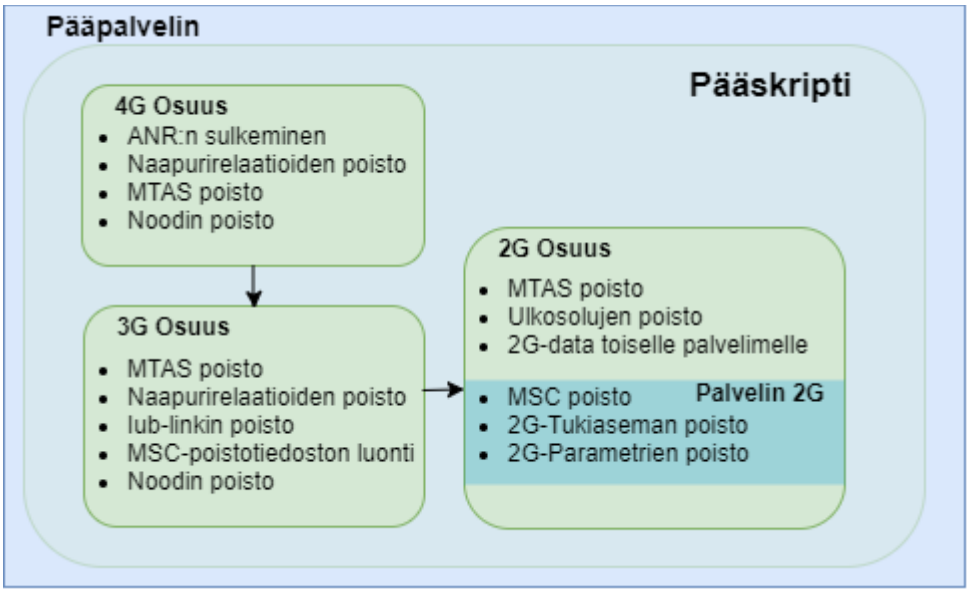
Expect ja Bash

Expect-ohjelmointikieli on yksinkertaisimmillaan vain komentojen ajamista käyttöjärjestelmässä skriptin avulla. Opinnäytetyössä se tuli tarpeen, kun Python ei omilla oletuskirjastoillaan kyennyt komentamaan verkkolaitteita toivotun tavan mukaisesti. Expect yhdistää käyttäjän antaman ja komentokehotteen vasteen, jonka jälkeen itse syötetty komento voidaan suorittaa. Jotta Expect-skripti saatiin paremmin toimimaan yhteistyössä palvelinympäristön kanssa, sekoitettiin se Bash-koodin sekaan. Bash toimii komentorivin tulkkina oletuksena Linux-käyttöjärjestelmissä. Sovellukset ja komennot, joita Linux-ympäristössä ajetaan, ovat usein Bash:lla ajettuja. Koska Bash on vastuussa useiden ohjelmien toiminnasta, voidaan sitä käyttää myös uusien sellaisten ohjelmoimiseen. Se kykenee käyttämään tiedostoja järjestelmässä ja kommunikoimaan tietokantojen kanssa. Yleisellä tasolla Bash:ia voidaan hyödyntää arkisien operaatioiden kanssa sen vähäiselläkin ymmärtämisellä. (What is a Bash Script, 2020) Opinnäytetyössä Bash:ia ei lopulta käytetty kuin sekoittaakseen siihen Expect-koodia, jolla tuotettiin haastavampiin etäyhteyskohteisiin tiedonhaku. Skriptin hakema tuloste ohjattiin tiedostoon, mistä Pythonilla toimiva päätyökalu keräsi tarpeelliset parametrit.

5 Työn toteutus

5.1 Työkalun kokonaiskuva

Työkalun kehitys aloitettiin käymällä läpi käsin mitä kaikkea tukiasemakohtaista dataa löytyy radioverkosta. Ensimmäisenä prosessissa voitiin poissulkea lähes kokonaan tukiasemasta löytyvät tiedot. Tämä siksi, koska tukiasemaan ei ole suurella todennäköisyydellä yhteyttä poistotehtävän alkaessa. Poistojärjestys on suunniteltu mahdollistaakseen työkalun uudelleenajon, jos skripti jostain syystä keskeytyisi. Työkalun ajossa käytetyt parametrit ja tiedot poistuvat verkosta siis viimeisenä. Työkalu-skriptiä ajetaan laitevalmistajan oman palvelinympäristön päällä, missä sijaitsee lähes kaikki työkalun toimintaan liittyvät liitetiedostot ja -sovellukset (ks. kuvio 15). Osan teknologiasta ollen jakautuneen vanhempaan palvelinympäristöön tarvitaan osa skriptistä rakentaa siihen ympäristöön. Tämä esiintyy työssä nimellä 2G-palvelin ja alkuperäinen palvelin ympäristö nimellä Pääpalvelin. Työkalun ulkopuolella toimiva relaationtarkistus-skripti ohjataan ajetuksi Crontab:lla, mikä ajastaa toiminnon suorituksen kahteen kertaan päivässä. Relaationtarkistus tapahtuu Pääpalvelimella ja lokitiedot tulostuvat myös sinne omaan kansioonsa.

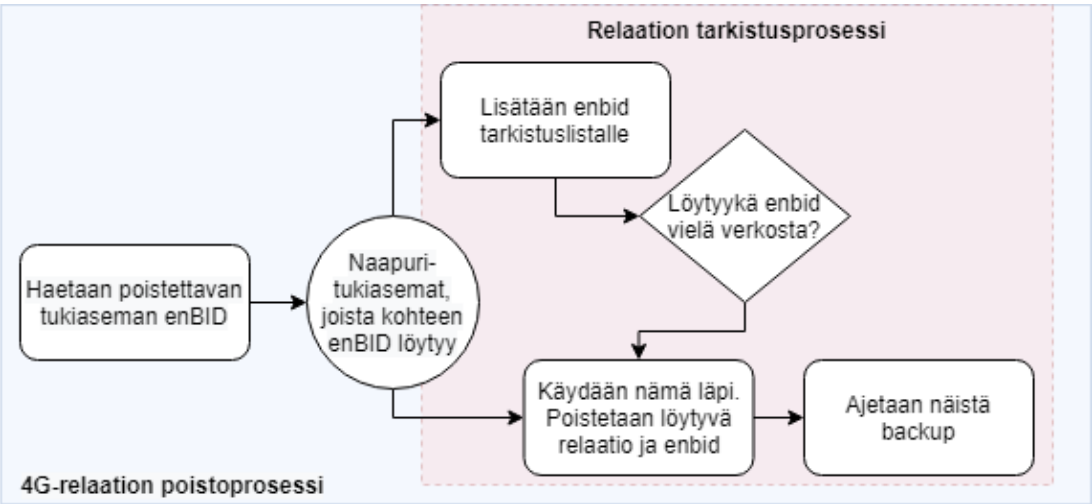


Kuvio 15 Prosessin topologia

5.2 Toimenpiteet 4G:ssä

ANR ja naapuruusrelaatioiden poisto

Riippuen siitä onko tukiasema vielä yhteydessä asiakasverkossa, täytyy poistettavasta tukiasemasta käydä asettamassa solut lukittuun tilaan ja poistaa ne. Tämä johtuen siitä, että ANR luo automaattisia naapuruuksia 4G-solujen välillä, jolloin prosessin seuraavassa vaiheessa tehdyt toimenpiteet menevät hukkaan ja osa tehtävistä jäisi jälkikäsittelyyn. Tämän jälkeen hoidetaan naapuritukiasemien relaatioiden ja ulkosolun poisto (ks. kuvio 16). Verkosta haettiin poistettavan 4G-tukiaseman tukiasematunnuksen avulla ulkoiset solut, jotka voitiin iteroimalla poistaa verkosta helposti. Naapuruusrelaatiot 3G:stä 4G:hen ollessa frekvenssiin kohdistuvia eikä soluun tähtääviä, eivät täten vaadi toimenpiteitä poistolle.



Kuvio 16 4G-relaation poistoprosessi

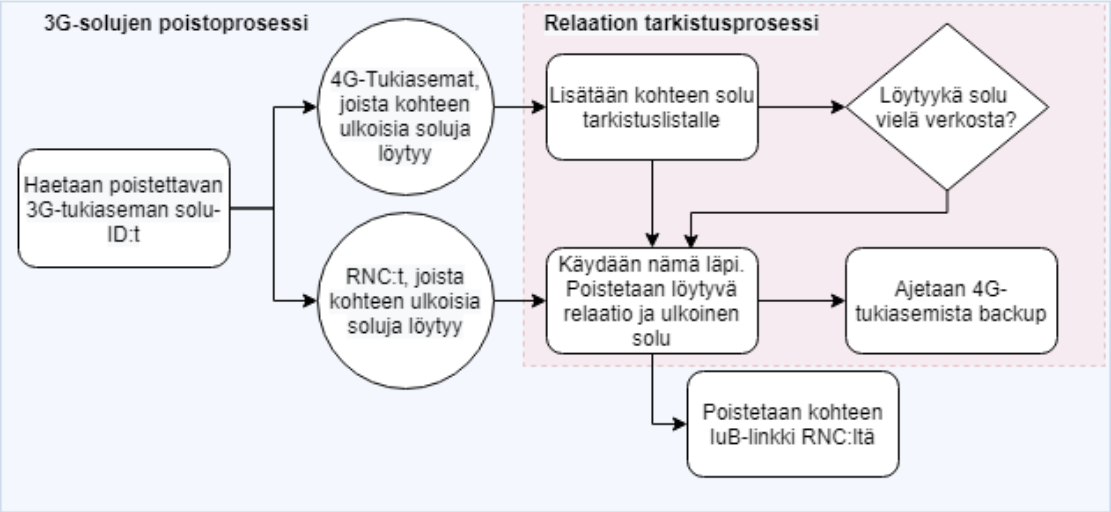
MTAS poisto palvelimelta

MTAS:n poistoon käytettiin toimeksiantajan valmista skriptiä, joka ajettiin saman palvelimen päällä missä itse pääskriptikin pyöri. MTAS osuus toteutettiin käyttäen OS-kirjastoa Pythonissa, jolloin skripti suoritti sen ajoympäristössä. Tiedot poistoa varten vedettiin tietokannasta, jossa käytössä olevien tukiasemien tiedot ovat tallessa.

5.3 Toimenpiteet 3G:ssä

Naapuruusrelaatiot 3G-tukiasemassa

Samalla tavalla miten 4G:ssä naapurisolut ja relaatiot haettiin ja poistettiin, voidaan 3G:ssä seurata samalla kaavalla kyseistä poistoprosessia. Skripti tekee iteratiivisen listauksen jokaisesta poistettavan 3G-tukiaseman solusta ja hakee löytyvät relaatiot ja ulkosolut. Naapuruuksia löytyy 3G-solujen välillä sekä myös 4G-solusta 3G-soluun (ks. kuvio 17). Aikaisemmin mainitussa tapahtuu poisto RNC:ssä. Riippuen löytyykö tukiasema RNC:den rajalta, voi poistoa tapahtua myös toiselta kuin tukiasemaa hallitsevalta ohjaimeltakin. Näissä tapauksissa myös ulkoinen solu poistetaan ohjaimesta. Jälkimmäisessä tapauksessa tapahtuu naapuruusrelaation poistaminen 4G-tukiasemalta. Kuten mainittuna 4G osuudessa, voi poistettava naapuruus löytyä tukiasemasta mikä ei ole yhteydessä hallintaverkkoon. Näissä tapauksissa lisätään kyseinen relaatio jälkikäsittelyn listalle.



Kuvio 17 3G-solujen poistoprosessi

Työkalun skriptissä kerätään 3G-solujen Id:t listalle, joita käytetään 3G-osuuden eri vaiheissa. Poistettavan kohteen ulkoisia 3G-soluja haetaan 4G-tukiasemista globaalilla haulla. Tästä saadaan Pythonilla elementti, mistä voidaan iteratiivisesti käydä läpi kaikki tukiasemat mihin haku kohdistuu. Jokaisen tukiaseman kohdalla suoritetaan relaation poisto ja kyseinen kohde lisätään varmuuskopiointilistalle, joka ajetaan skriptin päättyessä. Naapuruudet ja ulkoiset solut 3G-solujen välillä poistetaan RNC:ltä lähes samalla menetelmällä. Kun poistoprosessi on ajettu molemmille tekniikoille, suoritetaan sama haku millä saadaan 3G:n ulkoiset solut. Jos haku tuottaa tuloksia, lisätään ne yksilöittäin jälkitarkistuslistalle (ks. kuvio 18).

```
Solulista = []
try:
#Lisätään 3G-Solut listalle
    for row in Solut3G.output():
        Solulista.append(str(row[2]))
    for Solu in Solulista:
        GetSite = "Hae 3G-Ulkoinen-Solu=%s" % (Solu)
        try:
        #Poistetaan 4G->3G Naapuruudet
            for row in GetSite.output():
                Neigh_Site = row[0]
                Ext4G3G = "Poista %s 3G-Naapuruus=%s" % (Neigh_Site, Solu)
                print(execute(Ext4G3G))
                backup.append(Neigh_Site)
        except Exception as e:
            print("\n!!!  %s !!!\n" % e)
        try:
        #Poistetaan 4G->3G Ulkoinen solu ja 3G<->3G naapurit/solut
            Ext4Gc = "Poista 3G-Ulkoinen-Solu=*%s" % (Solu)
            Ext3G3G = "Poista 3G-3G-Naapuruus=%s" % (Solu)
            Ext3Gc = "Poista 3G-Ulkoinen-Solu=%s" % (Solu)
            txt("Poistetaan %s Ulkoiset Solut " % (Solu))
            print(execute(Ext4Gc,Ext3G3G,Ext3Gc))
        except Exception as e:
            print("Ei Extcellia\n %s" % e)
    #Lisätään löytyvät solut korjauslistalle
    for Solu in Solulista:
        try:
            with open(Remaiming_Relations, "a") as outf:
                GetSite = "Hae 3G-Ulkoinen-Solu=%s" % (Solu)
                try:
                    for row in GetSite.output():
                        Neigh_Site = row[0]
                        outf.write("3G;%s;%s;\n" % (Solu,Neigh_Site) )
                except:
                    pass
        except Exception as e:
            print("Error : %s" % e)
```

Kuvio 18 3G Naapuruuksien ja ulkoisten solujen poistaminen

Iub-linkki

Iub-linkin poiston yhteydessä poistuu kaikki soludata ja parametrit kyseisen 3G-tukiaseman kohdalta. Tällöin lähtee kohdetukiaseman 3G-soluista lähtevät naapuruudet ja loputkin viittaukset tukiasemaan radioverkkopuolella. Poisto tapahtuu tukiasemaa hallitsevalla RNC:llä ja ei itse toimenpiteenä ole haastava.

MSC tietojen poistaminen 3G:ssä

Solutiedot MSC:ltä voidaan poistaa vain toisen erillisen palvelimen kautta. Tiedot 3G-soluista siirtyvät 2G-tietojen mukana mikä selitetään 5.4 kappaleessa tarkemmin. Uuden korvaavan tukiaseman solujen tietojen, vaikka ollessakin erilaisia, voi konfliktia syntyä tiedonkäsittelyssä MSC:llä ja sekoittaa raportteja tuloksista. Tällöin on vain hyvää käytäntöä poistaa nämäkin tukiaseman tiedot.

Uudelleen ajaminen

Poistaessa tukiasemaa voi aina olla mahdollista työkalun kaatuminen. Vaikka työkalu on kehitetty poistamaan kriittisimmät tiedot mitä tarvitaan prosessin muissa vaiheissa viimeisenä, ei kaikkeen pystytä valmistautumaan. Esimerkkinä viimeisenä vaiheena toteutettu 2G-osio vaatii 3G-tietoja, koska sieltä löytyy ulkoisia soluja tähän suuntaan. Työkaluun lisättiin siksi ominaisuus syöttää käsin 3G-tietoja, jos itse NodeB olisi tuhottu ja tietoja siitä ei voisi enää hakea verkosta. Python-skriptissä (ks. kuvio 19) yritetään hakea tukiasemaa ja sen tietoja verkosta. Jos tukiasemaa ei löydy on se merkki siitä, ettei sitä ole enää olemassa ja voidaan ohittaa 3G-poistovaihe. 2G-skriptiä varten syötetään 3G-tukiaseman RNC ja solunimet. Kun tarvittavat solunimet on lisätty, voidaan lopettaa syöttäminen ”Q”:lla ja työkalu jatkaa GSM-osioon

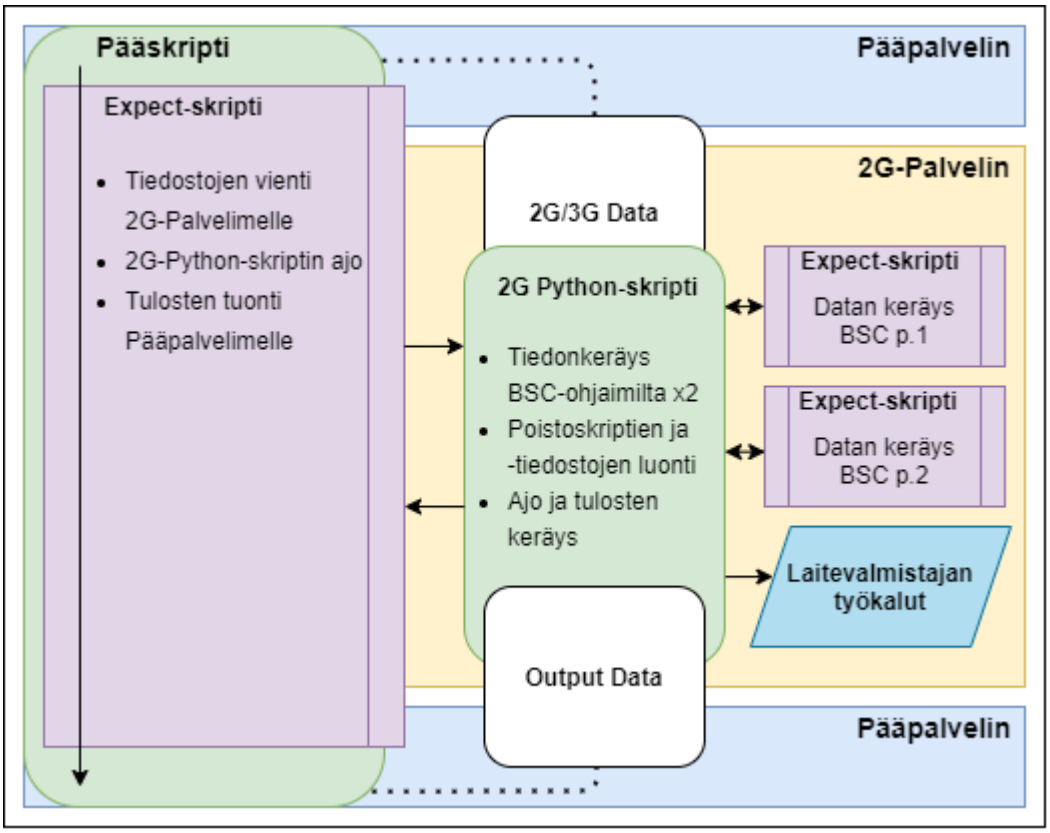
```
msclista = []
Swap3G = 'Hae %s'  % (saitti)
try:
    for row in Swap3G.output():
        print("Poistetaan seuraava saitti %s" % (str(row[0])))
        RNC_2G = str(row[1])
except Exception as e:
    txt("%s Not Found -->" % (saitti))
    print("\n!!!  %s !!!\n" % e)
    try:
        RNC_2G = raw_input("Syota tukarin RNC : ")
        while True:
            korjaus = raw_input("Syota kaikki 3G-Solunimet (Q, kun kaikki lisatty) : ")
            if korjaus not in ("Qq"):
                korjaus = korjaus.replace(" ","")
                msclista.append(korjaus.upper())
            if korjaus in ("Qq"):
                break
        #Ajetaan GSM-Työkalu ja päätetään skriptin toiminta
        gsmtool()
        sys.exit()
    except Exception as e:
        print("\n!!!  %s !!!\n" % e)
        sys.exit()
```

Kuvio 19 Käsinsyöttö 2G-puolta varten

5.4 Toimenpiteet 2G:ssä

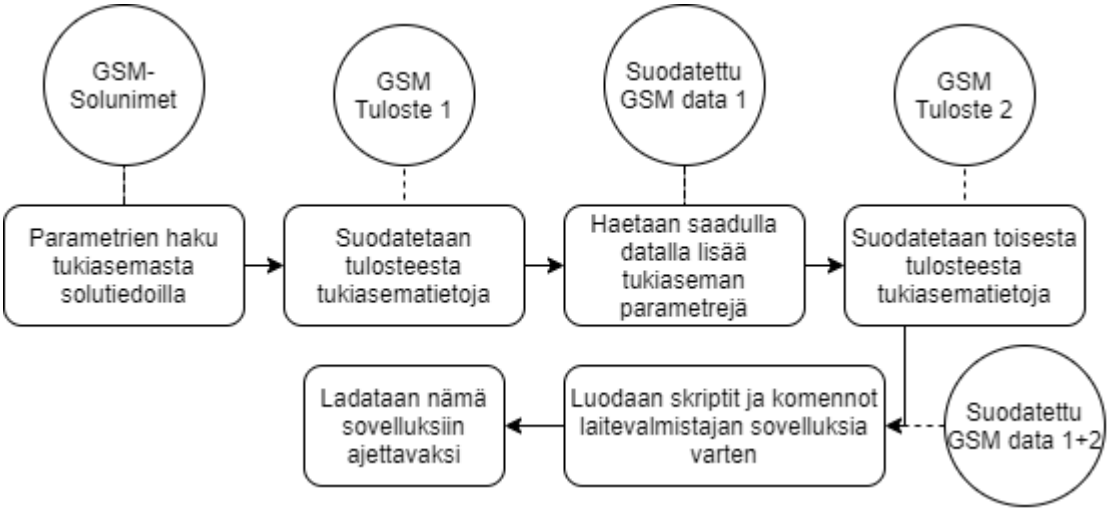
Eroten 4G:stä ja myös hiukan 3G:stä tapahtuu 2G-tukiaseman parametrien ja datan poistaminen lähes kokonaan toisella erillisellä palvelimella. Yhdistääkseen pääskriptin toiminnan toiseen palvelimeen toteutuksessa käytetään *Expect*-skriptikieltä Python-kirjastojen ollessa rajallisia mahdollistaakseen ssh/scp:n hyödyntämisen. Pääskriptillä ajetaan Expect-skripti, joka vie 2G/3G-datan 2G-Palvelimelle. Toiselle palvelimelle on tehty Python-skripti, joka hakee taas omilla Expect-skripteillä tarkat tiedot BSC-

ohjaimilta. Tästä datasta muodostetaan tapposkriptit laitevalmistajan omiin työkaluihin, mitkä 2G-Palvelimen Python-skripti pistää käyntiin (ks. kuvio 20).



Kuvio 20 2G-Skripti

Pääpalvelimelta yhdistäessä 2G-Palvelimelle käytetään salauksena esijaettua avainta. Pääskripti ajaa Expect-skriptin, minkä toimenpiteitä on siirtää 3G-tukiaseman dataa tiedostona kohdepalvelimelle scp:llä. Tämän jälkeen kutsutaan 2G-Palvelimella Python-skripti, joka annetuilla tiedoilla lähtee keräämään 2G-tukiaseman tietoja sitä hallitsevasta BSC:stä. Tämä joudutaan tekemään kaksi kertaa johtuen siitä, että Expect-skriptillä saadut tiedot ovat sellaisessa formaatissa, mitkä ovat helpompi parsia kuntoon Pythonin avulla. Toisella juoksukerralla käytetään aikaisemmin saatuja tietoja millä saadaan tarkat tiedot poistokomentoja ja tiedostoja varten. Laitevalmistajan työkalut tuottavat tulosteen mikä palautetaan takaisin pääskriptille ja saadaan täten vakuus onnistuneista prosesseista. Tiedonhaku- ja poistoskriptinprosessi havainnoituina kuviossa 21.



Kuvio 21 2G Poistoprosessi

Poistettavat verkkoelementit 2G-tekniikassa ovat hyvinkin samat mitä 3G:ssä poistettiin, jokseenkin eri palvelinalustalla ja käyttäen laitevalmistajan eroavia työkaluja. Poisto jakautuu kahteen osaan, tukiasemaan määritettyjen laitteistojen poistamiseen sekä parametrien poistamiseen. Nämä molemmat prosessit tapahtuvat BSC:llä, mutta erilaisessa formaatissa ja käyttäen laitevalmistajan eri sovelluksia. Tukiaseman laitteistoon kohdistuvassa poistossa lähtee kohteen paikalliset solutiedot ja niitä hallitsevan yksikön konfiguraatiot. Radioverkkoon määritetyt naapuruudet, soluparametrit ja Abis-linkki poistetaan ohjaimesta käyttäen laitevalmistajan toista sovellusta. Tämä sovellus toimii graafisella pohjalla, mutta mahdollistaa poistettavien tietojen tuonnin tiedostona. Tukiaseman poistotyökalu aikaisemmin haettujen tietojen perusteella luo tällaisen tiedoston ja lataa sen sovellukseen. Loppu prosessista joudutaan tekemään käsin muutaman toimenpiteen työkalua tehdessä olleen mahdottomia automatisoida.

Tiedonhaku 2G-palvelimella

Muokatuksi esimerkkikoodiksi otettu DU-yksikön poistotiedostojen luonti suoritetaan sekoittamalla Pythonia (ks. kuvio 22), Expectiä ja Bashia (ks. kuvio 23). Python toimii 2G-palvelimen päällä pohjana skripteille ja hoitaa tarvittavien tiedostojen sekä palveluiden käynnistämisen. Periaatteena se valmistaa mallitiedostoista suoritettavia Expect/Bash-skriptejä, mitkä puolestaan hakivat BSC-ohjaimelta tukiaseman tietoja.

Mallitiedostoon saadaan muokattua kohteen BSC, solut, tulostetiedosto ja tukiaseman yksikön malli. Kun poistettavan kohteen Expect/Bash-skripti on luotu, suoritetaan se käyttämällä Pythonin OS-kirjastoa.

```
def DU():
    path = "/Polku_Tukiaseman_Kansioon/"
#Expect-Skriptit, tulosteet ja sovellus-tiedostot
    SolufDst = "%srauta2g.exp" % path
    RautafDst = "%skaikki2g.exp" % path
    Solu_Out = "%srauta2g.txt" % path
    Rauta_Out = "%skaikki2g.txt" % path
    Softa1 = "%s%s_Softa1.txt" % (path, saitti)
    Softa2 = "%s%s_Softa2.txt" % (path, saitti)
#Luetaan pohjatiedosto
    with open(SolufSrc, 'r') as outf:
        filee = outf.read()
#Korvataan pohjatiedostosta #-merkatut kohdat
    for index,Solu2g in enumerate(Solut,start=1):
        filee = filee.replace('#Solu'+index, Solu2g)
    #BSC ja Solut saatu aikaisemmalla haulla viittaamalla tukiaseman nimeen
    filee = filee.replace("#BSC",bsc).replace("#OutPut", Solu_Out).replace("#RAUTA", "DU")
#Kirjataan muokattu teksti omaan tiedostoon
    with open(SolufDst, "w") as outf:
        outf.write(filee)
#Ajetaan tehty Expect-skripti
    os.system("/bin/bash %s" % SolufDst)
```

Kuvio 22 2G Poistotiedoston prosessi

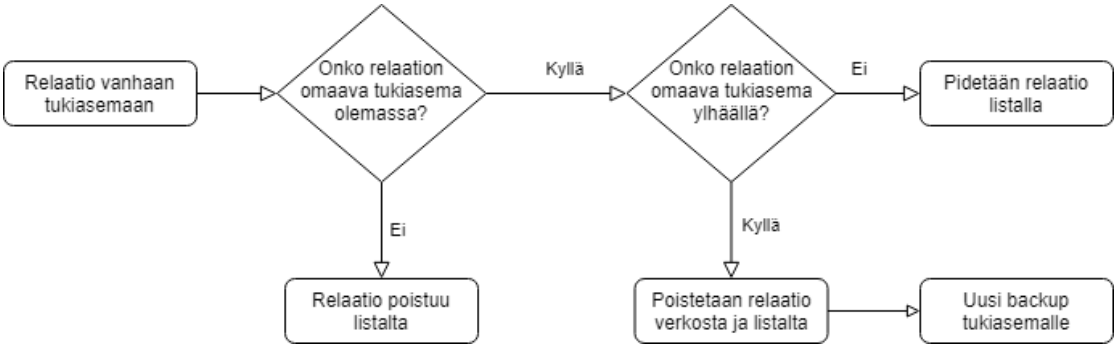
Bash:iin sekoitettu Expect-koodin tarkoitus on luoda yhteys kohdetukiasemaa hallitsevaan BSC:hen ja hakee sieltä solunimeen viittaavia DU-yksikön tietoja. Funktio ”HaeInfoa” (ks. kuvio 23) tehdään jokaiselle solulle silmukoinnin kautta. Kuvio 23:ssa ”log_user 1” tuottaa tulosteen ajetuista komennoista ja skriptissä myös ohjataan kyseinen tuloste määrättyyn kohdetukiaseman tiedostoon. Samanlainen Bash/Expect-skripti toteutetaan myös toisella kerralla, jolloin tuloksena haetaan vain eri tietoa. Kun molemmat tiedonhakuprosessit on tehty, voidaan luoda laitevalmistajan sovelluksia varten tarvittavat skriptit ja tiedostot. Nämä saadaan osittain skriptin kautta myös suoritukseen, mutta vaatii käyttäjältä toimenpiteitä.

```
#!/bin/bash
cd $(dirname $0)
# Haetaan ohjaimelta tulosteet ja ohjataan tiedostoon
HaeInfoa()
{
/usr/bin/expect << Loppu >> #OutPut
log_user 1
spawn ssh #BSC
expect ":"
send -- "UNIT=#RAUTA, Solu=$Solunimi;\r"
expect ":"
send -- "quit\r"
Loppu
echo "" >> #OutPut
echo "" >> #OutPut
}
# Solut, johon komennot kohdistuu looppina
for Solunimi in #Solu1 #Solu2 #Solu3 #Solu4 #Solu5
        do
                HaeInfoa
done
exit
```

Kuvio 23 Expect/Bash -skriptiosuus

5.5 Jälkitoimenpiteet

Pääskriptin päästyä loppuun asti yleisenä olettamuksena on, että vaihtuvaan kohteeseen viittaavat parametrit on poistettu. Kuitenkin muiden verkkolaitteiden alhaalla olon takia saattaa osa prosesseista jäädä tekemättä työkalussa. Tämän vuoksi työkalu kerää poistettavien kohteiden statuksen ja tekee niistä koosteeksi loppuun tulostuksen ja oman tiedoston. Tapauksessa missä esimerkiksi 4G-solujen välinen relaatio on jäänyt poistamatta kohteen naapuritukiaseman ollessa poissa käytöstä työkalun ajohetkellä, kehitettiin ajastettu skripti mikä juoksee tiettyinä kellonaikoina jääneet relaatiot poistettavaksi (ks. kuvio 24). Tämä ajastettu toiminto suoritetaan kahteen kertaan päivässä työajan ulkopuolella. Tällöin alhaalla olevat tukiasemat voisivat olla todennäköisemmin toiminnassa, koska tukiasemien huoltotoimenpiteet tuppaavat asettumaan toimistotyötuntien ajoille.



Kuvio 24 Relaation tarkistusprosessi

Relaatioiden tarkistusprosessi saadaan alkuun avaamalla tiedosto, mihin poistotyökalu on kirjannut suorittamien tukiasemakohteiden sukupolven, solutietoja ja naapuruuksia. Tämän tiedoston data avataan helpommin käsiteltävään muotoon ja sarakkeille määrätään omat tunnukset. Malliesimerkiksi (ks. kuvio 25) on otettu 4G-tekniikkaan kohdistuvat relaation tarkistukset. Skripti suorittaa ehdon tekniikan tarkistukselle, kuuluuko se 3G- vai 4G-sukupolvelle. Kohteen ollessa 4G tarkistetaan kohteen verkkostatus. Jos kohde ei ole hallintaverkkoon yhteydessä ei voida toimintoja myöskään suorittaa. Jos kyseinen käsky ei anna ollenkaan tulosta, on tämä varma merkki siitä, että kyseinen tukiasema on myös poistettu, jolloin prosessi tämän kohdalla voidaan unohtaa. Tukiaseman ollessa linjoilla hallintaverkossa suoritetaan samanlaiset operaatiot mitä nähtiin kappaleessa 5.2, jossa poistettiin naapurillisten tukiasemien naapuruus ja eNBID viittaukset poistokohteeseen.

```
def Clear_Relations():
    for rel in relaatio:
        rel = rel.replace("\n","")
        info = rel.split(";")
        #Jakaminen osiin
        Gen = info[0]
        Solu = info[1]
        Site_Neig = info[2]
        Get_Sync = "Check %s SynkkaStatus" % (Site_Neig)
        ### 4G ###
        if Gen == "4G":
            #Komennot eNodeB:n poistoon
            DelRel4g = "Poista %s 4G-Relaatio=%s" % (Site_Neig, Solu)
            DelFunc4g = "Poista %s 4G-EnodeB=%s" % (Site_Neig, Solu)
            try:
                for line in Get_Sync[0]:
                    if str(line[1]) == "SYNCHRONIZED":
                    #Jos tukiasema synkronoitu, suorita komennot
                        try:
                            print(Site_Neig)
                            print(execute(DelRel4g) )
                            print(execute(DelFunc4g) )
                        except Exception as e:
                            print("\n!!!  %s !!!\n" % e)
                    else:
                        print("\n%s - No Connection\n" % (Site_Neig) )
            #Jos Get_Sync ei toimi, tukiasema on poistettu
            except:
                print("%s - Tukiasema Poistettu" % (Site_Neig) )
```

Kuvio 25 Relaatiotarkistus 4G:ssä

Kun lista tarkistettavista relaatioista on käyty läpi, voidaan kirjoittaa relaatiotarkistus-tiedosto uudestaan. Tämä hoituu käymällä läpi listan kohteet ja viedään ne ehtojen kautta läpi. Esimerkki koodiosuudeksi otettu taas 4G (ks. kuvio 26), missä suoritetaan sama datan mallintaminen luettavaksi. Jotta tiedetään mitä kirjataan takaisin tiedostoon uuteen relaatiotarkistus-käsittelyyn, käytetään hakutulokset ehtojen läpi. Jos ulkoista solua käsiteltävälle tukiasemalle ei enää löydy suoritetaan kohteesta varmuuskopiointi. Ulkoisen solun vielä löytyessä käsiteltävästä tukiasemasta ajetaan tämän data takaisin tiedostoon.

```
def Check_and_Rewrite():
    for rel in relaatio:
        rel = rel.replace("\n","")
        info = rel.split(";")
        #Jakaminen osiin
        Gen = info[0]
        Solu = info[1]
        Site_Neig = info[2]
        #Tulos jos tukiasemaa ei löydy
        No_Inst = "Ei tuloksia"
        ### 4g ###
        if Gen == "4G":
            #Tarkistus Ulkoinen eNodeB Löytyy
            GetenBF = execute("Hae %s Ulkoinen_eNodeB=%s" % (Site_Neig, Solu)).output()
            #Jos ei -> BackUp
            if No_Inst in GetenBF:
                print("%s - Ei tuloksia -> Poistuu listalta" % Site_Neig)
                BackUP(Site_Neig)
            #Jos löytyy, appendaus takaisin listaan
            else:
                unsynched.append(rel)
    ## Kirjataan epäsynkronoidut tukiasemat takaisin tiedostoon
    print("Tukiasemat, mitkä vielä putsaamatta ->")
    for remaining in unsynched:
        print(remaining)
    with open(Uncleared,"w") as outf:
        for remaining in unsynched:
            outf.write("%s\n" % remaining)
```

Kuvio 26 Epäsynkronoitujen kirjaus takaisin

6 Johtopäätökset ja pohdinta

6.1 Tavoitteet

Opinnäytetyön tavoitteena oli lähteä tuottamaan olemassa olevalle ja pitkälle tulevaisuuteen jatkuvan prosessin nopeuttamista ja automatisointia toimeksiantajalle. Tässä prosessissa keskipisteenä oli vanhan tukiasematekniikan poistaminen verkosta kokonaan, koska uusi korvaava tukiasema tulisi konfliktiin tämän kanssa ja oli täten olennaista saada tiedot pyyhittyä. Näiden tapahtuvien prosessien määrän ollessa tuhansissa ja jatkuen useampia vuosia, tarvitsi luodun työkalun skriptin olla hyvin kommentoitu, jos sen kehitystä joudutaan siirtämään muille henkilöille. Tämän lisäksi myös verkon elementeissä ja sen sovellusten versioissa voi tapahtua muutoksia, mitkä voivat vaikuttaa työkalun skriptin toimintaan.

Ensimmäisinä vaiheina työkalulle oli tavoitteena saada poistettua tukiaseman 4G- ja 3G-tekniikoiden sisältämät tiedot, koska nämä olivat ehkä yksinkertaisin alue aloittaa. Tämä johtui lähinnä vaadittavien poistoprosessien ollessa helppo toteuttaa Pythonilla laitevalmistajan omien Python-kirjastojen takia. Samaan aikaan oli tärkeää, että työkaluun syötettävät tiedot kävisivät tarvittavien ehtojen läpi ennen kuin se lähtisi toteuttamaan poistoprosessia. Näitä ehtoja oli esimerkiksi tukiaseman löytyminen verkosta, vain tietynlaisten merkkien salliminen ja käyttäjän antaman syötteen tuloksen tarkistus verkosta. Näillä varmistettaisiin oikean kohteen poistaminen ja ei vahingossakaan poistettaisi verkosta mitään mitä ei pitäisi. Redundanttisuuden ja varatoimenpiteiden vuoksi oli asetettu myös ehdoiksi, että työkalu loisi tulosteen käydyistä toimenpiteistä ja teettäisi tarvittavat komennot ja tiedostot niihin osioihin missä tukiaseman tietojen puhdistaminen ei onnistunut. Nämä voitaisiin ajaa joko käyttäjän toimesta myöhempänä ajankohtana tai antaa automatisoidun skriptin hoitaa puhdistettavat kohteet uudestaan. Viimeisimpänä tavoitteiden toteuttamisjärjestyksessä työkalulle opinnäytetyön versiossa oli 2G-puolen automatisointi. Tämä lähinnä sen takia, kun manuaalisesti hoidettuna prosessi 2G:ssä ei veisi niin merkittävää aikaa mitä 3G ja 4G osiot veisivät. Toisena syynä 2G oli erillisen palvelinalustan

päällä missä syntaksi ei ollut samanlainen ja vaati pientä innovointia puutteellisten Python-kirjastojen takia.

6.2 Tulokset

Opinnäytetyön lopputuloksena saatiin työkalu halutun tehtävän suorittamiseksi, eli vaihtuvan tukiaseman tietojen poistamiseen. Tässä tehty työkalu suorittaa prosessin alusta loppuun vaatien vain alle minuutin käyttäjän omaa aikaa. Poistoprosessi etenee vaiheittain aina 4G-teknologiasta 2G:hen. Virheiden tapahtuessa ne tulostuvat vasteeseen ja työkalun toiminta pysähtyy. Suurin osa ongelmista syntyy käyttäjän väärin syöttämästä tukiaseman tunnuksesta ja työkalu ei tällöin voi hoitaa työvaiheita ja sen toiminta pysähtyy. Jos tukiaseman tunnus on syötetty kuitenkin oikein, on mahdollisuutena syöttää käsin tämän tukiaseman tietoja komentokehotteeseen, jos niitä ei enää hallintaverkosta löydy. Tätä varten olisi voinut tietenkin kerätä toimeksiantajan eri tietokannoista varmuuden vuoksi tukiasemien parametritietoja ja muita olennaisia ominaisuuksia tällaisten tilanteiden varalta, kun tietoja ei enää hallintaverkosta löydä. Tämä ei kuitenkaan toteutunut työssä, kun oli prioriteettina melko kaukana halutummista toiminnoista.

Tekniikoista 3G ja 4G saatiin automatisoitua lähes kokonaan. Jopa siihen tilanteeseen, kun käynnistetty poistoprosessi ei hoitanut jokaista operaatiota onnistuneesti, jolloin epäonnistumiset kirjattiin muistiin ja ajettiin automaattisesti myöhemmällä ajankohdalla. Tähän ei päästy 2G:n osalla, kun operointimenetelmät olivat rajusti erilaiset ja painottui enemmän käyttäjän toimenpiteisiin. Jokseenkin nämäkin todella minimaalisina ja vaatii käyttäjältä vain raportin tarkistusta ja laitevalmistajan sovelluksen käynnistämistä. Näihin haetaan vielä parannuksia ja uusia tapoja toteuttaa jatkokehityksen vaiheessa.

6.3 Jatkokehitys

Työkalu saatiin luotua tilaan, missä se toteuttaa lähes kokonaan automaattisesti prosessin koskien poistuvaa tukiasemaa. Tässä versiossa oleva työkalu on ollut jo aktiivisesti käytössä useita satoja kertoja ja on todettu useamman käyttäjän kautta ketteräksi ja toimivaksi. Kokonaisuudessaan prosessiin työkalun kautta menee n. 10 minuuttia. Jos tähän lisäisi puoli tuntia, saataisiin keskimäärin aika manuaaliselle toteutukselle. Tässä isona huomiona kuitenkin työkalun käyttäjä kuluttaa tämän ajan itse, eikä mikään automaattiinen poistoskripti. Tuhannen prosessin kohdalla säästettäisiin siis jo 500 työtuntia. Työkalun nopeutta voitaisiin kiristää alle viiteen minuuttiin ”threadauksella”, jolla luodaan viereisiä prosesseja skriptille. Tässä ongelmana saattaa syntyä ruuhkaa palvelimella, jos useampi henkilö käyttää työkaluja ja näiden aliprosessit voivat epäonnistua. Tällöin varmuus tietojen pyyhkimisestä laskisi ja tuottaisi ongelmia uuden korvaavan tukiaseman käyttöönotossa.

Jatkokehityksessä yritetään etsiä tapoja jo kyseisen olemassa olevan poistoskriptin nopeuden kehittämiselle. Olennaisempana kehityskohteena on kuitenkin uuteen korvaavaan tukiasemaan liittyvät toimenpiteet. Näihin liittyy tukiaseman tietojen parsiminen annetusta datasta ja niiden sopeuttaminen uuden ja vanhan tukiasematekniikan ympäristöön. Seuraavassa työkalun versiossa on toteutus tehty graafiselle käyttöliittymälle, minkä kukin käyttäjä asentaa virtuaalikoneen päälle. Tämä vaatii enemmän valmisteluja kuin nykyinen versio, mutta sillä saadaan aikaiseksi käyttömukavuutta ja parempaa kuvaa käyttäjälle prosessin tilanteesta. Tämän lisäksi tehokkuus skriptissä kasvaa, kun mahdollisuus Pythonin eri kirjastoihin mahdollistuu ja toteutuksessa voidaan aikaisemman version Expect-skripti jättää kokonaan pois. GUI-toteutus tehdään PySimpleGUI-kirjastolla, joka on yksinkertainen graafisen käyttöliittymän kirjasto Python-kielellä.

Tulevassa työkalussa syötetään vain tukiasemapaikkaa yksilöivä tunnus komentokehotteeseen, jonka avulla se hakee poistuvan tukiasematekniikan tiedot poistotyökalua varten sekä tilalle tulevan tukiaseman parametrit ja tiedot tuoreesta datasta. Tunnuksen syöttämisen jälkeen annetaan mahdollisuus valikkomenetelmällä aloittaa

poistotyökalun toiminta, parametrien ja naapuruusrelaatioiden ajaminen sekä uuden että vanhan tukiasematekniikan verkkolaitteisiin. Myös kohteen tilanne vaihdon suhteen merkitään ylös Exceliin, mistä nähdään missä tilanteessa prosessi etenee. Kaikki toimivat suoraan pelkän yksilöivän tunnuksen perusteella ja käyttäjä näkee esimerkiksi poistotyökalun tulosteen graafisen käyttöliittymään avautuvasta komentokehotteesta.

Lähteet

3G UMTS Network Architecture. N.d. Viitattu 19.9.2020. https://www.electronics-notes.com/articles/connectivity/3g-umts/network-architecture.php

Applications for Python. N.d. Viitattu 27.10.2020. https://www.python.org/about/apps/

Base Station Controller. N.d. Viitattu 16.9.2020. https://www.oliviawireless.com/iot-glossary-dictionary/base-station-controller

DNA yrityksenä. N.d. Viitattu 6.9.2020. https://corporate.dna.fi/yrityksena/dnalyhyesti#historia

Ericsson Antenna System. N.d. Viitattu 27.9.2020. https://www.ericsson.com/en/networks/offerings/5g/5g-radio/antenna-system

Ericsson Radio System / mmWave. N.d. Viitattu 28.9.2020. https://www.ericsson.com/en/portfolio/networks/ericsson-radio-system/radio/mmwave

Ericsson Radio System / Radio. N.d. Viitattu 27.9.2020. https://www.ericsson.com/en/portfolio/networks/ericsson-radio-system/radio

GSM Network Architecture. N.d. Viitattu 19.9.2020. https://www.electronics-notes.com/articles/connectivity/2g-gsm/network-architecture.php

Laadullinen tutkimus. 2015. Viitattu 6.9.2020. https://koppa.jyu.fi/avoimet/hum/menetelmapolkuja/menetelmapolku/tutkimusstrategiat/laadullinen-tutkimus

MAPS™ LTE for X2 Interface Emulator. N.d. Viitattu 28.9.2020. https://gl.com/lte-x2-application-protocol-testing-maps.html

Mobile Switching Centre. N.d. Viitattu 16.9.2020. https://www.oliviawireless.com/iot-glossary-dictionary/mobile-switching-center

Määrällinen tutkimus. 2015. Viitattu 6.9.2020. https://koppa.jyu.fi/avoimet/hum/menetelmapolkuja/menetelmapolku/tutkimusstrategiat/maarallinen-tutkimus

Radio Network Controller. N.d. Viitattu 8.10.2020. https://www.idt.com/us/en/application/communications/radio-network-controller-rnc

Samaoui, S. El Bouabidi, I. Obaidat, M. S. Zarai, F. & Mansouri, W. 2015. Modeling and Simulation of Computer Networks and Systems. Methodologies and Applications 3-32. Viitattu 28.9.2020. https://www.sciencedirect.com/topics/computer-science/management-entity

Soveltavasta tutkimuksesta. N.d. Viitattu 22.9.2020. https://oppimateriaalit.jamk.fi/yamk-kasikirja/soveltavat-tutkimusmenetelmat/

System Architecture Evolution (SAE) and the Evolved Packet Core (EPC). N.d. Viitattu 28.9.2020. https://www.artizanetworks.com/resources/tutorials/sae_tec.html

UMTS UTRA / UTRAN: radio network subsystem RNS. N.d. Viitattu 19.9.2020. https://www.electronics-notes.com/articles/connectivity/3g-umts/radio-access-network-utra-utran.php

What is a Bash Script. 2020. Viitattu 30.10.2020. https://linuxhint.com/bash_script/

What is a mobile telephony application server (mobile TAS) / MMTel-AS. N.d. Viitattu 27.9.2020. https://www.metaswitch.com/knowledge-center/reference/what-is-a-mobile-telephony-application-server-mobile-tas-mmtel-as