Reunalaskenta-arkkitehtuurit

Lauri Vene

Pro gradu -tutkielma
HELSINGIN YLIOPISTO
Tietojenkäsittelytieteen laitos

Helsinki, 15. toukokuuta 2018

HELSINGIN YLIOPISTO — HELSINGFORS UNIVERSITET — UNIVERSITY OF HELSINKI

Tiedekunta — Fakultet — Faculty	Laitos — Institution — Department
Matemaattis-luonnontieteellinen	Tietojenkäsittelytieteen laitos

Tekijä — Författare — Author
Lauri Vene

Työn nimi — Arbetets titel — Title

Reunalaskenta-arkkitehtuurit

Oppiaine — Läroämne — Subject
Tietojenkäsittelytiede

Työn laji — Arbetets art — Level	Aika — Datum — Month and year	Sivumäärä — Sidoantal — Number of pages
Pro gradu -tutkielma	15. toukokuuta 2018	53

Tiivistelmä — Referat — Abstract

Tiivistelmä.

ACM Computing Classification System (CCS):

Avainsanat — Nyckelord — Keywords
Reunalaskenta

Säilytyspaikka — Förvaringsställe — Where deposited

Muita tietoja — Övriga uppgifter — Additional information

Sisältö

1 Johdanto

Reunalaskenta (Multi-access edge computing, MEC) tuo laskentaresursseja lähemmäksi asiakasta. Verkkoyhteyksistä puhuttaessa etäisyys tarkoittaa loogista etäisyyttä, jonka mittarina toimii verkkoyhteyden viive asiakkaan ja palvelun välillä [Satyanarayanan, 2017]. Asiakkaan lähelle tuotavien laskentaresurssien on tarkoitus mahdollistavat palveluiden tuottaminen pienemmällä viiveellä ja vakaammalla verkkoyhteydellä.

Reunalaskennan kohderyhmä muodostuu pääasiassa mobiililaitteista. Tässä yhteydessä mobiililaitteella tarkoitetaan akkuvirralla toimivia ja langattomia verkkoyhteyksiä käyttäviä laitteita, kuten älypuhelimia. Mobiililaskenta (mobile computing) on mobiililaitteilla suoritettavaa ohjelmistojen suoritusta. Mobiililaskenta on luonteeltaan rajoittuneempaa kuin kiinteällä laitteistolla suoritettava laskenta [Ha et al., 2013]. Mobiililaskentaa rajoittaa mobiililaitteen laskentaresurssien määrä, verkkoyhteyden laadun vaihtelu, mobiililaitteen kestävyys ja käytettävissä olevan energian määrä [Satyanarayanan, 1996]. Mobiililaitteiden valmistajat joutuvat tasapainottamaan laskentaresurssien määrää ja virrankulutusta yhdessä laitteen koon kanssa. Lisäksi akkuvirran rajallisuus johtaa kompromisseihin mobiililaitteen suorituskyvyssä [Satyanarayanan, 2001].

Perinteiset verkkopalvelut sijaitsevat keskitetyissä palvelinsaleissa – pilvessä. Koska pilvi tarjoaa näennäisesti äärettömästi laskentaresursseja, se saattaa kuulostaa houkuttelevalta vaihtoehdolta siirtää laskentaa pilveen suoritettavaksi. Asiakkaasta nähden kaukana sijaitseva pilvi sisältää muutamia ongelmia. Asiakkaan ja pilven välinen yhteys sisältää merkittävän määrän viivettä [Satyanarayanan et al., 2009], joten se ei sovellu reaaliaikavaatimuksen sisältävien palveluiden suorittamiseen. WAN-verkossa esiintyvään viiveeseen ei juurikaan enää voida vaikuttaa, vaan ainoana vaihtoehtona on laskentaresurssien tuominen lähemmäksi asiakasta [Satyanarayanan et al., 2009]. Lisäksi tietoliikenteen määrän kasvun seurauksena, kaistanleveys internetin läpi kulkevan tietoliikenteen osalta ei välttämättä kykene täyttämään palveluille asetettuja vaatimuksia. Esimerkki reaaliaikaisesti tehtävä kasvojentunnistus mobiililaitteen välittämästä videovirrasta (video stream) vaatii kaistanleveyttä sekä pientä viivettä.

Myös mobiililaitteiden suorituskykyyn liittyvien kasvavien odotuksien seurauksena, laskennan siirtäminen mobiililaitteilta muualle suoritettavaksi on nostanut reunalaskennan varteenotettavaksi ratkaisuvaihtoehdoksi. Reunalaskenta muun muassa tarjoaa mobiililaitteille mahdollisuuden siirtää resurssi-intensiivistä laskentaa lähellä sijaitsevalle reunalaskennasta vastaavalle yksikölle. Laskennan siirrolla tarkoitetaan etälaskentaa (offload). Etälaskennan motivaationa voi toimia esimerkiksi akkuvirran säästäminen tai suoritusajan lyhentäminen. Tämä tulee kuitenkin laskennan siirtoon kuluvan ajan ja energian kustannuksella. Reunalaskentaa voi hyödyntää myös muihin käyttötarkoituksiin. Se mahdollistaa palveluiden tuottamisen lähempänä

asiakasta. Reunalaskennan tarjoamana palveluna voi olla esimerkiksi reunalla sijaitsevien resurssien hyödyntäminen välimuistina. Toinen esimerkki reunalaskennan mahdollistamasta palvelusta voisi olla jonkin viivekriittisen palvelun, kuten pelipalvelimen, suorittaminen reunalla sijaitsevilla resursseilla.

Tutkielman kirjoitushetkellä reunalaskentaa ei ole yleisesti tarjolla. Reunalaskennan käyttöönottoa rajoittaa kokonaisvaltaisten ratkaisujen puute. Reunalaskennan käyttöönotto edellyttää reunalaskennan tarjoavan reunajärjestelmän kehittämistä ja rakentamista. Reunajärjestelmä koostuu laitteistoja ohjelmistokomponenteista. Reunalaskennan eteen on tehty paljon tutkimustyötä, mutta suuri osa siitä keskittyy matalamman tason yksityiskohtiin ja mekanismeihin. Tällaisia ovat esimerkiksi ohjelmiston jakaminen etänä suoritettaviin osiin, sekä näiden osien siirrettävyyden kannattavuuden päättely suoritusaikana.

Tämä tutkielma käsittelee reunajärjestelmiä reunalaskenta-arkkitehtuurien avulla. Tällaiset arkkitehtuurit kuvaavat reunajärjestelmän toimintaa korkealla tasolla ja antavat yleiskuvan reunajärjestelmän keskeisimmistä ominaisuuksista. Tutkielmaan on valittu reunalaskenta-arkkitehtuurit, joiden tavoitteena on reunajärjestelmän toteuttaminen osaksi mobiiliverkkoja. Nämä arkkitehtuuriehdotukset kuvaavat reunajärjestelmän ja mobiiliverkon mekanismeja, jotka mahdollistavat reunalaskennan tuottamisen mobiiliverkossa toimiville mobiililaitteille. Tutkielman tavoitteena on tunnistaa reunalaskenta-arkkitehtuureissa esiintyvät ominaisuudet. Tunnistettujen ominaisuuksien avulla arkkitehtuuriehdotuksien erojen jäsentäminen on helpompaa.

Tutkielma jäsentyy seuraavasti. Luvussa 2 esitellään reunajärjestelmään liittyvät keskeiset käsitteet. Keskeiset käsitteet sisältävät määritelmän reunajärjestelmän toimintaympäristölle ja reunalaskentaan liittyvää terminologiaa. Lisäksi, koska tutkielman kohteena on nimenomaan mobiiliverkkoihin sijoittuvat reunajärjestelmät, esitellään LTE-tyyppisen mobiiliverkon rakenne ja toiminta yleisellä tasolla. Reunalaskennan toteuttaminen vaatii erilaisten teknologioiden käyttöönottoa. Luvussa 3 käsitellään reuna-arkkitehtuuriehdotuksien yhteydessä ilmenneet teknologiat. Ensimmäisenä käsiteltävänä teknologiana on pilvipalveluidenkin tuottamiseen käytettävät virtuaalikoneet. Virtuaalikoneiden lisäksi käsitellään SDN, jonka tavoitteena on verkkoliikenteen ohjelmallisen ohjaamisen helpottaminen. Viimeisenä käsitellään NFV, joka pyrkii verkkotoiminnallisuuksien virtualisointiin. Luvussa 4 kuvataan reunalaskenta-arkkitehtuureista identifioidut ominaisuudet. Luvussa 5 kuvataan reunajärjestelmien rakennetta sekä arkkitehtuurin vaikutus rakenteeseen. Ehdotetut reunalaskenta-arkkitehtuurit esitellään luvussa 6. Arkkitehtuurien esittelyn jälkeen kuvataan ehdotuksien eroavaisuudet tunnistettujen ominaisuuksien pohjalta. Lisäksi käsitellään tutkielmassa käsiteltyjen arkkitehtuurien yhteensopivuus ETSI MEC spesifikaation kanssa. Tutkielma päättyy luvussa 7 esitettävään yhteenvetoon.

2 Reunalaskennan perusteet

2.1 Motivaatio

Reunalaskennan tavoitteena on tarjota etälaskentaa (offload) ja muita palveluita lyhyillä viiveillä. Etälaskenta mahdollistaa kevyempien ja vähemmän energiaa kuluttavien asiakaslaitteiden valmistamisen, tinkimättä laitteilla tarjottavien palveluiden laadusta. Palveluiden tarjoaminen lyhyillä viiveillä tarjoaa mahdollisuuden reaaliaikaiseen käyttökokemukseen, siitä huolimatta että palvelu on toteutettu asiakaslaitteen ulkopuolella.

Reunajärjestelmiä voi käsitellä monella eri tasolla. Yksi haarautumispiste on konteksti, jossa reunalaskentaa tarjotaan. ETSI MEC spesifikaatio ottaa nykyään huomioon mobiiliverkon lisäksi myös muut verkot, kuten WiFi:n ja kiinteät yhteydet [Taleb et al., 2017]. Tässä tutkielmassa pääpaino on mobiiliverkkoihin suuntautuvissa ratkaisuissa.

Motiivina reunajärjestelmän liittämiseksi osaksi mobiiliverkkoa, toimii mobiiliverkon ja reunalaskennan yhteinen kohderyhmä sekä mobiiliverkon jo olemassa oleva hajautettu infrastruktuuri. Mobiiliverkossa asiakaslaite voi verkon puitteissa liikkua paikasta toiseen ilman, että yhteys mobiiliverkon palveluihin katkeaa missään vaiheessa. Mobiiliverkon palveluiden toimintaa voi siis kutsua saumattomaksi. Täten onkin perusteltua olettaa, että reunalaskentaan pätee yhtäläiset palveluvaatimukset kuin mobiiliverkkoon.

Reunajärjestelmän näkökulmasta mobiiliverkko on uhka ja mahdollisuus. Mobiiliverkko tarjoaa infrastruktuurin, jota reunajärjestelmä voi parhaansa mukaan pyrkiä hyödyntämään. Hyödyntämisellä tarkoitetaan esimerkiksi mobiiliverkossa olevien toimintojen käyttämistä osana reunalaskennan toteuttamista sekä yhteisien tietoliikenneverkkojen käyttämistä. Mobiiliverkon toimintojen muuttaminen on sen hajautetun rakenteen vuoksi työlästä ja kallista. Reunajärjestelmä on itse myös hajautettu järjestelmä, joten mobiiliverkon osittaista päivittämistä ei voida pitää ylitsepääsemättömänä esteenä. Varsinkaan jos reunalaskentaa on tavoitteena tarjota mobiiliverkon yhteydessä. Mikäli reunajärjestelmän toteutettaisiin pääosin erilliseksi järjestelmäksi mobiiliverkon läheisyyteen, tarkoittaisi se että ylläpidettävänä olisi kaksi hajautettua järjestelmää. Kahden erillisen järjestelmän ylläpitäminen aiheuttaisi epäilemättä myöskin kuluja. Täten reunajärjestelmän toteuttaminen olemassa olevan järjestelmän yhteyteen vaatii kompromisseja.

Reuna-arkkitehtuuriehdotuksilla on vastuu reunajärjestelmän toteutuksen rungosta. Huomioitavia tekijöitä reunajärjestelmää toteutettaessa ovat muun muassa reunalaskennan toiminnan jouhevuus, tarpeellisten investointien määrä, järjestelmän joustavuus ja muutoksien tarve olemassa oleviin toimintoihin.

2.2 Keskeiset käsitteet

Tässä kappaleessa esitellään reunajärjestelmään liittyvät keskeiset käsitteet ja toimijat. Koska reunalaskentaa suorittava reunajärjestelmä sijoittuu osaksi olemassa olevaa hierarkiaa, esitellään myös reunan ympärillä sijaitsevat toimijat. Reunan itsensä lisäksi käsitellään siis asiakaslaitteet ja pilvi, niiltä osin kuin ne liittyvät reunajärjestelmän käyttöön tai toimintaan. Koska tässä tutkielmassa käsiteltävien reuna-arkkitehtuurien pyrkimyksenä on mahdollistaa reunalaskenta mobiiliverkkoa hyödyntäen, esitellään LTE-tyyppisen mobiiliverkon rakenne ja keskeisimmät toimijat. Lopuksi määritellään, mitä tässä tutkielmassa tarkoitetaan reuna-arkkitehtuurilla.

2.2.1 Asiakaslaite

Asiakaslaite (user equipment) on yleisnimitys reunan tai pilven palveluita kuluttavalla laitteelle. Asiakaslaite -käsitettä ei ole rajattu mihinkään tiettyihin laitteisiin ja asiakaslaite voikin olla esimerkiksi älypuhelin, älylasit tai verkkoyhteydellä varustettu auto. Asiakaslaitteiden yhdistävänä tekijänä on siis jonkinlainen yhteys reuna- tai pilvipalveluihin. Yleisesti asiakaslaitteen käyttämä yhteys on tyypiltään langaton. Yksinkertaisuuden vuoksi tässä tutkielmassa asiakaslaitteen voi ajatella älypuhelimena, jollei toisin mainita.

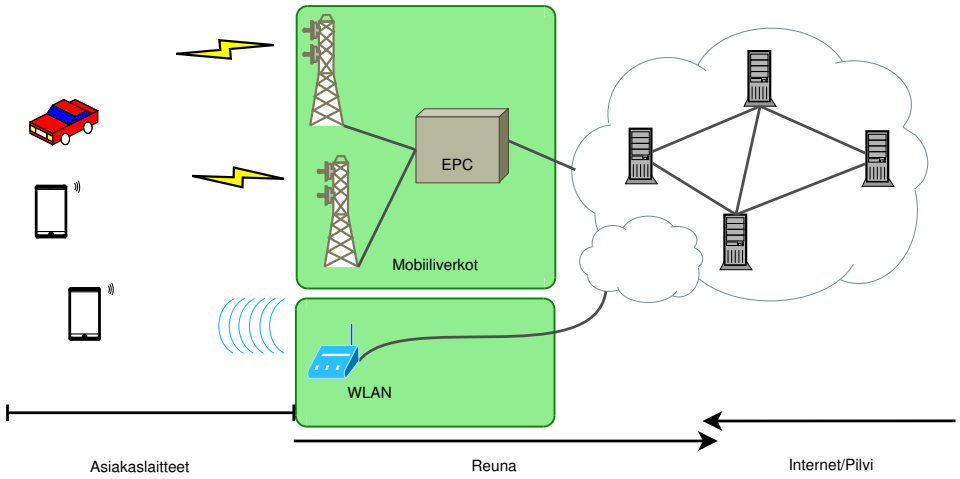
Verkkohierarkian näkökulmasta asiakaslaitteet ovat lehtisolmuja. Tämä tarkoittaa että asiakaslaitteet toimivat ainoastaan palveluiden kuluttajina eivätkä siis tarjoa itse palveluita. Myöskään kulutettavien palveluiden tyypillä ei ole juurikaan merkitystä, kun asiaa käsitellään infrastruktuuri tai arkkitehtuuritasolla.

Konkreettinen esimerkki asiakaslaitteesta, joka hyödyntää reunapalvelua, voisi olla jokin ajoneuvo. Ajoneuvolla on mobiiliyhteys reunapalveluun jonka tarkoitus on välittää tietoa liikenteestä muille ajoneuvoille. Esimerkiksi tilanteessa jossa edellä on ruuhkaa, voitaisiin muille lähistöllä oleville ajoneuvoille välittää tieto tästä, jolloin voidaan valita jokin toinen reitti määränpäähän.

2.2.2 Reuna

Reuna koostuu useista toiminnallisista entiteeteistä, jotka voidaan jakaa sekä fyysisiin että loogisiin kokonaisuuksiin. Tässä kappaleessa lähdetään liikkeelle määrittelemällä reuna-alue, joka edustaa reunan toimialuetta sekä reunaa yleisenä käsitteenä. Tämän jälkeen määritellään reunasolmu, joka on reunajärjestelmän keskeisin fyysinen rakennuspalanen. Lopuksi esitellään pääasiassa ohjelmallisella tasolla ilmenevät toimijat reuna-alusta ja reunasovellus.

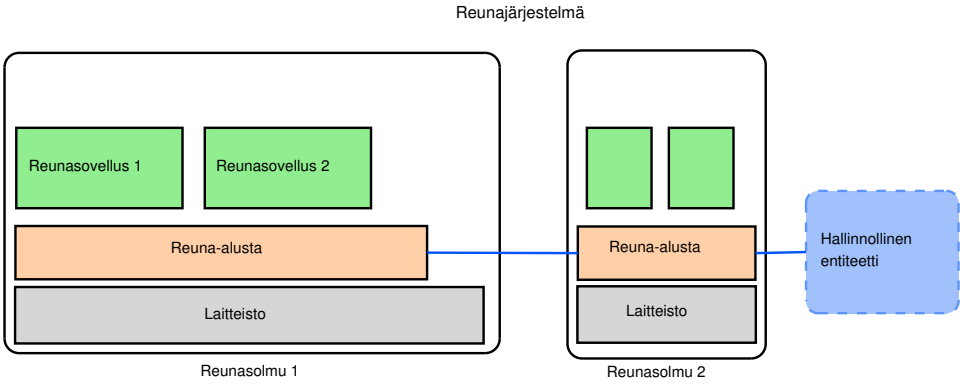
Reuna-alue Reunalla viitataan alueeseen, joka alkaa asiakaslaitteen yhteyspisteestä. Tämä yhteyspiste voi olla esimerkiksi WLAN-tukiasema tai mobiiliverkon tukiasema. Tästä ensimmäisestä yhteyspisteestä reuna-alue



Kuva 1: Asiakaslaitteiden, reunan ja pilven alueet

laajenee kohti runkoverkkoa ja pilveä. Kuvassa 1 reuna on kuvattu alkavan mobiiliverkon tukiasemasta tai WLAN-tukiasemasta. Joidenkin näkemyksien mukaan myös asiakaslaitteet luetaan osaksi reunaa, tällöin niihin viitataan termillä reunalaite (edge device) [Garcia Lopez et al., 2015]. Tämän tutkielman puitteissa asiakaslaitteet luetaan omaksi osakseen reuna-alueen toisella laidalla. Reunan laajuudelle ei siis ole yksiselitteistä määritelmää, vaan termiä usein sovelletaan käytettävän kontekstin mukaan siten, että reuna-alue rajautuu kontekstissa esiintyvien toimijoiden mukaan edellä mainitulle välille. Esimerkiksi mobiiliverkon tukiasemien yhteyteen rakennettua reunajärjestelmää käsiteltäessä, reunalla tarkoitetaan ainoastaan reunajärjestelmän asiakaslaitteita palvelevien osien muodostamaa vyöhykettä. Kuten jo aiemmin mainittu, reunan keskeisenä etuna muihin palveluihin voidaan pitää reunapalveluiden ja asiakaslaitteen välisen viiveen vähäisyyttä. Yleisenä nyrkkisääntönä reunalle voidaankin pitää verkkoyhteyksien viivettä suhteessa muuhun internettiin, koska reuna-alueella palveluiden viiveiden tulisi siis olla muuta internettiä nopeampia. Toisin sanoen reunan palveluiden tulisi sijaita lähempänä kuin pilvessä sijaitsevien palveluiden.

Reunasolmu Tämän tutkielman kontekstissa reunasolmulla (mobile edge host) tarkoitetaan yksittäistä reunalaskentaa tarjoavaa entiteettiä[ETSI, 2016b]. Reunasolmu -termillä viitataan reunasolmun fyysiseen laitteiston ja reunasolmun toiminnalliseen kokonaisuuteen. Reunasolmu sisältää reunasovelluksien suorittamiseen tarvittavat resurssit, jotka ovat pääasiassa tavallisia palvelinresursseja kuten laskenta, tallennustila ja verkkoyhteydet. Reunasolmu voi koostua esimerkiksi mobiilitukiaseman ja palvelinlaitteiston muodostamasta kokonaisuudesta. Reunasolmun toiminnallinen kokonaisuus sisältää erinäisiä toimintoja, joiden tehtävänä on mahdollistaa reunasovelluksien



Kuva 2: Esimerkki reunajärjestelmän rakenteesta.

suorittaminen. Tällaisia ovat muun muassa tietoliikenteen reitittäminen ja virtualisointi-alustan tarjoaminen, sekä sen hallinnointi. Kuvassa 2 on esitetty esimerkki reunasolmun sisällä olevista entiteeteistä. Seuraavaksi käsiteltävä reuna-alusta kattaa osan reunasolmun toiminnoista.

Kuten aiemmin mainittu, reunajärjestelmä koostuu joukosta reunasolmuja, jotka ovat maantieteellisesti hajautettuja. Reunasolmut siis eroavat toisistaan vähintään sijainnin perusteella, mutta voivat erota myös käytettävissä olevien resurssien osalta. Resurssien osalta reunasolmu voi olla mitä tahansa vähäisillä laskenta ja tallennus resursseilla varustetun WiFi-tukiaseman ja kokonaisen palvelinklusterin väliltä. Reunasolmun sijaintiin vaikuttaa käytössä oleva reuna-arkkitehtuuri. Reunan rakennetta ja reunasolmujen sijaintia käsitellään tarkemmin luvussa 5.

Reuna-alusta Reuna-alusta (edge platform) on ohjelmistotason toimija. Se tarjoaa rajapinnan reunasovelluksien suorittamista varten [ETSI, 2016b]. Toisin sanoen se siis tarjoaa reunasovelluksille toimintaympäristön. Reuna-alustan tehtävät eivät rajaudu ainoastaan yksittäiseen reunasolmuun, vaan sen lisäksi se hoitaa hallinnollisia tehtäviä kuten tietoliikenteen ohjausta. Lisäksi reuna-alustan tehtäviin voidaan lukea reunasovelluksia suorittaviin virtuaalikoneisiin liittyvät hallinnolliset toimet. Esimerkkinä reuna-alustan tehtävistä on kappaleessa 4.1 esiteltävä virtuaalikoneiden live migraatio reunasolmulta toiselle. Reunasovelluksien lisäksi reuna-alusta voi itsessään tarjota palveluita, jotka eivät suoranaisesti ole reunasovelluksia vaan esimerkiksi nopea kommunikaatioväylä laitteelta-laitteelle (machine-to-machine), jota voi käyttää esimerkiksi ajoneuvojen väliseen kommunikointiin. Koska reuna-alustan tehtäviin kuuluu hallinnolliset toimet, voidaan sen ajatella laajentuvan myös reunasolmun ulkopuolella sijaitseviin hallinnosta vastaaviin entiteetteihin.

Reunasovellus Reunasovelluksella (edge application) tarkoitetaan yksittäistä reunasolmulla suoritettava ohjelmistoa [ETSI, 2016b], jonka kuluttajana voi toimia asiakaslaite tai toinen reunasovellus. Reunasovellukset tarjoavat *reunapalveluita*. Reunasovellus ei ota kantaa millaista palvelua sillä tuotetaan.

Reunasovelluksen tuottama reunapalvelun tyyppiä säätelee pääasiassa reuna-alusta sekä reunalaskennan liiketoimintamalli. Teoriassa mahdollisuudet ovat siis samat kuin pilvessä. Esimerkkejä reunasovelluksen tuottaman palvelun tyypeistä ovat yhteiskäyttöinen tai käyttäjäkohtainen. Esimerkki käyttäjäkohtaisesta reunasovelluksesta voisi olla käyttäjäkohtainen virtuaalikone johon käyttäjä voi ottaa etäyhteyden. Esimerkkinä yhteiskäyttöisestä reunasovelluksesta voisi olla pelipalvelin, jota voivat käyttää reunasolmun toimipiirissä olevat pelaajat.

Reunalaskenta ja reunasovellus viittaavat yleensä lähes samaan asiaan, mutta eri konteksteissa. Reunasovellus on ohjelmallinen entiteetti, joka sijaitsee reunajärjestelmän sisällä. Reunalaskenta puolestaan viittaa reunajärjestelmän palveluiden käyttämiseen.

Reunalaskennan tyypit Reunalaskennan tyyppejä ovat etälaskenta (offloading) ja asiakaslaitetta tukevat erilaiset reunapalvelut. Arkkitehtuuritasolla reunalaskennan tyyppiin ei oteta juurikaan kantaa. Jotkin suunnittelupäätökset voivat kuitenkin vaikuttaa siihen millaisia reunalaskennan tyyppejä reunajärjestelmä lopulta tukee tai painottaa. Etälaskennalla tarkoitetaan työyksiköiden (task) siirtämistä ulkoiselle laskentaa suorittavalle instanssille, joka palauttaa asiakaslaitteelle laskennan lopputuloksen. Reunajärjestelmässä suoritettava etälaskenta voidaan jakaa kahteen alakategoriaan: kokonaan reunalla suoritettaviin ja osittain reunalla suoritettaviin sovelluksiin[Mach and Becvar, 2017]. Kokonaan reunalla suoritettava laskenta on tyypiltään sellaista, että sitä ei voi jakaa useamman instanssin suoritettavaksi. Tällöin vaihtoehdoksi jää suorittaa laskenta kokonaan paikallisesti tai siirtää se reunasolmulle suoritettavaksi. Osittainen etälaskenta on vaihtoehto sovelluksille, jotka on mahdollista pilkkoa suoritettaviin työyksikköihin. Sovelluksen työyksiköt jakautuvat usein niihin jotka on mahdollista siirtää reunalle suoritettavaksi, sekä niihin joita ei ole mahdollista siirtää. Esimerkiksi valokuvan ottaminen on työyksikkö, jota ei ole mahdollista tehdä muuten kuin paikallisesti asiakaslaitteella [Mach and Becvar, 2017]. Päätös etälaskennan tekemistä on riippuvainen laskennan tavoitteesta. Tavoitteina voi olla esimerkiksi asiakaslaitteen virransäästön maksimointia tai laskennan suorituksen keston minimointia. Reunapalvelu voi olla tyypiltään asiakaslaitteella suoritettavan sovelluksen toimintaa tukeva palvelu. Sen voi ajatella esimerkiksi pelipalvelimeksi, joka suorittaa pelimekaniikan simuloinnin ja asiakaslaitteelle jää tehtäväksi ainoastaan pelitilan visuaalinen esittäminen.

2.2.3 Pilvi

Armbrust et al. esittävät artikkelissaan määritelmän pilvelle [Armbrust et al., 2010]. Pilvellä viitataan sekä pilvessä tuotettaviin palveluihin, että itse järjestelmään. Järjestelmällä tarkoitetaan pilvi-alustaa, joka mahdollistaa palveluiden tuottamisen. Artikkelin mukaan pilven tuottamisen yhtenä keskeisimpänä mahdollistajana on toiminut edullisille sijainneille rakennetut suuret palvelinsalit. Toisin sanottuna pilvipalvelut tuotetaan palvelinsaleissa, jotka ovat keskitettyjä ja sijaitsevat kauempana asutuksesta.

Keskeinen osa pilvi paradigmaa on myös tapa jolla palveluita tuotetaan. Pilvialusta tarjoaa asiakkailleen mahdollisuuden skaalata palveluitaan näennäisesti äärettömyyksiin. Tämän lisäksi aika, jossa käytössä olevien resurssien lisäys tai poisto voidaan tehdä, on huomattavasti lyhyempi verrattuna perinteiseen palvelinsaliin, jossa palvelun käyttäjä omistaisi itse palvelun tuottamiseen käytettävän laitteiston. Tässä tutkielmassa pilvellä viitataan kuitenkin enemmänkin järjestelmän rakenteeseen ja sijaintiin, kuin palveluiden tuottamiseen käytettävään malliin.

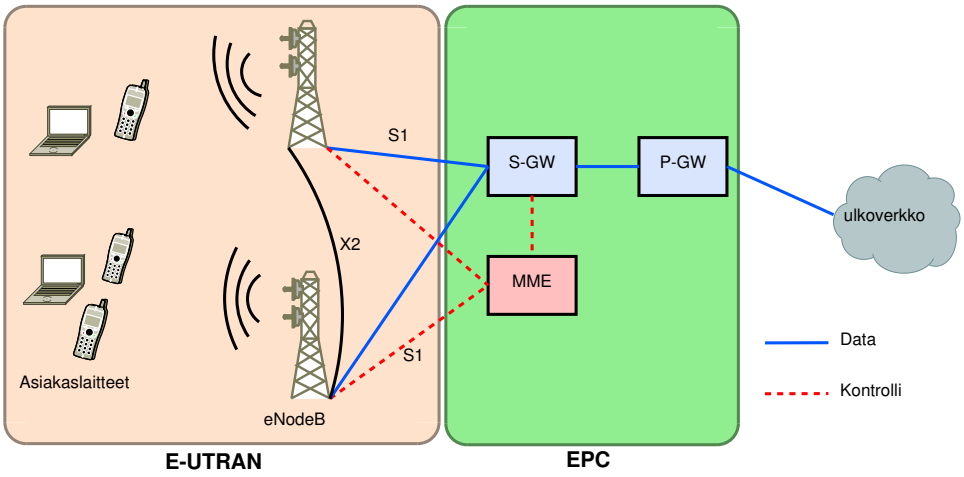
Reunan ei ole tarkoitus korvata pilveä, vaan täydentää sitä. Myöskään pilven ja reunan välinen raja ei ole tarkka. Tässä tutkielmassa pilvellä tarkoitetaan suhteessa reunaa kauempana sijaitsevia palveluita. Pilvi ulottuu verkon ytimestä kohti asiakaslaitetta ja saattaa olla osittain päällekkäin reunan kanssa. Kuten aiemmin mainittu, reuna lähestyy verkon ydintä asiakaslaitteen suunnasta.

Pilven ja reunan väliin on ehdotettu vielä yhtä toteutusparadigmaa – sumua (fog computing). Sumun tavoitteet ovat samankaltaisia kuin reunan. Sumun tavoitteita ovat muun muassa pienet viiveet, maantieteellisesti hajautettu rakenne ja langattomien laitteiden palvelu [Bonomi et al., 2012]. Sumun on tarkoitus olla hieman kuten pilvi, mutta hajautetumpana ja pilveen verrattuna huomattavasti lähempänä käyttäjää. Reunaan verrattuna sumu on hieman kuin reuna, joka on viety kauemmaksi verkon reunasta. Sumua ei käsitellä tämän enempää tässä tutkielmassa, vaan keskitytään lähempänä reunaa sijaitsevaan laskentaan.

2.2.4 Mobiiliverkko

Tässä tutkielmassa käsiteltävät reuna-arkkitehtuurit sisältävät yhdistävänä tekijänä tavoitteen toimia mobiiliverkon yhteydessä. Täten reuna-arkkitehtuurien suunnittelupäätöksiä tarkasteltaessa on tarpeen ymmärtää mobiiliverkon osat yleisellä tasolla. Yksinkertaisuuden vuoksi tämän tutkielman puitteissa mobiiliverkoksi oletetaan LTE:n mukainen arkkitehtuuri, joka koostuu E-UTRAN tyyppisestä radioverkosta ja EPC tyyppisestä runkoverkosta. Seuraavaksi käydään läpi mobiiliverkon arkkitehtuurin tämän tutkielman kannalta merkitykselliset toimijat ja toiminnot.

Korkealla tasolla tarkasteltuna mobiiliverkko koostuu kahdesta osiosta:



Kuva 3: Yksinkertaistettu LTE-tyyppisen mobiiliverkon rakenne.

radioverkosta ja runkoverkosta. 3GPP kehittämässä LTE standardissa radioverkon sisältävä osuus on nimeltään E-UTRAN (Evolved UMTS Terrestrial Radio Access Network) ja runkoverkon osuus on nimeltään EPC (Evolved Packet Core). E-UTRAN ja EPC väliset yhteydet on kuvattu kuvassa 3.

E-UTRAN tehtävänä on toimia rajapintana asiakaslaitteen ja EPC:n välillä. E-UTRAN sisältää verkon puolella pääasiallisena toimijana eNodeB (Evolved nodeB) tyyppisiä tukiasemia [ETSI, 2017a]. Tukiasemista on olemassa muutamia erilaisia variaatioita, mutta tässä tutkielmassa käsitellään ainoastaan perustapausta. Tukiasema on asiakaslaitetta lähimpänä sijaitseva funktionaalinen verkon osa ja sen seurauksena se on houkutteleva kohde reunalaskennan ratkaisuille. Tietoliikenteen näkökulmasta tukiaseman voi ajatella *reunan* viimeisenä etappina ennen asiakaslaitteita.

ENodeB tarjoaa asiakaslaitteiden suuntaan radioyhteyden. EPC:n suuntaan eNodeB:t ovat yhteydessä S1 rajapinnan avulla. Lisäksi eNodeB:t voivat olla toisiinsa yhteydessä X2 rajapinnan kautta. S1:stä käytetään eNodeB:n ja EPC:n väliseen kommunikointiin. Tämä sisältää sekä hallinnollisen viestinnän, että asiakkaan tietoliikenteen kuljettamisen. X2 rajapintaa puolestaan käytetään tukiasemien väliseen kommunikointiin. ENodeB välisten X2 yhteyksien tavoitteena on nopeuttaa tukiasemien välistä kommunikaatiota, esimerkiksi handoverin yhteydessä tehtävää asiakaskontekstin siirtoa varten [Nohrborg, 2017]. Handoverilla tarkoitetaan asiakaslaitteen radioyhteyden siirtoa toiselle tukiasemalle. Handover käsitellään tarkemmin kappaleessa 4.1.

Mobiiliverkon runkona toimiva EPC koostuu useista loogisista komponenteista. Tämä siis tarkoittaa että toiminnallisuudet voivat fyysisesti sijaita samassa laitteessa. Tässä tutkielmassa on tarpeen ymmärtää perusteet seuraavista alikomponentista: MME (Mobility Management Entity), S-GW

(Serving Gateway) ja PDN GW (P-GW, Packet data network gateway) [ETSI, 2016a].

- **MME** on EPC:n hallinnollinen entiteetti joka vastaa muun muassa asiakaslaitteen tunnistamisesta ja handoveriin liittyvistä toimista EPC:n sisällä. Toisin kuin S-GW ja P-GW, MME ei käsittele asiakaslaitteiden tietoliikennettä.
- **S-GW** eli palveluyhdyskäytävä toimii asiakaslaitteen EPC:n sisäisenä kiintopisteenä. S-GW reitittää asiakaslaitteen liikennettä P-GW:n ja E-UTRAN välillä.
- **P-GW** eli pakettiverkon yhdyskäytävän tehtävänä on toimia asiakaslaitteen ja mobiiliverkon ulkopuolisten IP-verkkojen yhteyspisteenä.

[Firmin, 2017]

Mobiiliverkossa käytävä kommunikaatio voidaan jakaa kahteen kerrokseen: kontrollikerrokseen ja tietoliikennekerrokseen. Kontrollikerroksella välitettävät viestit ovat tarkoitettu hallinnollisiin toimintoihin mobiiliverkon sisällä. Tietoliikennekerros välittää asiakkaan tietoliikennettä internetin ja asiakaslaitteen välillä. Asiakkaan tietoliikenne kulkee tukiaseman ja P-GW:n välillä GTP tunneloituna (GPRS Tunnelling Protocol) [Puente et al., 2015]. Tämä tarkoittaa että mobiiliverkon sisällä tietoliikennettä ei ohjata asiakaslaitteen tietoliikenteen tunnisteiden pohjalta.

2.3 Reuna-arkkitehtuuri

Tässä tutkielmassa käsiteltävät arkkitehtuurit koostuvat oletusarkkitehtuurin (framework) tai suunnittelumallin (design model) tyylisistä arkkitehtuurikuvauksista. Oletusarkkitehtuuri jakaa järjestelmän jonkin tehtävän toteuttaviin kokonaisuuksiin, kuten komponentteihin ja moduuleihin [Tuovinen, 2017b]. Suunnittelumallit puolestaan kuvaavat tarkemmin järjestelmän komponentteja sekä niiden toimintoja. Suunnittelumallit ovat kuitenkin ainoastaan osajoukko suunnittelupäätöksistä [Tuovinen, 2017a], joten ne jättävät osan päätöksistä toteuttajalle. Lisäksi suunnittelumallit kuvaavat järjestelmää ainoastaan jollain tietyllä abstraktiotasolla ja koko järjestelmän kattavaan kuvaamiseen tarvittaisiin useita sisäkkäisiä ja hierarkkisia suunnittelumalleja. Tässä tutkielmassa käsiteltävät reuna-arkkitehtuurit ovat korkean tason luonnehdintoja järjestelmän ja sen osien toiminnasta, jossain tietyssä ympäristössä. Myös ETSI:n MEC referenssiarkkitehtuuri [ETSI, 2016b] sisältää vaatimusmaisesti esitetyt tarvittavat toimet, mutta ei kuvaa niiden toteutusta. Toisin sanoen tutkielmassa käsiteltävät arkkitehtuurit kattavat vain osajoukon kaikista järjestelmän toteuttamiseksi tarvittavista suunnittelupäätöksistä.

Reuna-arkkitehtuurissa toiminnalliset osat on jaettu loogisiin kokonaisuuksiin, eli tehtävien mukaan kasattuihin komponentteihin. Reuna-arkkitehtuurit

asettavat rajoitteita myös toimintojen fyysiselle sijoittumiselle, jolloin osa toimijoista on jaoteltu fyysisiin komponentteihin. Joskin kappaleessa 3.3 esiteltävä NFV tekee toimintojen fyysisestä sijoittumisesta dynaamisempaa ja vähemmän laitteistoon sidonnaista. Kappaleessa 2.2.2 käsiteltyjen reunan keskeisten käsitteiden mukaan tässä tutkielmassa reuna-arkkitehtuurit keskittyvät reunasolmun ja reuna-alustan toiminnalliseen määrittelyyn. Tämän lisäksi reunajärjestelmää käsitellään alijärjestelmänä, eli osana suurempaa kokonaisjärjestelmää, joka koostuu muun muassa mobiiliverkosta. Kuten jo aiemmin mainittiin reuna-arkkitehtuurit eivät ota kantaa reunasovelluksien toimintaan

3 Mahdollistavat teknologiat

Seuraavaksi käsitellään reunalaskennan mahdollistavat teknologiat virtualisointi SDN ja NFV. Tähän esiteltäväksi valittujen teknologioiden joukko perustuu tämän tutkielman luvussa 6 esitettyjen arkkitehtuuriratkaisujen yhteydessä esiintyneisiin teknologioihin. Edellä mainittujen teknologioiden lisäksi, reunalaskentaa käsittelevissä julkaisuissa on esitetty myös muita reunalaskentaa mahdollistavia teknologioita, kuten verkon viipalointi (network slicing) [Taleb et al., 2017], jonka käsittely sivuutetaan tämän tutkielman yhteydessä.

3.1 Virtualisointi

Virtualisoinnilla tarkoitetaan tietokoneohjelmiston suorittamista ohjelmallisesti toteutetun rajapinnan päällä. Toisin sanoen virtualisointi erottaa suoritettavan ohjelmiston suorittavasta laitteistosta. Usein virtualisoitava ohjelmisto on kokonainen käyttöjärjestelmä, jolloin suoritettavaa instanssia kutsutaan virtuaalikoneeksi. Virtualisointi mahdollistaa laitteiston resurssien jakamisen useamman toisistaan erillään olevan virtuaalikoneen kesken. Juurikin resurssien jakamisen mahdollisuus yhdessä palvelinlaitteiston suorituskyvyn kasvun kanssa on johtanut virtuaalikoneiden käytön yleistymiseen palvelinsaliympäristössä.

Virtuaalikoneen ja laitteiston välillä toimivaa ohjelmistoa kutsutaan hypervisoriksi. Hypervisorin pääasiallinen tehtävänä on tarjota rautatason resursseja virtuaalikoneiden käytettäväksi. Yksi hypervisorin tehtävistä on virtuaalikoneiden luominen. Sen avulla voidaan määrittä virtuaalikoneen käytössä olevat resurssit, esimerkiksi virtuaalikoneen käytettävissä olevan muistin tai prosessorien määrä. Hypervisorilla on myös monia muita toiminnallisuuksia, joita ei tässä tutkielmassa käydä läpi tämän enempää.

Virtuaalikoneet ovat vahvasti esillä reunalaskenta-arkkitehtuurien yhteydessä. ETSI:n reunalaskennan referenssiarkkitehtuurissa [ETSI, 2016b] virtuaalikoneet esitetään reunapalvelun tuottamisen välineenä ja Taleb et

al. esittävät kirjallisuuskatsauksessaan virtuaalikoneet yhdeksi reunalaskennan mahdollistavista teknologioista [Taleb et al., 2017]. Virtuaalikoneiden dynaamisuus verrattuna tavallisesti suoritettavaan ohjelmistoon on reunalaskennan näkökulmasta haluttu ominaisuus. Virtuaalikoneet tarjoavat myös helpon tavan jakaa reunasolmun resursseja usean toisistaan erillisen palvelun välillä. Toisena etuna on virtuaalikoneiden vahva eristys (isolation) muusta järjestelmästä ja muista virtuaalikoneista.

Virtuaalikoneet eivät aseta rajoitteita tarjottavan palvelun tyypille ja niiden avulla onkin mahdollista tarjota monia erilaisia palvelumalleja. Yksi ehdotetuista palvelumalleista on käyttäjäkohtaisen virtuaalikoneet [Satyanarayanan et al., 2009, Wang et al., 2015].

Käyttöjärjestelmän virtualisointi sisältää merkittävästi yleisrasitetta (overhead) [Xavier et al., 2013]. Etenkin tilanteissa joissa varsinainen reunasovellus on kevyt, voi kokonaisen käyttöjärjestelmän virtualisointi olla ylimitoitettua. Vaihtoehtoinen ratkaisu virtuaalikoneille ohjelmiston säiliöinti (container) [Soltesz et al., 2007]. Säiliöinnillä tarkoitetaan käyttöjärjestelmätason virtualisointia, jossa käyttöjärjestelmä kutsut ja ABI-kerroksen (Application Binary Interface) kutsut ovat virtualisoituja [Soltesz et al., 2007]. Säiliöinnin pääasiallisena etuna on pienempi yleisrasite verrattuna virtuaalikoneisiin, mutta se tulee heikomman eristyksen kustannuksella [Soltesz et al., 2007]. Tämän tutkielman puitteissa reunasovelluksien toteutusmekanismiksi oletetaan virtuaalikoneet.

Tulevissa kappaleissa perehdytään tarkemmin virtuaalikoneiden hyödyntämistä osana reunapalveluita. Kappaleessa 4.1 käsitellään virtuaalikoneiden siirtelyä suorituslaitteistolta toiselle, ilman että itse järjestelmän suorittamista tarvitsee keskeyttää siirron ajaksi. Kappaleessa 6.1 käsitellään cloudletiksi nimettyä virtuaalikoneisiin pohjautuvaa reunapalvelun tuottamisjärjestelmää.

3.2 Software-defined networking

Perinteinen tietoliikenneverkko koostuu joukosta erikoistuneista verkkolaitteita. Tällaisia laitteita ovat esimerkiksi kytkin, palomuuri ja reititin. Näiden laitteiden joukko muodostaa hajautetun rakenteen, jossa jokainen laite täytyy konfiguroida erikseen. Laitevalmistajat tarjoavat hallinnointityökaluja, joiden avulla valmistajan omia laitteita on mahdollista konfiguroida suljettua rajapintaa käyttäen. Verkkoinfrastruktuurin hallintaa kuitenkin vaikeuttaa eri laitteiden konfiguraatioiden erilaisuus, sekä puute kokonaisvaltaiselle ohjelmalliselle konfiguroitavuudelle. Tästä johtuen verkkoon tehtävät muutokset ovat työläitä ja riskialttiita [Kreutz et al., 2015].

Perinteisessä tietoliikennelaitteistossa tiedonvälityskerros ja kontrollikerros ovat tiukasti toisistaan riippuvaisia. Tiedonvälityskerroksella tarkoitetaan itse tietoliikennepakettien välittämiseen käytettävää kerrosta, eli tietoliikennettä ohjaavia laitteistoja. Kontrollikerroksella tarkoitetaan tietoliikenteen

reitittämiseksi käytettävää logiikkaa, kuten esimerkiksi reititystauluja. Kontrollikerros on siis tällä hetkellä hajautettuna laitteiston mukana ympäri verkkoa. Tästä johtuen verkko on usein hyvin staattinen ja muutokset kankeita.

SDN (Software-defined networking) eli ohjelmallisesti määritetty verkko on yleistyvä paradigma verkkoympäristöissä. SDN on ratkaisu, jossa tiedonvälitys- ja kontrollikerros on erotettu toisistaan. SDN:ssä ei ole erikoistunutta verkkolaitteistoa vaan nykyinenen tiedonvälitykseen käytetty laitteisto korvattaisiin yleisillä reitittävillä laitteilla[1]. Kontrollikerroksen toiminnasta vastaa SDN Controller. Se on loogisesti keskitetty entiteetti, joka vastaa näiden reitittävien laitteiden ohjaamisesta.

SDN Controllerin ja reitittävän laitteiston välille oletetaan hyvin määritelty rajapinta, jonka kautta reitittäviä laitteita voidaan hallita [Kreutz et al., 2015]. SDN siis toteuttaa *Separation of Concerns* -periaatetta jakamalla verkon reititysmäärittelyjen konfiguroinnin ja itse laitteistopohjaisen toteutuksen omiin osiinsa. SDN Controller tarjoaa rajapinnan ylöspäin ohjelmalliselle verkkokonfiguroinnille ja hoitaa sääntöjen tulkkaamisen alaspäin. OpenFlow on yksi tunnetuimmista SDN Controllerin ja verkkolaitteiden välisestä protokollasta[2].

Tarve SDN pohjaisille ratkaisuille kumpuaa jo aiemmin mainitusta konfiguraation työläydestä ja dynaamisuuden tarpeesta. Esimerkiksi virtuaalikoneiden käytön yleistyessä tarve verkon ohjelmalliselle hallittavuudelle on kasvanut [Jammal et al., 2014]. Virtuaalikoneiden sijainti verkossa ei välttämättä ole kiinteä, vaan virtuaalikone saattaa siirtyä esimerkiksi migratoinnin seurauksena. Virtuaalikoneita saattaa tulla ja poistua verkosta. Perinteisessä verkkoympäristössä esimerkiksi MAC osoitetaulujen päivittäminen saattaa aiheuttaa yhteyskatkoksia palvelimeen [Jammal et al., 2014]. SDN pohjaisia ratkaisuja on jo olemassa, mutta niiden käyttö ei vielä ole korvannut perinteisiä verkkolaitteita.

3.3 Network Function Virtualization

NFV (Network function virtualization) lähtee liikkeelle ongelmasta, jossa nykyisen verkkolaitteiston käyttöikä on lyhyttä ja toiminnot ovat hajautettuina useisiin suljettuihin (proprietary) laitteisiin [ETSI, 2012].

NFV:n tarkoituksena on eriyttää verkkolaitteisto ja verkkotoiminnot. Tämä toteutuisi siten, että erillistä verkkolaitteistoa ei enää tarvittaisi ja nykyisten verkkolaitteiden toiminnallisuudet toteutettaisiin tavallisella palvelinlaitteistolla ohjelmatasolla. Periaate on hyvin samankaltainen kuin perinteisten palvelimien siirtyminen virtuaalikoneita hyödyntävään ympäris-

[1] Reitittämisellä tarkoitetaan tässä yhteydessä pakettien ohjausta ja välitystä yleisessä mielessä

[2] `https://www.opennetworking.org/software-defined-standards/specifications/`

töön. Virtualisoidulla verkkotoiminnallisuudella (jossain yhteyksissä VNF, Virtualized network function) tarkoitetaan ohjelmallisesti toteutettua verkkotoimintoa. Tämä mahdollistaa verkkotoiminnallisuuksien suorittamisen tavallisella palvelinlaitteistoilla hyprvisorin päällä.

Verkkotoimintojen toteuttaminen virtuaalisina, mahdollistaisi useamman toiminnon sijoittamisen samaan laitteistoon. Tämä ainakin teoriassa mahdollistaisi myös paremman skaalautuvuuden. Myös verkkotoiminnallisuuksien käyttöönotto helpottuu mikäli erillisen laitteiston asennusta ei tarvita. Virtuaalisten verkkotoimintojen avulla on myös mahdollista säästää kaappitilaa, sekä pienentää sähkönkulutusta [Nguyen and Cheriet, 2016].

NFV on soveltuva mille tahansa data-tason prosessoinnille ja kontrollitason toiminnoille [ETSI, 2012]. NFV siis soveltuu toteutustavaksi monille erilaisille verkkoitoiminnoille mobiiliverkoissa ja perinteisissä tietoliikenneverkoissa. Esimerkkeinä käyttökohteista ETSI:n NFV whitepaperissä on esitetty muunmuassa reitittimet, palomuurit, kuormantasaajat, eNodeB:t ja MME:t. ETSI on pohtinut NFV:n laajamittaisen käyttöönoton mahdollisuuksia 5G-verkkoingrastruktuurissa ja mainitsee reunalaskennan yhtenä sen käyttötapauksena [ETSI, 2017b].

Yksi NFV:hen pohjautuvista ideoista on C-RAN (Cloud radio acceess network), joka pyrkii virtualisoimaan tukiaseman toinnot. Käytännössä tukiaseman fyysiseen sijaintiin vietäisiin kuituyhteys ja RRU (Remote radio unit). RRU sisältäisi radiosignalointiin tarvittavan laitteiston sekä laitteiston joka muuttaa radion ja kuituyhteyden välillä olevaa tietoliikennettä analogisen ja digitaalisen muodon välillä. Kuituyhteys välittää signaalin ohjelmallisesti toteutetulle BBU:lle (Baseband Unit), joka hoitaa kaikki tukiaseman toiminnot[Chih-Lin et al., 2014]. BBU:t voitaisiin virtualisoida ja suorittaa keskitetysti palvelinsaleissa. Toistaiseksi tukiaseman resurssit on mitoitettu suurimman mahdollisen kuorman mukaan. Virtualisoinnin avulla BBU:n resurssit olisivat paremmin skaalattavissa. Kappaleessa 6.5 esitellään reuna-arkkitehtuuri, joka rakentuu C-RAN tyyliseen verkkoon.

Mobiiliverkossa toimivan reunalaskennan näkökulmasta NFV:n potentiaali on ilmeinen. Jos suurin osa mobiiliverkon toiminnoista voitaisiin virtualisoida ja siirtää tavalliselle palvelinlaitteistolle, voisi reunalaskentaa suorittaa samalla laitteistolla, eikä se vaatisi erillistä laitteistoa omille toiminnoilleen. NFV:n haasteena on toteutuksien puute.

4 Ominaisuudet

Tässä luvussa käsitellään reunalaskentaan liittyviä ominaisuuksia. Ominaisuudet ovat tyypiltään korkean tason suunnittelupäätöksiä. Jokainen ominaisuus edustaa jotain reunalaskennan mahdollistavaa toiminnallisuutta tai toimijaa. Reuna-arkkitehtuureita analysoimalla voitiin erottaa neljä ominaisuutta. Live migraatio kuvaa reunajärjestelmän keinoja siirrellä reunasovelluksia reu-

nasolmujen välillä. Integraatio mobiiliverkkoihin esittää reunajärjestelmän ja mobiiliverkon yhteistoiminnallisuuden tyyppiä. Kommunikaatio käsittelee reunajärjestelmän ja asiakaslaitteen välistä tietoliikenneyhteyttä. Viimeisenä käsitellään reunajärjestelmän hallintaa. Hallinta sisältää päätöksiä järjestelmän hallinnollisten toimien toteuttamiseksi, kuten hallinnollisten entiteettien sijainnin järjestelmässä. Ominaisuuksien käsittely on toteutettu yleisellä tasolla. Arkkitehtuurikohtaiset toteutusehdotukset näille ominaisuuksille käsitellään luvussa 6.

Ominaisuudet eivät ole luonteeltaan täysin itsenäisiä, vaan ne ovat toisistaan riippuvaisia. Riippuvuuksista seuraa, että kunkin ominaisuuden toteutuksen vahvuudet ja heikkoudet tulee ottaa huomioon laajemmassa kontekstissa.

4.1 Live migraatio

Live migraatio, eli suorituksen aikanen siirto, tarkoittaa virtualisoidun ohjelman tai virtuaalikoneen siirtämistä laitteistolta toiselle, siten että virtuaalikoneen käyttö ei keskeydy siirron ajaksi [Clark et al., 2005]. Useimmiten tällaista toiminnallisuutta käytetään palvelinkeskuksissa virtuaalikoneiden siirtelyyn. Palvelinkeskuksissa siirtämiseen käytetään nopeita sisäisiä yhteyksiä, jolloin tiedon siirtoon käytettävät väylät ovat nopeita. Syitä live migraation suorittamiseksi on muutamia. Perinteisessä palvelinympäristössä live migraation syinä ovat virrankulutuksen optimointi, kuorman tasaaminen tai fyysisen laitteiston huoltoon ottaminen [Soni and Kalra, 2013].

Reunalaskentaan sovellettuna live migraation tehtävänä on laskentaa tai palvelua suorittavan virtuaalikoneen siirtäminen reunasolmujen välillä. Syyt joiden vuoksi reunalaskentaympäristössä tehdään live migraatiota koskevat pääasiassa asiakaslaitteen liikkumista verkossa, jolloin virtuaalikone siirretään asiakaslaitetta lähimpänä olevalle reunasolmulle. Reunalaskentaa suoritettaessa voi myös tulla tilanne, jossa laskentaa suorittavalla reunasolmulla ei ole resursseja laskennan suorittamiseksi. Tällöin asiakkaan virtuaalikone saatetaan joutua joudutaan siirtämään toiselle reunasolmulle, jolla on enemmän resursseja.

Handover/handoff

Virtuaalikoneille tehtävän live migraation ja mobiiliverkossa suoritettavan handoverin ideat ovat samankaltaisia. Molemmissa tavoitteena asiakasta palvelevan entiteetin siirtäminen ilman että asiakas huomaa siirtoa. Handoverilla tarkoitetaan asiakaslaitteen radioyhteyden siirtämistä tukiasemalta toiselle tukiasemalle. LTE:ssä handover päätös tehdään eNodeB:ssä asiakaslaitteen välittämän mittaustuloksen perusteella. Handover voidaan tehdä esimerkiksi tilanteessa jossa asiakaslaitteen ympäristössä on toinen tukiasema, joka voisi palvella asiakaslaitetta paremmin [ETSI, 2016a, s. 96]. LTE:ssä yhteys

on tyypiltään tunnelimainen ja handoverissa tunnelin toinen pää siirretään toiselle tukiasemalle. Yhteyden siirron alkaessa tukiasema välittää kohteena olevalle tukiasemalle asiakaslaitetta koskevat tilatiedot. Kun tilatiedot on välitetty, asiakaslaitteen ja tukiaseman välinen yhteys katkaistaan ja asiakaslaite muodostaa yhteyden uudelle tukiasemalle. Asiakaslaitteen tietoliikennettä puskuroidaan S-GW:n toimesta sillä välin kun yhteys on katkaistuna. Yleensä handover on niin nopea toimenpide, että laitteen käyttäjä ei sitä huomaa. Handover on nopea toimenpide koska sen aikana siirrettävän tiedon määrä on vähäinen. Handoverin sisältämä asiakaskonteksti koostuu pääasiassa erilaisista asiakaslaitteen ja tunneleiden tunnisteista.

Live migraation toiminta

Live migraatiossa virtuaalikone siirretään palvelimelta toiselle palvelimelle, ilman että virtuaalikoneen käyttö keskeytyy. Perinteinen live migraatio on toteutettu siten, että virtuaalikoneen suoritustilan eli prosessorin tilan ja muistin siirretään toiselle palvelimelle.

Siirtäminen suoritetaan iteraatioittan siten, että ensimmäisellä iteraatiolla kaikki muistisivut siirretään kohdelaitteelle [Clark et al., 2005]. Seuraavilla kierroksilla lähtölaitteelta siirretään vain ne muistisivut joille on tapahtunut muutoksia edellisen siirron alkamisen jälkeen. Tätä jatketaan kunnes muutoksia sisältävien muistisivujen määrä ei vähene iteraatioittan. Tässä vaiheessa lähtöpisteenä oleva virtuaalikone pysäytetään ja loput virtuaalikoneen muistisivut ja tilatiedot siirretään kohdelaitteelle. Tämän jälkeen virtuaalikone käynnistetään uudessa sijainnissa.

Live migraatioon liittyy myös erilaisia optimointeja ja lähestymistapoja, joita ei tässä tutkielmassa sen tarkemmin avata. Palvelinsaliympäristössä on myös yleistä, että virtuaalikoneen tallennustilaa ei ole tarpeen siirtää, koska se on toteutettu levypalvelimen avulla, jolloin ainoastaan yhteys täytyy siirtää [Clark et al., 2005]. Mikäli näin ei ole, myös tallennustila tulee siirtää laitteelta toiselle. Live migraatio vaatii myös avoimina olleiden tietoliikenneyhteyksien siirtämisen [Clark et al., 2005].

Live migraation suorituskykyä mitataan seuraavilla metriikoilla [Soni and Kalra, 2013]:

- Migraation kesto: Aika joka kuluu migraation suorittamiseen
- Katkon kesto: Aika jona palvelut eivät käytettävissä
- Siirretyn datan määrä: Migraatiosta aiheutuva tietoliikenteen määrä
- Migraation yleisrasite: Paljonko migraatio vie järjestelmän resursseja
- Suorituskyvyn alentuma: Siirrettävän virtuaalikoneen suorituskyvyn heikentyminen siirron aikana

Näistä kolme ensimmäistä ovat keskeisimpiä [Farris et al., 2017]. Tavoitteena perinteisessä live migraatiossa on mahdollisimman lyhyt käyttökatko [Ha et al., 2015].

Reunalaskennassa

Reunalaskentaympäristössä suoritettavan live migraation toiminnallisuus on pääpiirteittäin sama kuin edellä kuvattu. Reunalaskennassa live migraatiolla viitataan reunasovelluksen suoritusaikaiseen siirtoon reunasolmujen välillä. Live migraation tavoitteena on taata asiakaslaitteen parempi palvelu. Keinona tuon tavoitteen saavuttamiseksi on esimerkiksi reunasovelluksen siirtäminen asiakasta lähempänä sijaitsevalle reunasolmulle. Reunalla tehtävän live migraation erona palvelinsaliympäristöön on juurikin suoritusympäristö.

Päällimmäisenä erona on palvelinsaleissa tehtävään migraatioon on käytettävien yhteyksien nopeus. Reunasolmujen väliset yhteydet ovat oletettavasti hitaampia ja nopeus saattaa vaihdella[Ha et al., 2017]. Hitaat yhteyden johtavat pitkään siirtoaikaan, joka puolestaa johtaa pidempään käyttökatkokseen ja täten palvelun laadun heikkenemiseen. Reunajärjestelmissä ei myöskään oletettavasti ole jaettua levyplalvelinta, vaan reunalaskennan yhteydessä live migraatio sisältää myös massamuistin siirron.

Esimerkkitilanteessa jossa asiakaslaite on siirtynyt alkuperäisen reunasolmun kantaman ulkopuolelle reunasovelluksen käyttö voi jatkua reititysmuutoksen avulla. Asiakaslaitteen siirtyessä kauemmaksi alkuperäisestä reunasolmusta, viiveet kasvavat ja palvelun laatu heikkenee. Reunajärjestelmä aloittaa asiakasta palvelevan reunasovelluksen migraation lähempänä sijaitsevalle reunasolmulle. Huomion arvoinen seikka on että asiakkaan palvelun laatu pysyy heikentyneenä niin kauan kuin live migraatio on käynnissä ja alkuperäisen reunasolmun edelleen palvellessa asiakasta. Toisin sanoen reunalaskennassa live migraation kannalta on tärkeää minimoida live migraatioon kuluva kokonaisaika [Ha et al., 2015]. Kokonaisajan minimoimiseksi joudutaan todennäköisimmin luistamaan katkoksen kestosta. Etenkin hitailla yhteyksillä tiedonsiirtoon kuluva aika on kertaluokkaa suurempi kuin katkoksen kesto [Ha et al., 2017]. Siirtoajan minimoimiseksi on pyrittävä pitämään siirrettävän datan määrä mahdollisimman pienenä [Ha et al., 2015].

Palvelinsaleja ja reunalaskentaympäristöä erottaa myös se, että palvelinsaliympäristössä virtuaalikoneiden live migraatiot voidaan pääsääntöisesti tehdä koordinoidusti ilman tiukkkoja aikarajoitteita. Reunalaskentaympäristössä migraatiopäätös perustuu usein johonkin ulkoiseen tapahtumaan. Todennäköisintä onkin että migraatiopäätös tehdään samankalataisin perustein kuin edellä esitellyn handoverin. Keskeisin syy migraatiolle oletettavasti on asiakaslaitteen liikkuminen verkossa, mutta syynä voi olla myös esimerkiksi reunasolmun ruuhkautuminen. Asiakaslaitteiden liikkumiseen pohjautuva migraatio yhdistettynä heikkoihin tietoliikenneyhteyksiin on kuitenkin ongelmallinen. Voidaan esimerkiksi kuvitella tilanne jossa aamulla

kaupungin keskustaan saapuvat työmatkailijat aiheuttavat "migraatiotulvan". Suuri määrä migraatioita saattaa ruuhkauttaa reunasolmujen käytössä olevat tietoliikenneyhteydet ja käyttää suuren osan käytössä olevista resursseista.

Lähes kaikki läpikäydyt reunalaskenta-arkkitehtuurit tiedostivat tarpeen reunasovelluksien migraatiolle, mutta kovin harvassa määriteltiin täsmällisiä toimia sen toteuttamiseksi. Cloudlettien ympärillä tehdyn tutkimuksen yhteydessä on esitetty migraatioratkaisua, joka lyhentäisi migraation kestoa merkittävästi. Cloudlet ratkaisu käsitellään tarkemmin kappaleessa 6.1. Onkin realistista olettaa, että varsinaiseen reunajärjestelmään toteutettava live migraatio toiminnallisuus optimoitaisiin reunaympäristöön sopivaksi, kuten cloudlettien yhteydessä on pyritty tekemään.

4.2 Integraation tyyppi

Lähes kaikki mobiiliverkossa toimivat asiakaslaitteet kuuluvat reunalaskennan kohderyhmään. Täten reunajärjestelmän integrointi osaksi mobiiliverkkoa on luonnollinen tavoite. Lisäksi mobiiliverkkoon sijoitettu reunajärjestelmä vaikuttaa hyvältä ratkaisulta tilanteessa jossa asiakaslaite on liikkuva [Gusev and Dustdar, 2018]. Tässä yhteydessä integraatiolla tarkoitetaan mobiiliverkon ja reunajärjestelmän tapaa hoitaa reunapalveluiden tarjoaminen mobiiliverkon kautta.

Tutkielmassa käsiteltävät reuna-arkkitehtuurit ovat esittäneet erilaisia keinoja reunajärjestelmän ja mobiiliverkon yhdistämiseksi. Kun reunajärjestelmää ollaan integroimassa osaksi olemassa olevaa mobiiliverkkoa, on tavoitteena pyrkiä pitämään uusittavan laitteiston määrä kohtuullisena. Pääasiallisena syynä tähän on mobiiliverkon suljettu ympäristö, johon mobiiliverkon operaattori joutuu erikseen hankkimaan tarvittavat toiminnallisuudet.

Vaihtoehtojen toisessa ääripäässä on koko järjestelmän uusiminen, siten että se tukee reunalaskentaa osana mobiiliverkon muita toiminnallisuuksia. Toisessa ääripäässä mobiiliverkkoon ei tehdä lainkaan muutoksia ja reunajärjestelmä lisätään joidenkin mobiiliverkon ulkopuolisten toimijoiden avulla.

Integraation haasteena on mobiiliverkon jatkuva kehitys. Arkkitehtuuriehdotukset ovat jossain määrin ottaneet näitä kehityspolkuja huomioon. Yksi näistä kehityspoluista on SDN ja NFV käyttöönotto mobiiliverkoissa [Heinonen et al., 2014, ETSI, 2012]. Näiden avulla tavoitellaan mobiiliverkon laitteiston ja toiminnallisuuden välisen sidonnaisuuden vähentämistä sekä skaalautuvuuden parantamista.

Integraation tyypistä seuraa epäsuorasti myös reunajärjestelmän Tarjoja. Varsinkin ehdotukset joissa reunajärjestelmä on tiukasti sidoksissa mobiiliverkkoon reunajärjestelmän tarjoaja on sama kuin mobiiliverkon operaattori. Heikompaa sidonnaisuutta ehdottavat ratkaisut saattavat tarjota mahdollisuuden että reunajärjestelmää tarjoaa jokin kolmas osapuoli. Kysymykseksi siis jää, onko integraation tuloksena yksi järjestelmä, vai kaksi järjestelmää

joiden välillä on jonkinlainen yhteys.

Reuna-arkkitehtuurien ehdottamat tavat mobiiliverkon ja reunajärjestelmän integraatioon on jaettu tässä tutkielmassa kolmeen ryhmään:

- **Suorat integraatiot** sisältävät uusien toimintojen lisäämistä osaksi mobiiliverkkoa. Tämänkaltainen ratkaisu edellyttää muutoksia myös olemassa olevien komponenttien toimintaan.
- **Epäsuorat integraatiot** tarjoavat reunalaskentaa mobiiliverkon ulkopuolella. Ei edellytä toiminnallisia muutoksia mobiiliverkkoon, mutta saattaa edellyttää muita mobiiliverkon muutoksia.
- **Läpinäkyvät integraatiot** vaativat muutoksia mobiiliverkkoon, mutta eivät vaadi muutoksia olemassa olevien komponenttien toimintaan.

Suorat integraatiot edustavat niiden ratkaisujen joukkoa jotka muokkaavat olemassa olevan mobiiliverkkoarkkitehtuuria. Näiden ratkaisujen voidaan olettaa olevan kalleimpia, koska yhteistoiminnallisten toimijoiden lisääminen olemassa olevaan infrastruktuuriin vaatii muidenkin toimijoiden päivittämistä. Pääasiassa nämä ratkaisut tähtäävät lisäämään reunalaskentaa viidennen generaation mobiiliverkkoihin, mutta myös LTE:hen pohjautuvia ratkaisuja on esitetty. Koska viidennen generaation määrittelytyö on vielä kesken, ehdotetut ratkaisut rakentuvat osittain oletuksien päälle. Esimerkkinä suorasta integraatiosta on CONCERT (esitellään kappaleessa 6.5). Suora integraatio mahdollistaa reunalaskennan tuomisen niin lähelle reunaa kuin mahdollista.

Epäsuoralla integraatiolla tarkoitetaan ratkaisuja, joiden pääasiallinen toiminnallisuus on sijoitettu mobiiliverkkoarkkitehtuurin ulkopuolelle. Tällaiset ratkaisut mahdollistavat reunajärjestelmän tuottamisen kolmansilla osapuolilla. Esimerkkinä tällaisesta ratkaisusta on FMC (käsitellään kappaleessa 6.2). Haasteena tämänkaltaisissa ratkaisuissa on optimaalisen reunasolmun valinta, koska se joudutaan tekemään asiakaslaitteen antamien tietojen pohjalta. Onkin siis huomattava, että asiakaslaite joutuu osallistumaan reunalaskentaan liittyviin hallinnollisiin toimiin.

Läpinäkyvissä ratkaisuissa reunalaskennan mahdollistavat toiminnot on toteutettu siten, että suoria yhteyksiä mobiiliverkon toimintoihin ei ole. Periaate on hieman samankaltainen kuin läpinäkyvässä välityspalvelimessa (transparent proxy). Ratkaisut ovat mobiiliverkon näkökulmasta "näkymättömiä". Läpinäkyvässä ratkaisussa mobiiliverkko ja reunajärjestelmä eivät jaa hallinnollisia entiteettejä. Molemmat järjestelmät ovat siis itsenäisiä järjestelmiä. Läpinäkyvyys tarkoittaa käytännössä, että johonkin kohtaa mobiiliverkon sisäisiä yhteyksiä, lisätään monitori, joka mahdollistaa tietoliikenteen seuraamisen ja muokkaamisen. Monitorin tehtävänä on tarkkailla mobiiliverkon tapahtumia sekä ohjata haluttu tietoliikenne reunalaskennalle. Monitoritoiminnallisuuden toteuttamisen helpottamiseksi käytössä on tai

käyttöön oletetaan SDN, NFV tai molemmat. Vaikka läpinäkyvillä ratkaisuilla ei ole kommunikaatiorajapintaa mobiiliverkon hallinnollisiin toimijoihin, niille oletetaan yhteys mobiiliverkon sisäiseen verkkoon. Tämä sitoo ne osaksi mobiiliverkon infrastruktuuria ja käytännössä tämä tarkoittaa, että reunajärjestelmää ylläpitää mobiiliverkon operaattori. Esimerkkinä läpinäkyvästä ratkaisusta on MobiScud, joka käydään tarkemmin läpi kappaleessa 6.4.1.

4.3 Kommunikaatio

Reunalaskenta vaatii toimiakseen kommunikaatioväylän asiakaslaitteen ja reunajärjestelmän välille. Asiakaslaitteen ja reunajärjestelmän välisen kommunikaatioväylän toteuttamisen haasteena on tapa, jolla mobiiliverkkojen sisällä tapahtuva reititys on toteutettu. Kuten jo aiemmin mainittu, mobiiliverkon sisäisen reititys ei käytä reitittämiseen asiakastietoliikenteen IP-osoitteita, vaan asiakastietoliikenne ohjataan GTP tunneloituna P-GW:n kautta Internettiin. Reunajärjestelmän integraation tyypillä on selkeä vaikutus tapaan, jolla kommunikaatioväylä toteutetaan. Reuna-arkkitehtuurit esittivät erilaisia keinoja kommunikaatioväylän toteuttamiseksi.

Koska mobiiliverkko ei käsittele asiakkaan tietoliikennettä IP-tasolla, mobiiliverkko on asikastietoliikenteen näkökulmasta näkymätön. Tämä tarkoittaa, että normaali reititystoiminnallisuus ei mahdollista esimerkiksi asiakaslaitteeltä lähtöisin olevan, reunasolmulle tarkoitetun paketin ohjaamista kohde IP-osoitteen perusteella. Integraation tyypistä riippuen mobiiliverkon sisäisiä tietoliikenteen ohjausmetodeja saatetaan joutua muokkaamaan. Ensimmäinen ratkaistava ongelmana on siis, että kuinka asiakaslaitteen tietoliikenne voidaan ohjata mobiiliverkon sisällä sijaitsevalle reunajärjestelmälle. Tämän lisäksi joudutaan ottamaan huomioon tilanteet, jossa asiakaslaitteen tai reunpalvelun sijainnit verkossa vaihtuvat.

Tietoliikenteen haarauttaminen

Ensimmäiseen ongelmaan on ehdotettu erilaisia ratkaisuja, joiden toimintamalli riippuu pääasiassa siitä, kuinka tiukasti reunajärjestelmä on integroitu osaksi mobiiliverkkoa. Ongelman ratkaisussa on otettava huomioon, että tietoliikennettä on aina kahteen suuntaan. Tämän lisäksi asiakaslaitteen tietoliikenne pitää jatkossa haarauttaa suuntautumaan joko reunajärjestelmälle tai internettiin. Koska perinteisesti asiakaslaitteen tietoliikenne on voitu ohjata P-GW:n kautta ulos verkkoon, ei mobiiliverkon sisällä ole valmiiksi mekanismia jonka avulla asiakkaan tietoliikennettä voisi ohjata. Tähän ongelmaan on ehdotettu ratkaisumallia, joka perustuu ajatukseen tietoliikennettä monitoroivasta entiteetistä. Tällaisella monitorilla olisi mahdollisuus tarkkailla mobiiliverkossa kulkevan tietoliikenteen kohdetta, esimerkiksi kohde IP-osoitetta, ja ohjata paketit tarvittaessa reunajärjestelmälle tai asiakaslaitteelle (mikäli ne ovat lähtöisin reunajärjestelmältä). Tämä jättäisi kaiken

reunajärjestelmälle kuulumattoman tietoliikenteen koskemattomaksi.

LIPA (Local IP Access) ja SIPTO (Selected IP Traffic Offload) ovat ehdotuksia mobiiliverkkoon tehtävästä reititysratkaisusta [Samdanis et al., 2012a, Samdanis et al., 2012b]. LIPA ja SIPTO lisäisivät tukiasemien yhteyteen L-GW:n (Local Gateway), jonka mahdollistaisi asiakaslaitteen tietoliikenteen ohjaamisen tukiaseman yhteydestä esimerkiksi reunasolmulle. Tällöin tietoliikenteen ei tarvitse kulkea EPC:n kautta. L-GW:n tarkoitus olisi siis mahdollistaa nopeammat yhteydet tukiaseman läheisyydessä sijaitsevaan verkkoon. Tätä olisi mahdollista hyödyntää reunalaskennassa, mutta se monimutkaistaisi olemassa olevan mobiiliverkon toiminnallisuutta [Cho et al., 2014]. LIPA:a ja SIPTO:a ei kuitenkaan ole ehdotettu minkään reuna-arkkitehtuurin yhteydessä ratkaisuksi tietoliikenteen ohjaamiseen.

Reuna-arkkitehtuureissa tietoliikenteen tarkkailua ja ohjausta mahdollistava toiminnallisuus on ehdotettu toteutettavan SDN:llä, NFV:llä tai jollain ratkaisuspesifillä toiminnallisuudella. SDN:n käyttö ratkaisuna tarjoaa tavan reitittää liikennettä mobiiliverkon sisällä. Se lisäksi mahdollistaa reitityksien ohjelmallisen muokkaamisen, jolloin tiettyjen yhteyksien reittiä voidaan dynaamisesti muokata. SMORE:n yhteydessä esitetyssä ratkaisussa E-UTRAN ja EPC:n välille sijoitetaan SDN kerros, jossa eNodeB ja EPC:n välistä tietoliikennettä voidaan monitoroida. Koska tällä välillä oleva tietoliikenne on GTP tunneloitua, paketteja joudutaan purkamaan ja uudelleen paketoimaan. Tilanteessa jossa paketin suunta on asiakaslaitteelta reunalle, paketin tunnelointi joudutaan purkamaan ja varsinainen paketti välitetään reunapalvelulle. Reunalta asiakaslaitteelle suuntautuvat paketit joudutaan uudelleen kapseloimaan GTP:n mukaisiksi. Uudelleen kapselointi vaatii erinäisten mobiiliverkon sisäisten tunnisteiden seurantaa ja hyödyntämistä. Tarkempi kuvaus toiminnasta annetaan SMORE:n käsittelyn yhteydessä kappaleessa 6.4. Ratkaisu spesifisen toiminnallisuuksien yhteydessä noudatetaan hyvin samankaltaista toimintamallia. Esimerkiksi Small Cell Cloud (käsitellään kappaleessa 6.3) ratkaisun yhteydessä pakettien monitorointi on sijoitettu tukiasemiin, josta GTP tunnelointi alkaa. Täten paketit voidaan monitoroida ennen GTP tunnelointia ja välittää tarpeen mukaan reunasolmuille.

Monitorointia suorittavalla entiteetillä on myös muuta käyttöä kuin asiakasliikenteen ohjaaminen reunajärjestelmälle. Kontrollikerroksen pakettien tarkkailu paljastaa esimerkiksi tukiasemien välisen handoverin alkamisen. Mobiiliverkon kontrollikerroksen monitorointia voidaan hyödyntää reunajärejestelmän hallinnollisten toimien toteuttamiseen.

Liikkuvuuden vaikutus kommunikaatioväyliin

Toinen ongelma liittyy tilanteeseen, jossa asiakaslaite siirtyy sijainnista toiseen. Tavoitetilana on, että tietoliikenneyhteys asiakaslaitteen ja reunasovelluksen välillä säilyy liikkumisesta huolimatta. Mobiiliverkkoa käyttävän asiakaslaitteen liikkuminen johtaa yleensä jossain vaiheessa handoveriin. Verk-

koyhteys siis siirtyy kulkemaan toisen tukiaseman kautta. Asiakaslaitteen tietoliikenneyhteyksien kannalta handover on käytännössä huomaamaton. Handover on toiminnallisuutena mobiiliverkon sisäinen toiminnallisuus. Handover on suunniteltu tukemaan ulkoverkkoon S-GW:n ja P-GW:n kautta kulkevien yhteyksien muutoksia, siten että asiakaslaitteen tietoliikenneyhteydet eivät katkea. Reunajärjestelmä joutuu siis itse hallinnoimaan liikkuvuuteen liittyviä tapahtumia. Integraation tyypillä on tässäkin keskeinen merkitys. Esimerkiksi suoran integraation järjestelmissä on teoriassa mahdollista vaikuttaa handoverien toimintaan. Läpinäkyvät järjestelmät puolestaan joutuvat reagoimaan mobiiliverkon handovereihin monitorointitoiminnallisuutensa kautta.

Reunalaskenta-arkkitehtuurien yhteydessä on esitetty ratkaisuja, jotka säilyttävät asiakaslaitteen yhteyden reunasolmuun handoverin yhteydessä. Ratkaisut kuitenkin monimutkaistuvat, mikäli myös reunasovelluksen sijaintia halutaan siirtää. Reunalaskennan siirto oletetaan toteutetuksi kuten live migraatio kappaleessa 4.1 esitettiin. Tilanteessa jossa ainoastaan asiakaslaitteen sijainti muuttuu, yhteyden säilyttäminen vaatii reunasolmun sijainnista riippuen vain vähäisiä toimia. Esimerkiksi MobiScud arkkitehtuurissa, monitorilta saadaan tieto handoverin vaiheista ja tämä tieto välitetään MobiScudin hallinnolliselle entiteetille. Tämän tiedon saatuaan hallinnollinen entiteetti päivittää SDN reitityssääntöjä (flow rule), siten että tietoliikenne virtaa mobiiliverkon yhteysmuutoksista huolimatta reunasovellukselle.

Pelkän yhteyden varmistamisen ei kuitenkaan oleteta riittävän ja reuna-arkkitehtuurit esittävätkin reunapalvelun live migraatiota. Mikäli reunapalvelua ei siirretä, asiakaslaitteen ja reunasolmun välinen etäisyys saattaa kasvaa, joka näkyy palvelun laadussa kasvavina viiveinä [Taleb and Ksentini, 2013, Wang et al., 2015]. Lähes jokainen arkkitehtuuri ehdotti reunapalvelun live migraatiota reunasolmujen välillä. Migraation seurauksena virtuaalikoneen IP-osoite todennäköisesti vaihtuu. Koska tietoliikenneyhteydet pohjautuvat normaalist IP-osoitteisiin, aiheuttaa tämä muutos haasteita palvelun jatkuvuuden kannalta. Asiakaslaitteen ja reunapalvelun yhteyden muodostamiseen on ehdotettu toisenlaisia ratkaisuja. Follow Me Cloud (käsitellään tarkemmin kappaleessa 6.2) yhteydessä ehdotetaan ratkaisuksi asiakaslaite tunnisteeseen ja reunapalveluun liittyvän palvelutunnisteeseen perustuvaa tunnistetta. FMC:ssä edellä mainituista tunnisteista generoidaan sessio/palvelu tunniste (Session/Service ID), jonka avulla asiakaslaitteeseen liittyvä reunapalvelu tai muu sessio voidaan tunnistaa ja siihen sessioon liittyvät yhtedet uudelleenmuodostaa. Sessio/palvelu tunnisteen avulla asiakaslaite ja reunapalvelu voidaan siis yhdistää IP-osoitteiden muutoksista riippumatta. Tunnisteen lisääminen vaatii järjestelmään entiteetin, joka ylläpitää listaa käytössä olevista tunnisteista, sekä tarjoaa niiden aksessointiin menetelmän. Tämä saattaa monimutkaistaa palveluiden saavutettavuutta.

Asiakaslaitteen ja reunapalvelun välisen yhteyden säilyttäminen on hyvin riippuvainen reuna-arkkitehtuurin muista toiminnallisuuksista. Asiakaslait-

teiden ja reunapalveluiden väliseen kommunikaatioon vaikuttavat esimerkiksi reunajärjestelmän virtuaalikoneiden migraatio politiikka sekä muut palveluiden tuottamiseen liittyvät valinnat.

4.4 Hallinta

Reunajärjestelmän hallinta koostuu joukosta reunajärjestelmän toimintaa sääteleviä entiteettejä. Tutkielmassa käsiteltävistä reunalaskenta-arkkitehtuureista identifioitiin päävastuussa oleva hallinnollinen toimija sekä joitakin hallinnollisia toimintoja. Edellä kuvattu live migraation laukaiseminen ja kommunikaatioväylän muodostaminen olivat keskeisimmät tunnistetut hallinnolliset toiminnot. Näiden toimien perusperiaate on, että hallinnollinen toimija saa herätteen jostain mobiiliverkon tapahtumasta ja välittää tarvittavat toimenpiteet reunasolmuilla sijaitsevalle reuna-alustalle.

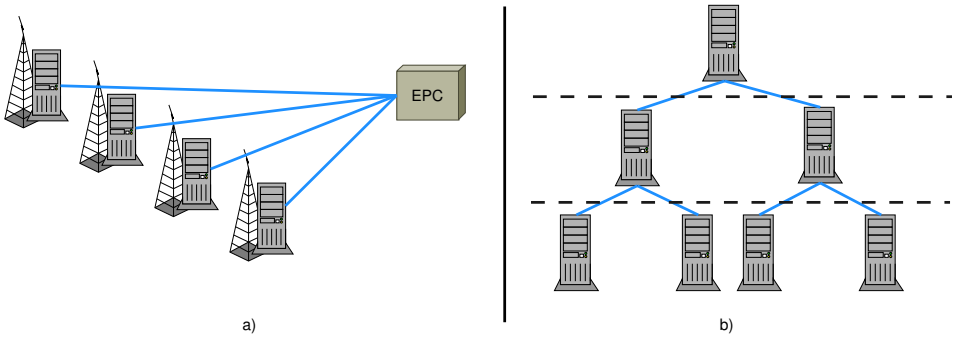
Pääasiassa hallinnollisten toimijoiden rooli esiintyy reunalaskenta-arkkitehtuurissa passiivisena tarkkailijana, jonka tehtävä on reagoida mobiiliverkon tapahtumiin. Esimerkki tällaisesta tapahtumasta on mobiiliverkossa tapahtuva handover, johon hallinnollinen entiteetti saattaa haluta reagoida esimerkiksi aloittamalla reunalaskennan migraation lähemmäksi käyttäjää. Myös aktiivisten hallinnollisten toimijoiden tarve tiedostetaan, mutta tällä arkkitehtuuritasolla niitä ei määritellä. Aktiivisten hallinnollisten suunnittelu jätetään reunajärjestelmän toteuttajalle. Aktiivisia toimia ovat muun muassa reuna-alustalla suoritettavien virtuaalikoneiden elinkaaren hallinta [Yousaf and Taleb, 2016]. Kappaleessa 6.1 esitellään Cloudlet niminen ratkaisu, joka kuvaa virtuaalikoneiden elinkaaren hallintaa ja ehdottaa live migraatio toiminnallisuutta reunasolmujen välille.

Hallinnollinen vastuu voi olla reunajärjestelmässä hajautettuna tai keskitettynä. Hajautetussa mallissa hallinnollinen entiteetti vastaa jostain reunasolmujen osajoukosta [Lobillo et al., 2014]. Hajautus voi myös olla hierarkinen jolloin yhdellä hallinnollisella entiteetillä on ylemmän tason hallinnollinen toimija joka tarpeen tullen vastaa reunalaskennan organisoinnista [Mach and Becvar, 2017].

Kuten muutkin ominaisuudet, myös hallinnollisen toimijan sijainti riippuu reunajärjestelmän integraation tyypistä. Suorassa integraatiossa hallinnollinen toimija on mahdollista sijoittaa käytännössä mihin tahansa [Mach and Becvar, 2017]. Läpinäkyvässä integraatiossa hallinnollinen toimija sijaitsee yleensä lähellä monitoritoiminnallisuutta, mutta kuitenkin mobiiliverkon ulkopuolella. Epäsuorassa integraatiossa hallinnolliset toimijat sijaitsevat mobiiliverkon ulkopuolella.

5 Rakenne

Seuraavaksi esitellään reunajärjestelmän rakennetta. Koska aihepiirinä on arkkitehtuurit, varsinaisten toteutettavien reunajärjestelmien rakennetta voi-



Kuva 4: Esimerkki rakenteet a) litteä rakenne b) hierarkkinen rakenne kolmella tasolla

daan tarkastella ainoastaan implisiittisesti. Eli mitä arkkitehtuurit mahdollistavat tai rajoittavat. Ensimmäiseksi esitellään reunan fyysisen rakenteen eri mallit, jossa keskeisessä osassa ovat reunasolmujen sijainnit. Tämän jälkeen käsitellään reuna-arkkitehtuurien vaikutuksen toteutettavaan järjestelmään.

Erilaiset rakenteet voidaan jakaa kahteen päätyyppiin: litteään ja hierarkkiseen. Litteä rakenne edustaa yksinkertaisempaa lähestymistapaa resurssien asetteluun. Hierarkkinen rakenne puolestaan edustaa hallinnollisesti monimutkaisempaa järjestelmää, jossa reunasolmut voivat delegoivat tehtäviä suhteessa tehokkaammille ja keskitetymmille reunasolmuille.

Reunan rakenteeseen vaikuttaa myös tapa jolla järjestelmä tuotetaan. Esimerkiksi tilanne jossa reunajärjestelmä toteutetaan siten, että yksityisen toimijat, kuten yritykset, voivat hankkia omia reunasolmuja ja liittää niitä osaksi reunajärjestelmää. Tällöin reunajärjestelmän rakenne on vaihteleva eikä siihen todennäköisesti litteä. Tällaisessa tilanteessa keskeiseksi tekijäksi nouseekin reunapalveluiden tuottamiseen käytettävä liiketoimintamalli. Yksinkertaisuuden vuoksi tässä kappaleessa aihetta käsitellään, kuin reunan rakenne olisi yhden entiteetin, esimerkiksi operaattorin, päätettävissä.

Reunavyöhyke Reunavyöhyke on tämän tutkielman puitteissa määritelty termi, jolla viitataan vyöhykkeeseen, jolla reunasolmut sijaitsevat. Reunavyöhyke voi olla kapea tai leveä, jotka vastaavat suurin piirtein seuraavaksi määriteltävää litteää ja hierarkkista rakennetta. Kapea vyöhyke tarkoittaa että reunasolmut sijaitsevat verkossa samalla tasolla. Leveä vyöhyke puolestaan viittaa rakenteeseen jossa reunasolmujen etäisyys asiakkaasta vaihtelee merkittävästi. Leveässä järjestelmässä reunasolmut voivat sijaita esimerkiksi lähimmillään tukiasemassa ja kauimmillaan mobiiliverkon ulkopuolella sijaitsevassa palvelinsalissa.

5.0.1 Litteä rakenne

Litteällä rakenteella tarkoitetaan että reunasolmut sijaitsevat vain yhdessä kerroksessa. Litteän järjestelmän keskeisin päätös on valita taso, jolle reunasolmut sijoitetaan.

Yksinkertainen esimerkki litteästä toteutuksesta olisi reunajärjestelmä, jossa reunasolmut sijoitettaisiin mobiiliverkon tukiasemien yhteyteen kuten kuvassa 4 a). Tämän kuvan esimerkin järjestelmä toteuttaisi äärimmäistä hajautusta. Se olisi fyysisesti erittäin lähellä asiakaslaitetta ja näin ollen pystyisi teoriassa tarjoamaan palvelua kaikkein pienimmällä viiveellä [Mach and Becvar, 2017]. Vaikka viiveet olisivat pienet, on tällaisessa ratkaisussa ongelmia. Järjestelmän käyttö olisi teoriassa mahdollista ainoastaan niiden tukiasemien ympäristössä, joissa reunajärjestelmä sijaitsee. Tämän seurauksena reunalaskennan laajamittainen käyttöönotto olisi riippuvainen tukiasemiin sijoitettavien reunasolmujen hankinnasta. Kun otetaan huomioon trendi, jossa mobiiliverkon solut pienevät, niin ainoastaan tukiasemiin sijoitettavien reunasolmujen käyttöönotto vaikuttaa epärealistiselta skenaariolta.

Mikäli oletetaan, että siellä missä laskennan tarve on suurin, on myös eniten ihmisiä. Yleensä tällaisissa sijainneissa myös reunajärjestelmän resursseja varten tarvittavien tilojen hinnat ovat korkeammat, jolloin reunalaskennan tuominen lähelle asiakasta nostaa reunajärjestelmän hintaa [Mao et al., 2017]. Vaihtoehtoinen ratkaisu olisi siirtää reunasolmuja kauemmaksi tukiasemista. Esimerkiksi siten, että yksi reunasolmu vastaisikin useamman tukiaseman kautta tulevasta reunalaskennasta. Mobiiliverkko kontekstissa, reunasolmu sijaitsisi siis joko tukiasemien ja EPC:n välillä tai vasta EPC:n takana. Useamman tukiaseman niputtamista yhden reunasolmun vastuulle kutsutaan klusteroinniksi [Bouet and Conan, 2017]. Keskeisenä haasteena klusteroinnissa on valita oikean kokoiset klusterit ja sijoittaa oikea määrä resursseja kuhunkin klusteriin [Malandrino et al., 2016]. Klusterin suuruus ja viive todennäköisesti korreloivat, jolloin on varottava tekemästä klustereista tarpeettoman suuria. Klusteroinnin pitäisi kuitenkin mahdollistaa helpompi ja edullisempi käyttöönotto, koska tarvitaan suhteessa vähemmän reunasolmuja ja reunasolmujen ylläpito on näin ollen edullisempaa. Kompromissina on kuitenkin suurempi viive reunasolmujen ja asiakaslaitteiden välillä. Onkin siis tärkeää että reunasolmuja ei viedä liian kauas reunasta, jottei palvelun laatu heikkene [Mao et al., 2017].

Kuitenkin riippumatta litteän rakenteen sijainnista, sen pohjimmaisena ongelmana on resurssien kohdistuminen jollekkin tietylle palvelualueelle. Ympäristössä jossa reunalaskennan määrä vaihtelee, seuraa tilanne jossa reunasolmu on joko ylikuormitettuna tai alikuormitettuna suhteessa käytössä oleviin resursseihin [Tong et al., 2016]. Jotta reunasolmut pystyisivät suoriutumaan kaikesta reunalaskennasta, reunasolmujen resurssit jouduttaisiin mitoittamaan rasitushuippujen mukaan. Tästä seuraa lähes jatkuvaa alikuormitusta, joka tarkoittaa että reunajärjestelmän kustannukset kasvavat

resurssihankintojen seurauksena. Tämän ongelman ratkaisuksi on esitetty hierarkkista järjestelmää.

5.0.2 Hierarkkinen rakenne

Hierarkkinen rakenne on parannusehdotus reunalaskennan oletetulle litteälle rakenteelle [Tong et al., 2016]. Reunasolmujen hierarkkisella asettelulla tarkoitetaan järjestelmää, jossa reunasolmut on jaettu kerroksiin. Alimmalla kerroksella sijaitsevat reunasolmut ovat lähimpänä asiakaslaitetta, mutta niiden sisältämät laskentaresurssit ovat vähäisiä. Ideana on että alemmalla kerroksessa sijaitsevat reunasolmut voivat siirtää laskentaa ylemmällä kerroksella sijaitsevalle reunasolmulle, jolla on enemmän resursseja. Tasojen määrälle ei ole mitään rajaa, mutta useimmat ehdotukset sisältävät kaksi tai kolme tasoa. Kuvassa 4 b) esimerkki kolmetasoisesta hierarkiasta. Ylemmällä tasolla sijaitsevalla reunasolmulla on vastuullaan useampi alemman tason reunasolmu. Hierarkkinen rakenne on siis puun mallinen.

Hierarkkisen rakenteen keskeisenä tavoitteena on reunajärjestelmän resurssien käyttöasteen parantaminen [Tong et al., 2016]. Ylemmillä kerroksilla sijaitsevat resurssit ovat suuremman joukon käytössä, hieman kuten litteässä mallissa tukiasemia klusteroitaessa. Hierarkkisessa rakenteessa on kuitenkin riskinsä. Järjestelmän voidaan olettaa monimutkaistuvan jos laskentaa tehdään useassa kerroksessa. Ja etenkin järjestelmän ylempien kerroksien etäisyys asiakaslaitteisiin kasvaa, joka voi potentiaalisesti johtaa viiveiden kasvuun ja palvelun laadun heikkenemiseen.

Tong et al. esittävät tutkimuksessaan vertailua litteälle ja erilaisille hierarkkisille rakenteille [Tong et al., 2016]. Tilanteessa jossa järjestelmään on jaettavissa kiinteä määrä resursseja, kolmeen kerrokseen jaettu järjestelmä vaikutti optimaalisimmalta. Vertailun mittarina käytettiin aikaa joka reunapalvelulla kului vastata annettuihin laskennallisiin tehtäviin. Kolmikerroksisen mallin etuina oli laskentaresurssien määrä ylemmillä kerroksilla. Tällöin etenkin ruuhkaisissa tilanteissa se suoriutui tehtävistään nopeammin kuin litteä. Lisäksi tutkimuksessa todettiin, että ylemmillä tasoilla sijaitseva tehokkaampi laskenta pystyy kompensoimaan siirtämisestä aiheutuvaa viivettä. Huomattakoon kuitenkin että tutkimuksessa käytettävä kuorma oli tyypiltään hienojakoista ja pientä.

5.0.3 Yhteenveto

Hierarkkinen järjestelmä on potentiaalinen vaihtoehto litteälle rakenteelle. Koska reunalaskennalle on monia hyvin erilaisia sovelluskohteita, hierarkkinen rakenne vaatii tarkempaa tutkimusta erilaisilla reunalaskennan tyypeillä. Litteän rakenteen heikkoina puolina voidaan pitää sen kyvyttömyyttä sopeutua kuorman epätasaisuuteen, sekä järjestelmän ylläpidon kustannuksia, mikäli järjestelmää päätetään hajauttaa aggressiivisesti. Litteä järjestelmä

on rakenteeltaan yksinkertainen ja se tukee esimerkiksi olemassa olevaa mobiiliverkon tukiasemien rakennetta.

5.1 Arkkitehtuurin vaikutus

Edellisessä kappaleessa järjestelmän rakenteen oletettiin olevan vapaa sijoittamaan reunasolmut haluamiinsa paikkoihin. Todellisuudessa reuna-arkkitehtuurit kuitenkin asettavat joitakin rajoitteita reunasolmujen sijaintien suhteen. Seuraavaksi käsitellään reuna-arkkitehtuurin vaikutusta reunasolmujen sijoitteluun.

Tutkielmaan valitut arkkitehtuurit sitovat jo lähtökohtaisesti reunajärjestelmää osaksi mobiiliverkkoa. Integraation tyypillä on tässäkin yhteydessä vaikutus siihen, kuinka paljon reunajärjestelmän rakenne on sidoksissa mobiiliverkkoon. Tämän lisäksi kommunikaatioväylien toteutus vaikuttaa rakenteeseen. Arkkitehtuureissa joiden kommunikaatioväylät eivät nojaa mobiiliverkon sisällä tapahtuvaan reititykseen ovat vapaampia sijoittamaan reunasolmuja. Hallinnolliset toimijat ovat, joko mobiiliverkkoon integroituina tai mobiiliverkon ulkopuolella. Hallinnollisien entiteettien sijaintiin vaikuttaa pääasiassa se, onko niillä rajapintaa mobiiliverkon sisäisten toimijoiden kanssa. Mikäli on, niin ne sijaitsevat loogisesti osana mobiiliverkkoa. Koska reunasolmujen sijoittuminen on hyvin riippuvaista arkkitehtuuristaehdotuksesta, reunasolmujen sijoittumista käsitellään seuraavassa kappaleessa jokaisen arkkitehtuuriehdotuksen yhteydessä.

Reunanajärjestelmän riippuvuutta mobiiliverkkoon voidaan yrittää pienentää, esimerkiksi ottamalla SDN käyttöön. SDN mahdollistaa tietoliikennevirtojen ohjaamisen siten, että reunajärjestelmän tietoliikennevirrat eivät vaikuta mobiiliverkon toimintaan. Tämän seurauksena reunajärjestelmä on täysin vapaa sijoittamaan reunasolmut haluamiinsa sijainteihi. Keskeinen viiveeseen vaikuttava ratkaisu on SDN verkon sijoittuminen mobiiliverkon hierarkiassa.

Toinen ehdotettu ratkaisu on liittä reunasolmut tukiasemiin. Tämä pakottaa järjestelmälle litteän rakenteen. Tällainen järjestelmä on siis reunasolmujen osalta sidoksissa mobiiliverkon rakenteeseen. Small Cell Cloud (käsitellään kappaleessa 6.3) esittää tämänkaltaista ratkaisua. Ideana on että tukiasemien tarjoamat solut ovat pieniä, jolloin tukiaseman kantama olisi pieni ja täten palveltavien asiakaslaitteiden määrä myös vähäinen. Tällöin tukiasemaan tarvittavien reunalaskentaresurssien määrä on myös vähäisempi, verrattuna tilanteeseen jossa tukiasemalla olisi enemmän palveltavia.

Kolmas ja viimeinen tyyppi on kokonaan mobiiliverkon ulkopuolelle sijoitetut reunasolmut. Tämä antaa käytännössä täydellisen vapauden sijoitella reunasolmuja. Esimerkki tämänkaltaisesta arkkitehtuurista on Follow Me Cloud (käsitellään kappaleessa 6.2). Reunajärjestelmän sijoittaminen mobiiliverkon ulkopuolelle tarkoittaa, että asiakaslaitteen ja reunasolmun välinen viive on vähintään se aika joka tietoliikenteellä kuluu mobiiliverkossa. Fol-

low Me Cloudin tapauksessa oletetaankin, että mobiiliverkon aiheuttamaa viivettä voitaisiin pienentää ja että reunasolmut sijaitsisivat välittömästi mobiiliverkon ulkopuolella.

6 Esitetyt ratkaisut

Reunalaskennan toteuttamiseksi on esitetty useita erilaisia ratkaisuja. Sen sijaan että esitettäisiin ratkaisu yksittäiselle toiminnolle, reunalaskenta-arkkitehtuuri pyrkii toteuttamaan tarpeeksi suuren osajoukon toteuttamisen kannalta kriittisistä ominaisuuksista. Seuraavaksi käydään läpi ehdotettuja arkkitehtuureja ja lopuksi tarkastellaan ETSI:n dokumentaatioiden esittämä reunalaskennan referenssiarkkitehtuuri ja reunalaskennalle asetetut vaatimukset.

6.1 Cloudlet

Cloudlet on virtuaalikoneita hyödyntävä reuna-alusta ratkaisu [Satyanarayanan et al., 2009]. Tarkemmin ottaen cloudlet on enemmänkin arkkitehtuurielementti kuin kokonainen arkkitehtuuri. Muihin tässä tutkielmassa käsiteltäviin arkkitehtuuriehdotuksiin verrattuna cloudlet onkin hieman suppeamman laajuinen konsepti. Sen merkitys reunalaskennan ilmentymisessä on kuitenkin niin huomattava että se käsitellään omana kokonaisuutenaan tässä tutkielmassa. Seuraavaksi käsitellään cloudletin perustoiminnallisuutta sekä toimintaperiaatteita.

Cloudletin toiminta perustuu käyttäjäkohtaisten virtuaalikoneiden suorittamiseen reunasolmulla. Virtuaalikoneiden käyttö tuo järjestelmään useita haluttuja ominaisuuksia. Koska cloudletin tapauksessa reunalaskennan oletetaan olevan ainoastaan hetkellistä tai väliaikaista[Satyanarayanan et al., 2009], on virtuaalikoneinstanssien käyttö luonnollista. Virtuaalikoneet tarjoavat erinomaisen tavan jakaa suoritusalustan resursseja käyttäjäkohtaisille instansseille. Koska virtuaalikoneet eivät ole tiukasti sidoksissa suoritusalustaan, voidaan niitä ainakin teoriassa siirrellä helposti. Tietoturva näkökulmasta virtuaalikoneet tarjoavat myös vahvan eristyksen jokaiselle virtuaalikoneinstanssille. [Ha et al., 2015] Lisäksi virtuaalikone ei ota kantaa suoritettavaan ohjelmistoon, jolloin se tarjoaa valinnanvapauden ohjelmointikielten ja käyttöjärjestelmän suhteen

Vaikka virtuaalikoneet ovat helposti alustalta toiselle siirrettävässä muodossa, on otettava huomioon reunakonteksti. Tavallinen virtuaalikoneen tilavaatimus muodostuu virtuaalisesta kovalevystä, muistin kopiosta, prosessorin tilatiedosta ja metatiedoista. Virtuaalikoneiden koot vaihtelevat, mutta esimerkkitapauksissa Ha et al ja Satyanarayanan et al käyttämien virtuaalikoneiden suuruus oli 8GB kovalevyä ja 1GB RAM [Ha et al., 2013, Satyanarayanan et al., 2009]. Reunaympäristössä tämän kokoisten tiedostojen siirtely aiheuttaisi varmasti ruuhkaa ja odottelua. Tämän kokoisten virtuaalikoneiden siirtely vaatii siis aikaa ja kaistaa. Myöskään näin suurten

tiedostojen tallentaminen ei ole erityisen mielekäs vaihtoehto. Alkuperäisen cloudlet ehdotuksen keskeinen sisältö onkin optimoida reunalaskennassa käytettävien virtuaalikoneiden kokoa siten että niiden tallentaminen, sekä siirtäminen olisi mielekästä [Satyanarayanan et al., 2009].

Cloudletin keskeinen oivallus on käyttää virtuaalikoneissa yhteistä pohjaa (base) ja erottaa pohjasta käytön aikana muuttuneet osat erilliseksi pinnoitteeksi (VM overlay). Pohjalla tarkoitetaan virtuaalikoneen lähtötilaa, joka voidaan uudelleen käyttää useamman käyttäjän virtuaalikoneiden käyttöönotossa. Käytön aikana pohjana toimineeseen virtuaalikoneeseen tulee muutoksia käyttäjän toiminnan mukaan, esimerkiksi asennettujen sovelluksien osalta. Näiden käyttäjän tekemien muutoksien seurauksena tallennustilan ja muistin tila eroaa joiltain osin pohjan mukaisesta lähtötilasta. Kun käyttäjä lopettaa virtuaalikoneen käytön, cloudlet järjestelmä alkaa muodostamaan käyttäjäkohtaista pinnoitetta. Yksinkertaistettuna pinnoite tehdään siten, että käyttäjän virtuaalikonetta verrataan pohjana toimineeseen virtuaalikone-kuvaan. Tämän jälkeen muuttuneet osat kerätään omaan tiedostoonsa ja tämä tiedosto pakataan pienemmän tiedostokoon saavuttamiseksi. Satayanaran et al ensimmäisessä cloudletteja käsittelevässä julkaisussa [Satyanarayanan et al., 2009] virtuaalikoneen muutoksien tallentaminen pinnoitteeksi tuottaa noin 100-200 megatavun kokoisen tiedoston Suhteessa lähtökohtana toimineeseen virtuaalikoneeseen, pinnoite on tiedostokooltaan kertaluokkaa pienempi. [Satyanarayanan et al., 2009]

Alkuperäisen idean mukaan pinnoitteen erottamisen jälkeen, pinnoite voitaisiin siirtää joko pilveen tai asiakaslaitteelle odottamaan seuraavaa käyttökertaa. Ideaa on kuitenkin jatkojalostettu tukemaan live migraatiota, joka käsitellään tämän kappaleen loppupuolella.

Kun käyttäjä haluaa jälleen ottaa virtuaalikoneensa käyttöön alkaa niin sanottu dynaaminen virtuaalikone synteesi (Dynamic VM synthesis), jossa pohja ja pinnoite yhdistetään suorituskelpoiseksi virtuaalikoneeksi. Pinnoitteen sisältämät muutokset palautetaan virtuaalikonepohjaan jotta se vastaisi tilaa johon käyttäjä oli sen jättänyt. Pohjan luomiseen, pinnoitteen erottamiseen ja dynaamiseen virtuaalikone synteesiin liittyy huomattava määrä yksityiskohtia joita ei käsitellä tässä tutkielmassa. Niiden sisältämistä yksityiskohdista on mahdollista lukea lisää [Ha et al., 2013] tutkielmasta.

Tiedostokooltaan pienempi virtuaalikoneen tallennusmuoto ei kuitenkaan tule ilman kustannuksia. Pinnoitteen erottaminen ja pakkaaminen vaativat huomattavan määrän laskentaresursseja. Alkuperäisessä ideassa pinnoitteen luominen oli prosessi joka voitiin tehdä ilman aikarajoitetta. Kun järjestelmään lisättiin VM handoff, siihen kuluva aika muuttui merkittäväksi tekijäksi. Myös dynaaminen virtuaalikoneen synteesi vaatii resursseja. Jaetussa suoritusympäristössä korkea yleisrasite näkyy muiden samalla laitteistolla tarjottavien palveluiden laadun heikkenemisessä, tässä tapauksessa siis muiden virtuaalikoneiden. Lisäksi cloudlet ympäristössä virtuaalikoneen käynnistäminen vaatii virtuaalikoneen syntetisoinnin pohjasta ja pinnoit-

teesta, joka luonnollisesti vie myös aikaa, eikä virtuaalikone käynnisty yhtä nopeasti kuin perinteinen virtuaalikone.

Cloudletin toiminnallisuuden edellytyksenä on joukko pohja-aihioita, joita käyttäjät voivat ottaa käyttöön. Sujuvan toiminnan takaamiseksi reunasolmulla tulisi olla tarvittavat pohjat jo valmiiksi. Eräs toimintamalli on säilöä reunasolmulla ainoastaan suosituimpien pohjien joukko ja tarpeen mukaan puuttuvia pohjia voitaisiin ladata pilvestä lisää.

VM handoff

Cloudlet ympäristössä suoritusaikaista virtuaalikoneen siirtoa kutsutaan termillä *VM handoff* [Ha et al., 2015]. Cloudletiltä toiselle tehtävä live migraatio muistuttaa toiminnaltaan mobiiliverkon tukiasemien välistä handoffia, mutta sovellettuna virtuaalikoneisiin. Kappaleessa 4.1 esitettyjen live migraation metriikkojen osalta VM handoff eroaa palvelinsalista. VM handoff tavoittelee mahdollisimman lyhyttä migraation kestoa, verrattuna palvelinsalien mahdollisimman lyhyeen katkon kestoon. Syyksi tälle esitetään että asiakaslaitteen ja virtuaalikoneen heikentyneen yhteyden korjaaminen on korkeammalla prioriteetilla kuin varsinaisen katkon keston lyhentäminen [Ha et al., 2015]. Toisin kuin palvelinsaliympäristössä olevat lähiverkkoyhteydet, cloudlettien välinen yhteys ei välttämättä ole kovin nopea. Perinteisen live migraation iteratiivinen toiminta ole hitailla yhteyksillä kannattavaa. Viimeinen ero tavallisen live migraation ja VM handoffin välillä on cloudletin pohjaan ja pinnoitteeseen perustuva toimintamalli.

Cloudletit voivat hyödyntää yhteistä pohjaa VM handoffin yhteydessä, jolloin siirrettävän tiedon määrä on vähäisempi. Etenkin hitailla yhteyksillä siirrettävän tiedon määrän minimointi on keskeisiä tavoitteita. VM handoffin tehokkuutta on vertailtu tavalliseen live migraatioon Ha et al tutkimuksessa [Ha et al., 2017]. Siinä todettiin että etenkin hitailla yhteyksillä VM handoff saavutti merkittävästi lyhyemmän migraation keston ja selvisi lähes kaikissa tapauksissa huomattavasti pienemmällä tiedonsiirrolla. VM handoffin aiheuttaman katkon kesto oli hieman pidempi verrattuna live migraatioon. VM handoff on kuitenkin erittäin resurssi-intensiivinen toimenpide, eikä tutkimuksessa otettu tarkemmin kantaa sen vaikutukseen reunasolmun muuhun toiminnallisuuteen. Tutkimuksessa ei oteta kantaa useamman virtuaalikoneen siirtäminen samanaikaisesti.

Yhteenveto

Cloudlet kattaa reuna-arkkitehtuurista vain pienen osan, joka keskittyy yksittäisellä reunasolmulla suoritettavaan reuna-alustaan. Cloudletin keskeinen toiminnallisuus sisältää virtuaalikoneita hyödyntävän toimintaympäristön joka tukee laskennan suoritusaikaista siirtoa cloudletiltä toiselle VM handoff tekniikalla. Cloudlet jättää toimintaympäristön avoimeksi eikä siis sitou-

du esimerkiksi mobiiliverkkoihin. Cloudlet ei myöskään esitä keinoja, joilla tietoliikenne ohjataan virtuaalikoneelle tai kuinka yhteys siirrettyyn virtuaalikoneeseen säilytetään. Näistä syistä johtuen cloudlettiä voidaan pitää hyvin suppeana reunalaskennan ratkaisuna, joka vaatii ympärilleen hallinnollisia toimintoja, kuten seuraavaksi käsiteltävät reuna-arkkitehtuurit.

6.2 Follow me cloud

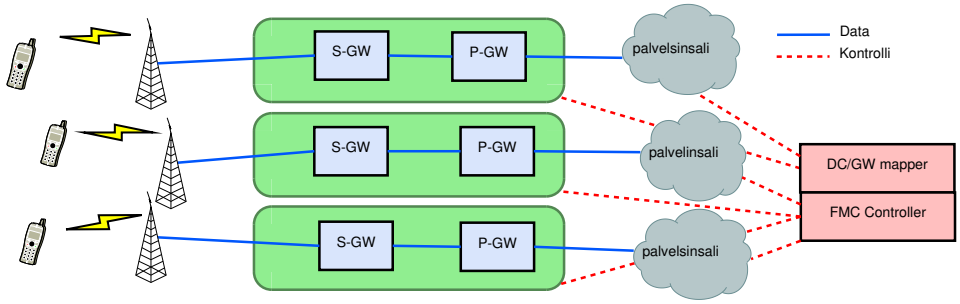
Asiakaslaitteiden määrän kasvu sekä tietoliikenne-intensiivisien sovelluksien käytön yleistyminen aiheuttaa suurta kuormaa verkkoinfrastruktuurille [Taleb and Ksentini, 2013]. Ongelmat johtuvat pääasiassa mobiiliverkon keskitetystä rakenteesta, jossa tietoliikenneyhteydet kerätään kulkemaan keskitetyn pisteen kautta [Taleb and Ksentini, 2013]. Tämän lisäksi yhteyksissä on huomattavasti viivettä, koska asiakkaan ja palvelimen väliset yhteydet ovat pitkiä. Follow me cloud (FMC) ehdottaa ratkaisua, jossa operaattorin keskitetty verkkorakenne muutettaisiin hajautetuksi [Taleb and Ksentini, 2013]. Käytännössä tämä tarkoittaisi, että EPC:n ja ulkoverkon välisten yhdyskäytävien, eli P-GW instanssien, määrää lisättäisiin. Toinen osa ideaa on, että P-GW:t voitaisiin hajauttaa palvelinsaleihin lähemmäksi asiakkaan käyttämiä palveluita. Tämä siksi, että myös mobiiliverkkojen operaattorit ovat huomanneet hajautustarpeen ja alkaneet hajauttaa mobiiliverkkoja [Taleb and Ksentini, 2013]. FMC:n arkkitehtuuri edellyttää hajautettua EPC:tä ja hajautettuja palvelinkeskuksia.

Asiakaslaitteen yhdistäessä verkkoon, verkkoyhteyksien kiintopisteenä toimii todennäköisimmin lähimpänä sijaitseva P-GW. Perinteisessä mobiiliverkossa keskitetystä rakenteesta johtuen, asiakaslaite pysyy yhden P-GW:n asiakkaana, vaikka liikkuisi mobiiliverkossa. FMC lähteekin ratkaisemaan ongelmaa jossa asiakaslaitteelle tarjottaisiin aina optimaalisin[3] yhteys tavoiteltuun palveluun.

FMC:n ratkaisemat ongelmat voidaan jakaa kahteen osaan: optimaalisen reitin valitseminen ja optimaalisen reitin ylläpitämiseen käyttäjän liikkuessa. Reitinvalinta jakautuu matkaan, jonka asiakkaan yhteys kulkee mobiiliverkossa, eli tarkemmin ottaen EPC:n sisällä, sekä matkaan P-GW:ltä kohdepalvelimeen. Tämän saavuttamiseksi mobiiliverkon arkkitehtuurin tarvitaan palvelinkeskuksien ja P-GW välisiä mappauksia tekevä entiteetti.

FMC tarjoaa optimaalisen reitin löytämiseen ratkaisua, jossa mobiiliverkon läheisyyteen lisättäisiin hallinnolliset entiteetit FMC controller ja DC/GW mapper. Lisäksi ehdotuksessa mobiiliverkkoon lisätään tunnisteet, joiden avulla asiakaslaitteen sijainti saadaan selville asiakaslaitteen käyttämästä yhteydestä. DC/GW mapper tehtävä on valita asiakkaan yhteyttä parhaiten vastaava palvelinsali. DC/GW mapperin paritus tehdään esimerkiksi yhdistämällä tietyn P-GW:n (voidaan päätellä esimerkiksi IP-vyöhykkeen

[3]optimaalisella tarkoitetaan tässä yhteydessä parasta mahdollista, käyttäen mittareina fyysistä sijaintia ja kuormaa



Kuva 5: Follow me cloud arkkitehtuuri, jossa mobiiliverkko on hajautettu useampaan yhdyskäytävään

avulla) kautta tulevat yhteydet ennalta määrättyyn palvelinsaliin. Kuvassa 5 asiakaslaitteet ovat yhdistyneenä oman S-GW/P-GW-parin kautta, kyseistä yhdyskäytävää vastaavaan palvelinsaliin.

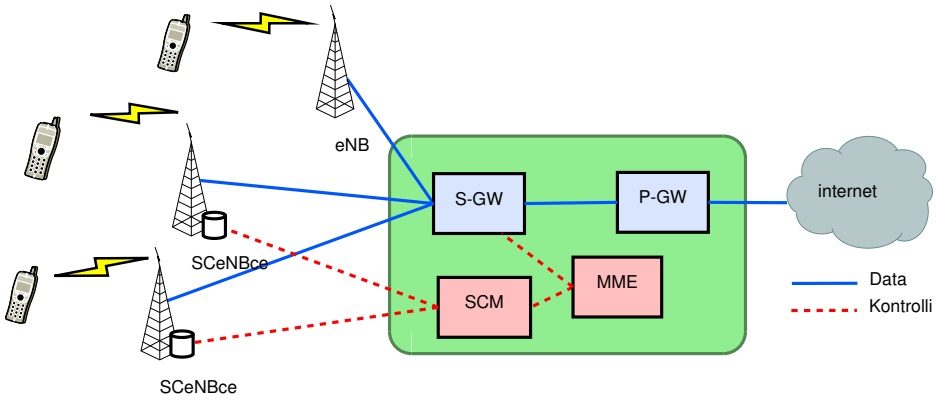
Toinen osa ratkaisua on asiakaslaite–palvelinsali-yhteyden ylläpitäminen. Yhteyden ylläpitäminen asiakkaan liikkuessa vaatii jonkin verran muutoksia tapaan, jolla käyttäjän siirtymiseen mobiiliverkossa reagoidaan. Hajautetussa mobiiliverkossa tutkiaseman vaihtuessa myös tietoliikenne yhteyskäytävänä toimiva P-GW saattaa vaihtua. Tämän seurauksena asiakaslaitteen IP-osoite vaihtuu ja yhteys reunapalveluun katkeaa. FMC on esittänyt että asiakas ja reunapalvelu yhdistettäisiin erillisen tunnisteen perusteella, joka generoidaan asiakastunnisteen ja palvelutunnisteen pohjalta. Tämä uusi tunniste korvaa IP-osoitteet ja mahdollistaan yhteyksien uudelleenmuodostuksen tilanteissa joissa asiakaslaitteen tai reunapalvelun IP-osoitteet vaihtuvat.

FMC controllerin tehtävänä on laukaista palvelun siirtäminen optimaalisempaan palvelinsaliin esimerkiksi asiakaslaitteen yhteysmuutoksien seurauksena. FMC controller tekee päättelyä siitä kannattaako reunapalvelua siirtää.

FMC eroaa muista mobiiliverkon ratkaisuista sillä, että se ei tuo palvelinresursseja osaksi mobiiliverkkoa. Lisäksi FMC ei tee mobiiliverkon infrastruktuuriin minkäänlaisia muutoksia. FMC pyrkii takaamaan nopeimman mahdollisen yhteyden haluttuun palveluun. FMC jättääkin auki mahdollisuuden, että reunapalvelut tarjoaa joku muu kuin operaattori itse. Koska reunapalvelimet sijaitsevat mobiiliverkon ulkopuolella FMC ei edellytä muutoksia myöskään mobiiliverkon sisäiseen reititykseen.

6.3 Small Cell Cloud

Small Cell Cloud (SCC) on reuna-arkkitehtuuri ehdotus LTE-tyyppiseen mobiiliverkkoon. Idean premisseinä toimivat laskentaresurssien lisääminen mobiiliverkon tukiasemiin ja mobiiliverkon solujen pieneneminen. Näillä toimilla voitaisiin vastata tietoliikenteen määrän kasvuun sekä tarjota reuna-



Kuva 6: SCC arkkitehtuuri, jossa SCM on integroitu osaksi EPC:tä [Lobillo et al., 2014]

laskentaa uutena palveluna. SCC:n ratkaisuympäristönä toimii LTE verkko, jonka kehitystä ehdotettu ratkaisu pyrkii myötäilemään. Pienemmät solut tarkoittavat että tukiasemat sijaitsevat lähempänä asiakasta, joka puolestaan implikoi asiakaslaitteille nopeampaa tiedonsiirtoväylää radiorajapinnassa [Lobillo et al., 2014]. Seuraavaksi esitellään SCC yleisellä tasolla, aloittaen läpikäynti toimijoista. Lopuksi käsitellään tietoliikenneratkaisuja, joita SCC:n yhteydessä on ehdotettu.

SCC:ssä laskentaresurssit on sijoitettu eNodeB tukiasemiin. SCC:n tapauksessa puhutaan SCeNBce:stä (Small-cell eNodeB computing-enhanced), eli laskentaresursseilla varustetusta piensolutukiasemasta [Lobillo et al., 2014]. Reunalaskentaan käytettävät resurssit ovat integroituina osaksi tukiasemaa [Puente et al., 2015]. Ajatuksena on että tukiaseman toiminnot suoritettaisiin samalla laitteistolla kuin reunasovellukset. Tämänkaltainen menettely mahdollistaa tukiaseman ja reuna-alustan yhteistoiminnallisuutta. Yhteistoiminta mahdollistaa esimerkiksi radiorajapinnasta saatavien tietojen hyödyntämisen osana etälaskennan kannattavuuden päättelyä. Lisäksi yhteisillä jaetuilla resursseilla vältyttäisiin erillisen laitteiston lisäämiseltä [Puente et al., 2015].

SCC:n hallinnollisista toimista vastaava entiteetti on Small Cell Manager (SCM) [Lobillo et al., 2014]. SCM on MME:n kaltainen itsenäinen toimija, mutta vastaa ainoastaan reunalaskentaan liittyvistä hallinnollisista toimista. Tarkoituksena on että SCM on tietoinen reunasolmujen resurssien tilasta ja on kykenevä tekemään päätöksiä reunasovelluksien siirtämisestä sijainnista toiseen. Lisäksi SCM:n tehtävänä on vastata muun muassa asiakaslaitteiden pyyntöihin laskentaresursseista osoittamalla käytettävissä oleva virtuaalikone [Dolezal et al., 2016]. SCM voi myös sähkön säästämiseksi sulkea SCeNBce:n reunalaskentatoiminnallisuuden. SCM:n sijoittelu mobiiliverkon sisällä on jätetty toteuttajan vastuulle, mutta ehdotettuja sijainteja ovat muun muassa MME:n yhteydessä ja täysin itsenäisenä toimijana [Lobillo et al., 2014]. SCM

toteutusvaihtoehdot sisältävät sekä hajautettuja että keskitettyjä malleja.

SCC ympäristössä perustoiminnallisuus menisi siten että reunalaskentaa tahtova asiakaslaite välittäisi pyynnön SCeNBce:lle joka edelleen välittäisi pyynnön SCM:lle. SCM toimii reuna-alustan roolissa ja valitsee SCeNBce:lle virtuaalikoneen, jolla reunalaskentaa voidaan suorittaa. [Dolezal et al., 2016]

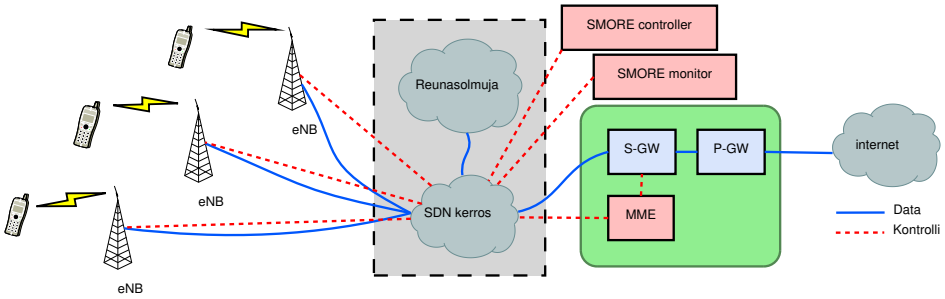
Koska SCC:n ehdotetaan olevan tiukasti integroituna LTE mobiiliverkkoon, vaatii se olemassa olevien rajapintojen lisäksi uuden rajapinnan SCM vaatimien toiminnallisuuksien toteuttamiseksi. SCM tarvitsee rajapinnan MME:hen jonka kautta se voi esimerkiksi autentikoida asiakaslaitteita [Lobillo et al., 2014]. Joskaan SCC:n käyttöönotto ei vaadi mobiiliverkon täysimittaista uudelleen rakentamista, edellyttää se toimiakseen ainakin olemassa olevien tukiasemien korvaamisen SCeNBce tyyppisillä tukiasemilla. Lisäksi käyttöönotto vaatii puuttuvien rajapintojen lisäämistä MME:n yhteyteen. Laskentaresurssien sijoittaminen tukiasemaan tekee resursseista hyvin paikallisesti hyödynnettäviä. Onkin siis tarkkaan pohdittava resurssien määrää tukiasemassa koska ne eivät ole kovin helposti muiden solujen hyödynnettävissä.

6.3.1 Kommunikointi reunapalveluun

Kuten kappaleessa 4.3 käsiteltiin, tavallisesti asiakaslaitteen tietoliikenne kulkee mobiiliverkon läpi koskemattomana GTP-tunnelin sisällä. SCC:n asiakaslaitteen ja reunalaskennan välisen tietoliikenteen reititys on esitelty julkaisussa [Puente et al., 2015], johon tämä kappale pohjautuu. SCC:n tapauksessa asiakaslaitteen liikenne haluttaisiin ohjata kohteesta riippuen joko tukiasemassa sijaitsevalle reunasolmulle tai normaalia reittiä pitkin ulkoverkkoon. SCC:ssä on päädytty lisäämään SCeNBce tukiasemaan tietoliikennettä monitoroiva toiminnallisuus.

Kommunikaatio asiakaslaitteen ja tukiaseman välillä suoritetaan DRB väylällä (Data Radio Bearer). Tukiasema identifioi asiakaslaitteet radiorajapinnassa DRB ID:n (Data Radio Bearer ID) avulla. Asiakaslaitteelta verkkoon suuntautuva liikenne paketoidaan tukiaseman toimesta GTP-tunneliin. Jokaisen GTP-tunnelin tunnisteena toimii TEID (Tunnel Endopoint ID). Asiakaslaitteen tietoliikenteen välitys tukiaseman sisällä vaatii siis DRB ID:n ja TEID:n sisältävän muunnostaulun käyttämistä. Tätä samaa periaatetta voi hyödyntää reunasolmulle suuntautuvan tietoliikenteen identifiointiin.

Tukiasemaan lisättävän monitoritoiminnallisuus tarkkailee asiakaslaitteen lähettämien pakettien kohde IP-osoitetta, ja mikäli kyseessä on reunapalvelun IP-osoite, monitori ohjaa paketin reunapalvelulle. Monitorointi tehdään ennen GTP-tunnelointia. Samalla paketista voidaan poimia asiakaslaitteen IP-osoite, joka kirjataan muunnostauluun asiakasta vastaavan TEID:n ja DRB ID:n kanssa. Toiseen suuntaan tietoliikenne toimii siten, että reunapalvelun asiakaslaitteelle lähettämästä tietoliikenteestä poimitaan kohde IP. Kohde IP:tä vastaava DRB ID poimitaan muunnostaulusta ja reunapalvelulta tulevat paketit muutetaan DRB väylälle sopiviksi. Tällaisen



Kuva 7: Yksittäinen SMORE instanssi kuvattuna. Välissä sijaitseva harmaa alue kuvaa reunajärjestelmän sijaintia. Mukaelma julkaisussa [Cho et al., 2014] esitetystä kuvasta.

toiminnallisuuden toteuttamisen seurauksena kommunikaatio on asiakaslaitteen ja reunapalvelun näkökulmasta kuin mikä tahansa muukin IP-pohjainen kommunikaatio.

Asiakaslaiteen ja reunasolmun välisen kommunikaation säilymiseen, tilanteessa, jossa asiakaslaite siirtyy toiselle tukiasemalle, ei oteta kantaa. Tällaisen toiminnallisuuden tarve kuitenkin on, sillä SCM esittelyn yhteydessä mainittiin reunasovelluksien, eli virtuaalikoneiden, siirtelyn mahdollisuus.

6.4 SMORE ja MobiScud

SMORE (Software defined network Mobile Offloading aRchitecturE) on mobiiliverkkoihin suunnattu reunalaskentaratkaisu [Cho et al., 2014]. Keskeisin osa ratkaisua on SDN:n käyttöönotto mobiiliverkon sisäisissä yhteyksissä. SMORE:n tapauksessa reunajärjestelmä sijoittuu eNodeB ja S-GW välille. Kuvassa 7 on esitetty SMORE:n yleinen rakenne.

Ratkaisun kantavana ideana on SDN-kerros. Se sijaitsee alkuperäisen idean mukaan MTSO:ssa (Mobile Telephone Switching Office), joka on tukiasemien yhteyksin kokoumapiste. SDN-kerroksessa sijaitsee SMORE monitor, joka tarkkailee SDN-kerroksen läpi kulkevaa mobiiliverkon kontrollikerroksen liikennettä. SMORE monitor poimii tietoliikenteestä kontrollikerroksella esiintyviä tunnisteita kuten IMSI (International Mobile Subscriber Identity), GTP tunneleiden TEID tunnisteita, asiakaslaitteen IP, S-GW IP, eNodeB IP. Nämä tunnisteet tallennetaan tietokantaan. Näitä tietokantaan poimittuja tunnisteita käytetään SDN reittien muodostamiseen sekä mobiiliverkon tapahtumien tietojen välittämiseen SMORE controllerille. SMORE controller on vastuussa reunasovelluksien käynnistämisestä sekä SDN reitityssääntöjen päivittämisestä. SMORE controller voi tietokantaan tallennettujen tietojen perusteella muokata reitityssääntöjä, siten että asiakkaan tietoliikenne ohjautuu reunasolmuille.

Tietoliikenteen haarauttaminen tehdään SDN-kerroksessa virtuaalisten

porttien avulla. Asiakkaan mobiiliverkon sisäinen GTP tunneloitu liikenne ohjautuu tähän porttiin. Portin sisällä on toiminnallisuus joka tarkkailee GTP pakettien sisällä olevan asiakkaan tietoliikennepakettien kohde IP-osoitteita [Cho et al., 2014]. Mikäli kohde IP on reunasolmu, virtuaalisen portin sisällä oleva toiminnallisuus purkaa GTP tunneloinnin ja ohjaa paketin reunasolmulle. Reunasolmulta asiakaslaitteelle suuntautuva liikenne hoituu siten että SMORE monitorin poimimista tiedoista saadaan asiakaslaitteen IP-osoitetta vastaava GTP TEID. Tämän avulla voidaan suorittaa GTP paketointi ja paketti voidaan välittää normaalin toiminnallisuuden mukaisesti mobiiliverkon kautta asiakaslaitetta palvelevalle eNodeB:lle ja siitä edelleen asiakaslaitteelle.

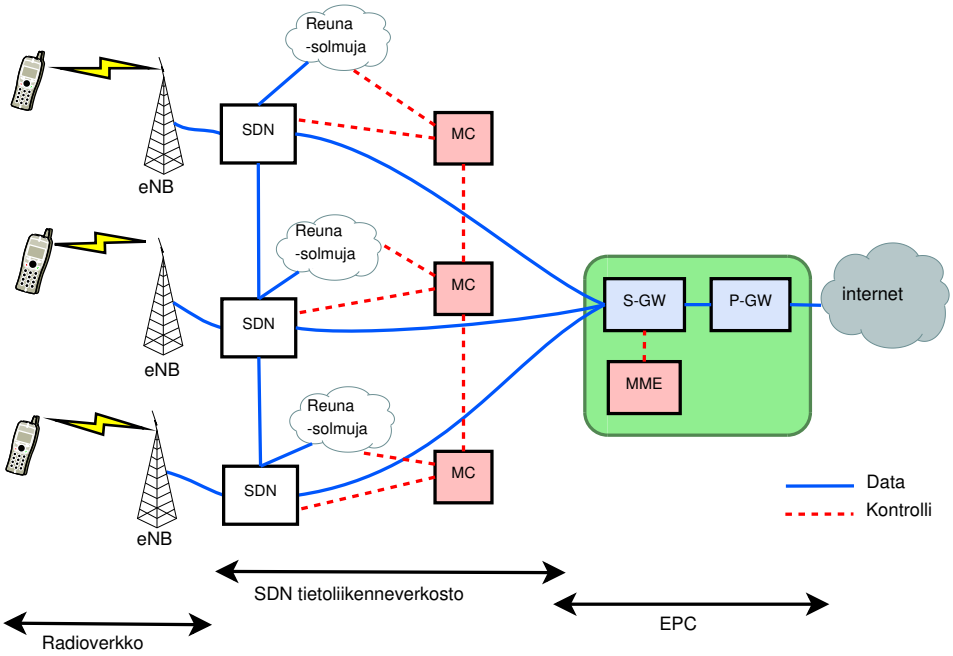
SMORE controllerin vastuulle kuuluu myös reunasolmujen tarjoamien palveluiden hallinnointi. Esimerkiksi asiakaslaitteen liikkuessa verkossa SMORE monitor päivittää tietokantaan tarvittavat tunnisteet ja huomauttaa SMORE controlleria tapahtumasta. SMORE controller reagoi tähän päivittämällä SDN reititykset. Tämä varmistaa sen että asiakaslaitteen yhteys reunapalvelulle säilyy. SMORE controller on keskitetty entiteetti ja se voi hallita useampaa SDN-kerrosta.

Koska SDN-kerros ei muokkaa mobiiliverkon sisäistä tai muuhun internettiin suuntautuvaan tietoliikennettä, SMORE-reunajärjestelmä on asiakaslaitteen ja mobiiliverkon näkökulmasta näkymätön. Koska asiakaslaitteella ei ole yhteyttä SMORE-järjestelmään, reunalaskentaa tarjotaan ehdotuksen mukaan kahdella tavalla. Ensimmäisessä tapauksessa reunalaskentaa voidaan pyytää ulkoisen palvelun kautta, joka pyytää SMORE controlleria ohjaamaan asiakaslaitteen tietoliikenteen reunapalveluun. SMORE controllerille oletetaan verkkoyhteys internettiin, jonka avulla reunapalveluita voidaan ottaa käyttöön esimerkiksi pilpipalvelusta käsin. Toinen ehdotettu vaihtoehto on että SMORE:n palveluita myydään tilauksesta ja SMORE:n omassa tietokannassa ylläpidetään niiden asiakkaiden tietoja joilla reunalaskenta on käytössä.

SMORE:n keskeinen ominaisuus on integraatio mobiiliverkkoon ilman että mobiiliverkkoon muuten tarvitsee tehdä muutoksia. Ainoana edellytyksenä on SDN-kerroksen lisääminen tukiasemien ja EPC:n välille. SMORE ei tarjoa reunalaskennan siirtelylle keinoja, koska reunaresurssien oletetaan sijaitsevan niin keskitettyinä että siirtelylle ei ole tarvetta.

6.4.1 MobiScud

MobiScud on SMORE:n ideoihin pohjautuva reunalaskenta-arkkitehtuuri [Wang et al., 2015]. MobiScud:n perustoiminnallisuus on hyvin samankaltainen kuin SMORE:ssa. MobiScud painottaa enemmän käyttäjän tarvetta liikkua verkossa ja SMORE:sta poiketen toteuttaa mekanismin reunalaskennan migraatiolle. MobiScud ottaa myös huomioon mobiiliverkon hajautumista koskevan trendin.



Kuva 8: MobiScud arkkitehtuuri. Mukaelma julkaisussa [Wang et al., 2015] esitettyyn kuvaan.

Kuten SMORE, myös MobiScud olettaa erillisen SDN-kerroksen toteuttamista. MobiScudin tapauksessa reunalaskentaresurssien oletetaan sijaitsevan tukiaseman välittömässä läheisyydessä. MobiScud olettaa reunalaskennan toteutusmetodiksi henkilökohtaiset virtuaalikoneet. Tämän lisäksi MobiScud olettaa hallinnollisen elementin olevan hajautettuna SDN kerroksiin. MobiScud Controller (MC) sisältää samat toiminnallisuudet kuin SMORE:n controller ja monitor. MC kuitenkin laajentaa toimintoja lisäämällä mahdollisuuden reunalaskennan live migraatiolle.

MobiScudin arkkitehtuuri on esitetty kuvassa 8. Kuvasta on jätetty pois eNodeB:n ja MME:n väliset kontrolliyhteydet kuvan yksinkertaistamiseksi. Reunajärjestelmän resurssien oletetaan sijaitsevan tukiasemien läheisyydessä. Toisistaan erillään olevat reunajärjestelmän osat pystyvät kommunikoimaan SDN verkon avulla toisilleen. Tämä muistuttaa loogisesti samaa järjestelmää kuin eNodeB-tukiasemien välillä olevat X2 yhteydet. MC on hajautettuna verkossa ja MC:n instanssit sijaitsevat reunasolmujen yhteydessä. MC instanssien hallinnollinen vastuu koskee omassa vaikutuspiirissä olevaa SDN verkkoa sekä reunasolmuja. Tämä tarkoittaa että MC hoitaa asiakaslaitteen ja reunasolmun väliset SDN reitiẗykset. Lisäksi MC on yhteydessä muihin MC instansseihin.

Reunalaskennalle suuntautuvan tietoliikenteen haarauttaminen on käytännössä toteutettu samoin kuin SMORE:ssa. SDN kerroksen sisällä tarkkaillaan

ja tarpeen mukaan puretaan GTP-tunnelointi paketeista, joiden kohteena on reunapalvelut. Reunapalveluilta asiakaslaitteelle suuntautuva tietoliikenne vastaavasti paketoidaa SDN kerroksessa GTP muotoon ja välitetään asiakaslaitteelle.

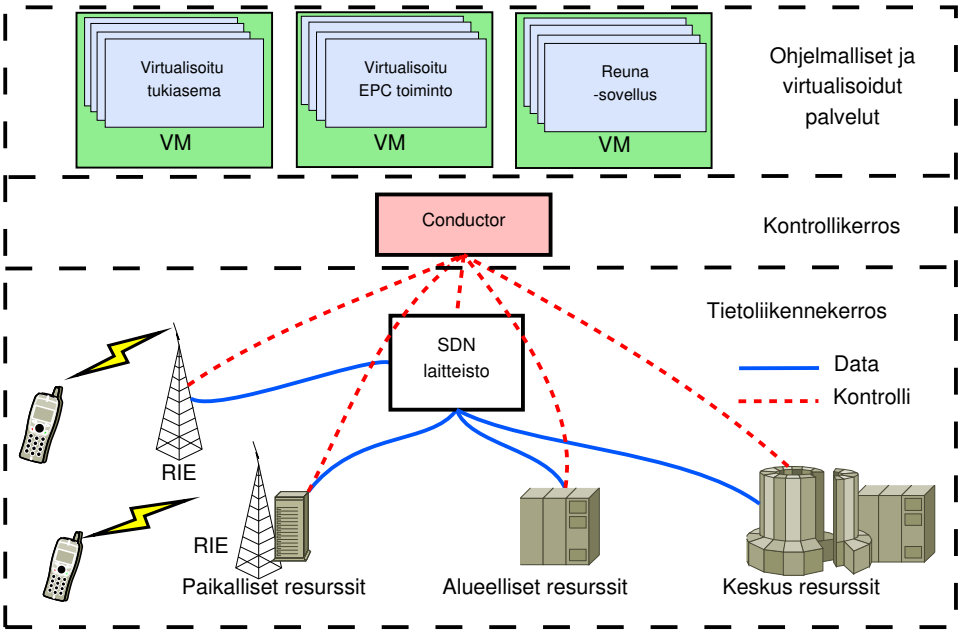
Reunalaskennan migraatiotoiminnallisuus MobiScud:ssa koostuu yhteyden säilyttämisestä ja virtuaalikoneen live migraatiosta. Reunalaskennan siirron laukaisevana tekijänä toimii MC:n monitoritoiminnallisuuden havaitsema eNodeB:n handoveriin liittyvä kontrollikerroksen viestintä. Ensimmäisessä vaiheessa varmistetaan asiakaslaitteen yhteys reunasolmulle. Se tehdään siten että ENodeB:n handoveriin liittyvästä viestinnästä poimitaan handoverin kohteena olevan eNodeB:n tiedot. Näihin sisältyy muun muassa uuden GTP-tunnelin TEID (kohteena olevan eNodeB:n ja S-GW:n välinen) ja kohteena olevan eNodeB:n IP. Lähde ja kohde MC:t muodostavat yhdessä näiden tietojen perusteella SDN reitityksen, jonka avulla reunasolmulle suuntautuva tietoliikenne kulkeutuu asiakkaan reunalaskennasta vastaavalle virtuaalikoneelle uuden tukiaseman kautta. Tästä seuraa yhteyden pituuden kasvua reunapalvelun ja asikaslaitteen välillä, joka ilmenee suurempana viiveenä. Osittain samaan aikaan SDN verkon reititysmuutoksien kanssa, aloitetaan virtuaalikoneen live migraatio lähempänä sijaitsevalle reunasolmulle. Kun migraatio valmistuu, SDN reititykset vanhaan reunasolmuun korvataan uusilla, jotka ohjaavat liikenteen reunasolmulle, jolle virtuaalikone migratoitiin.

Vastaavia SDN ja monitorointiin pohjautuvia reunalaskentaratkaisuja on ehdotettu myös muualla, esimerkiksi [Schiller et al., 2018].

6.5 CONCERT

CONCERT on virtualisointiin keskittyvä mobiiliverkon ja reunalaskennan arkkitehtuuri [Liu et al., 2014]. Nykyinen LTE-verkko kostuu joukosta erilaisia laitteita, joilla jokaisella on jokin spesifi tehtävä. CONCERT pyrkii virtualisoinnilla vähentämään tällaisten laitteiden määrää. Kyseessä on siis NFV:tä hyödyntävä ratkaisu. Ehdotuksen tavoitteena on että virtualisoinnille yleiset hyödyt pätisivät myös tässä. Tavoitteena on että toimintojen käyttöönotto olisi vähemmän riippuvainen fyysisiten laitteiden asentamisesta, toimintojen tuottamiseen käytettävät resurssit skaalautuisivat paremmin ja että resurssien käyttöaste olisi parempi. Näiden tavoitteiden ehtona on käytännössä kaikkien mobiiliverkon toimintojen muuntaminen ohjelmalliseen muotoon ja virtualisoida ne. Tämä pätee kaikkiin laitteisiin alkaen tukiasemista ja edeten aina EPC:n komponentteihin asti. Koska muutos on suuri, CONCERT:n tavoitteena on seuraavan sukupolven mobiiliverkko, eikä integraatio nykyisiin mobiiliverkkoihin. CONCERT:n idea on pääpiirteittäin yhdenmukainen ETSI:n NFV näkemyksen [ETSI, 2017b] kanssa, NFV:n hyödyntämisestä viidennen generaation mobiiliverkkojen yhteydessä.

Yleisen palvelinlaitteiston hyödyntäminen palveluiden tuottamiseen mahdollistaa useiden eri toimintojen sijoittamisen virtualisoituina samaaan fyy-



Kuva 9: CONCERT arkkitehtuuri. Mukaelma julkaisussa [Liu et al., 2014] esitetystä kuvasta

siseen laitteistoon. CONCERT ehdotuksen on tuottaa sekä mobiiliverkon toiminnallisuudet, että reunapalvelut samalla laitteistolla. Mobiiliverkon toimintojen virtualisointia on ehdotettu ennenkin. Esimerkiksi kappaleessa 3.3 käsiteltiin C-RAN tyyppistä ehdotusta. C-RAN tavoitteena on virtualisoida tukiaseman toiminnalliset osat, siten että ne voitaisiin keskittää virtualisoituina palvelinsaleihin. CONCERT:n mukaan tämänkaltainen ratkaisu johtaisi liiallisesti keskitettyyn ja fyysisesti kaukana sijaitsevaan toteutukseen, jossa reunalaskennan tai muiden lyhyistä viiveistä riippuvaisten palveluiden toteuttaminen ei onnistuisi [Liu et al., 2014]. Lisäksi CONCERT haluaisi tukiasemien toimintojen lisäksi virtualisoi nykyisen EPC:n alaiset toiminnot. CONCERT ehdottaa liiallisen keskittämisen välttämiseksi resurssien sijoittelua hierarkiseen malliin. Kolmetasoisen hierarkian resurssit on jaettu kolmeen tasoon: paikalliseen, alueelliseen ja keskus. Paikallisen tason resurssit olisivat kaikkein vähäisimmät ja ne sijatsisivat tukiasemien välittömässä yhteydessä. Aluellisen tason resurssit olisivat suuremmat kuin paikallisen ja sijaitsisivat kauempana. Keskus tason resurssit olisivat kaikkein suurimmat, mutta sijaitsisivat kaikkein kauimpana. Hierarkisen rakenteen ansiosta kaikkein tiukimman aikavaatimuksen sisältävä reunalaskenta voidaan suorittaa lähimmillä resursseilla ja muu laskenta välittää kauempana sijaitseville resursseille.

CONCERT jakaa järjestelmän kontrolli- ja tietoliikennekerrokseen (control/data plane). Tietoliikennekerros koostuu mobiiliverkon ja reunalaskennan

mahdollistavasta laitteistosta, joka sisältää palvelinresurssit, radioliikenteestä vastaavan laitteiston (Radio Interfacing Equipment, RIE) sekä SDN kytkimistä. Huomautettakoon että RIE vastaavanlainen toimija kuin C-RAN ehdotuksen RRH. Tietoliikennekerroksella on edellä kuvattu hierarkinen rakenne, jonka sisäisestä tietoliikenteen välittämisestä vastaa SDN. Kontrollikerroksen ainoana toimijana on conductor, joka vastaa CONCERTin hallinnollisista toimista. Reunalaskennan kannalta merkityksellisenä toimintona conductorissa on LCM (Locatio-Aware Computing Management). LCM vastaa reunalaskentaan liittyvien resurssien hallinnasta. Tämä sisältää esimerkiksi käytettävien reunaresurssien osoittamisen reunalaskennan aikavaatimuksien ja mahdollisten resurssi vaatiumuksien mukaan. Lisäksi kontrollikerroksen tehtävänä on lisäksi konfiguroida SDN reititykset siten että reunapalvelut ovat saavutettavissa.

CONCERTin arkkitehtuurikuvaus sisältää maininnan virtualisoitujen palveluiden dynaamisesta migratoinninsta, mutta ei määrittele sen tarkempaa kuvausta sen toiminnasta. Voidaan kuitenkin olettaa että se kattaa reunalaskennan siirtämistä esimerkiksi tilanteissa joissa käytössä olevan reunasolmun resurssit eivät riitä tai aikavaatimuksiin voitasiin vastata paremmin jollain toisella sijainnilla.

Virtualisoinnin voidaan sanoa olevan CONCERT:n ydin. CONCERT hyödyntää virtualisointia mobiiliverkon toimintojen tuottamiseen ja reunalaskennan tarjoamiseen. Tämän mahdollistaa yhtenäiset resurssit molemmille, sekä vähentää tarvetta erillisille laitteitoille. Hierarkkisen rakenteen avulla CONCERT tavoittelee parempaa resurssien käyttöastetta sekä reunapalveluiden vaatimien aikavaatimuksien täyttämistä. NFV, SDN ja mobiiliverkon toimintojen toteuttaminen ohjelmallisina ovat kaikki edellytyksiä CONCERT:n toteutumiseksi. Toistaiseksi kaikki näistä ovat vasta kehityksessä. Kyseessä on kokonaisuudessaan hyvin suuri muutos mobiiliverkon toimintaan, eikä ainoastaan vanhojen toiminnallisuuksien uudellentoteutus virtuaalisina.

6.6 ETSI MEC

ETSI (European Telecommunications Standards Institute) on eurooppalainen telealan standardointijärjestö. ETSI on aloittanut reunalaskennan arkkitehtuurin, sekä sen touteuttamiseksi vaadittavien toimintojen standardoimisen.

ETSIn MEC spesifikaatio määrittelee reunapalvelun tuottamiseksi vaadittavat ominaisuudet, jotka reunainfrastruktuurin tulee toteuttaa. Spesifikaatio listaa myös mahdollisia, mutta ei vaadittuja toimintoja. Listaamalla vaaditut toiminnot stanrardi pyrkii yhtenäistämään reunalaskennan konsepteja.

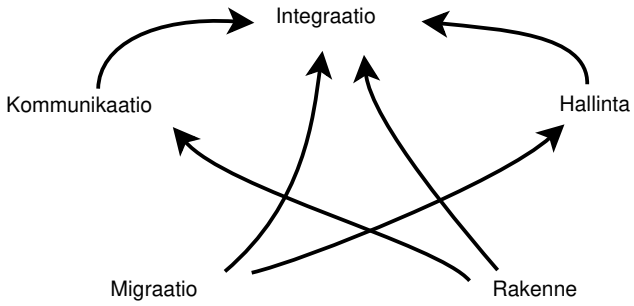
6.6.1 Vaatimukset

Vaatimukset on jaettu kategorioihin sen mukaan mihin toiminnallisuuteen vaatimus liittyy. Vaatimuksien kategoriat ovat yleiset vaatimukset (generic

requirements), palvelu vaatimukset (service requirements), hallinta vaatimukset (operation and management requirements) ja viimeisenä kategoriana on kokoelma vaatimuksista, joiden teemoina ovat turvallisuus, sääntely ja veloitus (Security, regulation, charging requirements)[ETSI, 2016c].

Yleiset vaatimukset ovat luonteelta korkean tason kuvauksia reuna-infrastruktuurin toiminnallisuuksista. Yleiset vaatimukset kategorisoitu seuraaviin luokkiin: viitemallista, reunapalvelun sovelluksien elinkaaresta (application lifecycle), reunapalvelun sovellusympäristöstä (application enviroment) sekä liikkuvuuden tuesta (support of mobility). Viitemallin tulee vaatimuksien mukaan hyödyntää mukaan NFV ratkaisuja hallinnollisten toimien toteuttamiseen, mikäli mahdollista. ETSI:n vaatimuksen mukaan reunapalvelun tulee voida sijaita käytännössä missä tahansa kohdassa radiomaston runkoverkon reunan välillä. Sijainnin vapaus myös tarkoittaa että reuna-alusta(?) ei voi olla riippuvainen alla olevasta infrastruktuurista. Reunapalvelun sovelluksien elinkaareen liittyvät vaatimukset koskevat pääasiassa reunapalvelun toimijoiden oikeuksia päättää reunan sovelluksien käynnistämisestä ja sulkemista. Reunapalveluiden sovellusympäristöä koskevissa vaatimuksissa esitetään, että sovelluksien autenttisuus ja eheys pitää pystyä varmentamaan. Sovellusympäristön täytyy myös mahdollistaa reunasovelluksen käyttöönotto toisella reuna-isännällä, ilman erikoisempaa mukautusta (without specific adaptation) [ETSI, 2016c]. Mobiiliverkkojen ollessa kyseessä, asiakaslaitteiden liikkuminen tukiasemalta toiselle, on keskeinen käyttötapaus. Tämä heijastuu myös vaatimukseen liikkuvuuden tukemisesta reunapalveluissa. Vaatimuksena on että asiakaslaitteen ja reunapalvelun välisen yhteyden tulee säilyä, vaikka asiakaslaite siirtyisi solusta toiseen tai asiakaslaite siirtyisi sellaiseen soluun joka on toisen reuna-isännän vastuualuetta.

Palveluvaatimukset on joukko vaatimuksia, jotka keskittyvät takaamaan reuna-infrastruktuurin perimmäiset palveluperiaatteet. Lista palveluvaatimuksista on pitkä, joten tähän tutkielmaan on poimittu ainoastaan osa vaatimuksista. Täydellinen lista löytyy [ETSI, 2016c] julkaisusta. Palveluvaatimukset kuvaavat toiminnallisuuksia, joiden avulla reunapalveluita voidaan tuottaa. Tällaisesta esimerkkinä tietoliikenteen reitittämiseen liittyvät vaatimukset, joiden keskeinen tehtävä on kuvata mahdolliset tietoliikennereitit reunapalveluun ja ulos reunapalvelusta. Yksi ehkä keskisimmistä palveluvaatimuksista on reuna-alusta mahdollisuus suodattaa ja muokata verkkoliikennettä. Lisäksi kuvataan että reunapalveluiden toimintaa ei haluta rajoittaa pelkästään asiakas-palvelu tyyppisen toimintamalliin. Reunapalveluiden tuottamiselle onkin annettu mikropalvelu -tyyppinen (microservice) kuvaus, jossa palveluntuottajat voivat toimia myös toisten reunapalveluiden kuluttajana (consumer).



Kuva 10: Reuna-arkkitehtuurien ominaisuuksien yleiset riippuvuudet

6.6.2 Viitekehys ja referenssiarkkitehtuuri

ETSIn esittämän viitekehys esittää reunalaskentaan liittyvät korkean tason entiteetit. Nämä entiteetit on jaettu kolmeen tasoon: järjestelmä, isäntä ja verkko(system, host ja network). Verkkokerros sisältää verkkoyhteyksistä vastaavat entiteetit. MEC:n tapauksessa verkkokerros koostuu ainakin kolmesta osasta: sisäverkko, ulkoverkko ja televerkko. Isäntäkerros koostuu reunapalveluiden virtualisointiin ja reunapalvelun hallinointiin keskittyvistä entiteeteistä. Järjestelmäkerros koostuu korkeamman tason hallinnosta vastaavista elementeistä.

Referenssiarkkitehtuurissa on esitetty funktionaaliset entiteetit. Funktionaalisten entiteettien toiminta on kuvattu Referenssiarkkitehtuurissa reunaisännällä (edge-host) tarkoitetaan entiteettiä, joka tarjoaa virtualisointiinfrastruktuurin, sekä reunalaskennan toteuttamiseen vaadittavat resurssit (laskenta, tallennus ja verkko). Reuna-alustalla (Mobile edge platform) tarkoiteen sitä entiteettiä joka mahdollistaa reunapalveluiden käyttämisen. Reuna-alusta siis mahdollistaa reunapalveluiden saavuttamisen, eli käytännössä tarjoaa rajapinnan asiakaslaitteen suuntaan. Tähän kuuluu siis palvelurekisterin ylläpitäminen, reitityssääntöjen ylläpitäminen, sekä liikenteen välittäminen reunapalveluille. Reuna-alusta voi myös itse tarjota palveluita. Esimerkkinä tällaisesta voisi olla tunnistautumispalvelu. Reuna-applikaatioilla tarkoitetaan reuna-alustalla suoritettavia virtuaalikoneita, jotka suorittavat reunapalveluiden tuottamiseksi tarkoitettuja ohjelmistoja.

6.7 Vertailu

Seuraavaksi käydään läpi edellä esitettyjen reuna-arkkitehtuuriehdotuksien ominaisuuksia ja käsitellään niiden vaikutuksia itse toteutettavaan järjestelmään. Kunkin arkkitehtuuriehdotuksen ominaisuusjoukko ohjaa toteutusta erilaisiin ratkaisuihin. Taulukoon 1 on tiivistetysti koottu kunkin arkkitehtuuriratkaisun ominaisuudet. Lopuksi käsitellään reuna-arkkitehtuurien yhteneväisyyksiä ja eroja ETSI:n MEC spesifikaation kanssa.

On tärkeää huomata että käsiteltyjen ominaisuuksien välillä on riippuvuk-

sia. Tämän seurauksena jonkin ominaisuuden toteuttaminen tietyllä tavalla saattaa estää tai rajoittaa joidenkin toiminnallisuuksien toteuttamista.

Reuna-arkkitehtuurien keskeisin ominaisuus on tapa jolla järjestelmä integroituu osaksi mobiiliverkkoa. Kaikki muut ominaisuudet vaikuttavat olevan riippuvaisia tästä. Kuvassa 10 on esitetty ominaisuuksien väliset riippuvuudet. Rakenne on tyypiltään implisiittinen ominaisuus. Se ei siis pääsääntöisesti ole reuna-arkkitehtuurin määrittelemä, vaan sen toteutus on riippuvainen reuna-arkkitehtuurin muista ominaisuuksista.

Kappaleessa 4.2 esiteltyjen integraatiotyyppien jaon mukaan SCC ja CONCERT edustavat suoraa integraatiota, SMORE ja MobiScud edustavat läpinäkyvää integraatiota ja FMC edustaa epäsuoraa integraatiota.

SCC liittyy mobiiliverkon toimintoihin niin sanotusti natiivien yhteyksien avulla. Se edellyttää uusia rajapintoja mobiiliverkon nykyisiin komponentteihin. Tavoitteena on SCC:n hallinnollisten tarpeiden täyttämiseen. Lisäksi SCC edellyttää reunaresurssien integrointia osaksi tukiasemaa. Tukiasemaan sidottujen reunaresurssien vuoksi SCC:n rakenne on litteä. SCC:n mukaan uudistetun tukiaseman, eli SCeNBce:n, tehtäviin kuuluu reunapalveluille tarkoitetun tietoliikenteen ohjaaminen. Tämä tarkoittaa että SCeNBce monitoroi tietoliikennettä ja ohjaa esimerkiksi kohde IP:n perusteella reunapalvelulle tarkoitetut paketit.

CONCERT on toinen suoran integraation reunajärjestelmä. Ehdotuksen tavoitteena on NFV:n laajamittainen käyttöönotto, jonka seurauksena mobiiliverkon ja reunajärjestelmä voisivat jakaa laskentaresursseja. Järjestelmä rakentuu kolmeen kerrokseen hierarkisesti sijoitettujen resurssien varaan. Nämä resurssit ovat reunalaskennan ja mobiiliverkon toiminnallisuuksien käytettävissä. Mobiiliverkon ja reunalaskennan hallinnolliset toimet on yhdistetty Conductor nimiselle entiteetille. Conductor sisältää useita alitoimintoja joiden tehtäviin kuuluu muun muassa resurssien jakaminen kullekkin toiminnallisuudelle. CONCERT ottaa hallinnollisiin tehtäviin kantaa ainoastaan yleisellä tasolla ja yksityiskohdat jätetään avoimeksi. Oletettavaa on että CONCERT:ssa mobiiliverkon ja reunajärjestelmän reititys tapahtuu samassa kontekstissa, jolloin järjestelmä ei vaadi erillistä toiminnallisuutta reunasolmuille suuntautuvan tietoliikenteen ohjaamiseen.

MobiScud ja SMORE edustavat läpinäkyvää integraatiota. SDN-kerros eNodeB ja EPC:n komponenttien välillä mahdollistaa tietoliikenteen monitoroinnin sekä ohjaamisen. SDN-kerros mahdollistaa mobiiliverkon nykyisten toimintojen säilyttämisen ennallaan. Koska tietoliikenne reunaresursseille ohjataan SDN-kerroksen sisällä, reunasolmujen sijoittelu ei ole suoranaisesti riippuvaista mobiiliverkosta. Teoriassa reunasolmujen sijoittelu on täysin vapaa. SMORE ja MobiScud ehdottavat toisistaan poikkeavaa sijaintia reunasolmuille. SMORE:ssa yhdelle reunasolmulle klusteroidaan suuri joukko tukiasemia, kun taas MobiScud ehdottaa reunasolmujen sijoittelua yksittäisten tukiasemien yhteyteen. Tämä tarkoittaa että järjestelmät eroavat yksittäiseen reunasolmuun sijoitettujen laskentaresurssien määrältä huomat-

tavasti. SMORE:n tapauksessa yhden reunasolmun vastuualueeksi oletetaan niin suuri joukko tukiasemia, että laskennan siirrolle ei ole tarvetta, eli järjestelmä ei toteuta live migraatiota. MobiScud puolestaan hajautetummilla resursseillaan edellyttää reunalaskennan siirtoa ja ehdottaakin sille toiminnallisuutta. SMORE:n ja MobiScud:n yhteisenä ominaisuutena on mekaniikka, jolla tietoliikenne ohjataan reunasolmuille siten että se ei vaikuta mobiiliverkon toimintaan. Molemmissa ratkaisuissa SDN-kerros sisältää toiminnallisuuden, joka haarauttaa reunasolmuille tarkoitettua liikennettä monitoroimalla mobiiliverkon GTP liikennettä. Tarpeen mukaan GTP tunneloitujen pakettien GTP paketointi puretaan ja välitetään SDN-kerroksen sisällä reunasolmulle. Samaa monitorointitoiminnallisuutta käytetään molemmissa ratkaisuissa myös mobiiliverkon kontrollitason seuraamiseen, joka mahdollistaa mobiiliverkon tapahtumiin reagoinnin. MobiScud perustaa reunalaskentaan siirtoon liittyvät päätökset näihin mobiiliverkon tapahtumien kautta saatuihin tietoihin.

Ainoa epäsuoraa integraatiota edustava ratkaisu on FMC. FMC:n tavoitteena on viedä mobiiliverkon yhteydet nopeammin ulkoverkossa sijaitseville palvelinresursseille. Mobiiliverkon ulkopuolelle sijoitetut resurssit mahdollistavat sen että reunajärjestelmää voi ylläpitää jokin ulkoinen taho. FMC ei siis suoranaisesti ota kantaa tapaan jolla reunaresurssit on järjestetty. Mutta on perusteltua olettaa että ne sijaitsisivat mobiiliverkon P-GW komponenttien läheisyydessä. FMC:n tapa mahdollistaa reunapalvelun ja asiakaslaitteen välinen kommunikaatio perustuu sessiotunnistesiin. Se mahdollistaa asiakaslaitteen ja reunpalvelun IP-osoitteiden vaihtumisen, ilman että viitteet asiakaslaitteen ja reunapalvelun välillä katkeavat. Järjestelmä ei siis edellytä tavallisesta poikkeavaa mekaniikkaa tietoliikenteen reitittämiseksi. Ainoa edellytys on asiakaslaitteen sovellukseen sekä reunapalvelun toiminnallisuuden ohjelmallisen toimijan joka huolehtii yhteyksien varmistamisesta sessiotunnisteiden avulla.

Myös hallinnollisten toimien toteuttaminen on osittain riippuvaista integraatiosta. Suoran integraation järjestelmissä hallinnollisella entiteetillä on oma rajapinta mobiiliverkon toimintojen kanssa. Toisaalta se edellyttää olemassa olevaan järjestelmään rajapinnan toteuttamisen, mutta lisäksi tarjoaa helpon keinon tehdä hallinnollisia toimia mobiiliverkon sisällä. Läpinäkyvässä järjestelmässä hallinnollinen entiteetti on riippuvainen mobiiliverkon viestien monitoroinnin kautta saatavasta tiedosta. MobiScud ja SMORE tarkkailevat SDN-kerroksen avulla mobiiliverkon kontrollitason viestintää. Vaikka läpinäkyvät integraatiot ovat loogisesti erillään mobiiliverkosta, edellyttävät ne pääsyä mobiiliverkon sisäisiin tietoliikenneväyliin. Täten todennäköisintä on että järjestelmää ylläpitää mobiiliverkon operaattori eikä ulkoinen taho. Epäsuoraa integraatiota edustavassa FMC:ssä hallinnolliset toiminnot edellyttävät tietoja mobiiliverkosta. Tieto on tyypiltään staattista, joten aktiivista tiedosiirtoa mobiiliverkon ja FMC:n välille ei tarvita. Edellytyksenä on esimerkiksi taulukko, josta voidaan lukea asiakaslaitteen IP-osoittetta

vastaava fyysinen alue tai P-GW:n sijainti. Tämä mahdollistaa FMC:n ohjata asiakaslaitteen yhteys lähimmälle palvelinsalille.

SMORE, MobiScud ja FMC esittelevät toiminnot, joilla asiakaslaitteen liikkumisesta seuraavat yhteysmuutokset otetaan huomioon reunajärjestelmässä. Varsinaiseen reunapalvelun siirtämiseen oletetaan live migraation kaltainen mekanismi lähes kaikissa ehdotuksissa. Ainoastaan SMORE olettaa reunasolmun sijaitsevat niin keskeisessä yhteyspisteessä että reunapalveluiden siirtelylle ei ole tarvetta. Tällä arkkitehtuuritasolla ei ole kannattavaa ottaa kantaa reuna-alustan tehtäviin, koska se rajoittaisi arkkitehtuurin sovellettavuutta erilaisten reunapalveluiden tarjoamiseen.

Yhteensoppivuus ETSI MEC vaatimuksiin

ETSI MEC spesifikaation teknisiä vaatimuksia esittelevä dokumentti [ETSI, 2016c] määrittelee reunalaskennalle yleiset periaatteet ja konkreettisemmin määriteltyjä toiminnallisia vaatimuksia. Yleiset periaatteet ovat korkean tason tavoitteita reunajärjestelmälle. Reuna-arkkitehtuurien kannalta merkityksellisimmät periaatteet ovat NFV yhteensopivuus, liikkuvuuden tukeminen ja käyttöönoton riippumattomuus. Toiminnallisuudet vaatimukset on jaettu useampaan kategoriaan, jotka esiteltiin kappaleessa 6.6.1. Ainoastaan osa toiminnallisista vaatimuksista on relevantteja arkkitehtuureja tarkasteltaessa ja suurin osa näistä vaatimuksista koskee reuna-alustan ohjelmallisia toimijoita, joihin ei reuna-arkkitehtuureissa oteta kantaa. Lisäksi suurin osa vaatimuksista on ainoastaan täsmennyksiä yleisiin periaatteisiin. Seuraavaksi käydään läpi yleisten periaatteiden toteutuminen ehdotetuissa reuna-arkkitehtuureissa.

Tässä tukielmassa käsiteltyjen reuna-arkkitehtuurien yhteydessä ei esiintynyt NFV yhteensopivuutta rajoittavia tekijöitä. Eli kaikki käsitellyt arkkitehtuurit olivat pääsääntöisesti NFV yhteensopivia. CONCERT oli ainoa reuna-arkkitehtuuri joka nimenomaisesti rakentui NFV:n varaan. Muissa arkkitehtuureissa NFV:tä pidettiin enemmänkin mahdollisuutena. Esimerkiksi SCC:n yhteydessä esitettiin, että tukiasemaan sidottuja resursseja voisi käyttää myös tukiasemien omien toimintojen tuottamiseen NFV:n avulla.

Perinteisesti mobiiliverkoissa liikkuvuuden tukeminen tarkoittaa, että handover tapahtumat eivät ilmene asiakaskerroksella, esimerkiksi puheluissa ja tietoliikenneyhteyksissä. Samaa palveluvaatimusta on luonnollista olettaa myös reunajärjestelmältä. Käytännössä reunajärjestelmän tulee vähintää ylläpitää asiakaslaitteen ja reunapalvelun välistä yhteyttä. Tämän lisäksi palvelun laadun takaamiseksi kyseeseen tulee myös reunapalveluiden siirto. Liikkuvuuden tukeminen reunajärjestelmissä koostuu siis yhteyden ylläpidosta ja reunapalveluiden siirtelystä. Tässä tukielmassa SMORE, MobiScud ja FMC esittivät mekanismin yhteyden ylläpitämiseksi asiakaslaitteen liikkuessa verkossa. SMORE ja MobiScud toteuttivat yhteyden ylläpitämisen SDN reititysmuutoksilla. FMC puolestaan nojaa erilliseen sessiotunnisteeseen, joka häivyttää IP-osoitteiden muutokset. Lähes kaikki reuna-arkkitehtuurit

tiedostivat tarpeen reunapalveluiden siirtelyn tarpeelle. SMORE:a lukuunottamatta kaikki arkkitehtuurit olettivat virtuaalikoneiden live migraatiota jossain muodossa. SMORE oletti että reunasolmut sijaitsevat verkossa niin keskitetyssä pisteessä, että varsinaista reunasovelluksien live migraatiota ei ole tarpeen toteuttaa. ETSI:n MEC referenssi arkkitehtuuri olettaa reunajärjestelmän toteuttavan Cloudlet VM handoff kaltaista toiminnallisuutta [ETSI, 2016b]. On kuitenkin täysin reunapalveluiden tyypistä riippuvaista, kuinka liikkuvuuteen on kannattavaa reagoida [ETSI, 2016c].

Käyttöönoton riippumattomuudella tarkoitetaan, että reunajärjestelmän käyttöönottajalla olisi mahdollisuus sijoitella reunajärjestelmän osat omien vaatimuksiensa mukaan [ETSI, 2016c]. Pääasiassa tämä koskee reunasolmujen sijoittelua. Tässä tutkielmassa reunasolmujen sijoittelua on käsitelty implisiittisenä ominaisuutena, jonka määrittelevinä ominaisuuksina toimivat integraation tyyppi ja kommunikaatio. Teoriassa eniten vapauksia reunajärjestelmän toteuttamiseen tarjoavat läpinäkyvää integraatiota toteuttavat SMORE ja MobiScud. Nämä järjestelmät edellyttävät ainoastaan yhden yhteyspisteen mobiiliverkkoon ja ovat muuten täysin vapaasti järjesteltävissä. Suoran integraation reuna-arkkitehtuureissa edellytetään jotain tiettyä rakennetta. SCC edellyttää reunaresurssien sijoittamisen tukiasemiin. SCC kuitenkin loogisesti erottaa reunajärjestelmän ja mobiiliverkon haarauttamalla reunapalveluille tarkoitetun tietoliikenteen erillisellä monitoritoiminnallisuudella. Toinen suoraa integraatiota edustava järjestelmä, CONCERT, edellyttää hierarkista asettelua reunaresursseille. CONCERT:n NFV painotteisuus kuitenkin takaa toiminnallisuuksien vapaan sijoittelun. CONCERT tarjoaa SCC:hen verrattuna enemmän vapauksia resurssien sijoitteluun. FMC, ainoana epäsuorana itengraationa, tarjoaa myös vapauden sijoitella reunajärjestelmän, mutta ainoastaan mobiiliverkon ulkopuolelle. FMC:tä ei siis voi pitää erityisen riippumattomana, koska reunapalveluita tarjoavat resurssit joudutaan sijoittelemaan pakettiverkon yhdyskäytävien läheisyyteen.

Taulukko 1: Reunalaskenta-arkkitehtuurien ominaisuudet tiivistetysti

Ominaisuus	**Integraatio**	**Rakenne**	**Migraatio**	**Hallinta**	**Kommunikaatio**
FMC	Epäsuora	Vapaa	Palveluiden siirto ulkopuolisten salien välillä	FMC Controller	Tavalliset reitityksen, palveluiden ja asiakaslaitteen yhdistämiseen erillinen sessiotunniste
SMORE	Läpinäkyvä	Vapaa	Ei sisällä	SMORE Controller	SDN monitori ja reititys
MobiScud	Läpinäkyvä	Vapaa	Live migraatio	MobiScud Controller	SDN monitori ja reititys
SCC	Suora	Litteä	Live migraatio	SCM	Monitori ja reititys tukiasemassa
CONCERT	Suora	Hierarkinen	Live migraatio	Conductor	SDN reititys mobiiliverkossa

7 Yhteenveto

Reunalaskennan tarjoaminen mobiiliverkon kautta vaatii kahden eri järjestelmän yhteistoiminnallisuutta. Olemassa oleva mobiiliverkko on hajautettu ja suljettu järjestelmä. Tämän seurauksena siihen tehtävien muutoksien tulee olla tarkkaan harkittuja. Tässä tutkielmassa esiteltiin viisi erilaista reuna-arkkitehtuuriehdotusta, joista jokainen sisälsi uniikin joukon reunajärjestelmän ominaisuuksia.

Tämä tutkielma identifioi reuna-arkkitehtuuriehdotuksista viisi ominaisuutta. Nämä ominaisuudet edustavat jonkin suunnittelupäätöksen vaikutusta johonkin reunalaskennan mahdollistavaan toimintoon tai toimijaan. Tunnistetut ominaisuudet olivat live migraatio, integraation tyyppi, kommunikaatio, hallinta ja rakenne. Rakenne oli muista ominaisuuksista poikkeava, koska se määräytyy implisiittisesti muiden arkkitehtuurissa tehtyjen päätöksien pohjalta.

Ominaisuuksien joukossa ilmeni selkeitä riippuvuksia ja riippuvuuksien perusteella keskeisimmäksi arkkitehtuurissa tehtäväksi päätökseksi todettiin integraation tyyppi. Reunajärjestelmään toteuttamiseen käytettävällä integraation tyypillä on keskeinen vaikutus kaikkien muiden ominaisuuksien toteuttamiseen. Etenkin reunajärjestelmää toteutettaessa integraation tyyppi vaikuttaa siihen, kuinka paljon olemassa olevaa mobiiliverkkoa voidaan hyödyntää tai paljonko sitä tulee uusia.

Tukielman lopussa käsiteltiin reuna-arkkitehtuuriehdotuksien yhteensopivuutta ETSI MEC spesifikaation vaatimuksiin. Esitetyt arkkitehtuurit olivat pääsääntöisesti yhteensopivia, mutta parhaimmat lähtökohdat vaikuttivat olevan läpinäkyvää integraatiota edustavilla arkkitehtuureilla. Jokaisessa arkkitehtuurissa kuitenkin on omat kompromissinsa ja ehdotettujen arkkitehtuurien vertailu vaatisi yhdenmukaistettuja testejä. Koska reunalaskennan keskeisenä muuttujana on asiakaslaitteen ja reunapalvelun välinen viive, ei pelkät ohjelmalliset simulaatiot välttämättä riitä antamaan tarpeeksi tarkkaa kuvaa järjestelmän toiminnasta. Etenkin NFV toiminnallisuuksiin nojautuvien ratkaisujen osalta kysymykseen tulee NFV:llä toteutettujen toimintojen suorituskyky.

Lähteet

[Armbrust et al., 2010] Armbrust, M., Fox, A., Griffith, R., Joseph, A. D., Katz, R., Konwinski, A., Lee, G., Patterson, D., Rabkin, A., Stoica, I., et al. (2010). A view of cloud computing. *Communications of the ACM*, 53(4):50–58.

[Bonomi et al., 2012] Bonomi, F., Milito, R., Zhu, J., and Addepalli, S. (2012). Fog computing and its role in the internet of things. In *Proceedings*

of the first edition of the MCC workshop on Mobile cloud computing, pages 13–16. ACM.

[Bouet and Conan, 2017] Bouet, M. and Conan, V. (2017). Geo-partitioning of mec resources. page 43–48, New York, NY, USA. ACM.

[Chih-Lin et al., 2014] Chih-Lin, I., Huang, J., Duan, R., Cui, C., Jiang, J. X., and Li, L. (2014). Recent progress on c-ran centralization and cloudification. *IEEE Access*, 2:1030–1039.

[Cho et al., 2014] Cho, J., Nguyen, B., Banerjee, A., Ricci, R., Van der Merwe, J., and Webb, K. (2014). Smore: software-defined networking mobile offloading architecture. In *Proceedings of the 4th workshop on All things cellular: operations, applications, & challenges*, pages 21–26. ACM.

[Clark et al., 2005] Clark, C., Fraser, K., Hand, S., Hansen, J. G., Jul, E., Limpach, C., Pratt, I., and Warfield, A. (2005). Live migration of virtual machines. In *Proceedings of the 2nd Conference on Symposium on Networked Systems Design & Implementation-Volume 2*, pages 273–286. USENIX Association.

[Dolezal et al., 2016] Dolezal, J., Becvar, Z., and Zeman, T. (2016). Performance evaluation of computation offloading from mobile device to the edge of mobile network. In *Standards for Communications and Networking (CSCN), 2016 IEEE Conference on*, pages 1–7. IEEE.

[ETSI, 2012] ETSI (2012). Network functions virtualisation. `https://portal.etsi.org/NFV/NFV_White_Paper.pdf`. [Verkkoaineisto, Luettu 2018-03-19].

[ETSI, 2016a] ETSI (2016a). Lte; evolved universal terrestrial radio access (e-utra) and evolved universal terrestrial radio access network (e-utran); overall description; stage 2. `http://www.etsi.org/deliver/etsi_ts/136300_136399/136300/14.05.00_60/ts_136300v140500p.pdf`. [Verkkoaineisto, Luettu 2018-03-13].

[ETSI, 2016b] ETSI (2016b). Mobile edge computing (mec); framework and reference architecture. `http://www.etsi.org/deliver/etsi_gs/MEC/001_099/003/01.01.01_60/gs_MEC003v010101p.pdf`. [Verkkoaineisto, Luettu 2018-02-23].

[ETSI, 2016c] ETSI (2016c). Mobile edge computing (mec); technical requirements. `http://www.etsi.org/deliver/etsi_gs/MEC/001_099/002/01.01.01_60/gs_MEC002v010101p.pdf`. [Verkkoaineisto, Luettu 2018-02-23].

[ETSI, 2017a] ETSI (2017a). Lte; evolved universal terrestrial radio access network (e-utran); architecture description (3gpp ts 36.401 version 14.0.0

release 14). http://www.etsi.org/deliver/etsi_ts/136400_136499/136401/14.00.00_60/ts_136401v140000p.pdf. [Verkkoaineisto, Luettu 2018-03-14].

[ETSI, 2017b] ETSI (2017b). Network functions virtualisation. http://portal.etsi.org/NFV/NFV_White_Paper_5G.pdf. [Verkkoaineisto, Luettu 2018-03-19].

[Farris et al., 2017] Farris, I., Taleb, T., Iera, A., and Flinck, H. (2017). Lightweight service replication for ultra-short latency applications in mobile edge networks. In *Communications (ICC), 2017 IEEE International Conference on*, pages 1–6. IEEE.

[Firmin, 2017] Firmin, F. (2017). The evolved packet core. http://www.3gpp.org/technologies/keywords-acronyms/100-the-evolved-packet-core. [Verkkoaineisto, Luettu 2018-03-14].

[Garcia Lopez et al., 2015] Garcia Lopez, P., Montresor, A., Epema, D., Datta, A., Higashino, T., Iamnitchi, A., Barcellos, M., Felber, P., and Riviere, E. (2015). Edge-centric computing: Vision and challenges. *SIGCOMM Comput. Commun. Rev.*, 45(5):37–42.

[Gusev and Dustdar, 2018] Gusev, M. and Dustdar, S. (2018). Going back to the roots—the evolution of edge computing, an iot perspective. *IEEE Internet Computing*, 22(2):5–15.

[Ha et al., 2015] Ha, K., Abe, Y., Chen, Z., Hu, W., Amos, B., Pillai, P., and Satyanarayanan, M. (2015). Adaptive vm handoff across cloudlets. *Technical report, Technical Report CMU-C S-15–113, CMU School of Computer Science.*

[Ha et al., 2017] Ha, K., Abe, Y., Eiszler, T., Chen, Z., Hu, W., Amos, B., Upadhyaya, R., Pillai, P., and Satyanarayanan, M. (2017). You can teach elephants to dance: agile vm handoff for edge computing. In *Proceedings of the Second ACM/IEEE Symposium on Edge Computing*, page 12. ACM.

[Ha et al., 2013] Ha, K., Pillai, P., Richter, W., Abe, Y., and Satyanarayanan, M. (2013). Just-in-time provisioning for cyber foraging. In *Proceeding of the 11th annual international conference on Mobile systems, applications, and services*, pages 153–166. ACM.

[Heinonen et al., 2014] Heinonen, J., Partti, T., Kallio, M., Lappalainen, K., Flinck, H., and Hillo, J. (2014). Dynamic tunnel switching for sdn-based cellular core networks. In *Proceedings of the 4th workshop on All things cellular: operations, applications, & challenges*, pages 27–32. ACM.

[Jammal et al., 2014] Jammal, M., Singh, T., Shami, A., Asal, R., and Li, Y. (2014). Software defined networking: State of the art and research challenges. *Computer Networks*, 72:74–98.

[Kreutz et al., 2015] Kreutz, D., Ramos, F. M., Verissimo, P. E., Rothenberg, C. E., Azodolmolky, S., and Uhlig, S. (2015). Software-defined networking: A comprehensive survey. *Proceedings of the IEEE*, 103(1):14–76.

[Liu et al., 2014] Liu, J., Zhao, T., Zhou, S., Cheng, Y., and Niu, Z. (2014). Concert: a cloud-based architecture for next-generation cellular systems. *IEEE Wireless Communications*, 21(6):14–22.

[Lobillo et al., 2014] Lobillo, F., Becvar, Z., Puente, M. A., Mach, P., Presti, F. L., Gambetti, F., Goldhamer, M., Vidal, J., Widiawan, A. K., and Calvanesse, E. (2014). An architecture for mobile computation offloading on cloud-enabled lte small cells. In *2014 IEEE Wireless Communications and Networking Conference Workshops (WCNCW)*, pages 1–6.

[Mach and Becvar, 2017] Mach, P. and Becvar, Z. (2017). Mobile edge computing: A survey on architecture and computation offloading. *IEEE Communications Surveys & Tutorials*, 19(3):1628–1656.

[Malandrino et al., 2016] Malandrino, F., Kirkpatrick, S., and Chiasserini, C.-F. (2016). How close to the edge?: Delay/utilization trends in mec. In *Proceedings of the 2016 ACM Workshop on Cloud-Assisted Networking*, pages 37–42. ACM.

[Mao et al., 2017] Mao, Y., You, C., Zhang, J., Huang, K., and Letaief, K. B. (2017). A survey on mobile edge computing: The communication perspective. *IEEE Communications Surveys Tutorials*, PP(99):1–1.

[Nguyen and Cheriet, 2016] Nguyen, K. K. and Cheriet, M. (2016). Virtual edge-based smart community network management. *IEEE Internet Computing*, 20(6):32–41.

[Nohrborg, 2017] Nohrborg, M. (2017). Lte overview. `http://www.3gpp.org/technologies/keywords-acronyms/98-lte`. [Verkkoaineisto, Luettu 2018-04-20].

[Puente et al., 2015] Puente, M. A., Becvar, Z., Rohlik, M., Lobillo, F., and Strinati, E. C. (2015). A seamless integration of computationally-enhanced base stations into mobile networks towards 5g. In *2015 IEEE 81st Vehicular Technology Conference (VTC Spring)*, pages 1–5.

[Samdanis et al., 2012a] Samdanis, K., Taleb, T., and Schmid, S. (2012a). Traffic offload enhancements for eutran. *IEEE Communications Surveys & Tutorials*, 14(3):884–896.

[Samdanis et al., 2012b] Samdanis, K., Taleb, T., and Schmid, S. (2012b). Traffic offload enhancements for eutran. *IEEE Communications Surveys & Tutorials*, 14(3):884–896.

[Satyanarayanan, 1996] Satyanarayanan, M. (1996). Fundamental challenges in mobile computing. In *Proceedings of the Fifteenth Annual ACM Symposium on Principles of Distributed Computing*, PODC '96, pages 1–7, New York, NY, USA. ACM.

[Satyanarayanan, 2001] Satyanarayanan, M. (2001). Pervasive computing: vision and challenges. *IEEE Personal Communications*, 8(4):10–17.

[Satyanarayanan, 2017] Satyanarayanan, M. (2017). The emergence of edge computing. *Computer*, 50(1):30–39.

[Satyanarayanan et al., 2009] Satyanarayanan, M., Bahl, P., Caceres, R., and Davies, N. (2009). The case for vm-based cloudlets in mobile computing. *IEEE Pervasive Computing*, 8(4):14–23.

[Schiller et al., 2018] Schiller, E., Nikaein, N., Kalogeiton, E., Gasparyan, M., and Braun, T. (2018). Cds-mec: Nfv/sdn-based application management for mec in 5g systems. *Computer Networks*, 135:96 – 107.

[Soltesz et al., 2007] Soltesz, S., Pötzl, H., Fiuczynski, M. E., Bavier, A., and Peterson, L. (2007). Container-based operating system virtualization: A scalable, high-performance alternative to hypervisors. In *Proceedings of the 2Nd ACM SIGOPS/EuroSys European Conference on Computer Systems 2007*, EuroSys '07, pages 275–287, New York, NY, USA. ACM.

[Soni and Kalra, 2013] Soni, G. and Kalra, M. (2013). Comparative study of live virtual machine migration techniques in cloud. *International Journal of Computer Applications*, 84(14).

[Taleb and Ksentini, 2013] Taleb, T. and Ksentini, A. (2013). Follow me cloud: interworking federated clouds and distributed mobile networks. *IEEE Network*, 27(5):12–19.

[Taleb et al., 2017] Taleb, T., Samdanis, K., Mada, B., Flinck, H., Dutta, S., and Sabella, D. (2017). On multi-access edge computing: A survey of the emerging 5g network edge cloud architecture and orchestration. *IEEE Communications Surveys & Tutorials*, 19(3):1657–1681.

[Tong et al., 2016] Tong, L., Li, Y., and Gao, W. (2016). A hierarchical edge cloud architecture for mobile computing. In *INFOCOM 2016-The 35th Annual IEEE International Conference on Computer Communications, IEEE*, pages 1–9. IEEE.

[Tuovinen, 2017a] Tuovinen, A.-P. (2017a). Luentokalvot: Ohjelmistoarkkitehtuurin suunnittelu.

[Tuovinen, 2017b] Tuovinen, A.-P. (2017b). Luentokalvot: Ohjelmistoarkkitehtuurin suunnitteluperiaatteita.

[Wang et al., 2015] Wang, K., Shen, M., Cho, J., Banerjee, A., Van der Merwe, J., and Webb, K. (2015). Mobiscud: A fast moving personal cloud in the mobile network. In *Proceedings of the 5th Workshop on All Things Cellular: Operations, Applications and Challenges*, pages 19–24. ACM.

[Xavier et al., 2013] Xavier, M. G., Neves, M. V., Rossi, F. D., Ferreto, T. C., Lange, T., and De Rose, C. A. (2013). Performance evaluation of container-based virtualization for high performance computing environments. In *Parallel, Distributed and Network-Based Processing (PDP), 2013 21st Euromicro International Conference on*, pages 233–240. IEEE.

[Yousaf and Taleb, 2016] Yousaf, F. Z. and Taleb, T. (2016). Fine-grained resource-aware virtual network function management for 5g carrier cloud. *IEEE Network*, 30(2):110–115.