TAMPEREEN AMMATTIKORKEAKOULU
Tietotekniikan koulutusohjelma
Ohjelmistotekniikka

Opinnäytetyö

Tom Kiljo

MobilePGP, PGP-salaustekniikka S60-ympäristössä

Työn ohjaaja Lehtori Tony Torp

Työn teettäjä Digia Finland Oy, valvojana diplomi-insinööri Pasi Pitkänen

Tampere 5/2009

TAMPEREEN AMMATTIKORKEAKOULU
Tietotekniikka
Ohjelmistotekniikka

Tekijä	Kiljo Tom
Työn nimi	MobilePGP, PGP-salaustekniikka S60-ympäristössä
Sivumäärä	46 sivua
Valmistumisaika	Toukokuu 2009
Työn ohjaaja	Lehtori Tony Torp
Työn teettäjä	Digia Finland Oy, valvojana diplomi-insinööri Pasi Pitkänen

TIIVISTELMÄ

Työn aiheena on toteuttaa PGP-sovellus Nokian S60-ympäristöön. PGP (Pretty Good Privacy) on salaustekniikka, joka on erityisen suosittu sähköpostiviestien salaukseen. Tällä hetkellä vastaavaa sovellusta ei ole tarjolla mobiililaitteille. Tarve mobiilille ratkaisulle on kuitenkin suuri.

Työssä käytetään hyväksi Digia Finland Oy:lle Tampereen teknillisen yliopiston projektityönä Symbian-alustalle käännettyä avoimen lähdekoodin PGP-komentorivisovellusta. Työssä komentorivisovelluksen ympärille luodaan graafinen käyttöliittymä, jolla voidaan helposti käsitellä salattua tietoa, erityisesti sähköpostiviestejä.

Työssä tarkastellaan myös useita Symbian-ympäristön suunnittelu- ja toteutusmalleja sekä tutkitaan syvällisesti eri teknologioita, kuten Open C:tä ja viestintäarkkitehtuuria.

Avainsanat Symbian OS, S60, Open C, PGP, salaustekniikka

TAMPERE UNIVERSITY OF APPLIED SCIENCES
Computer Systems Engineering
Software engineering

Writer	Kiljo Tom
Thesis	MobilePGP, PGP encryption tool for S60 platform
Pages	46 pages
Graduation time	May 2009
Thesis supervisor	Tony Torp (lecturer)
Co-operating company	Digia Finland Plc, Supervisor: Pasi Pitkänen (MSc)

ABSTACT

The subject of this thesis is to implement a PGP application for Nokia S60 environment. PGP or Pretty Good Privacy is a encryption technique that is very popular for encrypting email. For now there is no application available for mobile devices. The need for such an application is still large.

An open source PGP command line application ported to Symbian environment will be used in the work. The application has been ported to Symbian environment for Digia Finland Plc by a group of Tampere University of Technology students as part of their project work course. In the thesis a graphical user interface will be created around the command line application which can be used to easily handle encrypted information, especially email messages.

During the work Symbian design and implementation patterns will be studied. Also different technologies such as Open C and the messaging architecture will be explored deeply.

Keywords Symbian OS, S60, Open C, PGP, encryption technique

ALKUSANAT

Tämä tutkintotyö on tehty vuosien 2008 ja 2009 aikana Digia Finland Oyj:lle. Työn tarkoituksena oli toteuttaa PGP-sovellus Nokian Symbian-pohjaiseen S60-ympäristöön, jota voidaan hyödyntää niin sisäisesti kuin myös markkinoinnissa asiakkaiden suuntaan. Työn myötä olen saanut myös kehittää omaa osaamistani.

Tahtoisin kiittää Pasi Pitkästä työn aiheesta ja mahdollisuudesta toteuttaa sitä kokopäiväisesti työssäni. Erityiskiitokset Lappeenrannan työtovereilleni Matti Laamaselle, Matti Väänäselle ja Tomi Paanaselle kovasta panostuksesta käyttöliittymän toteutuksessa sekä sovelluksen testaamisessa. Kiitokset myös työtoverilleni Iita Pousille avusta käyttöliittymän suunnittelussa.

Tampereella 28. huhtikuuta 2009

_____

Tom Kiljo

SISÄLLYSLUETTELO

LYHENTEET JA TERMIT

PGP Pretty Good Privacy, salaustekniikka suojattuun sähköiseen viestintään.

GnuPG Gnu Privacy Guard, OpenPGP-standardiin perustuva avoimen lähdekoodin sovellus.

MIME Multipurpose Internet Mail Extension, sisällön tyypin kuvaukseen käytetty tekniikka.

ANSI C American National Standars Instituten julkaisema standardi C-ohjelmakoodille.

POSIX Portable Operating System Interface for Unix, alunperin Unix-pohjaisille käyttöjärjestelmille suunnattu alustariippumaton standardi, käytössä myös muissa käyttöjärjestelmissä.

PIPS Posix on Symbian OS, kokoelma POSIX-standardeja noudattavia kirjastoja Symbian-ympäristössä.

DLL Dynamic Link Library, jaettu tai ajonaikaisesti ladattava kirjasto.

UID Unique identifier, yksilöivä tunniste.

RSA Julkisen avaimen salausalgoritmi.

DSA Digital Signarure Algorithm, digitaalisen allekirjoituksen salausalgoritmi.

MTM Message Type Module, tapa laajentaa Symbian OS:n viestintäominaisuuksia.

1 JOHDANTO

Työn tehtävänä oli ohjelmoida sovellus, jolla voidaan suorittaa PGP-salaustekniikkaan liittyviä operaatioita, erityisesti sähköpostiviestinnässä. Sovellus koostuu useasta eri osasta. Työn painopiste on erityisesti Symbian-ympäristöön siirretyssä GnuPG-komentorivisovelluksessa sekä sen käyttöön liittyvässä kirjastossa.

PGP on laajalti käytetty salaustekniikka erityisesti sähköpostiviestinnässä. PGP:n tuomat turvallisuutta parantavat ominaisuudet tekevät siitä tehokkaan apuvälineen salatun tiedon jakamiseen. Tämän hetken tiedon mukaan ei ole todistettu, että PGP:n salausta oltaisiin pystytty purkamaan matemaattisesti. Erityisesti sen tuomaa turvallisuutta arvostavat tietotekniikka-alan yritykset sekä avoimen lähdekoodin yhteisöt. PGP:tä tukevia ohjelmia on tarjolla, niin ilmaisina, kuin kaupallisinakin versioina. PGP:stä ja sen ominaisuuksista on kerrottu luvussa 2.1.

MobilePGP:n päämääränä on tuoda PGP-salaustyökalut myös Nokian Symbian-pohjaiseen S60-ympäristöön. Vastaavaanlaista sovellusta ei toistaiseksi ole.

Työn tavoitteena on käytettävän sovelluksen toteuttamisen lisäksi oppia syvällisemmin eri teknologioita ja tapoja toteuttaa sovelluksia Symbian OS:n päälle. Käytännössä siis: erilaiset suunnittelumallit, Open C ja valmiiden ANSI C -lähdekoodien käyttö Symbian OS:llä (luvussa 2.2), viestintäarkkitehtuurin (*messaging architecture*) käyttö (luvussa 6.1), sekä käyttöjärjestelmän liitännäisten soveltaminen (luvussa 8).

Työ on jatkoa Tampereen teknillisen yliopiston projektityökurssin projektille, jonka tavoitteena oli tuoda GnuPG-ohjelma Nokian S60-ympäristöön.

2 TAUSTAA

2.1 PGP

PGP (*Pretty Good Privacy*) on alunperin Philip Zimmermannin vuonna 1991 kehittämä salaustekniikka ja ohjelma, jotka kantavat samaa nimeä. PGP perustuu niin kutsuttuun julkisen ja salaisen avaimen salaustekniikkaan, mutta tarjoaa lisäksi myös mekanismin, jolla julkiset avaimet voidaan sitoa käyttäjänimiin ja/tai sähköpostiosoitteisiin. PGP:n perusajatus on, että käyttäjällä on oma avainpari, joka koostuu toisistaan eroavista julkisesta sekä salaisesta avaimesta. Avaimet liittyvät toisiinsa matemaattisesti, mutta kumpaakaan avainta ei kuitenkaan voida johtaa toisesta. Tieto voidaan salata julkisella avaimella ja purkaa vain sitä vastaavalla salaisella avaimella. /6/

Jokaisella käyttäjällä on oma avainrengas, jossa säilytetään muiden käyttäjien julkisia avaimia sekä omia salaisia avaimia. Käyttäjät jakavat omia julkisia avaimiaan henkilöille, joiden kanssa he haluavat välittää salattua tietoa. Yleisimmin julkiset avaimet lähetetään avainpalvelimelle, josta jokainen voi tarvittaessa hakea tarvitsemansa avaimet omaan avainrenkaaseensa.

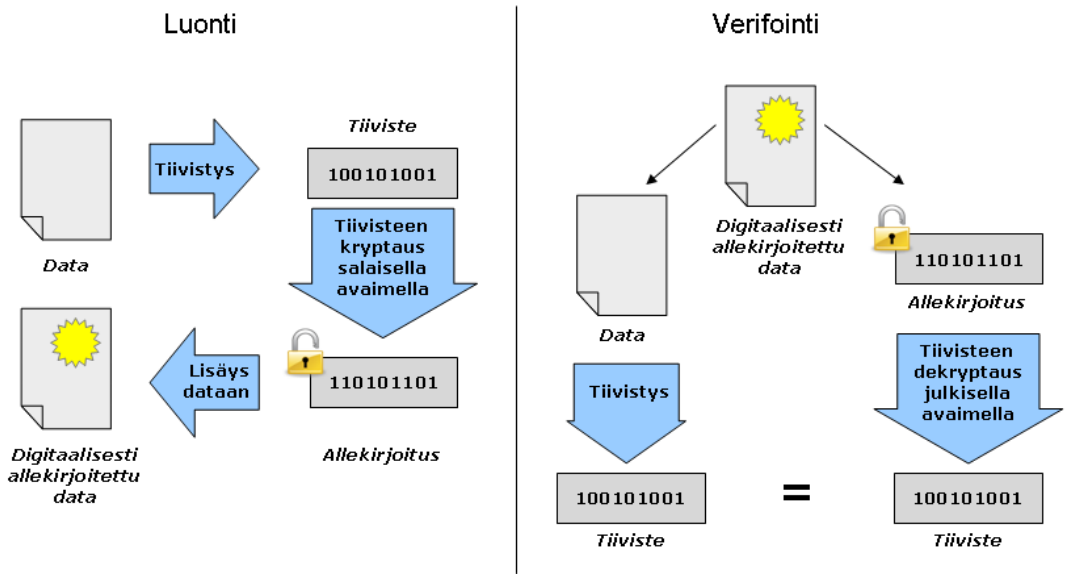
PGP-turvallisuus perustuu siihen olettamukseen, että käytettyjä algoritmeja ei pystytä murtamaan suoralla kryptoanalyysillä tämän hetken laitteistoilla. Tällä hetkellä ei ole tiedossa tapaa, jolla pystyttäisiin murtamaan PGP-salausta. Haavoittuvuuksia paikataan myös välittömästi, kun ne tulevat kehittäjien tietoon, esimerkiksi lisäämällä uusia salausalgoritmeja. Hyvä osoitus PGP-salauksen turvallisuudesta on se, että vuodesta 2007 eteenpäin Ison-Britannian poliisi on murtamisyritysten sijaan voinut lain voimalla vaatia salaisen avaimen ja salasanan luovutusta.

Nykyiset PGP-salaustekniikkaa käyttävät ohjelmat pohjautuvat IETF:n (*Internet Engineering Task Force*) tuottamaan OpenPGP-standardiin. Esimerkiksi GnuPG (*Gnu Privacy Guard*) on OpenPGP-määritysten mukainen ohjelma. /7/

2.1.1 Digitaaliset allekirjoitukset /6/

PGP tukee tiedon oikeellisuuden sekä eheyden todentamista digitaalisia allekirjoituksia käyttämällä. Allekirjoituksia käytetään usein yhdessä tiedon salauksen kanssa. Tällä tavalla voidaan varmistua siitä, että lähettäjä on sama henkilö kuin hän väittää olevansa ja että tietoa ei ole käsitelty tai se ei ole muuttunut.

Digitaalinen allekirjoitus luodaan joko RSA- tai DSA-algoritmeilla. Allekirjoituksen luontiin käytetään viestistä muodostettua tiivistettä (*hash*) sekä lähettäjän salaista avainta. Viestin vastaanottaja purkaa alkuperäisen viestin sekä tiivisteen digitaalisesta allekirjoituksesta käyttäen lähettäjän julkista avainta. Palautetusta viestistä luodaan uusi tiiviste, jota voidaan verrata alkuperäiseen. Viestin eheys on säilynyt, mikäli uusi ja alkuperäinen tiiviste ovat täysin samat. Digitaalisen allekirjoituksen luonti ja verifiointi on esitetty kuvassa 1.



Kuva 1: Digitaalisen allekirjoituksen luonti ja verifiointi.

2.1.2 Varmenteet /6/

OpenPGP-standardi määrittelee käsitteen *luotetut allekirjoitukset*, joita voidaan käyttää muiden avainten varmentajien luomiseen. Luotettu allekirjoitus osoittaa käyttäjän olevan varma siitä, että avain kuuluu väitetylle käyttäjälle ja että avain on tarpeeksi luotettava varmentamaan askelta alempana olevia avaimia. Näitä avaimia voidaan kutsua varmentajiksi. Avainten varmentamista sekä varmentajien käyttämistä luottamusverkon rakentamisessa käsitellään luvussa 2.1.3.

PGP on myös aina sisältänyt mekanismin identiteettivarmenteiden purkamiseen. Kadonnut tai vaarantunut salainen avain vaatii aina identiteettivarmenteen purkamisen, mikäli kommunikaation turvallisuus halutaan säilyttää.

Avaimen ja käyttäjän väliseen omistussuhteen varmistamiseen ei valitettavasti ole täysin varmaa keinoa, mikä on yhteinen ongelma kaikille julkista / salaista avainparia käyttäville salaustekniikoille. PGP tarjoaa käyttäjälle kuitenkin valinnan mahdollisuuden siitä, haluaako tämä luottaa tai olla luottamatta toisten julkisiin avaimiin.

2.1.3 Luottamusverkko /4/

Salattaessa tietoa sekä varmistettaessa digitaalisia allekirjoituksia on erittäin tärkeää varmistaa, että vastaanottajan julkinen avain todellakin kuuluu oikealle henkilölle. Avaimeen liitetty käyttäjänimi ja / tai sähköpostiosoite ei vielä todista, että henkilö on todellakin avaimen haltija. Perinteisesti avaimen oikeellisuus on tarkistettu vertaamalla omassa avainrenkaassa sijaitsevan julkisen avaimen digitaalista sormenjälkeä henkilökohtaisesti (puhelimitse, paperilla jne.) avaimen omistajan kanssa ennen avaimen allekirjoitusta. 40 merkkiä pitkän numerosarjan vertaaminen kaikkien julkisten avainten omistajien kanssa voi kuitenkin olla melko työlästä. Tästä syystä PGP käyttää perinteisen allekirjoituksen lisäksi joustavaa tapaa varmistua avainten oikeellisuudesta niin kutsutulla luottamusverkolla (*Web of trust*).

Luottamusverkkoa käyttämällä käyttäjä voi asettaa avaimille eri asteisia luottamustasoja (*Owner trust*), joiden avulla voidaan kontrolloida avainrenkaassa olevien muiden avainten oikeellisuutta.

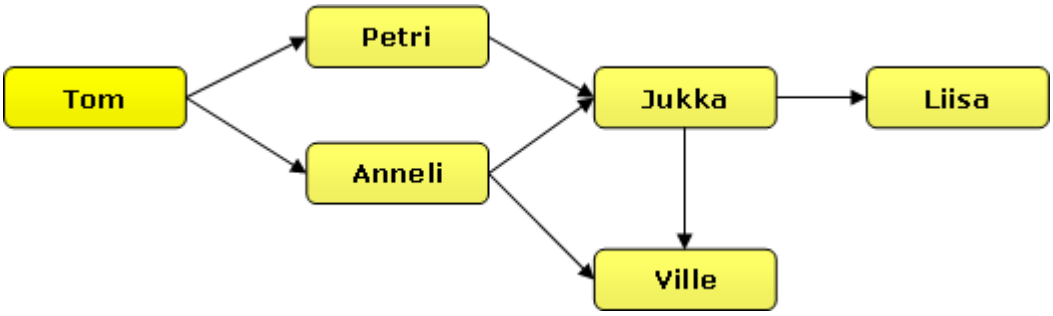
Luottamustasoja ovat:

- ei luotettu
- marginaalisesti luotettu
- täysin luotettu.

Avainta voidaan pitää varmennettuna, mikäli se täyttää seuraavat ehdot:

1. Avaimen on allekirjoittanut tarpeeksi moni luotettava avain
    - avain on allekirjoitettu omalla avaimella.
    - tai avain on allekirjoitettu yhdellä täysin luotetulla avaimella.
    - tai avain on allekirjoitettu kolmella marginaalisesti luotetulla avaimella.
2. Allekirjoitusten polku avaimesta omaan avaimeen on korkeintaan viiden askeleen päässä.

Kuvassa 2 on kuvattuna kuvitteellinen rakenne luottamusverkosta, jossa henkilön Tom avain on polun alkupisteessä. Poiketen todellisesta tilanteesta esimerkeissä avaimen varmentamiseen tarvitaan vain kaksi marginaalisesti luotettua avainta ja pisin polku avaimeen on kolmen askeleen päässä. Esimerkeissä henkilöiden Petri ja Anneli avaimet ovat täysin varmennettuja, koska ne on allekirjoitettu suoraan Tomin avaimella.



Kuva 2: Esimerkki luottamusverkosta.

Mikäli Annelin avain on täysin luotettu, ovat henkilöiden Jukka ja Ville avaimet täysin varmennettuja. Mikäli Annelin ja Petrin avaimet ovat marginaalisesti luotettuja, tulee Jukan avaimesta täysin varmennettu, koska kaksi marginaalisesti luotettua avainta riittää täyteen varmennukseen. Mikäli edelliseen esimerkkiin muutetaan Jukan luotettavuus marginaaliseksi, tulee Villen avaimesta myös täysin varmennettu. Liisan avain on kuitenkin vain marginaaliseti varmennettu, koska Jukan marginaalinen luotettavuus ei yksinään riitä varmentamiseen. (Taulukko 1.)

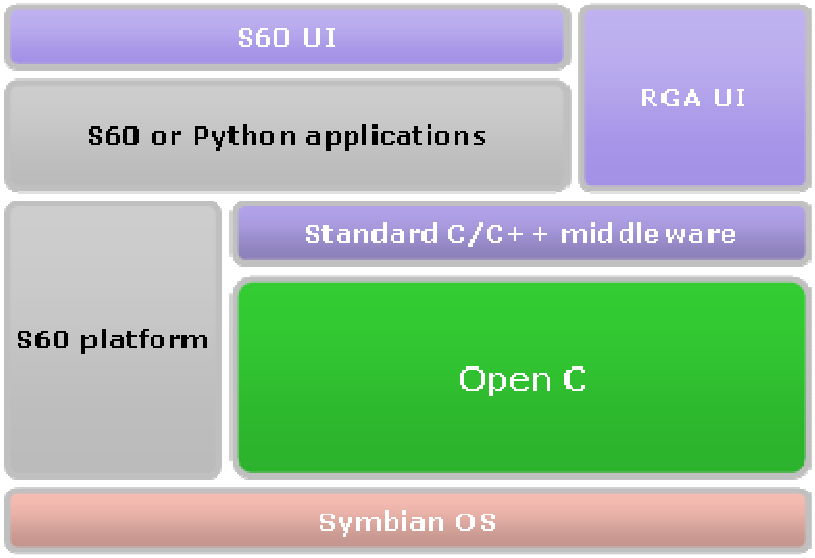
Taulukko 1: Esimerkki luottamuksen vaikutuksesta varmennukseen kuvassa 2.

Luottamus		**Varmennus**	
Marginaalinen	Täysi	Marginaalinen	Täysi
	Anneli		Anneli, Petri, Jukka, Ville
Anneli, Petri		Ville	Anneli, Petri, Jukka
Anneli, Petri, Jukka		Liisa	Anneli, Petri, Jukka, Ville

2.2 Open C /8/

Open C koostuu suuresta joukosta standardeja C-kirjastoja. Tämän teknologian avulla kehittäjät pystyvät siirtämään näitä standardeja noudattavia sovelluksia, sekä kirjastoja muista ympäristöistä S60-ympäristöön. Open C luo tällä tavoin uusia mahdollisuuksia S60-sovellusten kehittämiseen ja on usein hyvä vaihtoehto sekä

sovellusten loogisten toteutusten että middleware-toteutuksiin. Open C:n rajapintojen tehokkuus on rinnastettavissa S60:n rajapintoihin, useissa tapauksissa käyttö on jopa tehokkaampaa. Open C S60-arkkitehtuurissa on esitetty kuvassa 3.



Kuva 3: Open C Nokian S60-arkkitehtuurissa.

Vaikka Open C ei tarjoakaan kaikkia tai edes täydellisiä POSIX- ja middleware-standardikirjastoja, se tuo kehittäjien ulottuville yhdeksän hyvin tunnettua toteutusta:

- **Libc**, joukko standardeja C-kirjastoja, jotka tarjoavat esimerkiksi standardit I/O-funktiot, merkkijono-operaatiot, muistin varaukset sekä aikaan ja päiväykseen liittyvät funktiot.
- **Libdl**, tarjoaa rajapinnan DLL:ien lataamiseen.
- **Libpthread**, tarjoaa rajapinnan useiden säikeiden luontiin ja hallinnointiin prosessin sisällä.
- **Libm**, tarjoaa aritmeettisia ja matemaattisia funktioita.
- **Libz**, tarjoaa funktioita tiedon pakkaamiseen, purkamiseen sekä pakatun tiedon eheyden tarkistamiseen.
- **Libcrypt**, tarjoaa kryptografisia funktioita tiedon salaukseen sekä salasanojen tiivistämiseen (*eng. hash*).
- **Libcrypto**, tarjoaa palveluita, joilla voidaan toteuttaa cryptografisten standardien mukaisia sovelluksia, kuten SSL, SSH, OpenPGP.

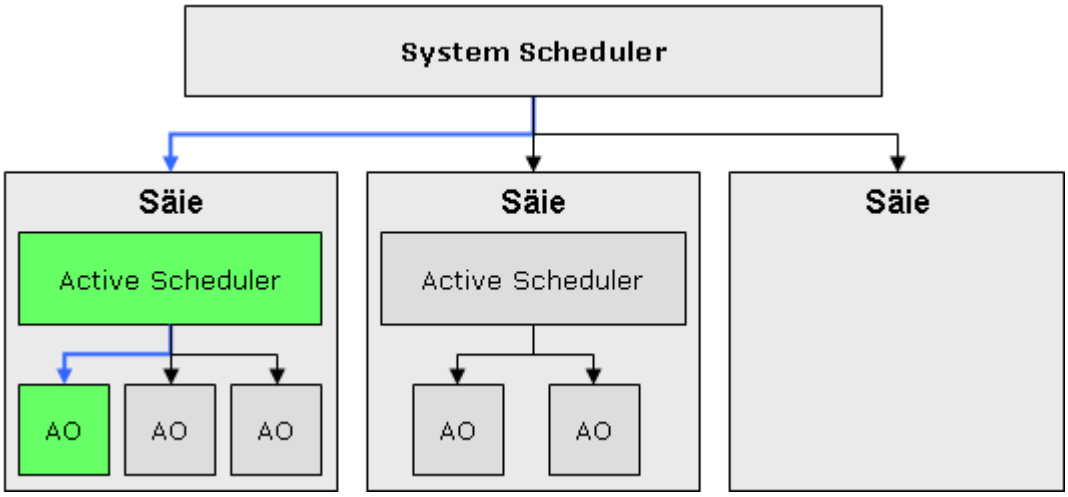
- **Libglib**, tarjoaa yleiskäyttöisiä rajapintoja tyyppimuunnoksiin, merkkijonojen käsittelyyn, ym.
- **Libssl**, tarjoaa toteutuksen SSL 2.0/3.0- ja TLS 1.0 -protokollille.

Open C ei tarjoa rajapintoja S60-käyttöliittymäkomponentteihin tai sovellusmoottoreihin, kuten kalenteriin, kontakteihin tai viestintäkomponentteihin. Myös joitain rajoitteita on otettava huomioon siirrettäessä sovelluksia toisesta ympäristöstä. Nämä johtuvat Symbian OS:n rajoituksista, joita ovat esimerkiksi signalointi sekä prosessien hallinta *fork()* ja *exec()*-funktioilla.

2.3 Aktiiviset oliot /2/

Symbian OS on suunniteltu hyvin asynkroniseksi. Aktiiviset oliot ovatkin laajasti käytössä oleva suunnittelumalli, joka kapseloi sisäänsä asynkronisen pyynnön ja sen vastauksen käsittelyn. Aktiiviset oliot muodostavat moniajoarkkitehtuurin, jossa useampia asynkronisia pyyntöjä voidaan käsitellä yhden säikeen sisällä. Tämä luo huomattavia resursseja säästäviä etuja verrattuna perinteiseen säikeillä toteutettuun moniajoon.

Järjestelmävuorontaja (*System Scheduler*) antaa kullekin säikeelle suoritusaikaa riippuen sen prioriteeteistä. Mikäli säikeelle on rekisteröity aktiivivuorontaja (*Active Scheduler*), tämä jakaa säikeen suoritusaikaa puolestaan sille rekisteröidyille aktiivisille olioille. Myös aktiivisten olioiden suorituksessa käytetään prioriteettijärjestystä. Kuvassa 4 on esitetty aktiivisten olioiden arkkitehtuuri. Kuvassa vasemmanpuoleinen säie on saanut suoritusaikaa järjestelmävuorontajalta, jota aktiivivuorontaja on jakanut eteenpäin suoritusvuorossa olevalle aktiiviselle oliolle.

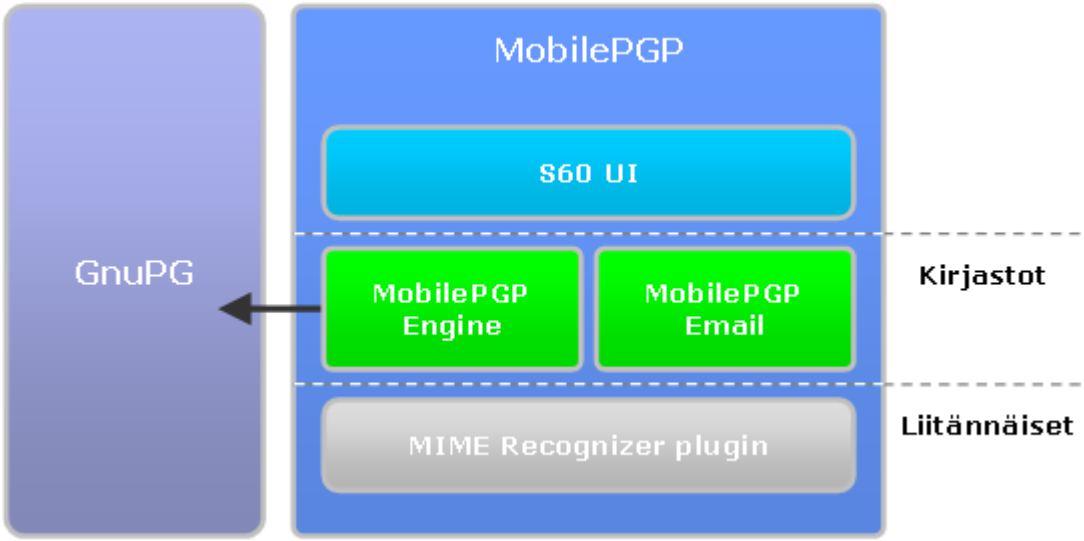


Kuva 4: Aktiivivuorontaja ja aktiiviset oliot.

3 SOVELLUKSEN ARKKITEHTUURI

Tässä kappaleessa esitellään sovelluksen arkkitehtuuri korkealla tasolla. Arkkitehtuurin eri osien tarkemmat kuvaukset ja sisältö on esitelty myöhemmissä kappaleissa.

Sovellus koostuu PGP-komentoriviohjelmasta (*GnuPg*), graafisesta käyttöliittymästä (*S60UI*), kahdesta kirjastosta (*MobilePGP Engine, MobilePGP Email*) sekä tiedostontyypintunnistusliitännäisestä (*MIME Recognizer plugin*). Sovelluksen korkean tason arkkitehtuuri on esitetty kuvassa 5.



Kuva 5: Sovelluksen arkkitehtuuri.

GnuPG:tä voidaan pitää koko sovelluksen ytimenä, sillä se hoitaa kaikki salaukseen liittyvät tehtävät sekä ylläpitää käyttäjän avainrenkaita ja luottamusverkkoa. GnuPg:n toiminnasta ja Symbian S60-ympäristöön siirtoon liittyvistä asioista on kerrottu lisää luvussa 4.

MobilePGP Engine -kirjasto tarjoaa Symbian C++-rajapinnan GnuPg:n käsittelyyn. Sen tarjoamien luokkien avulla salausoperaatiot sekä avainten hallinta on tehty mahdollisimman yksinkertaiseksi. MobilePGP Enginen toiminnasta on kerrottu lisää luvussa 5.

MobilePGP Email -kirjasto tarjoaa rajapinnan olemassa olevien sähköpostitilien, sekä paikallisten kansioiden käsittelyyn. Sen avulla voidaan myös päivittää paikallinen sähköpostilaatikko, noutaa viestejä palvelimelta sekä lähettää viestejä. MobilePGP Emailin toiminnasta on kerrottu lisää luvussa 6.

Graafinen käyttöliittymä yhdessä MobilePGP Engine- ja MobilePGP Email -kirjastojen kanssa muodostavat MobilePGP-ohjelman. Graafinen käyttöliittymä ja sen toiminnot esitellään luvussa 7.

Tiedostontyypin tunnistusliitännäinen on erillinen komponentti, jonka tarkoitus on tehdä MobilePGP:n käytöstä käyttäjäystävällisempää. Liitännäinen rekisteröi MobilePGP:n PGP-salaukseen liittyvien tiedostotyyppien käsittelijäksi. Tällä tavalla tuodaan MobilePGP lähemmäksi päivittäistä käyttöä ja S60-käyttöliittymää. Liitännäisen toiminnasta kerrotaan enemmän luvussa 8.

4 GNU PRIVACY GUARD

Gnu Privacy Guard (lyhyesti GnuPG tai GPG) on avoimen lähdekoodin OpenPGP-standardeja noudattava PGP-komentorivityökalu. Se tarjoaa kryptografisten operaatioiden lisäksi avainten hallinnan käyttäjän omissa avainrenkaissa sekä yhteyden avainpalvelimiin. GnuPG on myös suunniteltu alustariippumattomaksi noudattamalla POSIX-standardeja. Tämä luo hyvän pohjan GnuPg:n tuomiseen myös Symbian S60 -ympäristöön Open C -kirjastojen avulla.

Työssä Symbian S60 -ympäristöön tuotu versio GnuPg:stä on 1.4.9. Myös uudempi versio 2.0 on saatavilla, mutta vanhempaan versioon päädyttiin, koska se on helpompi integroida muiden sovellusten kanssa sekä tuoda uuteen ympäristöön. GnuPg:tä voidaan tällä hetkellä käyttää useiden alustojen päällä (Linux, Windows, Mac), sähköpostiohjelmissa, PHP-koodista jne.

4.1 Tuodut ominaisuudet sekä ongelmat

Vaikka GnuPg onkin ohjelmoitu suurimmaksi osaksi alustariippumattomaksi ja eri alustojen variaatioita käyttäen, ei tuonti Symbian-ympäristöön ole täysin mutkatonta. Tämä johtuu siitä, että Symbian asettaa rajoituksia joillekin POSIX-standardeille, kuten prosessien ohjausfunktioille *fork()* ja *exec()*. Tästä syystä GnuPg:n Symbian versiosta on karsittu pois esimerkiksi kaikki avainpalvelimiin liittyvät operaatiot. Tärkeimmät osiot, kuten kaikki kryptografiset operaatiot, omien avainrenkaiden hallinta sekä luottamusverkon hallinta on kuitenkin saatu onnistuneesti mukaan.

GnuPg: Symbian -version ominaisuudet:

- Datavirran, sekä tiedostojen salaus, allekirjoitus, salaus ja allekirjoitus, salauksen purku, allekirjoituksen vahvistus.
- Yhden tai useamman avaimen tai avainparin tuonti, poisto, allekirjoittaminen, luottamustietojen asettaminen.
- Kokonaisen julkisen tai salaisen avainrenkaan tuonti.

- Luottamusverkon tietokanta (*Trust Database*).

GnuPg:n tuominen Symbian-ympäristöön ei ole aivan mahdoton tehtävä, joskin paljon aikaa vievä, johtuen suuresta koodirivien määrästä (noin 16000 riviä). Johtuen pois jätetyistä ominaisuuksista sekä alustariippuvaisten tiedostojen pois jättämisestä GnuPg:n Symbian-versio on kaiken kaikkiaan noin 10000 koodiriviä pitkä.

Suurimman osan GnuPg:n tuomisessa työtä aiheuttivat tietotyyppien muunnokset ilman oikeanlaista tyyppimuunnosta. Tyypillisin tällainen muunnospari oli char- ja byte-tietotyypit. Koska Symbian-kääntäjä ei tällaisia tapauksia hyväksy, koodiin on tehty todella suuri määrä tyyppimuunnoksia.

4.2 Muutokset ja kääntäminen Symbian S60 -ympäristössä

Tehdyt muutokset varsinaiseen lähdekoodiin ovat kauttaaltaan melko pieniä, mikäli ei lasketa mukaan useita satoja, ellei tuhansia tyyppimuunnoksia. Pääasiassa otsikko- ja lähdekooditiedostoista on kommentoitu viittauksia alustariippuvaisiin otsikkotiedostoihin, sekä ohjelman asetustiedostoa on muokattu vastaamaan Symbian-ympäristöä. Yhtenä merkittävänä muutoksena on ohjelman sisääntulofunktioon *main()* lisätty ominaisuus, jolla ohjelman tulostevirrat *stdout* ja *stderr* voidaan ohjata asetustiedostossa määriteltyihin tiedostoihin. Tämä ominaisuus voidaan myös jättää käännösvaiheessa pois päältä jättämällä siihen tarkoitettu lippu määrittelemättä asetustiedostosta.

Jotta GnuPg voidaan kääntää Symbian-ympäristössä, sille on luotu vaadittavat komponenttimäärittelytiedosto (bld.inf) sekä projektimäärittelytiedosto (.mmp). Komponenttimäärittelytiedoston tarkoituksena on kuvata, miten projekti käännetään. Siinä on esitelty, mille kohdealustoille projekti voidaan kääntää ja mitkä projektimäärittelytiedostot käännetään. Projektimäärittelytiedosto määrittelee projektin kääntämisen alusta- ja kääntäjäkohtaisesti. Esimerkissä 1. on otos GnuPg:n projektimäärittelytiedoston tärkeimmistä kohdista sekä niiden selitykset.

```
TARGET              gpg.exe
TARGETTYPE          exe
UID                 0 0xEBC87F2D
EPOCSTACKSIZE       0x8000
CAPABILITY          ReadUserData WriteUserData

SYSTEMINCLUDE       \epoc32\include\stdapis
SYSTEMINCLUDE       \epoc32\include
USERINCLUDE         ..\inc

STATICLIBRARY       libcrt0.lib
LIBRARY             libc.lib
LIBRARY             libm.lib
LIBRARY             euser.lib
```

Esimerkki 1: GnuPg:n projektimäärittelytiedosto.

Esimerkissä projektimäärittelytiedosto alkaa projektin kohteen TARGET ja sen tyypin TARGETTYPE määrittelyllä. Koska kyseessä on suoritettava komentorivisovellus, annetaan tyypiksi exe. /9/

Seuraavaksi määritellään ohjelma UID (*Unique system identifier*), joka kertoo ohjelman toiminnasta sekä erottaa sen muista komponenteista järjestelmässä. UID on kolmeosainen, ja sen kaksi viimeistä osaa voidaan määritellä itse. UID1 määrittelee kohteen tyypin, tämä päätellään kohteen tyypin perusteella. UID2 antaa lisämäärittelyn kohteelle, tämä riippuu kohteen tyypistä. Esimerkiksi graafisella käyttöliittymällä varustetun ohjelman UID2 olisi 0x100039CE. Koska kyseessä on komentorivisovellus, on UID2:n arvoksi asetettu nolla. UID3 identifioi ohjelman koko käyttöjärjestelmän tasolla, ja sen on oltava uniikki. /9/

Oletusarvoisesti prosessille, jossa ohjelmaa ajetaan, on varattu 8 kilotavua muistia. Tämä ei kuitenkaan kaikissa tapauksissa ole GnuPg:lle riittävä määrä, joten määrittelyllä EPOCSTACKSIZE on muistin määrää kasvatettu 32 kilotavuun.

Määrä on ilmoitettu heksadesimaaleina (0x8000 hex => 32 768 dec). Suurin mahdollinen määrä on 80 kilotavua. /9/

Jotta ohjelma pystyy lukemaan ja kirjoittamaan omaan yksityiseen hakemistoonsa on sille annettava kapabiliteetti määrittelyllä CAPABILITY-oikeudet Read / WriteUserData. Kapabiliteetit ovat osa Symbian OS:n turvallisuusmallia, jota ei käydä tässä työssä lävitse. /9/

Jotta tarvittavat otsikkotiedostot saadaan käyttöön, kerrotaan määrittelyillä SYSTEMINCLUDE (käyttöjärjestelmän otsikkotiedostot) ja USERINCLUDE (ohjelman otsikkotiedostot) kansioiden polut, joista nämä löytyvät. /9/

Koska kyseessä ei ole Symbian C++-ohjelma eikä ohjelman tulokohtaa haluta muuttaa sen mukaiseksi, täytyy ensimmäisenä määritellä staattinen kirjasto libcrt0.lib STATICLIBRARY-määrittelyllä. Tällä tavoin ohjelman tulokohtaa ei tarvitse muuttaa Symbian C++:n mukaiseksi ja säästytään ohjelman turhalta muokkaamiselta. /9/

Viimeisenä liitetään tarvittavat kirjastot ohjelmaan määritteellä LIBRARY. Ensimmäisenä liitetään Open C:n yleinen kirjasto libc, sekä ohjelman tarvitsema matematiikka kirjasto libm. Viimeisenä liitetään Symbian OS:n euser kirjasto, jota monet Open C:n kirjastot tarvitsevat toimiakseen. /9/

4.3 Käyttö

GnuPg:tä käytetään kuten mitä tahansa komentoriviohjelmaa Symbian-ympäristössä. Tämä ei kuitenkaan ole aivan arkipäiväinen asia monelle Symbian ohjelmoijallekaan, joten tässä kappaleessa esitellään, kuinka se tapahtuu käytännössä. On myös huomioitava, että GnuPg:n käyttö edellyttää, että kohdelaitteelle on asennettuna Symbian OS PIPS -versio 1.3 tai uudempi.

GnuPg käynnistetään uuteen prosessiin toisen prosessin sisältä käyttämällä RProcess-kahvaluokkaa. Komentoriviparametrit annetaan RProcess-oliolle deskriptorina. Koska GnuPg ajetaan omassa prosessissaan asynkronisesti, sen päättymistä ei esimerkiksi voida suoraan päätellä. Tämän vuoksi RProcess-oliolle on annettava myös TRequestStatus-olio, jota seurataan joko aktiivisessa oliossa tai globaalin User-luokan staattisella metodilla WaitForRequest. Prosessin päätyttyä TRequestStatus-olio pitää sisällään GnuPg:n pääohjelman paluuarvon, tai tiedon siitä mikäli se on aiheuttanut jonkin laittoman toimenpiteen ja tästä syystä lopetettu.

```
RProcess process;
TRequestStatus status;

process.Create(_L(”gpg.exe”), _L(”--help”));  (1)
process.Logon(status);                        (2)
process.Resume();                             (3)
User::WaitForRequest(status);                 (4)
```

Esimerkki 2: GnuPg:n käyttö kooditasolla.

Esimerkissä 2 on esitetty GnuPg:n käytöstä Symbian C++-koodissa. Ensimmäisenä luodaan prosessi antamalla suoritettavan ohjelman koko nimi sekä komentoriviparametrit (kohta 1.). Tässä tapauksessa pyydetään GnuPg:tä tulostamaan ohjeet help parametrilla. Seuraavaksi asetetaan TRequestStatus-olio RProcessille (kohta 2.), jolloin voidaan seurata prosessin päättymistä. Prosessi käynnistetään (kohta 3.) ja jäädään odottelemaan sen suorituksen loppumista (kohta 4.).

Lista kaikista GnuPg:n komennoista sekä parametreista on saatavilla GnuPg:n dokumentaatiosta. /5/

5 MOBILEPGP ENGINE

MobilePGP Engine -kirjasto tarjoaa Symbian C++ -rajapinnan GnuPg:n käyttöön. Kirjasto sisältää moottorin ja sen apuluokkien lisäksi tietorakenteita operaatioiden tulosten sekä avainten käsittelyyn. Kirjastoa voidaankin pitää sovelluksen tärkeimpänä moduulina heti GnuPg:n jälkeen.

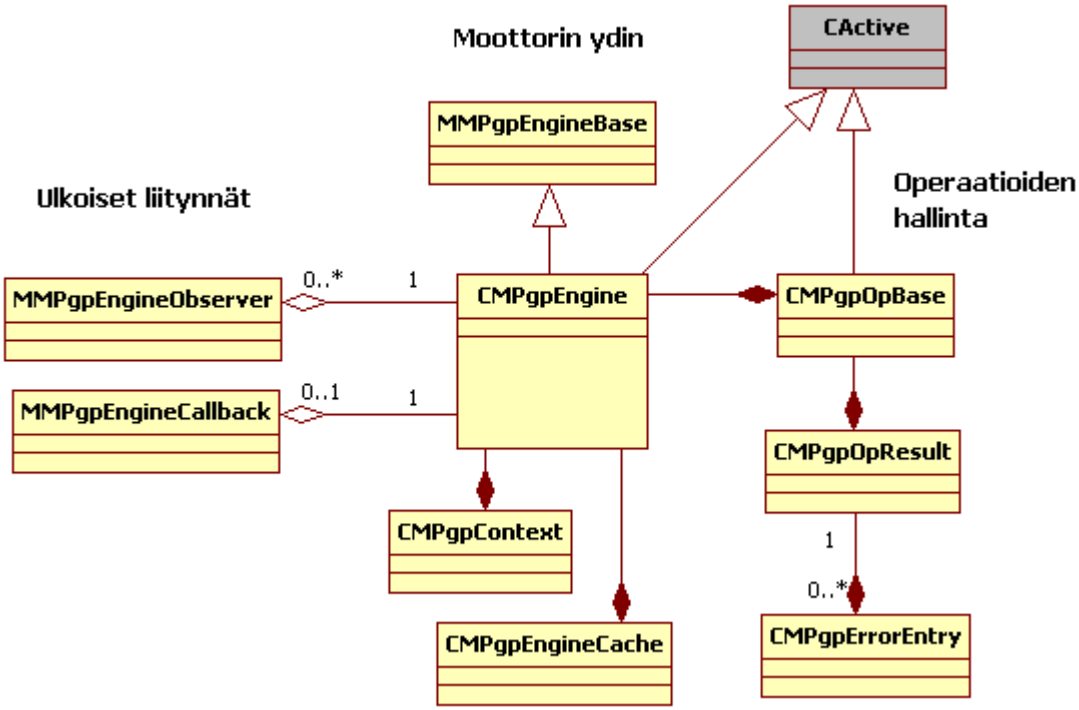
Moottorin rajapinnassa on esitelty yleisimmät PGP-operaatiot, kuten:

- Avainten ja avainrenkaiden tuonti
- Avainten poisto
- Avainten paikallinen allekirjoitus sekä luottamustasojen säätö
- Viestien ja tiedostojen salaus ja purku
- Allekirjoitusten luonti
- Allekirjoitusten tarkistus.

5.1 Moottorin arkkitehtuuri

MobilePGP Engine koostuu joukosta tiedon käsittelyyn, ohjaukseen ja tallennukseen tarkoitettuja luokkia. Itse moottorin ydin ei yksinään juurikaan suorita minkäänlaista tiedon prosessointia, vaan ohjaa tietoa niitä käsitteleville olioille. Moottorin sisäinen toteutus on asynkroninen sekä hajautettu osiin. Toteutuksessa on käytetty useampia suunnittelumalleja, kuten aktiivisia olioita, sekä adaptaatio-, tarkkailija-, sekä takaisinkutsumalleja.

Koska joidenkin kryptografisten operaatioiden suoritus saattaa kestää kauan (1… 3 sekuntia), kaikki PGP-operaatiot on toteutettu asynkronisesti. Tällä tavalla vältytään tilanteelta, jossa moottoria käyttävä sovellus olisi toimintakyvytön pitkän aikaa. Aktiiviset oliot antavat myös mahdollisuuden käyttäjälle keskeyttää operaatiot halutessaan. Moottorin luokat voidaan jakaa kolmeen osaan niiden luonteen perusteella: moottorin ydin, operaatioiden hallinta ja ulkoiset liitynnät. Moottorin yksinkertaistettu arkkitehtuuri luokkakaaviona on esitetty kuvassa 6.



Kuva 6: MobilePGP Enginen moottori.

Moottorin ydin koostuu kolmesta luokasta: *CMPgpEngine*, *CMPgpContext* ja *CMPgpEngineCache*. *CMPgpEngine* on moottorin ydin, joka tarjoaa PGP operaatiolta perityn *MMPgpEngineBase*-rajapinnan kautta. Se on myös aktiivinen olio, mikä mahdollistaa operaatioiden suorittamisen asynkronisesti. *CMPgpEngineContext* on luokka moottorin sisäisen toiminnan ohjaamiseen sekä asetusten hallintaan. *CMPgpEngineCache* toimii moottorin välimuistina, esimerkiksi avainten listauksille.

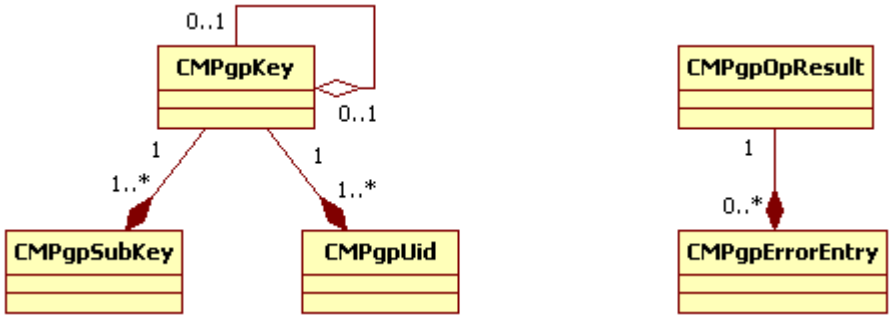
Operaatioiden hallinta koostuu *CMPgpOpBase* kantaluokasta periytetyistä operaatioluokista, joilla on oma sisäinen toteutus. Kaikki operaatioluokat ovat aktiivisia olioita, ja pystyvät käsittelemään tietoa pienissä osissa. Kantaluokka tarjoaa myös yleisiä metodeja operaation tulosten käsittelyyn. Tuloksista on kerrottu tarkemmin luvussa 5.2.

Ulkoisiin liityntöihin kuuluvat moottorin tarkkailija- ja takaisinkutsurajapinnat. Kaikki moottorin tarkkailijat perivät *MMPgpEngineObserver*-rajapinnan, jonka kautta moottori ilmoittaa omasta tilastaan. Takaisinkutsurajapinnan

MMPgpEngineCallback toteuttava luokka pystyy tarjoamaan moottorille operaatiokohtaisia lisätietoja, kuten esimerkiksi salasanan.

5.2 Tietorakenteet

MobilePGP Engine tarjoaa tiedon käsittelyyn myös joukon tietorakenteita. Näitä rakenteita ovat operaatioiden tulokset, virheet, sekä PGP-avaimet (kuva 7).



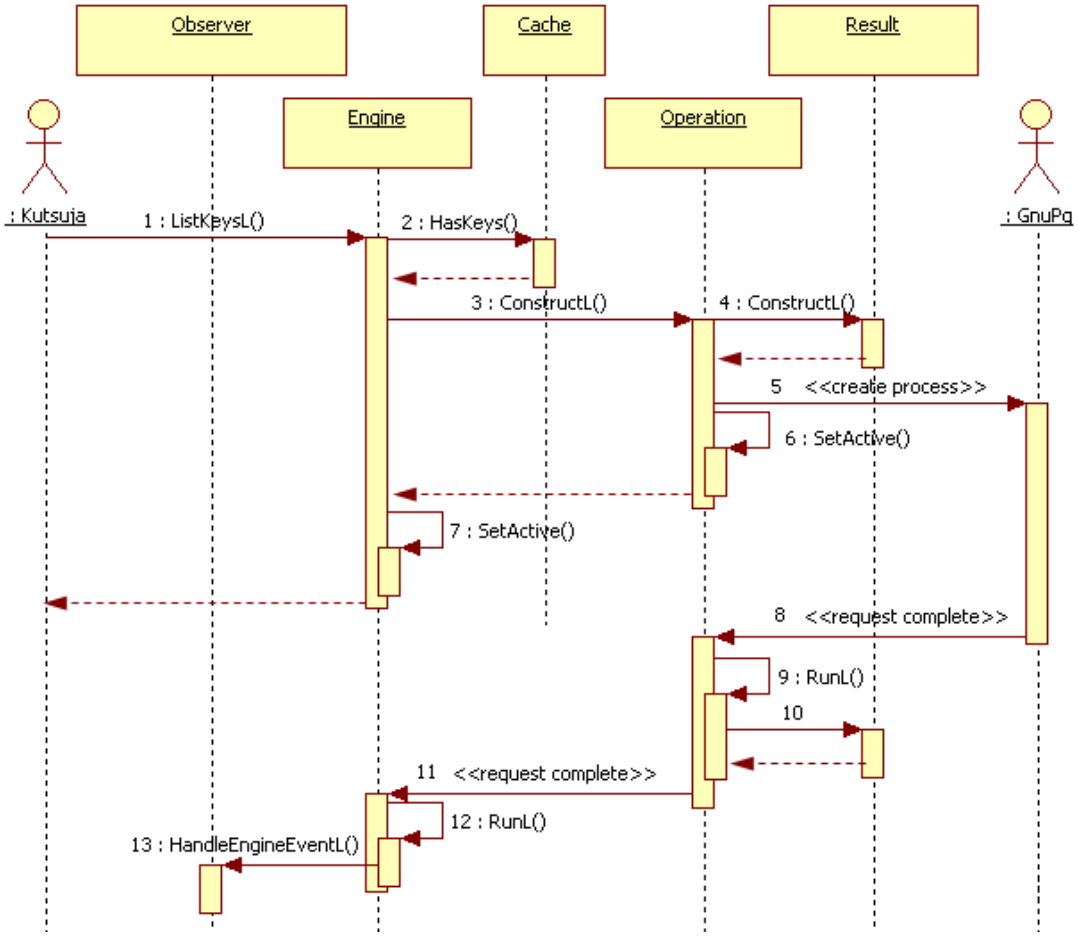
Kuva 7: MobilePGP Enginen tietorakenteita.

PGP-avain koostuu kolmesta luokasta: varsinaisesta avaimesta *CMPgpKey*, yhdestä tai useammasta aliavaimesta *CMPgpSubKey* sekä yhdestä tai useammasta avaimeen liitetystä käyttäjätiedosta *CMPgpUid*. Avaimella voi myös olla viittaus toiseen avaimeen, tätä käytetään liittämään yhteen julkinen ja salainen avain avainpariksi.

Jokaisen PGP-operaation tulokset saadaan *CMPgpOpResult*-pääluokasta periytetyiltä luokilta. Tulosten sisältö riippuu suoritetetusta operaatiosta. Esimerkiksi avainten listauksessa tulosluokka sisältää haetut avainrakenteet. Pääluokka hoitaa kaikille operaatioille yhteisten virhetilanteiden tallennuksen. Jokaista virhetilannetta varten luodaan uusi virheolio *CMPgpErrorEntry*, joka sisältää tarkemmat tiedot tapahtuneesta virheestä.

5.3 Moottorin toiminta

Käyttäjä kutsuu moottorin asynkronista palvelua, jolloin moottori aloittaa operaation alustuksen. Alustuksen aikana moottori tarkistaa välimuistista, pystyykö se tarjoamaan vastauksen palvelupyyntöön tai johonkin sen osaan ilman jatkotoimenpiteitä. Jotkut operaatiot saattavat myös vaatia lisäohjeita ennen aloitusta, tällöin moottori pyytää näitä tietoja takaisinkutsuoliolta. Kun alustus on suoritettu, moottori luo operaatiota vastaavan olion, kutsuu sen toisen vaiheen rakentajametodia ja asettuu itse odottamaan operaation valmistumista. Tämän jälkeen operaatio-olio aloittaa suorituksensa. Operaation suorituksen jälkeen operaatio-olio suorittaa loppuun moottorin asynkronisen pyynnön. Tämän jälkeen moottori ilmoittaa sille rekisteröidyille tarkkailijoille operaation valmistumisesta. Kuvassa 8 esitetään esimerkki avainlistauksen hakemisesta.



Kuva 8: Avainlistauksen hakeminen.

1. Kutsuja pyytää moottoria listaamaan halutut avaimet *ListKeysL()*-metodilla.
2. Moottori tarkistaa välimuistilta *HasKeys()*-metodilla, onko avaimet saatavilla siltä. Jos avaimet löytyvät välimuistista, hypätään suoraan kohtaan 7 ja suoritetaan moottorin asynkronin pyyntö itse loppuun, jolloin hypätään suoraan kohtaan 12.
3. Moottori luo ja alustaa operaatio-olion *ConstructL()*-metodilla.
4. Operaatio-olio luo ja alustaa tulosolion *ConstructL()*-metodilla.
5. Operaatio-olio käynnistää GnuPg:n uudessa prosessissa.
6. Operaatio-olio asettaa itsensä aktiiviseksi ja jää odottamaan GnuPg:n valmistumista *SetActive()*-metodilla.
7. Moottori asettaa itsensä aktiiviseksi ja jää odottamaan valmistumista.
8. GnuPg:n suoritus päättyy, jolloin operaatio-oliolle annetaan tästä tieto.
9. Operaatio-olion *RunL()*-metodi käynnistyy, jolloin suoritetaan tarvittavat operaatiot avainrakenteiden luontiin.
10. Avainrakenteet ja virheet tallennetaan tulosolioon.
11. Operaatio-olio signaloi moottoria.
12. Moottorin *RunL()*-metodi käynnistyy, jolloin tehdään tallennukset välimuistiin.
13. Moottori kutsuu tarkkailijan *HandleEngineEventL()*-metodia ja ilmoittaa operaation valmistumisesta.

6 MOBILEPGP EMAIL

MobilePGP Email -kirjasto tarjoaa rajapinnan sähköpostien käsittelyyn, joka tapahtuu viestintäarkkitehtuurin kautta. Se käyttää hyväkseen olemassa olevia sähköpostitilejä, koska se ei pysty luomaan tai muokkaamaan niitä. MobilePGP Email -kirjasto on luotu siitä syystä, että S60:n sähköpostiohjelmaa ei pystytty laajentamaan siten, että se olisi pystynyt käsittelemään PGP-salattuja viestejä.

6.1 Viestintäarkkitehtuuri /3/

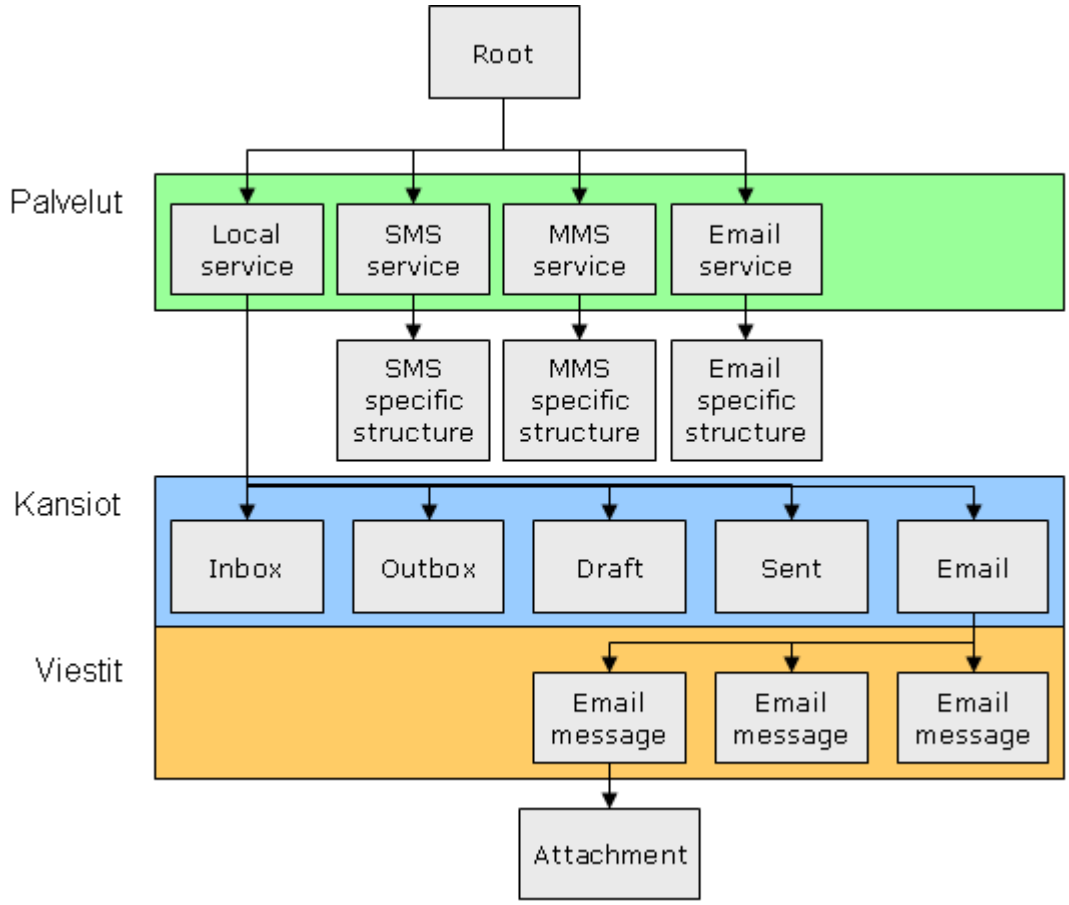
Viestintäarkkitehtuuri (*Messaging Architechture*) koostuu useammasta viestintään liittyvästä komponentista, kuten viestipalvelimesta (*Messaging Server*), viestintäkeskuksesta (*Messaging Centre*), lähetyspalvelimesta (*SendAs Server*) ja tarkkailijakehyksestä (*Watcher Framework*). Koska koko arkkitehtuuri on hyvin laaja, käydään tässä luvussa lyhyesti lävitse vain MobilePGP Emailia koskevia komponentteja.

Viestipalvelin

Viestipalvelin käsittelee ja hallinnoi viestiluetteloa (*Messaging Index*) sekä viestivarastoa (*Messaging Store*). Viestipalvelin tarjoaa myös yhteyden asiakaspalveluihin, kuten alkioiden seuraamiseen ja protokollakohtaisiin palveluihin. Viestipalvelinta käsitellään asiakaspuolella *CmsvSession*-luokan kautta. Muutoksia viestivarastossa voidaan seurata rekisteröimällä viestipalvelimelle *MmsvSessionObserver*-luokasta periytetty olio.

Viestiluettelo

Luettelo on rakenteeltaan hierarkkinen ja koostuu viestintäalkioista. Alkiot voivat olla esimerkiksi palveluita, kansioita, asetuksia ja viestejä. Esimerkki viestiluettelon rakenteesta on esitetty kuvassa 9. Viestipalvelin ei ota kantaa alkion sisältöön tai käsittelyyn, vaan ohjaa niille tulevat pyynnöt protokollakohtaisille moduuleille (*MTM, Message Type Module*).



Kuva 9: Viestivaraston hierarkkinen rakenne.

Jokaista alkiota mallinnetaan geneerisellä *TmsvEntry*-luokalla. Se sisältää yleisiä tietoja alkiosta, kuten uniikin tunnisteen, tyypin (kansio, viesti), protokollakohtaisen moduulin ja palvelun, johon alkio kuuluu.

Viestivarasto

Jokaiseen viestiluettelon alkioon liittyy tiedostovarasto, jota käytetään tallentamaan viestin pysyviä tietoja. Tällaisia tietoja ovat esimerkiksi viestin sisältö ja vastaanottaja. Varaston tietorakenne riippuu aina protokollakohtaisesta määrittelystä.

Protokollakohtaiset moduulit

Koska *TmsvEntry*-luokka on geneerinen kaikille alkioille, sillä ei voida käsitellä esimerkiksi sähköpostiviestin vastaanottajia ja viestiä. Jotta sähköpostiviestiä voidaan käsitellä, instantioidaan viestipalvelimen kautta viestintäkeskuksesta sille

tarkoitettu asiakaspään moduuli. Tämän moduulin kautta tehdään kaikki tarvittavat muutokset sähköpostiviestiin.

Lähetyspalvelin

Lähetyspalvelimen rajapinta antaa mille tahansa sovellukselle mahdollisuuden lähettää viestejä. Yhteys lähetyspalvelimeen luodaan *RsendAs*-kahvaluokan avulla. Luokka tarkistaa tarjolla olevat tilit, joilla viesti voidaan lähettää sekä varmistaa dynaamisesti, että sitä käyttävällä ohjelmalla on tarvittavat oikeudet viestin lähettämiseen. Lähetettävä viesti luodaan, muokataan ja lähetetään *RsendAsMessage*-luokan kautta. Luokka käyttää *RsendAs*-olion yhteyttä lähetyspalvelimeen lähettääkseen viestin.

6.2 Moottorin arkkitehtuuri

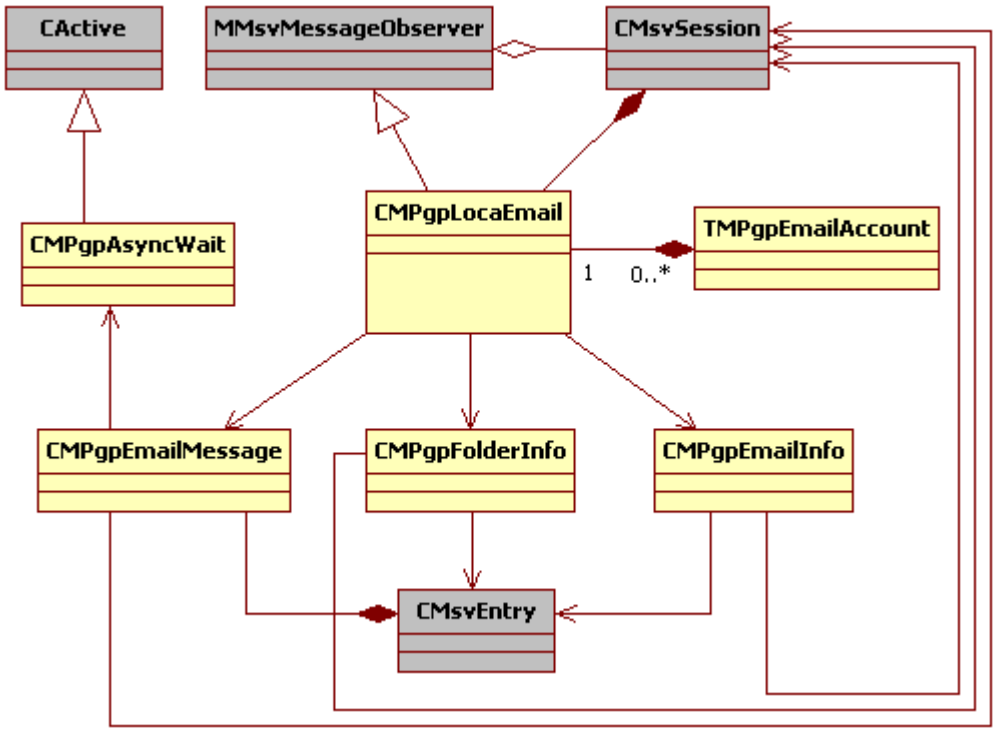
Sähköpostimoottorin toiminta on jaettu kahteen loogiseen osaan: paikalliseen- ja etäsähköpostin hallintaan. Etäsähköpostin hallinta on periytetty paikallisesta hallinnasta, ja se tuo näin uusia ominaisuuksia. Paikallisen sähköpostin hallinta on toteutettu synkronisesti, kun taas etäsähköpostin hallinta on täysin asynkronista.

MobilePGP Emailin rajapinnat tarjoavat seuraavan toiminnallisuuden:

- Sähköpostitilien sekä paikallisten kansioiden selaamisen.
- Viestien sekä kansioiden tietojen haun.
- Uusien viestien luomisen, joko vastauksena, edelleenlähetyksenä tai kokonaan uutena viestinä.
- Viestien sisällön muokkaamisen.
- Sähköpostien otsikkotietojen synkronoinnin etäpostipalvelimelta.
- Sähköpostien ja liitetiedostojen noutamisen etäpostipalvelimelta.
- Viestien lähettämisen.

6.2.1 Paikallinen sähköposti

Paikallinen sähköpostin hallinnan ydin on *CMPgpLocalEmail*-luokka. Se luo yhteyden viestintäpalvelimeen *CmsvSession*-luokan kautta, sekä asettaa itsensä tarkkailemaan viestivaraston muutoksia toteuttamalla *MmsvMessageObserver*-rajapinnan. Luokka pitää sisällään tiedon olemassaolevista sähköpostitileistä, joiden tiedot on tallennettu *TMPgpEmailAccount*-luokkiin. Kun haluttu sähköpostitili on valittu, moottori hakee viestintähakemistosta kyseisen tilin alla olevat kansiot ja viestit väliaikaiseen talteen, mikä mahdollistaa sähköpostitilin selailun. (Kuva 10.)



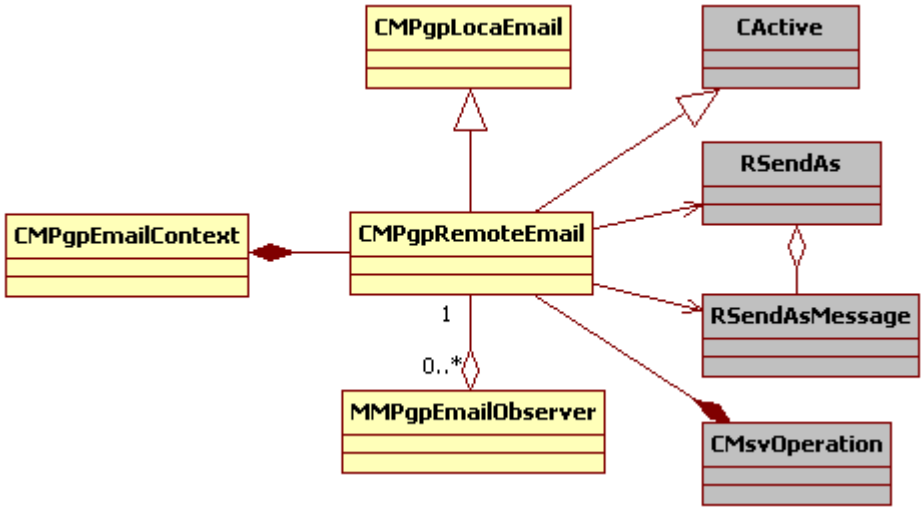
Kuva 10: Paikallisen sähköpostin moottorin arkkitehtuuri.

Moottori pystyy luomaan tietoluokkien ilmentymiä hakemiston soluista. *CMPgpFolderInfo*-luokka kertoo kaiken tarvittavan kansiosoluista ja *CMPgpEmailInfo*-luokka sähköpostiviestin otsikkotiedot. Moottori pystyy myös luomaan kokonaisen sähköpostiviestin kuvaamiseen tarkoitetun *CMPgpEmailMessage*-luokan ilmentymän, joko olemassaolevasta viestistä, tai luomalla uuden.

CMPgpAsyncWait-luokka on tarkoitettu käytettäväksi *CMPgpEmailMessage*-luokan asynkronisissa operaatioissa. Luokan avulla pystytään suorittamaan asynkronisia operaatioita synkronisesti ilman, että muun sovelluksen toiminta keskeytyy.

6.2.2 Etäsähköposti

Etäsähköpostin hallinnan ydin on *CMPgpRemoteEmail*-luokka. Se perii *CMPgpLocalEmail*-luokan ja tarjoaa näin laajan skaalan toiminnallisuutta. *CMPgpEngineContext*-luokka on tarkoitettu etäsähköpostimoottorin sisäisen toteutuksen ohjaamisen, sekä asetusten hallintaan.

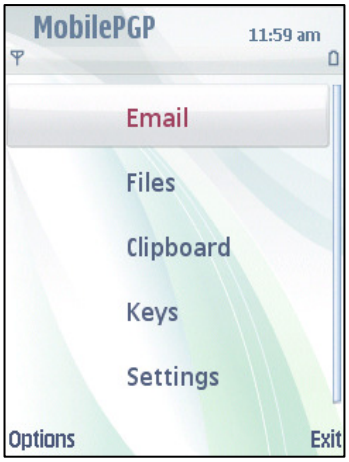


Kuva 11: Etäsähköpostin moottorin arkkitehtuuri.

Moottori pystyy suorittamaan sähköpostioperaatioita kuten synkronoinnin ja viestien hakemisen käyttämällä *CmsvOperation*-luokkia. Sähköpostien lähetys hoidetaan *RsendAs*- ja *RsendAsMessage*-luokkien avulla. Koska kaikki nämä operaatiot ovat asynkronisia, voidaan moottorille asettaa *MMPgpEmailObserver*-luokan toteuttavia tarkkailijoita. Moottori ilmoittaa tilastaan sekä operaatioiden suorituksen valmistumisesta näille tarkkailijoille. (Kuva 11.)

7 GRAAFINEN KÄYTTÖLIITTYMÄ

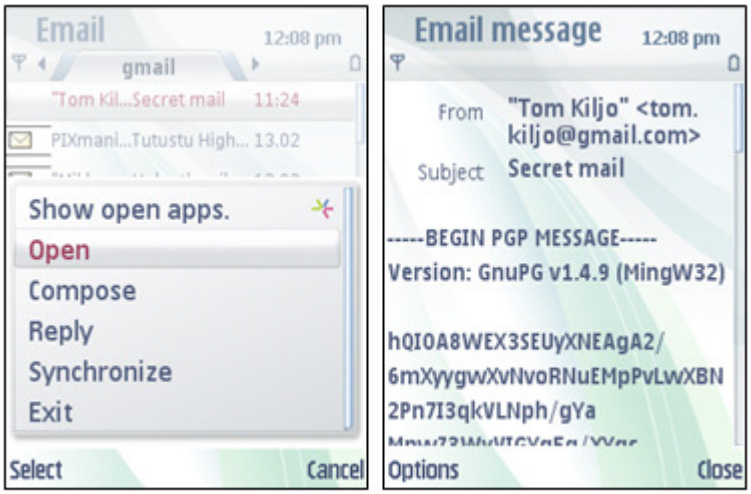
Graafisen käyttöliittymän suunnittelu mobiililaitteelle on varsin haastava tehtävä. Näytön koko ja hallintanäppäinten määrä on usein pieni. Käyttöliittymän suunnittelussa haluttiin säilyttää S60:lle ominainen ulkoasu käyttämällä sen tarjoamia valmiita komponentteja mahdollisimman paljon. Räätälöidyt komponentit luovat myös valtavan määrän lisätyötä, mikä saatiin myös huomata sähköpostitoimintoja tehtäessä. Ohjelman päävalikko on esitetty kuvassa 12. Seuraavaksi tarkastellaan hieman tarkemmin käyttöliittymän toiminnallisuutta.



Kuva 12: MobilePGP:n päävalikko.

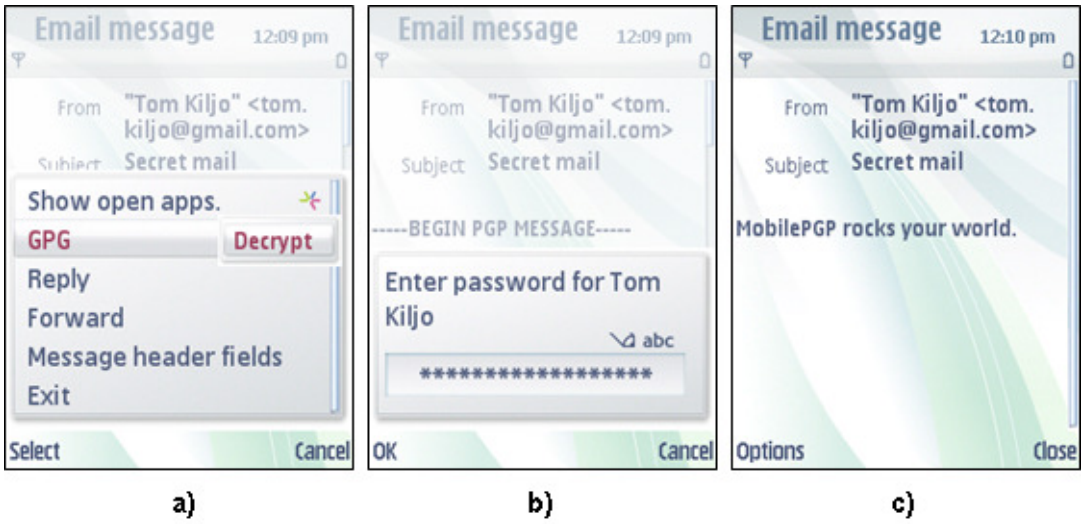
7.1 Sähköposti

MobilePGP sisältää oman sisäisen sähköpostiasiakasohjelman. Sähköpostinäkymään siirrytään päänäkymästä valitsemalla kohta Email. Näkymät ja hallinta ovat tässä komponentissa hyvin samanlaiset kuin S60:n viestiohjelmassa. Sähköpostitilejä ja paikallisia kansioita voidaan selata vasemmalla ja oikealla nuolinäppäimellä. Päänäkymän valikosta (kuva 13 kohta a) löytyvät komennot viestin ja kansioiden avaamiseen, uuden viestin kirjoittamiseen, viestiin vastaamiseen sekä tilin synkronointiin sähköpostipalvelimen kanssa. Myös viestin luku- ja muokkausnäkymä muistuttaa S60:n sähköpostiohjelman ulkonäköä ja toiminnallisuutta (kuva 13 kohta b).



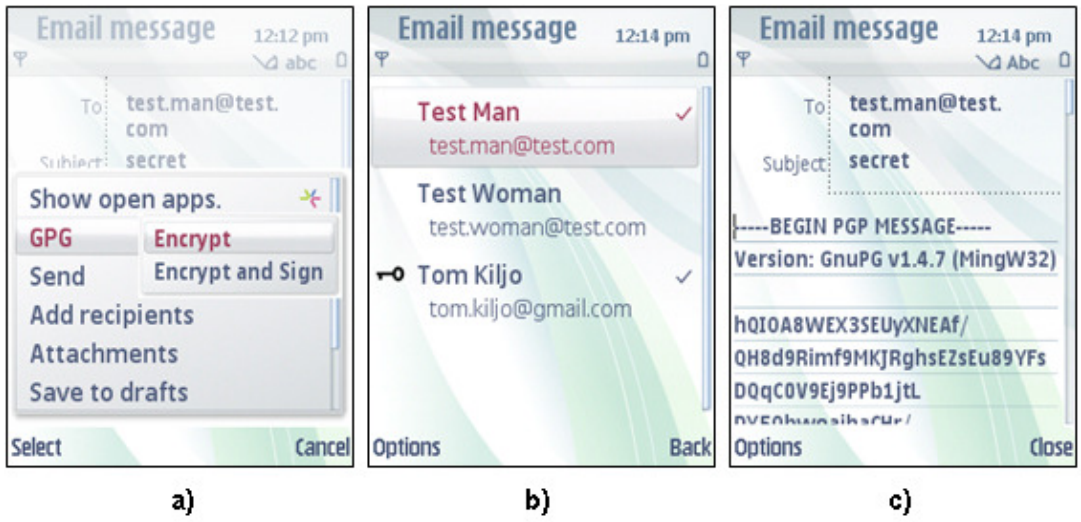
Kuva 13: Sähköpostin päänäkymät.

PGP-salattu sähköposti saadaan purettua avaamalle se lukunäkymään ja valitsemalla GPG alavalikosta komento Decrypt (kuva 14 kohta a). Ohjelma pyytää valitsemaan salaisen avaimen, jolla viesti puretaan, mikäli oletusavainta ei asetuksissa ole määritelty. Seuraavaksi ohjelma pyytää avaimen salasanaa (kuva 14 kohta b). Onnistuneen viestin purkamisen jälkeen viesti näkyy salaamattomana lukunäkymässä (kuva 14 kohta c).



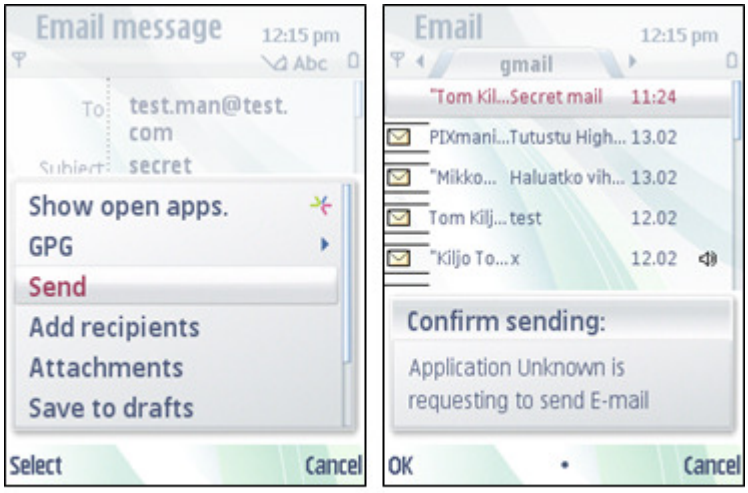
Kuva 14: Sähköpostin salauksen purkaminen.

Salattujen viestien luominen onnistuu valitsemalla sähköpostin päänäkymän valikosta komennon Compose. Tämä avaa muokkausnäkymän, jossa viesti voidaan kirjoittaa sekä lisätä sille vastaanottajat. Kun viesti on valmis, valitaan GPG-alavalikosta komento Encrypt tai Encrypt and Sign, mikäli viesti halutaan myös digitaalisesti allekirjoittaa (kuva 15 kohta a). Ohjelma näyttää seuraavaksi listauksen julkisista avaimista (kuva 15 kohta b). Tässä näkymässä valitaan ne avaimet, joille halutaan antaa oikeus purkaa salattu viesti. Onnistuneen salauksen jälkeen salattu viesti näkyy muokkausnäkymässä (kuva 15 kohta c).



Kuva 15: Sähköpostin salaaminen.

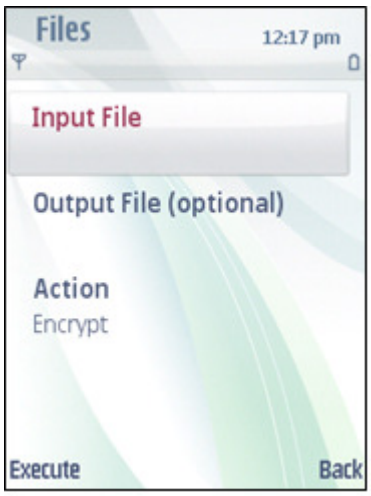
Luotu sähköposti voidaan tallentaa paikalliseen luonnos -kansioon valitsemalla muokkausnäkymän valikosta komento Save to drafts. Sähköposti voidaan myös lähettää valitsemalla valikosta komento Send (kuva 16 kohta a). Tämän jälkeen lähetyspalvelin pyytää käyttäjältä vahvistuksen sähköpostin lähettämiseen (kuva 16 kohta a).



Kuva 16: Sähköpostin lähettäminen.

7.2 Tiedostot

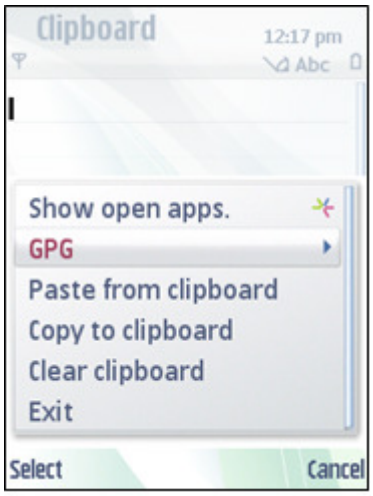
MobilePGP:llä voidaan hallinnoida myös yksittäisiä salattuja tai salaamattomia tiedostoja. Näkymään siirrytään päänäkymän kohdasta Files. Näkymä sisältää kolme asetusvaihtoehtoa: syötettävän tiedoston, vaihtoehtoisen ulostulotiedoston, sekä PGP-operaation (kuva 17). Esimerkiksi jos halutaan purkaa salattu tiedosto, valitaan se kohtaan Input File. Mikäli halutaan tallentaa purettu tiedosto toiseen kansioon tai toisella nimellä, täytetään kohta Output File. Viimeisenä valitaan PGP-operaatioksi salauksen purku eli Decrypt ja painetaan vasenta valintanäppäintä Execute. Ohjelma näyttää operaation tuloksen sen suorituksen jälkeen



Kuva 17: Tiedostojen hallinta.

7.3 Leikepöytä

MobilePGP:llä voidaan myös hallinnoida käyttöjärjestelmän leikepöytää. Tämä antaa käyttäjälle mahdollisuuden kopioida esimerkiksi salattuja viestejä leikepöydälle jossain muussa ohjelmassa ja purkaa ne sitten MobilePGP:llä. Näkymään siirrytään päänäkymän kohdasta Clipboard. Leikepöydän valikko sisältää komennot tekstin kopiointiin ja tuomiseen leikepöydältä, sekä sen tyhjentämiseen (kuva 18). Alavalikko GPG sisältää PGP-operaatiot, kuten tekstin salaamisen, salaamisen ja allekirjoittamisen sekä purkamisen.



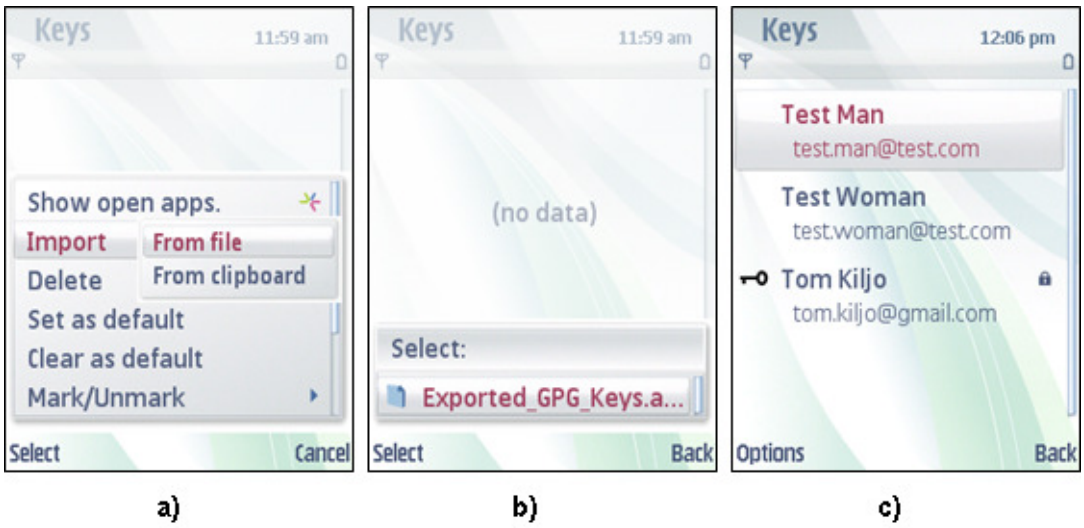
Kuva 18: Leikepöydän hallinta.

7.4 Avainten hallinta

Yksi tärkeimmistä ominaisuuksista on avainten ja avainrenkaan hallinnointi. Avainten hallintanäkymään siirrytään päänäkymän kohdasta Keys. Näkymän valikko tarjoaa komennot avainten tai avainrenkaiden tuontiin tiedostosta tai leikepöydältä, yhden tai useamman avaimen poistoon, oletusavaimen asettamiseen, avainten allekirjoittamiseen sekä avainten luottamustietojen muokkaamiseen.

Avaimet tai avainrenkaat tuodaan ohjelmalla valitsemalla näkymän alavalikosta Import, joko From file, jos tieto tuodaan tiedostosta tai From clipboard, jos tieto tuodaan leikepöydältä (kuva 19 kohta a). Jos avaimet tuodaan tiedostosta, valitaan se seuraavassa dialogissa (kuva 19 kohta b). Kun avainten tuonti on suoritettu, näytetään käyttäjälle vahvistusviesti, jossa kerrotaan, kuinka monta avainta tiedosto sisälsi ja kuinka monta niistä lisättiin avainrenkaaseen.

Avainrenkaassa olevat avaimet näkyvät listana näkymässä (kuva 19 kohta c). Avainsymboli avaimen vasemmalla puolella kertoo, että kyseessä on julkisen ja salaisen avaimen muodostama avainpari. Lukkosymboli avaimen oikealla puolella osoittaa oletusavainta.



Kuva 19: Avainten hallinta.

7.5 Asetukset

MobilePGP:n asetuksia hallitaan asetusnäkymästä. Näkymään siirrytään päänäkymän kohdasta Settings. Näkymä on jaettu kahteen osaan: PGP-asetuksiin ja sähköpostiasetuksiin. Asetusnäkymää voidaan vaihtaa vasemmalla ja oikealla nuolinäppäimellä.

PGP-asetuksiin kuuluvat oletusavain (Default key), oletusavaimen käyttö kaikissa PGP-operaatioissa (Always use default key), varmentamattomien avainten sallinta salausoperaatiossa (Always trust keys), salasanojen tallentaminen välimuistiin (Cache passwords) sekä salasanojen tallennusaika (Passwords expiration) (kuva 20 kohta a).

Sähköpostiasetuksissa voidaan määritellä, että kaikki sähköpostinäkymästä lähetetys viestit salataan (Encrypt outgoing messages) sekä allekirjoitetaan (Sign outgoing messages) automaattisesti (kuva 20 kohta b). Myös salattujen viestien purku voidaan automatisoida, kun viesti avataan sähköpostinäkymän lukunäkymässä (Decrypt automatically).

Kuva 20: Asetukset.

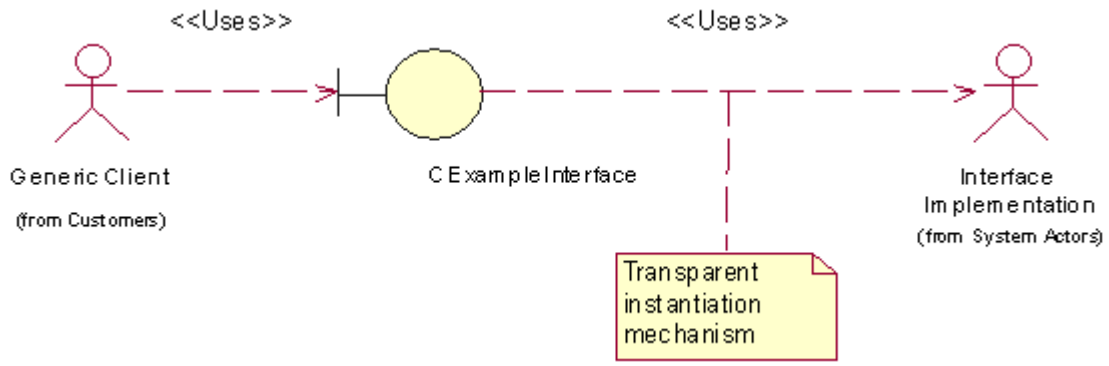
8 TIEDOSTONTYYPIN TUNNISTUSLIITÄNNÄINEN

Tiedoston tyypin tunnistusliitännäinen (*MIME recognizer plugin*) tuo MobilePGP:n käytön mukaan S60-käyttöliittymään, kuten tiedostojenhallintaan. Symbian OS -versio 9.1:stä eteenpäin tunnistusliitännäiset ovat ECom-liitännäisiä. Tässä kappaleessa esitellään ECom-liitännäisten arkkitehtuuri sekä tunnistusliitännäisen toiminta.

8.1 ECom-liitännäiset /1/

ECom tarjoaa kehyksen Symbian-ympäristöön, joka pitää sisällään yhtenäistetyn tavan liitännäisten rajapintojen rekisteröintiin ja etsimiseen sovelluskehyksestä, oikean toteutuksen valitsemisen sekä liitännäisten version hallinnan. Tämä poistaa polymorfisten dynaamisten kirjastojen käytöstä aiheutuneita toteutuksen päällekkäisyyksiä, jotka olivat väistämättömiä, koska jokainen liitännäinen joutui tarjoamaan oman toteutuksensa edellämainittujen prosessien suorittamiseen.

ECom toimii kuten mikä tahansa liitännäisjärjestelmä. Asiakas ottaa yhteyttä olioon tiedon prosessointia varten. Yleiset piirteet olion toiminnasta tiedetään, mutta tarkempi toteutus selviää vasta ajonaikaisesti.



Kuva 21: ECom-liitännäisarkkitehtuurin roolit.

Kuvassa 21 kuvataan ECom-arkkitehtuurin eri roolit. Rooleja ovat asiakasohjelma (*kuvassa Generic Client*), liitännäisen rajapinta (*kuvassa CExampleInterface*),

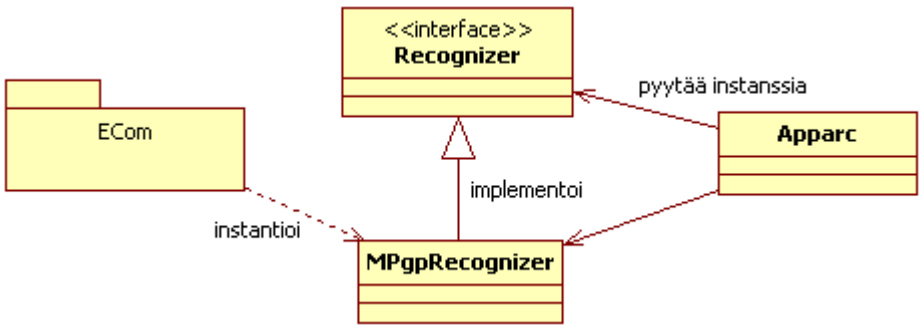
rajapinnan toteutus (*kuvassa Interface Implementation*) sekä mekanismi, joka tarjoaa asiakkaalle yhteyden rajapinnan toteutukseen (*kuvassa Transparent instantion mechanism*).

Käytännössä asiakas pyytää rajapinnan kautta suoritettavaksi jonkinasteista tiedon prosessointia, jolloin ECom-kehys etsii siihen parhaiten sopivan toteutuksen, lataa sen ajonaikaisesti sekä ohjaa kutsun ladatulle toteutukselle.

8.2 Toiminta

Symbianin tunnistusliitännäisten arkkitehtuuri mahdollistaa omien liitännäisten luonnin ja lisäämisen käyttöjärjestelmän tunnistusliitännäisten listaan. Tätä listaa pitää yllä käyttöjärjestelmän tunnistekehys. Sovellusarkkitehtuuri (*application architechture* tai *apparc*) lataa liitännäiset tunnistekehykselle käyttöjärjestelmän käynnistyksen yhteydessä. Listalla voi olla useita tunnistusliitännäisiä, jotka viittaavat samaan MIME-tyyppiin, joten ne on järjestetty hierarkkiseen listaan ohjelmoijien antamien prioriteettien mukaan.

Sovellusarkkitehtuuri pyytää tarvittaessa tiedon käsittelyyn tunnistekehykseltä MIME-tyyppiin viittaavan tunnisteliitännäisen ilmentymän (kuva 22). Mikäli liitännäisiä löytyy useampi kuin yksi tunnistekehyksen listalta, valitaan niistä suurimman prioriteetin omaava. Sovellusarkkitehtuuri käyttää tätä liitännäistä etsimään sopivaa sovellusta tiedoston käsittelyyn.



Kuva 22: MpgpRecognizer-olion ilmentymän luominen. /1/

9 TULOSTEN TARKASTELU

9.1 Tavoitteiden saavuttaminen

Tutkintotyölle asetetut tavoitteet täyttyivät. Työn tuloksena saatiin toteutettua Nokian S60-ympäristöön toimiva PGP-sovellus, jota voidaan käyttää salattujen sähköpostien ja liitetiedostojen purkamiseen ilman työasemaa. Työn aikana saatiin myös paljon arvokkaita kokemuksia erilaisista suunnittelumalleista, tekniikoista ja teknologioista Nokian S60- ja Symbian-ympäristössä.

Suurimpia haasteita asettivat GnuPG:n tuomisessa S60-ympäristöön dokumentaation puutteellisuudet sekä usein hyvin sekava lähdekoodi. Dokumentaation ajoittainen puutteellisuus osoittautui myös haasteeksi Symbian puolella, erityisesti viestintäarkkitehtuuria tutkittaessa ja MobilePGP:n viestintäratkaisuja tehtäessä. Myös kommunikointi eri paikkakunnalla toimivien käyttöliittymäohjelmoijien kanssa asetti omat haasteensa, mistä kuitenkin selvittiin hyvin pienen harjoittelun jälkeen.

Työ onnistuttiin toteuttamaan lähestulkoon aikataulussaan. Suunnitteluvaiheessa oli vielä hieman epäselvää, kuinka laajasta kokonaisuudesta todellisuudessa olikaan kysymys. Mielestäni työssä onnistuttiin kuitenkin hyvin, ottaen huomioon kuinka laaja toiminnallisuus saatiin toteutettua. Myös projektin johtamisesta ja toimimisesta hajautetuilla resursseilla saadusta kokemuksesta on varmasti hyötyä jatkossa.

9.2 Jatkokehitysajatuksia

Jatkossa MobilePGP:n kehitykseen voisi tuoda mukaan GnuPG:n kehittäjien luoman avoimen lähdekoodin kirjaston PGPME, joka on rajapinta GnuPG:tä käyttävien kolmansien osapuolten sovelluksille. PGPME:n käyttöä tutkittiin myös tämän työn aikana, mutta siitä jouduttiin toistaiseksi luopumaan teknisten haasteiden vuoksi.

Myös yhtenä jatkokehitysajatuksena olisi tutkia OpenPGP SDK:n käyttöä Symbian-ympäristössä. OpenPGP SDK on avoimeen lähdekoodiin perustuva projekti, jonka tarkoituksena on tarjota OpenPGP-standardiin perustuva kirjasto PGP-sovellusten kehittäjille. OpenPGP SDK ei ole valmis sovellus, joka tarjoaa avainten tai luottamusverkkojen hallintaa, kuten GnuPG, mutta se antaa mahdollisuuden luoda nämä toiminnallisuudet Symbian-ympäristössä. /10/

LÄHDELUETTELO

Painetut lähteet

1 Edwards, L., Barker, R., et al. 2004. Developing Series 60 Applications: A Guide for Symbian OS C++ Developers, 1. painos. USA, Addison-Wesley. 749 s.

2 Digia training team 2006. Symbian OS Active Objects. Tamepere, Digia 130 s.

3 Digia training team 2007. Symbian OS Communication Programming Tampere, Digia. 220 s.

Sähköiset lähteet

4 Gnu Privacy Handbook [www-sivu]. [viitattu 20.9.2008]. Saatavissa: http://gnupg.org/gph/en/manual.html.

5 Gnu Privacy Documentation [www-sivu]. [viitattu 22.3.2008]. Saatavissa: http://www.gnupg.org/documentation/index.en.html

6 Wikipedia [www-sivu]. [viitattu 19.9.2008]. Saatavissa: http://en.wikipedia.org/wiki/Pretty_Good_Privacy.

7 Callas, J. 2007. RFC 4880: OpenPGP Message Format [www-sivu]. [viitattu 26.9.2008]. Saatavissa: http://www.ietf.org/rfc/rfc4880.txt.

8 S60 3rd Edition C++ Developer's Library [www-sivu]. [viitattu 10.3.2009]. Saatavissa: http://www.forum.nokia.com/.

9 Symbian OS 9.1 [www-sivu]. [viitattu 10.3.2009]. Saatavissa: http://www.symbian.com/developer/techlib/v9.1docs/doc_source/index.html

10 OpenPGP SDK [www-sivu]. [viitattu 19.3.2009]. Saatavissa: http://openpgp.nominet.org.uk/cgi-bin/trac.cgi.