Liittymä Kansalliseen palveluväylään: kartoitus, dokumentointi ja kehitys

Mikko Korhonen

2020 Laurea

Laurea-ammattikorkeakoulu

Liittymä Kansalliseen palveluväylään: kartoitus, dokumentointi ja kehitys

Mikko Korhonen

Tietojenkäsittelyn tradenomi

Opinnäytetyö

Kesäkuu, 2020

Laurea-ammattikorkeakoulu **Tiivistelmä**
Tietojenkäsittelyn
Tradenomi (AMK)

Mikko Korhonen
Liittymä Kansalliseen palveluväylään: kartoitus, dokumentointi ja kehitys
Vuosi 2020 Sivumäärä 42

Tämän opinnäytetyön tavoitteena oli tutkia ja dokumentoida toimeksiantajana toimivan Digia Oyj:n ylläpitämä integraatio liityntäpalvelinten kautta Kansalliseen palveluväylään yrityksen oman integraatiosovelluksen kautta sekä löytää kehitysehdotuksia siihen. Toisena tavoitteena oli luoda matalan tason ohjeistusta yrityksen jatkuvien palveluiden tiimille, jotka ylläpitävät ja kehittävät kyseistä liittymää. Erityisesti ohjeistus uusien asiakkaiden liittämiseksi ja vikatilanteiden selvittämiseksi oli dokumentaation lähtökohta.

Opinnäytetyön teoreettinen viitekehys koostuu keskeisistä käsitteistä sekä yleisistä integraatioissa käytetyistä malleista. Sen jälkeen käydään läpi tutkimusmenetelmät ja niihin liittyviä asioita, sekä jälkeen käydään läpi tutkimuksessa löydettyjä keskeisimpiä komponentteja, joista itse dokumentaatio koostuu.

Tutkimusmenetelminä on käytetty pääasiassa kirjallisuuskatsausta, mutta opinnäytetyössä toteutettiin myös pieni avoin haastattelu. Materiaalia kerättiin useista kirjallista lähteistä pääpainon ollessa kuitenkin sähköisissä lähteissä. Vaikka tutkimus koostui kahdesta erilaisesta tutkimuksesta, joista toinen keskittyi isoon määrään kirjallista materiaalia ja toinen pieneen kohdennettuun haastatteluun, ei tutkimusta luokitella monimenetelmätutkimukseksi.

Kirjallisuuskatsauksen tuloksena saatiin sekä yleisluontaista kuvaa kyseisestä liittymästä, että yksityiskohtaisempaa ohjeistusta, sekä muutamia kehitysehdotuksia. Haastattelun tuloksena saatiin useampi kehitysehdotus. Tulokset vastasivat isolta osin tutkimuksen tavoitetta. Toisaalta raportin kannalta jouduttiin tyytymään monelta osin yleiskuvaan tietoturvasyistä. Itse tuotos osoittautui hyödylliseksi käytännönkin kannalta.

Asiasanat: integraatio, liityntäpalvelin, liittymä

Laurea University of Applied Sciences **Abstract**
Degree Programme in Business Information Technology
Bachelor's Thesis

Mikko Korhonen

Interface to the National Data Exchange Layer: Mapping, Documentation and Development

Year 2020 Pages 42

The aim of this Bachelor's thesis was to study and document the integration maintained by Digia Plc, via access servers into the National data exchange layer through the company's own integration application and to find development proposals for it. The goal was to create low-level guidance for the company's continuous services team that maintains and develops that interface. Especially guidelines for connecting new customers and resolving fault situations was the starting point for the documentation.

The theoretical framework of the thesis consists of key concepts and general models used in integrations. After that, the research methods and related issues are reviewed. It then introduces through the key components found in the study, which make up the documentation itself.

The research methods used were mainly a literature review, but a small open interview was also conducted in the thesis. Material was collected from several written sources, with the main emphasis on electronic sources. Although the study consisted of two different studies, one focused on a large amount of written material and the other on a small targeted interview, the study is not classified as a multi-method study.

The literature review resulted in both a general picture of the connection in question and more detailed guidance as well as a few suggestions for development. As a result of the interview, several development proposals were received. The results were largely in line with the aim of the study. On the other hand, the report had to be satisfied in many respects with the Overview for security reasons. The output itself proved to be useful in practice.

Keywords: integration, data exchange layer server, connection

Sisällys

1 Johdanto

Sovellusten ja järjestelmien integraatiosta voi kuulla ja lukea paljon tänä päivänä. Kuitenkaan näissä yhteyksissä ei tyypillisesti selitetä tarkemmin mitä termillä tarkoitetaan. Lyhyesti selitettynä integraatiolla tarkoitetaan kahden tai useamman järjestelmän liittämistä toisiinsa, jotta samaa tietoa voisi hyödyntää useammassa sovelluksessa ilman manuaalista tietojen siirtämistä. Yleensä tietoa myös muutetaan toiseen muotoon, jotta vastaanottavat järjestelmät osaavat tulkita kyseistä tietoa.

Yritysten ja organisaatioiden tarpeet integraatiolle lähtevät yleensä siitä, että käytössä on eri ohjelmistoja ja järjestelmiä monenlaisiin yhä yleistyviin tarpeisiin, mutta niitä ei ole tehty toimimaan ja keskustelemaan keskenään. Esimerkiksi tilausten tekeminen saatetaan tehdä sähköpostilla, mutta varastosaldoja ja talouslukuja ylläpidetään omissa järjestelmissään tai joskus jopa pelkässä taulukkolaskentaohjelmassa. Tällöin tiedon hakuun ja yhdistämiseen kuluu paljon aikaa ja resursseja, sekä muodostuu niin sanottuja informaatiosiiloja. Yksi integraation tärkeimmistä tehtävistä onkin automatisointi. Siirtojen automatisoinnilla aineistojen siirtoja saadaan yhä luotettavammiksi ja samalla optimoitua henkilötyövoiman käyttöä. Tämä tuo yrityksille ja yhteisöille rahallista säästöä, joustavuutta työhön ja tehokkuutta sekä informaatio teknisiin ratkaisuihin, että työntekijöiden jokapäiväiseen työhön.

Tässä opinnäytetyössä tehdyn tutkimuksen aiheena oli asiakkaalle räätälöidyn tilaus- ja toimitusliittymän kartoittaminen, dokumentaation parantaminen ja itse liittymän kehittämisen kartoittaminen. Työn on tarkoitus tuottaa käytännön työssä tarvittavaa ohjeistusta ja dokumentaatiota opinnäytetyön asiakasyrityksenä toimivalle Digia Finland Oyj:n jatkuvien palvelujen tiimille, joka valvoo, ylläpitää ja kehittää osaltaan kyseistä liittymää.

Tämän opinnäytetyön tärkeimpiä komponentteja on Suomen valtion tarjoaman X-Road teknologiaan pohjatuvan Palveluväylän läpi tapahtuva tilaus- ja toimitusliikenne. Palveluväylän tarjoaa Suomen valtion suomi.fi sivusto, jonka palvelujen tarkoituksena on auttaa yksityisiä ja julkisia organisaatioita parantamaan omia digitaalisia palveluitaan turvallisesti ja helposti sekä yhtenäistää digitaalisia palveluita Palveluväylän ja X-Road teknologian avulla. Yhtenäinen standardi liittymiseen ja autentikointeineen voivat osaltaan nopeuttaa digitalisaatiota ja järjestelmien integraatiota. Opinnäytetyön tuottamalla dokumentaatiolla haluttiin varautua myös jatkoon. Tulevaisuus näyttää tuleeko X-Road ja Palveluväylä olemaan isojakin osatekijöitä organisaatioiden ja valtioiden järjestelmien yhtenäistämisessä. Joka tapauksessa opinnäytetyöllä haluttiin varautua myös siihen.

Tässä opinnäytetyössä kerrotaan ensin yleisesti integraatioista. Sen jälkeen kuvaillaan tutkimuksessa käytettyjä tutkimusmenetelmiä, jonka jälkeen kuvaillaan tarkemmin käytännön

kannalta itse liittymän tärkeimpiä osia ja niihin liittyviä havaintoja sekä Kansalliseen palveluväylään liittyviä tärkeitä havaintoja. Lopuksi käydään läpi haastattelun tulokset ja tehdään johtopäätökset löydetyistä kehitysideoista liittymän osalta.

2 Työn lähtökohdat

Opinnäytetyön toimeksiantajana oli ohjelmistoalalla toimiva yritys Digia Finland Oyj, joka tuottaa asiakasyrityksille monenlaisia ratkaisuja ja tukipalveluja, ja joka tunnetaan erityisesti integraatioratkaisuistaan. Kyseessä oleva kansalliseen palveluväylään liittyminen iSuite-integraation kautta on toteutettu ja ylläpidetty asiakasyritykselle Digia Oyj:n kautta. Digia vastaa myös palvelun kehittämisestä itse integraation osalta sekä palvelinten ylläpidon osalta.

Tämän työn lähtökohtana oli tarve omassa ja tiimin työssä. Tutkimus sai alkunsa tarpeesta, koska vanhat palveluntekijät ja ylläpitäjät olivat siirtyneet muihin osastoihin yrityksen sisällä tai lähteneet muihin yrityksiin töihin. Liittymän dokumentointi oli jäänyt huonoksi eikä kenelläkään ollut hyvää kokonaiskuvaa palvelun toiminnasta, tulevaisuudesta ja kehittämisestä. Tutkimuksesta hyötyy yrityksemme Digia Oyj ja erityisesti oma osastomme, joka pyörittää liittymän ylläpitoa iSuite-integraation osalta, osan AWS-palvelinten ja Linuxin osalta sekä itse X-Road päivitysten osalta Kansallisesta palveluväylästä.

2.1 Tutkimuskohteen kuvaus ja kehittämistavoite

Kyseessä oleva sairaaloita ja sairaalatarvikkeita toimittavia yrityksiä yhdistävä tilaus- ja toimitusliittymä on tällä hetkellä toteutettu usealla eri palvelimella Amazonin AWS pilvessä. Tutkimuskohde sisältää toimittajien ja asiakkaiden välisiä integraatioita Digian oman iSuite-nimisen integraatio-ohjelmiston ja Palveluväylän välillä sekä välittäen tilauksia asiakasyritysten palvelimille heidän omien ERP-järjestelmien käsiteltäviksi. Integraatio sisältää muutamia eri integraatiomalleja, joista lisää myöhemmässä luvussa.

Tämän opinnäytetyön tavoite on siis ohjeistusten ja dokumentaation parantaminen päivittäiseen työhön liittyvien komponenttien osalta sekä itse liittymän kehittämismahdollisuuksien kartoittaminen ja osittain toteuttaminen. Opinnäytetyön tuottaman dokumentaation tuottamaa ohjeistusta voidaan hyödyntää heti päivittäisessä työssä ja osaa materiaalia voidaan myös hyödyntää tulevaisuudessa toisissa integraatioissa, jotka hyödyntävät Kansallista palveluväylää. Tarkoitus ei ole dokumentoida koko liittymää.

2.2 Aihealueen rajaus

Opinnäytetyön aiheen valitseminen oli helppoa, koska sille oli selkeä tarve omassa työssäni, mutta työn rajaaminen koskemaan liittymän tärkeimpiä komponentteja työni kannalta olikin haastavampaa.

Tässä opinnäytetyössä keskitytään kuvaamaan liittymän integraatioita, X-Road teknologiaa sekä muutamia palvelimille tärkeitä Linux teknologioita. Opinnäytetyön tarkoitus on saada dokumentoitua kokonaiskuva liittymästä sekä tarkempia käytännön ohjeita liittymän konfigurointiin, ohjaamiseen ja käyttämiseen. Kansallinen palveluväylä on Suomi.fi -sivuston tarjoama palvelu, jota ylläpitää Digi- ja väestövirasto ja se toimii kaikille yrityksille ja organisaatioille ilmaiseksi saatavana palveluväylänä. Opinnäytetyön kohde on toteutettu toiselle asiakkaana olevalle yritykselle. Edellä mainituista seikoista johtuen tapahtuu suurin osa dokumentaatiosta yrityksen sisäiseen verkkoon, eikä tarkempia kuvauksia voida julkaista tässä opinnäytetyössä. Niiltä osin dokumentaatiosta, jota on yleisesti nähtävillä esimerkiksi Internetissä, voitiin tärkeimpiä löytöjä kirjata tähän opinnäytetyöhön. Näitä löydöksiä olivat erityisesti Palveluväylään liittyvät havainnot.

Kyseiseen kansalliseen palveluväylään liittyminen iSuite-integraation kautta liittyy myös kevyenä ERP-järjestelmänä toimiva Portaali. Tämä Portaali rajataan myös opinnäytetyön ulkopuolelle, koska tiimimme ei ylläpidä sitä, ja kyseinen ERP-järjestelmä oli toteutettu Kontena Oyn konttiteknologialla. Kontena Oy lopettaa toimintansa taloudellisista syistä, joten Portaali on juuri siirtymässä toisen konttiteknologian piiriin.

Pois opinnäytetyöstä rajattiin myös tiettyjen sanomatyyppien siirtämisestä ja valvomisesta vastaava ActiveMQ-viestipalvelin. ActiveMQ on suosituin avoimen lähdekoodin viestienvälitysjärjestelmä, joka on vahvasti Java-pohjainen ja täten alustariippumaton.

2.3 Tutkimuskysymykset

Tämän opinnäytetyön oli tarkoitus vastata kysymyksiin mitä osa-alueita kyseessä olevaan liittymään kuuluu ja miten niitä käytetään, sekä miten kyseistä liittymää saisi parannettua. Edellä mainittujen kysymysten oli tarkoitus tuottaa dokumentaatiota ja ohjeistusta sekä konkreettisia kehitysehdotuksia liittymän parantamiseksi.

2.4 Keskeiset käsitteet

AWS: Amazon Web Services (AWS) on Amazonin tuottama pilvipalvelu, joka tarjoaa erilaisia työvälineitä verkkopalvelujen rakentamiseen Amazonin koneisaleissa.

HTTPS ja HTTP: HTTP (Hypertext Transfer Protocol eli hypertekstin siirtoprotokolla) on pääasiassa selainten ja WWW-palvelinten käyttämä tiedonsiirtomenetelmä. HTTPS (Hypertext

Transfer Protocol Secure) on http-protokollan ja TLS/SSL salausten yhdistelmä tiedon suojattuun siirtoon.

Järjestelmäintegraatio: Järjestelmäintegraatio on kokoelma toimintapajoja ja teknologioita, joilla saadaan eri järjestelmät eri sijainneissa keskustelemaan keskenään ja hyödyntämään samaa tietoa, vaikka järjestelmät eivät muuten olisi yhteensopivia.

Kansallinen palveluväylä: Suomi.fi -sivuston tarjoama palvelu, jota ylläpitää Digi- ja väestövirasto ja toimii kaikille yrityksille ja organisaatioille saatavana palveluväylänä.

Liityntäpalvelin: Organisaatioiden palvelimia, joihin asennettuna palveluväylän sanomansiirron mahdollistava liityntäohjelmisto. Nämä palvelimet keskustelevat keskenään julkisen Internetin yli.

Linux: Linux viittaa Linux-ydintä käyttäviin käyttöjärjestelmiin. Tähän opinnäytetyöhön oleellisesti kuuluvat Linux-käyttöjärjestelmät ovat Red Hat ja Ubuntu.

Ohjelmointirajapinta: Ohjelmointirajapinta on määritelmä, joka mahdollistaa ohjelmien keskustelun keskenään tekemällä erilaisia pyyntöjä. Ohjelmointirajapintoja kutsutaan yleensä englanninkielisen nimensä lyhenteellä API (Application programming interface).

REST: REST (Representational State Transfer) on arkkitehtuurimalli, joka keskustelee myös lähettämällä pyyntöjä palveluntarjoajalle, mutta erona SOAPiin on se, että REST käyttää vain http-protokollaa.

SOAP: SOAP (Simple Object Access Protocol) on XML-pohjainen tietojenvaihtoon tarkoitettu protokolla ja kieli. SOAP lähettää pyyntöjä palveluntarjoajan rajapintaan ja saa paluuna vastauksen palveluntarjoajalta.

TLS/SSL: (Transport Layer Security) on protokolla, jolla salataan yleensä liikenne julkisessa Internetissä. SSL-protokolla (Secure Socket Layer) on TLS:n aikaisempi nimi vanhemmilla versioilla.

Web service: Ohjelmistojärjestelmä, joka mahdollistaa verkon yli tapahtuvan tietokoneiden välisen vuorovaikutuksen. Käytännössä tällä tarkoitetaan yleensä Wordl Wide Web -pohjaisia rajapintoja.

X-Road: Alkujaan Virossa kehitetty X-Road on tiedonsiirtoprotokolla ratkaisu, jonka tarkoitus on mahdollistaa tietoturvallinen integraatioratkaisu. X-Road on ilmainen ja avoimen lähdekoodin protokolla, joka on julkaistu MIT avoimen lähdekoodin lisenssin alla eli se on kaikkien saatavilla ilmaiseksi.

3 Järjestelmäintegraatio

Termi system integration, eli vapaasti käännettynä järjestelmäintegraatio, syntyi tiettävästi 1950-1960-lukujen taitteen aikoihin. Järjestelmät räätälöitiin yleensä asiakaskohtaisesti, ja monenlainen joukko laitteistoja ja käyttöjärjestelmiä, jotka eivät olleet yhteensopivia keskenään, johtivat pian ongelmiin pyrittäessä laajentamaan ja yhdistämään ratkaisujen toiminnallisuutta. Näihin ongelmiin ryhdyttiin kehittämään erilaisia integraatioratkaisuja. Järjestelmäintegraatioiden juurina pidetään yleensä tietokoneaikakauden suuria projekteja, joista kaksi olivat SAGE ja SABRE. SAGE (Semi-Automatic Ground Environment) oli Yhdysvaltojen ilmavoimien vuonna 1949 alulle paneman selvityksen pohjalta syntynyt projekti, jonka tarkoitus oli automatisoida vanhentunut ilmavalvontajärjestelmä. SAGE poiki useita vastaavia projekteja armeijan käyttöön. Tyypillistä näille projekteille oli useat laitteisto- ja ohjelmistotoimittajat. Näiden erilaisten järjestelmien tekemistä yhdessä automatisoiduiksi kokonaisuuksiksi alettiin kutsumaan termillä system integration. (Tähtinen 2005, 17.)

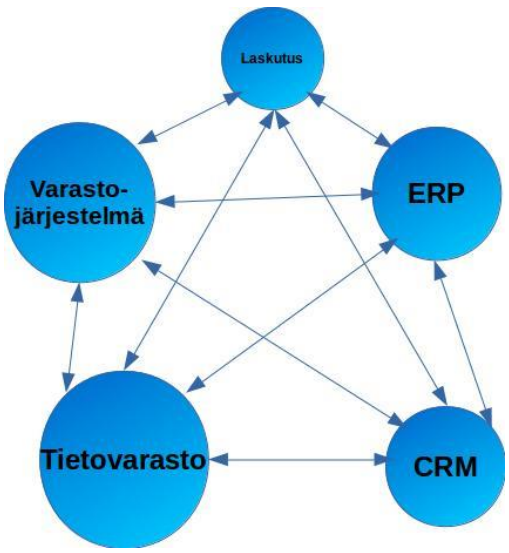
Siviilipuolella ensimmäinen suuri projekti erilaisten järjestelmien integroimiseksi oli SABRE. Nimi juontaa juurensa IBM:n projektista Semi-Automatic Business Environment Research (SABER). Projekti vaati yli kymmenen vuoden suunnittelu- ja kehitystyön ennen kuin American Airlines otti sen käyttöön vuonna 1964. Lentoyhtiö automatisoi sillä lentojensa paikkavarauksen ja lipunmyynnin. (Tähtinen 2005, 17-18.)

Digitalisaatio on levinnyt maailmalla viimeisten vuosikymmenten aikana laajalti ja esimerkiksi IoT (englanniksi Internet of Things) eli esineiden internet on tulossa sekä organisaatioiden että ihmisten jokapäiväiseen elämään vauhdilla. 5G verkkojen yleistyessä ennustetaan keskenään keskustelevien älykotien, puettavan teknologian, älyvaatteiden ja itsestään ajavien ajoneuvojen yleistyessä tulemaan ohjelmistoille ja esineille suurempi tarve kykyyn kommunikoida keskenään ja jakaa tietoa. Tämä tulee asettamaan integraatiolle uusia tarpeita, mutta myös haasteita ja kehitystarpeita.

Nykypäivän yrityksille integraatiot ovat yksi tärkeimmistä informaatioteknologian mahdollistamista mahdollisuuksista tehostaa yrityksen toimintaa ja kilpailukykyä informaation saralla. Integraatiot voivat olla lähes näkymättömiä jokapäiväisessä työssä, mutta niiden tuomat hyödyt voivat olla mittavia. Olipa kyse toiminnan tehostamisesta automatisoimalla tai tiedon hyödyntämisessä organisaation sisäisesti tai uusien palvelujen tuottamisessa, voivat integraatiot olla merkittävässä roolissa, jos niitä hyödynnetään oikein. Integraatioita voidaan toteuttaa usealla eri mallilla. Seuraavaksi esitellään kolme erilaista yleismallia.

3.1 Point-to-point-arkkitehtuurimalli

Point-to-point -integraatiolla tarkoitetaan tapaa, jolla jokainen integroitava järjestelmä liitetään toisiinsa suorilla yhteyksillä. Tämä integraatio on yksinkertaisimmillaan kahden järjestelmän välinen suora integraatio. Tähän tarpeeseen kehitettiin point-to-point -integraatio. Tätä mallia kutsutaan myös Siltayhteistoiminta-malliksi, koska tässä mallissa tieto liikkuu suorilla yhteyksillä, eli siltojen kautta järjestelmien välillä. (Josuttis 2007, 103). Muutaman järjestelmän integroivana arkkitehtuurimallina se on hyvinkin toimiva ja kustannustehokas ratkaisu, mutta mitä enemmän järjestelmiä integroidaan, sen vaikeammin hallittava, ylläpidettävä ja vähemmän vikasietoinen integraatiosta tulee. Esimerkiksi kolmella järjestelmällä integraatio toimii vielä kolmella joko yhden- tai kahdensuuntaisella yhteydellä, mutta neljännen järjestelmän mukaan tullessa yhteyksiä tarvitaan jo kuusi ja viidellä järjestelmällä tarvitaan jo kymmen, mutta jos integraatiot toimivat molempiin suuntiin tarvitaan niitä jo kaksikymmentä, kuten kuviossa 1 on kuvattuna. Yli kolmen integraation kohdalla point-to-point -integraatio ei ole enää tehokas ratkaisu, koska yhteyksien määrä kasvaa neliöllisesti liitettävien järjestelmien lukumäärään nähden. Tästä syystä tätä mallia kutsutaankin integraatiospagetiksi. Edellä mainituista syistä johtuen seuraava kehityssuunta laajempien integraatiokokonaisuuksien hallintaan oli keskitetty integraatio.



Kuvio 1: Point-to-point -malli

3.2 Hub-and-Spoke -arkkitehtuurimalli

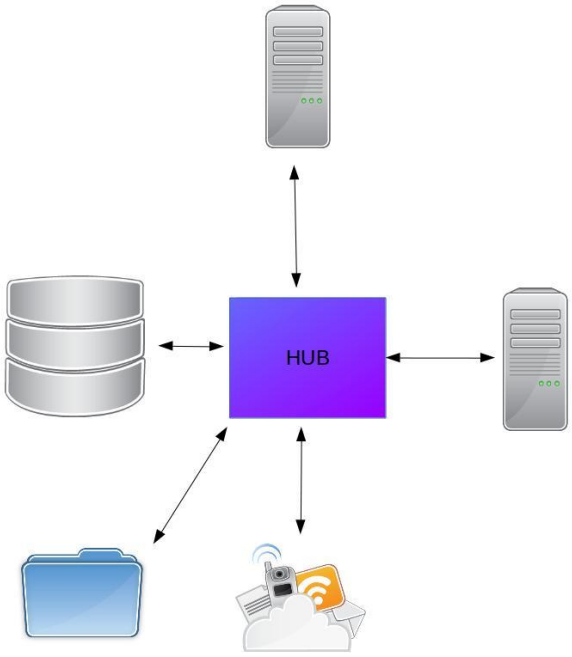
Vielä 2000-luvun alussa integraatiomalleista puhuttaessa ja niitä toteuttaessa tarkoitettiin pitkälti hub-and-spoke -arkkitehtuurimallia. Hub-and-spoke on malli, jossa järjestelmien vä-

lissä on niin sanottu puhelinkeskus. Jokainen integroitava järjestelmä eli spoke kytkeytyy tähän puhelinkeskukseksi kutsuttuun keskitettyyn integraatioratkaisuun eli hubiin. (Kuvio 2). Tämän keskitetyn mallin ansiosta sanomien välitystä ja muunnosta voidaan ohjata hallitusti. (Bussler 2003, 103.)

Hub-and-spoke -mallin suurimpia hyötyjä verrattuna point-to-point -malliin on uusien järjestelmien liittäminen ja vanhojen päivittäminen, koska kaikki järjestelmät on liitetty keskushubiin. Tällöin ei siis tarvitse tehdä muutosta jokaiseen vastaanottavaan ja lähettävään järjestelmään, vaan muutos tehdään vain keskusjärjestelmään eli hubiin.

Vaikka keskitetyssä mallissa on paljon hyödyllisiä asioita, on siinä myös huonot puolensa. Keskitetyssä arkkitehtuurissa voi liika keskittäminen aiheuttaa pullonkaulan. Jos sanomia liikkuu paljon ja ne ovat isokokoisia, ruuhkautuu sanomien välitys tai ruuhkautumien saattaa kaataa koko järjestelmän. Tällöin myös laitteistolta vaaditaan enemmän suorituskykyä mikä puolestansa nostaa kustannuksia. (Tähtinen 2005, 143.)

Toinen heikkous tällä mallilla on vikasietoisuus. Vaikka nykyiset hub-and-spoke -integraatiot ovatkin rakennettu vikasietoisemmiksi, keskittäminen tuo ongelmia luotettavuuden suhteen. Mikäli keskitetty järjestelmä eli hub vikaantuu, saattaa se lamauttaa kaikkien järjestelmien välisen integraation ja pahimmassa tapauksessa lamaantuu koko yrityksen liiketoiminta. (Tähtinen 2005, 143-144.)



Kuvio 2: Esimerkki Hub-and-Spoke -mallista

3.3 SOA ja ESB

Service Oriented Architecture (SOA) tarkoittaa palvelukeskeistä arkkitehtuuria. Se ei ole yksittäinen teknologia vaan ajattelutapa, jolla kyetään tutkimaan ja kehittämään organisaatioiden arkkitehtuuria. Tämä on arkkitehtuurimalli, jossa erityyppiset ja eri tarkoituksiin tehdyt ohjelmistot julkaisevat ulospäin palveluilta, joita muut ohjelmistot pystyvät kutsumaan. SOA pyrkii myös olemaan riippumaton käyttöjärjestelmistä, laitteistoista sekä ohjelmointitekniikoista. Tyypillisesti SOA mallit toteutetaan JAVA- ja Mircosoft.Net tyyppisten web services -tyyppisten alustojen avulla. (Tähtinen 2005, 144.)

ESB eli Enterprise Service Bus on kokoelma tuotteita ja tekniikoita, joilla voidaan toteuttaa SOA-arkkitehtuurin mukaista sanomien välitystä. ESB-mallia pidetään yleensä hub-and-spoke -mallin kehittyneempänä versiona ja seuraavana sukupolvena. ESB on malli, jossa keskitetty ohjelmistokomponentti ohjaa integraatioita ja muunnoksia taustajärjestelmiin ja antaa nämä

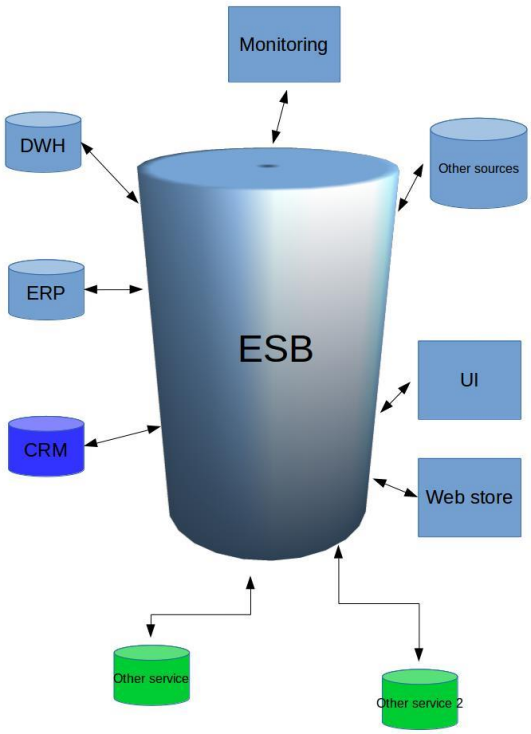
palvelut kaikkien sovellusten käyttöön. Vaikka ohjelmisto itse saattaa vaikuttaa monoliittiselta, on sen muuntimet ja integraatiomoottorit toteutettu jokaiseen liittymään erikseen, joten tätä mallia voidaan kutsua hajautetuksi malliksi.

Yksi SOA-mallin pääpiirre onkin tuki heterogeenisille järjestelmille. Nykypäivän isoissa organisaatioissa käytetään hyvinkin monia sovelluksia, eikä aina voida olettaa, että tietojärjestelmät olisivat ratkaisuiltaan läheskään samanlaisia. Tämä ajattelutapa on vielä korostunut pilvipalvelujen yleistyessä. Siksi SOA-ajatusmallille on keskeistä juuri tuki heterogeenisille ympäristöille. (Josuttis 2007, 57.)

ESB-mallia pidetään monesti samana, kuin message bus -mallia. Vaikka ne pohjimmiltaan ovatkin sama asiaa, on ESB-malli paljon muutakin kuin pelkkä message bus -väylä. Message bus-integraatiossa sovelluksella täytyy olla ennakkoon tiedossa viestien muoto. Sanomilla on niin sanotusti staattinen sidos. ESB-mallissa taas sanomille tarjotaan enemmän palvelua ja sen keskeisimpiä hyötyjä ovatkin:

- Sama sanoma monta vastaanottajaa. Sama sanoma voidaan muuntaa ja lähettää useammalle vastaanottavalle järjestelmälle samaan aikaan eri muodoissa.
- ESB-mallissa on yleensä keskitetty lokitiedostojen valvonta. Tällöin ei ole tarvetta kirjautua useaan eri järjestelmään tarkastelemaan lokitietoja.
- Skaalautuvuus. ESB-mallissa voidaan sanomia tänä päivänä yleensä käsitellä useammalla instanssilla. Hyvänä esimerkkinä on pilvipalvelujen tarjoaman kuormanjako.

ESB-malli pystyy tarjoamaan asiakkaalle yhden reitin kaikille tarvittaville palveluille. (Christudas 2008). Kuviossa 3 on esimerkki ESB-mallista, jossa on itse väylä, jonka läpi integroitavien järjestelmien sanomat kulkevat, mutta siihen on liitetty myös muita palvelua.



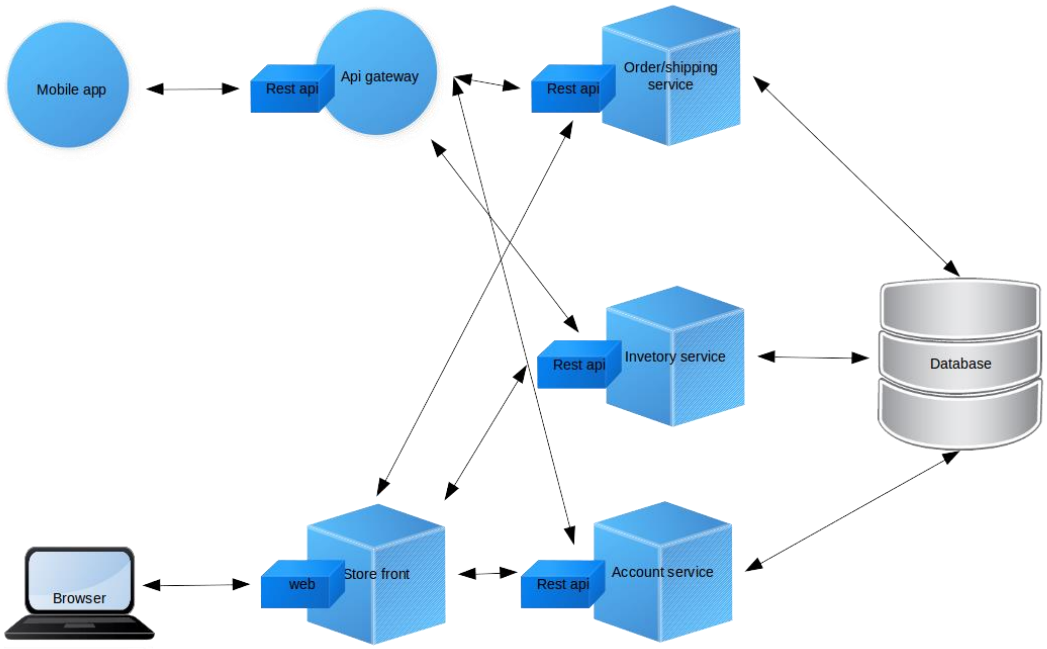
Kuvio 3: Esimerkki ESB-mallista

3.4 Mikropalvelut

Ennusteiden mukaan tulevaisuuden näkymät integraatioissa ovat mikropalveluarkkitehtuuriin perustuvia. Toiset hehkuttavat ESB-mallia jo vanhentuneeksi arkkitehtuuriksi, jota ei kannata tuoda enää yritysten integraatioihin. Harva kumminkaan tietää mitä mikropalveluilla oikeastaan tarkoitetaan. Se johtuu siitä, ettei ole paljoa tietoa, joka määrittelee, mikä on mikropalvelumalli ja miten se tehdään. Tiivistettynä mikropalvelujen arkkitehtuuri on tapa, jossa pienten palvelujen sarjassa jokainen toteuttaen omia prosesseja, kommunikoi keskenään yleensä http-protokollan avulla rajapintakutsujen kautta. Tästä käytännön esimerkki kuviossa 4. Tämän ketjun tarkoitus on olla kevyt, nopea sekä mahdollisimman automaattinen. Näiden palvelujen keskitetty hallinta on yleensä minimaalista ja niiden on yleensä tarkoitus olla mahdollisimman alustariippumattomia. (Fowler 2014.)

Huhut ESB:n kuolemasta ovat tällä hetkellä vahvasti liioiteltua ja monien yritysten järjestelmät ovatkin yhdistelmä näitä malleja eli ne ovat hybridimallisia. Viime vuosina useat ohjelmistotoimittajat ovat alkaneet tuoda omia pilvipohjaisia palvelumallejaan. Näitä ohjelmistoja

ja palveluita kutsutaan termillä SaaS (Software as a Service). Tämä mahdollistaa monille yrityksille helpon pääsyn eri järjestelmiin ilman omia konesaleja ja niiden ylläpitoa. Luomalla toisistaan riippumattomia palveluita saattavat Saas-palvelut alkaa purkamaan monoliittisia ohjelmisto- ja integraatioratkaisuja ja siirtymään kohti API-rajapintaratkaisuja. (Kivisaari 2016.)



Kuvio 4: Esimerkki mikropalveluista

4 Tutkimusmenetelmät

Tämä opinnäytetyö oli toimeksiantajana toimivalle yritykselle tehtävä liittymän kartoittaminen, dokumentoinnin ja ohjeistusten parantaminen sekä aluksi pienempänä osana oleva, mutta tutkimuksen edistyessä kasvavana alueena oleva liittymän kehittäminen tulevaisuudessa. Opinnäytetyö on myös elänyt matkan varrella vastaamaan paremmin sen tarpeita toimeksiantajan, ylläpitäjätiimin ja asiakkaan tarpeiden mukaan. Dokumentaation jälkeen arvioidaan sen hyödyllisyyttä, sekä miten tiimi voi ja itse pystyn käytännössä hyödyntää sitä, sekä mitä uutta tietoa se on tuottanut. Itse dokumentaatio on toteutettu toimeksiantajana toimineen yrityksen sisäiseen verkkoon ja se on jaettu useammalle sivulle eri otsikoiden alle.

Työ alkoi liittymän osien ja komponenttien etsimisellä vanhasta vajaasta dokumentaatiosta, omasta kokemuksesta liittymän parissa työskentelystä sekä olemassa olevan tiedon etsimisellä

ja kartoittamisella sähköisistä ja kirjallisista lähteistä. Näiden tietojen perusteella tein käsitekartan aiheesta. Käsitekartan perusteella alkoi systemaattisempi tiedon etsiminen ja kartoittaminen sekä etsimällä painettua materiaalia aiheista, sekä sähköisiä lähteitä verkosta. Myöhemmin pidettiin myös haastattelut sekä liittymän parissa nyt toimiville kehittäjille sekä vanhoille kehittäjille, jotka olivat siirtyneet toisiin tehtäviin.

4.1 Monimenetelmätutkimus

Tässä tutkimuksessa yhdistellään useampaa tutkimusmenetelmää. Tietoa kerätään laajalta alalla kirjallisuuskatsauksella, oman havainnoinnin perusteella ja pienen kohdennetun haastattelun perusteella. Laadullisen ja määrällisen tutkimusmetodien yhdistämistä samassa tutkimuksessa kutsutaan monimenetelmätutkimukseksi. Jos samassa tutkimuksessa yhdistellään kahta eri laadullista menetelmää tai kahta eri määrällistä menetelmää, sijoittuu tutkimus joko määrällisen tai laadullisen tutkimuksen metodologiseen viitekehykseen. Mutta jos tutkimuksessa yhdistellään laadullista aineistonkeruutta ja määrällistä aineistonkeruuta, luokitellaan tutkimus kuuluvan monimenetelmätutkimukseksi. (Bryman 2008,165). Tässä tutkimuksessa yhdisteltiin menetelmiä pieneltä alueelta haastatteluissa ja suuremmalta alueelta kirjallisuuskatsauksella, mutta molemmat koskivat tarkkaa aihetta, sekä molemmat tutkimukset tehtiin rinnakkain, joten opinnäytetyön tutkimusta ei luokitella monimenetelmätutkimus.

Määrällisellä, eli kvantitatiivisella tutkimuksella tarkoitetaan tutkimuksen suuntausta, joka perustuu tutkittavan kohteen kuvaamiseen ja tulkitsemiseen numeroiden ja tilastojen avulla. Määrällisessä tutkimuksessa ollaan usein kiinnostuneita vertailuista, syy- ja seuraussuhteista ja numeerisesta ilmiöiden selittämisestä. (Jyväskylän yliopisto 2015a.)

Laadullisella, eli kvalitatiivisella tutkimuksella tarkoitetaan tutkimuksen suuntausta, jossa pyritään ymmärtämään tutkittavan kohteen laatua ja ominaisuuksia kokonaisvaltaisesti. Laadullisen tutkimuksen menetelmissä yhteisinä ominaisuuksina korostuu kohteen ympäristöön ja taustaan ja sen tarkoitukseen ja merkitykseen liittyvät asiat ja näkökulmat. (Jyväskylän yliopisto 2015b.)

4.2 Kirjallisuuskatsaus

Kirjallisuuskatsauksen avulla pyritään samaan saamaan kokonaiskuvaa opinnäytetyön aihepiiristä. Sen avulla pyritään siis löytämään jo olemassa olevaa tutkittua ja dokumentoitua tietoa aiheesta. Opinnäytetöissä on yleensä teoreettinen viitekehys, jossa määritellään opinnäytetyön keskeiset käsitteet. Kyseinen viitekehys perustuu järjestelmälliseen tiedonhakuun kyseisistä käsitteistä ja aiheista. Tätä tutkimusta aihepiiristä ja käsitteistä kutsutaankin paremmin kuvaavasti toiselta nimeltään tutkimuskatsaukseksi, vaikka viralliselta nimeltään sen on kirjallisuuskatsaus. (Hirsjärvi, Remes & Sajavaara 2009, 121.)

Kirjallisuuskatsauksia on monenlaisia ja niitä voidaan käyttää moniin erilaisiin tarkoituksiin. Siksi kirjallisuuskatsaukset onkin jaoteltu ja nimetty toisistaan eroavasti. Tämä opinnäytetyö sivuaa monenlaista kirjallisuuskatsausta, mutta opinnäytetyö alkoi liittymän komponenttien kartoittamisella käsitekartaksi ja sitä kautta tiedon systemaattisella haulla käsitekartan termeillä ja niiden yhdistelmillä. Tällä perusteella opinnäytetyön voidaan luokitella systemaattiseksi kirjallisuuskatsaukseksi. Käytän myös kuvailua eli narratiivisen kirjallisuuskatsauksen tunnusmerkkejä täyttyy tutkimuksessa.

4.3 Haastattelututkimus

Toisena menetelmänä tässä opinnäytetyössä käytettiin avointa haastattelua. Haastattelu on yksi yleisimmistä menetelmistä tiedonkeräämiseen. Haastattelu sopiikin hyvin kehittämistehtäviin, mihin sitä onkin tässä opinnäytetyössä käytetty. Haastattelu sopii erityisesti tiedonkeruumenetelmäksi, kun halutaan tuoda yksilön omia näkökulmia ja mielipiteitä mahdollisimman vapaasti. Se sopii erityisesti asioiden selventämiseen ja syventämiseen. Haastattelu kannattaa myös yhdistää toisiin menetelmiin, koska ne useasti tukevat toisiaan. Haastattelut tehtiin aidossa toimintaympäristössä toimistona palveluympäristöissä, koska asioita on helpompi kuvailla ja muistaa, kun ollaan niiden äärellä. (Ojasalo, Moilanen & Ritalahti 2014, 106-107.)

Haastattelumenetelmiä on erilaisia ja tähän menetelmäksi on valittu avoin haastattelu. Avoimessa haastattelussa haastateltava ja haastattelija keskustelevat yleisesti ja vapaamuotoisesti kehittämiskohteesta. Molemmat osapuolet myös osallistuvat keskusteluun aktiivisesti. (Ojalalo ym. 2014, 108-109). Keskustelu oli myös erittäin epämuodollinen, joten sekin tuki avointa haastattelua menetelmänä. Avoimella haastattelulla saatiin lisätuloksia jo osittain tehtyyn kvantitatiiviseen tutkimukseen eli se tuki saatua materiaalia kirjallisuuskatsauksesta.

4.4 Validiteetti ja reliabiliteetti

Tutkimuksen pätevyys eli validius tarkoittaa käytetyn tutkimusmenetelmän tai mittarin kykyä mitata juuri sitä, mitä sillä oli tarkoituskin mitata tutkimuksessa. Pätevässä tutkimuksessa ei saisi olla systemaattisia virheitä. Tällä tarkoitetaan sitä, miten tutkijat ovat ymmärtäneet itse mittarin eli esimerkiksi kysymykset. Tulokset vääristyvät, jos vastaaja ymmärtää kysymykset eri tavalla kuin tutkija. Täten validiutta pitää miettiä jo tutkimusta suunniteltaessa. Tämä tarkoittaa käsitteiden, aineiston keräämisen ja mittareiden tarkkaa ja huolellista suunnittelua ja varmistamista, että käytettävät mittarit kattavat koko tutkittavan asian. (Hirsjärvi ym. 2009, 216-217.)

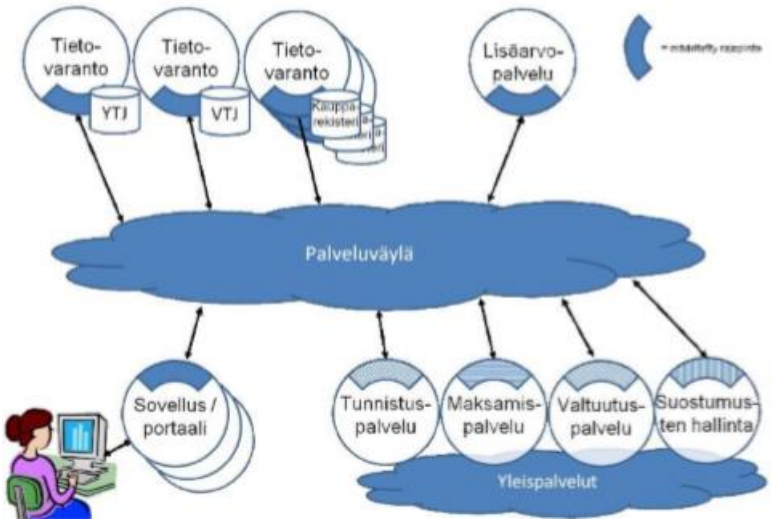
Tutkimuksen luetettavuus eli reliaabelius tarkoittaa tulosten tarkkuutta eli kuinka hyvin mittaustulokset ovat toistettavissa ja kuinka vähän sattumanvaraisia tuloksia esiintyy. Tämä tarkoittaa, että toisen tutkijan toistaessa sama tutkimus samalle henkilölle saadaan samat tulok-

set. Tosin Heikkilä esittää, että tutkimuksen luotettavuus on sidottu aikaan ja paikkaan. Tutkimuksen tuloksia ei siis yleistää vaikutusalueen ulkopuolelle, kuten toiseen aikaan tai toisen yhteyskuntaan. (Heikkilä 2004, 30; Hirsjärvi ym. 2009, 216.)

Tässä tutkimuksessa, kuten teknologiaan liittyvissä tutkimuksissa ja materiaaleissa yleensäkin, on aineisto erittäin sidottu aikaan. Teknologia muuttuu nopeasti, joten myös tämän liittymän komponentit tulevat muuttumaan ja kehittymään lähivuosina. Pelkästään tässä tutkimuksessa löydetyt kehitysehdotukset tulevat muuttamaan tämän liittymän kokonaiskuvaa. Osa hieman, osa paljonkin.

5 Kansallinen palveluarkkitehtuuri

Suomi.fi-palveluväylä on tiedonvälityskokonaisuus, joka toimii viestiväylänä siihen liitettyjen palveluiden ja tietovarantojen välillä. Palveluväylän tiedonvälitys perustuu alun perin Virossa toteutettuun X-Road-ratkaisuun ja sen uusimpaan versioon 6. Palveluväylää hyödyntäen voivat asiakasorganisaatiot käyttää ja rakentaa kokonaisuuksia eri tietolähteistä. Palveluväylän käyttö tarjoaa monia hyötyjä. Palveluväylä ehkä suurin hyöty on, että se tarjoaa yhtenäisen tavan tuoda tietoa eri järjestelmien välillä. Se tarjoaa myös standardoidut tietoturvaratkaisut ja avoimet rajapinnat KaPAan liitetyille tietolähteille ja palvelukomponenteille. (Suomi.fi, 2020a). Palveluväylä itsessään ei muuta tai luo toiminnallisuuksia tai varastoi tietoa. Palveluväylän hyödyt tulevat siihen liitetyistä tietolähteistä ja komponenteista, joista esimerkkejä kuviossa 6. Väylä itsenään on vain tiedonvälitysväylä.



Kuvio 5: Yleiskuva Kansallisesta palveluväylästä (Valtiovarainministeriö 2013)

Palveluväylien sisäinen liikenne on julkisen Internetin yli tapahtuvaa liityntäpalvelinten välistä liikennettä. Tiedonsiirron ajaksi liikenne salataan TLS (SSL) -salausprotokollalla, ja kaikki tieto allekirjoitetaan varmenteilla. X-Road-ratkaisun liityntäpalvelinten rajapinnoissa käytetään XML-kieleen pohjautuvaa SOAP (Simple Object Access Protocol) -tietoliikenneprotokollaa tai valinnaisesti REST-arkkitehtuuria sanomien siirtoon käyttäen http-protokollaa.

X-Road-ratkaisun keskeinen komponentti on liityntäpalvelin, jonka kautta yritysten ja organisaatioiden järjestelmät ja tietolähteet liitetään Palveluväylään. Jokaisella Kansalliseen palveluväylään liitetyllä järjestelmällä on oltava käytössään liityntäpalvelin, jonka välityksellä kaikki Palveluväylään lähetettävät tai sieltä vastaanotettavat sanomat kulkevat. Jokaisella järjestelmällä ei siis tarvitse olla omaa liityntäpalvelinta, vaan moni järjestelmä voi käyttää yhtä liityntäpalvelinta yhdessä esimerkiksi integraation kautta. Liityntäpalvelin vastaa monista asioista, esimerkiksi palvelukutsujen välittämisestä järjestelmien välillä, palvelukutsujen varmennekättelystä, tietoliikenteen ja sanomien salauksesta, käyttöoikeuksien hallinnasta ja lokitietojen kirjoittamisesta.

Palveluväylään liittyminen vaatii sitä, että liitettävä tietojärjestelmä joko pystyy lähettämään ja vastaanottamaan SOAP-sanomia X-Roadin edellyttämässä muodossa, tai noudattaa REST-arkkitehtuuria rajapintojen toteuttamiseen. Käytännössä siis SOAP-sanomien tulee sisältää tietyt X-Road-tiedonsiirtoprotokollan määrittelemät otsikkotiedot. Lisäksi SOAP-sanomien body-osaan on sisällytettävä kysely- ja vastausparametrit X-Road protokollan vaatimalla tavalla. Sanomiin on sisällytettävä myös X-Roadin vaatimat liittyjätiedot. X-Roadin versio 6 käyttää SOAP-versiota 1.1.

Tähän mennessä on puhuttu paljon SOAPista ja RESTistä, mutta niihin liittyy useasti väärinkäsityksiä, joten tutkimus vaati niiden selventämistä. SOAP (Simple Object Access Protocol) ja REST (Representational State Transfer) ovat molemmat verkkopalveluiden yhteysprotokollia. Viime vuosina REST on mennyt ohi suosiossa isosti. SOAP ja REST eroavat toisistaan paljon.

REST on yleinen arkkitehtuurimalli rajapinnoille. Se määrittelee millä operaatioilla palvelinta tietoa pyydetään, lisätään ja käsitellään. REST ei ota kantaa tiedon formaattiin ja se voikin olla esimerkiksi xml- tai json-muotoista, mutta siirtoon se käyttää yleensä vain http-protokollaa. REST on huomattavasti kevyempi rakenteeltaan ja osittain siksi sitä suositaan rajapinnoissa enemmän kuin SOAP-ratkaisuja. REST-rajapinnan kautta voidaan välittää dataa eri formaateissa. REST on enemmänkin tyyli, joka määrittelee joukon sääntöjä, miten verkkopalveluita käytetään.

SOAP puolestaan on XML-pohjainen protokolla ja huomattavasti raskaampi vaadittavien kehysten ja kenttien takia, mutta SOAP ei ole niin riippuvainen pelkästä http-protokollasta, vaan voi käyttää useita siirtoprotokollia. SOAP-pohjaisia rajapintoja ei yleensä pidetä niin helposti

lähestyttävinä kuin REST-rajapintoja. Vaikka SOAP-rajapinnat yleensä käyttävät http-siirto-protokollaa, on sen yksi ominaisuus se, ettei se ole sidottu pelkästään http-siirtoon.

5.1 Skeemat

Koska tässä liittymässä käytetään SOAP-protokollaa REST-arkkitehtuurin sijasta, edellyttää X-Road myös siihen liitettävien palveluiden kuvaamista WSDL (Web Service Description Language) -kielellä. Liityntäpalvelimen kysely- ja vastaussanomien rakenteen kuvaavat skeemat voidaan joko sisällyttää suoraan WSDL-kuvaukseen tai kuten tässä liittymässä, ne voidaan vaihtoehtoisesti kuvata erillisissä tiedostoissa, joihin viitataan WSDL-kuvauksessa import-määrityksen avulla. Tämä tuo selkeyttä ja uudelleenkäytettävyyttä eri kutsutyyppejä käyttäessä. Kuviossa 7 esimerkki skeemojen oikeasta käytöstä. Skeemojen käytössä esiintyy myös paljon ongelmia. Tästä syystä kuvattiin niiden käyttöä tarkemmin liitteessä 2.

Skeemoissa tulee käyttää nimiavaruuksia (namespace) eli verkkosijainteja ja nimetä elementit esimerkiksi etuliitteellä (prefix). WSDL-kuvauksessa käytettyjen datatyyppien täytyy löytyä nimiavaruuksien määrittämistä verkkosijainneista tai muuten WSDL-kutsu ei saa yhteyttä liityntäpalvelimella. Tämä näkyy liityntäpalvelimella punaiseksi menneestä WSDL-kutsuna. Tämä on huomioitava erityisesti siinä tapauksessa, että käytettävät nimiavaruuksien kohteet eivät ole yleisesti julkisessa internetissä saatavilla olevia, vaan ne on tehty esimerkiksi vain tiettyä, suljettua tarkoitusta varten. Tällöin nimiavaruuden sijainti pitää tarjota julkiseen internetiin, jotta sen käyttö on mahdollista. Tämä voi tapahtua esimerkiksi avaamalla palomuuri.

Skeemoissa on suositeltavaa, mutta ei pakollista käyttää elementFormDefault=”qualified”-määritystä. Sen mukaan kaikkien kyseiseen XML-skeemaan perustuvissa sanomissa käytettävien elementtien tulee täyttää skeemassa määritellyissä nimiavaruuksissa asetetut vaatimukset. Käytännössä sanomissa ei siis saa olla elementtejä, joita ei ole määritelty skeemassa tai skeeman käyttämissä nimiavaruuksissa. Tällä tavoin ehkäistään nimikonfliktien syntyminen ja varmistetaan, että jokainen elementti on yksilöitävissä yksiselitteisesti. (Suomi.fi, 2020b.)

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema elementFormDefault="qualified" xmlns:id="http://x-road.eu/xsd/identifiers"
targetNamespace="http://x-road.eu/xsd/xroad.xsd" xmlns:tns="http://x-road.eu/xsd/xroad.xsd"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
    <xsd:import schemaLocation="xroad6identifiers.xsd"
        namespace="http://x-road.eu/xsd/identifiers"/>
        <!-- Header fields -->
    <xsd:element type="id:XRoadClientIdentifierType" name="client"/>
    <xsd:element type="id:XRoadServiceIdentifierType" name="service"/>
    <xsd:element type="xsd:string" name="userId"/>
    <xsd:element type="xsd:string" name="id"/>
    <xsd:element type="xsd:string" name="protocolVersion"/>
    <xsd:element type="xsd:boolean" name="async"/>
  - <xsd:complexType name="authCertDeletionType">
      - <xsd:sequence>
          - <xsd:element type="id:XRoadCentralServiceIdentifierType" name="server">
              - <xsd:annotation>
                    <xsd:documentation>Identity of this security server. </xsd:documentation>
                </xsd:annotation>
            </xsd:element>
          - <xsd:element type="xsd:base64Binary" name="authCert">
              - <xsd:annotation>
                    <xsd:documentation>Hash of the authentication certificate that needs to be
                        deleted from the list of certificates authenticating this security server.
                        </xsd:documentation>
                </xsd:annotation>
            </xsd:element>
        </xsd:sequence>
    </xsd:complexType>
  - <xsd:complexType name="ClientRequestType">
      - <xsd:sequence>
          - <xsd:element type="id:XRoadCentralServiceIdentifierType" name="server">
              - <xsd:annotation>
                    <xsd:documentation>Identity of this security server. </xsd:documentation>
                </xsd:annotation>
            </xsd:element>
          - <xsd:element type="id:XRoadClientIdentifierType" name="client">
              - <xsd:annotation>
                    <xsd:documentation>Identity of the SDSB member or subsystem
                        requesting to be a client of this security server or who needs to be
                        deleted from the list of clients of this server server.
                        </xsd:documentation>
                </xsd:annotation>
            </xsd:element>
        </xsd:sequence>
    </xsd:complexType>
</xsd:schema>
```

Kuvio 6: Esimerkki skeemasta

Uusien asiakkaiden liittyessä kyseiseen tilausjärjestelmään, tulee heiltä usein testausvaiheessa pyyntöjä koskien sanomien validointeja. Asiakkaan niin halutessa tehdään heille valmiit muuntimet asiakkaiden palvelimille, mutta kaikki asiakkaat eivät halua valmista pakettia taloudellisista syistä, vaan haluavat itse kehittää sanomansa ja muuntimensa. Skeemat ja esimerkkisanomat ovat tarjolla kaikille asiakkaille. Kumminkin sanomien validointi skeemaa vastaan koetaan välillä hankalaksi, ja siihen haluttaisin toisenlainen ratkaisu. Tähän nousi kehitysehdotuksena testipalvelimelle tehtävän validointiliittymän tai rajapinnan teko. Vaihtoehtoisesti siihen voisi kehittää jopa oman käyttöliittymän, jonka kautta asiakkaat pääsisivät kokeilemaan sanomiaan ja saamaan tarvittaessa oikeat virheilmoitukset.

5.2 Liityntäkatalogi

Liityntäkatalogi on Suomi.fi-palveluväylän palveluista eli liitynnöistä ilmoittava ja ylläpitävä taho. Katalogin tarkoituksena on auttaa palveluiden tuottajia ja toteuttajia kehittämään ja ottamaan käyttöön tehokkaampia sähköisiä palveluita. Siellä ylläpidetään Palveluväylään liittyneiden organisaatioiden tietoja. (Suomi.fi, 2020d). Sitä kautta on myös mahdollista selata organisaatioille luvitettuja WSDL-tiedostoja ja sanomatyyppejä.

5.3 Palveluväylän siirtoprotokollat

Siirtoprotokollan valinta saattaa olla asiakkaalle tärkeä tietoturvan kannalta. Erityisesti http-protokollan muuttaminen tietoturvallisemmaksi https-protokollaksi saattaa olla jopa kynnyskysymys uusien päivitysten tullessa vanhoille asiakkaille sekä uusien liittyessä palveluun. Täten oli hyvä dokumentoida, miten vaihto tapahtuu tarvittaessa nopeasti. Liityntäpalvelimen ja asiakasjärjestelmien välillä kulkevassa salatussa liikenteessä on oletuksena sallittu TLS 1.2 (Transport Layer Security) ja seuraavat PFS Cipher Suitet (Perfect Forward Secrecy).

- TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 *
- TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256 *
- TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 *
- TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384 *
- TLS_DHE_RSA_WITH_AES_128_GCM_SHA256
- TLS_DHE_RSA_WITH_AES_128_CBC_SHA256
- TLS_DHE_RSA_WITH_AES_256_CBC_SHA25
- TLS_DHE_RSA_WITH_AES_256_GCM_SHA384

* merkki tarkoittaa, ettei ole käytössä Red Hat Enterprise Linuxin kanssa, jos käytetään Open JDK:ta (Open Java Development Kit). Tämä on erittäin merkittävä asia, koska tammikuusta 2019 lähtien Oracle muutti Javan lisenssiehtoja ajaen useat tahot käyttämään Open JDK:ta. Protokollan valinta ja testaus ovat olleet myös vaikeita monessa tapauksessa. Sen takia testaus kuvattu liitteessä 1.

Https-yhteyden muodostaminen sertifikaattia hyödyntäen tuli dokumentoitua melko tarkasti komentojen kanssa, koska monella muullakin on ollut ongelmia tämän kanssa. Erityisesti EU:n laajuisen GDPR-asetuksen voimaantulo aiheutti paljon töitä monissa yrityksissä, organisaatioissa ja yksityishenkilöillä tietoturvan näkökannalta.

5.4 Kansallisen palveluarkkitehtuurin tulevaisuus

Moni kaupunki ja virasto on valtion ohjeistamana ja rahallisen tuen ohjaamana lähtenyt toteuttamaan omia Palveluväylä ratkaisujaan ja moni yrityskin on liittynyt siihen joko suoraan

tai toisen palveluntarjoajan välityksellä. Kevään 2020 Liityntäkatalogin mukaan mukana on jo 140 organisaatiota joko suoraan tai toisen liittymän kautta. Todellisuudessa luku lienee korkeampi, koska osa organisaatioista liikkuu yhden liittymän kautta yhden liittymän tiedoilla. Viro on tässä asiassa huomattavasti pidemmällä. Heidän sivujen mukaan jopa 99 % Viron valtion palveluista käyttäisi jo X-Roadiin perustuvaa teknologiaa. X-Roadin käyttö on levinnyt myös moniin muihin maihin ympäri maailmaa.

Entä teknologian suuntaviivat? Palveluväylää kehitetään jatkuvasti ja vuoden 2020 aikana julkaistava X-Roadin versio 6.24 tarjoaa myös tuen kontitetun liityntäpalvelimen hallinnalle rajapinnan kautta ja mahdollistaa sitä kautta tehtävien automatisointia. Tämän kontitetun liityntäpalvelimen tarkoitus on mahdollistaa Palveluväylän käyttöönotto ilman isompia palvelinmaksuja. Kyseessä on Docker-teknologiaa hyödyntävä ohjelmisto, joka tarjoaa tulevaisuudessa kaikki liityntäpalvelimen palvelut ja se on myös ilmainen. Huonona puolena tämä versio ei tue sanomien lokitietojen kirjoittamista missään tasolla, koska siitä vastaavaa komponenttia ei sisällytetä tässä vaiheessa kontitettuun liityntäpalvelimeen. Tuotantokäyttöön tämä tulee syksyllä 2020. (Digi- ja väestötietovirasto, 2020.)

X-Roadin versio 6.24 tuo mielenkiintoisia mahdollisuuksia kyseessä olevan tilaus- ja toimitusliittymän kehittämiselle syksyn 2020 aikana. Liityntäpalvelinten vieminen kontteihin toisi taloudellisia säästöjä alkutyön jälkeen.

Kontit ovat ohjelmiston yksikkö, joka pakkaa koodin ja sen riippuvuudet, jotta sovellus toimisi mahdollisimman nopeasti ja luotettavasti. Docker kontti-imaget eli vedokset, ovat kevyitä ja itsenäisiä suoritettavia ohjelmistopaketteja, joka sisältää kaiken tarvittavan sen suorittamiseksi. Konttien ”imageista” tulee kontteja ajon aikana Dockerilla. Kontatut ohjelmistot toimivat aina samalla tavalla infrastruktuurista riippumatta. Kontit siis eristävät ohjelmistot omaan ympäristöönsä ja varmistavat, että ohjelmistot toimivat tasaisesti. (Docker, 2020.)

6 Liittymän nykytilanne

Kuten aikaisemmin mainittiin, alkoi liittymän dokumentointi kokoamalla liittymän keskeiset osat käsitekartaksi. Tästä käsitekartasta valittiin keskeisimpiä osioita liittymän dokumentoinnin ja kehittämisen kannalta. Osasta komponenteista riitti yleiskuva, osasta yritettiin saada mahdollisimman käytännönläheistä ja yksityiskohtaista tietoa, mutta kaikista osioista yritettiin löytää kehitysehdotuksia. Tässä osiossa lähdetään ensi muodostamaan kokonaiskuvaa itse liittymästä. Sen jälkeen katsomme Amazon Web Servicesiä, tutkimme käyttöjärjestelmiin liittyviä asioita ja sitten X-Roadiin liittyviä asioita.

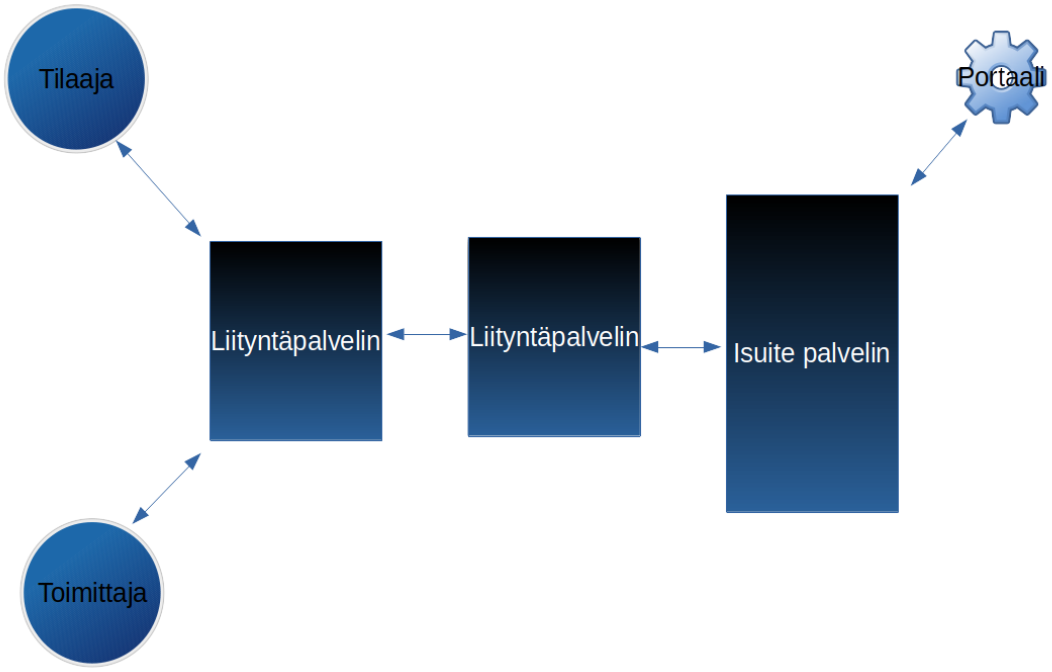
6.1 Liittymän kokonaiskuva

Kansallinen palveluväylä (KaPA) tunnetaan myös Suomi.fi-palveluväylänä. Palveluväylään tärkeänä osana kuuluvaa X-Road teknologiaa ylläpitää ja kehittää ydinkomponenttien osalta Nordic Institute for Interoperability Solution (NIIS), joka on Suomen ja Viron yhdessä perustama ja ylläpitämä voittoa tavoittelematon organisaatio. Suomen puolelta kehitystä, neuvoa ja dokumentaatiota tarjoaa Suomi.fi-palveluväylä. Kyseinen liittymä sisältää liittymän X-Road liityntäpalvelimille sekä Digian Oyj omia iSuite-nimisiä integraatio-ohjelmia, jotka toimivat integraatiovälineinä sekä Kansallisen Palveluväylän suuntaan, että myös asiakkaiden palvelimilla. Integraatio sisältää useita palvelimia sekä pilvessä, että asiakkaan ympäristöissä, X-Road-integraatioita ja iSuite-integraatioita. Valtio ohjaa ottamaan käyttöön X-Road liittymiä suosituksilla ja rahallisella tuella, joten Digia Oyj on toteuttanut näitä X-Road-liittymiä muillekin asiakkaille.

Tällä hetkellä liittymään kuuluu iSuite HUB integraatiotyökalua ajavat palvelimet, toinen tuotantoympäristölle ja toinen testiympäristölle. Sen lisäksi liittymään kuuluu kaksi eri liityntäpalvelinta, joista myös molemmista on tuotanto- ja testiympäristöt eli yhteensä neljä eri liityntäpalvelinta. Lisäksi jokaisella tavarantoimittajalla ja tilaajalla on omat palvelimet, joilla lähes kaikilla pyörii iSuite Client integraatio-ohjelma reitittämässä ja tekemässä muunnoksia sanomille. Tämän lisäksi Client myös kirjoittaa lokitiedostoa valvontaa varten sekä pyörittää muita palveluita asiakkaiden tarpeen mukaan. Yksinkertaistetun kokonaiskuvan liittymästä näkee kuviosta 5.

Sanomien kulku lähtee tilaajalta, jonka tilaussanoma menee Palveluväylälle, jossa sanomalle tehdään tarvittavat varmistukset. Sen jälkeen Palveluväylä lähettää sanoman ensimmäiselle liityntäpalvelimelle. Täällä sanoman liittyjätiedot myös tarkastetaan. Täältä sanoma reitietään toisen liityntäpalvelimen läpi iSuite HUB -integraatiotyökalulle. Täällä sanomalle tehdään vielä UBL-skeemaa vasten validointi, jonka jälkeen sanoma lähtee liityntäpalvelinten kautta asiakkaan palvelimelle. Täällä iSuite Client tekee tarvittaessa vielä muunnoksia sanomalle ja siirtää sen kohdekansioon.

Juuri liittymän kokonaiskuvan suoraviivaistamiseen löytyi muutamakin kehitysehdotus. Haastateltavilta tuli paljolti samoja ehdotuksia, joten näitä ehdotuksista puhutaan lisää haastatteluiden tuloksissa.



Kuvio 7: Yksinkertaistettu kokonaiskuva

6.2 Liittymän ympäristö

Liittymän tuotanto- ja testipalvelimet sijaitsevat Amazonin palvelimilla pilvessä. Palvelun nimi on Amazon Web Services (AWS) ja itse laskentakapasiteetin yksiköt ovat tyypiltään Amazon Elastic Compute Cloud (EC2). EC2-verkkopalvelu tarjoaa nopeasti muutettavissa olevan alustan usealla eri käyttöjärjestelmällä Amazonin pilvessä. Se on suunniteltu tarjoamaan asiakkaille instansseja, joiden kokoa ja määrää voidaan skaalata tarpeen mukaan nopeasti sekä isommaksi että pienemmäksi. Palveluita voidaan käyttää myös usealla maantieteellisellä alueella eli saatavuusvyöhykkeellä. (Amazon Web Services, Inc. 2020a)

Itse tallennustilan instansseille tarjoaa Amazon Elastic Block Store (EBS). EBS-tilaosiot ovat siis kiinni EC2-instansseissa ja ne toimivat tallennusosioina eli virtuaalisina kovalevyinä instansseille. Niiden elinkaari on riippuvainen instanssien elinkaaresta. Jokainen EBS-tilaosio replikoidaan automaattisesti sen saatavuusvyöhykkeellä, jotta esimerkiksi komponenttien viat eivät estä käytettävyyttä pitkäksi aikaa. Se tarjoaa myös skaalautuvuutta kapasiteettiin, suorituskykyyn sekä hinnoitteluun. (Amazon Web Services, Inc. 2020b.)

Itse palvelimet oli jo dokumentoitu, mutta kahden palvelimen Secure Shell -yhteyksien (SSH) avaimien nimeämisissä oli dokumentoitavaa ja korjattavaa. Secure Shell on etäkäyttöohjel-

misto, jolla voidaan ottaa salattuja yhteyksiä omasta järjestelmästä toiseen ja toisesta järjestelmästä eteenpäin tarvittaessa. SSH on alun perin Suomalaisen Tatu Ylösen kehittämä ohjelmisto.

Dokumentaation yhteydessä EC2-instanssien kuvauskentissä havaittiin epämääräisyyksiä. Nämä kentät päivitettiin vastaamaan paremmin instanssin käyttötarkoitusta. Tästä on suuri hyöty esimerkiksi varmuuskopioiden ottamisessa ja päivitysten tekimisessä.

Myös EC2-instanssien käyttöoikeusryhmät käytiin läpi. Nämä olivat hyvällä tasolla. Tietoturvasyistä niitä ei käydä tarkemmin läpi tässä opinnäytetyössä. Käyttöoikeusryhmät toimivat eräänlaisina virtuaalisina palomuureina instanssille sisään tulevan ja ulos menevän liikenteen ohjaamiseksi. Perussääntönä ryhmillä on, että kaikki sisään tuleva liikenne on kielletty, jos ei sitä ole erikseen sallittu, mutta kaikki uloslähtevä liikenne on sallittu. Kun instanssin käynnistää, voi sille määrittää yhden tai useamman käyttöoikeusryhmän. Jos ei EC2 instanssille määritä itse ryhmää, käyttää se oletusasetusryhmää. Käyttöoikeusryhmien sääntöihin voi tehdä muutoksia milloin tahansa, eikä se vaadi instanssien uudelleen käynnistystä. Yhdelle instanssille voi määrittää useamman käyttöoikeusryhmän tai yhtä ryhmää voi käyttää useammalla instanssilla samanaikaisesti. (Amazon Web Services, Inc. 2020c.)

Uuden päivityksen yhteydessä on hyvä ottaa aina varmuuskopio vanhasta versiosta, jos päivityksessä ilmenee virheitä ja joudutaan tekemään järjestelmän palautus vanhaan versioon. Tähän löytyi parikin tapaa. Kyseisessä liittymässä käytämme yleensä snapshot-tyyppistä varmuuskopiota eli tilannekuvaa. Tilannekuva ei ole perinteinen varmuuskopio, koska se ei ole AWS instanssin täydellinen kopio. Sen sijaan tilannekuva on varmuuskopio tallennusosiosta, jossa on tieto muutoksista edellisen tilannekuvan ottamisen jälkeen.

Tilannekuvan ottaminen on yksinkertainen ja nopea toimenpide. Sen takia se lisättiin vain linkkinä dokumentaatioon. Tilannekuvan saa valikosta kohdan Elastic block storen alta kohdasta Snapshots ja siellä luomalla valitsemalla create snapshot, mutta sen tilannekuvan palauttaminen toimivaksi instanssiksi onkin monimutkaisempi prosessi ja se vaati hieman dokumentointia. Tähän löytyi myös hyvä dokumentti pelkällä linkillä. Periaatteessa varmuuskopioidusta tilannekuvasta tehdään levyosia ja rikkinäisestä instanssista irrotetaan vanha osa ja uusi liitetään tilalle. Toisena vaihtoehtona varmuuskopioidusta tilannekuvasta voidaan tehdä image eli tarkka kopio ja palauttaa instanssi sitä kautta.

AWS-instansseja läpikäydessä huomattiin myös, että yhdessä testipuolen instanssissa ei ollut elastista, eli staattista IP-osoitetta. Tämä ongelma saatiin korjattua samalla. AWS käyttää elastisia, eli joustavia IP-osoitteita pilvipalveluissaan. AWS-arkkitehtuurissa asiakkailla on virtuaalisia yksityisiä pilviä (VPC) ja niissä sisällä on instansseja, joilla on julkinen IP-osoite. Tämä julkinen IP-osoite ohjaa liikennettä julkiseen Internetiin. Tämä tuottaa ongelmia siinä vaiheessa, kun instanssi käynnistetään uudelleen. Tällöin se saa uuden IP-osoitteen, jolloin

muut palvelimet, jotka ovat osana tässä integraatiossa, hukkaavat yhteyden kyseisiin AWS-instanssiin.

Kartoitettaessa AWS instanssien määriä ja tyyppejä tuli puheeksi myös näiden hinnoittelu. Tästä syystä pääsimme kartoittamisen yhteydessä tarkastamaan ja muokkaamaan instanssien kokoa ja tyyppiä rahallisen säästön aikaansaamiseksi. Tässä liittymässä hinnoittelu menee käytön mukaan. Teimme kehitysehdotuksen ja toteutimme tiputtamalla instanssien luokkaa suorittimien ja muistin osalta. Näin saavutimme taloudellista säästöä. Kun asiakkaiden määrä liittymässä kasvaa, saatamme joutua kasvattamaan tätä takaisin. Itse kustannusten laskemiseen on AWS-palvelussa oma laskuri. Instanssien pienentämisen jälkeen liittymän toimivuus varmennettiin tarkkailemalla palvelimilla prosessorin- ja muistin kuormitusta sekä tarkkailemalla lokitiedostoja. Tietyissä yhteysongelmissa myös integraatio sovellus lähettäisi virheilmoituksia ongelmista.

6.3 Käyttöjärjestelmät

Kyseinen integraatio Kansalliseen palveluväylään (KaPA) on toteutettu useilla eri palvelimilla käyttäen Ubuntu Linux ja Red Hat Linux -käyttöjärjestelmiä. Suomi.fi-palveluväylän testi- ja tuotantoympäristöihin voi liittyä vain virallisesti tuetuilla Linux jakeluilla ja ne ovat tällä hetkellä Ubuntu 14.04 LTS (Long Term Support) ja RHEL 7 (Red Hat Enteriprise Linux). Palveluväylän toimivuutta ei voida taata CentOS-Linuxilla, koska sen toimivuutta ei ole testattu. Kuitenkin kehitysympäristön puolella myös virallisten Linux-distrojen eli Linux-jakeluiden, kuten edellä mainittu CentOS, käyttö on periaatteessa mahdollista, mutta se tapahtuu aina organisaation omalla vastuulla. Näiden muiden distrojen ongelmatilanteisiin ei tarjota tukea palveluväylän ylläpidon puolesta. Ubuntun osalta on päätetty, ettei tukea tarjota Ubuntu 16 -versioille, vaan seuraava tuettu versio on Ubuntu 18. Keväällä 2020 julkaistavan Ubuntu 20.04 LTS version tuesta ei ole vielä tiedotetta.

Red Hat Enterprise Linux (RHEL) on Red Hatin (Red Hat Inc.) kehittämä kaupallisille markkinoille suunnattu Linux-jakelu. Jakelun kaupallinen kehitys on mahdollistanut korkeatasoisen turvatekniikan ja käytännöt tietojen suojaamiseksi ja tunkeutumisten estämiseksi. Käyttämällä Red Hat Enterprice Linuxia EC2-instansseilla voi hyödyntää myös Red Hatin omaa AWS-tukea.

Ubuntu on Debian-pohjainen Linux-käyttöjärjestelmä, joka on suunnattu työasemille, kotikäyttöön ja verkkopalvelimille ja jopa älypuhelimille. Ubuntu-projekti on sitoutunut avoimen lähdekoodin ohjelmistoihin ja ratkaisuihin, tosin sen sitoutumisesta ja suuntautumisesta on ollut toisinaan väittelyä. Ubuntu on suosituin alusta Linux-pinnoille ja AWS:llä löytyy satoja

sovelluksia ja sovelluspinoja Ubuntulle. Ubuntu ja siitä johdettu Linux Mint ovatkin kotitietokoneilla yleisimmät Linux-jakelut.

Yksi tärkeimmistä löydöksistä ja kirjoittamisessa oli liittymän tärkeimpien Servicien eli Linux-palvelujen löytäminen. Näillä palveluilla ohjataan itse palveluväylää ja vikatilanteissa yleensä riittää vian löytäminen ja palvelun uudelleen käynnistäminen. Taulukossa 1 on listattu tärkeimmät palvelut (service):

Palvelu	Tarkoitus	Loki
xroad-confclient	Client process for the global configuration distributor	/var/log/xroad/configuration_client.log
xroad-jetty (Ubuntu) xroad-jetty9 (RHEL)	Application server running the user interface	/var/log/xroad/jetty/jetty.log
xroad-proxy	Message exchanger	/var/log/xroad/proxy.log
xroad-signer	Manager process for key settings	/var/log/xroad/signer.log
nginx	Web server that exchanges the services of the user interface's application server and the message exchanger	/var/log/nginx/
postgresql	Tietokantapalvelin	/var/log/postqresql/

Taulukko 1: Tärkeimmät palvelut

Erityisesti päivitysten yhteydessä on välillä tapahtunut niin, ettei jokin palvelu ole noussut ylös ja sen on joutunut käynnistämään komentoriviltä. Vikaherkimmiksi näistä ovat osoittautuneet sanomien välityksestä vastaava xroad-proxy sekä käyttöjärjestelmää ylläpitävä xroad-jetty. Itse palvelujen nostamiseen käytettävät komennot ovat Linuxin normaalien palvelujen komentoja, jotka myös dokumentoitiin ohjeisiin. Yleensä liittymä on erittäin vakaa ja vikasietoinen. Virhetilanteen ilmaantuessa on palvelimen uudelleen käynnistäminen lähes aina ratkaissut ongelman.

Liittymää läpikäydessä tuli eteen muutamia poikkeuksia käyttöjärjestelmissä, joten niiden yhtenäistämisestä tuli samalla kehitysehdotus. Ubuntu Linux on myös hieman halvempi EC2-instanssien käyttöjärjestelmänä, joten siitä seuraisi rahallista säästöä. Tämä kehitysehdotus on

vielä toteuttamatta. AWS tukee myös muita Linux-käyttöjärjestelmiä, mutta niitä ei tarkasteltu Kansallisen palveluväylän tuen puutteen takia.

6.4 Sanomien muoto

Universal Business Language (UBL) on vakiinnutettujen tavanomaisten sähköisten XML-muotoisten yritysmaailmaan suunnattujen dokumenttien avoin kirjasto. Se sisältää erityisesti ostotilaukset, laskut, rahtikirjat ja tilaus- ja toimitusvahvistukset. UBL-muodon ja kirjaston on kehittänyt OASIS Technical Committee, johon kuuluu useita teollisuuden tietojen standardoinnista vastaavia järjestöjä. Sen tarkoitus oli alun perin poistaa faksi- ja paperipohjaisen tilausten ja asiakirjojen jatkuva uudelleen syöttäminen ja tarjota helppo pääsy sähköiseen kaupankäyntiin erityisesti pienille ja keskisuurille yrityksille. UBL-versio 2.1 hyväksyttiin OASIC-standardiksi jo 2013. UBL-muodosta löytyy jo kaksi uudempaa versiota, mutta ne enemmänkin lisäävät asiakirjatyyppejä kuin pakottavat muodon muutokseen. UBL juontaa juurensa kaupan alla paljon käytettyihin EDI-standardeihin ja myös muihin XML-pohjaisiin standardeihin.

Liittymässä liikkuvien tilaukseen ja toimitukseen liittyvien sanomien muotona käytetään muokattua UBL 2.1-muotoa. Itse sanomat ovat XML-tiedostoja, mutta sisältö on siis muokatun UBL 2.1 muodon vaatimusten mukaisia sisältäen XML-vaatimukset. Kevään 2020 aikana päivitettiin skeemaan muutama uusia kenttä, pakollinen kenttä asetettiin vapaaehtoiseksi ja muutama kenttä pakolliseksi. Pakollisten kenttien lisääminen on iso muutos, koska se vaatii muutoksia kaikilta asiakkailta tai ylläpitäjän toimesta asiakkaan sanomiin. Nämä muutokset tulivatkin asiakkaiden ja toimijoiden yhteyskokouksessa ja muutokset dokumentoitiin samalla. Sanomat validoidaan skeemaa vasten eikä niitä päästetä iSuite HUBilta eteenpäin sekä virheestä tiedotetaan asiakasta automaattisesti.

6.5 iSuite HUB ja client

ISuite HUB on Digia Oyj:n tuottama ja erittäin muokattava integraatiotyökalu erilaisten järjestelmien integroimiseksi. Se on helppokäyttöinen ja muokattavissa asiakkaiden tarpeiden mukaan. ISuite on Java-pohjainen, joten sen on alustariippumaton ja sopii loistavasti integraatiovälineeksi asiakkaiden palvelinten ja KaPAn välille. ISuite HUB sisältää käyttöliittymän, joka mahdollistaa sanomien tarkkailun, käsittelyn ja uusien integraatioiden luomisen. Täten ohjelmointia ei yleensä tarvita uusien muunnosten ja integraatioiden tekemiseksi. Se tarjoaa myös keskitetyn lokitiedostojen luomisen ja tarkkailun. ISuite HUBia voi käyttää omalta palvelimelta tai käyttöliittymälle voi kirjautua etänä. Se sopii myös loistavasti käytettäväksi pilvipohjaisissa ratkaisuissa. Itse iSuite HUB ja iSuite Client ovat Digian omia ohjelmistoja, joten ne ovat myös dokumentoitu hyvin. Itse ohjelmistojen dokumentointia ei käsitelty tässä opinnäytetyössä.

ISuite Client on palvelimille erittäin pienen jalanjäljen tekevä kevyt integraatiotyökalu. Se on myös erittäin muokattava työkalu. Opinnäytetyön kohteena olevassa liittymässä sitä käytetään pääasiassa asiakkaiden palvelimilla tilaus- ja toimitussanomien vastaanottamiseen, sanomien muokkaamiseen XML muotoon, SOAP- ja X-Road kehysten lisäämiseen, sekä http(s)-muotoiseen lähettämiseen liityntäpalvelimille ja sieltä asiakkaiden suuntaan takaisin.

Vaikka itse iSuite Client on dokumentoitu hyvin, sen asentamisesta palvelimille, mutta erityisesti Linux-palvelimille, tuli paljon huomioita ja tärkeimpiä kohtia ohjeistusten dokumentaatioon. Kuten aikaisemmin mainittu, tuli tämä dokumentaatio yrityksen sisäiseen verkkoon eikä yksityiskohtia siihen liittyen mainita tässä opinnäytetyössä. Yhtenä tärkeimpänä nostona voidaan kumminkin mainita liityntäpalvelimilta tapahtuvien WSDL-kutsujen muodon erilaisuus Windows- ja Linux-palvelinten välillä.

ISuite HUB ja iSuite Client toimivat tässä liittymässä, monen muun asian lisäksi, niin sanottuina sovitinpalveluna eli ne tekevät tarvittavat muutokset sanomiin. Jos ei integraatio-ohjelmisto hoitaisi tätä, voitaisiin se toteuttaa erillisenä sovitinpalveluna. Esimerkiksi GitHubissa on Digi- ja väestötietoviraston (DVV) ylläpitämä X-Road sovittimen lähdekoodi osoitteessa on https://github.com/petkivim/x-road-adapter-example

Dokumentaation varmistamiseksi on kehitysehdotuksena laittaa asiakkuuteen kuulumaton integraatiokehittäjä tiimistämme asentamaan palveluun liittyvän asiakkaan tilaus- ja toimitussanomien tietovirta.

7 Haastattelun tulokset

Haastatteluiden kommentteja ei julkaista suoraan opinnäytetyössä, vaan niistä kerrotaan vain tiivistelmä. Avoimessa haastattelussa kysyttiin viideltä asiantuntijalta, jotka kaikki ovat aikaisemin toimineet jatkuvien palveluiden tiimissä ja joilla oli kokemusta kyseisestä liittymästä. Kysymys oli, miten kehittäisit ja parantaisit liittymää nyt ja tulevaisuudessa?

Yleinen kehitysidea oli ottaa toinen liityntäpalvelin pois käytöstä. Tämä olisi mahdollista ja säästäisi palvelinten ylläpitokustannuksissakin. Se vaatisi neuvotteluja muutaman asiakkaan kanssa, mutta idea olisi erittäin toteutuskelpoinen. Toteutus vaatisi liikenteen uudelleen reitittämistä toisen liityntäpalvelimen läpi, mikä ei työmääräarviona olisi mahdoton ajatus.

Toinen usealla kehittäjällä esiin noussut kehitysidea oli suoraviivaistaa liittymää. Tällä hetkellä liikenne iSuite HUB -integraatio-ohjelmasta ohjataan takaisin liityntäpalvelimella ja sitä kautta vasta asiakkaalle. Muutosehdotuksena tuli, että iSuite HUBilta ohjattaisiin sanomat suoraan asiakkaiden palvelimille. Toteutus olisi helppo ja nopea tehdä iSuitella, mutta se vaatisi kaikkien asiakkaiden palvelimilta uudet palomuuriavaukset.

Kolmas kehitysehdotus oli myös saman Linux-jakelun käyttäminen kaikilla palvelimilla. Vaihtoehtoina olevista kahdesta valittavana olevasta jakeluista, eli Ubuntu ja RHEL, tähän olisi paras vaihtoehto Ubuntu-Linux kustannussyistä. Molemmat ovat tietoturvallisia, hyvin ylläpidettyjä ja omaavat laajan ohjelmistotuen.

Viimeisenä useammalta kehittäjältä tullut idea olisi palvelinten lokikirjoituksen parantaminen. Tästä löydettiin pieni sovellus Palveluväylän puolesta ja oman tiimin taholta kehitteillä on lokitietoja jatkuvasti tarkkaileva sovellus.

8 Yhteenveto ja johtopäätökset

Tutkimuksella oli tarkoitus kartoittaa ja dokumentoida liittymän tärkeimpiä komponentteja sekä löytää kehitettäviä kohteita tämän koko liittymän saralta. Pyrkimys oli hyödyttää toimeksiantajayritystä ja sen jatkuvien palvelujen tiimiä, joka valvoo, ylläpitää ja kehittää tätä liittymää. Opinnäytetyön edetessä nousi kiinnostus kehittää ja laajentaa Palveluväylän käyttöä tulevaisuudessa.

Integraatioilla on merkittävä rooli tämän päivän organisaatioiden it-järjestelmissä ja sen tarve kasvaa koko ajan. Integraatioilla tehostetaan, automatisoidaan ja helpotetaan jokapäiväistä työtä. Moni kaupunki onkin valtion ohjaamana lähtenyt toteuttamaan omia Palveluväylä ratkaisujaan ja moni yritys on liittynyt joko suoraan tai toisen palveluntarjoajan välityksellä. Kevään 2020 Liityntäkatalogin mukaan mukana on jo 140 organisaatiota joko suoraan tai toisen liittymän kautta. Todellisuudessa luku lienee korkeampi, koska osa organisaatioista liikkuu yhden liittymän kautta yhden liittymän tiedoilla. Digia Oyj:lle onkin tärkeää pysyä tässä kehityksessä mukana ja panostaa tämän teknologian osaamiseen.

Tutkimuksen materiaalina käytettiin laajaa määrää alan kirjallisuutta sekä Internetin sivustoja. Muiden yläpuolelle nousi Suomi.fi sivusto, joka tarjoaa ohjeistuksen ja materiaalin liityntäpalvelimille. Materiaalia saatiin myös pienimuotoisesta haastattelusta. Osa haastatteluiden kehitysehdotuksista oli täysin samoja kuin omat havainnot aikaisemmin. Tutkimuksessa nousseita kehitysehdotuksia olivat AWS EC2 -instanssin pienennys kustannus syistä, kumminkaan vaarantamatta sanomaliikennettä.

Opinnäytetyön hyötyjä tarkastellessa tulee ensimmäisen mieleen oman osaaminen kehittyminen. Erityisesti tutkimuskysymys, miten liittymän osa-alueita voitaisiin kehittää, tuotti useita hyötyjä ja kehitysehdotuksia:

- AWS pilvipalvelimen koon pienentäminen tuoden taloudellista säästöä. Tämä toteutettiin jo.

- Liityntäpalvelinten valvonnan parantaminen. Tähän löytyi yksi http yhteyttä valvova komentorivipohjainen ohjelma ja tekeillä on oman tiimin puolesta lokitiedostoja valvova ja sieltä virheilmoituksista ilmoittava sovellus. Tämä työ on aloitettu ja osittain toteutettu.
- Secure Shell -yhteyksien (SSH) avaimien nimeämisissä oli ristiriita, joka korjattiin samalla.
- Testipalvelimelle tehtävä muokattujen UBL 2.1 muotoisten tilaus- ja toimitussanomien validointi skeemaa vasten. Joko suorana liittymänä integraatio sovellukselle tai rajapintakutsuna. Tämä ehdotus on suunnitteluasteella.
- Linux-jakelujen yhtenäistämisenä ylläpidon helpottamiseksi ja taloudellisen säästön aikaansaamiseksi. Tämä tulee olemaan isompi urakka, joten sitä suunnitellaan ja toteutetaan myöhemmin.
- Liittymän suoraviivaistaminen suoralla yhteydellä integraatiosta. Tämä kehitysehdotus vaatii keskustelua tiimin sisällä.
- Toisen liityntäpalvelimen poisto reitittämällä liikenne uudestaan. Tämä ehdotus on vasta alustava ja vaatii suunnittelua ja keskustelua myös asiakkaiden kanssa.
- Palveluväylän PIN-koodin automaattinen syöttäminen palvelimen käynnistymisen yhteydessä. Tällä varmistetaan osaltaan liittymän nousemien palvelimen uudelleen käynnistymisen yhteydessä ilman manuaalista syöttämistä.
- Liityntäpalvelinten kontteihin vienti syksyn / talven 2020 aikana. Tämä odottaa syksyllä julkaistavan X-Road 6.24 kontitettua liityntäpalvelinta

Opinnäytetyön negatiivisena puolena olisin halunnut tuoda esiin paljon enemmän matalan tason löydöksiä ja ohjeistuksia tässä raportissa, mutta sekä tietoturvaan, että yrityssalaisuuksiin liittyvistä syistä johtuen, tuli opinnäytetyön raportista yleisluontaisempi muutamia kohtia lukuun ottamatta.

9 Oman oppimisen arviointi

Tiedon etsiminen ja analysointi opinnäytetyöhön auttoi ymmärtämään erityisesti Kansallisen palveluväylän rakennetta ja siihen liittyvän integraation valvontaa. Tässä opituilla tiedoilla onnistuisi palveluväylän ohjelmiston asentaminen ja sen liittäminen KaPA:n sekä monen muun ongelmatilanteen selvittäminen. Myös jo entuudestaan tutuista integraatio sovelluksista löytyi uusia tapoja sanomien käsittelyyn. Aika paljon tuli Linux osaamista opeteltua koskien yhteyksien kokeilua ja palvelinten ylläpitoa. Opinnäytetyössä opitut asiat ovat tukeneet osaamistani jokapäiväisessä työskentelyssäni integraatioiden parissa. Kokonaan uutena teknologiana tulee

Docker-konttiteknologian opettelu koskien löydettyä isoa kehitysehdotusta. Kirjallisissa materiaaleissa tuntui olevan suurta hypeä mikropalveluiden osalta. Siihen panostan myös tulevaisuudessa. Erityisesti yhdistelemällä niitä ESB-teknologian kanssa.

Lähteet

Painetut

Bryman, A., P 2008. Social Research Methods. 3. painos. New York: Oxford University Press.

Bussler, C., P 2003. B2B Integration. 1. painos. New York: Springer.

Carsten H., Uwe Z., P 2010. Process-Driven SOA: Patterns for Aligning Business and IT. Boca Rotan: CRC Press.

Heikkilä, T., P 2004. Tilastollinen tutkimus. 5. painos. Helsinki: Edita.

Hirsjärvi, S., Remes, P. & Sajavaara, P. 2009. Tutki ja kirjoita. 15. painos. Jyväskylä: Gummerus Kirjapaino Oy.

Josuttis, N., P 2007. SOA in practice. 3. painos. Sebastopol, CA: O'Reilly Media

Kananen, J., P 2014. Toimintatutkimus kehittämistutkimuksen muotona. Miten kirjoitan toimintatutkimuksen opinnäytetyönä? Jyväskylä: Jyväskylän ammattikorkeakoulu

Linthicum, D., P 2000. Enterprise Application Integration. 3. painos. Boston: Addison-Wesley Professional

Ojasalo, K., Moilanen, T. & Ritalahti, J. 2014. Kehittämistyön menetelmät. Uudenlaista osaamista liiketoimintaan. Helsinki: SanomaPro Oy

Tähtinen, S. 2005. Järjestelmäintegraatio: Tarve, vaihtoehdot, toteutus. Helsinki: Talentum.

Sähköiset

Amazon Web Services, Inc. 2020a. Replacing an Amazon EBS volume using a previous snapshot. Viitattu 11.6.2020. https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ebs-restoring-volume

Amazon Web Services, Inc. 2020b. Amazon EC2 security groups for Linux instances. Viitattu 9.5.2020.
https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ec2-security-groups

Amazon Web Services, Inc. 2020c. Amazon EC2 features. Viitattu 9.5.2020.
https://aws.amazon.com/ec2/features/

Christudas, B. 2008. Service Oriented Java Business Integration. Viitattu 17.4.2020.
https://subscription.packtpub.com/book/application_development/9781847194404

Digi- ja väestötietovirasto, 2020. Palveluväylän saa syksyllä käyttöön pilvipalveluna – investoinnit palvelimiin eivät enää välttämättömiä. Viitattu 20.5.2020.
https://dvv.fi/artikkeli/-/asset_publisher/palveluvaylan-saa-syksylla-kayttoon-pilvipalveluna-investoinnit-palvelimiin-eivat-enaa-valttamattomia

Docker. 2020. What is a Container? Viitattu 20.5.2020.
https://www.docker.com/resources/what-container

Fowler M. 2014. Microservices. Viitattu 7.5.2020.
https://martinfowler.com/articles/microservices

Jyväskylän yliopisto. 2015a. Päivitetty 23.4.2015. Määrällinen tutkimus. Viitattu 5.5.2020
https://koppa.jyu.fi/avoimet/hum/menetelmapolkuja/menetelmapolku/tutkimusstrategiat/maarallinen-tutkimus

Jyväskylän yliopisto. 2015b. Päivitetty 23.4.2015. Laadullinen tutkimus. Viitattu 5.5.2020
https://koppa.jyu.fi/avoimet/hum/menetelmapolkuja/menetelmapolku/tutkimusstrategiat/laadullinen-tutkimus

Kivisaari T. 2016. Digiarjessa blogi. Hallitsetko API:si? Viitattu 17.5.2020.
http://blog.digia.com/hallitsetko-apisi

Suomi.fi. 2020a. Suomi.fi-palveluväylä. Luotettavaa tiedonsiirtoa. Viitattu 19.5.2020.
https://palveluhallinta.suomi.fi/fi/sivut/palveluvayla/esittely

Suomi.fi. 2020b. Suomi.fi-tuki. X-Road-tiedonsiirtoprotokolla. Viitattu 19.5.2020.
https://palveluhallinta.suomi.fi/fi/tuki/artikkelit/592bf54a03f6d100018db5d4

Suomi.fi. 2020c. Suomi.fi-tuki. Asiakasjärjestelmän liittäminen liityntäpalvelimeen. Viitattu 19.5.2020.
https://palveluhallinta.suomi.fi/fi/tuki/artikkelit/592fbd1603f6d100018db5f8

Suomi.fi. 2020d. Suomi.fi-Liityntäkatalogi. Hakemisto Suomi.fi-palveluväylän liitynnöistä. Viitattu 19.5.2020.
https://liityntakatalogi.suomi.fi/

Kuviot

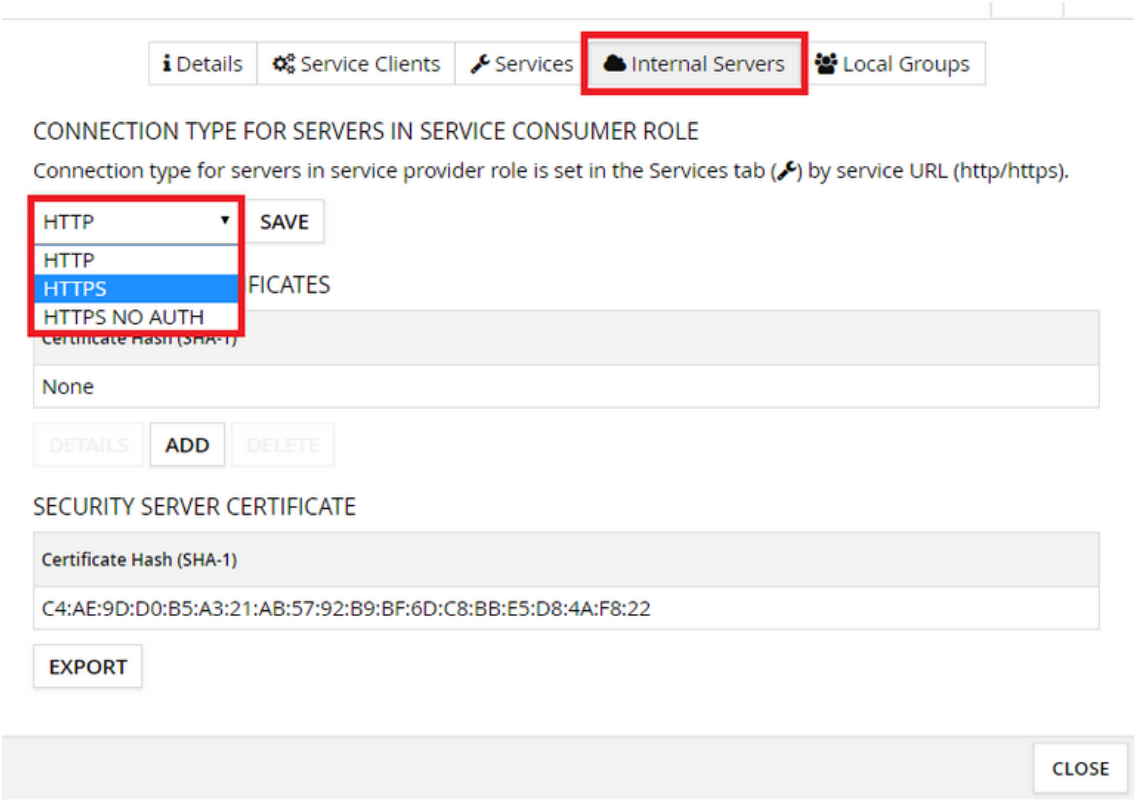
Taulukot

Liitteet

Liite1: http(s)-protokollan valinta

Liityntäpalvelimen käyttöliittymältä protokolla valitaan kohdasta Connection type for servers in service consumer role. Vaihtoehtoina on kolme eri arvoa, jotka eivät olleetkaan ihan niin yksiselitteisiä kuin olisi voinut olettaa.

- HTTP: Asiakaspalvelin voi tehdä kutsuja joko http- tai https-protokollilla ilman, että varmennetta tarvitaan.
- HTTPS: Kutsuessa palvelun täytyy esittää TLS-asiakasvarmenne, joka on listattuna liityntäpalvelimen käyttöliittymän kohdassa Internal TLS Certificates.
- HTTPS NO AUTH: Asiakaspalvelimen täytyy esittää asiakasvarmenne siitä huolimatta, että liityntäpalvelin ei tarkista varmennetta Internal TLS Certificates -listasta, mutta ilman tarkistusta tähän kelpaa lähes mikä tahansa sertifikaatti. Myös itsetehty (self-signed).



Kuvio 8: http(s) valinta ja sertifikaatti (Suomi.fi)

Asiakasjärjestelmän ja liityntäpalvelimen välisen liikenteen salaava sertifikaatti lisätään TLS Certificates -kohdan alla olevasta ADD-painikkeesta. Tämän jälkeen käyttöliittymä ohjeistaa,

miten asiakasjärjestelmän varmenne ladataan liityntäpalvelimille. Varmenteen lisäämisen jälkeen tulee Internal TLS Certificates -listaukseen tiivistesumma varmenteesta. (Suomi.fi, 2020c.)

Asiakasvarmenteen voi halutessaan testata melko helposti Linuxin komentoriviltä. Ensin luodaan uusi asiakasjärjestelmän yksityinen avain seuraavalla komennolla:

openssl genrsa -out clientprivatekey.pem 2048.

Kuten edellä mainittiin, voi palvelussa käyttää niin sanottua self-signed avainta, joten tässä testissä voidaan käyttää sitä. Se luodaan komennolla:

openssl req -new -x509 -key clientprivatekey.pem -out clientcert.pem -days 365.

Komennon perässä olevien päivien määrää voidaan säätää. Tästä on hyötyä erityisesti, jos asiakas haluaisi tehdä oman sertifikaattinsa ilmaiseksi.

Tämän jälkeen kirjaudutaan liityntäpalvelimen käyttöliittymälle. Sieltä valitaan testattavan alijärjestelmän asetukset ja sen internal servers -välilehti. Vaihda connection type for servers in service consumer role -asetuksen arvoksi testatuksen kohteena oleva https. Lisää luomasi clientsert.pem-tiedostoon tallennettu varmenneavain internal TLS certificates -listaan. Kehitysympäristön liityntäpalvelin tarjoaa testipalvelua getRandom, jolla voidaan testata yhteyttä sillä alijärjestelmällä, jolle asiakasvarmenne luotiin. Testaa yhteyttä komennolla Linuxin curl-komennolla ja getRandom xml-tiedostolla:

curl -E ./clientcert.pem --key ./clientprivatekey.pem -k -d @getRandom.xml --header "Content-Type: text/xml" -X POST https://{host}/

Vertailun vuoksi yritä samaa kutsua ilman varmenneavainta, mutta käytä curl-komennon attribuuttia -k, jottei komento verifioisi varmennetta:

curl -k -d @getRandom.xml --header "Content-Type: text/xml" -X POST https://{host}/

Lopputuloksena pitäisi saada virheilmoitus:

Server.ClientProxy.SslAuthenticationFailedClient (SUBSYSTEM:FI-DEV/GOV/0245437-2/TestClient) specifies SSLAUTH but did not supply SSL certificate

Liite 2: havaittuja yksityiskohtia skeemojen käytöstä

WSDL-dokumenteissa on käytettävä document/literal-sidontaa (binding style=”document”; use=”literal”). Palvelukutsujen sidontatyylin (binding style) on oltava document. Use-attribuutin arvon on puolestaan oltava literal.

XSD-tiedostoja käytettäessä pitää huomioida, että xml:include ei toimi, koska käyttäjällä ei ole pääsyä WSDL:n tarjoajan tietojärjestelmään. Sen sijaan xml:import toimii, mutta XSD-tiedosto pitää tarjota julkiseen internetiin ja importissa pitää antaa absoluuttinen osoite eikä vain relatiivista osoitetta. Esimerkiksi tämä on oikea tapa:

<xsd:import namespace="xyz" schemaLocation="http://palvelu-x.yritys.com/model.xsd"/>

Tämä taas on väärä tapa:

<xsd:import namespace="xyz" schemaLocation="model.xsd/>