LAB University of Applied Sciences

Siivouskutsujärjestelmän integraatio

LAB-ammattikorkeakoulu
Insinööri (YAMK), IoT:stä tekoälyyn
2022
Juha Stenberg

Tiivistelmä

Tekijä(t)	Julkaisun laji	Valmistumisaika
Stenberg, Juha	Opinnäytetyö, YAMK	Syksy 2022
	Sivumäärä	
	48	

Työn nimi

Siivouskutsujärjestelmän integraatio

Tutkinto ja koulutusala

Insinööri (ylempi AMK), IoT:stä tekoälyyn

Toimeksiantajan nimi, titteli ja organisaatio (jos opinnäytetyöllä on toimeksiantaja)

2M-IT Oy

Tiivistelmä

Nykyisin leikkaussalien ohjauspaneelit ovat toiminnanohjausjärjestelmiä, jolla hallitaan leikkausalin toimintoja. Uutena ominaisuutena oli tarve integroida SMS-viestien lähettäminen leikkaussalin siivouspyynnöistä reaaliaikaisesti helpottamaan henkilöstön kommunikointia siivouksista. Siivouskutsujärjestelmän integraatio -kehittämishankkeen tavoitteena oli toteuttaa tekstiviesteihin perustuva viestintäjärjestelmä.

Hankkeessa integroitiin yhteen leikkaussalien ohjausjärjestelmä, integraatioalusta ja SMS-palvelu. Työ sisälsi suunnittelun, arkkitehtuurin, toteutuksen, testauksen, käyttöönoton ja jatkuvaan palveluun siirtymisen vaiheet. Työssä on esitelty suunnittelua ja toteutettua siivouskutsujärjestelmää jatkokehitysehdotuksineen. Ensemble integraatioalustalla ohjauspaneelin ja SMS-viestien käsittelyt ovat ohjelmoitu ObjectScript -ohjelmointikielellä.

Kehittämishankkeen on suunniteltu valmistuvan ja käyttöön otettavan uuden sairaalan leikkaussalien aloittaessa toimintansa loppuvuodesta 2022.

Asiasanat

Ensemble, integraatio, leikkaussali, ObjectScript, REST, SMS

Abstract

Author(s)	Type of Publication	Published
Stenberg, Juha	Master's Thesis	Fall 2022
	Number of Pages	
	48	

Title of Publication

Integration of the cleaning request system

Degree and field of study

Master of Engineering, From IoT to AI

Name, title and organisation of the client (if the thesis work is commissioned by another party)

2M-IT Oy

Abstract

Nowadays operating rooms control panels are enterprise resource planning systems that manages the operations of the operating rooms. A new feature was the need to integrate the sending of SMS messages about cleaning requests in the operating room in real time to facilitate the staff's communication about cleanings. The goal of the integration of the cleaning request system development project was to implement a system based on text messages.

In the project the operating room control system, the integration platform and the SMS service were integrated together. The work included the phases of design, architecture, implementation, testing, introducing of the system and transition to continuous service. In the development project presents the design and implementation of SMS cleaning request system with suggestions for further development. On the integration platform handling of the control panel and SMS messages are programmed using ObjectScript language.

The project is planned to complete and put into service when the operating rooms of the new hospital begin their operations in end of 2022.

Keywords

Ensemble, integration, ObjectScript, SMS, REST, operating room

Sisällys

LYHENTEET JA TERMIT

API	Application Programming Interface, Sovellusohjelmointirajapinta
DVV	Digi- ja väestötietovirasto
Ensemble	InterSystemsin integraatioalusta
HTTP	Hypertext Transfer Protocol, hypertekstin siirtoprotokolla
HVA	Hyvinvointialue
JSON	JavaScript Object Notation, avoimen standardin tiedostotyyppi
REST	Representational state transfer, ohjelmistoarkkitehtuurityyli rajapintakuvaukseen
SMS	Short Message Service, tekstiviesti
XML	Extensible Markup Language, merkintäkielien standardi

1 Johdanto

Sosiaali- ja terveydenhuollon tietojärjestelmäuudistukset takaavat valtavan kasvun alalla, mikä lisää tarvetta toteuttaa mitä erilaisimpia integraatiota järjestelmien välille (Hiltula 2021). Tässä kehittämishankkeessa on tutkittu ja toteutettu 2M-IT:n toimesta uuteen sairaalaan siivouskutsujärjestelmän integraatio. Kehittämishankkeen tuottaman toiminnallisuuden on tarkoitus parantaa sairaalan leikkaussalien viestintää, kunnossapitoa ja huoltamista lähes reaaliaikaisesti siivoustarpeisiin vastaten. Toteutus on rakennettu niin, että sitä voidaan monistaa uusiin leikkaussaleihin joko sellaisenaan tai osittain, mikäli laitetoimittaja tai viestipalveluntuottaja ei ole sama.

Integraatio tarkoittaa yleisesti eri tekniikoilla tai alustoilla toteutettujen ohjelmistojen tai järjestelmien toisiinsa liittämistä, jolloin liitetyt osat kommunikoivat keskenään. Useimmiten integraation suurin hyöty syntyy automatisoinnista, joka mahdollistaa sen, että laitteistojen ja ohjelmiston välitykseen ei tarvita enää manuaalista työtä. Automatisointi yleensä minimoi inhimilliset virheet, mahdollistaen kuitenkin integraatiovalvonnan lähes reaaliaikaisesti. Näin ollen manuaalista työtä tarvitaan vain häiriötilanteiden ratkaisemiseen, mikä säästää resursseja. Eri järjestelmien välinen integrointi varmistaa tiedonkulun sekä tiedonliittymisen toisiin järjestelmiin. (Alfame 2018.)

Kehittämishanke oli toimitusprojektimuotoinen ja sen toteutuksen eri vaiheet käydään työssä läpi koko ketjun osalta lähdejärjestelmästä kohdejärjestelmään, selventäen arkkitehtuuria, REST- ja HTTP-sanomamalleja ja ObjectScript-ohjelmointia. Tehdyn integraation ansiosta aiemmin soittamalla tehty laitoshuoltajien kutsuminen leikkaussaleihin automatisoitiin SMS-viestinnäksi, jolloin työaika saatiin kohdistettua tehokkaammin leikkaussalien muuhun toimintaan. Työn keskiössä on leikkaussaleissa olevien ohjauspaneelien siivouskutsujen käsittely ja konvertointi Ensemble-integraatioalustalla sekä SMS-viestien generoiminen. Kehittämishankkeen tarkoituksena oli rakentaa järjestelmien välille toimiva integraatio, jotta leikkaussalien näyttöpaneelista saataisiin viestitettyä siivouksien tilannetiedot. Uusi sairaala aloitti toimintansa vuoden 2022 lopussa ja järjestelmä haluttiin tuotantokäyttöön toiminnan alkaessa.

Työ rajattiin koskemaan SMS-viestien lähettämistä, koska mahdollinen reaaliaikainen leikkaussalien siivoustarpeiden tilannekuvanäyttö toteutetaan jatkokehityskohteena ja erillisenä projektina. Tällä hetkellä jokaisen leikkaussalin näyttöpaneeli näyttää SMS-viestien viimeisimmän tilannetiedon. Asiakkaan tuotannossa käyttämä Ensemble versio oli vuodelta 2016, joten uusimpien IRIS-versioiden tukemien toiminnallisuuksien käyttäminen ObjectScript-koodauksessa ei ollut kaikilta osin mahdollista.

2 Tutkimuskysymys ja tutkimusmenetelmä

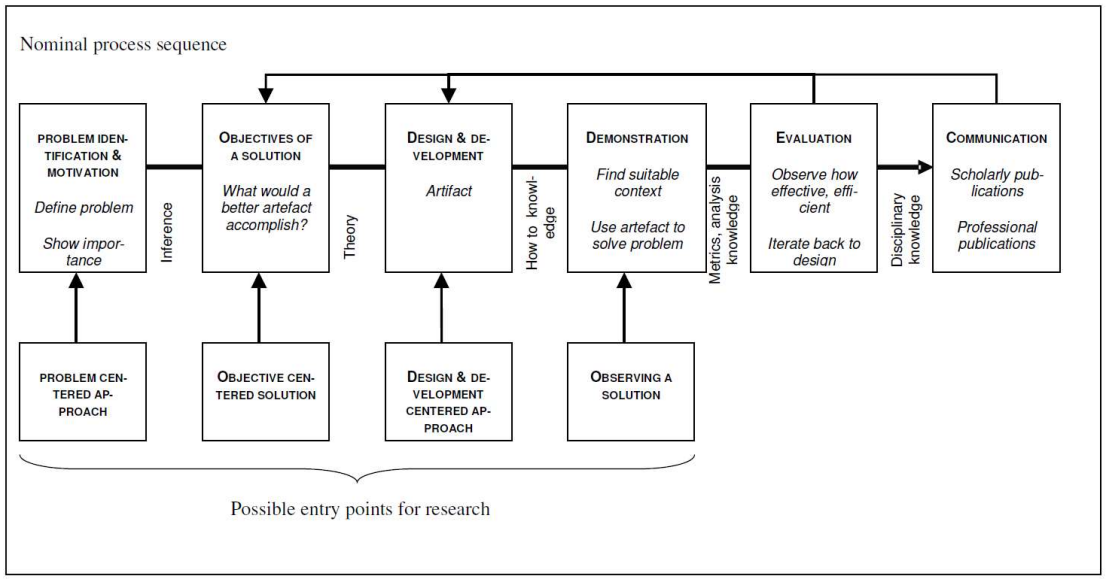
Kehittämishankkeen päätutkimuskysymykset olivat seuraavat:

- Miten järjestelmän arkkitehtuuri on toteutettu?
- Miten leikkaussalin siivouskutsujärjestelmän ohjauspaneeli integroidaan teknisesti Ensemble-integraatioalustaan?
- Miten integraatioalustalla ohjelmoidaan ObjectScript-kielellä leikkaussalin siivouspyynnöstä REST-sanomasisältöinen SMS-viesti?
- Miten välitetään halutunlainen SMS-viesti operaattorin palveluun?

Kehittämishankkeen tutkimusmenetelmäksi valittiin suunnittelutieteellinen tutkimus eli englanniksi design science, koska se soveltuu tämän tyyppiseen käytännön tutkimus- ja kehittämistyöhön. Suunnittelutieteellisessä tutkimuksessa viitataan artefaktiin, joka voi olla konstruktio, malli, menetelmä tai toteutus. IT-järjestelmäkehityksessä se on usein myös ohjelmisto. Menetelmän tavoitteena oli suunnittelutieteellisen tutkimuksen periaatteiden mukaisesti löytää sopivin ratkaisu käsillä olevaan ongelmaan kehittämällä artefakti eli ohjelmisto. Alla on esitelty suunnittelutieteellisen tutkimuksen kuusi osa-aluetta ja tätä on selkeytetty kuvassa 1. (Peffers ym. 2006.)

1. *Ongelman tunnistaminen ja motivaatio.* Määrittele tutkimusongelma ja perustele ratkaisun arvo, jotta se motivoi etsimään ratkaisua ja hyväksymään tulokset. Näin on helpompi ymmärtää tulokset paremmin ja tunnistaa ongelma.
2. *Ratkaisun tavoitteet.* Päättele ratkaisun tavoitteet ongelmankuvauksesta. Tavoitteet voivat olla parempia kuin nykyiset tai tuovan ratkaisuja ongelmiin, joita ei ole vielä havaittu.
3. *Suunnittelu ja kehitys.* Luo artefaktinen ratkaisu. Se tarkoittaa halutun toiminnallisuuden ja arkkitehtuuriin suunnittelua ja kehitystä, joka tuottaa itse määritetyn artefaktin.
4. *Havainnollistaminen.* Havainnollista artefaktin vaikutus ongelman ratkaisuun. Se voi sisältää artefaktin käytön esimerkin, testaamisen, simuloinnin, esimerkkitapauksen tai todisteen avulla.
5. *Arviointi.* Tarkkaile ja arvioi, miten hyvin artefakti tukee ongelman ratkaisuja. Vertaile määriteltyjen tavoitteiden ja lopputuloksen tuloksia.

6. *Viestintä.* Keskustele menetelmän vaiheista ja siihen liittyvistä ongelmista sekä artefaktin hyödyllisyydestä ja sen tuomista uusista mahdollisuuksista. Käy läpi myös suunnittelun tarkkuus ja tehokkuus.



Kuva 1. Suunnittelutieteellisen tutkimusprosessin prosessimalli (Peffers ym. 2006, 93)

Suunnittelutieteellinen tutkimus soveltuu erinomaisesti rakenteeksi kehittämishankkeeseen. Ensimmäisessä vaiheessa tunnistettiin haaste tai ongelma, miten välittää tieto lähdejärjestelmästä luotettavasti kohdejärjestelmään ja siitä vastaanottajan tietoon. Tiedon välittämiseen on käytetty lähes aina integraatioalustaa, mikäli asiakkaalla on ollut sellainen käytettävissä, joten alusta alkaen oli selvää, että integraatioalustan kautta integroidaan järjestelmät toimimaan yhtenäisesti. Motivointi tai perustelut eivät siis olleet tarpeellisia.

Toisessa vaiheessa määritettiin tavoitteet, toimintalogiikka ja jatkokehityskohteet järjestelmälle. Tavoitteena oli saada järjestelmä ensin toimimaan ja myöhemmin kehittää sitä lisää, jotta saataisiin useampi toteutusvaihe. Kolmannessa vaiheessa suunniteltiin ratkaisu luomalla arkkitehtuurikuvaus ja prosessikaavio, jonka jälkeen aloitettiin toteutus. Toteutus jakaantui kahteen toteutusvaiheeseen; integroinnit lähdejärjestelmärajapinnasta integraatioalustaan ja integraatioalustan rajapinnalta SMS-palveluun. Neljännessä vaiheessa havainnollistettiin integraation toimivuutta testiympäristössä ja tämän jälkeen tuotantoympäristössä.

Viidennessä vaiheessa todennettiin, että integraatiototeutus on halutunlainen ja SMS-sanomat menevät perille. Kuudennessa vaiheessa on käyty keskusteluja vanhojen leikkaussalien integroimisesta siivousjärjestelmään sekä reaaliaikaisen tilannekuvanäytön toteutuksesta. Kehittämishankkeen artefaktin ensimmäinen osa oli testiympäristöön tehty toteutus, jonka jälkeen ensimmäinen versio siirrettiin tuotantoympäristöön testauskäyttöön. Tuotannon testaus toteutettiin osittain loppukäyttäjien tutustuessa uuden näyttöpääteen toimintoihin. Saatujen havaintojen ja käyttökokemuksien perusteella kehitystyötä jatkettiin testiympäristössä, kunnes lopullinen toteutusversio vietiin testiympäristöstä tuotantokäyttöön.

3 Integraatiot sosiaali- ja terveydenhuollon näkökulmasta

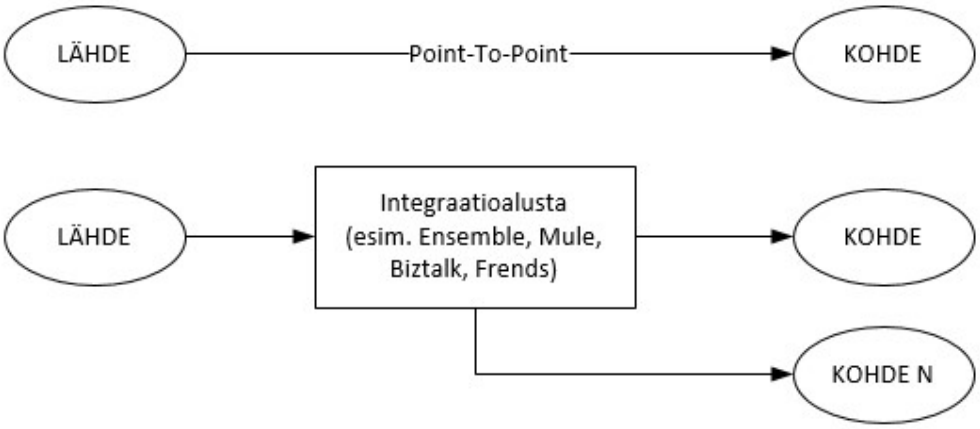
Suomessa sosiaali- ja terveydenhuollon tietojärjestelmien perustana toimii asiakas- ja potilastietojärjestelmä (APTJ), joka tarjoaa perusohjelmistot käyttäjilleen. APTJ-järjestelmiä tarjoaa useampi toimija (esimerkiksi TietoEvry, CGI, Esko Systems, Epic, Abilita) ja nämä järjestelmät ovat pääsääntöisesti ikääntyneitä järjestelmiä (esimerkiksi Lifecare/Effica, Uranus/Omni, Esko, Uranus, Pegasos/Omni, Abilita ja Apotti). (Kuntaliitto 2020.) Tästä syystä Suomessa on paljon pieniä ja keskisuuria terveysteknologiayrityksiä, jotka tarjoavat nykyaikaisia, laadukkaita ja elintärkeitä ohjelmistotuotteitaan hyvinvointialueille (HVA). Nämä ohjelmistotuotteet integroidaan osaksi olemassa olevia APTJ-järjestelmiä. Tyypillisesti näitä integraatioita voi olla useita satoja, mikä tarkoittaa tarvetta rakentaa suuri määrä suojattuja verkkoyhteyksiä lähde- ja kohdejärjestelmien välille. Suomessa yleisin sairaanhoitopiirien integraatioalusta on InterSystemsin tuote, joista Ensemble on ollut markkinoilla vuodesta 2003, mutta uudempia tuotteita ovat HealthShare ja Iris for Health. (InterSystems 2022.)

Integraatioalustapalvelin toimii edusta- tai välityspalvelimena APTJ-järjestelmille eli APTJ-järjestelmistä vältetään suoria yhteyksiä julkiseen internettiin. Kun dataliikenne kierrätetään integraatioalustan kautta, se tarjoaa ylimääräisen kerroksen tietoturvaa. Yleisesti arkkitehtuurisesti on todettu hyväksi, että julkisen internetin ja APTJ-järjestelmän välissä on edustapalvelin, ettei APTJ-järjestelmä ole suorassa yhteydessä julkiseen verkkoon. Käytännössä tämä tarkoittaa sitä, että hakkeri voi onnistua murtautumaan palomuurin läpi, mutta matka pysähtyy edustapalvelimeen, eikä hakkeri pääse käsiksi APTJ-järjestelmän tietoihin. Yksinkertaisimmillaan integraatio voi olla pelkkää läpijuoksutusta eli integraatioalusta ei koske sanomien sisältöön millään tavalla. (Traficom 2021.)

InterSystemsin integraatioalustan tyypilliset integraatiot ovat HL7-sanomaliikennettä ja tiedonsiirtoliikennettä (HTTP, HTTPS, FTP, SFTP, Web Service, API tai levyjako). Yleisiä käytettyjä tiedostoformaatteja ovat TXT, XML, JSON ja CSV. Haasteellisten integraatioiden prosesseissa parsitaan, konvertoidaan, tuotetaan tietoa ObjectScript-koodikielen avulla. Integraatioalusta tarjoaa järjestelmille valvonnan, joka on erityisen kriittinen palvelu sairaalamaailmassa. Integraatiovalvonnan kannalta HL7-sanomaliikenne, laboratorio, kuvantaminen, DVV:n aineistot ja Kelan Kanta-palvelut ovat tyypillisesti sellaisia palveluita, joiden toimivuus on sairaalan toiminnan kannalta erittäin tärkeää. Mikäli jokin laite, yhteys, tiedosto tai sanoma ei saavuta kohdettaan, häiriöstä generoituu automaattisesti toiminnanohjausjärjestelmään heräteilmoitus. Tämä mahdollistaa sen, että asia voidaan korjata ennen kuin siitä ehtii muodostua isompi häiriö. Sanomaliikenteen monitoroinnilla ja häiriöiden pikaisilla korjauksilla pyritään estämään potilasturvallisuuden vaarantuminen. (2M-IT 2020.)

3.1 Yleisimmät integraatiotyypit sosiaali- ja terveydenhuollossa

Yleisesti sosiaali- ja terveydenhuollon tietojärjestelmäintegraatiot ovat tyypiltään point-to-point tai integraatioalustan välityksellä toimivia. Point-to-point tarkoittaa kahden järjestelmän välistä suoraa yhteyttä (kuva 2). Integraatioalustan kautta saadaan valvonta ja näkyvyys integraatioihin, mitä yleensä point-to-point integraatioissa ei ole. Lisäksi integraatioalustalla voidaan välittää tietoa yhdestä lähteestä useaan kohteeseen, mikä ei ole mahdollista point-to-point integraatioissa. Valvonnan, ylläpidon ja hallittavuuden kannalta integraatioalusta on parempi ratkaisu, mutta mikäli integraatioalustaa ei olisi, point-to-point olisi kustannustehokas vaihtoehto. Point-to-point yhteyksiä rakennetaan yleensä silloin, mikäli saman ohjelmistotoimittajan sovellukset ovat yhteydessä toisiinsa. (Flashnode Oy 2022.)



Kuva 2. Yleisimmät integraatiomallit (Flashnode Oy 2022)

3.2 Integraation esiselvitys ja alkutilanne

Kehittämishankkeessa asiakasorganisaatio oli valinnut leikkaussalien näyttöpäätetoimittajan, jonka tuotteesta oli tarjolla rajapinta siivouskutsujärjestelmäintegraatiolla. Asiakkaalla ei ollut valmista rajapintaa, miten käsitellä näyttöpäätteeltä saatavaa REST/API-sanomaa mutta asiakkaalla oli jo valmiina integraatioalusta Ensemble, joten REST/API-sanomien käsittely oli järkevintä toteuttaa käyttäen olemassa olevaa ympäristöä. Tässä vaiheessa asian ratkaisemiseksi mukaan liittyi 2M-IT, jonka työnkuvaksi muodostui integraatioalustan ylläpito-, valvonta- ja kehityspalveluiden tarjoaminen asiakkaalleen. Yrityksenä 2M-IT on kasvanut Suomen suurimmaksi sosiaali- ja terveydenhuollon tietoteknisiä palveluja tuottavaksi julkisomisteiseksi yhtiöksi, joten 2M-IT oli oikea yritys tehtävään (2M-IT 2022).

Integraation toteuttamisen suunnittelu aloitettiin esiselvittämällä integraatiotarpeet, joita olivat lähde- ja kohdejärjestelmien osalta siirrettävän aineiston sisältö, siirtoaikataulut, kansiorakenteet, arkistointiasiat, säilytysajat, tietoliikennekuviot, mahdolliset salausavaimet, sertifikaatit ja käyttäjätunnukset. Mikäli integraatio vaati siirrettävän aineiston muokkausta, sitä pystyttiin tekemään integraatioalustan kautta kulkeville integraatioille. Yleensä tietoliikenneyhteyksien muodostamiseen tarvitaan palomuuriavauksia ja tällaisiin toimenpiteisiin sosiaali- ja terveydenhuoltoympäristöissä vaaditaan arkkitehtuurikuvaus, riskikartoitus ja vaikutusten arviointi. Kun edellä mainitut asiat ja työmääräarvio oli hyväksytty ja tilattu, voitiin työt aloittaa.

4 Toiminnanohjausjärjestelmät

Uuteen sairaalarakennukseen rakennettiin 12 leikkaussalia, jotka ovat kalustettu ja sisustettu lähes samalla tavalla. Leikkaussalissa suoritettavat kirurgiset toimipiteet eli leikkaukset määrittävät lopullisen kalustuksen, toisin sanoen mitä sairaalalaitteita eri saleihin on asennettu. Leikkaussalien salinohjausjärjestelmä ja sen toimintalogiikka on samalainen kaikissa uusissa leikkaussaleissa.

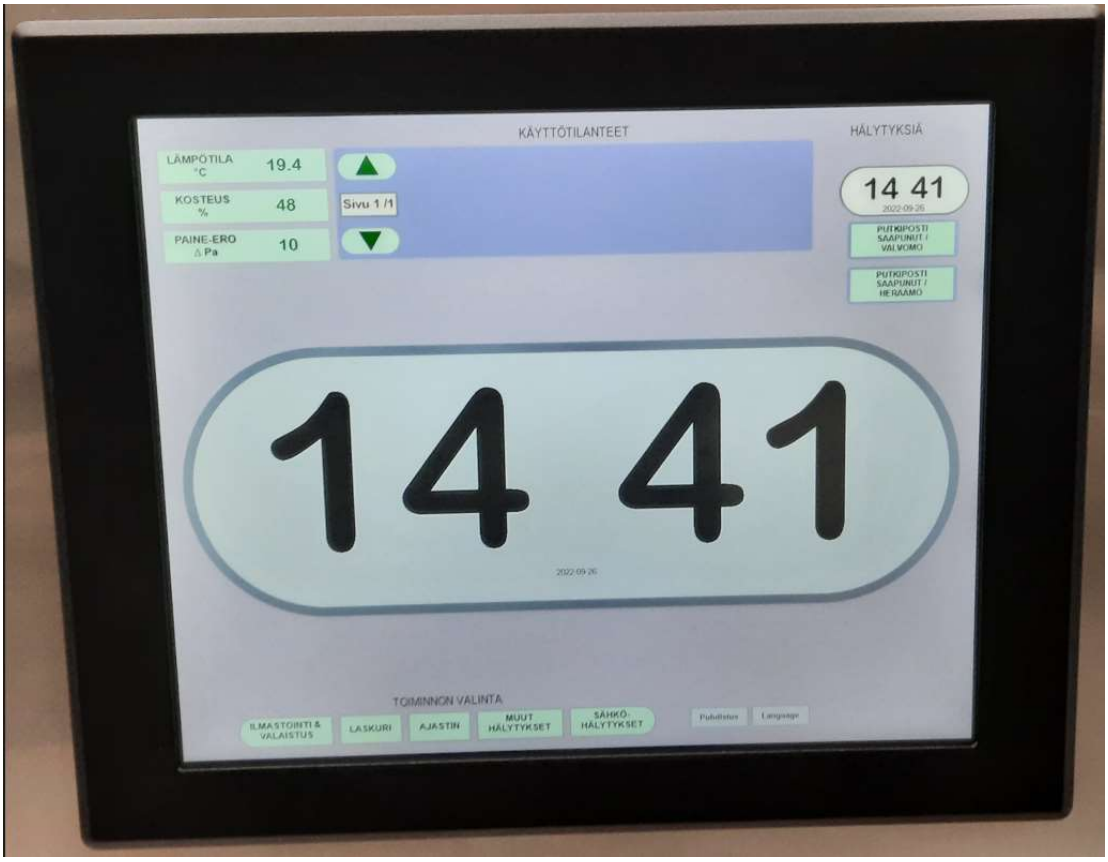
4.1 Leikkaussalit

Uudessa sairaalassa on erityyppisiä leikkaussaleja, ja suurin osa leikkaussaleista palvelee ortopedian tarpeita. Kaksi ortopedian salia ja kolmas sali ovat yhteisiä Thoraxin (rintakehän) kanssa, joka viittaa siihen, että luu- ja tukielinsairauksiin erikoistuneella kirurgian ja lääketieteen alalla on tarvetta tämän tyyppiselle leikkaussalille. Kaksi salia tuli gastroenterologian eli vatsaelimiin liittyvien leikkausten tarpeisiin (kuva 3). Urologian leikkaussaleja tuli myös kaksi, jossa lisäksi toisessa tehdään ERCP-operaatiota (sappi- ja haimatiehyiden tähystys). Traumatologian leikkaussalissa korjataan tapaturmien seurauksena syntyneitä vammoja. Sektio-salissa tehdään keisarinleikkauksia. Hybridisali tarkoittaa sitä, että salissa pystytään yhdistämään korkealaatuinen röntgenkuvantaminen mihin tahansa aivokirurgiaan ja anestesiaan. (GE Healthcare 2021.) Lisäksi saleja tuli päivystyksen ja gynekologian tarpeisiin.

Kuva 3. Gastronomian leikkaussali

4.2 Salinohjausjärjestelmä

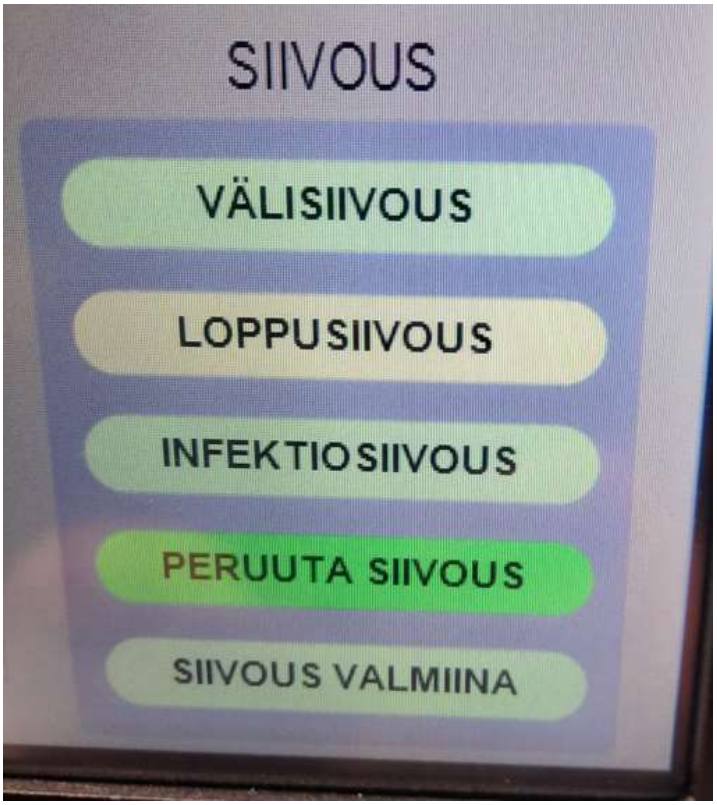
Uuden rakennuksen leikkaussalit sisältävät kosketusnäyttöpääteeltä hallittavan salinohjausjärjestelmän, jolla ohjataan muun muassa valaistusta ja ilmastointia (kuva 4). Lisäksi ohjauspaneeli tarjoaa reaaliaikaista tilannetietoa valvomosta, heräämöstä, lämpötilasta, kosteudesta, paine-erosta, kellonajasta ja hälytyksistä. Näiden ominaisuuksien lisäksi salinohjausjärjestelmä tarjoaa myös muita ominaisuuksia, kuten siivouskutsujärjestelmän.



Kuva 4. Leikkaussalinohjausjärjestelmän päänäyttö

4.2.1 Siivouskutsusovellus

Leikkaussalien siisteydestä huolehtiminen on perusedellytys asiakkaiden pikaiselle paranemiselle ja tästä syystä infektioiden torjunta leikkausosastolla on tärkeää. Siivouskutsusovelluksella pyritään minimoimaan leikkaussaliliikennettä, pidetään ympäristö siistinä ja tehostetaan käyttövalmiutta. Siivouskutsutoimintoja painamalla saadaan laitoshuolto ja leikkaussalin henkilöstö tietoiseksi tarvittavista toimenpiteistä ja tilanteesta leikkaussalissa (kuva 5). Aiemmin siivouspyynnöt tehtiin soittamalla laitoshuoltajille, joten siivouskutsusovelluksella saadaan napin painalluksella tarvittavat asiat kerrottua. Tämä vapauttaa leikkaussalin henkilökunnan resursseja suorittamaan muita toimenpiteitä ja säästää heidän työaikaansa.



Kuva 5. Siivouskutsusovelluksen päänäyttö

Ohjauspaneelilla siivoussovelluksessa hailakan keltainen väri näyttää, mitä käyttäjä on viimeksi painanut eli mikä SMS-viesti on viimeksi lähetetty. Kirkkaan vihreä väri näyttää tilannetta, mitä ei ole tässä toteutuksessa konfiguroitu järjestelmään eli se ei ole käytössä. Vaalean vihreä on perustaustaväri toiminnoille.

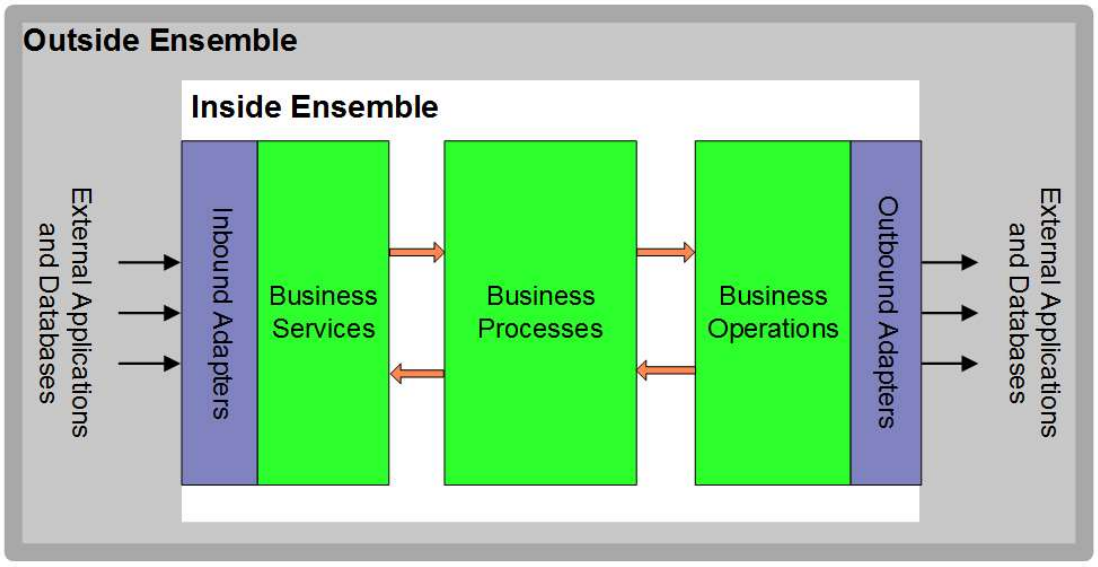
4.3 Integraatioalusta Ensemble

Ensemble-integraatioalusta helpottaa korkean suorituskyvyn mahdollistavien sovellusten rakentamista ja yhdistää datan ja sovellukset toisiinsa. Lisäksi Ensemble on stabiilialusta, ja tästä syystä varsinkin pankki- ja terveydenhuoltoalalla alusta on saavuttanut suuren suosion. Ensemble mahdollistaa erilaisten sanomatyyppien muokkaamisen, korjaamisen ja uudelleen lähettämisen kesken sanomanvälitysprosessin. Ensemble-sovellus sisältää valmiina erilaisille datoille sopivia vastaanotto- ja lähetysluokkia, joiden avulla erilaiset järjestelmät on mahdollista integroida nopeasti käyttöönotettavaksi. Ensemble sisältää Caché -tietokannan, mihin sanomahistoria tallennetaan halutun säilytysajan mukaisesti. (InterSystems 2018.)

4.3.1 Ensemblen pääkäsitteet

Ensemblen toiminta perustuu tuotantomäärityksiin (Production Configuration), joilla kommunikoidaan ulkoisten järjestelmien kanssa sekä elementteihin, jotka suorittavat tuotannon sisäistä käsittelyä. Tuotannon elementit tunnetaan *Business Hosts* -nimellä, joita on kolmenlaisia. Nämä palvelevat eri käyttötarkoituksia (kuva 6):

- *Business Services* hyväksyvät pyynnöt tuotannon ulkopuolelta ja välittävät ne Ensemblen isäntäluokille käsittelyä ja suorittamista varten.
- *Business Processes* hyväksyvät pyynnöt tuotannon isäntäluokilta *Business Services*:ltä tai *Business Processes*:lta. Lisäksi prosessi käsittelee pyynnöt tai välittää ne eteenpäin muille Ensemblen isäntäluokille käsiteltäväksi.
- *Business Operations* hyväksyy pyynnöt Ensemblen sisällä olevilta isäntäluokilta – *Business Services*:ltä tai *Business Processes*:lta. Lisäksi operaatio käsittelee pyynnöt tai välittää ne Ensemblen ulkopuolisille tahoille käsittelyä varten.

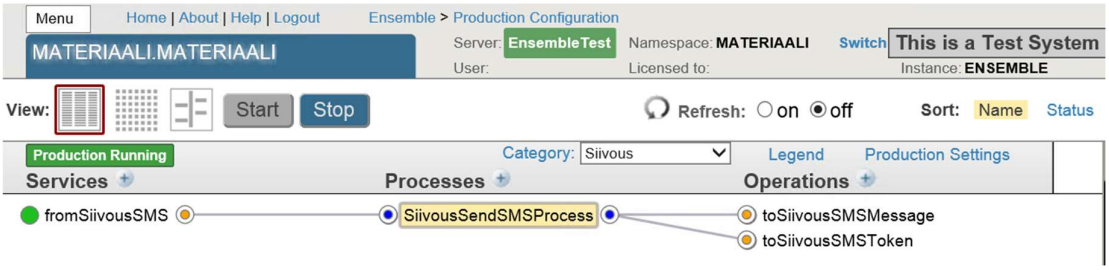


Kuva 6. Ensemblen käsitteellinen yleiskuva (Intersystems 2022a.)

Business Hosts:t kommunikoivat keskenään Ensemble-viestien kautta. Kaikki Ensemble-viestit tallennetaan Ensemble-sanomavarastoon ja ne ovat nähtävissä hallintaportaalin kautta.

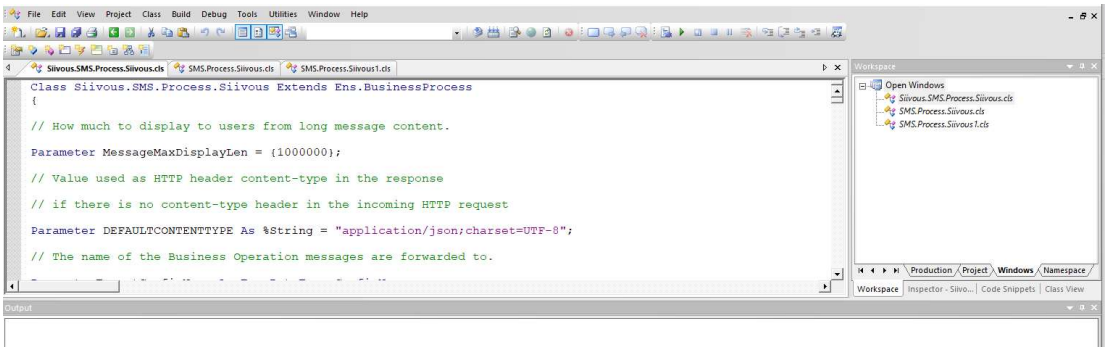
4.3.2 Ensemblen sovellukset

Ensemblen hallintaportaali (Management Portal) tarjoaa graafisen käyttöliittymän integraatioiden hallinnoinnille, kehittämiselle ja ylläpitämiselle. Hallintaportaali on tärkein Ensemblen työkaluista, koska sen avulla käyttäjä pystyy näkemään integraatioiden kokonaistilanteen hyvin nopeasti. Portaali on erittäin kätevä ja nopea työkalu konfiguraatiomuutoksien toteuttamiseen (kuva 7). Ylläpidon kannalta hallintaportaali on tärkein työkalu, koska sieltä näkee tilannekuvan häiriötilanteissa parhaiten. (InterSystems 2022b.)



Kuva 7. Esimerkki Ensemblen hallintaportaalin näkymästä testiympäristöstä

Ensemble-ohjelmiston integraatioiden kehitystyökalu Studio (kuva 8), tarjoaa ohjelmointikehitys- ja käännösympäristön integraatioiden toteuttamiselle integraatioalustalla. Studio mahdollistaa muun muassa integraatioiden nopean muokkaamisen, luokat (tietokantaluokat, verkkopalveluluokat), rutiinit ObjectScriptillä, syntaksin värityksen ja syntaksin tarkistuksen ObjectScriptille, Javalle, SQL:lle, JavaScriptille, HTML:lle ja XML:lle. Lisäksi Studiossa on tuki kehittäjäryhmille, jotka työskentelevät sovelluksen lähdekoodin yhteisen arkiston kanssa sekä graafinen lähdekoodin debuggeri ja ominaisuudet sovellusten lähdekoodien projektointiin. (InterSystems 2022c.)



Kuva 8. Esimerkki Ensemblen kehitystyökalu Studion näkymästä testiympäristössä

Terminal-pääte on yksinkertainen komentorivikäyttöliittymä ObjectScript-komentojen syöttämiseen ja nykyisten arvojen näyttämiseen. Se on hyödyllinen syntaksin ja toiminnallisuuksien oppimisen kannalta, koska sen avulla voi suorittaa ObjectScript-komentoja (kuva 9). Se helpottaa ohjelmointia, kehitystyötä ja vian selvitystä, kun pystyy todentamaan koodin osien toimivuutta irrallisena osana kokonaisuudesta. Toisin sanoen terminaalissa pystyy etukäteen testaamaan toimintoja, jotka käyttäjä esimerkiksi haluaa kirjoittaa koodiin ja varmistua lopputuloksesta. (InterSystems 2022d.)

Kuva 9. Esimerkki Ensemblen Terminal-kehitystyökalun näkymästä testiympäristössä

4.4 Tekstiviestit

SMS-tekstiviestit (Short Message Service) ovat viestejä, joita lähetetään matkapuhelinverkossa laitteiden välillä. Tekstiviesti keksintönä täytti 3.12.2022 30 vuotta, kun brittiläinen Neil Papworth lähetti ensimmäisen ”Merry Christmas” -viestin tietokoneesta mobiililaitteeseen. Vuonna 1984 Matti Makkonen esitteli idean kannettavien puhelinten viestipalvelusta, joten Suomessa keksijänä on pidetty häntä. (Linnake 2022.) Tekstiviestit kulkevat matkapuhelinverkossa 160 merkin pituisina lähetyksinä. Mikäli tekstiviesti on pidempi, jokainen alkava 160 merkin osa on uusi tekstiviesti. Viesti kuitenkin toimitetaan yhtenäisenä vastaanottajalle. Viestit välitetään tekstiviestikeskuksen kautta, joka säilyttää viestin tallessa, kunnes se on saatu toimitettua perille. Kehittämishankkeessa tekstiviestin muodostaminen ja lähettäminen viestinvälityspalveluun oli avainasemassa.

4.4.1 Viestinvälityspalvelu

Teleoperaattorit tarjoavat SMS-viestinvälityspalveluita yrityksille asiakkaiden, henkilöstön ja sidosryhmien kanssa kommunikointiin. SMS-viestivälityspalvelun kautta voi esimerkiksi lähettää kohdennettuja viestejä, viestiä henkilöstölle tärkeistä tapahtumista tai tiedottaa

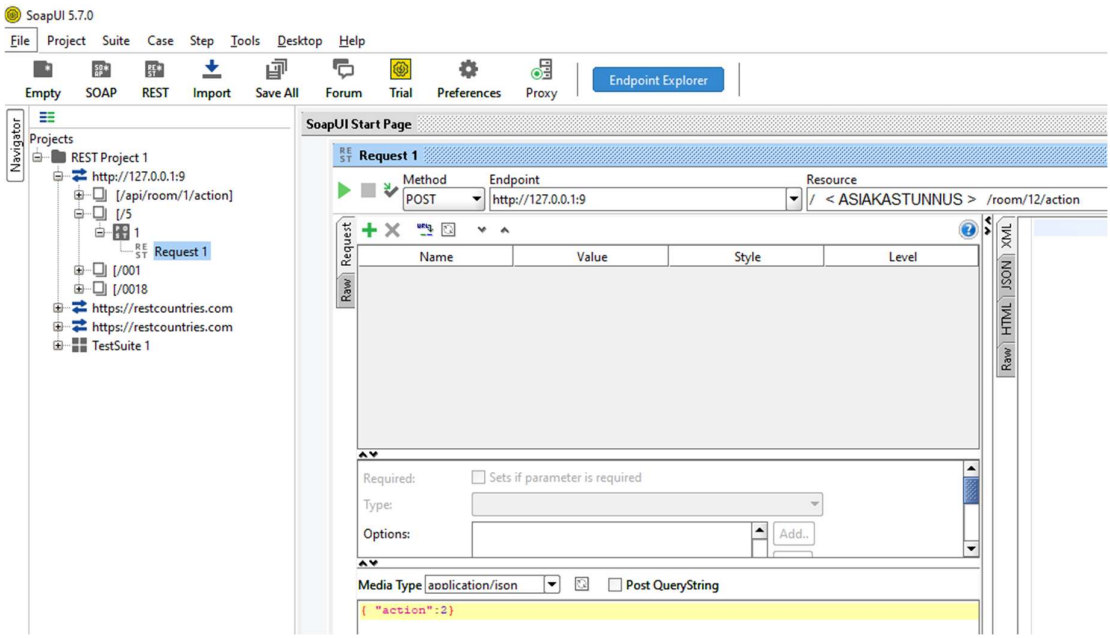
erilaisista häiriö- ja vikatilanteista. SMS-viestiliikenne on yleensä kaksisuuntaista, mutta tässä toteutuksessa viestiliikenne on yhdensuuntaista. Yksisuuntaisessa palvelussa voidaan lähettää viesti asiakkaan tietojärjestelmistä sovitun rajapinnan kautta loppuasiakkaan matkapuhelimeen. SMS-viestinvälityspalvelu mahdollistaa tekstiviestien lähettämisen tietojärjestelmästä matkapuhelinverkkoon, mutta tekstiviestien vastaanottamiselle matkapuhelinverkosta tietojärjestelmään ei ole tarvetta tässä toteutuksessa. (DNA 2022.)

4.5 Muut sovellukset

Ohjelmiston ja integraation kehittämisen aikana ei useinkaan ole saatavilla laitteistoja, lähdejärjestelmiä tai kohdejärjestelmiä, jolloin joudutaan käyttämään erilaisia tarkoituksia varten tehtyjä aputyökaluohjelmia. Kehittämishankkeen aikana käytettiin SoapUI:ta ja Wireshark:ia testaamisessa ja ongelmien selvitystyön apuna. Aputyökaluohjelmien lisäksi tietosisällön verifiointiin ja validointiin käytettiin julkisia, internetistä löytyviä sivustoja, esimerkiksi XML:n ja JSON:n formaateille.

4.5.1 SoapUI

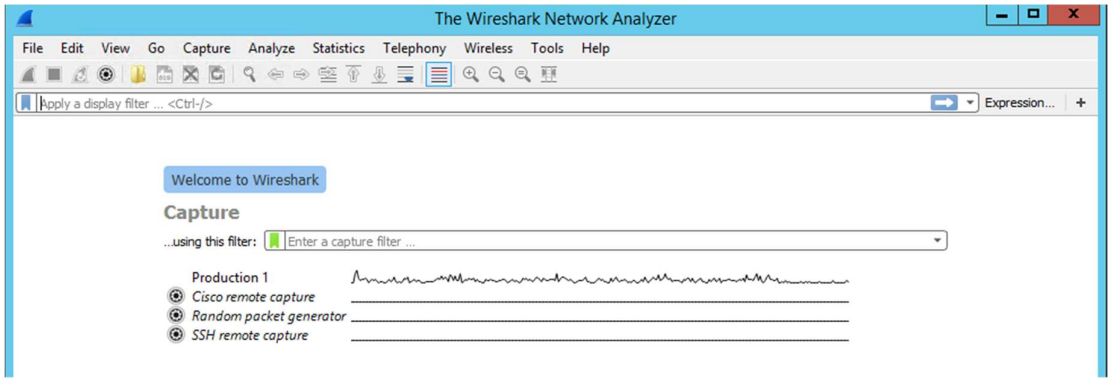
SoapUI-testaustyökalua (Simple Object Access Protocol User Interface) käytettiin leikkaussalin ohjauspaneelin siivouskutsujen simulointiin testiympäristöissä (kuva 10) (SoapUI 2022). Käytännössä API-kutsu rakennettiin SoapUI:hin, joka välitettiin integraatioalustan testiympäristöön. Lisäksi kehitysvaiheessa simuloitiin SoapUI:lla REST-kutsuja tokenin noutamiseen ja SMS-viestien lähettämiseen, millä pystyttiin varmistamaan kaikkien tarvittavien parametrien mukana olo. Näin pystyttiin etukäteen kehittämään ja testaamaan integraation toimivuutta. Testausta siivouskutsujärjestelmälle paikan päällä sairaalan leikkaussaleissa tapahtui kerran, joten oikeasta ohjauspaneelista tulevaa kutsua ei testattu kuin muutama viikko ennen tuotannon käynnistymistä. Tästä johtuen SoapUI oli kehittämistyössä tärkeä työkalu toteutuksen kannalta.



Kuva 10. Esimerkki SoapUI-testaustyökalusta

4.5.2 Wireshark

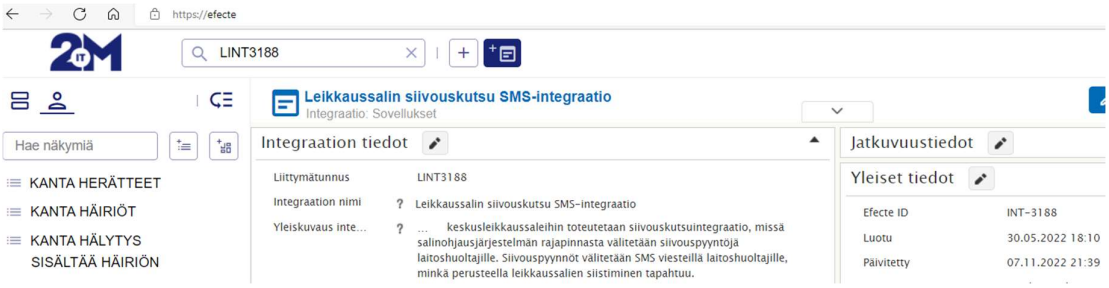
Wireshark-pakettianalysaattorilla analysoitiin tietoliikennettä ja varmistettiin integraatioalustalta lähtevien sanomien sisällön oikeellisuus sekä saatiin näkyviin alkuvaiheen API-kutsujen virheelliset parametrimääritykset (Wireshark 2022). Wiresharkin huono puoli on se, että mikäli sen unohtaa sammuttaa tiedon keräämisen jälkeen, ohjelma täyttää käytettävissä olevan levytilan (kuva 11). Hyvää Wireshark:ssa on se, että se on ilmainen ja avoimen lähdekoodin sovellus.



Kuva 11. Esimerkki Wireshark-testaustyökalusta

4.5.3 Efecte

2M-IT:llä integraatioalustalla tapahtuvat hälytykset ohjataan Efecte-toiminnanohjausjärjestelmään (kuva 12) (Efecte 2022). Siivouskutsujärjestelmä integraatioon on konfiguroitu hälytyksiä, joista generoituu Efecteen ilmoituksia häiriötilanteista. Lisäksi Efecteen on luotu rakennetusta järjestelmästä tietokortteja, jotka sisältävät toimintaohjeita eri tilanteisiin. Efecte ratkaisutietokanta- ja tiketöintijärjestelmä-tyyppiset palvelut ovat laajasti käytössä suurien ja keskisuurien yritysten ja julkishallinnon palveluorganisaatioissa.



Kuva 12. Esimerkki Efecten integraatiokortista

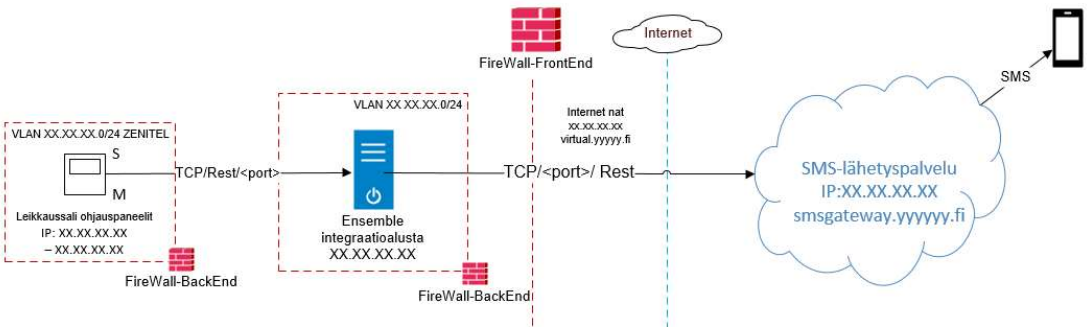
5 Protokollat ja arkkitehtuurit

5.1 Tuotantoympäristö ja arkkitehtuuri

Uuteen sairaalaan rakennettiin virtuaalilähiverkko (VLAN), johon leikkaussalien ohjauspaneelit (12 kappaletta) konfiguroitiin. Jokaiselle ohjauspaneelille oli määritelty oma IP-osoite, jolla mahdollistettiin sanomaliikenne vanhan sairaalaverkon puolelle, integraatioalusta Ensemblelle. Ohjauspaneelin ja Ensemblen välinen sanomaliikenne tapahtui sisäverkossa, joten palomuurisääntöihin oli tarve lisätä uuden verkon IP-osoite- ja porttitiedot.

Ensembleltä siivouskutsuliikenne välitettiin julkisen operaattorin tarjoamaan SMS-palveluun, mistä muodostettu SMS-viesti välitettiin Ensemblellä määritettyihin puhelinnumeroihin. Ensemblen IP-osoite ei näy verkkoon, koska osoitteenmuunnos (NAT) piilottaa sen. SMS-palvelu ja muu sairaalan verkosta julkiseen internettiin suuntautuva liikenne näkyy ulospäin vain yhdestä osoitteesta tulevana. Tässä toteutuksessa sairaalla oli jo operaattorin tarjoama yritysverkko, joten integraatio ei vaatinut erillistä IPSEC LAN-to-LAN-tunnelin rakentamista.

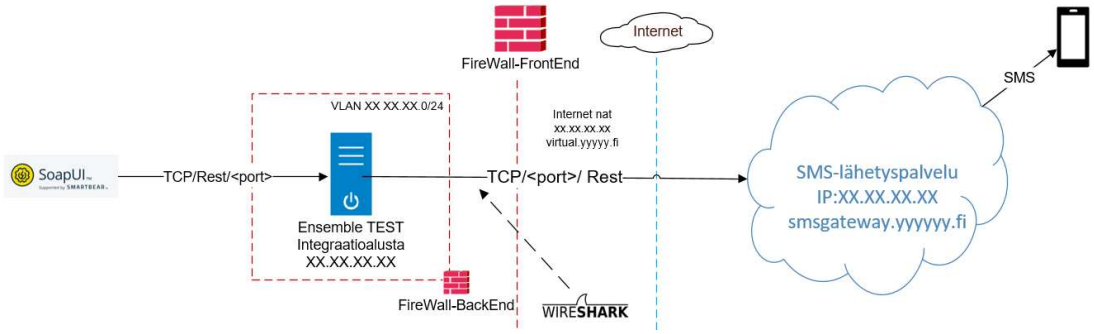
Uudessa sairaalaverkossa olivat laitteet, joilla liikennöitiin integraatioalustalle Ensemblellä. Ensemble-palvelimet sijaitsivat vanhan sairaalaverkon puolella, joten tarvittavat F5-palomuuriavaukset oli tehtävä halutulle IP:lle ja portille, jotta HTTP REST-liikenne Ensemblelle alkoi toimia. SMS-viestinvälityspalveluun oli olemassa tunneli, jonka kautta oli pääsy palveluun. Toteutettu verkkoarkkitehtuuri oli kuvan 13 mukainen.



Kuva 13. Esimerkki siivouskutsujärjestelmän tuotannon verkkoarkkitehtuurista

5.2 Testiympäristö ja arkkitehtuuri

SMS-siivouskutsujärjestelmän kehittäminen aloitettiin määrittelyistä, ennen kuin lähde- ja kohdejärjestelmät olivat saatavilla. Kehitystyö oli aloitettava simuloimalla näyttöpäätekutsuja käyttäen SoapUI-työkalua, millä ensin rakennettiin mahdollisimman oikeanlainen HTTP REST -kutsu. Testi Ensemblellä luettiin sisään tämän kutsu, mistä otettiin talteen halutut parametrit. Tämän jälkeen HTTP REST-kutsuilla haettiin SMS-viestiliikenteen mahdollistaneet parametrit ja lopuksi SMS-viestit lähettiin operaattorin tarjoamaan palveluun. Wireshark:lla monitorointiin TCP-liikennettä, minkä perusteella pystyttiin muokkaamaan REST-sanomien sisältöä oikeanlaiseksi. SoapUI- ja Wireshark-työkalut ja Testi Ensemble ovat edelleen olennainen osa testausta, kehitystä ja simulointia, kun siivouskutsujärjestelmää jatkokehitetään (kuva 14).



Kuva 14. Esimerkki siivouskutsujärjestelmän testiympäristön verkkoarkkitehtuurista

5.3 API-ohjelmointirajapinta

API (Application Programming Interface) tarkoittaa ohjelmointirajapintatekniikkaa, jolla luodaan yhteyksiä verkkopalvelujen, laitteiden, tietokonejärjestelmien ja sovellusten välille. API liittymien suurin hyöty tulee siitä, että rajapinnan avulla voi helposti siirtää tietoja ohjelmien välillä. API-rajapinta sisältää käskyjä, joilla voi esimerkiksi hakea tietoja ja käyttää taustajärjestelmän toimintoja ilman, että ulkopuolisia tarvitsee päästää itse järjestelmään. (Helsingin kaupunki 2020.) Siivouskutsujärjestelmässä ohjauspaneelista on sisäinen REST-API-rajapinta (Private API) integraatioalustaan. Sisäisen REST-API:n on tarkoitettu pelkästään yhdenjärjestelmän tai organisaation omassa hallinnassa olevien järjestelmien käyttöön. Toinen käytetty API-REST-rajapinta integraatioalusta ja SMS-viestipalvelun välissä on tyypillään julkinen API (Public API). Molemmissa siivouskutsujärjestelmän API-rajapinnoissa

sanomat ovat HTTP-liikennettä, jotka ovat pyyntö- ja vastaussanomia JSON- ja XML-formaatissa.

5.4 REST-rajapinta

REST-rajapinnalla (Representational State Transfer) tarkoitetaan sovellusarkkitehtuurimallia, jolla tietoja saadaan tuotua ja vietyä sisään sovellukseen esimerkiksi JSON-muodossa. REST ei ole varsinainen standardi, vaan yleinen tapa tietojärjestelmien tiedonvaihtoon. Tästä syystä eri ohjelmistojen REST-toteutukset voivat hieman poiketa toisistaan. Uusien sovellusten ja tietojärjestelmien kohdalla suosittu liitostapa on REST-rajapinnan kautta. Kehittämishankkeessa käytetty SMS-palvelu on toteutettu rajapintapalveluna, jossa käytetty rajapintaprotokolla on REST-API. REST-API on sovellusohjelmointirajapinta, mikä noudattaa REST-arkkitehtuuria koskevia tyylirajoituksia ja tekee mahdolliseksi vuorovaikutuksen REST-verkkopalveluiden kanssa. (TRPlane.com 2021.)

REST-palvelu vaikuttaa samanlaiselta kuin ”perinteinen” HTTP-palvelu koska REST-rajapinnan vastaukset pyyntöihin ovat samanlaisia kuin perus HTTP-palvelinten vastaukset. Palvelimelle määriteltyä REST-rajapintaa voi kutsua HTTP-metodeilla GET, POST, PUT ja DELETE. Metodit vastaavat luonti-, luku, päivitys- ja poistotoimintoja. On olemassa myös muita metodeja, mutta ne ovat harvinaisia. (Fredrich 2022.) Siivouskutsujärjestelmässä on kolme REST-rajapintaa, kutsu integraatioalustalle ja kaksi kutsua SMS-palveluun. Käytetyt kutsut ovat POST-tyyppisiä, mutta palvelun kehittyessä voi myöhemmin tulla tarve GET-tyyppiselle rajapinnalle.

5.5 JSON-formaatti

JSON (JavaScript Object Notation) on yksinkertainen formaatti tietojen tallentamiseen ja siirtämiseen. JSON:ia käytetään usein, kun tietoja lähetetään palvelimelta verkkosivulle. JSON syntaksisääntöjä ovat tekstimuotoisuus, tietojen jaottelu nimi - arvo -pareiksi ja tietojen erottaminen pilkulla toisistaan. Koodi JSON-tietojen lukemista ja luomista varten voidaan kirjoittaa millä tahansa ohjelmointikielellä. (w3schools.com 2022a.) Siivouskutsujärjestelmässä ObjectScript-kieltä käytetään JSON-tietojen lukemiseen ja käsittelyyn.

5.6 XML-formaatti

XML (eXtensible Markup Language) on merkintäkielien standardi, joka on suunniteltu tallentamaan ja siirtämään määriteltyä rakenteellista tietoa. XML-kieltä käytetään laajasti formaattina tiedonvälitykseen järjestelmien välillä, koska se on sekä ihmisen luettavissa että koneellisesti luettavassa muodossa. (w3schools.com 2022b.) Siivouskutsujärjestelmässä integraatioalustalla Ensemblessä sanomaliikenteen syntaksi on XML-muotoista.

5.7 Valvonta ja monitorointi

Integraatioalusta Ensemble toimii valvonta- ja monitorointijärjestelmänä sairaalan kriittisimmille integraatioille. Kun Ensemble on havainnut häiriön, tästä muodostetaan automaattisesti Efecten herätetiketti. Efecten herätetikettien tietojen perusteella valvontaa pystytään kohdentamaan oikeisiin järjestelmiin ja tarvittaessa tarkistetaan lähde-, kohde- tai Ensembleltä toimivuus. Mikäli herätetiketti osoittautuu yksittäiseksi tilanteeksi ja tuotanto on palautunut normaaliksi, herätetiketti kuitataan pois eikä se aiheuta sen enempää toimenpiteitä. Mikäli herätetiketti osoittautuu jatkuvaksi häiriöksi, silloin avataan häiriötiketti. Häiriötiketti pystytään Efectessä olevan integraatiodokumentoinnin perusteella ohjaamaan oikealle taholle, mikä nopeuttaa häiriön selvitystyötä ja korjaavia toimenpiteitä. Jos havaittu häiriö osoittautuu kriittiseksi ja uhkaa tuotantoa, aloitetaan siitä MI-prosessi (Major Incident). MI-prosessi ajaa kaiken muun toiminnan ohi. Tämä tarkoittaa sitä, että häiriö on ratkaistava tai on löydyttävä jokin keino, miten tuotanto voi jatkua. MI-prosessi on niin kauan käynnissä, kunnes häiriötilanne on ratkaistu tai onnistuttu kiertämään.

Mikäli häiriö osoittautuu ongelmaksi, se yleensä tarkoittaa sitä, että ongelma vaatii muutoksen. Mikäli muutos ei ole aikakriittinen, silloin avataan ongelmatiketti ja mahdollisesti muutostiketti. Muutostiketti menee muutoshallintaprosessin kautta, jotta ongelman korjaus saadaan järjestelmään toimitusaikataulujen mukaisesti. Palveluiden laadun varmistusta varten ovat SLA:t (Service-Level Agreement), joissa on kuvattu ratkaisuajat eri asteisille häiriöille. Tavoiteratkaisuajat kertovat, kuinka nopeasti häiriöt tulisi ratkaista, ettei häiriö haittaisi kohtuuttomasti tuotantoa. Toteutettu siivouskutsujärjestelmä on kytketty osaksi integraatioiden valvontaa ja monitorointia.

6 ObjectScript -ohjelmointikieli

6.1 ObjectScriptin synty

Caché ObjectScript (COS) on nykyisin vain ObjectScript, koska InterSystems osti Caché-yrityksen ja näin ollen omistaa Caché -tietokannan ja sitä kautta ObjectScript-ohjelmointikielen. ObjectScript perustuu vanhaan MUMPS-ohjelmointikieleen (Massachusetts General Hospital Utility Multi-Programming System), joka on tarkoitettu liiketoimintalogiikan, integraatioiden ja dataprosessoinnin ohjelmistojen nopeaan ja helppoon kehittämiseen sairaalaympäristöissä. InterSystems julkaisi Caché-tietoalustan alkuperäisen version vuonna 1997, mutta myöhempien kärkituotteiden Ensemble, Caché, HealthShare ja IRIS on vakiinnuttanut johtavan asemansa terveydenhuollon tietokanta- ja integraatioalustatoimittajana. (InterSystems 2018.)

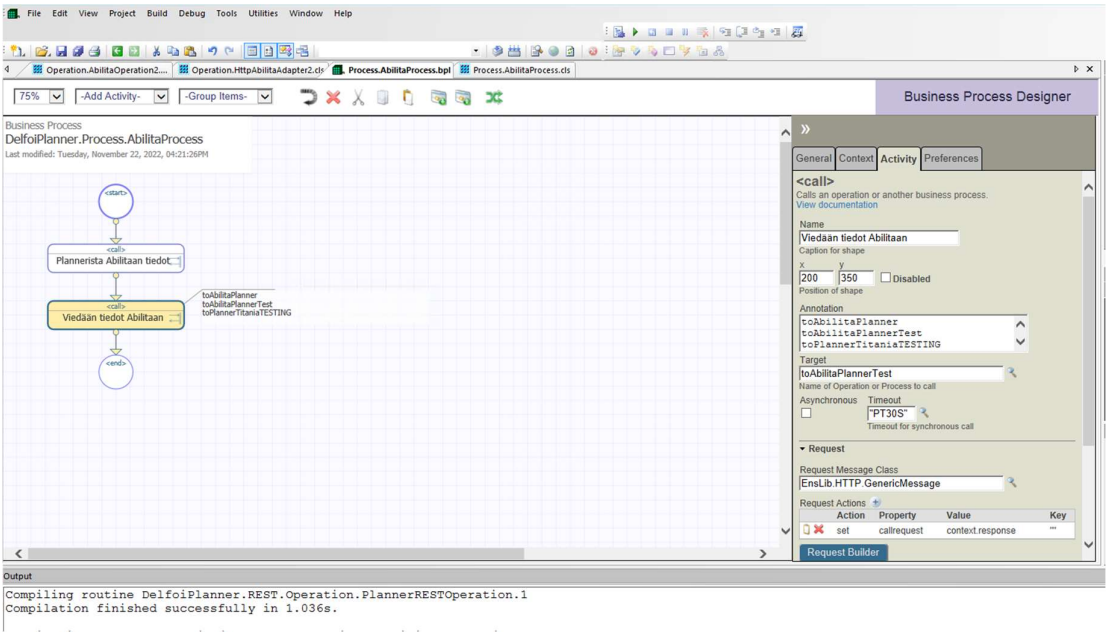
6.2 MUMPS historia

MUMPS-kieltä kutsutaan myös M-kieleksi, joka on yleiskäyttöinen ohjelmointikieli sisältäen ACID-tapahtumien tuen. Käsitteenä ACID (Atomicity, Consistency, Isolation, Durability) tarkoittaa joukkoa tietokantatapahtumien ominaisuuksista, joiden tarkoituksena on varmistaa tietojen oikeellisuus virheistä, sähkökatkoista ja vahingoista huolimatta. MUMPS sovelluksia käytettiin 1960-luvun lopulla vastaanotossa ja laboratorion raportoinnissa. MUMPSin tehokkuus perustuu siihen, että se toimii suoraan tietokannan sisällä. Kyseisen rakenteen ansiosta tiedon siirtäminen ja tallentaminen järjestelmän sisällä ohjelmistojen välillä on nopeaa. Lopulta MUMPSin käyttö levisi koko terveydenhoitoalalle ja sitä alettiin käyttää ohjelmointikielenä sähköisille sairaskertomuksille. (Campbell 2018.)

6.3 ObjectScript-ohjelmointi

ObjectScript-ohjelmointia voi tehdä InterSystemsin Management Portal- ja Studio-työkaluilla. ”Yksinkertaisiin” ohjelmointeihin soveltuu Management Portal-työkalu, jolla pystyy graafisesti rakentamaan komponenteilla ja pienellä koodauksella Business Prosesseihin ObjectScript-koodia. ObjectScriptin yhteensopivuus Management Portaalin kanssa tarkoittaa sitä, että koodia voi kehittää graafisesti tai koodinäkymästä Studio-työkalulla kehittäen.

Management Portal mahdollistaa nopeat muutokset ohjelmakoodiin, koska koodin voi kääntää, suorittaa ja testata samassa ympäristössä (kuva 15).



Kuva 15. Esimerkki Management Portaalin graafisesta kehitysympäristöstä

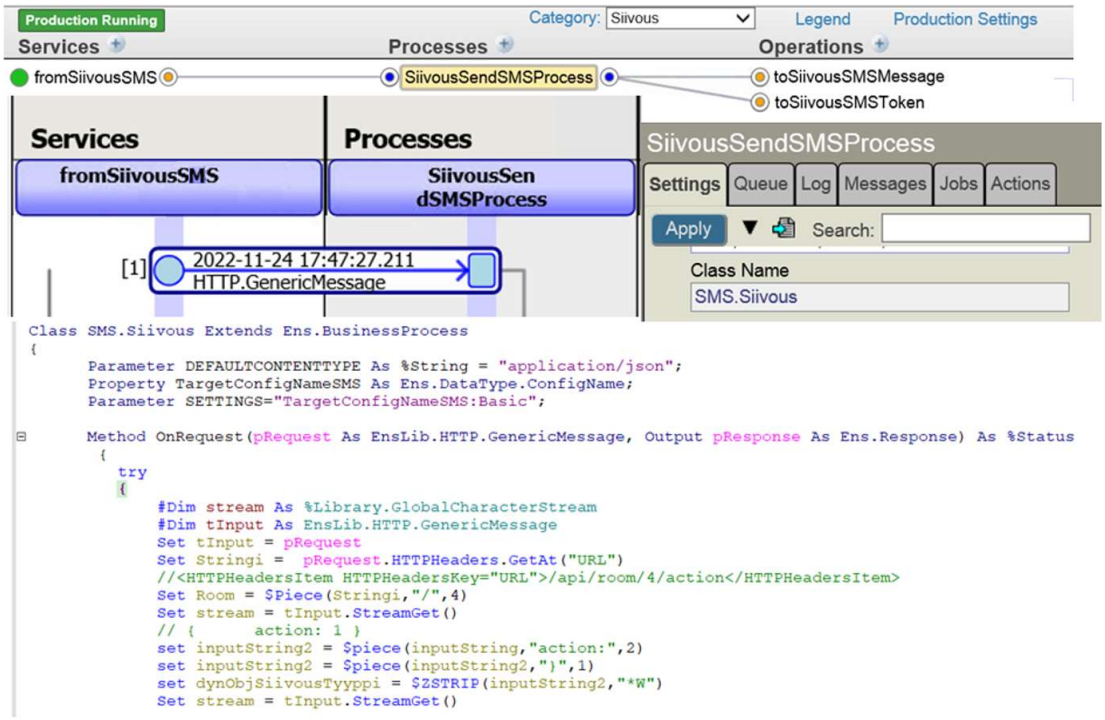
Studio-työkalulla voi rakentaa myös graafisesti ObjectScript-koodia. Ohjelmoijan on kuitenkin helpompi ymmärtää toimintaa ohjelmakoodista kuin graafisesta näkymästä, joka piilotetaan komponenttien osien alle. Studiolla koodinkirjoitusnäkymässä on helpompi etsiä ja poistaa virheitä sekä seurata ohjelmakoodin suoritusta, koska ohjelmoija näkee suoritettavan koodin jokaisen rivin. Graafisesta näkymästä virheellisesti toimivan tai väärin koodatun kohdan löytäminen on haasteellisempaa. Yleensä graafisesti on helpompi rakentaa koodia, kun vertaa koodia (kuva 16) ja graafista toteutusta.

```
///
Class DelfoiPlanner.Process.AbilitaProcess Extends Ens.BusinessProcessBPL
{

/// BPL Definition
XData BPL [ XMLNamespace = "http://www.intersystems.com/bpl" ]
{
<process language='objectscript' request='Ens.Request' response='Ens.Response' height='2000' width='2000' >
<context>
<property name='response' type='EnsLib.HTTP.GenericMessage' instantiate='0' >
</property>
</context>
<sequence xend='200' yend='450' >
<call name='Plannerista Abilitaan tiedot' target='toPlannerAbilita' async='0' timeout='"PT30S"' xpos='200' ypos='250' >
<request type='EnsLib.HTTP.GenericMessage' />
<response type='EnsLib.HTTP.GenericMessage' >
<assign property="context.response" value="callresponse" action="set" />
</response>
</call>
<call name='Viedään tiedot Abilitaan' target='toAbilitaPlannerTest' async='0' timeout='"PT30S"' xpos='200' ypos='350' >
<annotation><![CDATA[toAbilitaPlanner
toAbilitaPlannerTest
toPlannerTitaniaTESTING]]></annotation>
<request type='EnsLib.HTTP.GenericMessage' >
<assign property="callrequest" value="context.response" action="set" />
</request>
<response type='EnsLib.HTTP.GenericMessage' />
</call>
</sequence>
</process>
}
```

Kuva 16. Esimerkki Studiosta kuva 15 esitettynä ObjectScript-koodina

6.4 ObjectScript – siivouspyyntö

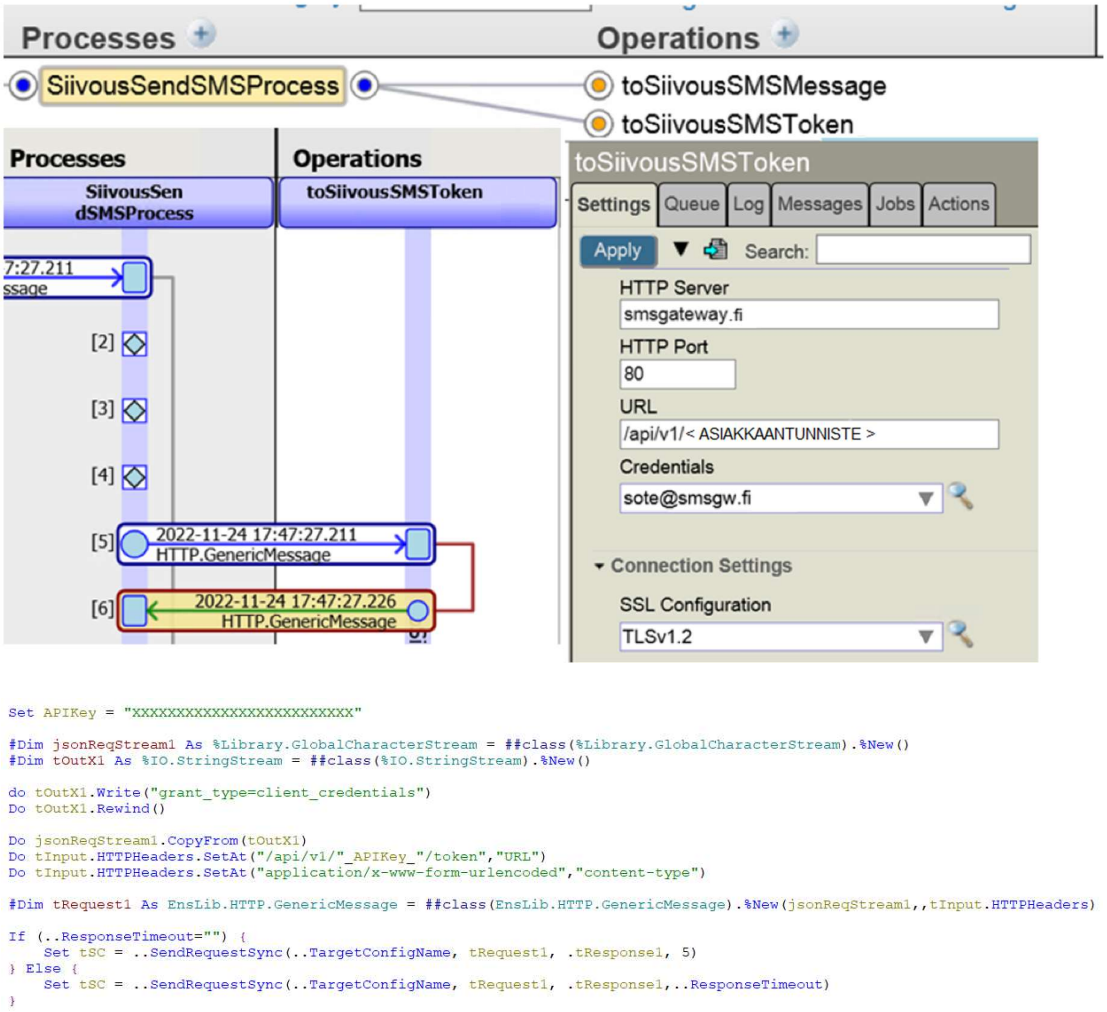
Business Service (fromSiivousSMS) ottaa vastaan siivouspyynnön ja välittää sen Business Prosessille (SiivousSendSMSProcess). Siivouskutsujärjestelmää varten on integraatioalustalle luotu Business Prosessille oma kustomoitu luokka (Siivous). Luokan rakenteen aluksi määritellään tarvittavat parametrit, ominaisuudet ja kustomoitavat parametrit. Koodiesimerkissä on esitetty toisen Business Operaation (TargetConfigNameSMS) lisääminen koodiin, mikä määritellään Management Portaalin parametrinä. Business Prosessin OnRequest-metodi ottaa vastaan leikkaussalin ohjausjärjestelmästä tulleen HTTP POST-sanoman. Lisäksi se ottaa URL:sta talteen muuttujiin siivottavan leikkaussalin tiedot sekä stream:stä JSON-tietona siivoustyypin (kuva 17). Uusimmassa HealthShare ja IRIS versioissa on parempia tapoja kuin $PIECE-funktio, mitä tässä toteutuksessa käytettiin merkkijonon talteen ottamisessa.



Kuva 17. Esimerkki ObjectScriptillä ohjauspaneelin tietojen lukeminen muuttujiin

6.5 ObjectScript – token-pyyntö

Business Prosessin OnRequest -metodiin on koodattu token-pyyntö SMS-palvelulle, jotta siivouspyyntöjärjestelmän viestejä voi välittää viestin SMS-palvelun lähetysrajapintaan. Business Operaatioon (toSiivousSMSToken) välitetään HTTP POST-kutsu Operaattorin REST-API-rajapintaan. Operaattorin toimittama palvelutunnus toimii API-avaimena, kun tokenia pyydetään SMS-palvelun tarjoajalta. HttpHeaderin URL-parametriin lisätään API-avain ja content-type, jotka olivat pakollisia tietoja yhteyden muodostuksen kannalta. Lisäksi HTTPMessagen Stream vaatii grant_type tietojen konfiguroinnin, jotta palvelu sai kirjautumistunnukset palveluun. Tarvittavat parametrit palveluun kirjautumiseen otetaan Business Operaatiolta, joka välittää HTTP POST-kutsun SMS-palvelun token REST-API rajapintaan. Rajapinnasta paluuviestinä palautuu Business Prosessille token, tokenin tyyppi ja voimassaolon kestoaika (kuva 18).




```
Set APIKey = "XXXXXXXXXXXXXXXXXXXXXXXX"

#Dim jsonReqStream1 As %Library.GlobalCharacterStream = ##class(%Library.GlobalCharacterStream).%New()
#Dim tOutX1 As %IO.StringStream = ##class(%IO.StringStream).%New()

do tOutX1.Write("grant_type=client_credentials")
Do tOutX1.Rewind()

Do jsonReqStream1.CopyFrom(tOutX1)
Do tInput.HTTPHeaders.SetAt("/api/v1/"_APIKey_"/token","URL")
Do tInput.HTTPHeaders.SetAt("application/x-www-form-urlencoded","content-type")

#Dim tRequest1 As EnsLib.HTTP.GenericMessage = ##class(EnsLib.HTTP.GenericMessage).%New(jsonReqStream1,,tInput.HTTPHeaders)

If (..ResponseTimeout="") {
    Set tSC = ..SendRequestSync(..TargetConfigName, tRequest1, .tResponse1, 5)
} Else {
    Set tSC = ..SendRequestSync(..TargetConfigName, tRequest1, .tResponse1,..ResponseTimeout)
}
```

Kuva 18. Esimerkkikoodi token-pyynnöstä viestipalveluun

6.6 ObjectScript – viestisisällön muodostus

Business Prosessin koodissa luetaan talteen saatu token, voimassaoloaika ja tokenin tyyppi. Kuvan 19 mukaisesti asetetaan viestin parametri, jota varten on rakennettu for-silmukka. Viestin rakenne on määritetyn formaatin mukainen, joten muuttuvia parametrejä toteutuksessa ovat token, puhelinnumero, leikkaussali ja siivoustyyppi. Osa parametreistä asetaan HTTPHeaderin parametreiksi, mutta itse viestin sisältö viedään HTTPMessage Stream:iin.

```
// ASETA LÄHETTÄJÄ
Set SenderText = "XXXXXSote"
        Set dll = $char(44) // comma

// NumberList toteutus
  for a = 1:1:$length(PhoneNumberList1, dll) {

    if (a > 20)
    {
        Set tSC= $$$ERROR($$$GeneralError, "SMS viestien generointi saavuttanut maksimin")
        $$$ThrowOnError(tSC)
        break
    }

    set currentnumber = $piece(PhoneNumberList1, dll, a)
    $$$TRACE("testing for "_currentnumber)

    // SMS-viestin sisällön muodostaminen
    // MALLI: <Stream>{"sender":"SOTE","destination":["+358400XXXXXX"],"text":"Hei, Sali 14 paikka 2 loppusiivous "} </Stream>
    Set Erotin=$Char(34) // yksinkertainen hipsu " = 39 , "" tupla=34
    Set Vali=":"
    Set Pilkku=","

    Do tInput.HTTPHeaders.SetAt("/api/v1/"_API_Key_"/sms","URL")
    Do tInput.HTTPHeaders.SetAt("Bearer "_..CurrentToken,"authorization")
    Do tInput.HTTPHeaders.SetAt("application/json;charset=utf-8","Content-Type")

    Set SMSBodyStr = "{"
    Set SMSBodyStr = SMSBodyStr_Erotin_"recipient"_Erotin_Vali_"{"_Erotin_"number"
    Set SMSBodyStr = SMSBodyStr_Erotin_Vali_Erotin_currentnumber_Erotin
    Set SMSBodyStr = SMSBodyStr_"}"_Pilkku_Erotin
    Set SMSBodyStr = SMSBodyStr_"data"_Erotin_Vali_"{"_Erotin_"message"_Erotin_Vali_Erotin
    Set SMSBodyStr = SMSBodyStr_"Hei, "_RoomTeksti_" "_SiivousTyyppiTeksti_Erotin
    Set SMSBodyStr = SMSBodyStr_"}}"
    Set SMSBodyStr1 = $ZCONVERT(SMSBodyStr,"O","UTF8")
```

Kuva 19. Esimerkkikoodi viestisisällön muodostamisesta

6.7 ObjectScript – viestin lähetys

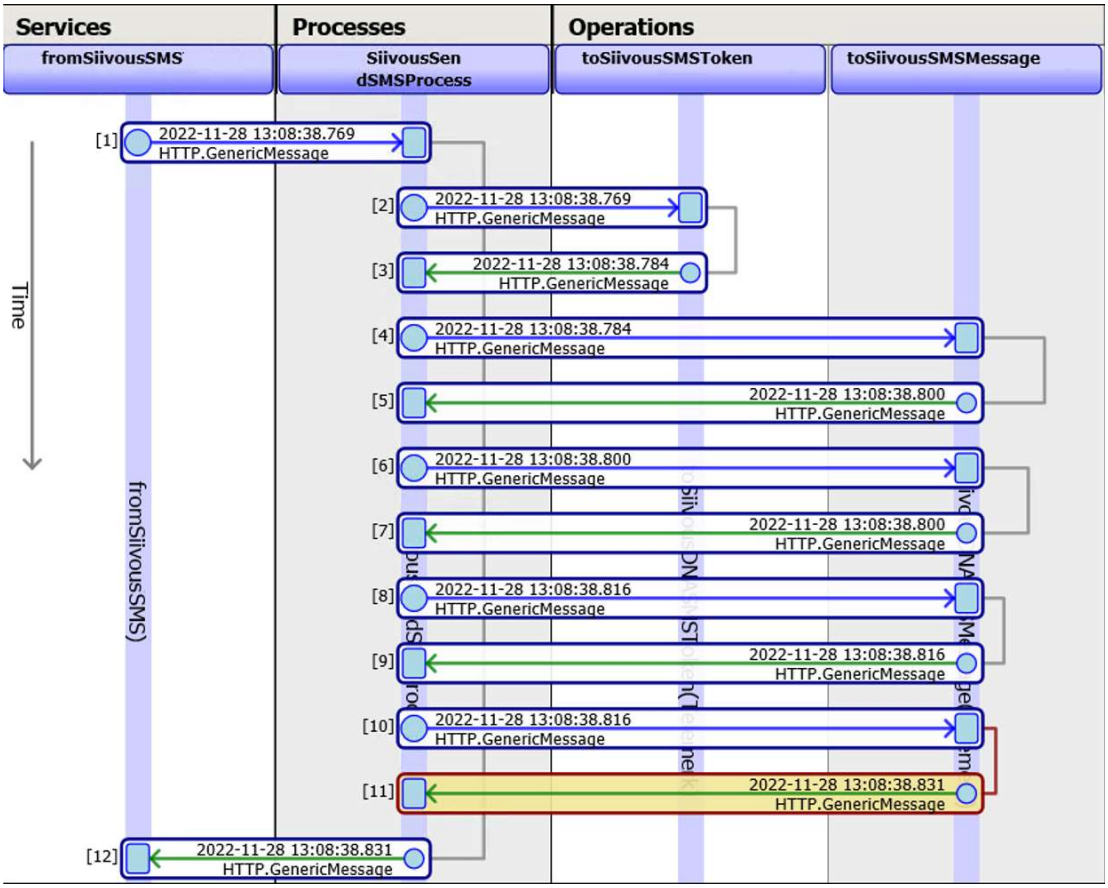
Business Process lähettää Business Operaatiolle (toSiivousSMSMessage) viestin eri puhelinnumerolla, niin monta kertaa kuin puhelinnumerolistalle on eri koodisäännöin kerättyjä numeroita löydetty. Silmukassa viestin lähetystä toistetaan, niin kauan, kunnes lista on tyhjä numeroista. Kuvassa 20 on esitetty koodi viestien lähettämiseen, missä tietosisältö tulee kuvan 19 muuttujista *SMSBodyStr1* ja *tInput.*

```
#Dim jsonReqStream As %Library.GlobalCharacterStream = ##class(%Library.GlobalCharacterStream).%New()
#Dim tOutX As %IO.StringStream = ##class(%IO.StringStream).%New()

Do tOutX.Write(SMSBodyStr1)
Do tOutX.Rewind()    // Siirtää Streamin osoittimen alkuun
Do jsonReqStream.CopyFrom(tOutX)


#Dim tRequest As EnsLib.HTTP.GenericMessage = ##class(EnsLib.HTTP.GenericMessage).%New(jsonReqStream,,tInput.HTTPHeaders)

Set tSC = ..SendRequestSync(..TargetConfigNameSMS, tRequest, .tResponse,..ResponseTimeout)
$$$TRACE("Checking SendRequestSync return status")
$$$ThrowOnError(tSC)

// Kutsuvalla järjestelmälle palautettava virheilmoitus
#Dim rResponse As EnsLib.HTTP.GenericMessage
#Dim tOut As %IO.StringStream
#Dim responseBody As %String = ""
#Dim tHttpStatus As %String = ""

If $ISOBJECT(tResponse)
{
    Set tHttpStatus = tResponse.HTTPHeaders.GetAt("StatusLine")
}

// If all OK
If $$$ISOK(tSC) && $Case(tHttpStatus,"HTTP/1.1 200 OK":1,:0)
{
    $$$TRACE("Status=Ok and response.HttpStatus="_tHttpStatus)
    Set pResponse = tResponse
    // No response or http status <> 200
} Else
{
    If $ISOBJECT(tResponse)
    {
        $$$TRACE("tSC is error and response.HttpStatus="_tHttpStatus)
        $$$TRACE("tResponse is an object so returning that to caller")

        Set tSC = $$$ERROR($$$GeneralError, tHttpStatus)
```

Kuva 20. Esimerkkikoodi viestin lähetyksestä

6.8 Ensemblen sanomanvälitys

Käytetty SMS-viestipalvelu ei tukenut massalähetystä, joten tästä syystä viestit rakennetaan yksitellen silmukassa ja jokainen viesti lähetetään erikseen. Kuvassa 21 on esitetty koko integraatioprosessi, missä ensin REST-siivouspyyntöviesti saapuu Business Serviceen (fromSiivousSMS). Business Service ohjaa viestin Business Prosessiin (SiivousSendSMSProcess), mistä lähetään REST-pyyntöviesti SMS-palvelun Business Operaatioon (toSiivousSMSToken) token-avainta varten. Saatu token-avain parametreineen luetaan Businees Prosessilla (SiivousSendSMSProcess), ja luodaan SMS-viestin sisältö. Viesti on lähetetty SMS-palveluun Business Operaatioon (toSiivousSMSMessage) neljä kertaa, mikä tarkoittaa neljää eri puhelinnumeroa. Viestin lähetys palautuu jokaisen lähetyskerran jälkeen Business Prosessiin, missä tarkistetaan puhelinnumerolista. Kun lista on tyhjä, kuittaa Business Prosessi ”HTTP/1.1 200 OK” -sanomat valmiiksi Ensemblelle.



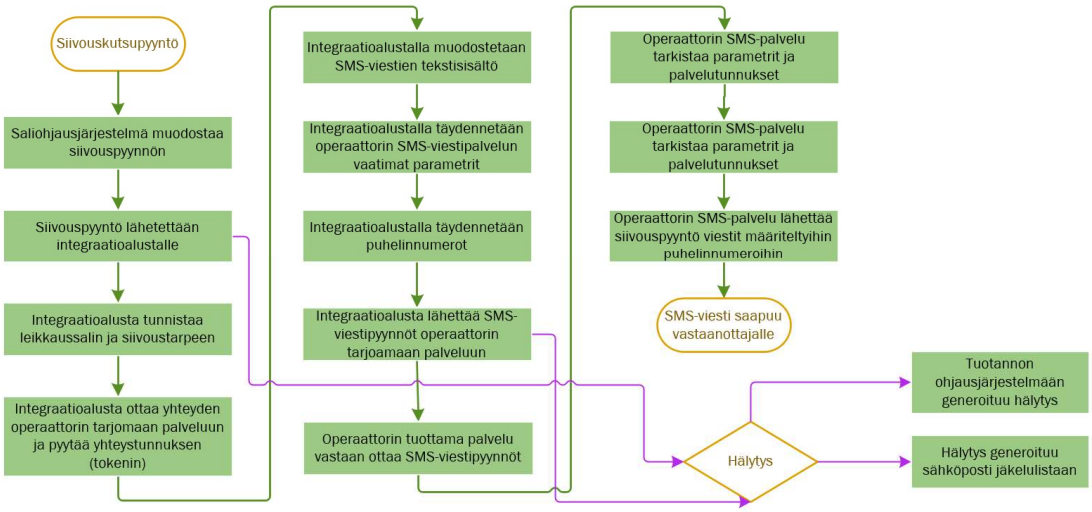
Kuva 21. Esimerkki Ensemblen sanomanvälityksestä

7 Toiminnallisuus

7.1 Toimintalogiikka

Vuokaaviossa tarkastellaan siivouskutsujärjestelmän toimintalogiikkaa Ensemblen näkökulmasta, koska sieltä on paras näkyvyys lähde- ja kohdejärjestelmiin (kuva 22). Ensemblen prosessilla luetaan lähdejärjestelmästä tulevat kutsut ja välitetään ne viestin välityspalveluun. Integraatiossa tapahtuvien hälytyksien monitorointi on myös rakennettu tapahtumaan automaattisesti Ensemblellä, mistä tiedot välitetään Efecteen.

Leikkaussalin ohjauspaneelista saadaan siivouskutsupyyntö viestin muodostustarpeesta Ensemblelle, missä muodostetaan tekstiviestin sisältö, joka lähetetään tekstiviestikeskuksen kautta haluttuihin vastaanottajien matkapuhelimiin. Viestien lähetys tapahtuu haluttuihin ja määriteltyihin numeroihin, jotka Ensembleltä lähetetään SMS-palvelun rajapintaan. Viestin sisältö määritellään Ensemblellä, eikä SMS-palvelu koske sen sisältöön. Integraatiossa SMS-palvelu ei kuittaa Ensemblelle, että viesti on toimitettu perille. Ensemble sen sijaan valvoo ja saa kuittauksen, että viesti on toimitettu onnistuneesti SMS-palveluun lähetettäväksi.



Kuva 22. Esimerkki siivouskutsujärjestelmän vuokaaviosta

7.2 SMS-palvelun esiselvitys ja tilausprosessi

Siivouskutsujärjestelmää toteutettaessa tehtiin esiselvitys, mitä olemassa olevia SMS-palveluita sairaalalla oli jo käytössä. Samalla selvitettiin, löytyisikö partnereilta ja alihankkijoilta jo olemassa olevaa palvelua SMS-viestien lähettämiseen. Sopivia SMS-palveluita ei löytynyt tai palvelun hinnoittelu muodostui syyksi, miksi päädyttiin uuden palvelun tilaamiseen ja pystyttämiseen.

Sopivan SMS-palvelun toimittajan löydyttyä palvelun tilausprosessi käynnistyi tilauslomakkeen täyttämisellä. Tilauksen dokumenttiin täytettiin mm. organisaatio, vastuuhenkilötiedot, tilattava palvelu, SMS-rajapinnan tekniset tiedot sekä liittyvän sovelluksen tiedot. Tilauksen, tilauksen käsittelyn ja palvelun käyttöönoton valmistuttua, pääsimme jatkamaan toteutusta viestinlähetysrajapintaa vasten.

7.2.1 SMS-palvelun palvelunkuvaus ja liittyminen palveluun

SMS-palveluun liittymistä varten toimitetut tekniset tiedot sekä palvelukuvaus- ja REST API -sovelluskehitys -dokumentit mahdollistivat viestipalvelun toteutuksen. Palvelun käyttöön liittyvät asiat olivat kuvattu palvelunkuvausdokumentaatiossa. Toteutuksen suunnittelun määritykset ja vaatimukset integraatioalustalle ohjelmoidulle integraatiolle SMS-palveluun liittymisen osalta olivat kuvattu REST API-sovelluskehitys dokumentissa. Lähtökohtana oli, että käyttö vaatii internet-yhteyden, jolla yhdistetään Ensemble operaattorin palvelinalustaan. Yhteydet toteutettiin salattuna IPSEC LAN-to-LAN toteutuksena.

7.3 Siivouskutsun vastaanotto

Leikkaussalin ohjauspaneelin siivoussovelluksen haluttua toimintoa painaessa, generoituu HTTP-pyyntökutsu Ensemblelle. POST-pyynnössä välitetään parametreinä siivottava leikkaussali ja siivoustyyppi (kuva 23).

HTTP-kutsu	**Data (JSON)**	**Paluuarvo**
POST /api/sali/{saliId}/toiminto	{ toiminto: toimintoTyyppi }	HTTP statuskoodi 200/OK

Kuva 23. Esimerkki siivouspyyntö HTTP-kutsu

Ensembleen on konfiguroitu *Business Service* kuuntelemaan *HTTP*-protokollan mukaista saapuvaa liikennettä määritettyyn porttiin. Leikkaussalin ohjauspaneeleihin on konfiguroitu portin tiedot, mitä käytetään, kun on tarve siivoukselle. Ensemble pollaa sovituin aikavälein porttiin mahdollisesti saapuvaa sanomaliikennettä. Kun Ensemble on havainnut uuden HTTP-sanoman eli saanut XML:n kääritytyn JSON-formaattisen POST-sanoman, välitetään tieto Ensemblen sisällä Business Prosessille (kuva 24).

```
<?xml version="1.0" ?>
<!-- type: EnsLib.HTTP.GenericMessage  id: 20207923 -->
<HTTPMessage xmlns:s="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <Stream>{ action: 2 }</Stream>
    <Type>GC</Type>
    <HTTPHeaders>
        <HTTPHeadersItem HTTPHeadersKey="CharEncoding"
        xsi:nil="true"></HTTPHeadersItem>
        <HTTPHeadersItem HTTPHeadersKey="EnsConfigName">fromSiivousSMS
        </HTTPHeadersItem>
        <HTTPHeadersItem HTTPHeadersKey="HTTPVersion">1.1</HTTPHeadersItem>
        <HTTPHeadersItem HTTPHeadersKey="HttpRequest">POST</HTTPHeadersItem>
        <HTTPHeadersItem HTTPHeadersKey="IParams">0</HTTPHeadersItem>
        <HTTPHeadersItem HTTPHeadersKey="RawParams"
        xsi:nil="true"></HTTPHeadersItem>
        <HTTPHeadersItem HTTPHeadersKey="TranslationTable">RAW</HTTPHeadersItem>
        <HTTPHeadersItem
        HTTPHeadersKey="URL">/api/room/a2/action</HTTPHeadersItem>
        <HTTPHeadersItem HTTPHeadersKey="content-length">18</HTTPHeadersItem>
        <HTTPHeadersItem HTTPHeadersKey="expect">100-continue</HTTPHeadersItem>
        <HTTPHeadersItem
        HTTPHeadersKey="host">ensemble.ad.fi:9</HTTPHeadersItem>
    </HTTPHeaders>
</HTTPMessage>
```

Kuva 24. Esimerkki Business Servicelle saapuvasta siivouskutsusta

7.4 Siivouskutsun prosessointi

Business Prosessi lukee Business Servicen lähettämän sanoman, jota varten on ohjelmoitu kustomoitu ObjectScript-luokka. Koodissa tarkistetaan, että saapunut sanoma on tyypiltään oikeanlainen HTTP generic message. Jokaisen ohjauspaneelin sanomassa tulee HTTP-otsikon (header) URL-tiedossa parametrinä huonetieto, joka luetaan koodilla. Koodi tunnistaa siivousta pyytävän leikkaussalin tämän tiedon perusteella ja tieto luetaan talteen SMS-viestin muodostamista varten. HTTP-viestin (HTTPMessage) runko-osan stream:ssä

saadaan ohjauspaneelilta valittu siivoustyyppi, joka luetaan myös talteen SMS-viestin muodostamista varten.

7.5 SMS-viestin käyttöoikeuspyyntö

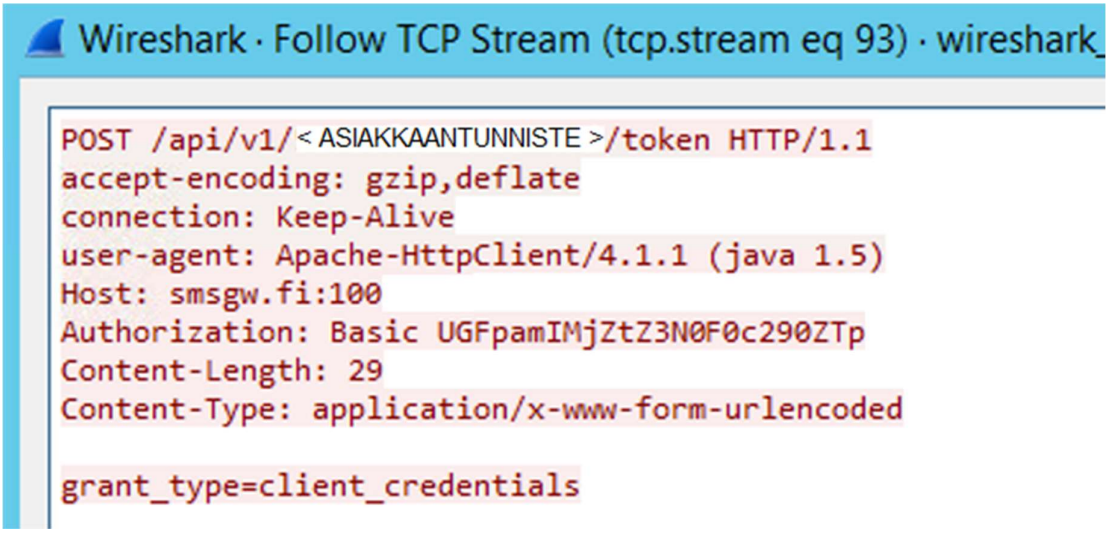
SMS-viestien muodostus on kaksivaiheisesti toteutettu integraatioalustan Business Prosessille. Ensimmäisessä vaiheessa pyydetään käyttöoikeuspyyntö (token) SMS-palvelusta ja toisessa vaiheessa SMS-viestit lähetetään. Integraatioalustan Business Process luo käyttöoikeuspyynnön Business Operaatiolle. Käyttöoikeuspyyntö on toteutettu REST API -kutsulla palveluntarjoajan SMS-palveluun. Palveluntarjoaja on toimittanut asiakaskohtaisen tunnuksen, joka on osa käyttöoikeuspyyntöä. Käytännössä Business Prosessissa täytetään ObjectScript-koodilla HTTPHeaderKey:n parametrit, jotta Business Operaatiolla on tarvittavat tiedot generoida HTTP-pyyntö käyttöoikeudesta SMS-palveluun.

Business Operaatiosta otetaan yhteys operaattorin tarjoamaan SMS-palvelun palvelimeen, porttiin ja URL:iin. Lisäksi Business Operaatio täydentää HTTPMessage:n Streamin sisällöksi palveluun kirjautumiseen vaaditun tunnuksen ja salasanan. Nämä tiedot välitetään SMS-palveluun. POST-operaation avulla hankitaan käyttöoikeustunnus (kuva 25).

```
<?xml version="1.0" ?>
<!-- type: EnsLib.HTTP.GenericMessage  id: 20207924 -->
<HTTPMessage xmlns:s="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <Stream>grant_type=client_credentials</Stream>
    <Type>GC</Type>
    <HTTPHeaders>
        <HTTPHeadersItem HTTPHeadersKey="CharEncoding" xsi:nil="true"></HTTPHeadersItem>
        <HTTPHeadersItem HTTPHeadersKey="EnsConfigName">fromSiivousSMS)</HTTPHeadersItem>
        <HTTPHeadersItem HTTPHeadersKey="HTTPVersion">1.1</HTTPHeadersItem>
        <HTTPHeadersItem HTTPHeadersKey="HttpRequest">POST</HTTPHeadersItem>
        <HTTPHeadersItem HTTPHeadersKey="IParams">0</HTTPHeadersItem>
        <HTTPHeadersItem HTTPHeadersKey="RawParams" xsi:nil="true"></HTTPHeadersItem>
        <HTTPHeadersItem HTTPHeadersKey="TranslationTable">RAW</HTTPHeadersItem>
        <HTTPHeadersItem HTTPHeadersKey="URL">/api/v1/< ASIAKKAANTUNNISTE >/token</HTTPHeadersItem>
        <HTTPHeadersItem HTTPHeadersKey="content-length">18</HTTPHeadersItem>
        <HTTPHeadersItem HTTPHeadersKey="content-type">application/x-www-form-urlencoded</HTTPHeadersItem>
        <HTTPHeadersItem HTTPHeadersKey="expect">100-continue</HTTPHeadersItem>
        <HTTPHeadersItem HTTPHeadersKey="host">ensemble.ad.fi:9</HTTPHeadersItem>
    </HTTPHeaders>
</HTTPMessage>
```

Kuva 25. Esimerkki Business Operaation token-pyynnöstä SMS-palveluun

Kuvassa 26 on Wiresharkilta otettua TCP-Stream:iä SMS-palveluun menevästä token-pyynnöstä. Onnistunut HTTP-vastaus (HTTP/1.1 200 OK) palauttaa tokenin parametreineen, ja paluusanomassa palautuu pääsytunnus, tunnustyyppi ja käyttöikä.



Kuva 26. Esimerkki Wireshark-työkalun monitorointinäkymästä

Ensemblellä paluusanoman vastaanottaa Business Prosessi, joka kaivaa paluusanomasta tarvittavat parametrit SMS-viestien muodostamiseen (kuva 27).

```
<?xml version="1.0" ?>
<!-- type: EnsLib.HTTP.GenericMessage  id: 20207925 -->
<HTTPMessage xmlns:s="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <Stream>
    {"access_token":"eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsImtpZCI6IjIzMDQ5ODE1MWMyMTRiNzg4ZGQ5N2YyMmI4NTQxMGE1In
    M4KG_wBy94NpA8HTpAfT-KEEz5izcrHMOu4zBS3_yizde2duQ8VpHvYHWLWyu20RqugA__xemRostyYrzSQp0hSJoRRHpJ9gUghNOk6Y1h4
    5Tlz4VNxokHLc5bMrM2Y_uZ7lO7yrA3zkHyK88efEfziL32fhRN5R1THcn5ioZ8Md5iW6_gNh1JkZf6mpI7L0ZCrA4WbiZgYlN4Ov51FqHf
    jBcllS70A8F7fmamWsXnvQ9lD-v4_tesW-KHjIupxytxK5kn2kNZVCg","token_type":"bearer","expires_in":86400}</Stream>
    <Type>BG</Type>
    <HTTPHeaders>
        <HTTPHeadersItem HTTPHeadersKey="CONNECTION">keep-alive</HTTPHeadersItem>
        <HTTPHeadersItem HTTPHeadersKey="CONTENT-LENGTH">605</HTTPHeadersItem>
        <HTTPHeadersItem HTTPHeadersKey="CONTENT-TYPE">application/json; charset=utf-8</HTTPHeadersItem>
        <HTTPHeadersItem HTTPHeadersKey="DATE">Mon, 07 Nov 2022 15:18:34 GMT</HTTPHeadersItem>
        <HTTPHeadersItem HTTPHeadersKey="SERVER">nginx/1.17.7</HTTPHeadersItem>
        <HTTPHeadersItem HTTPHeadersKey="STRICT-TRANSPORT-SECURITY">max-age=157680000</HTTPHeadersItem>
        <HTTPHeadersItem HTTPHeadersKey="StatusLine">HTTP/1.1 200 OK</HTTPHeadersItem>
    </HTTPHeaders>
</HTTPMessage>
```

Kuva 27. Esimerkki token-pyynnön paluusanomasta

Taulukossa 1 on esitetty token-pyynnön paluusanomassa tulleet pakolliset parametrit.

Nimi	HTTP	Tietotyyppi	Pakollinen/Valinnainen	Kuvaus
Content-Type	Header	String	Pakollinen	Määrittää HTTP- paluusanoman Body:n sisällön mediatyypin. Palautettu sisältö on JSON-muodossa, joten median MIME-tyypiksi on asetettu "application/json". Merkki enkoodaus on UTF-8.
access_token	Body	String Format: JSON Web Token (JWT)	Pakollinen	Valtuutuspalvelimen myöntämä käyttöoikeustunnus.
token_type	Body	String	Pakollinen	Myönnetyn tunnuksen tyyppi. Ainoa mahdollinen "bearer".
expires_in	Body	Integer	Pakollinen	Käyttöoikeustunnuksen käyttöikä sekunneissa.

Taulukko 1. Käyttöoikeuspyyntö -paluu sanoman parametrit

7.6 Token uudelleenkäyttö

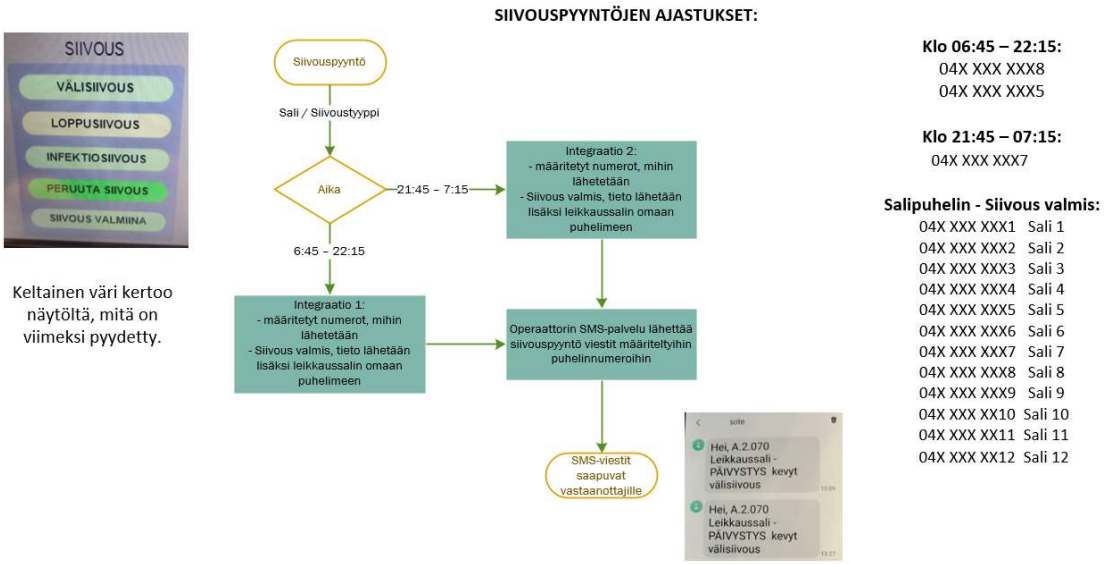
Kun Business Prosessi on saanut käyttöoikeustunnuksen (token), otetaan se ObjectScript koodilla talteen tekstityyppiseen muuttujaan, kuten käyttöoikeuspyynnön käyttöikä ja tyyppi. Samalla otetaan talteen ajanhetki, milloin käyttöoikeuspyyntö on saatu. Lisäämällä käyttöoikeuspyynnön käyttöikään saantihetken aika, saadaan arvo, milloin käyttöoikeuspyyntö vanhenee. Tämä mahdollistaa käyttöoikeuspyynnön uudelleenkäytön ja käyttöoikeuspyyntöä ei tarvitse pyytää uudestaan, mikäli jo pyydetty on voimassa ja on tarve lähettää uusia SMS-siivouspyyntöjä. Lasketusta vanhenemisajankohdasta on vähennetty 10 minuuttia, koska SMS-palvelussa voi olla viivettä ja ruuhkaa. Suojapuskuriaika estää sen, että ei synny tilannetta, missä järjestelmä yrittää lähettää SMS-viestejä käyttöoikeuspyynnön ollessa jo ummessa. Toisin sanoen koodissa on tiedossa käyttöoikeuspyynnön vanheneminen, joten tarvittaessa uusi käyttöoikeuspyyntö pyydetään uudelleen, kun vanha on umpeutunut tai lähellä umpeutumista.

7.7 Lähetettävät SMS-viestit ja ajastukset

Asiakas on määritellyt reunaehdoiksi sen, että laitoshuoltajille lähetään kaikki viestit. Leikkaussaleissa on omat puhelimet ja niihin puhelimiin lähetetään ainoastaan ”siivous valmis” -viestit kyseisestä leikkaussalista. Leikkauksia saleissa tehdään joka päivä. Ainoastaan haluttu ominaisuus on siivouskutsujen välittyminen klo 21.45-07.15 päivystysnumeroon ja klo 06.45-22.15 laitoshuollon numeroihin. Ajastukset ovat toteutettu Ensemblen automaattiajastuksilla.

7.7.1 Siivoustyypit

Leikkaussaleissa on tarve suorittaa erityyppisiä siivouksia sekä kuitata valmiit siivoukset tai perua siivoukset. Järjestelmässä varsinaisia siivoustyyppejä ovat välisiivous, loppusiivous ja infektiosiivous. Kuvassa 28 leikkaussalin kosketusnäyttöpäätteen siivoussovellus kertoo, miltä valikko näyttää. Lisäksi kuvassa on esitetty rakenne, miten erinumeroihin ohjataan viestejä.



Kuva 28. Siivouksien viestien ohjaussuunnitelma

7.8 SMS-viestin sisällönmuodostus ja lähetys

Jotta tarvittavasta siivouksesta voidaan muodostaa SMS-viestisisältö, tiedossa on oltava leikkaussali, siivoustyyppi sekä aktiviinen käyttöoikeuspyyntö. Business Prosessissa ObjectScript-koodin avulla rakennetaan määrätyn muotoinen viestirakenne, joka on HTTPMessage Stream:n sisältö. Viestin sisällöksi määritellään vastaanottajan numero, viestinlähettäjän tiedot ja viestin sisältö. HTTPHeaderKey parametrit täytetään Business Prosessissa, joista tärkein on ”authorization”-kenttään täytettävä tunnustyyppi ja käyttöoikeuspyyntö (token).

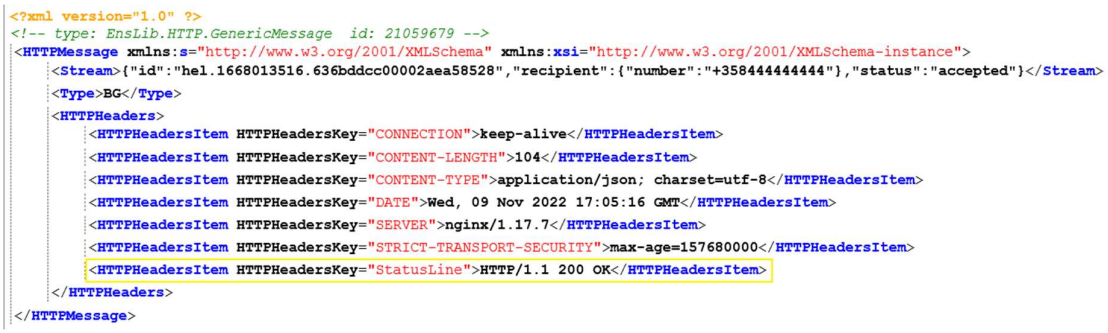
Kun tarvitut parametrit ovat täytetty, Business Prosessi välittää sanoman eteenpäin Business Operaatiolle SMS-viestin lähettämistä varten. Business Operaatiosta otetaan yhteys operaattorin tarjoamaan SMS-viestipalvelun palvelimeen, porttiin ja URL:iin. Lisäksi Business Operaatio täydentää HTTPMessage:n Streamin sisällöksi palveluun kirjautumiseen vaaditun tunnuksen ja salasanan. Nämä tiedot välitetään SMS-viestipalveluun. POST-operaatio sallii valtuutetun palvelusovelluksen lähettää lähtevän tekstiviestin (kuva 29).

```
<?xml version="1.0" ?>
<!-- type: EnsLib.HTTP.GenericMessage  id: 21059678 -->
<HTTPMessage xmlns:s="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <Stream>{"recipient":{"number":"0444444444"},"data":{"message":"Hei, TUOTANTO: Leikkaussali 1 - PROTEESI  siivous valmis"}}</Stream>
  <Type>GC</Type>
  <HTTPHeaders>
    <HTTPHeadersItem HTTPHeadersKey="CharEncoding" xsi:nil="true"></HTTPHeadersItem>
    <HTTPHeadersItem HTTPHeadersKey="Content-Type">application/json;charset=utf-8</HTTPHeadersItem>
    <HTTPHeadersItem HTTPHeadersKey="EnsConfigName">fromSiivousSMS</HTTPHeadersItem>
    <HTTPHeadersItem HTTPHeadersKey="HTTPVersion">1.1</HTTPHeadersItem>
    <HTTPHeadersItem HTTPHeadersKey="HttpRequest">POST</HTTPHeadersItem>
    <HTTPHeadersItem HTTPHeadersKey="IParams">0</HTTPHeadersItem>
    <HTTPHeadersItem HTTPHeadersKey="RawParams" xsi:nil="true"></HTTPHeadersItem>
    <HTTPHeadersItem HTTPHeadersKey="TranslationTable">RAW</HTTPHeadersItem>
    <HTTPHeadersItem HTTPHeadersKey="URL">/api/v1/< ASIAKASTUNNISTE >/sms</HTTPHeadersItem>
    <HTTPHeadersItem HTTPHeadersKey="authorization">Bearer
    eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsImtpZCI6IjIzMDQ5ODE1MWMyMTRiNzg4ZGQ5N2YyMmI4NTQxMGE1In0.eyJhcHBpZCI6IjVhNzgwZmE4LWNkOTktNDdkZC05ZTIxLW
    6y9JsqWThLNwrre9uMadBUAyrYicCRDH1f3Tp01Dv_pzfI36mfqV1vbblzhF2FUj3SOdp9HcAomHRDs3AeVNspnNhstqIczUsB73lypDFu0B_jNuC0pxotkmy3foKHQxI5lMktwvot
    cpWY21_WMjj9FlI9McWA</HTTPHeadersItem>
    <HTTPHeadersItem HTTPHeadersKey="content-length">18</HTTPHeadersItem>
    <HTTPHeadersItem HTTPHeadersKey="content-type">application/x-www-form-urlencoded</HTTPHeadersItem>
    <HTTPHeadersItem HTTPHeadersKey="expect">100-continue</HTTPHeadersItem>
    <HTTPHeadersItem HTTPHeadersKey="host">ensemble.ad.fi:1</HTTPHeadersItem>
  </HTTPHeaders>
</HTTPMessage>
```

Kuva 29. Esimerkki Business Operaatiolta lähtevästä SMS-viestistä

Onnistunut HTTP-vastaus (HTTP/1.1 200 OK) palauttaa onnistuneesta sanomanvälityksessä operaattorin SMS-viestipalveluun (kuva 30). Palvelun tarjoavalla operaattorilla ei ollut tarjolla viestinvälityspalvelussaan viestien massalähetysominaisuutta (puhelinnumerolista), joten lähetettävät numerot luetaan koodissa talteen omaan listaan ja viestit lähetetään yksitellen, kunnes lista on tyhjä. Lähetyssilmukka Business Prosessin ja Business Operaation välillä toistuu niin kauan, kuin kaikki halutut viestit ovat lähetetty. SMS-viestipalvelu lähettää

viestin määritettyihin puhelinnumeroihin ja näin vastaanottaja tietää, missä siivous on tarpeellista, valmistunutta tai peruttu.

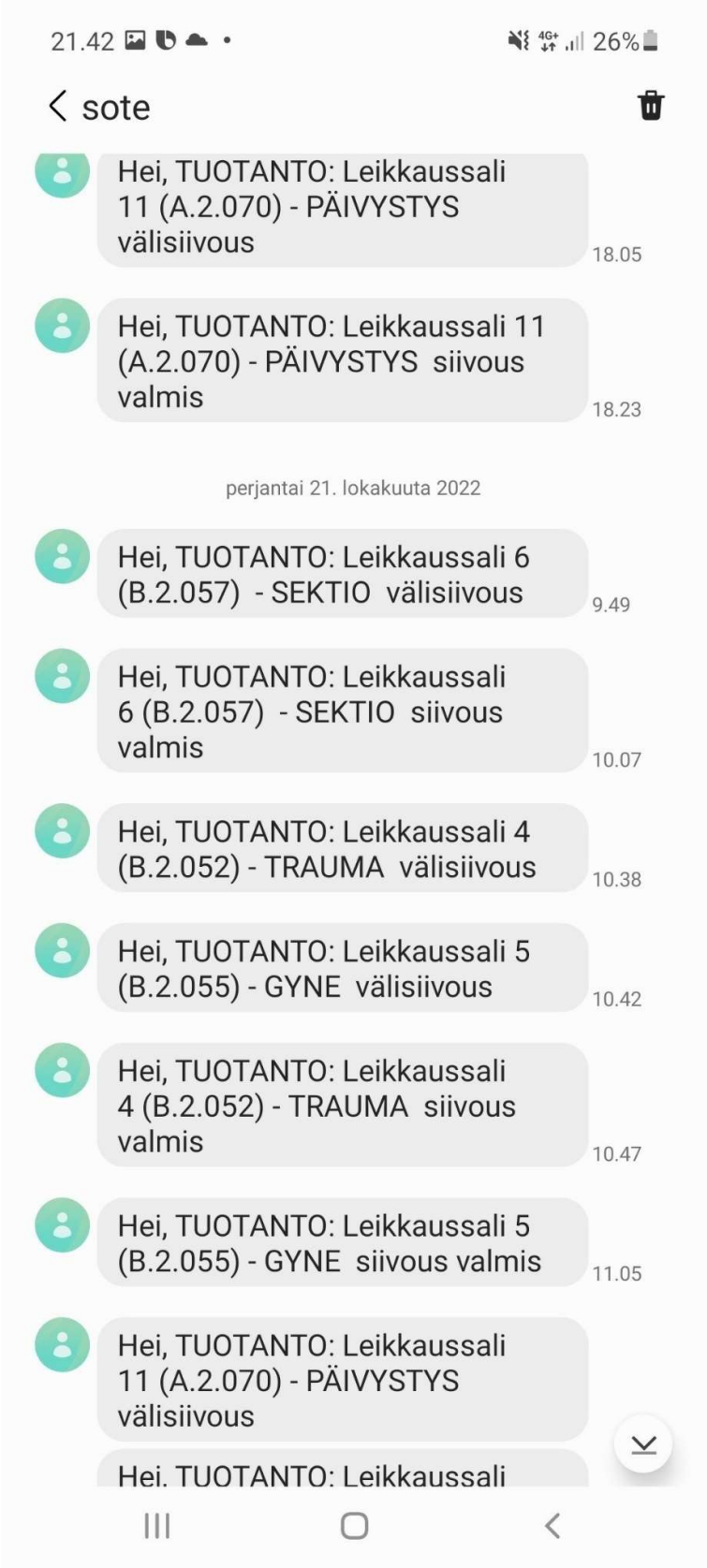


```
<?xml version="1.0" ?>
<!-- type: EnsLib.HTTP.GenericMessage  id: 21059679 -->
<HTTPMessage xmlns:s="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <Stream>{"id":"hel.1668013516.636bddcc00002aea58528","recipient":{"number":"+358444444444"},"status":"accepted"}</Stream>
  <Type>BG</Type>
  <HTTPHeaders>
    <HTTPHeadersItem HTTPHeadersKey="CONNECTION">keep-alive</HTTPHeadersItem>
    <HTTPHeadersItem HTTPHeadersKey="CONTENT-LENGTH">104</HTTPHeadersItem>
    <HTTPHeadersItem HTTPHeadersKey="CONTENT-TYPE">application/json; charset=utf-8</HTTPHeadersItem>
    <HTTPHeadersItem HTTPHeadersKey="DATE">Wed, 09 Nov 2022 17:05:16 GMT</HTTPHeadersItem>
    <HTTPHeadersItem HTTPHeadersKey="SERVER">nginx/1.17.7</HTTPHeadersItem>
    <HTTPHeadersItem HTTPHeadersKey="STRICT-TRANSPORT-SECURITY">max-age=157680000</HTTPHeadersItem>
    <HTTPHeadersItem HTTPHeadersKey="StatusLine">HTTP/1.1 200 OK</HTTPHeadersItem>
  </HTTPHeaders>
</HTTPMessage>
```


Kuva 30. Esimerkki SMS-viestipalvelun paluusanomasta

7.9 Vastaanottajan SMS-viestit

Siivouskutsujärjestelmä toimii lähes reaaliaikaisesti, sillä leikkaussalin ohjauspaneelin pyynnön saapumisesta integraatioalustalle kuluu maksimissaan 5 sekuntia (määritelty pollausväli). Integraatioalustaa ei ole järkevää kuormittaa pienentämällä pollausväliä, sillä alustalla voi olla satoja tai jopa tuhansia integraatioita, mitkä kuormittavat samalla tavalla palvelimen resursseja. Integraatioalustalla sanomien lähettäminen vie noin 0,2-0,3 sekuntia, minkä jälkeen operaattorin SMS-palvelu lähettää viestit. Yleensä puhelin saa viestin noin 2-3 sekunnissa ohjauspaneelin painalluksesta, joten tällä perusteella voidaan sanoa järjestelmää lähes reaaliaikaiseksi. Ulkoasultaan lähetyt viestit ovat samanlaisia, mutta vastaanottaja, leikkaussali ja siivouksen tyyppi ovat viesteissä muuttuvia. Lisäksi viesteissä näkyy lähetysaika, minkä perusteella pystytään seuraamaan, mitkä leikkaussalit ovat missäkin tilassa (kuva 31).



Kuva 31. Esimerkkiviestejä siivoustapahtumista

7.10 Virhetilanteet

Siivouskutsujärjestelmästä nousee toiminnanohjausjärjestelmä Efecteen herätteitä, mikäli lähde-, kohde- tai Ensemble -järjestelmässä tapahtuu jotain ennalta arvaamatonta. Ensemblelle on toteutettu valvonta leikkaussalien ohjauspaneelien osalta siten, että 12 tunnin inaktiivisen sanomaliikenteen jälkeen Efecteen muodostuu heräte. Mikäli tokenin nouto, viestin lähetys SMS-palveluun epäonnistuu tai ei saada yhteyttä, näistä tilanteista muodostetaan myös herätteitä. Mikäli Ensemblen saama viestisisältö leikkaussalin ohjauspaneelilta tai SMS-palvelun tokenista on virheellistä, muodostuu tilanteista herätteitä.

8 Yhteenveto, jatkokehitys ja pohdinta

8.1 Toteutuksen arviointi

Projektin toteutus käsitti siivouskutsujärjestelmän kokonaistoimituksen alusta loppuun, mikä tarkoitti osallistumista esiselvittelyyn, tarjouksen tekoon, suunnitteluun, toteutukseen, testaukseen, käyttöönottoon ja ylläpitoon. Tämä tarkoitti sitä, että työ sisälsi yhteistyötä lähde- ja kohdejärjestelmätoimittajien kanssa. Lisäksi yhteistyö useiden eri organisaatiotahojen kanssa oli tarpeellista, koska heidän toimittamia ICT-palveluita tarvittiin järjestelmän toteuttamiseen.

Esiselvitysvaiheessa selvitettiin ohjauspaneelin rajapinnat ja toimintalogiikka, jotta muodostettiin käsitys, miten integraatioalustalla käsitellään saapuvia siivouspyyntöjä. SMS-viestipalveluiden tuottajaa kartoitettiin ensin sisäisesti 2M-IT:ltä ja Hyvinvointialueen olemassa olevista palveluista. Tarkoitukseen sopivaa tai hinnoittelultaan kilpailukykyistä palvelua ei löytynyt, joten päädyttiin uuden palvelun hankkimiseen. Tämä tarkoitti uusia sopimuksia ja selvitystyötä, miten viestipalveluun liitytään.

Kun tarvittavat tiedot olivat kasassa, toteutuksesta piirrettiin arkkitehtuurikuva. Arkkitehtuurin hyväksynnän jälkeen integraation toteutusvaihe alkoi kehitysympäristössä. Ensimmäiseksi rakennettiin ohjauspaneelin siivouskutsun käsittely, missä luettiin haluttu siivoustyyppi ja leikkaussalitieto. Suunnitteluvaiheessa oli tiedossa siivoustyypit sekä leikkaussalit, jotka pystyttiin yhdistämään ohjauspaneelin pyyntöihin. SMS-viestin muodostamista varten rakennettiin määritellyn muotoinen token-pyyntö viestipalveluun ja paluusanomasta otettiin talteen token-avain. Saatua token-avainta käytetään lähes voimassaoloajan loppuun asti viestien lähetyksessä ja uusi pyydetään vanhan umpeutuessa. Viestisisällön täyttäminen tehdään saatujen parametrien perusteella ja sen jälkeen viesti on valmis lähetettäväksi. Tuki useisiin numeroihin viestien lähettämisestä toteutettiin silmukan avulla, kun taas päivystäjän puhelimeen viestien lähetys toteutettiin integraatioalusta-ajastuksien kautta.

Toteutusmielessä projektin läpivienti oli mielenkiintoinen ja haasteellinen, koska uuden kehittäminen on haasteellista. Määrittelyjen perusteella toteutus antaa mahdollisuuden väärinymmärryksiin ja virheisiin, jotka ilmenevät yleensä vasta testausvaiheessa. Järjestelmien integroiminen toimimaan keskenään yhteen on haasteellista, mutta palkitsevaa, kun toteaa järjestelmän toimivan suunnitellulla tavalla.

8.2 Toteutuksen haasteet

Kokemus ObjectScript-ohjelmoinnin perusteista hyödyttää ja nopeuttaa toteutusta, joten kielelle ominaiset asiat kannattaa opiskella ja ymmärtää, ennen kuin aloittaa ohjelmoinnin. Haasteellista ohjelmoinnin kannalta oli myös se, että tuotantoympäristö oli vuodelta 2016. Uudempien työkalujen HealthShare ja Iris mukana on tullut ObjectScritpt:in uusia toiminnallisuuksia ja luokkia, joita vanha Ensemble -ympäristö ei tue. Lisäksi InterSystemsin internetsivustojen koodiohjeet ovat yleensä uudella tavalla toteutettu, joten toteutusesimerkkejä vanhalla tavalla oli vaikeampi löytää. Toteutuksessa on siis jouduttu tekemään asioita eri tavalla, esimerkiksi merkkien lukemista merkkimuotoisista muuttujista merkki kerrallaan, koska saatavilla ei ollut sellaista luokkaa, joka olisi osannut sen suoraan tehdä.

Aikataulullisesti työn toteuttamiseen varattiin reilusta aikaa, mutta kiireiset hyvinvointialueiden projektit ajoivat ohi ja tämän toteutus jäi viime metreille. Toteutus meni kuitenkin tuotantoon suunnitellussa aikataulussa, joten siltä osin projekti meni maaliin ajallaan. Toteutettua koodia on kehitetty eteenpäin toteutuksen ollessa jo tuotannossa. Konfigurointimuutoksia ja -asetuksia on muutettu tuotannossa, mutta virheitä toteutuksessa ei ole havaittu muutaman kuukauden tuotannossa pyörimisen jälkeen.

8.2.1 Väärä palvelutunniste

Projektin aikana SMS-palveluiden toimittaja toimitti väärän palvelutunnisteen, millä kirjautuminen haluttuun palveluun osoittautui mahdottomaksi. Usean päivän vianselvityksen jälkeen ongelma ratkesi vertaamalla toimitettuja tietoja ja määrittelydokumentaatiota. Toimitetussa palvelutunnisteessa oli vähemmän merkkejä kuin määrittelydokumentaation esimerkissä. Toimittajan asiakaspalvelua oli jo useita päiviä aiemmin pyydetty tarkistamaan, että palvelutunniste on oikein. Asiakaspalvelu ei ollut sitä tehnyt ja vasta kun heille se kerrottiin, löytyi vihdoinkin oikea palvelutunniste. Oikealla palvelutunnisteella ohjelmakin sai yhteydet viestipalveluun ja kehitystyö jatkui.

8.2.2 Väärät puhelinnumerot

Tuotannon käyttöönoton jälkeen tuli ilmoitus, että erikoistilanteessa ”siivous valmis” -viestit eivät mene perille. Tässä tilanteessa lähetetään viesti laitoshuollon lisäksi leikkaussalin puhelimeen. Tarkistaminen ja testaaminen osoitti, että järjestelmä toimii ja viestit lähtevät oikein. Tilanne siivousjärjestelmän toteuttajan ja viestijärjestelmän toimittajan kannalta oli

erikoinen, koska testatessa viestit tulivat perille niin testistä kuin tuotannostakin. Vertailemalla ohjelmakoodia testin ja tuotannon välillä tulos oli, että koodit ympäristöissä olivat identtisiä. Myöhemmin osoittautui, että toimitetut puhelinnumerot olivat vääriä, ja ”siivous valmis” -viestit menivät hetken vääriin numeroihin.

8.3 Jatkokehitysmahdollisuudet

Ensemblelle toteutettu SMS-viestipalvelu tarjoaa hyvinvointialueella mahdollisuuksia alkaa kehittämään uusia viestipohjaisia palveluita muihin sairaalajärjestelmiin. Tämä vaatii vain lähdejärjestelmien sovittamista ja viestisisällön muokkaamista. Kohdejärjestelmän toiminnallisuudet ovat jo kerran ratkaistu, joten muutoksia kohdepäähän ei tarvita. Olemassa olevaa palvelua voi hyödyntää myös muihin käyttötarkoituksiin, mikäli sellaisia tarpeita ilmenee.

2M-IT:n näkökulmasta integraatioalustan toiminnallisuus on monistettavissa muihin hyvinvointialueiden ympäristöihin, mikäli on tarve rakentaa SMS-viestinvälityspalveluita ja käytössä on saman operaattorin SMS-palvelua. Lähdejärjestelmien sovittaminen ja viestin muodostus ovat niitä muutoksia, joita kustomoidun luokan siirtämisen lisäksi palvelun käyttökuntoon saattaminen vaatii muissa ympäristöissä.

8.3.1 Tilannekuvanäyttö

Alun perin asiakkaalla oli toiveita, että toteutukseen voisi liittää lähes reaaliaikaisen tilannekuvanäytön eri leikkaussalien siivouksien tilanteesta. Tätä ominaisuutta ei ole toteutettu ja asiakkaan käyttökokemuksien perusteella heillä ei välttämättä ole tarvetta tilannekuvanäytölle. Hyvinvointialueiden budjetit ovat rajallisia, joten näillä näkymin resurssit laitetaan kriittisempien integraatioiden toteuttamiseen. Kustannukset eivät kata saatavaa hyötyä tilannekuvan toteutukselle, joten toteutus on toistaiseksi jäissä.

8.3.2 Vanhojen leikkaussalien liittäminen siivouskutsujärjestelmään

Toisena jatkokehityskohteena oli sairaalan vanhojen leikkaussalien liittäminen siivouskutsujärjestelmään. Tätä hanketta rajoittaa se, että vanhoissa leikkaussaleissa ei ole samanlaista salinohjausjärjestelmään kuin uusissa. Tämä tarkoittaa sitä, että tarve olisi toteuttaa erillinen työasemasovellus tai puhelinapplikaatio. Sovelluksella lähetettäisiin siivouskutsuja

integraatioalustalle, josta viestit muodostettaisiin viestinvälityspalveluun ja lopulta lähetyspalvelu lähettäisi viestit. Tätä jatkokehityskohdetta ei ole viety eteenpäin, mutta sitä ei ole toistaiseksi myöskään hylätty. Tämän kehitys otetaan työn alle mahdollisesti tulevaisuudessa.

8.3.3 Siivouskutsujärjestelmän ohjauspaneelin värit

Leikkaussalin ohjausjärjestelmän näyttöpaneelissa olevien sovelluksien väritykset ovat epäselviä. Värit ovat liian hailakoita ja niiden tarkoituksien erottaminen toisistaan vaatii tarkkaavaisuutta. Yleisesti on tunnettua, että puna-viherväri ovat hankalia erottaa toisistaan, mikäli värinäkökyvyn kanssa on haasteita. Yksi jatkokehityksen kohde on Leikkaussalin toiminnanohjausjärjestelmän ohjauspaneelin näytön värityksen uudelleen suunnittelu, joka ei koske pelkästään siivoussovellusta, vaan myös kaikkia muita ohjauspaneelista löytyviä sovelluksia.

8.4 Pohdinta

Kehittämishankkeen tavoitteena oli toteuttaa siivouskutsujärjestelmä tekstiviestein, jonka avulla laitoshuoltajat ja leikkaussalista vastaava saa lähes reaaliaikaisen tiedon siivouksen tilanteesta koskien 12 uutta leikkaussalia. Sairaala toimialaosaamisen yhdistäminen tekniikkaan uudessa ympäristössä tarjosi haasteen ratkaistavaksi. Kehittämishanke oli eräänlainen näköalapaikka nykyaikaiseen sairaalateknologiaan alusta loppuun. Hankkeen aikana piirrettiin arkkitehtuurikuvaus järjestelmästä ja hyväksytettiin asiakkaan prosessin mukaisesti, minkä jälkeen oli mahdollista saada tietoliikenneavaukset järjestelmien välille. Tavoite toteutettiin yhdistämällä leikkaussalien toiminnanohjausjärjestelmän näyttöpaneelien rajapinta integraatioalustaan ja integraatioalustan rajapinta SMS-palveluun, mistä viestit lähetettiin määriteltyihin puhelinnumeroihin määritetyillä kriteereillä. Lopputuloksena oli siis tuotantokäytössä oleva järjestelmä, mitä sairaalassa ei ole aiemmin ollut.

Integraatioalustalle Business Prosessiin ohjelmoitiin toimintalogiikka ObjectScript:llä, missä päätoiminnallisuudet ohjelmoitiin ja testattiin palasissa omina ohjelman osina. Toteutuksen neljä päätoiminallisuutta olivat siivouspyynnön käsittely, tokenin käsittely, viestisisällön muodostaminen ja viestien lähetys tekstiviestipalveluun. Ohjelmoinnin haasteellisin osa työstä oli sovittaa erilaiset käyttötapaukset yhteen silmukkaan, millä perusteella viestit lähetettiin vastaanottajilleen. Ohjelmointivaiheessa viestien lähetykseen oli järkevää asettaa maksimimäärärajoitus, millä estetään ”satojen” tai ”tuhansien” viestien massalähetys.

Lähetyssilmukoita ohjelmoitaessa on aina olemassa virheen riski, mikä tässä tapauksessa realisoituessaan olisi voinut tarkoittaa kallista laskua viestien päätyessä maksulliseen viestipalveluun.

Suunnitellut ominaisuudet saatiin testattua ja toteutettua suunnitellun aikataulun mukaisesti, ja järjestelmä meni tuotantokäyttöön syyskuussa 2022 uuden sairaalan aloittaessa toimintansa. Käyttökokemuksia tai palautetta järjestelmästä ei ole saatu, joten päätöksiä jatkokehittämisestä tämän hankkeen osalta ei ole vielä kirjottamisen hetkellä. Oletus on, että rakennettua viestipalvelua tullaan sairaalassa laajentamaan ainakin muihin sovelluksiin. Hyvinvointialueiden muutostyöt ovat käynnissä parhaillaan koko Suomessa, koska hyvinvointialueet (HVA) aloittavat toimintansa 1.1.2023. Tämä tarkoittaa ei HVA-hankkeiden viivästymistä, koska HVA työt ovat luonnollisesti nyt etusijalla. Joka tapauksessa suunnitelmissa on siivouskutsujärjestelmän jatkokehittäminen 2M-IT:n toimesta asiakkuudesta riippumatta.

Lähteet

2M-IT. 2022. Ratkaisut sosiaali- ja terveyspalveluiden kehittämiseksi. Viitattu 5.10.2022. Saatavissa https://2m-it.fi/

2M-IT. 2020. 2M-IT esittely: Sovelluspalvelut. Viitattu 24.10.2022. Saatavissa https://2m-it.fi/wp-content/uploads/2020/11/Esittely-2M-IT-Sovelluspalvelut-20201110_.pdf

Alfame. 2018. Järjestelmäintegraatio, mitä se on selkokielellä?. Viitattu 5.10.2022. Saatavissa https://www.alfame.com/ajankohtaista/jarjestelmaintegraatio-mita-se-on-selkokielella#:~:text=Integraatio%20tarkoittaa%20eri%20tekniikoilla%20tai,sek%C3%A4%20tiedon%20liittymisen%20toisiin%20j%C3%A4rjestelmiin

Campbell, J. 2018. Defining Health IT: MUMPS. Viitattu 24.11.2022. Saatavissa https://blog.galenhealthcare.com/2018/09/13/defining-health-it-mumps/

DNA. 2022. DNA SMS Gateway. Viitattu 28.10.2022. Saatavissa https://www.dna.fi/wholesale/viestinta/sms-gateway

Efecte. 2022. Viitattu 29.10.2022. Saatavissa https://www.efecte.com/fi/

Flashnode Oy. 2022. Integraatiot Viitattu 24.10.2022. Saatavissa https://www.itewiki.fi/opas/integraatiot/

Fredrich,T. 2022. Using HTTP Methods for RESTful Services. Viitattu 1.11.2022. Saatavissa https://www.restapitutorial.com/lessons/httpmethods.html

GE Healthcare. 2021. Vaasan hybridileikkaussali – paljon enemmän kuin leikkaussalin ja angiosalin summa. Viitattu 2.11.2022. Saatavissa https://www.gehealthcare.fi/insights/article/vaasan-hybridileikkaussali

Helsingin kaupunki. 2020. Helsingin kaupungin API-linjaukset. Viitattu 30.10.2022. Saatavissa https://digi.hel.fi/documents/247/Helsingin_kaupungin_API-linjaukset_2020.pdf 4-5.

Hiltula, K. 2021. Onko soten tietojärjestelmien muutosbudjetti miljardia euroa vai puolet siitä? Hyvinvointialueiden ja ministeriön laskelmat eroavat rajusti. Viitattu 20.10.2022. Saatavissa https://yle.fi/a/3-12235923

InterSystems. 2022. InterSystems IRIS Data Platform 2022.2 Viitattu 14.10.2022. Saatavissa https://docs.intersystems.com/irislatest/csp/docbook/DocBook.UI.Page.cls

InterSystems. 2018. Using Caché ObjectScript. Viitattu 16.10.2022. Saatavissa https://docs.intersystems.com/latest/csp/docbook/DocBook.UI.Page.cls? KEY=GCOS

InterSystems. 2022a. Introduction to Interoperability Productions. Viitattu 16.10.2022. Saatavissa https://docs.intersystems.com/irislatest/csp/docbook/DocBook.UI.Page.cls?KEY=EGIN_intro

InterSystems. 2022b. Using the Management Portal. Viitattu 25.10.2022. Saatavissa https://docs.intersystems.com/irislatest/csp/docbook/DocBook.UI.Page.cls?KEY=GSA_USING_PORTAL

InterSystems. 2022c. Introduction to Studio. Viitattu 25.10.2022. Saatavissa https://docs.intersystems.com/irislatest/csp/docbook/DocBook.UI.Page.cls?KEY=GSTD_INTRO

InterSystems. 2022d. Introduction to the Terminal. Viitattu 25.10.2022. Saatavissa https://docs.intersystems.com/irislatest/csp/docbook/DocBook.UI.Page.cls?KEY=GTER_INTRO

Kuntaliitto. 2020. Asiakas- ja potilastietojärjestelmien tilannekuva ja sen analyysi 2020. Viitattu 15.10.2022. Saatavissa https://www.kuntaliitto.fi/sites/default/files/media/file/APTJ-tilannekuva2020_AKUSTI110620_0.pdf

Linnkake, T. 2022. Tekstiviesti täyttää 30 vuotta – vaikka käyttö vähenee, yhdessä tapauksessa se on ehdoton: ”Mikään ei pysty samaan”. Viitattu 22.11.2022. Saatavissa https://www.is.fi/digitoday/mobiili/art-2000009178455.html

Peffers, K.,Tuunanen, T., Gengler, C., Rossi, M., Hui, W., Bragge J. 2006. The design science research process: A model for producing and presenting information systems research 88-94.

SoapUI. 2022. Accelerating API Quality Through Testing. Viitattu 26.10.2022. Saatavissa https://www.soapui.org/

Traficom. 2021. Ohje yhdyskäytäväratkaisujen suunnitteluperiaatteista ja ratkaisumalleista Viitattu 20.11.2022. Saatavissa https://www.kyberturvallisuuskeskus.fi/sites/default/files/media/file/Yhdyskaytavaratkaisuohje.pdf

TRPlane.com. 2021. Mitä REST API tarkoittaa. Viitattu 30.10.2022. Saatavissa https://www.trplane.com/fi/mit%C3%A4-rest-api-tarkoittaa/

w3schools.com 2022a. What is JSON?. Viitattu 3.11.2022. Saatavissa https://www.w3schools.com/whatis/whatis_json.asp

w3schools.com 2022b. XML Tutorial. Viitattu 3.11.2022. Saatavissa https://www.w3schools.com/xml/

Wireshark. 2022. About Wireshark. Viitattu 26.10.2022. Saatavissa https://www.wireshark.org/