ERIKOISSAIRAANHOIDON ASIAKASHALLINNAN TEHOSTAMINEN

Tekstiviestien käyttö erikoissairaanhoidon jononhallinnan työkaluna

LAHDEN AMMATTIKORKEAKOULU
Tietotekniikan koulutusohjelma
Ohjelmistotekniikan suuntautumisvaihtoehto
Opinnäytetyö
Kevät 2007
Matti Aronen

Lahden Ammattikorkeakoulu
Tietotekniikan koulutusohjelma

MATTI ARONEN: Erikoissairaanhoidon asiakashallinnan tehostaminen
Tekstiviestien käyttö erikoissairaanhoidon jononhallinnan työkaluna

Ohjelmistotekniikan opinnäytetyö, 39 sivua

Kevät 2007

TIIVISTELMÄ

Tämä opinnäytetyö käsittelee asiakkuudenhallinnan tehostamista tekstiviesti-palvelun avulla. Sairaaloiden toimenpideaikoja muutetaan ja peruutetaan suhteettoman paljon. Tavoitteena on säästää työntekijöiden aikaa ja siten tehostaa yhteiskunnan varojen käyttöä nostamalla sairaalan käyttöastetta.

Työ alkaa sairaalan ajanvarausprosessin selvittämisellä. Kun se tunnetaan hyvin, voidaan myös pyrkiä sen tehostamiseen. Prosessi käydään läpi Päijät-Hämeen keskussairaalan päiväkirurgian yksikön jononhoitajan kanssa ja sen eteneminen dokumentoidaan.

Työn teoriaosa keskittyy tutkimaan erilaisia keinoja lähettää ja vastaanottaa tekstiviestejä ohjelmallisesti. Tutkimuksen tarkoituksena on saada riittävät tiedot sopivimman keinon valintaan. Tekstiviestiliikenne on mahdollista toteuttaa kahdella tavalla: joko käyttämällä GSM-laitetta tai kytkeytyä tekstiviesti-keskukseen niiden tarjoamien rajapintojen kautta.

Tekstiviestin välityksen lisäksi on pohdittu, miten voidaan asettaa useita palveluita saman puhelinnumeron taakse: miten voidaan tunnistaa, mitkä viestit kuuluvat millekin palvelulle.

Tutkimusosion jälkeen on kuvattu palvelun toteutus. Palvelu on kuvattu rakenteellisesti sekä toiminnallisesti UML-kaavioiden avulla.

Työn tuloksena luodaan tekstiviestipalvelu, joka onnistuu vastaamaan sille asetettuihin tavoitteisiin. Jononhoitajan aikaa säästyy, kun asiakkaiden kanssa kommunikointiin kulutetaan vähemmän aikaa.

Asiasanat: tekstiviestipalvelu, tekstiviestirajapinnat, asiakkuudenhallinta, jonotuspalvelu

Lahti University of Applied Sciences
Faculty of Technology

MATTI ARONEN: Making client management more efficient in special health care
Utilization of SMS messages in special health care queue management

Bachelor's Thesis of Software Engineering, 39 pages

Spring 2007

ABSTRACT

This thesis deals with making client management more efficient by utilizing SMS services. It is not uncommon to reschedule or cancel appointment times in hospitals. The goal was to save employees time and public funds by improving the utilization rate of the hospital.

The work began by studying the operation model in client management. The process was discussed and documented with the nurse who manages the client queue in the Päijät-Häme Central Hospital

The theory part of the thesis examines different methods of sending and receiving SMS messages from the software point of view. The purpose was to gain enough information about different technologies to be able to decide which offers the best approach to current situation. SMS traffic can be implemented with either a GSM device or by utilizing an SMS center over its interface.

In addition to sending and receiving SMS messages, the possibility to establish multiple services on one phone number was investigated, in other words, how it is possible to tell from an SMS message what service it was meant for.

In the empirical part, the implementation of the service is described. The structure and functionality of the service is illustrated with the help of UML charts.

The product of the thesis is a SMS service which fulfills the expectations set to it. The time of the client queue managing nurse is saved, because communication with clients takes less of her time.

Key words: SMS service, SMS interfaces, customer management, queuing service

SISÄLLYS

LYHENTEET JA TERMIT

Lyhenne / Termi	Selitys
Apache	WWW-palvelinohjelmisto
ASCII	Yleisesti käytetty merkistö, jossa on 256 erilaista merkkiä (American Standard Code for Information Interchange)
AT-komennot	tunnettu myös nimellä Hayes-komennot, käytetään modeemien ja GSM-modeemien hallinnointiin (ATtention)
Cgi-ohjelma	Ohjelma, jonka www-palvelin suorittaa
HTTP(S)	HyperText Transfer Protocol (over Ssl)- Tiedonsiirtoon, etenkin HTML-dokumenttien, käytetty protokolla
Javascript	Skriptikieli, jolla voidaan tukea www-sivujen toiminnallisuutta, skriptien suoritus tapahtuu asiakaskoneella (yleensä www-selaimessa)
Kuvauskieli	Kieli, jonka avulla määritellään dokumenttien ja esimerkiksi web-käyttöliittymien ulkoasu
MySQL	Avoimen lähdekoodin tietokantamoottori
PHP	Hypertext PreProcessor - www-ohjelmointiin käytetty ohjelmointikieli, jonka tulkinta ja suoritus tapahtuu www-palvelimella
Python	Korkean tason ohjelmointikieli, jolla voidaan myös tuottaa www-palveluita
SME	Tekstiviestikeskusten rajapintoja hyödyntävä sovellus (Short Message Entity)
SMS	Short Message Service - Tekstiviesti
SMSC	Tekstiviestikeskus (SMS Center)

WWW	World Wide Web - Maailmanlaajuinen tietoverkko, joka sisältää lukemattoman määrän dokumentteja, jotka sijaitsevat www-palvelimilla. Dokumentit siirretään palvelimilta asiakkaille nähtäviksi HTTP-protokollaa käyttämällä
XHTML	HTML versiosta 4 edelleenkehitetty ja tarkemmin määritelty kuvauskieli, jonka avulla määritellään tiedon esittämismuoto www-selaimessa

1 JOHDANTO

Jokainen käyttämätön toimenpideaika aiheuttaa sairaaloille ja yhteiskunnalle huomattavia lisäkustannuksia. Päijät-Hämeen keskussairaalan päiväkirurgian osastolla noin joka kolmas asiakas haluaa siirtää hänelle annettua toimenpideaikaa ja sama trendi on nähtävissä myös muilla osastoilla ja sairaaloissa. Korkeiden kustannusten lisäksi toimenpideaikojen tehoton hyödyntäminen johtaa siihen, että toimenpidejonot eivät lyhene, vaikka kapasiteettia niiden lyhentämiseen olisi.

Quickclic Finland Oy on toiminut terveydenhuollon asiakaspalvelun tehostamisessa jo vuodesta 2003. Ensimmäiset palvelut liittyivät hammashuollon ajanvarauksen tehostamiseen. Quickclic Finland Oy on pieni yritys, jossa omistajan lisäksi työskenteli 4 henkilöä: 1 myyjänä ja innovaattorina sekä 3 kehittäjänä. Vuonna 2006 lokakuussa TietoEnator Oyj osti Quickclic Finland Oy:n terveydenhuollon liiketoiminnan sekä henkilökunnan ja muodosti siitä oman yksikön Healthcare & Welfare -divisioonaan.

Hoitoprosessi päiväkirurgiassa ja sairaaloissa eroaa hammashuollon prosessista niin paljon, ettei hammashuollon käyttöön luotuja työkaluja voida käyttää suoraan. Asiakashallintaa tulee tehostaa jo olemassaolevan prosessin pohjalta kokonaisprosessia tehostamalla ja tarvittaessa muuttamalla. Jononhallintaa alettiin tehostaa sosiaali- ja terveysministeriön tuella yhteistyössä Päijät-Hämeen keskussairaalan sekä Teknillisen Korkeakoulun kanssa; Quickclic Finland Oy tuotti palvelun PHKS:n kanssa ja Teknillisen Korkeakoulun tutkija selvitti, miten se vaikutti sairaalan toimintaan.

Projektin tavoitteena on nostaa leikkaussalien käyttöastetta ja samalla säästää jononhoitajan aikaa tuottavaan työhön. Prosessin tueksi kehitetään palvelu, jonka avulla pyritään muuttamaan kiirekutsumenettelyä. Quickclic Finland Oy:n aikaisempien kokemusten perusteella tekstiviestipalveluiden avulla voidaan tehostaa asiakkuudenhallintaa, joten ongelmaa lähestytään ensin tekstiviesti-palvelun kautta. Palvelun käyttöönotto vaatii myös muutoksia nykyiseen toimintamalliin, joten asiakkailta tiedustellaan halukkuutta olla tekstiviesti-palvelun piirissä jo poliklinikalla asiakasta jonoon lisättäessä.

Aiemmin toteutetut ja hyväksi havaitut palvelut on toteutettu www-sovelluksina, ja vastaava ratkaisu soveltuu myös tähän tilanteeseen sen ylläpidettävyyden ja helpon asennettavuuden vuoksi. Käytössä on www-palvelin, jolla toimii Apache-web palvelin PHP 4 -tuella sekä MySQL-tietokantamoottori.

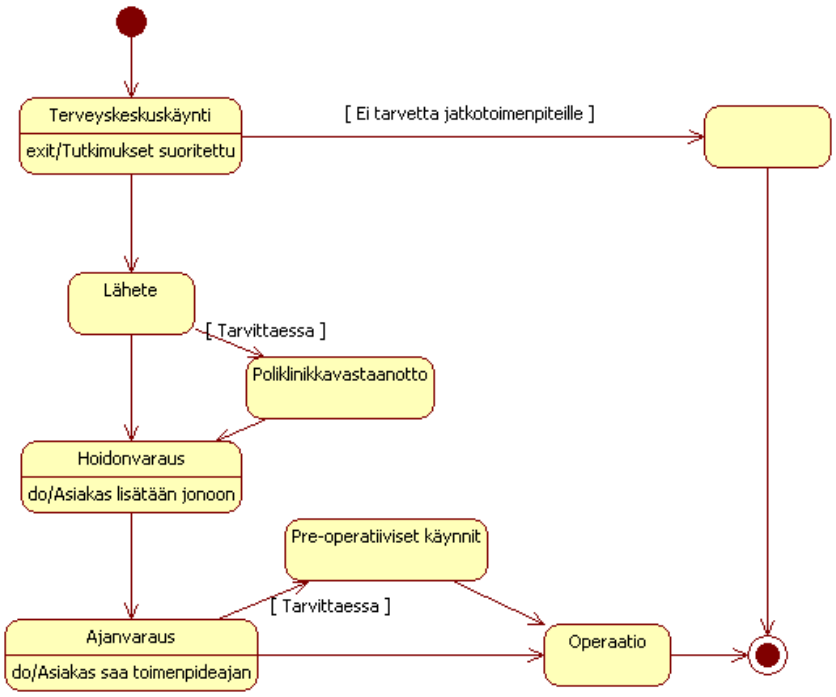
Tämän työn luvussa 2 käsitellään erikoissairaanhoitoa toimintaympäristönä ja kuvataan erikoissairaanhoidon ajanvarausprosessi. Luvussa 3 esitellään palvelun määrittely ja suunnittelu sekä perehdytään erilaisiin tapoihin lähettää ja vastaanottaa tekstiviestejä ohjelmallisesti. Luvussa 4 on esitelty palvelun toteutus ja kuvattu ratkaisut, joihin päädyttiin määrittelyn ja suunnittelun pohjalta. Yhteenveto työstä on luvussa 5.

2 TOIMINTAYMÄRISTÖ

2.1 Erikoissairaanhoidon ajanvarausprosessi

Erikoissairaanhoidossa ajanvarausprosessi alkaa yleensä jo terveyskeskuksessa. Prosessin eteneminen on kuvattu kuviossa 1. Prosessi alkaa, kun asiakas käy lääkärillä, joka toteaa jatkohoidon tarpeen. Usein jatkohoitoon joudutaan jonottamaan, kiireellisyysluokasta riippuen, pitkiäkin aikoja. Tällainen jatkohoito on esimerkiksi kirurginen toimenpide sekä siihen liittyvät laboratorio- ja muut poliklinikkakäynnit. Jatkohoidon tarpeen toteamisen jälkeen lääkäri kirjoittaa asiakkaalle lähetteen jatkohoitoon.

Lähetteen saatuaan asiakkaalle määritellään tarvittavat jatkotutkimukset alkuperäisen tutkimuksen ja diagnoosin perusteella. Kun toimenpide on päätetty, pääsee asiakas toimenpidejonoon. Tarvittaessa asiakas käy vielä polikinikalla vastaanotolla diagnoosin tarkentamiseksi.



KUVIO 1. Ajanvarausprosessi erikoissairaanhoidossa

Toimenpidejonossa olevalle asiakkaalle tehdään varsinainen hoidonvaraus, kun toimenpiteelle voidaan määritellä tarkka aika. Toimenpideajan varauksen yhteydessä varataan myös pre-operatiiviset käynnit, joissa asiakasta valmistellaan toimenpiteeseen ja suoritetaan vaaditut esitutkimukset. Viimeisenä päästään itse toimenpiteeseen, joka suoritetaan sille varattuna ajankohtana.

Kukin asiakkaan ajanvarauksista vaatii sairaalan sisäisiä aikojen ja resurssien varauksia. Mikäli asiakkaan aika perutaan, merkitään ensin ajat vapaiksi ajanvarauskalenteteihin ja peruutetaan laitteiden varaukset. Tämän jälkeen pyritään etsimään uusi asiakas toimenpiteineen vapautuneelle ajalle soittamalla jonossa oleville asiakkaille, joilla käynti poliklinikalla on ajankohtainen. Kun sopiva asiakas löydetään, varataan ajat kalentereista uudelle asiakkaalle ja tehdään tarvittavien resurssien varaukset. Mikäli asiakasta ei löydetä, jäävät resurssit käyttämättä ja sairaalan käyttöaste laskee.

Uuden asiakkaan löytäminen on usein aikaa vaativa prosessi ja sitoo yhden hoitohenkilön puhelimeen. Puhelut ovat usein pitkiä, koska asiakkaan tulee päättää, voiko tulla peruutetulle ajalle ja olla tarvittaessa pois töistä ja niin edelleen. Varsinkin kun peruutus tapahtuu lähellä toimenpiteen ajankohtaa, ei uutta asiakasta useinkaan ehditä löytää. Asiakkaiden löytymistä on pyritty helpottamaan siten, että poliklinikalla asiakkailta tiedustellaan heidän kiinnostustaan lähteä toimenpiteeseen lyhyellä varoitusajalla ja samalla mahdollisesti päästä toimenpiteeseen lyhyemmässä ajassa. Vaikka osa asiakkaista onkin ennalta määritelty “kiirelähtijöiksi”, ei voida aina olettaa, että hekään voivat tulla mille tahansa heille ehdotetulle ajalle. Lähes poikkeuksetta joudutaankin soittamaan useille asiakkaille ennen kuin uusi asiakas on löydetty aikansa peruuttaneen tilalle.

Peruutukset johtavat siis moninkertaiseen työhön ajanvarauksessa. Kun vielä otetaan huomioon se, että peruutuksia on noin 30 % toimenpiteistä, voidaan todeta, että kyseessä on suuri ongelma, joka kuluttaa hoitohenkilökunnan työaikaa huomattavasti tuottavasta työstä. Toukokuussa 2006 tehdyn tilaston mukaan Pohjois-Karjalan keskussairaalassa aikojen uudelleenjärjestelyjä tehtiin 44 prosentissa kaikista ortopedian toimenpiteissä. (Martikainen 2007, 6.)

2.2 Ajanvarausprosessin tukeminen

Ajanvarausprosessia kehittämällä voidaan vastata ihmisten muuttuneeseen elämäntyyliin. Se ei nykyisellään ole modernisoitunut samassa tahdissa kuin ihmisten elämäntyyli on muuttunut kiireellisemmäksi. Tämä puolestaan johtaa usein siihen, että asiakkaiden kalenterit ovat niin täynnä, että hoitotoimenpiteille ei helposti löydy aikaa. Uusia toimintatapoja tulee etsiä ja olemassa olevaa toimintaa tulee kehittää samalla huomioiden ihmisten tottuminen sähköiseen mediaan ja viestintään: internetin käytöstä, sähköpostista, matkapuhelimista ja tekstiviesteistä on tullut osa ihmisten arkea.

Sähköiset yhteydenpitomenetelmät vapauttaisivat henkilökunnan puhelimista suorittavan työn pariin. Lisäksi asiakkaille voitaisiin tarjota pidempi aika vastauksen antamiseen kuin puhelun kesto ja siten mahdollisuus järjestää pääsynsä toimenpiteeseen. Kun viestintä tapahtuu vielä ainakin osin automatisoidusti, kulutetaan mahdollisimman vähän työntekijöiden aikaa uuden asiakkaan löytämiseen. Tekstiviestit soveltuvat hyvin tällaiseen toimintaan. Ihmisillä on useimmin matkapuhelin mukanaan kuin tietokone internet-yhteydellä. Nopea asiakkaan tavoittaminen on olennaista tehokkaassa viestinnässä.

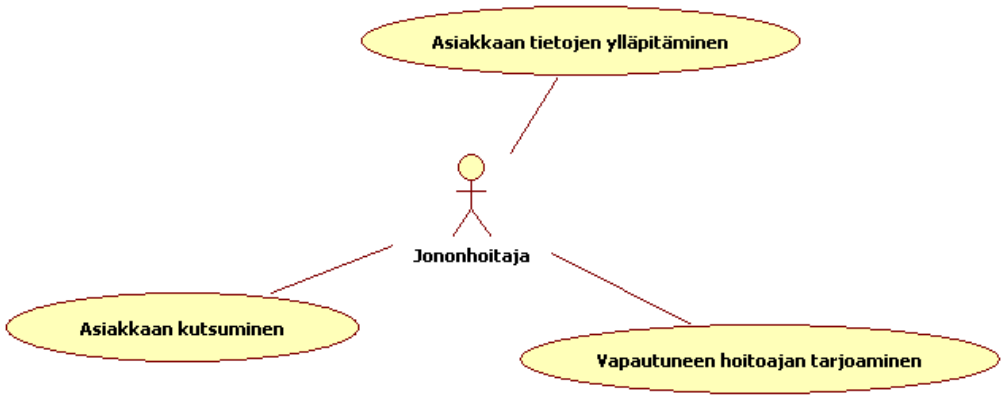
Ajanvarausprosessia pyritään tukemaan sen heikoimmassa kohdassa eli peruutustilanteissa. Tarkemmin sanottuna asiakasrajapinnassa. Tueksi kehitetään palvelu, joka automatisoidusti tiedustelee asiakkailta mahdollisuutta saapua toimenpiteeseen ehdotettuna ajankohtana. Tiedustelu tullaan toteuttamaan tekstiviesteillä, jotka mahdollistavat nopean kaksisuuntaisen viestinnän. Prosessin edetessä pyritään löytämään sopivin tapa lähettää ja vastaanottaa tekstiviestejä ohjelmallisesti.

3 PALVELUN MÄÄRITTELY JA SUUNNITTELU

3.1 Määrittely

Palvelu toteutetaan jononhallinnan työkaluksi, joten tekstiviestien lähetyksen lisäksi palvelussa tulee olla jononhallinnan ominaisuuksia. Palvelun tulee olla mahdollisimman helppokäyttöinen, jotta sillä säästetään aikaa jononhallinnassa. Palvelun suunnittelu aloitettiin valitsemalla kohteet, joilla palvelua kokeillaan. Päijät-Hämeen keskussairaalan päiväkirurgian jononhoitaja valitsi kohteiksi seuraavat toimenpidejonot: olkapää-, käsi-, ascopia- (tähystys), polvi-, hallux- (esimerkiksi vaivaisen luut) sekä jalkajonot.

Seuraavaksi suunniteltiin ne käyttötapaukset, jotka palvelta edellytetään. Kuviot 2, 3 ja 4 ovat käyttötapauskaavioita palvelun toiminnasta.



KUVIO 2. Jononhoitajan käyttötapaukset

Asiakkaan tietojen ylläpitäminen käsittää asiakkaan tietojen syöttämisen, ylläpitämisen ja muokkaamisen. Asiakkaan tiedot käsittävät nimen, matkapuhelinnumeron, sähköpostiosoitteen sekä jononlisäyspäivämäärän. Lisäksi asiakkaat jaetaan palvelussa jonoihin hoidonvarausryhmän mukaisesti.

Vapautuneen hoitoajan tarjoaminen -käyttötapaus tarkoittaa peruuntuneen sen tarjoamista toiselle asiakkaalle tekstiviestillä. Esiehtona käyttötapaukselle on se,

että jononhoitajalla on toimenpideaika sekä asiakas, jolle sitä voi tarjota. Jälkiehtona on se, että tarjousviesti on lähetetty asiakkaalle onnistuneesti.

Asiakkaan kutsuminen käyttötapaus tarkoittaa tilannetta, jossa asiakas on vastannut tarjoukseen myöntävästi ja jononhoitaja päättää varmistaa toimenpideajan hänelle. Esiehtona on se, että asiakas on vastannut tarjoukseen myöntävästi. Jälkiehto puolestaan on se, että tarjousviesti on lähetetty asiakkaalle onnistuneesti ja lisäksi hoitoontulo-ohjeistus on lähetetty asiakkaan sähköpostiin, mikäli se on palvelun tietoihin lisätty.

KUVIO 3. Asiakkaan käyttötapaukset

Asiakkaan tehtävä palvelun toiminnassa on vastaanottaa tarjousviestejä ja vastata niihin. Tarjoukseen vastaamisen esiehtona on, että asiakkaalle on onnistuneesti lähetetty tarjousviesti. Asiakasta ohjeistetaan palvelun käytössä, jotta hän osaa vastata tarjousviestiin oikealla tavalla. Loppuehtona on se, että asiakas on vastannut tarjoukseen palvelun ymmärtämällä tavalla. Mikäli asiakas ei vastaa määräajassa, aiheutuu poikkeus, jollon palvelu ilmoittaa siitä jononhoitajalle.

KUVIO 4. Pääkäyttäjän käyttötapaukset

Listattujen käyttötapausten lisäksi tulee palvelussa olla jononhoitajan, eli käyttäjän, tunnistaminen sekä käyttäjän pääsyn rajoittaminen vain hänelle sallittuihin jonoihin ja niiden asiakkaisiin. Lisäksi palvelulla tulee olla pääkäyttäjä, jonka ainoa käyttötapaus on palvelun ylläpito kuvion 4 mukaisesti.

3.2 Tekstiviestiliikenne

3.2.1 Yleistä

Palvelun vaatimuksiin on määritelty, että sen tulee kyetä lähettämään ja vastaanottamaan tekstiviestejä. Tekstiviestien lähetys ja vastaanotto on toteutettavissa ohjelmallisesti kahdella tavalla: ensinnäkin käyttämällä GSM-modeemia tai matkapuhelinta ja toiseksi hyödyntämällä tekstiviestikeskusta (SMSC), joita esimerkiksi operaattoreilla on.

KUVIO 5. Tekstiviestin kulku matkapuhelimesta toiseen

Normaalisti lähetettäessa tekstiviestiä matkapuhelimesta toiseen voidaan operaattorin ja matkapuhelinten välillä tapahtuva tekstiviestiliikenne jakaa kolmeen vaiheeseen, kuten kuviosta 5 voidaan nähdä. Ensimmäisessä vaiheessa tekstiviesti lähetetään matkapuhelimesta operaattorille ja operaattori vastaa lähetystiedolla. Toisessa vaiheessa operaattori välittää viestin kohde-matkapuhelimeen, joka vastaa operaattorille välitystiedolla. Kolmannessa vaiheessa operaattori antaa tiedon lähettäjälle viestin välityksestä tai sen muusta tilasta.

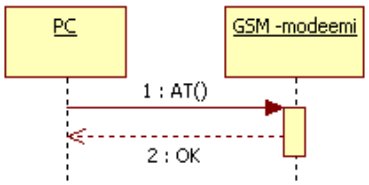
Tekstiviestikeskusta käyettäessä tekstiviesti ohjataan tietokoneelta suoraan tekstiviestikeskukseen, joka rajapinnasta riippuen voi tarjota samat toiminnot kuin GSM-modeemi. Tekstiviestikeskusten kapasiteetti on yleensä huomattavasti suurempi kuin mihin GSM-modeemia tai -puhelinta käyttämällä saavutetaan.

Tekstiviestikeskus hoitaa viestien välityksen GSM-verkkoon ja takaisin. Keskus voi palauttaa lähetys- ja tilatiedot lähettävälle palvelulle. Periaatteellinen ero

GSM-modeemin tai -puhelimen käytössä on se, että lähetys tapahtuu suoralla liitynnällä tekstiviestikeskukseen ilman välittäjää (GSM-modeemia). Käytännössä tämä tarkoittaa sitä, että viestien hallinnointi ei tapahdu AT-komennoilla vaan tekstiviestikeskuksen tuntemaa rajapintaa käyttäen.

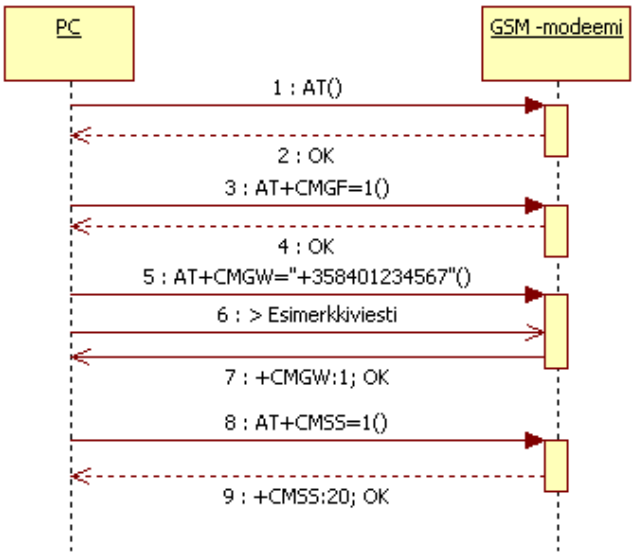
3.2.2 GSM-laitteen käyttö tekstiviestien lähetykseen ja vastaanottoon

Tekstiviestiliikennettä varten voidaan palveluun kytkeä GSM-puhelin tai -modeemi. Tällöin matkapuhelinta voidaan hallinnoida tietokoneelta sarjamuotoista datayhteyttä hyödyntäen. Hallinnointi tapahtuu erityisillä AT-komennoilla, jotka välitetään modeemille ja joihin modeemi tai puhelin vastaa.



KUVIO 6. AT-komentojen käyttö

Yksinkertaisin keskustelu tietokoneen ja GSM-modeemin välillä käsittää yhden AT-komennon (1) ja siihen saatavan vastauksen (2), kuten kuviossa 6 on esitetty.

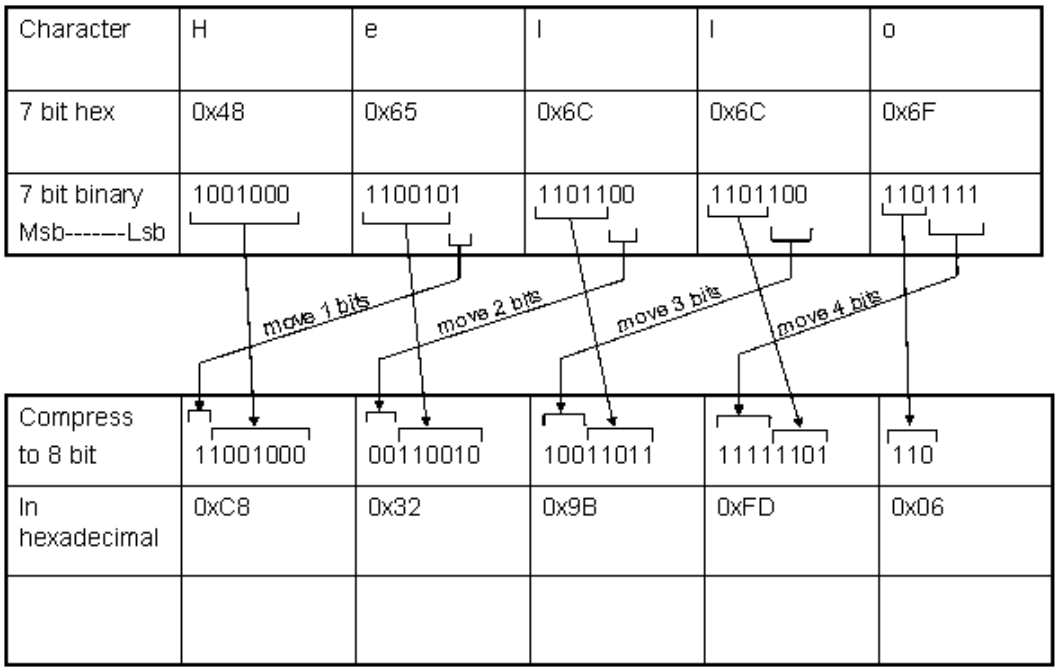


KUVIO 7. Tekstiviestin lähetys AT-komennoilla

Kuviossa 7 esitellään tekstiviestin lähettämiseen vaaditut AT-komennot, sekä niihin GSM-modeemilta saatavat vastaukset. Viestin lähetys aloitetaan tarkastamalla, onko modeemi valmiina AT-komennolla (1). Seuraavaksi asetetaan modeemi teksti -tilaan, jolloin viesti voidaan syöttää normaalina tekstinä käyttämättä PDU-koodausta (3). Seuraavaksi määritellään vastaanottava matkapuhelinnumero (5) ja lähetettävä viesti (6). Viestin kirjoitus lopetetaan ctrl-z-merkkiin. GSM-modeemi tallentaa syötetyn tekstiviestin, luo sille indeksi-numeron, jonka se puolestaan palauttaa PC:lle (7). Viimeisenä annetaan varsinainen lähetyskäsky (8), jossa modeemia käsketään lähettämään viesti, jonka indeksi on 1. Modeemi vastaa jälleen palauttaen lähetetyn viestin tunnisteella (9). Viimeisenä matkapuhelin kuittaa tekstiviestin lähetyksen viestillä OK (AT Command Set For Nokia GSM And WCDMA Products 2005).

Modeemin hallinnointiin ohjelmatasolla ei kuitenkaan yleensä tarvita AT-komentojen laajaa tuntemusta, vaan useita, myös avoimen lähdekoodin, rajapintoja on olemassa. Tällaiset rajapinnat kykenevät yleensä kommunikoimaan erilaisten laitteiden kanssa ja hallitsemaan laitekohtaiset erot. Esimerkiksi kaikki laitteet eivät tue tekstimuotoista viestin syöttöä, vaan viestit tulee luoda PDU-koodauksella. Yleensä tietotekniikassa käytetään 8 bittiä (tai sen moninkertaa)

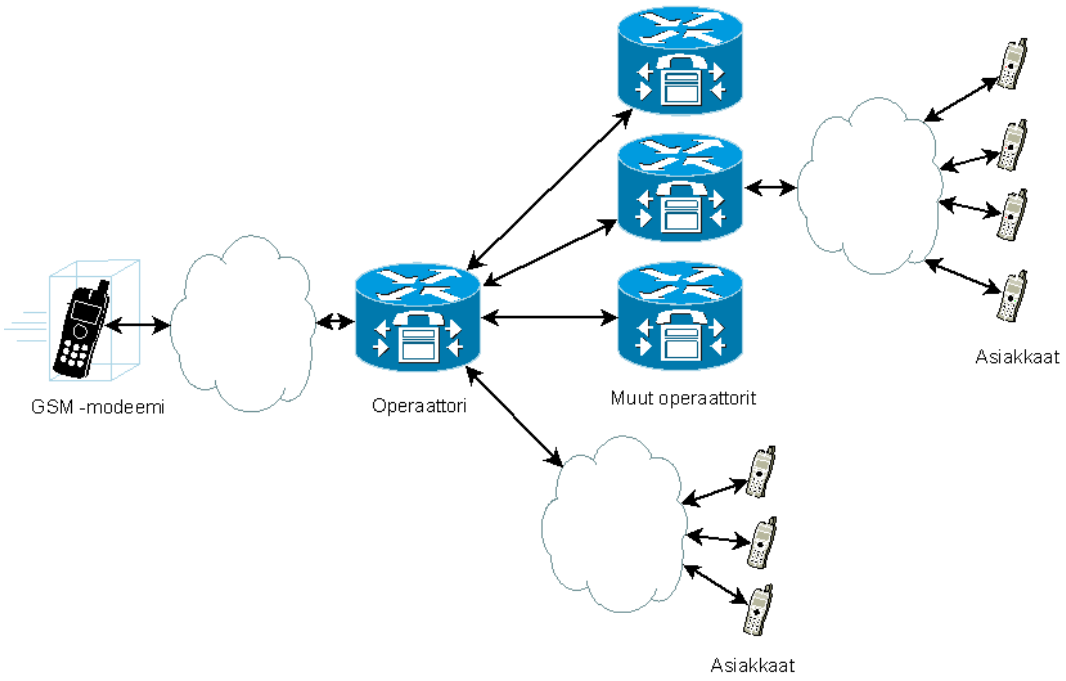
kuvaamaan yhtä kirjainta, mutta tekstiviestissä käytetään 7-bittistä merkistöä. PDU-koodauksella muutetaan 7-bittiset merkit 8-bittisiksi tavuiksi, jotka voidaan suoraan lähettää muokkaamattomana.



KUVIO 8. PDU -koodaus (SMS reading and SMS sending to the Phone from HyperTerminal, 2007)

Kuvio 8 esittää 7-bittisen ASCII-merkkijonon muunnoksen PDU-koodauksella. 7-bittisessä ASCII –merkistössä on määritelty 128 erilaista merkkiä. Tekstiviestien lähetykseen käytetty teknologia puolestaan käsittelee 8-bitin sarjoja tavuina. Koodaus voitaisiin ratkaista myös lisäämällä jokaiseen tavuun 0-bitti alkuun, jolloin tavun arvo pysyisi samana. PDU-koodauksella saavutetaan kuitenkin se etu, että tavuja yhdistelemällä tarvitsee lähettää vain 7 tavua jokaista 8:aa ASCII-merkkiä kohden.

Kuviossa 8 on koodattu merkkijono “Hello”. Koodaus etenee siten, että ensimmäiseen koodatuun tavuun sijoitetaan ensimmäisen merkin bitit vähiten merkitseviksi biteiksi sekä toisen merkin viimeinen bitti eniten merkitseväksi bitiksi. Toiseen tavuun sijoitetaan loput bitit toisesta merkistä vähiten merkitseviksi biteiksi sekä kolmannesta merkistä 2 vähiten merkitsevää bittiä tavun eniten merkitseviksi biteiksi ja niin edelleen.



KUVIO 9. Tekstiviestin kulku käytettäessä GSM -modeemia

Kuviosta 9 voidaan nähdä tekstiviestien kulku, mikäli lähetykseen käytetään GSM-modeemia tai puhelinta. Tekstiviesti lähetetään GSM-modeemilta operaattorille, jonka liittymä modeemiin on tilattu. Operaattori välittää viestin tarvittaessa muille operaattoreille, ja viime kädessä viesti toimitetaan asiakkaille asti.

GSM-modeemin tai puhelimen käyttö tekstiviestien välittämiseen on kustannuksiltaan edullinen vaihtoehto. Aloituskustannukset jäävät pieniksi, koska matkapuhelimia saa kohtuullisen edullisesti samoin kuin matkapuhelinliittymiä. Lisäksi tekstiviestien hinnat nykyisissä matkapuhelinliittymissä ovat kilpailukykyisiä muihin vaihtoehtoihin nähden. Ongelmaksi muodostuu kuitenkin huono läpivienti tekstiviestiliikenteessä; modeemi kykenee yleensä lähettämään tai vastaanottamaan alle kymmenen tekstiviestiä minuutissa (Short Message Service / SMS Tutorial 2006). Lisäksi GSM-liittymään ei voida liittää lyhytnumeroa, eli normaali matkapuhelinnumeroa lyhyempää numeroa, joita yleensä tekstiviestipalveluissa käytetään. Myöskään maksullisia palveluita ei voida luoda tekstiviestiveloitus-perusteisesti.

3.2.3 SMSC-vaihtoehdot

GSM-modeemien ja puhelinten lisäksi on mahdollista kytkeytyä suoraan operaattorin tai muun palveluntarjoajan tekstiviestikeskukseen. Yleisesti tunnettu rajapinta on etenkin Nokian valmistamissa keskuksissa käytetty CIMD2-rajapinta (Computer Interface to Message Distribution). Maailmanlaajuisesti yleisin käytetty protokolla on SmsForum.net:n kehittämä SMPP (Short Message Peer to Peer) (Short Message Service / SMS Tutorial 2006). Lisäksi tekstiviestikeskusten yleisiä liityntärajapintoja ovat UCP/EMI (Universal Computer Protocol/External Machine Interface) sekä OIS (Open Interface Specification), mutta näitä ei käsitellä tässä työssä syvällisemmin. Näiden lisäksi useat palveluntarjoajat tukevat korkeamman tason rajapintoja tekstiviestikeskuksiinsa. Tällaisia ovat esimerkiksi HTTP GET/POST- tai SOAP-rajapinnat (Simple Object Access Protocol).

3.2.4 CIMD2-rajapinta

Operaattorit tarjoavat yleensä tekstiviestejä hyödyntäville palveluille ja sovelluksille CIMD- tai CIMD2-rajapinnan. Se mahdollistaa TCP/IP-yhteyden viestinvälityspalvelimelle, joka välittää viestit sovellukselta gsm-liittymään ja takaisin.

CIMD2 rajapintamääritys kuvaa 3 erilaista liityntäratkaisua (CIMD Interface Specification 2005, 9):

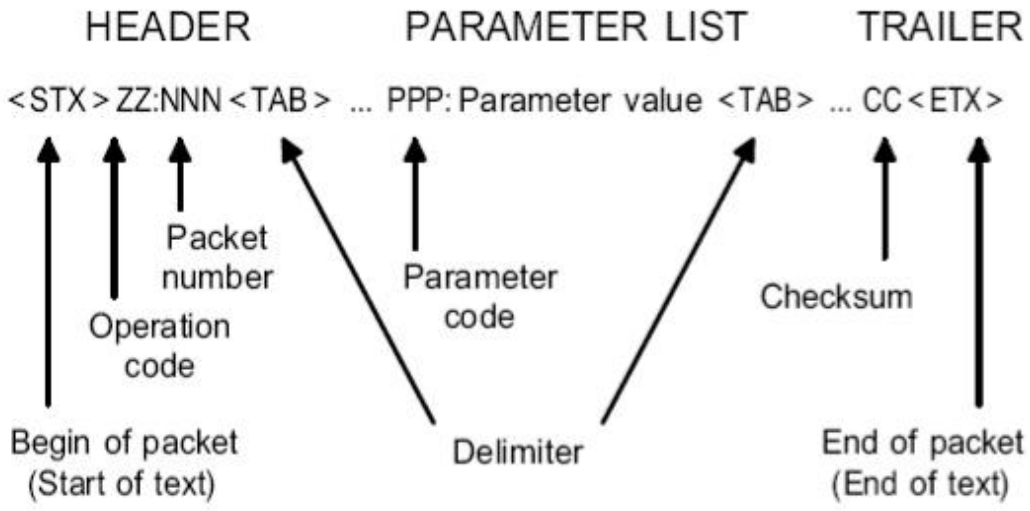
1. Vain viestien lähetykseen tarkoitettu rajapinta

2. Kyselyrajapinta, jossa vastaanottava sovellus käy suorittamassa kyselyjä viestinvälityspalvelimelta

3. Vastaanottorajapinta, jossa vastaanottavan sovelluksen tulee aina olla kuuntelemassa mahdollisia viestejä, jotka viestinvälityspalvelin lähettää palvelimelle.

Rajapinnan kautta välitetään pyyntöjä ASCII-merkistöä käyttäen. ASCII-taulukon ulkopuoliset merkit korvataan merkkijonosarjoilla, esimerkiksi ‘Ä’ korvataan merkkijonolla ‘_A”’ (CIMD Interface Specification 2005, 59 - 65).

Pyynnöillä on aina tietty rakenne; ne alkavat tavulla, jonka desimaaliarvo on 2 (stx - start of text) ja päättyvät tavuun, jonka desimaaliarvo on 3 (etx - end of text). Erottimena otsikkotietojen, parametrien ja päättötietojen välillä käytetään sarkain-merkkiä (desimaaliarvo 9). Parametrikoodien ja niiden arvojen sekä operaatiokoodien ja pakettinumeroiden välissä käytetään kaksoispistettä : (desimaaliarvo 58). Rakennetta havainnollistetaan kuviossa 10.



KUVIO 10. CIMD2-rajapinnan mukaisen pyynnön rakenne (CIMD Interface Specification 2005)

CIMD2-rajapinnan pyynnöt sisältävät aloitustavun jälkeen otsikkotiedot. Näitä ovat operaatiokoodi ja paketin numero. Operaatiokoodeja ovat esimerkiksi 01 - kirjautuminen - ja 03 - viestin välityspyyntö (CIMD Interface Specification 2005, 12). Pakettien numerointi tapahtuu siten, että palvelun tekstiviestikeskukselle lähettämät viestit numeroidaan aina parittomilla numeroilla välillä 0 - 255. Kun paketin numero 255 saavutetaan, otetaan käyttöön pakettinumero 01. Vastaavasti tekstiviestikeskuksen lähettämät paketit numeroidaan parillisilla numeroilla, aloittaen paketista numero 00. Numeroinnilla on tarkoitus estää saman paketin tulkinta moneen kertaan.

Otsikkotietojen jälkeen seuraavat viestin parametrit. Esimerkiksi kirjautumis-paketissa välitetään parametrit 010 - käyttäjän tunnus ja 011 - salasana. Kullakin viestityypille on määritelty pakolliset ja valinnaiset parametrit, joita paketti voi sisältää.

Parametrien jälkeen pyynnössä on lopputiedot. Näihin tietoihin luetaan tarkistussumma sekä viestin lopetusmerkki. Kussakin CIMD-protokollan viestissä voi olla tarkistussumma, jonka avulla voidaan tarkistaa, että viesti on vastaanotettu samanlaisena kuin se on lähetetty. Tarkistussumman käyttö ei ole pakollista, mutta Nokian valmistamat tekstiviestikeskukset käyttävät sitä aina lähettämissään viesteissä.

Tarkistussumman laskeminen on suoraviivaista ja yksinkertaista. Se tapatuu siten, että viestin kaikki tavut käydään yksi kerrallaan läpi, ja merkkiä vastaava numeroarvo lisätään tarkistussummaan. Jokaisen summauksen jälkeen summasta huomioidaan vain vähiten merkitsevä tavu. Tarkistussumman laskemiseen ei käytetä viestistä kahta viimeistä tavua eli itse tarkistussummaa eikä viestin lopetusmerkkiä. Tarkistesumma merkitään viestiin sen arvoa vastaavalla kaksinumeroisella heksaluvulla. Esimerkki tarkistesumman laskevasta ohjelmakoodista on liitteessä 1.

Tekstiviestin lähetykseen käytettävässä pyynnössä käytetään operaatiokoodia 03. Parametreina pyynnössä ovat vastaanottajan puhelinnumero (021) sekä viestin sisältö(033). Esimerkki viestin lähettämiseen käytetystä pyynnöstä on taulukossa 1.

TAULUKKO 1. Esimerkki CIMD -protokollan mukaisesta viestistä

ASCII	HEX	Selite
<stx>	02	Viestin aloittava merkki
03	3033	Viestin tyyppi; 03 = submit message
:	3a	Eroitinmerkki
001	3030 31	Viestin sekvenssinumero
<tab>	09	Eroitinmerkki (sarkain)
021	3032 31	Parametrin tunniste - kohdeosoite
:	3a	Eroitinmerkki
0401234567	3034 3031 3233 3435 3637	Parametrin arvo - vastaanottava puhelinnumero
<tab>	09	Eroitinmerkki
033	3033 33	Parametrin tunniste - viestin sisältö
:	3a	Eroitinmerkki
Contents of the message	436f 6e74 656e 7473 206f 6630 7468 6520 6d65 7373 6167 65	Parametrin arvo - viestin sisältö
<tab>	09	Eroitinmerkki
91	3931	Tarkistesumman arvo (heksaluku 91, kymmenjärj 145)
<etx>	03	Viestin lopetusmerkki

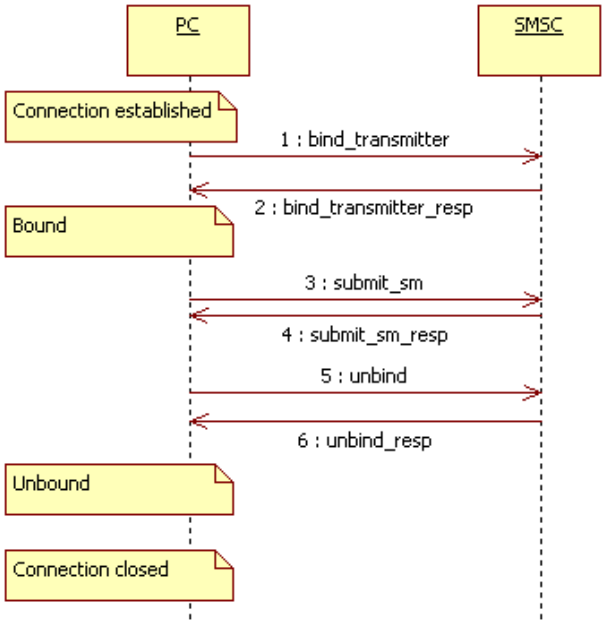
Tällaiseen pyyntöön tekstiviestikeskus vastaa viestillä, jonka operaatiokoodi on 53. Viestin parametrit ilmaisevat, onnistuiko lähetys vai ei. Onnistuneen viestinlähetyksen jälkeen saadaan vastaus, jonka parametreina ovat vastaanottajan puhelinnumero (021) sekä tekstiviestikeskuksen aikaleima (060, muotoa yymmddhhmmss). Viesti voidaan myöhemmin yksilöidä aikaleiman ja sekvenssinumeron avulla. Epäonnistuneen viestin lähetyksen jälkeen saadaan viesti, jossa on parametrit 900 (virhekoodi) sekä valinnainen parametri 901 (virheen selite).

3.2.5 SMPP-rajapinta

SMS Forum on voittoa tuottamaton järjestö, jonka tavoite on edistää tekstiviestien käyttöä langattomien tekngologioiden yhteydessä. Se on tuottanut SMPP-rajapinnan (Short Message Peer to Peer) tekstiviestien ja niihin liittyvän tiedon siirtämiseen tekstiviestikeskusten, välityspalvelinten ja tekstiviestipäätteiden välillä. SMPP tarjoaa joustavan rajapinnan, joka toisin kuin esimerkiksi CIMD, mahdollistaa asynkronisen viestinnän tekstiviestikeskuksen kanssa.

Kuten CIMD-protokolla, SMPP käyttää TCP/IP-protokollaa viestien välitykseen. Standardoitu portti, jota SMPP käyttää, on 2775, mutta myös muita vapaita portteja voidaan käyttää. Lisäksi SMPP:n protokollassa on yhteneväisyyksiä CIMD-protokollaan sen tarjoamissa liityntäratkaisuissa, eli rajapintaa voidaan käyttää vain viestien lähetystä käyttävässä ratkaisussa, vain viestejä vastaan-ottavassa ratkaisussa tai edellisten yhdistelmässä.

SMPP-protokolla on määritelty siten, että kommunikointi tekstiviestikeskuksen kanssa tapahtuu istuntopohjaisesti. Kuviossa 11 on esitelty yksinkertaisen istunnon kulku.



KUVIO 11. SMPP-istunnon kulku tekstiviestin lähetyksessä

Istunnon kulku tapahtuu siten, että yhteyden muodostamisen jälkeen tekstiviesti-keskukselle lähetetään bind_transmitter-viesti, jolla viestin lähettäjä tunnistautuu, ja samalla istunnon tila vaihtuu ”bound”-tilaan. Seuraavaksi tapahtuvat tekstiviestioperaatiot ja viimeiseksi istunto lopetetaan unbind-viestillä sekä yhteys katkaistaan (Short Message Peer-to-Peer Protocol Specification 2003, 34).

SMPP-protokollan yhteydessä pyynnöistä käytetään nimitystä PDU (Protocol Data Unit). Kukin PDU koostuu vähintään 16 tavusta, jotka käsittävät otsikko-

tiedot (Short Message Peer-to-Peer Protocol Specification 2003 ,53). Otsikkotiedoille on aina varattu 16 tavua PDU:n tyypistä riippumatta. PDU rungon määrää sen tyyppi. Esimerkiksi viestejä lähettävän palvelun istunnon avaaminen (bind_transmitter, tunnistekoodi 2) määrittelee seuraavat parametrit pakollisina:

- system_id, kirjautuvan järjestelmän tunniste
- password, kirjautuvan järjestelmän salasana
- system_type, järjestelmän tyyppi
- interface_version, käytettävän SMPP-protokollan versio
- addr_ton, lähettävän numeron tyyppi; 6 dec vastaa lyhytnumeroa
- addr_npi, lähettäjän osoitteen tyyppi; 14 dec tarkoittaa ip-osoitetta
- address_range, lähettävän palvelun osoite (numero).

SMPP versiosta 3.4 lähtien parametrit voivat olla joitakin neljästä eri tyypistä: kokonaisluku, c-tyylinen merkkijono (merkkijono, joka päättyy aina tavuun, jonka arvo on 0), merkkijono tai TLV (Tagged Level Value), joka muodostuu kolmesta osasta (1. parametrin tyyppi, 2. parametrin pituus sekä 3. parametrin arvo).

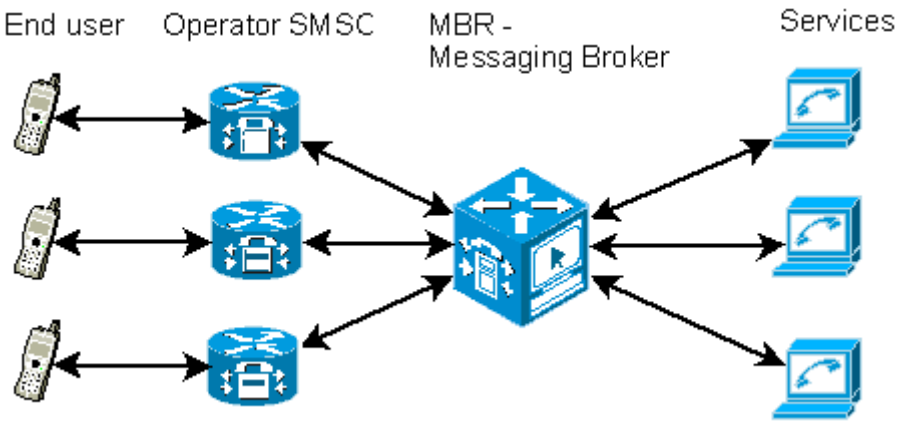
TAULUKKO 2. Esimerkki SMPP -protokollan mukaisesta PDU:sta

ASCII	**HEX**	**Selite**
??	0000 002b	command_length - viestin pituus (47 tavua)
??	0000 0002	message_type - Viestin tyyppi; 02 = bind_transmitter
??	0000 0000	command_status - bind_transmitter viestissä on aina 0
??	0000 0001	sequece_number - Viestin sekvenssinumero
System1	5379 7374 656d 3100	system_id - Lähettävän järjestelmän tunniste
Password	7061 7373 776f 7264 00	password - Lähettävän järjestelmän salasana
?	00	system_type - Lähettävän järjestelmän tyyppi
P	50	interface_version - Käytettävän rajapinnan versionumero
?	06	addr_ton - Lähettävän numeron tyyppi (06 - lyhytnumero)
?	0e	addr_npi - Lähettävän palvelun osoite (14 dec - IP-osoite)
18444	3138 3434 3400	address_range - Lähettävän palvelun numero, merkkijono päättyy 00 -merkkiin (NULL -merkki)

Bind_transmitter PDU:ssa ei ole yhtään vaihtoehtoista parametria, mutta jotkin parametrit, esimerkiksi addr_ton, voidaan asettaa arvoon 0, jollon käytetään oletusarvoja. Esimerkki bind_receiver PDU:sta on esitetty taulukossa 2.

3.2.6 Monioperaattorirajapinta (MBR)

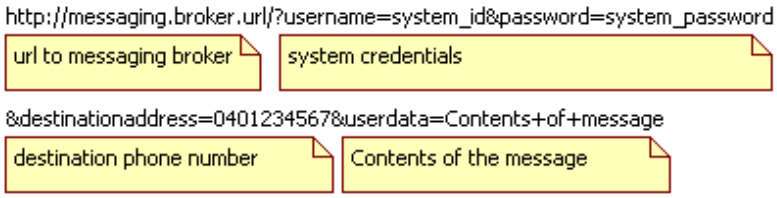
MBR-rajapinta on Elisa Oyj:n tarjoama rajapinta tekstiviestien lähettämiseen ja vastaanottamiseen. Se on huomattavasti yksinkertaisempi kuin CIMD2-rajapinta. Sitä käytetään standardi HTTP-pyyntöjen avulla (Metropolis Development Guide4, - SMS Messaging Broker 2004, 3). Rajapinnan yksinkertaisuus rajaa käyttömahdollisuuksia, mutta tarjoaa moneen ongelmaan helpon ratkaisun. Esimerkiksi MBR-rajapintaan kuuluvat operaattoreiden väliset Roaming-sopimukset, jolloin lyhytsanomanumeroita, eli normaaleja puhelinnumeroita lyhyempiä palvelunumeroita, voidaan käyttää palveluissa ja kuitenkin saavuttaa eri operaattoreiden verkossa toimivat asikkaat, kuten kuvasta 12 voidaan nähdä. Vastaavasti CIMD2-protokollaa käyttäen tulisi kytkeytyä kuhunkin operaattoriin erikseen. Myöskään erikoismerkeistä ei tarvitse huolehtia enempää kuin, että käyttää latin-1 yhteensopivaa merkistöä (esimerkiksi ISO-8859-1 tai ISO-8859-15).



KUVIO 12. MBR-viestinvälitysratkaisun periaatekuva

Viestin lähetys tapahtuu muodostamalla HTTP GET -pyyntö ja lähettämällä se operaattorin MBR palvelimelle kuvion 13 mukaisesti. Minimivaatimuksena on se,

että pyynnössä on asiakkaan tunnistetiedot “username” sekä “password” -atribuuteissa ja lisäksi “userdata”, johon itse viesti sisällytetään, ja “destinationaddress”, joka on vastaanottavan GSM-liittymän numero.



KUVIO 13. Esimerkki MBR-rajapinnan mukaisesta HTTP-pyynnöstä

Viestin vastaanotto tapahtuu siten, että operaattorin MBR-palvelin muodostaa HTTP GET -pyynnön, jossa on mukana parametrit “userdata”, asiakkaan kirjoittama tekstiviesti, “originatingaddress”, asiakkaan puhelinnumero, sekä muita tietoja, esimerkiksi asiakkaan operaattorin koodi ja numero, johon viesti on lähetetty.

MBR-viestinvälitysratkaisussa palvelut ovat suoraan yhteydessä viestinvälitys-palveluun, joka välittää viestit operaattoreiden tekstiviestikeskuksille. MBR-palveluun voidaan liittää uusia palveluita tilaamalla niitä Elisa Oyj:ltä; viestien eroittelu eri palveluille voi tapahtua asiakkaan numeron, palvelun lyhytsanoma-numeron, viestin ensimmäisen sanan, operaattorin tai näiden yhdistelmien mukaan.

4 TOTEUTUS

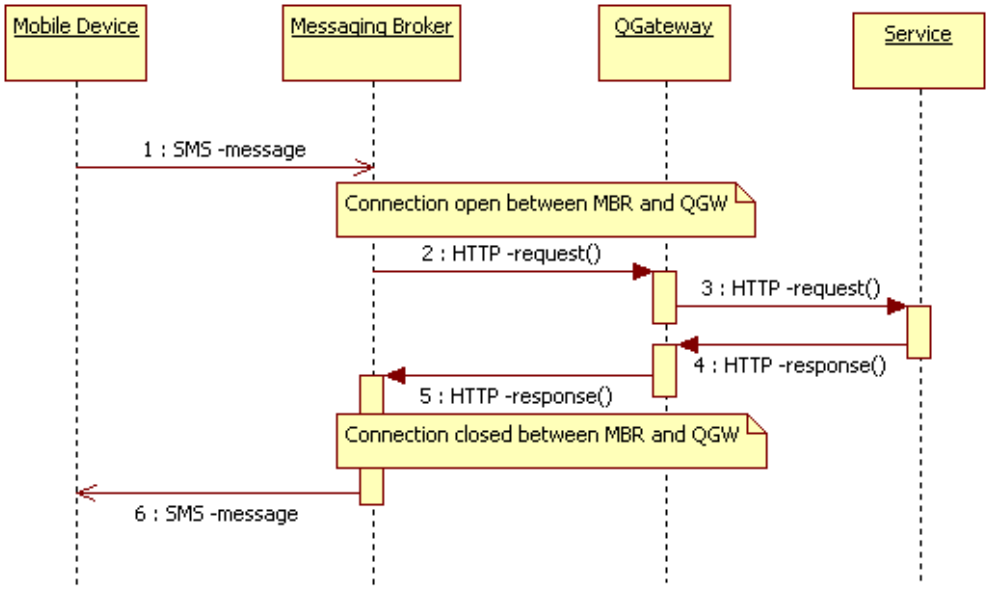
4.1 Palvelun toteutuksessa käytetyt tekniikat

4.1.1 Tekstiviestirajapinnan valinta

Monioperaattorimallin mukainen rajapinta on selkeästi helpommin toteutettavissa kuin CIMD2- tai SMPP-rajapinnat. Kun lisäksi huomioidaan se, että tekstiviestien ohjaaminen eri operaattoreiden verkossa tulisi hoitaa operaattorikohtaisesti SMPP- tai CIMD2-rajapintaa käyttäen, tulee MBR-rajapinnan käyttöönotto halvemmaksi. Toisaalta tekstiviestikohtaiset kulut ovat MBR-rajapinnalla suuremmat.

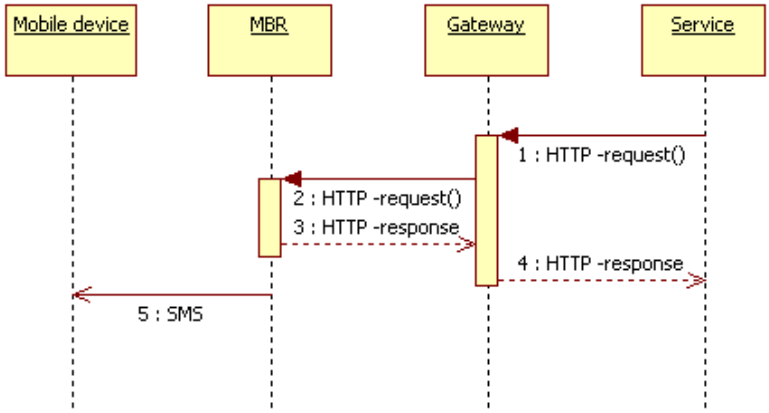
Käytössä olevista vaihtoehdoista päädyttiin monioperaattorimallin mukaiseen ratkaisuun sillä varauksella, että mikäli tekstiviestien määrä palveluissa kasvaa huomattavan suureksi, harkitaan CIMD2- tai SMPP-rajapinnan käyttöönottoa. Palvelun aiheuttama viestimäärä kuukaudessa on noin 50 kpl, joten viestiliikenteestä ei odoteta suuria kuluja.

Käyttöön valittiin siis MBR-rajapinta, sen nopean käyttöönoton ja yksinkertaisen toteutuksen vuoksi. Lisäksi www-palvelua luotaessa oli käytössä www-palvelin, jota voitiin myös käyttää tekstiviestien vastaanottamiseen. Viestien ohjaamiseen palveluiden ja operaattorin tekstiviestikeskuksen välillä luotiin oma palvelunsa, QGateway, jonka avulla palveluiden tunnistesanoja voitiin hallinnoida.



KUVIO 14. Viestin kulku toteutetussa ratkaisussa

QGateway-palvelusta päätettiin tehdä palvelun kannalta läpinäkyvä MBR-rajapinnan suuntaan, eli MBR-palvelun lähettämät HTTP-pyynnöt ohjattiin sellaisenaan palvelulle, jolle viesti oli tarkoitettu. Vastaavasti palveluiden muodostamat pyynnöt ohjattiin sellaisenaan MBR:lle. Poikkeuksen muodostavat kuitenkin palvelun tunnistamistiedot: kuten MBR-palvelukin, myös QGateway vaatii palveluiden tunnistamaan itsensä viestinvälitysvaiheessa. QGateway käyttää vastaavasti omia tunnistautumistietojaan lähetettäessa viestiä edelleen MBR:lle. Kuviossa 14 on esitetty viestien kulku asiakaslähtöisessä viestinnässä (pull-viestin lähettäminen). Kun asiakas lähettää viestin, ohjaa Messaging Broker sen, sille asetettujen sääntöjen mukaisesti, vastaanottavalle palvelimelle, tässä tapauksessa QGatewaylle. QGateway puolestaan ohjaa pyynnön oikealle palvelulle, joka vastaa pyyntöön HTTP-rajapinnan mukaisella vastauksella, jonka rungossa on asiakkaalle välitettävä vastaus. QGateway välittää saamansa vastauksen Messaging Brokerille.



KUVIO 15. Tekstiviestin lähettäminen palvelulähtöisesti

Kuviossa 15 on kuvattu palvelulähtöisen tekstiviestin lähetys (push-viesti). Kun palvelu lähettää viestin, se muodostaa HTTP-pyynnön ja lähettää sen QGatewaylle. QGateway puolestaan välittää pyynnön viestinvälityspalvelimelle, jonka vastauksen QGateway palauttaa palvelulle. Viestinvälityspalvelin välittää lähetettävän tekstiviestin operaattorin tekstiviestikeskukseen, jonka kautta viesti päätyy asiakkaan puhelimeen.

4.1.2 Viestien ohjaaminen

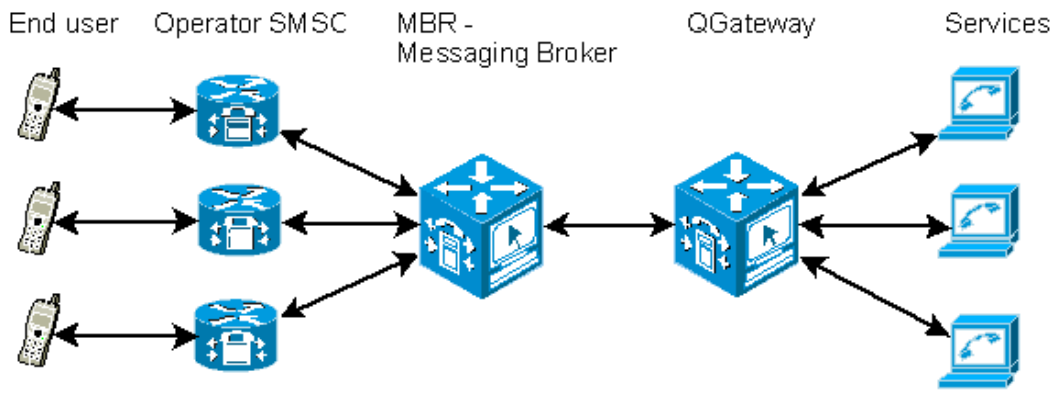
Koska käytössä oli vain yksi lyhytsanomanumero, jonka kautta käytettiin useita palveluita, piti viestit tunnistaa jollakin tavalla ja ohjata oikealle palvelulle. Alalla vallitseva de facto -standardi oli viestin ensimmäisen sanan perusteella tapahtuva tunnistaminen.

Viestejä vastaanottamaan luotiin erityinen palvelu, jonka tehtävänä oli jakaa lyhytnumeroon tulleet viestit eri palveluille. Lisäksi palveluiden piti kyetä lähettämään viestejä, mutta toisaalta ulkopuolisten pääsy palveluun tuli rajata.

Ensimmäisessä versiossa viesti tunnistettiin ensimmäisen sanan, eli palvelun tunnistesanan perusteella. Palvelu etsi vastaanotetusta viestistä ensimmäisen välilyönnin ja käsitteli sitä edeltävää merkkijonoa palvelun tunnistesanana. Käytäntö kuitenkin osoitti, että tekstiviestien kirjoittaminen tai ohjeiden

ymmärtäminen oli joillekin asiakkaille vaikeata ja tekstiviestien alussa saattoi olla välilyöntejä jo ennen ensimmäistä varsinaista sanaa.

Viestin vastaanottavaa palvelua kehitettiin siten, että se osasi ohittaa viestin alussa olevat blanko-merkit (välilyönnit, sarkaimet sekä rivinvaihdot) ja etsiä ensimmäisen sanan, joka oli edelleen eroitettu seuraavasta sanasta blanko-merkillä. Jälleen todettiin, että viestien välityksessä oli ongelmia, kun asiakkat saattoivat erottaa ensimmäisen sanan pisteellä, pilkulla tai muulla välimerkillä. Tämän jälkeen erotusmerkiksi otettiin regular expressionissa käytetty ”non word character” eli paikallisasetuksella ”fi” (suomi) kaikki merkit, jotka eivät ole numeroita (0-9), kirjaimia (a-ö tai A-Ö) eivätkä alaviiva (_), luetaan sanoja erottavaksi merkiksi.



KUVIO 16. Palvelussa käytetty viestinvälitysratkaisu

Kuviossa 16 nähdään luotu tekstiviestien välityksen ratkaisu. Vaikka MBR-palvelu olisi tarjonnut viestien ohjaamisen, päätettiin kuitenkin luoda oma viestinvälityspalvelin, jotta palveluiden hallinnointi olisi helpompaa ja se voitaisiin tehdä keskitetysti riippumatta ulkopuolisista toimijoista. Lisäksi asiakkaiden laskutustiedot tekstiviestien osalta olisivat kaikki yhdessä paikassa, josta ne voidaan lukea laskutusperusteiksi.

4.2 Käyttöliittymien luominen

Koska palvelu toteutettiin web-tekniikoita käyttäen, oli luonnollista valita myös käyttöliittymien luomiseen yleisesti käytettyjä web-tekniikoita. Suurimmat

ongelmat käyttöliittymien kannalta olivat eri selainten tavat käsitellä standardeja tekniikota eri tavoin (Ishida, 2005).

Vaihtoehtoina kuvauskieliksi olivat HTML ja XHTML, joista jälkimmäinen on huomattavasti tarkemmin määritelty. HTML:stä harkittiin versiota 4 sekä XHTML:stä versiota 1 ja Transitional dokumenttityyppiä. XHTML on hieman työläämpi toteuttaa kuin HTML, koska se on tarkemmin määritelty, mutta toisaalta tarkempi määrittely vähentää selainten erilaista käyttäytymistä. Lisäksi monet mobiililaitteiden selaimet tukevat XHTML-standardia mutta eivät perinteistä HTML:ää.

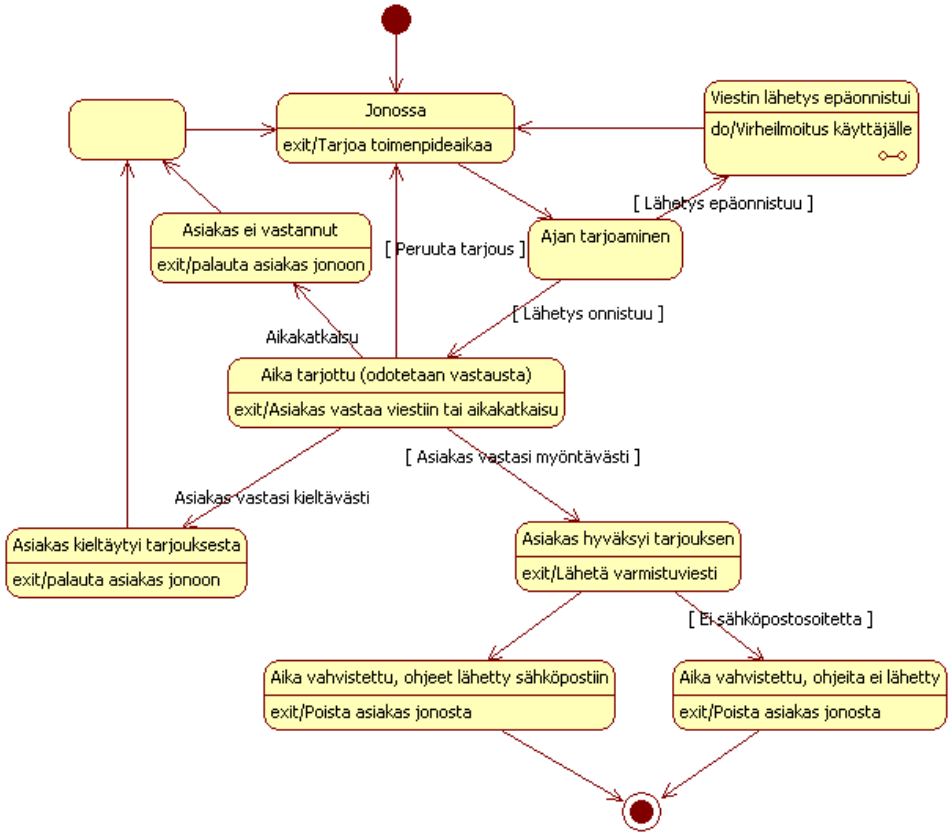
CSS:n avulla voidaan eriyttää käyttöliittymien osien esitystapa varsinaisen dokumentin rakenteesta ja siten selkeyttää (X)HTML-koodia. Palvelua tehtäessä CSS:stä oli tulossa versio 2, mutta se ei ollut vielä loppuun asti määritelty eikä suurin osa www-selaimista tukenut sitä. Tästä syystä palvelussa päädyttiin käyttämään CSS versiota 1. Käyttöliittymät pyrittiin suunnittelemaan siten, että mahdollisimman suuri osa muotoiluista kirjoitettaisiin CSS-tiedostoihin, ja (X)HMTL-koodin tehtäväksi jäisi käyttöliittymien rakenteen esittäminen.

Käyttöliittymien käytettävyyden tueksi päätettiin ottaa jokin skriptikieli. Ihannetapauksessa skriptikieltä käyttämällä voitaisiin vähentää palvelimen ja selaimen välistä tiedonsiirtoa ja lisäksi käyttöliittymien käytettävyyttä voitaisiin parantaa. Vaihtoehdot skriptikieliksi olivat VBScript sekä JavaScript.

VBScript:n avulla voidaan hyödyntää tehokkaasti Internet Explorer -selaimen ominaisuuksia, mutta vastaavasti muut selaimet eivät osaa hyödyntää sitä. Toisaalta palvelun käyttöliittymien suunnittelussa pyrittiin selainriippumattomuuteen, joten JavaScript oli valintana johdonmukaisempi. Skriptien kirjoittamisessa tuli ottaa huomioon myös eri selainten poikkeavuudet JavaScriptin tulkinnassa, joten skriptien määrä pyrittiin pitämään mahdollisimman vähäisenä.

4.3 Asiakkaan tilojen hallinta

Palvelussa oleville asiakkaille on määritelty erilaisia tiloja. Kun asiakas lisätään palveluun, asetetaan tilaksi ”Jonossa”. Asiakkaan tila vaihtuu palvelussa käyttäjän ja asiakkaan toimien perusteella.



KUVIO 17. Tilakaavio asiakkaan tiloista palvelussa

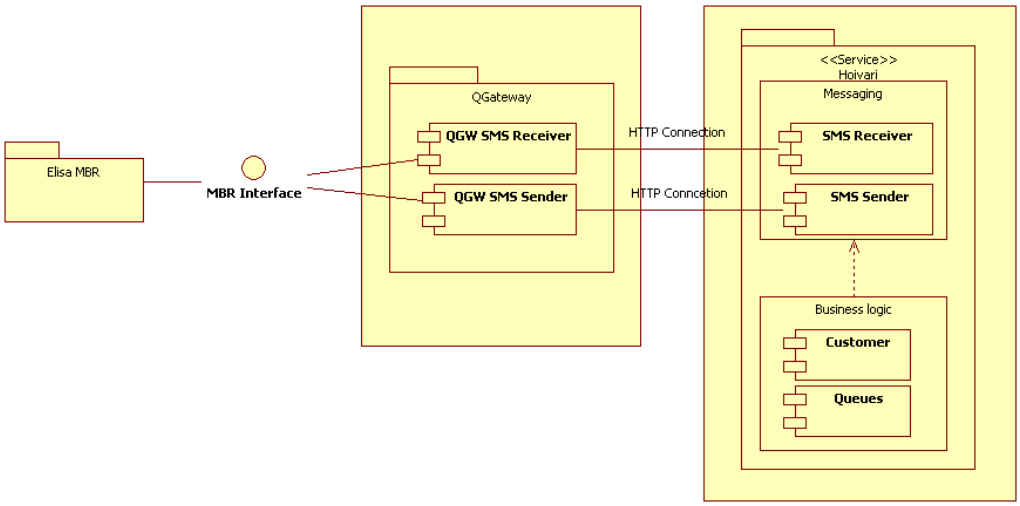
Jonossa on asiakkaan perustila, johon asiakkaat palautetaan kunnes toimenpideaika saadaan vahvistettua asiakkaalle. Kuviossa 17 nähdään asiakkaiden eri tilat. Vaikka asiakkaat järjestetään jonossa jonoonlisäys-päivämäärän mukaan, jonon ensimmäisenä näytetään kuitenkin ne asiakkaat, jotka ovat muussa tilassa kuin "jonossa", jotta "aktiivisten" asiakkaiden seuraaminen olisi helpompaa.

4.4 3+1 näkymää palvelun suunnittelussa

Ohjelmistojen laajuuden kasvaessa suunnittelun tärkeys korostuu jatkuvasti. Suunnittelun eri osa-alueet painottuvat riippuen muun muassa sovelluksen luonteesta, sen laajuudesta sekä valitusta toteutusteknologiasta (Aalto 1999, 1). Nokiassa laajasti käytetty suunnittelumenetelmä OMT ++ on suunniteltu siten, että se vastaa ainakin seuraaviin avainkysymyksiin: mistä loogisista osista järjestelmä muodostuu, millainen on järjestelmän ajonaikainen konfiguraatio, miten järjestelmä jaetaan kehitettäviin ohjelmistokomponentteihin, mitä riippuvuuksia ja rajapintoja on komponenttien välillä ja miten arkkitehtuuri toteuttaa tärkeimmät käyttötapaukset (Aalto 1999, 1). 3+1 näkymän malli on kehitetty Philippe Kruchtenin tunnetuksi tekemän "4 + 1” -näkymän pohjalta. Suurimmat erot näiden välillä ovat UML:n käyttö kuvauskielenä sekä näkymien sovittaminen OMT++ -menetelmään.

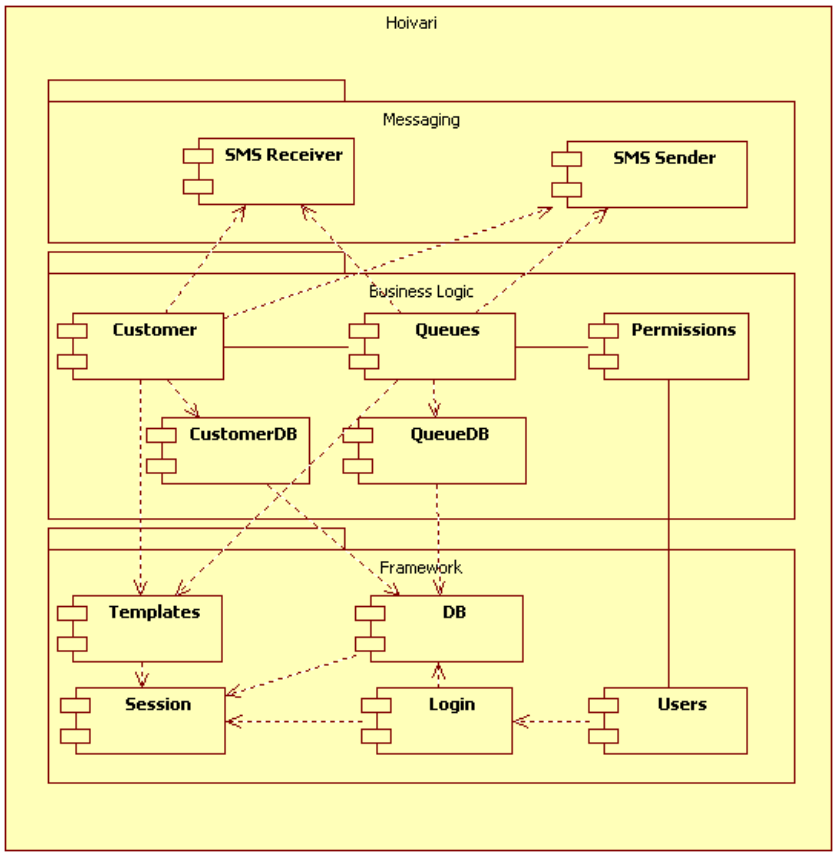
Palvelun luonteen vuoksi ei ajonaikaisen näkymän tuottaminen toisi suurta lisäarvoa palvelun suunnittelulle, koska kyseessä on palvelu, joka toimii tilakoneen tavoin, seuraten jonossa olevan asiakkaan tiloja. Asiakkaan tilakaavio on esitelty kuviossa 17.

Palvelu kokonaisuudessaan muodostuu kahdesta osasta: itse palvelusta sekä viestinvälitysratkaisusta, joka puolestaan on riippuvainen myös Elisa Oyj:n tarjoamasta MBR-palvelusta. Palvelun looginen rakenne voidaan nähdä kuviossa 18, josta ilmenee asiakas- ja jonotietojen riippuvuus viestinvälitysratkaisusta.



KUVIO 18. Hoivari-palvelun looginen näkymä

Palvelun arkkitehtuurinen malli on kolmikerroksinen. Jokaisella kokonaisuudella on ensinnäkin korkean tason luokka, joka toteuttaa käyttöliittymät, ja tulkitsee käyttäjän tekemät toiminnot ja pyynnöt. Toiseksi kutakin korkean tason luokkaa vastaa matalan tason luokka, joka muodostaa tietokantakyselyt ja tarjoaa funktionaalisen liitynnän tietokannan tietueisiin. Matalimmalla tasolla on vielä tukiluokkia, kuten DB-luokka, joka muodostaa tietokantayhteydet ja suorittaa muodostetut tietokantakyselyt. Lisäksi tukiluokkia ovat Template-luokka, jota käytetään käyttöliittymien muodostamiseen sivupohjien (templates) pohjalta, Session-luokka, jota käytetään istunnonaikaisten tietojen ylläpitämiseen sekä Login-luokka, joka tarjoaa käyttäjän tunnistuksen sekä käyttöoikeuksien hallinnan. Parhaiten palvelun komponentteja ja niiden välisiä riippuvuuksia kuvaa 3+1 näkymää mallin kehitysnäkymä, joka on nähtävissä kuviossa 19.

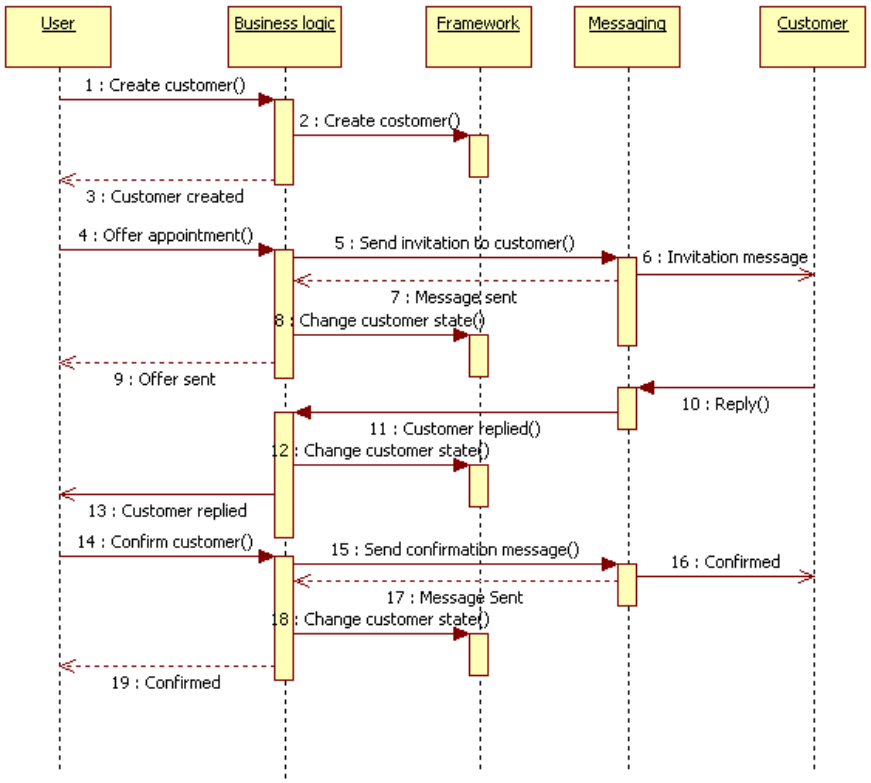


KUVIO 19. Hoivari-palvelun kehitysnäkymä

Kaikki alemman tason ”DB”-luokat tarjoavat korkeamman tason luokille perustoiminnot tietueiden ylläpitämiseen. Tällaisia ovat esimerkiksi lisäys-, muokkaus-, haku- ja poistotoiminnot. Näitä toimintoja yhdistelemällä ja tietoa suodattamalla sekä sen esitystapaa muuttamalla voitiin käyttöliittymän toiminnot muodostaa suoraviivaisesti ja päästiin keskittymään itse toiminnallisuuteen. Lisäksi useita toimintoja tarvittiin monessa eri näkymässä hieman eri parametrein, joten myös se tuki tietokantaluokkien eriyttämistä. Kuvio 17 esittää palvelun komponentit sekä niiden väliset riippuvuudet.

Käyttäjien hallintaan laajennettiin jo olemassaolevan frameworkin käyttäjä-luokkaa Hoivari-sovelluksen tarpeisiin. Lisäksi palvelulle luotiin oma permissions luokka, joka ylläpitää käyttäjäkohtaiset oikeudet ja joka tarjoaa ohjelmallisen rajapinnan käyttöoikeuksien tarkasteluun.

Viimeisenä 3+1 näkymän mallissa on skenaarionäkymä, joka kuvaa muissa näkymissä olevien elementtien rooleja, vastuita sekä niiden välisiä rajapintoja ja vuorovaikutusta. Sen tarkoitus on toimia yhteenvetona ja antaa käsitys ohjelman osien toiminnasta valittujen käyttötapausten yhteydessä (Aalto 1999, 3).



KUVIO 20. Skenaarionäkymä Hoivari-sovelluksesta

Kuviossa 20 esitelty skenaarionäkymä kuvaa Hoivari-palvelun osien vastuita ja toimintoja seuraavissa käyttötapauksissa: Asiakkaan ylläpitäminen (kuviossa esitelty asiakkaan lisääminen palveluun) (kohdat 1 - 3), toimenpideajan tarjoaminen (4 - 9), asiakkaan vastaaminen (10 - 13) sekä asiakkaan kutsuminen eli tarjouksen vahvistaminen (14 - 19). Kuviosta nähdään selkeästi, että Business Logic -moduuli tarjoaa käyttäjälle pääsyn palvelun toimintoihin muodostamalla käyttöliittymät ja käyttämällä muita moduuleita toiminnallisuuden toteuttamiseen. Asiakas puolestaan on aina yhteydessä Messaging, eli viestinvälitys, -moduuliin, joka puolestaan tarjoaa asiakkaalle pääsyn palvelun toimintoihin.

4.5 Palvelun toiminnalliset vaatimukset

4.5.1 Käyttäjien tunnistus

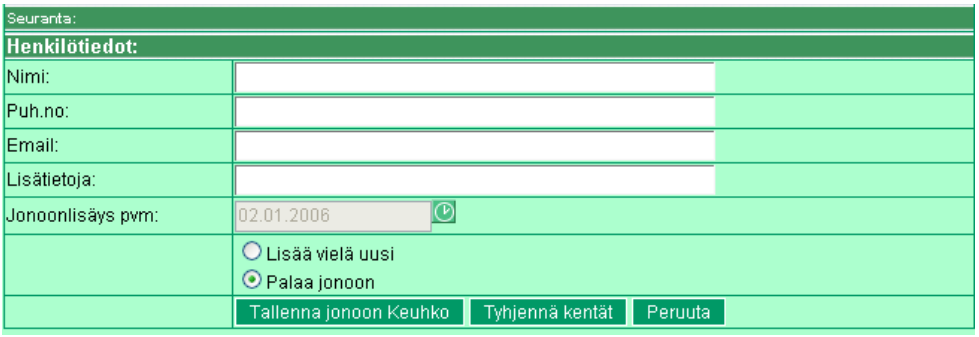
Palvelun sovelluslogiikka voidaan jakaa kolmeen osaan: käyttäjän tunnistaminen, jononhallinta sekä viestinvälitys. Käyttäjän tunnistamisen yhteydessä tarkistetaan lisäksi käyttäjien oikeudet olemassa oleviin asiakasjonoihin. Tunnistaminen tapahtuu yksinomaan käyttäjätunnuksen ja salasanan perusteella.

Istunnon ylläpitämiseen käytetään PHP:n sisäänrakennettua istuntoa, joka luo istuntoa aloittaessa evästeen, jossa on käytännössä satunnainen ja yksilöllinen sisältö. Kaikki istunnon tiedot tallennetaan kuitenkin palvelimelle tilapäis-tiedostoihin, jotka poistetaan istunnon päätyttyä.

Istunto päättyy kun käyttäjä klikkaa "kirjaudu ulos" -linkkiä käyttöliittymässä tai kun palvelu on ollut käyttämättömänä kolme tuntia. Myös istunnon vanheneminen on PHP:n ominaisuus, joten siitäkään ei tarvinnut huolehtia sovellusta toteutettaessa.

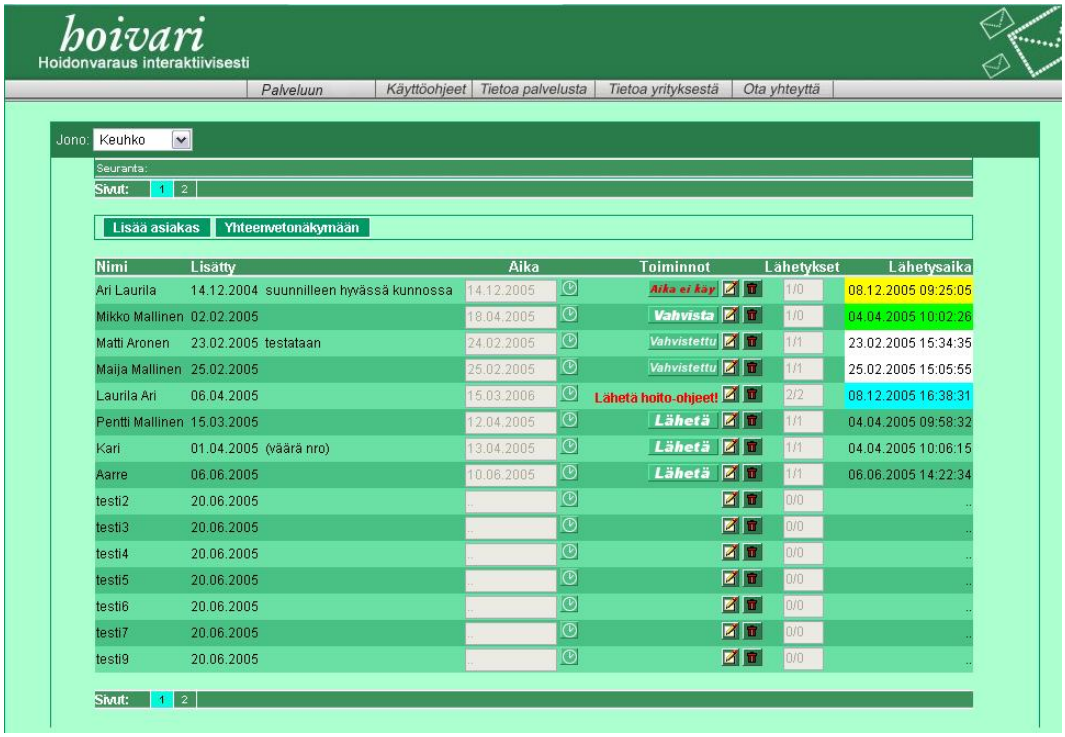
4.5.2 Jononhallinta

Jononhallinta palvelussa on suoraviivaista. Asiakkaita voi lisätä, heidän tietojaan voidaan muokata ja heitä voidaan poistaa jonoista. Käyttäjä voi määritellä asiakkaiden tiedoista nimen, puhelinnumeron, sähköpostiosoitteen, jonoon-lisäyspäivämäärän sekä vapaamuotoisen tekstin. Myöhemmin käyttäjien toiveesta lisättiin mahdollisuus siirtää asiakas jonosta toiseen.



KUVIO 21. Asiakkaan tiedot

Käyttöliittymä, jossa asiakkaan tietoja ylläpidetään, on nähtävillä kuviossa 21. Jonoonlisäyspäivämäärä on oletusarvoisesti kuluva päivä, mutta se voidaan vaihtaa miksi tahansa päiväksi.



KUVIO 22. Hoivari-palvelun jononäkymä

Jononäkymässä on asiakkaat on järjestetty jonoonlisäyspäivämäärän mukaan, ja kutakin asiakasta päästään muokkaamaan kyseisen asiakkaan rivillä olevalla muokkaa-painikkeella. Jononäkymän käyttöliittymä on esitetty kuviossa 22. Jononäkymään siirrytään yleensä yhteenvetonäkymän kautta. Siitä voidaan

seurata koko palvelun tilaa, koska siinä on esitettynä ne jonot, joissa asiakkaalta odotetaan vastausta tai asiakas on jo vastannut. Yhteenvetonäkymän esimerkki on nähtävillä kuviossa 23.

KUVIO 23. Hoivari-palvelun yhteenvetonäkymä

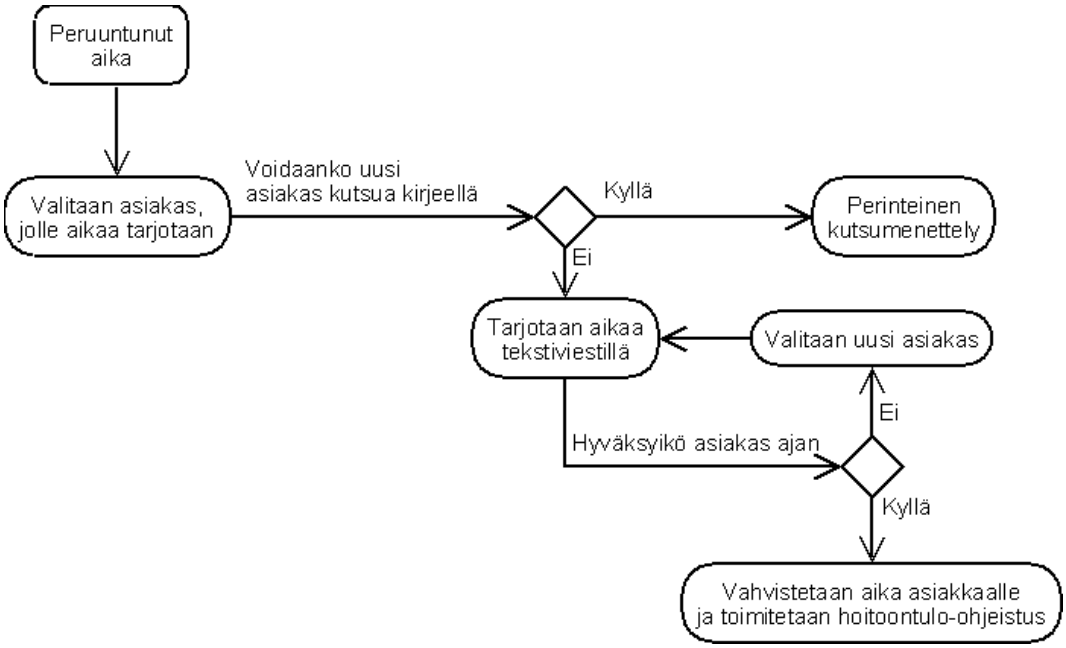
4.5.3 Peruuntuneiden aikojen tarjoaminen

Kun asiakkaat tutkimusten jälkeen lisätään toimenpidejonoon, tiedustellaan heiltä, mikäli heillä on mahdollisuus saapua toimenpiteeseen lyhyellä varoitusajalla. Asiakkaat, jotka ovat halukkaita saapumaan lyhyellä varoitusajalla, lisätään uuden kutsumenettelyn piiriin.

Asiakkaat peruuttavat toimenpiteitänsä erilaisiin syihin vedoten. Riippuen sairaalan käytännöstä asiakkaat saavat peruuttaa toimenpideaikansa yhden kerran ilman hyvää syytä. Yleensä hyväksi syyksi katsotaan lääketieteelliset perusteet; esimerkiksi toimenpiteitä ei yleensä voidan suorittaa, mikäli asiakas ei ole terve.

Mikäli asiakas peruuttaa toimenpiteen, kun siihen on reilusti aikaa, käytetään perinteistä kutsumenettelyä, eli valitaan jonosta uusi asiakas, jolle lähetään kirjeenä toimenpideaika sekä hoitoontulo-ohjeet. Mikäli toimenpide peruutetaan, kun siihen on alle viikko aikaa, käytetään uuden asiakkaan löytämiseksi luotua tekstiviestipalvelua. Kutsumenettelyn valinta ja sen eteneminen on esitetty kuviossa 24.

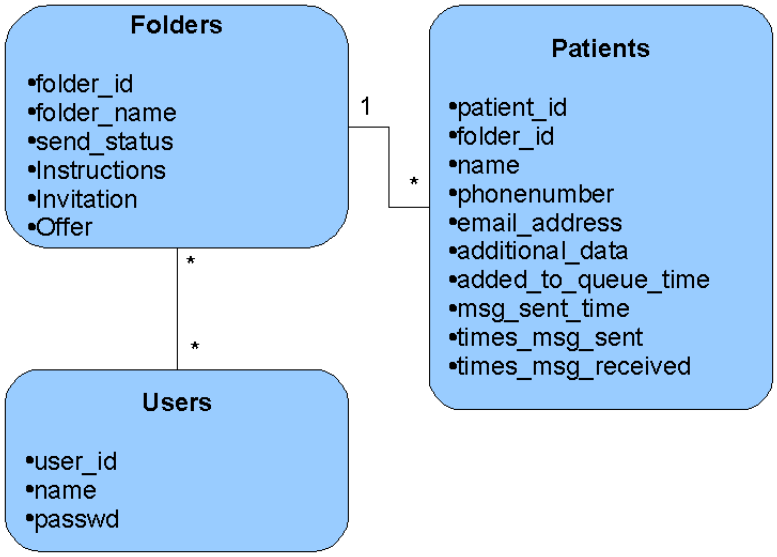
Ajan tarjoaminen tekstiviestipalvelun avulla toimii käytännössä siten, että jononhoitaja valitsee uuden asiakkaan jonosta, lähettää hänelle tekstiviestin, jossa toimenpideaikaa tarjotaan. Mikäli asiakas hyväksyy hänelle tarjotun ajan, hän vastaa tekstiviestiin ohjeistetulla tavalla. Käyttäjä näkee asiakkaan vastauksen ja toimii sen osoittamalla tavalla. Kun on löytynyt asiakas, joka hyväksyy tarjotun ajan, lähetetään hänelle hoitoontulo-ohjeistus ja tehdään vaaditut varaukset resurssienhallintajärjestelmään.



KUVIO 24. Peruuntuneen toimenpideajan myyminen

4.6 Tietokanta

Koska palvelu on pääosin jononhallintasovellus, on myös tietokanta keskittynyt jonojen ympärille. Tietokannan rakenne nähdään kuviosta 25. Valmiin tietokannan kolmesta taulusta kaksi liittyy jonoihin tai jonottamiseen ja yksi sisältää käyttäjien tunnistamiseen ja oikeuksiin vaadittavia tietoja.



KUVIO 25. Hoivari-sovelluksen tietokanta

Jonoista (folders) tietokantaan tallennetaan jonon nimi ja sekä kutsuviestin että vahvistusviestin pohjat. Palvelusta lähetettävät viestit ovat siis jonokohtaisia. Lisäksi jonoille annetaan luomishetkellä yksilöllinen tunnistenumero, jonka kautta yksittäisiin jonoihin voidaan sovelluslogiikassa ja tietokannan relaatioissa viitata.

Kustakin asiakkaasta tallennetaan tietokantaan nimi, sähköpostiosoite, asiakkaan puhelinnumero ja vapaamuotoinen teksti, jossa voi olla vapaamuotoista asiakkaan hoitoon liittyvää tietoa. Lisäksi heille määritellään jonoonlisäyspäivämäärä, jonka mukaan heidät järjestetään palvelussa ja viimeisimmän viestin lähetysaika. Lopuksi luodaan yhteys jonoon, jossa kyseinen asiakas on.

Myöhemmin asiakkaiden tietoihin lisättiin tiedot siitä, kuinka monta viestiä asiakkaalle on lähetetty ja montako viestiä asiakkaalta on vastaanotettu. Käyttäjät halusivat siis seurata jonossa olevien asiakkaiden aktiivisuutta.

Käyttäjistä tietokantaan tallennetaan puolestaan käyttäjän nimi, joka näkyy käyttöliittymässä. Lisäksi käyttäjien tunnistamista varten tietokantaan tallennetaan kirjautumistunnus, joka on enintään 50 merkkiä pitkä, ja salasanan heksa-muotoinen md5 -summa, joka on siis 32 merkkiä pitkä. Käyttäjän tunnistetietojen lisäksi tallennetaan tieto siitä, mihin jonoihin kyseisellä käyttäjällä on käyttöoikeus.

4.7 Kohdatut ongelmat

Koska palvelusta haluttiin saada nopeasti ensimmäiset prototyypit koekäyttöön, tuli alun suunnitteluista ja dokumentaatiosta vajavaiset, ja sen seurauksena sovelluskehityksessä kohdattiin joitakin ongelmia. Prosessi sinänsä oli hyvin suunniteltu, mutta toteutuksen kaikkia yksityiskohtia ei ollut otettu huomioon.

Ensimmäinen kohdattu ongelma oli se, että mikäli samaa aikaa tarjotaan useammalle asiakkaalle, tulisiko sopivan asiakkaan löytyessä siirtää muut asiakkaat takaisin jonoon ja ilmoittaa heille, että aika annettiin jollekin muulle. Kävi kuitenkin ilmi, että samassa jonossa voi olla useampia samanlaisia operaatioita, joten samaa sama aika voidaan myydä usealle asiakkaalle.

Ratkaisu toteutettiin siten, että toimenpideajat varmistetaan ja peruutetaan aina asiakaskohtaisesti, eli käyttäjä joutuu varmistamaan kunkin asiakkaan yksi kerrallaan.

Toisena eteen tulleena asiana oli asiakkaiden järjestäminen: asiakkaita ei välttämättä lisätä palveluun samassa järjestyksessä, kun heidät on lisätty jonoon. Näin ollen jononhoitajalta kysytään myös jonoonlisäyspäivämäärä, kun hän lisää asiakasta jonoon ja asiakkaat listataan jonotusajan mukaan järjestettynä.

4.8 Muutokset palveluun

Koska palvelusta tuotettiin prototyyppejä, oli odotettavissa, että siihen tulee muutoksia koekäytön jälkeen. Ensimmäiset muutokset koskivat lähinnä käyttöliittymää; sitä mikä oli käyttäjälle oleellista. Käyttöliittymän jononäkymästä poistettiin puhelinnumero ja sen tilalle laitettiin jonoonlisäyspäivämäärä ja tieto asiakkaalle lähetettyjen ja asiakkaalta vastaanotettujen viestien lukumäärät.

Seuraava muutos oli lisätä jokaiselle asiakkaalle kohta, johon jononhoitaja voi kirjoittaa jonotukseen liittyviä lisätietoja asiakkaasta. Lisäksi haluttiin toteuttaa mahdollisuus siirtää asiakas jonosta toiseen.

Viimeisenä käyttöliittymään lisättiin yhteenvetonäkymä, josta jononhoitaja voi nopeasti nähdä kaikkien jonojensa tilanteen; kun jonojen määrä kasvoi ja useammasta kuin yhdestä jonosta tarjottiin toimenpideaikaa asiakkaalle, piti jononhoitajan vaihtaa näkymää useiden jonojen välillä, että hän pystyi seuraamaan kokonaistilannetta. Yhteenvetonäkymästä hän pystyy seuraamaan hänelle tärkeitä tapahtumia kustakin jonosta samanaikaisesti.

5 YHTEENVETO

Tavoite oli tukea erikoissairaanhoidon asiakkuudenhallinnan prosessia. Sitä alettiin kehittää yhteistyössä Päijät-Hämeen keskussairaalan päiväkirurgian yksikön kanssa. Tavoite saavutettiin luomalla tekstiviestipalvelu ja muokkaamalla prosessia siten, että tekstiviestipalvelua voitiin hyödyntää.

Kuten odotettiinkin, palvelun käyttö säästää jononhoitajan aikaa. Lisäksi asiakkailta on tullut hyvää palautetta, koska palvelu antaa asiakkaille enemmän aikaa järjestää ehdotettu aika sopivaksi; aikaisemmin asiakas on joutunut päättämään tulemisestaan puhelinkeskustelun aikana ja nyt heillä on 2 tuntia aikaa antaa vastaus.

Asiakastyytyväisyyden lisäksi palvelun käyttöönoton jälkeen on saatu osa palvelun piiriin valituista jonoista tyhjennettyä ja jononhoitaja voi keskittyä paremmin edelleen pitkiin jonoihin. Palvelu on vakiinnuttanut asemansa osana päiväkirurgian osaston jononhoitajan työkaluja.

Palvelun tuottaminen eteni projektissa vaiheittain ja ennen lopullista versiota useita prototyppejä tuotettiin ja edelleenkehitetiin. Ongelmia aiheutui lähinnä siitä, että palvelun käyttäjät eivät osanneet kertoa, minkälainen palvelun tulisi olla, vaan heille piti luoda useita esimerkkejä, joiden toimintoja yhdistelemällä lopullinen tuote valmistettiin.

Palvelua ollaan edelleenkehittämässä Pohjois-Karjalan Sairaanhoitopiirin kanssa. Heidän avullaan on palveluun luotu integraatio hoidonvarausjärjestelmään, jollon jononhoitajan ei tarvitse syöttää asiakkaita käsin palveluun, vaan asiakkaiden tiedot siirtyvät automaattisesti. Lisäksi palvelun toiminnallisuutta on laajennettu.

Kun vastaavia palveluita aletaan kehittää, tulisi toiminnallisuutta miettiä asiakkaan kanssa tarkemmin ennen varsinaisen kehitystyön aloittamista. Mikäli kehitysprojektin alkuvaiheessa olisi käyttäjille esitelty toimimattomia käyttöliittymämalleja, olisi tuotettujen prototyyppien määrä todennäköisesti jäänyt pienemmäksi. Lisäksi, kun olen tutustunut Django-kehitysympäristöön

(http://www.djangoproject.com/) ja toteuttanut palveluita sen avulla, käyttäisin toteutuksessa todennäköisesti sitä. Se on osoittautunut tehokkaaksi kehitys-ympäristöksi, jonka avulla voidaan keskittyä sovelluslogiikkaan paremmin, koska tietokantayhteydet ja kyselyt suoritetaan malli-luokkien kentille automaattisesti. Lisäksi siinä on kehittynyt template-moottori, joka tukee muun muassa muuttujia, silmukoita ja ehtorakenteita. Django on python kehitysympäristö, joten sitä käytettäessä palvelimen tulee voida suorittaa python koodia. Esimerkiksi Apache versio 2:n saa mod_python-moduulin tai python koodia voidaan ajaa cgi-moduuleina. Juuri pythonin käyttö tekee koodista siirrettävää, koska python-tulkkeja saa lähes mille tahansa käyttöjärjestelmälle. Lisäksi siirrettävyyttä tukee se, että Django tukee useita tietokantamoottoreita.

LÄHTEET

Aalto, J. 1999. 3 + 1 UML-näkymää arkkitehtuuriin [verkkojulkaisu]. Saatavissa: http://www.pcuf.fi/

AT Command Set For Nokia GSM And WCDMA Products. 2005 [verkkojulkaisu]. [viitattu 18.4.2007]. Saatavissa: http://forum.nokia.com/

CIMD Interface Specification. 2005 [verkkojulkaisu]. [viitattu 18.4.2007] Saatavissa: http://forum.nokia.com/

Ishida, R. Serving XHTML 1.0 [verkkojulkaisu] [viitattu 13.4.2007]. Saatavissa: http://www.w3.org/International/articles/serving-xhtml/Overview.en.php

Martikainen, E. 2007. Avanto -Hankesuunnitelma.

Metropolis Developer's Guide 4 - SMS Messaging Broker 3.1. 2004 [elektroninen julkaisu]. [viitattu 10.4.2007] Saatavissa: Elisa Oyj

Short Message Peer-to-Peer Protocol Specification. 2003 [verkkojulkaisu]. [viitattu 12.4.2007] Saatavissa: http://smsforum.net/

Short Message Service / SMS Tutorial. 2006 [verkkojulkaisu]. [viitattu 1.4.2007] Saatavissa: http://www.developershome.com/sms/

SMS reading and SMS sending to the Phone from HyperTerminal. 2007 [verkkojulkaisu]. [viitattu 5.4.2007] Saatavissa: http://www.usbdeveloper.com/GSMPage/gsmpage.htm#SMS

LIITTEET

LIITE 1 CIMD -rajapinnan tarkistesumman laskeminen

```
function count_checksum($message_str)
{
    # alustetaan tarkistesumman arvo
    $checksum = 0;
    for ($i=0; $i<strlen($message_str); $i++) {
        # lisätään tarkistesummaan merkin arvo
        # ja otetaan huomioon vain lsb
        $checksum = ($checksum + ord($message_str[$i])) & 0xFF;
    }
    return dechex($checksum);
}
```

Vastaava funktio pythonilla toteutettuna:

```
# Funktio, joka laskee annetusta
# merkkijonosta tarkistesumman.
# Syötteenä annetaan lähetettävä viesti ilman
# lopetusmerkkiä ja tarkistesummaa.
def count_checksum(message_str):
    checksum = 0
    for c in message_str:
        checksum = (checksum + ord(c)) & 0xFF
    return checksum
```