TAMPEREEN AMMATTIKORKEAKOULU
Tietotekniikan koulutusohjelma
Ohjelmistotekniikan suuntautumisvaihtoehto

Tutkintotyö

Jussi Suojanen

Matkapuhelimen näytön virransäästömenetelmät

Työn valvoja: Jari Mikkolainen

Työn teettäjä: Nokia Oyj, Juhani Mättö

Tampere 2007

TAMPEREEN AMMATTIKORKEAKOULU
Tietotekniikka, ohjelmistotekniikka
Jussi Suojanen

Suojanen, Jussi — Matkapuhelimen näytön virransäästömenetelmät
Tutkintotyö — 38 sivua
Työn valvoja — Jari Mikkolainen
Työn teettäjä — Nokia Oyj, Juhani Mättö
Huhtikuu 2007
Hakusanat — Matkapuhelimet, näyttötekniikka, Symbian OS, Series60

TIIVISTELMÄ

Tässä työssä esitellään matkapuhelimen näyttöön liittyviä eri virransäästömenetelmiä. Menetelmistä käydään läpi logiikat jolla virtaa säästetään, sekä mitä mahdollisia hyötyjä ja haittoja niistä on esimerkiksi puhelimen käytettävyydelle. Esiteltävät menetelmät ovat: himmennys, partial mode, näytön sammutus, led-tekniikka sammutuksen rinnalla ja ambient light sensor. Lisäksi tutustutaan tekniikoihin näiden toimintojen taustalla. Käsiteltäviin tekniikoihin kuuluvat mm. light targetit, joilla hoidetaan laitteen ledien käsittely sekä central repository, joka on Symbian käyttöjärjestelmän tarjoama rajapinta laitteen asetusten käsittelyyn.

Edellä mainittujen asioiden lisäksi työssä esitellään vielä sleep mode / power save led-sovellus. Käytännössä tekniikka tarkoittaa näytön sammutusta ja tämän tukena toimivaa led-tekniikkaa, jolla poistetaan osa käytettävyysongelmista. Työssä esitellään sovelluksen käyttäjälle näkyvä osa, asetusmoduuli, jonka avulla käyttäjä voi valita power save ledin käyttöön tai pois käytöstä ja pinnan alla tapahtuvat toiminnot eli itse sleep mode -moduuli. Sleep mode -moduuli huolehtii näytön sammutuksesta ja ledin aktivoinnista.

TAMPERE POLYTECNIC UNIVERCITY OF APPLIED SCIENCES
Computer technology, software engineering
Jussi Suojanen

Suojanen, Jussi	Power save methods of the mobile phones display device
Engineering Thesis	38 pages
Thesis supervisor	Jari Mikkolainen
Comissioning company	Nokia Oyj, Juhani Mättö
April 2007	
Keywords	mobile phone, display technology, Symbian OS, Series 60

ABSTRACT

This thesis presents some of the power save methods that are used in mobile phones display devices. About these methods we will go through the logic of how the power is saved, and also the usability problems that might occur because of these techniques. Methods that are covered here are: dimming, partial mode, turning the display off completely, power save led alongside with the displays turn off and ambient light sensor.

Furthermore we will get to know some of the techniques that are used inside the device to control these power save methods. In this list are included among others light targets, that are used to control the led lights in the device, and Central repository, that is programming interface provided by Symbian OS for handling the settings.

We will also concentrate on sleep mode / power save led application. In practice it is an application that turns off the display and activates the power save led that is used to reduce the usability problems. Both the settings user interface and the sleep mode module that is not visible to the user will be handled here. From the settings user interface user can change the power save led to be “on” or “off” when sleep mode is activated and the sleep mode module basically turns off the display and activates the led.

ALKUSANAT

Tämä insinöörityö on tehty keväällä 2007. Työ on jaettu kolmeen osa-alueeseen: näytön virransäästömenetelmät, niiden taustalla toimivat tekniikat ja sleep mode/power save led -sovellus.

Tässä yhteydessä tahdon kiittää Juhani Mättöä työn ohjauksesta ja tiedon hankkimisesta sekä Heli Järventietä lukuisien käyttäjätutkimusten materiaaleista ja niiden analysoinnista insinöörikielelle.

Tampereella 30. huhtikuuta 2007

_____

Jussi Suojanen

SISÄLLYSLUETTELO

KÄYTETYT TERMIT JA LYHENTEET

Ambien light sensor	Sensori, jolla mitataan ympäristön valomäärää.
API	Application Programming Interface, ohjelmointirajapinta.
Descriptor / TDesC	Symbian käyttöjärjestelmän tapa käsitellä stringejä/merkkijonoja.
Kapabiliteetti	Kapabiliteeteilla määritetään, mitä oikeuksia ohjelmalla on käyttöjärjestelmässä. Ohjelma tarvitsee esimerkiksi ”WriteDeviceData” -kapabiliteetin, että sillä on oikeus tallentaa tietoja. Kapabiliteetit määritetään sovelluksen ”.mmp” -tiedostossa.
Käyttäjä	Matkapuhelimen käyttäjä.
Laite	Matkapuhelin.
Led	Light-Emitting diode, puolijohdekomponentti, joka säteilee valoa, kun siihen johdetaan sähkövirta.
Luokka	Olio-ohjelmoinnin määrittely, joka kapseloi tietyn toiminnallisuuden helposti ymmärrettäväksi kokonaisuudeksi.
Partial mode	Virransäästötekniikka, jossa käytetään vähemmän värejä ja mahdollisesti pienempää piirtoaluetta puhelimen näytössä.
TReal luku	Desimaaliluku.
TInt luku	Kokonaisluku.
SID-numero	Secure IDentification, turvallisuus tunnistenumero.
Symbian OS	Matkapuhelimen käyttöjärjestelmä.
UID-numero	Unique IDentification, uniikki tunnistenumero.
Unsigned integer	Positiivinen kokonaisluku.

1 JOHDANTO

Työn tavoitteena on tutkia ja esitellä matkapuhelinten näytöissä virransäästöön käytettäviä eri menetelmiä. Työn edetessä näiden hyödyt ja mahdolliset haitat tulevat tutuiksi myös allekirjoittaneelle.

Työssä tutustutaan tarkemmin ”sleep mode” / ”power save led”-tekniikkaan, joka käytännössä tarkoittaa laitteen näytön sammutusta ja vilkkuvan ledin aktivointia. Tehtäväni oli ohjelmoida kyseinen sovellus asiakkaan vaatimusten mukaisesti. Ohjelmasta käydään läpi sekä asetusten käyttöliittymä että itse toiminnallisuuspuoli.

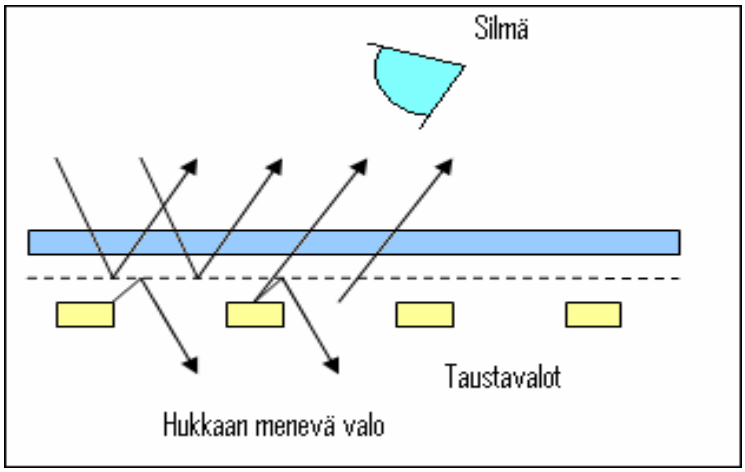
Tekniikasta selvitetään eri moduulien toimivuudet ja riippuvuudet, sekä julkaistaan luokkakaaviot. Lisäksi selitetään eri luokkien tehtävät, ja käydään läpi tärkeimpien funktioiden toiminnot. Työn tuloksena syntyy koulutus- ja perehdyttämismateriaali yrityksen käyttöön.

2 ERILAISET NÄYTTÖTYYPIT

Tässä kappaleessa esitellään yleisimmät matkapuhelinten näyttötyypit. Tarkoituksena on vertailla ja esitellä eri näyttöjen tekniikoita, sekä niiden mahdollisia hyötyjä ja haittoja. Esiteltävät näyttötyypit ovat: transflektiivinen (Transflective), emissiivinen (Emissive) ja transmissiivinen (Transmissive)/4/.

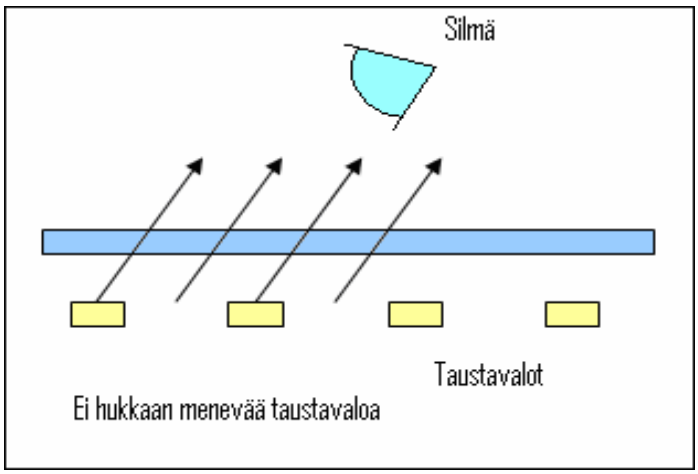
Tansflektiivinen näyttö (kuva 1) on yleisin näyttötyyppi. Siinä kennon alapuolelle on sijoitettu heijastava kalvo, jossa on taustavalon läpi päästäviä reikiä. Taustavaloina toimivat ledit ja lisäksi näytössä käytetään hyväksi ympäristöstä tulevaa valoa. Näyttöön tuleva valo osuu näytön alla olevaan heijastavaan kalvoon,

josta se heijastuu takaisin ja saa näytön kirkkaammaksi. Kirkkaassa päivänvalossa näyttö toimii vähemmällä virralla, koska heijastava kalvo käyttää hyväkseen ympäristön kirkkautta. Osa ledienkin valosta heijastuu kalvosta takaisinpäin, joten kalvo syö myös taustaledien tehoa. Heijastava kalvo pystyy heijastamaan ympäristön valoa sitä paremmin, mitä pienempiä kalvossa olevat reiät ovat, mutta toisaalta taustaledit toimivat sitä tehokkaammin, mitä suurempia kalvossa olevat reiät ovat. Mikäli reikiä ei ole ollenkaan, puhutaan reflektiivisestä (reflective) näytöstä, mutta tätä näyttötyyppiä ei juurikaan käytetä matkapuhelimissa.



Kuva 1 Transflektiivinen näyttö

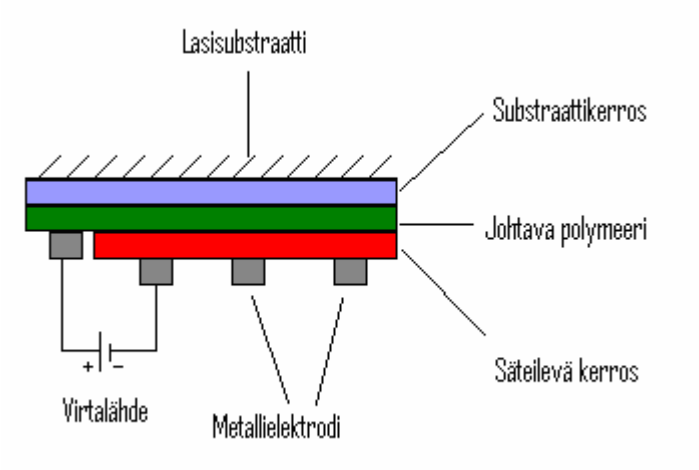
Transmissiivinen näyttö (kuva 2) on muuten samanlainen kuin transflektiivinen, mutta siinä ei ole valoa takaisin heijastavaa kalvoa. Näin ollen näytön valaisu on täysin taustaledien varassa. Heijastavan kalvon puuttuminen parantaa ledien hyötysuhdetta transflektiiviseen näyttöön verrattuna, joten esimerkiksi pimeässä (kun ympäristön valo puuttuu) ledejä voidaan käyttää pienemmällä teholla transflektiiviseen näyttöön verrattuna.



Kuva 2 Transmissiivinen näyttö

Vastaavasti kirkkaissa olosuhteissa transmissiivinen näyttö ei voi käyttää ympäristön valoa hyväkseen, joten taustavaloja on käytettävä suuremmalla teholla, mikä lisää virrankulutusta.

Emissiivisessä näytössä (kuva 3) ei ole taustavaloa eikä heijastavaa kalvoa, vaan jokainen yksittäinen pikseli säteilee itsessään valoa. Jokainen pikseli voidaan lisäksi kytkeä päälle tai pois, ja näin ollen virrankulutus on suoraan verrannollinen käytössä olevien pikseleiden lukumäärään.



Kuva 3 Emissiivisen näytön yksittäisen pikselin rakenne

3 ERILAISET VIRRANSÄÄSTÖMENETELMÄT /1/

Näytön virrankulutusta voidaan säädellä usealla eri menetelmällä. Useimmat niistä aktivoituvat puhelimen oltua jonkin aikaa käyttämättömänä. Aktivoitumisaika on hyvin pitkälle puhelinkohtainen ja riippuu mm. näytön virrankulutuksesta ja puhelimen käyttötarkoituksesta. Mikäli käyttäjä ei tarvitse puhelinta, virtaa ei kannata kuluttaa piirtämällä näytölle kaikki yksityiskohdat tai käyttämällä taustavaloja täydellä teholla. Tässä työssä käsiteltävät puhelimen käyttämättömyyteen perustuvat virransäästömenetelmät ovat: himmennys, partial mode, näytön sammutus ja viimeksi mainitun rinnalla käytettävä led-tekniikka.

Myös ympäristön valomäärää voidaan käyttää hyödyksi virrankulutuksen vähentämiseksi. Tähän perustuu mm. ambient light sensor -menetelmä. Sensorilla mitataan ympäristön kirkkautta ja näytön kirkkaus säädetään saatujen tulosten perusteella. Käyttäjä tarvitsee puhelintaan niin kirkkaissa kuin hämärissäkin olosuhteissa ja näissä eri olosuhteissa on näytön valontarve erilainen.

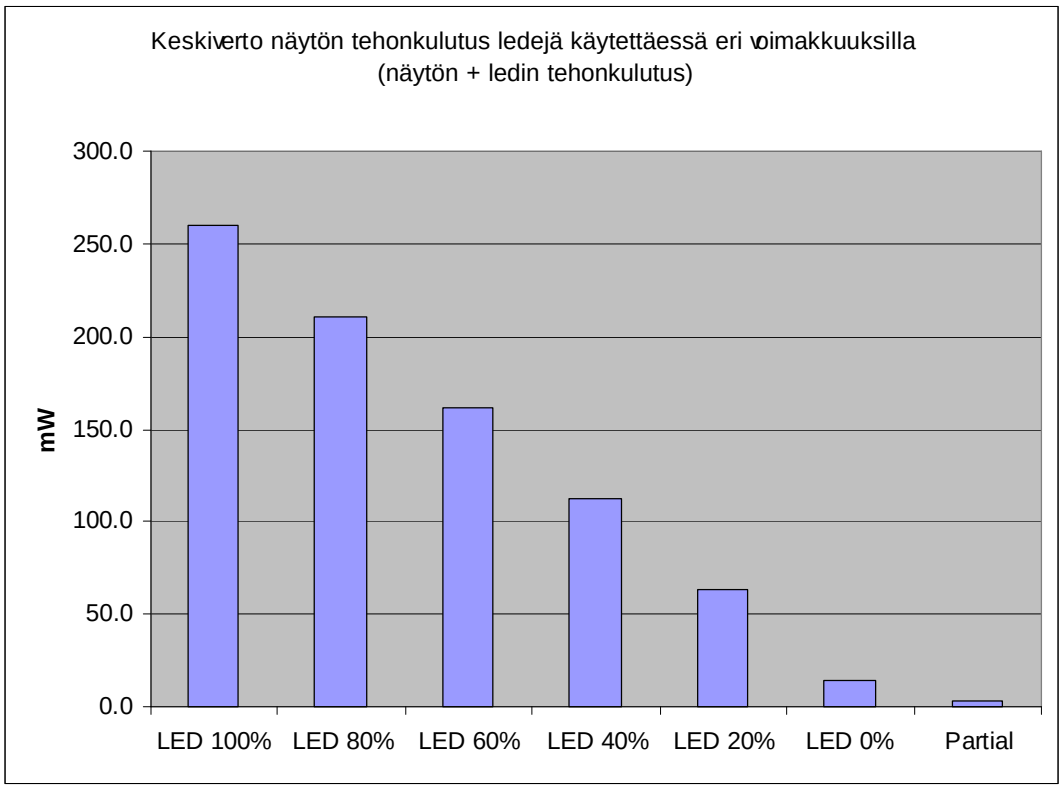
3.1 Himmennys

Himmennyksellä tarkoitetaan näytön taustavalojen läpi kulkevan virran vähentämistä. Tätä menetelmää voidaan käyttää, kun laite on ollut tietyn ajan käyttämättä. Himmennys tapahtuu yleensä vaiheittain. Esimerkiksi ensimmäisen viidentoista sekunnin jälkeen taustavaloja himmennetään kolmannekseen kirkkaudestaan ja tämän jälkeen, kun aikaa on kulunut yksi minuutti laitteen edellisestä toiminnasta, ne voidaan sammuttaa kokonaan.

Eri näyttötyypeillä himmennys toimii hieman eri lailla. Joillain heijastamattomilla näyttötyypeillä ei taustavaloa sammuteta kokonaan, vaan ledeihin jätetään erittäin pieni (5%) virta. Mikäli näin ei tehtäisi, ei näytöstä näkyisi mitään ja käyttäjä

saattaisi luulla laitteen sammuneen. Transflektiivisen näytön osalta taustavalo ei kirkkaissa olosuhteissa välttämättä ole tarpeellinen, koska näyttö heijastaa ympäristöstä tulevaa valoa takaisin. Tämän valon avulla näytössä näkyvät siihen piirretyt kuvat, eikä näyttöön tarvita ledien luomaa lisävalaistusta.

Himmennystekniikassa on tärkeää säätää aikajaksot niin, ettei käyttäjä ärsyynny näytön himmenemisestä ja sytytä näppäintä painamalla valoa uudelleen päälle. Tällöin taustavalot palavat taas täydellä teholla, ja halutun virransäästön sijaan kulutetaankin virtaa. Toisessa äärilaidassa näytön taustavalot palavat aina käytön jälkeen turhina liian pitkään. Monissa laitteissa käyttäjä voi säätää näytön taustavalon palamisajan mieleisekseen. Tämä ei välttämättä ole laitteen virrankulutukselle optimaalinen vaihtoehto, mutta laitteen käyttömukavuutta se lisää.



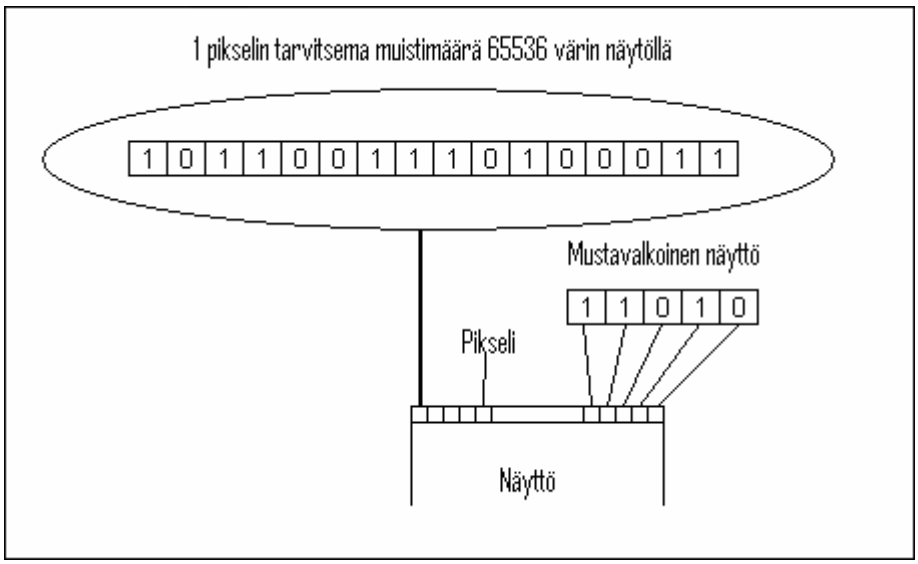
Kuva 4 Näytön tehonkulutus ledin eri voimakkuuksilla ja partial modessa /4/

Hyvän mielikuvan näytön virrankulutuksesta ja taustavalojen osuudesta siihen saa edellisellä sivulla olevasta kuvasta (kuva 4). Taulukkoon on laskettu runsaan 20 erilaisen näytön virrankulutusten keskiarvot sekä taustavalojen palaessa täydellä teholla että osittain niitä himmennettäessä. Ledejä käytettäessä täydellä teholla on näytön virrankulutus n. 260mW. Kun tätä arvoa verrataan 15mW:iin jonka näyttö kuluttaa ledien ollessa pois päältä, huomataan ledeillä valaisun muodostavan suuren osan näytön virrankulutuksesta. Tämän lisäksi kuvasta on hyvä panna merkille näytön virrankulutus taustavalojen ollessa pois päältä, verrattuna tilanteeseen, kun näyttö on partial modessa. Partial mode on tekniikka, jossa käytetään hyväksi pienempää piirtoaluetta ja kapeampaa väriskaalaa näyttöön piirrettäessä.

3.2 Partial mode

Partial mode aktivoituu yleensä himmennyksen jälkeen, kun laite on ollut käyttämättä yli minuutin. Tässä tekniikassa virransäästö perustuu pienemmän värimäärän käyttöön ja tarvittavan näyttöpinta-alan pienempään kokoon.

Värien määrän vähentämisessä virransäästö tapahtuu säästämällä laitteen muistia. Pienemmän värimäärän kontrollointiin tarvitaan pienempi määrä muistia, jolloin osa muistia voidaan ”sammuttaa”. Jos näytössä on normaalisti käytössä esimerkiksi 65536 väriä, tarvitaan yhden pikselin piirtämiseen yhteensä 16 bittiä muistia (2^16= 65536). Laitteen siirtyessä partial modeen käytössä voivat esimerkiksi olla vain musta ja valkoinen. Nämä värit pystytään ilmoittamaan yhdellä bitillä (1 tai 0), ja tällöin tarvittava muistimäärä on pienempi. Esimerkiksi kun pienennetään näytön värit 65536 kahteen väriin, pienenee muistin tarve yhteen kuudestoistaosaan (kuva 5).



Kuva 5 Yhden pikselin muistitarve 65536 värin näytöllä ja mustavalkoisella näytöllä

Partial modessa näytöstä tarvitaan käyttöön vain pieni osa (kuva 6). Pienemmällä piirtoalueella on käytössä vähemmän pikseleitä, jolloin muistin tarve vähenee. Emissiivisessä näytössä pikseleiden aktivointiin tarvittava virtamäärä pienenee, koska aktiivisten pikseleiden määrä pienenee. Osanäytön käytöllä on myös näyttöä säästävä ominaisuus. Näytön ollessa lepotilassa, eli kun siihen ei johdeta virtaa kuvan piirtämistä varten, se ei kulu. Vastaavasti kun näyttö on käytössä ja virtaa tarvitaan kuvan muodostukseen, näyttö kuluu. Partial modessa näyttö tarvitsee virtaa ainoastaan aktiiviseen osaan. Tällöin käytössä voi olla vain pieni alue, johon voidaan piirtää tarvittavat tiedot. Näitä tietoja voivat esimerkiksi olla päivämäärä, kello sekä saapuneet puhelut ja viestit. Koko muu näytön alue on tällöin lepotilassa, eikä siis kulu. Kuvan piirtokohtaa vaihtelemalla saadaan koko näytön ikää lisättyä. Lisäksi liikkuvan partial moden käyttö estää niin sanottua näytön palamista. Pitkään yhdessä paikassa oleva kuva kuluttaa näyttöä niin, että kuva ikään kuin syöpyy ruutuun.

Kuva 6 N95 partial mode

3.3 Sammuttaminen

Näytön sammuttaminen on tehokkain tapa säästää virtaa. Sammuttaminen tapahtuu yleensä partial moden jälkeen, mikäli tämä ominaisuus on puhelimessa käytössä. Sammuttamiseen liittyy useita käytettävyysongelmia. Suurimpana ongelmana on epätietoisuus laitteen tilasta ja kunnosta. Mistä käyttäjä tietää, että puhelin on päällä, mikäli näytössä ei näy mitään? Tässä tilanteessa puhelin saatetaan tahattomasti sammuttaa virtanäppäimestä, koska sitä yritetään tarpeettomasti käynnistää uudelleen. Osalle käyttäjistä täysin pimeä ja elottoman näköinen näyttö saa koko laitteen näyttämään hajonneelta /3/.

Lisäksi puhelimen näytöstä halutaan yhdellä vilkaisulla nähdä erilaisia tietoja. Tärkeimpinä näistä ovat päivämäärä, kellonaika, saapuneet puhelut, viestit, sähköpostit, puhelimen tila (kokous, yleinen), akun tila, verkon signaalin vahvuus, itse asetettu taustakuva ja asetettu herätys tai hälytys /2/.

Osalla käyttäjistä puhelin korvaa kellon. Heille on tärkeää, että puhelimesta heti sen esille otettuaan näkee kellonajan ja päivämäärän /2/. Tämä ei onnistu, jos puhelimen näyttö tulee ensin herättää, jotta halutut tiedot tulevat näkyviin.

Käyttäjälle on myös tärkeää tietää, onko joku yrittänyt ottaa häneen yhteyttä. Tämän takia puhelimesta tulisi nähdä saapuneet puhelut, viestit ja sähköpostit. Luonnollisesti tämäkään ei ole mahdollista sammutetusta näytöstä.

Mikäli käyttäjä odottaa saapuvaa puhelua, hänelle on tärkeää tarkistaa puhelimen tila. Tämä tarkoittaa käytännössä, että puhelin ei ole unohtunut äänettömälle, puhelimen akun tila on riittävä, ja verkon signaali on tarpeeksi vahva /2/.

Ylipäänsä käyttäjää ärsyttää, mikäli hänen täytyy aina itse tehdä jotain nähdäkseen puhelimesta haluamaansa tietoa /3/. Tämä taas pahimmillaan tarkoittaa sitä, että puhelimen näyttö herätetään jatkuvasti uudelleen. Myös näytön sytyttäminen kuluttaa virtaa ja pahimmillaan aggressiivinen käyttäjä onnistuu kuluttamaan virtaa toivotun säästön sijaan.

Kaiken edellä mainitun lisäksi käyttäjä ei myöskään halua uuden puhelimen hankittuaan sen näyttävän elottomalta. Vaikka käyttäjä tietääkin että puhelin on päällä näytön ollessa sammuneena, hän ei halua sen näyttävän toimimattomalta, esimerkiksi esitellessään sitä ystävilleen /3/.

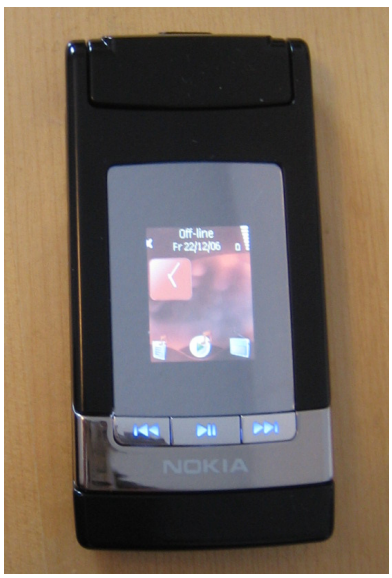
Näytön sytyttämisen tulee tapahtua jokaisesta puhelimen saamasta herätteestä. Näitä voi olla kolmenlaisia: toiminnallisuus, joka lähtee käyttäjästä itsestään, puhelimesta itsestään tai verkosta /3/. Oli toiminnallisuus sitten näppäinpainallus, saapuva puhelu, viesti tai aktivoituva hälytys on näytön sytyttävä uudelleen. Ongelmana tässä on, että näytön syttyminen ei tapahdu yhtä nopeasti kuin taustaledien syttyminen tai partial modesta palaaminen. Joitain käyttäjiä saattaa ärsyttää n. 1 sekunnin kestävä aika, joka kuluu näytön uudelleen sytyttämiseen. Tämän takia tekniikka on yleensä käyttäjän valittavissa joko käyttöön tai pois käytöstä.

Useiden käytettävyysongelmien vuoksi, näytön sammutuksen tukena käytetään usein led-tekniikkaa kertomaan laitteen tilasta. Led on yksinkertainen ja virtaa säästävä ratkaisu, jonka avulla voidaan käyttäjälle ilmoittaa mm. laitteen tila.

3.4 Ledin käyttö näytön rinnalla

Tätä tekniikkaa ei yksinään käytetä virransäästöön, mutta se toimii näytön sammuttamisen tukena. Kyseessä on siis näytöstä erillään oleva ledi, joka on sijoitettu laitteeseen näkyvälle paikalle. Tekniikan avulla voidaan korjata näytön sammuttamiseen liittyviä käytettävyysongelmia. Ledien avulla voidaan mm. ilmaista käyttäjälle: että laite ei ole sammunut, onko hänellä saapuneita puheluita, viestejä tai sähköposteja sekä mahdollisesti akun ja verkkosignaalin heikkous /3/.

Näytön ollessa pois päältä voidaan käyttäjälle ilmaista ledin avulla, että laite on vielä päällä. Apuna tässä käytetään ns. power save led -tekniikkaa. Laitteen näytön sammuttua antaa ledi valoimpulssin esimerkiksi 20 sekunnin välein. Tämä on yleensä riittävä kertomaan käyttäjälle, että laite on vielä käyttövalmiina. Tekniikka on käytössä mm. N76-laitteessa (kuva 7).



Kuva 7 Nokia N76

Vastaamattomien puheluiden ja saapuneiden viestien indikointi ledillä on käytössä mm. N93i-laitteessa. Käyttäjä voi itse määrittää kolmesta valittavasta väristä, punainen, vihreä tai sininen, oman värin saapuneille puheluille ja viesteille. Lisäksi laitteen latausta ja sen tilaa ilmaistaan ledin avulla. Vilkkuva punainen valo kertoo, että laite on latautumassa (kuva 8). Kun akku on täynnä, valo sammuu eikä enää vilku. Tämän ominaisuuden voi käyttäjä valita itse käyttöön tai pois käytöstä.

Kuva 8 Nokia N93i – Akku latautuu

Ledejä voidaan käyttää myös normaalin toiminnallisuuden tukena. Esimerkiksi meluisassa ympäristössä auttaa saapuvan puhelun huomaamista puhelimessa vilkkuva led-valo. Vilkkuva ledi auttaa myös, mikäli puhelin on pitkän matkan päässä. Lediä käytettäessä muun toiminnallisuuden tukena parannetaan puhelimen käytettävyyttä. Aiheesta on saatu positiivisia käyttäjäkokemuksia käyttäjätutkimuksessa /3/.

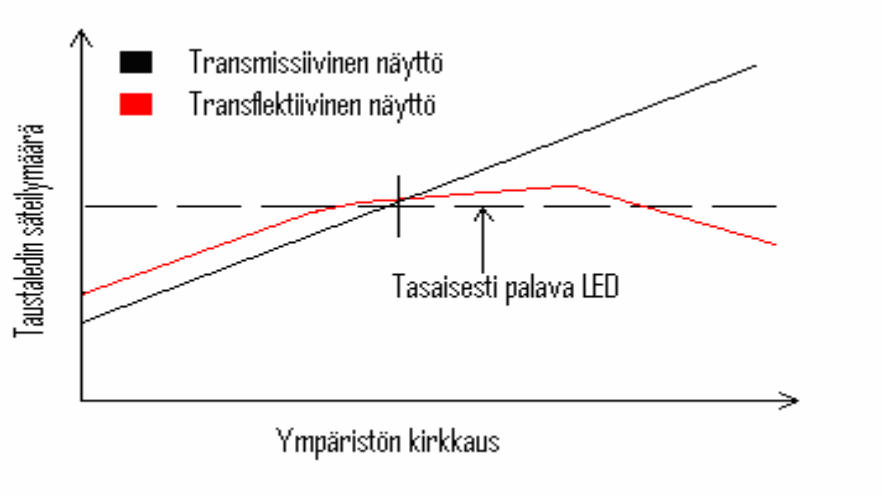
Vaikka kirkas ledi on monessa tilanteessa etu, osaa käyttäjistä jatkuva ledin palaminen tai välkkyminen voi myös ärsyttää. Varsinkin täysin pimeässä huoneessa, silmien totuttua pimeään, ledin välkkyminen saattaa näkyä erityisen voimakkaana. Tämän vuoksi ledin käyttö on useissa laitteissa käyttäjän valittavissa käyttöön tai pois käytöstä.

3.5 Ambient light sensorin käyttö virransäästössä

Ambient light sensor havaitsee ympäristön valonmäärän, ja tätä tietoa voidaan käyttää taustavalojen kirkkauden säätämiseen.

Kun laitetta käytetään pimeässä, näytön taustavaloa voidaan himmentää. Silmän pupillin laajetessa, se havaitsee paremmin ympäristöstä valoa säteilevät lähteet. Mikäli tässä tilanteessa näytön taustavalon kirkkaus on sama kuin normaalissa päivänvalossa, näyttää näyttö häiritsevän kirkkaalta.

Mikäli ambient light sensoria ei ole laitteessa ja taustavalot palavat vakioteholla ympäristön valoisuudesta riippumatta, laitteen näyttö on pimeässä liian kirkas ja kirkkaalla liian himmeä. Hämärässä sekä transmissiivisen että transflektiivisen näytön taustavalojen kirkkautta voidaan pienentää virrankulutuksen vähentämiseksi. Pienempi valoteho riittää myös transflektiivisessa näytössä, huolimatta heijastavasta kalvosta, josta osa taustavalon voimasta heijastuu hukkaan. Kirkkaassa ympäristössä voidaan tietyissä tapauksissa transflektiivisen näytön taustavalojen tehoa pienentää, mikäli heijastavan kalvon heijastavuus on riittävä – tällöin syntyy myös säästöä. Transmissiivisen näytön osalta taustavaloja sen sijaan ei voida himmentää kirkkaissa olosuhteissa, koska näytöstä puuttuu ympäristön valoa heijastava kalvo. Pientä hyötyä saavutetaan kuitenkin sillä, että normaaleissa olosuhteissa transmissiivisen näytön perusvaloteho voidaan säätää pienemmäksi ja käyttää suurempaa tehoa ainoastaan silloin, kun ympäristön valo on erittäin kirkas (kuva 9).



Kuva 9 Taustaledin säteilymäärä suhteessa ympäristön kirkkauteen

4 SLEEP MODE SOVELLUKSEN TAUSTALLA TOIMIVAT TEKNIIKAT

Tässä luvussa käsitellään tarkemmin Sleep mode -menetelmän teknisiä ominaisuuksia ja tutustutaan laitteessa pinnan alla tapahtuviin toimintoihin. Käsiteltävinä asioina ovat central repository, light targetit ja sleep mode -logiikan tarkempi tekninen kuvaus.

4.1 Sleep mode menetelmä

Sleep mode -menetelmällä tarkoitetaan matkapuhelimen näytön sammuttamista virrankulutuksen minimoimiseksi. Kokonaisuudessaan sleep mode -sovellus koostuu kahdesta osasta: asetusten käyttöliittymästä ja sleep moden aktivoimisen hoitavasta moduulista.

Asetukset ovat käyttäjälle näkyvä osa, jossa voidaan määrätä sleep mode käyttöön tai pois käytöstä. Sleep moden asetukset löytyvät laitteen yleisistä asetuksista ja

käyttäjä voi valita toiminnan olevan ”käytössä” tai ”ei käytössä”. Asetusten tila tallennetaan central repositoryyn (luku 4.2), josta sleep moden toiminnallinen moduuli voi niiden tilan tarkistaa.

Sleep moden aktivoiva moduuli on käyttäjältä piilossa. Tämä moduuli on aktiivinen koko ajan, kun laite on päällä. Se tarkkailee laitteen tilaa ja sen saamia herätteitä (ks. luku 3.3). Mikäli laite ei ole saanut herätettä tiettyyn ajanjaksoon mennessä, ja sleep mode on asetuksista valittu käyttöön, ohjelma sammuttaa laitteen näytön ja aktivoi laitteesta löytyvän power save ledin (ks. luku 3.4), joka kertoo käyttäjälle laitteen olevan vielä päällä.

Sleep mode on aina viimeisenä aktivoituva näytön virransäästömenetelmä. Laitteesta riippuen sitä ennen aktivoituvia toimintoja ovat esimerkiksi ledien himmennys ja partial mode. Esimerkkitapaus toiminnasta voisi olla seuraavanlainen: Käyttäjä laskee puhelimensa pöydälle lopetettuaan puhelun. Tämän jälkeen laite on käyttämättä 15s ja tällöin sen taustaledit himmennetään osittain. Ensimmäisen minuutin jälkeen taustavalot sammuvat kokonaan ja laite siirtyy partial modeen. Partial moden kestoksi on asetettu 5 minuuttia, ja kun tämä aika on kulunut loppuun, sleep mode aktivoituu. Samaan aikaan aktivoituu power save led, joka välähtää aina 20 sekunnin välein. Tästä käyttäjä huomaa, että hänen puhelimensa on vielä päällä eikä esimerkiksi rikki tai sammunut.

4.2 Light targetit

Light targeteilla tarkoitetaan joko laitteesta löytyvää yksittäistä lediä tai useamman ledin ryhmää. Tällaisia ovat mm. näytön taustaledit, näppäimistön valaisevat ledit, kameran apuna salamavalona toimiva led ja laitespesifiset ledit, kuten esimerkiksi näytön sammutuksen yhteydessä toimiva power save led.

Light targettien määrittäminen helpottaa laitteen ledien käyttöä. Kun light targeteille on määritetty valmiit enumeraatiot, on niiden käyttö ohjelmoitaessa helppoa. Yleensä laitteeseen on määritetty valmiiksi ainakin näytölle ja näppäimistölle omat targetit (EPrimaryDisplay, EPrimaryKeyboard). Käytön helpottamista varten näille kahdelle on lisätty myös yhteinen targetti, jolla saadaan molemmat ledi ryhmät syttymään yhtä aikaa (EPrimaryDisplayAndKeyboard). Mikäli kyseessä on kansinäytön omaava simpukkapuhelin, on myös tälle ja tämän mahdollisille näppäimille määritetty omat targetit (ESecondaryDisplay, ESecondaryKeyboard). Kuten aikaisemminkin, voidaan myös nämä kaksi kohdetta sytyttää yhdellä komennolla käyttämällä parametrina (ESecondaryDisplayAndKeyboard).

Näiden lisäksi laitteissa voi siis olla myös tuottekohtaisia erilaisiin käyttöihin tarkoitettuja ledejä tai lediryhmiä. Laitteessa voi toimia yksi voimakkaampi ledi esimerkiksi lamppuna avaimenreiän etsimiseen. Näille laitteille on määritetty omat targetit, joiden käyttö tulee varmistaa laitteen spesifikaatioista (ECustomTarget1-4).

Light targetteihin päästään käsiksi CHWRMLight -luokan kautta. Se tarjoaa rajapinnan, jonka kautta ledeille saadaan ohjelmoitua tarvittavat toiminnot (kuva10).

```
//Luodaan olio, jonka avulla käytämme ledejä
CHWRLight*  Light = CHWRLight::NewL();

//asetetaan ledi välkkymään
Light->LightBlinkL( CHWRMLight::ECustomTarget1,
                    KHWRMInfiniteDuration,
                    KLEDOnDuration,
                    KLEDOffDuration,
                    KHWRMDefaultIntensity );
```

Kuva 10 Esimerkki ledin käytöstä

Aluksi luodaan osoitin CHWRLight -olioon (kuva10). Tämän jälkeen asetamme ledin välkkymään ”LightBlinkL()” -funktiolla. Funktio saa parametreinaan halutun light targetin, ledin palamistyyli (tässä tapauksessa ledi palaa tasaisella teholla koko määritetyn palamisajan), ledin palamisajan, ajan jonka ledi on pois päältä sekä valon voimakkuuden.

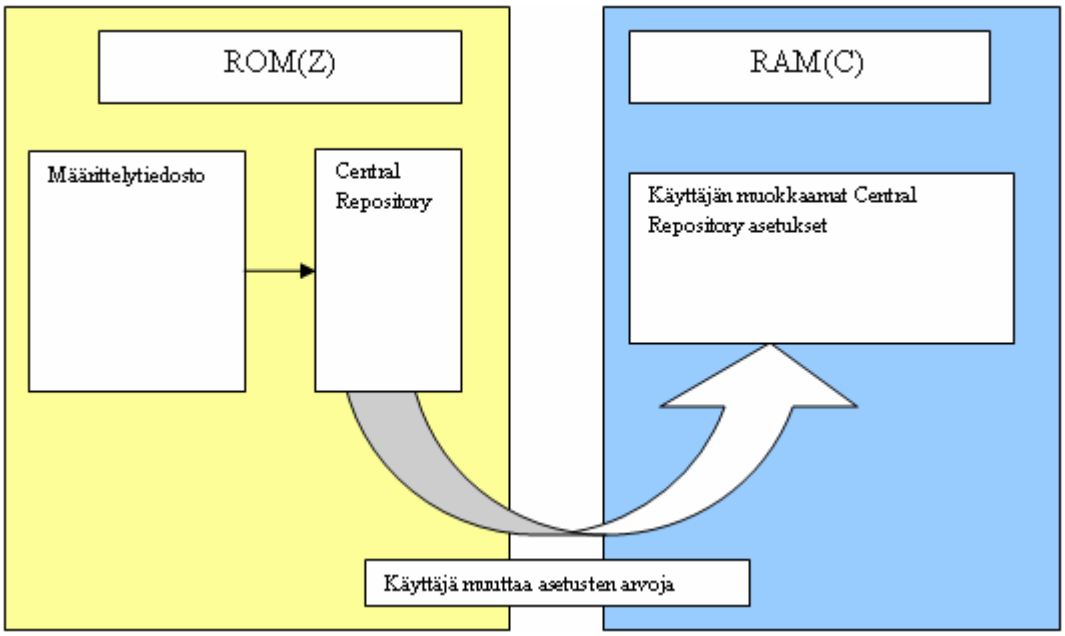
4.1 Central repository /5/

Central repository on Symbian käyttöjärjestelmän tarjoama rajapinta puhelimen asetusten käsittelyyn. Nimi central repository on hieman harhaanjohtava, sillä maksimissaan puhelimen käytössä voi olla jopa 2^{32} eri repositorya. Jokaisessa repository tiedostossa voi olla 2^{32} eri asetusta. Nämä asetukset voidaan merkitä joko positiivisina kokonaislukuina, desimaalilukuina tai merkkijonoina.

Repositoryt tunnistetaan UID(Unique IDentification) -numeron perusteella. Numero voi olla esimerkiksi ”0x12345678”, jossa kaksi ensimmäistä merkkiä ”0x” kertovat luvun olevan heksadesimaaliluku ja loput kahdeksan lukua toimivat repositoryn tunnisteena. Tunnisteen avulla ohjelmat pääsevät käsiksi oikeaan tiedostoon ja näin ollen muokkaavat oikean ohjelman asetuksia. Tämän lisäksi jokaisella asetuksella repositoryn sisällä on uniikki 32-bittinen positiivinen kokonaisluku tunnistenumerona, jota kutsutaan asetusavaimeksi. Avain voi esimerkiksi olla muotoa ”0x00000001”. Avaimen avulla pääsemme käsiksi repositoryn yksittäisen avaimen sisältämään asetustietoon.

Central repositoryyn tulee aina luoda oma repository ennen kuin sitä voidaan käyttää asetusten tallennukseen. Tämä tapahtuu luomalla uudelle repositorylle määrittelytiedosto, vaikkapa notepad ohjelmalla. Määrittelytiedosto tallennetaan oikeaan paikkaan puhelimen tiedostojärjestelmässä, ja sen avulla luodaan uusi repository. Uusia repositoryjä ei siis voi luoda dynaamisesti ajon aikana.

Määrittelytiedosto nimetään aina repositoryn UID:n mukaan (12345678.txt) ja siinä määritellään mm. repositoryn omistaja, asetusten avaimet, alkuarvot sekä niiden luku- ja kirjoitusoikeudet. Määrittelytiedosto luo central repositoryn ja se saa arvokseen määrittelytiedostossa määritetyt arvot. Central repository sijaitsee Z-asemalla, josta sen tietoja voidaan lukea, mutta jonne ohjelmilla ei ole oikeutta tallentaa muutoksia. Kun käyttäjä muuttaa repositoryn arvoja, luodaan muuttuneista avaimista kopio tiedosto C-asemalle, jonne ohjelmilla on oikeus tallentaa tietoa (kuva 11). Tämän jälkeen muokattujen avaimien asetukset luetaan C-asemalta. Asetusten arvot luetaan siis Z-asemalta kunnes käyttäjä on tehnyt niihin muutoksia.



Kuva 11 Central repositoryn sijainti laitteessa

Määrittelytiedosto

Repositoryn määrittelytiedostoja säilytetään central repositorylle varatussa hakemistossa, josta ne ladataan samalla kun repository rekisteröidään. Tiedostot voidaan asentaa hakemistoon kopioimalla ne suoraan osoitteeseen ”z:\private\10202BE9\” luotaessa laitteeseen uutta ohjelmapakettia.

Määrittelytiedosto (kuva 12) voidaan jakaa viiteen eri osaan: otsikko, omistaja(ei pakollinen), käyttöoikeus, metadata(ei pakollinen), avaimet ja niiden arvot. Kommentti merkkinä toimii ’#’.

```
cenrep
version 1
[owner]
0x1234567B
[platsec]
# Prosessin ID-numero tulee olla 100, ja sillä tulee olla tiedostojen tallennukseen
#oikeuttava kapabiliteetti, että sillä on kirjoitusoikeus repositoryn avaimiin
sid_rd=100 cap_wr=AllFiles, WriteDeviceData
[metada]
#default metadata arvo
0x00000100
[main]
0x1   int      0        0
0x2   int      1        1
0x3   string   “Hello”  “”     cap_rd=ReadDeviceData sid_wr=100
0x4   string   “moi”    “”     cap_rd=ReadDeviceData sid_wr=100
0x5   real     1.5      1.5
0x6   real     1e3      1e5
```

Kuva 12 Esimerkki repositoryn määrittelytiedostosta

Otsikko koostuu kahdesta ensimmäisestä tekstiä sisältävästä rivistä (kuva 13). Versionumero viittaa tekstiformaatin versionumeroon, eikä siis ole tiedoston luojan käyttämä versiohallinta menetelmä. Mikäli tiedoston kirjoittaja haluaa lisätä oman versionhallintanumeronsa, tulee se kirjoittaa kommentteihin.

```
cenrep
version 1
```

Kuva 13 Määrittelytiedoston otsikkorivit

Seuraava kenttä koostuu rivistä [owner] ja sitä seuraavasta rivistä, jossa on omistaja ohjelman UID-numero (kuva 14). Repositoryn määrittelytiedostolle tarvitaan omistaja, mikäli repositoryn avainten arvot aiotaan myöhemmin palauttaa

alkuperäisiin arvoihin laitteen tehdasasetusten palautuksen yhteydessä. Jos asetukset eivät kuulu palautettavien asetusten piiriin, voi tämä kenttä olla tyhjänä.

```
[owner]
0x1234567B
```

Kuva 14 Repositoryn omistajakenttä

Käyttöoikeuskentässä (kuva 15) määritetään tiedoston avaimien luku- ja kirjoitusoikeus. Yksittäiselle avaimelle voidaan määrittää luku- ja kirjoitusoikeudet myös erikseen avaimen määritys kentässä, mutta mikäli näin ei tehdä, siihen pätee käyttöoikeuskentässä määritetyt oikeudet. Jos tämä kenttä jätetään kokonaan tyhjäksi, eikä yksittäisille avaimille erikseen määritetä oikeuksia, ei millään prosessilla ole avaimien luku- tai kirjoitusoikeutta.

```
[platsec]
#Prosessin ID-numero tulee olla 1000, että sillä on repositoryn avaimien lukuoikeus

sid_rd=1000
```

Kuva 15 Lukuoikeus määräytyy prosessin SID-numeron perusteella

Kentän tunnisteena on rivi [platsec]. Tätä seuraa tieto, millä ohjelmilla on repositoryn avaimien luku- ja kirjoitusoikeus. Käyttöoikeus voidaan merkata asiakasprosessin SID(Security IDentification) -numeron perusteella, mitä kapabiliteettejä sen tulee omata, tai näitä tietoja voidaan käyttää yhdessä (kuva 16). Kapabiliteettejä voidaan määrittää maksimissaan seitsemän ja repositoryn tietoja käyttävän prosessin tulee omata määritetyistä kaikki. Mikäli käytössä on sekä prosessin SID-numero että kapabiliteetit, voidaan jälkimmäisiä määrittää maksimissaan kolme. Mikäli SID:n määrityksen jälkeen ei kapabiliteettejä aseteta, tulee niiden arvoksi automaattisesti ”AlwaysPass”. Tämä tarkoittaa kyseisen SID:n omaavalla moduulilla on avaimien luku- ja kirjoitusoikeus riippumatta sen kapabiliteeteista. Avaimien lukuoikeus merkitään aina ennen kirjoitusoikeutta.

```
# Prosessin ID-numero tulee olla 100,  ja sillä tulee olla tiedostojen tallennukseen
#oikeuttava kapabiliteetti, että sillä on kirjoitusoikeus repositoryn avaimiin

sid_rd=100 cap_wr=AllFiles,WriteDeviceData
```

Kuva 16 Lukuoikeus määräytyy SID-numeron, kirjoitusoikeus kapabiliteettien perusteella

Tietylle avainryhmälle voidaan myös määrittää muista tiedoston avaimista poikkeavat käyttöoikeudet (kuva 17).

```
#Mikä prosessi tahansa voi lukea avaimien 0x1000 – 0x2000 tilan, mutta vain
#WriteDeviceData ja NetworkControl kapabiliteetit omaava prosessi voi muuttaa
#niiden arvoa

0x1000 0x2000 cap_rd=AlwaysPass, cap_wr=WriteDeviceData, NetworkControl
```

Kuva 17 Avainryhmällä muista avaimista poikkeavat luku- ja kirjoitusoikeudet

Avainryhmän oikeuksien määrittämiseen voidaan käyttää myös maskia (kuva 18).

```
#Prosessin ID-numero tulee olla 1000, että sillä on avaimien luku-
#ja kirjoitusoikeus. Tämä koskee avaimia 0xAB0C, 0xAB1C, 0xABFC

0xAB0C mask= 0xFF0F sid_rd=1000 sid_wr=1000
```

Kuva 18 Esimerkki maskin käytöstä avaimien oikeuksien määrittämisen apuna

Eri avainryhmien käyttöoikeudet voidaan määrittää missä tahansa järjestyksessä. Mikäli kuitenkin käy niin, että samalle avaimelle määritetään oikeudet useammin kuin yhden kerran, tulee viimeisenä määritetyt oikeudet aina voimaan. Tämän vuoksi oikeuksia määritettäessä tulee olla tarkkana, sillä repositoryn määrittelytiedostossa tehdyt virheet ovat erityisen hankala huomata ohjelmien kehitysvaiheessa.

Vaikka repositoryjä ei voi luoda dynaamisesti, voidaan niihin kuitenkin luoda ajon aikana uusia avaimia. Avaimien luomiseen käytetään central repositoryn tarjoamaa Create() -funktiota. Uusia avaimia luotaessa ei siis tarvitse luoda uutta

määrittelytiedostoa. Metadata kentässä määritetään näille dynaamisesti luoduille avaimille alkuarvo (kuva 19).

```
[metada]
#default metadata arvo
0x00000100

#Metadata arvo, joka tulee avaimille välillä 0x200 – 0x500
0x200 0x500 0x00000020

#Metadata arvo avaimille 0xAB0C, 0xAB1C, 0xABFC jne
0xAB0C mask= 0xFF0F 0x00000040
```

Kuva 19 Esimerkki metadatan määrittämisestä

Viimeisessä kentässä määritetään repositoryn ensimmäisellä avauskerralla sisältämät tiedot. Kentän tunnisteena toimii rivi [main]. Tätä seuraavat tiedot ovat:

<avain> <tyyppi> <arvo> <metadata> [yksittäisen avaimen käyttöoikeus]

```
[main]
0x1    int       0        0
0x2    int       1        1
0x3    string    “Hello”  “”    cap_rd=ReadDeviceData sid_wr=100
0x4    string    “moi”    “”    cap_rd=ReadDeviceData sid_wr=100
0x5    real      1.5      1.5
0x6    real      1e3      1e5
```

Kuva 20 Yksittäisten avainten tiedot

Kuten jo aikaisemmin mainittiin, yksittäiselle avaimelle voidaan määrittää myös erikseen luku- ja kirjoitusoikeudet (kuva 20). Avaimelle määritetyt oikeudet voivat poiketa repositoryn käyttöoikeuskentässä määritetyistä. Tässä kentässä pätee täysin samat säännöt kuin käyttöoikeuskentässä. Määritettäessä yksittäiselle avaimelle käyttöoikeuksia, ovat ne aina avaimen paikkaan sidottuja. Mikäli avain tuhotaan, se saa uudelleen luotaessa samat käyttöoikeudet, mutta jos avain siirretään uuteen paikkaan, käyttöoikeudet eivät siirry sen mukana. Käyttöoikeudet pysyvät alkuperäisellä avain paikalla, ja jos kyseiseen paikkaan siirretään toinen asetus, saa se kyseiset käyttöoikeudet.

Repositoryn datan käsittely

Central repositoryyn päästään käsiksi repository rajapinnan (API) kautta. Rajapinta peritään CRepository luokasta ja otetaan käyttöön lisäämällä sitä käyttävän luokan header-tiedostoon rivi ”#inlcude <centralrepository.h>”. Tämän lisäksi koodin linkkaamista varten tulee projektiin lisätä vielä ”centralrepository.lib”- kirjasto.

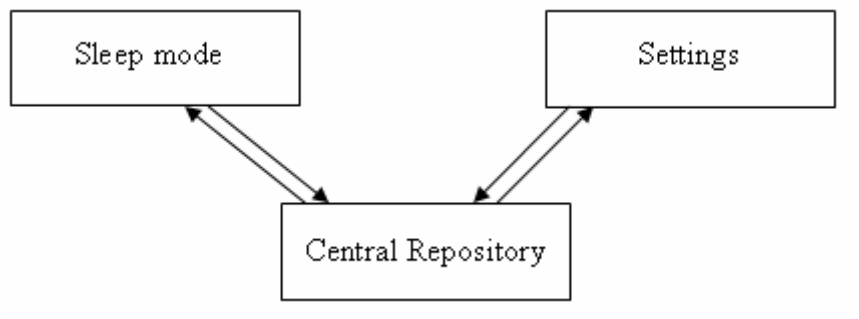
Repository avataan luomalla CRepository tyyppinen olio NewL()- tai NewLC()-funktiota, jotka saavat parametrikseen halutun repositoryn UID-numeron. Mikäli ohjelman tulee päästä käsiksi useamman kuin yhden repositoryn tietoihin, tulee sen luoda jokaista repositoryä kohden oma olio.

Repositoryn avaimien arvojen lukuun- ja kirjoitukseen löytyy Set()- ja Get()-funktiot. Molemmille annetaan parametreiksi avain, jonka tietoihin halutaan päästä käsiksi. Tämän lisäksi toisena parametrina annetaan luettavan tai kirjoitettavan tiedon tyyppi (joko TInt, TReal tai TDesC-tyyppinen tieto). Set()-funktio kirjoittaa tuon toisena parametrinaan saamaan tiedon avaimen arvoksi, ja Get()-funktio kopioi avaimen sisällön kyseiseen muuttujaan.

4.2 Asynkroninen tiedonvälitys central repositorya käyttäen

Asynkronisella tiedonsiirrolla tarkoitetaan tiedonsiirtoa, jossa tietoa lähettävä ja vastaanottava puoli voivat toimia toisistaan riippumatta (kuva 21). Tämä on moniajokäyttöjärjestelmässä perusedellytys toimivan kokonaisuuden rakentamiseksi. Kun toinen ohjelma lähettää tiedot, ei sen tarvitse jäädä odottamaan, että ohjelma tai moduuli, joka tietoja käyttää, on saanut ne vastaanotetuksi. Esimerkiksi Sleep mode ohjelmassa näyttöä ohjaavat virransäästöasetukset asetetaan silloin, kun käyttäjä niitä muokkaa. Virransäästö sen sijaan on käytössä koko ajan, ja asetuksia luetaan sitä mukaan kun niitä

tarvitaan. Viimeisimmät käyttäjän tallentamat asetukset ovat aina käytössä. Asetukset tallennetaan central repositoryyn josta ne ovat luettavissa milloin tahansa. Sleep mode ohjelma käy tarvittaessa tarkastamassa tiedot ja toimii central repositoryyn tallennettujen tietojen mukaan. Hyvänä puolena tekniikassa on myös se, että asetussovellus ei tarvitse olla päällä muutoin kuin asetuksia muutettaessa. Sitä ei siis tarvitse käynnistää joka kerta kun sleep mode tarkistaa asetuksia, joten samalla säästetään laitteen muistia.



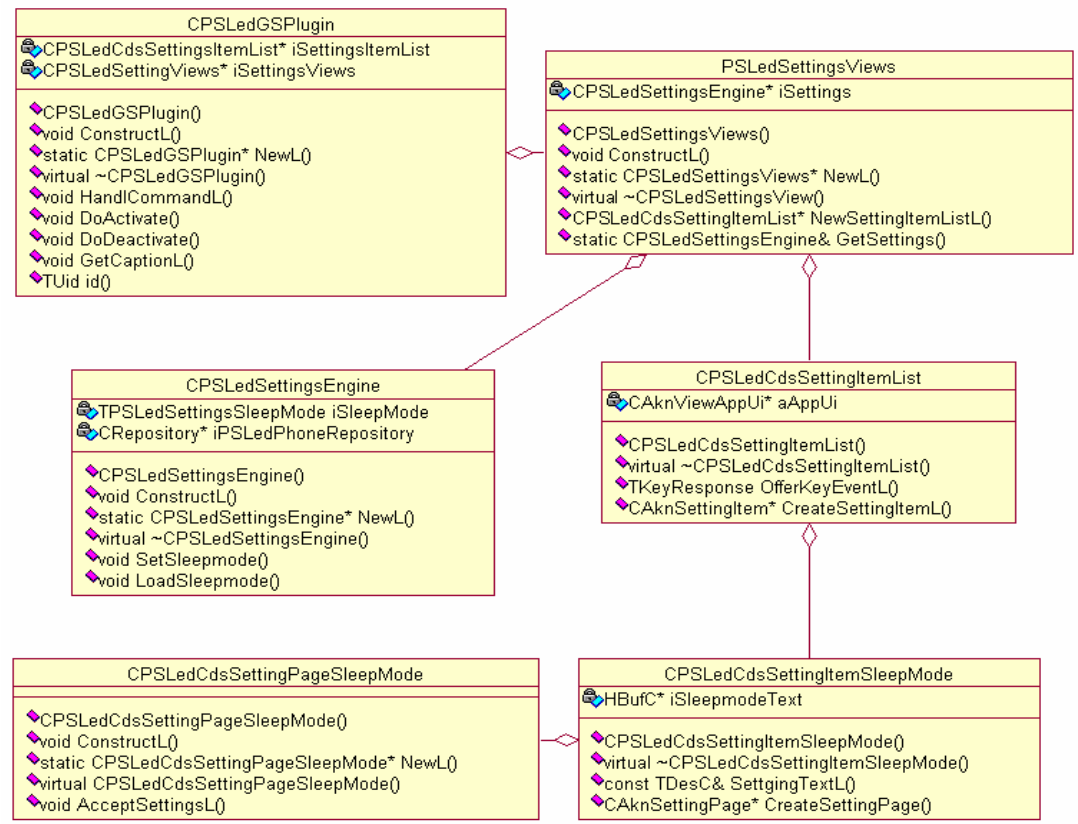
Kuva 21 Sleep mode sovelluksen ja sen asetusten välinen yhteys repositoryn avulla

5 SLEEP MODE / POWER SAVE LED –SOVELLUS

Tässä kappaleessa esitellään sleep mode -sovellus. Sovelluksesta läpi käydään sekä asetusten käyttöliittymä että näytön sammuttava ja ledin aktivoiva moduuli. Asetusten käyttöliittymä on käyttäjälle näkyvä osa, josta sleep moden tukena oleva led valitaan joko käyttöön tai pois käytöstä. Itse ledin kontrolloinnista ja näytön sammuttamisesta vastaava moduuli ei ole käyttäjälle näkyvissä, vaan se on taustalla aktiivisena koko sen ajan, kun laite on päällä. Se kuuntelee laitteen saamia herätteitä sekä laskee viimeisestä kulunutta aikaa sammuttaakseen näytön ja aktivoidakseen ledin.

5.1 Asetusten käyttöliittymä

Käyttäjälle näkyvä osa sovelluksesta on puhelimen yleisten asetusten alta löytyvä sleep mode asetus moduuli. Moduulin avulla käyttäjä voi valita näytön sammutuksen tukena toimivan ledin joko käyttöön tai pois käytöstä. Luokkakaavioon (kuva 22) tutustumalla saa kuvan moduulin luokista, sekä luokkahierarkiasta.

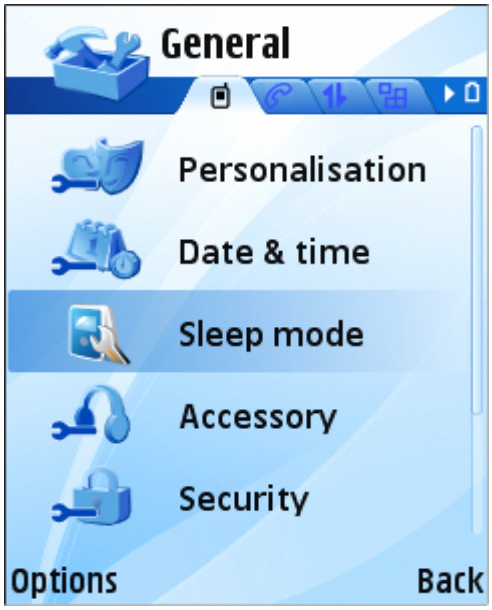


Kuva 22 Asetus moduulin luokkakaavio

CPSLedGSPlugin

CPSLedGSPlugin -luokka huolehtii moduulin ikonin latauksesta sekä nimen asetuksesta yleisten asetuksista löytyvään näkymään(kuva 23). Kun käyttäjä valitsee sleep mode asetukset muokattavaksi, kutsutaan luokan DoActivate() –

funktiota. Tässä funktiossa kutsutaan CPSLedSettingsItemView –luokan NewSettingsItemListL() -funktiota, joka luo resurssitiedostosta asetusten listanäkymän. NewSettingsItemListL() palauttaa osoitteen CPSLedSettingsItemList -olioon ja näin saadaan yhteys CPSLedPluginin ja CPSLedItemList:n välille.



Kuva 23 Yleisistä asetuksista löytyvä Sleep mode plugin

CPSLedSettingsView

CPSLedSettingsView -luokka toimii rajapintaluokkana asetuslistaan ja sovelluksen engine luokkaan. Se luo CPSLedCdsSettingsItemList -olion ja välittää sen osoitteen CPSLedPluginille. Lisäksi se luo CPSLedSettingsEngine -olion, joka toimii asetusten enginenä.

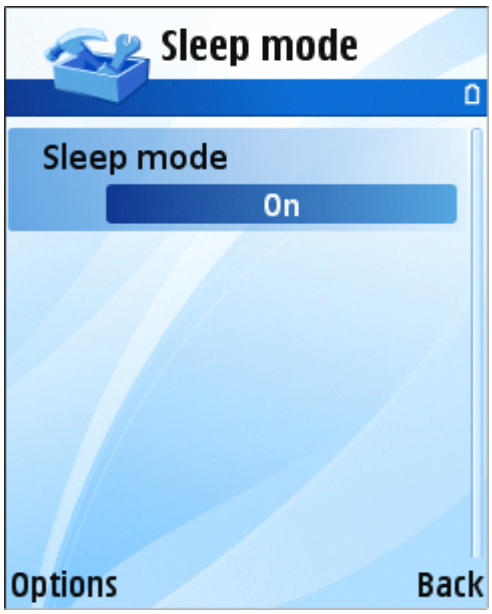
CPSLedEngine

CPSLedEngine -luokka toimii rajapintana Central repositoryyn. Luokka huolehtii uusien asetusten tallennuksesta sekä asetusten lataamisesta. Asetusten lataukseen luokalla on LoadSleepMode() -funktio, joka hakee repositorystä sovelluksen sen

hetkiset asetukset. Asetusten tallennukseen luokkaan on määritetty SetSleepMode() -funktio, joka tallentaa käyttäjän valitsemat asetukset repositoryyn.

CPSLedSettingsItemList

CPSLedSettingsItemList -luokassa kuunnellaan käyttäjän näppäin painalluksia OfferKeyEventL() -funktiossa ja luodaan CPSLedSettingsItemSleepMode -olio, kun käyttäjä valitsee asetuksen muokattavaksi (kuva 24). Mikäli moduuli käsittelisi useampia asetuksia, listattaisiin ne kaikki tässä luokassa.



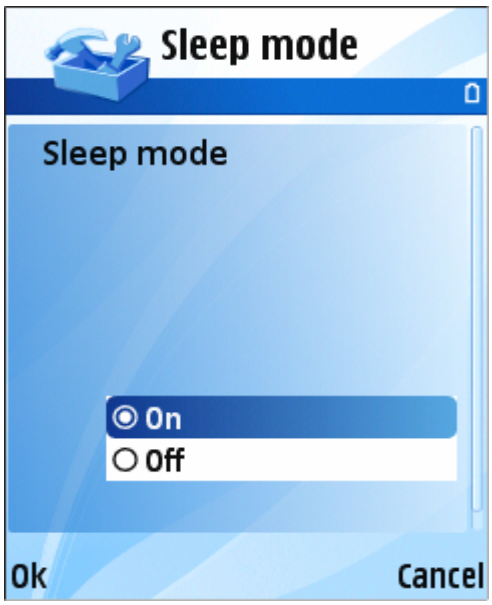
Kuva 24 Sleep mode pluginin sisältämien asetusten lista

CPSLedCdsSettingItemSleepMode

CPSLedCdsSettingItemSleepMode -luokka luo CPSLedCdsSettingPageSleepMode -olion, joka toimii asetusten muokkaamisnäkymänä (kuva 25). SettingsItem -luokka lataa resurssitiedoston perusteella oikeat tekstit, jotka näkyvät settingspage -näkymässä. Tämän lisäksi luokka lataa repositoryssä olevat asetusarvot enginestä CPSLedSettingsView -luokan välityksellä.

CPSLedCdsSettingPageSleepMode

CPSLedCdsSettingPageSleepMode -luokka luo asetus näkymän (kuva 25), josta käyttäjä voi muokata asetuksen arvot. Luokka välittää käyttäjän valinnan CPSLedSettingsView -luokan kautta CPSLedSettingsEngineen, joka tallentaa tiedot repositoryyn.

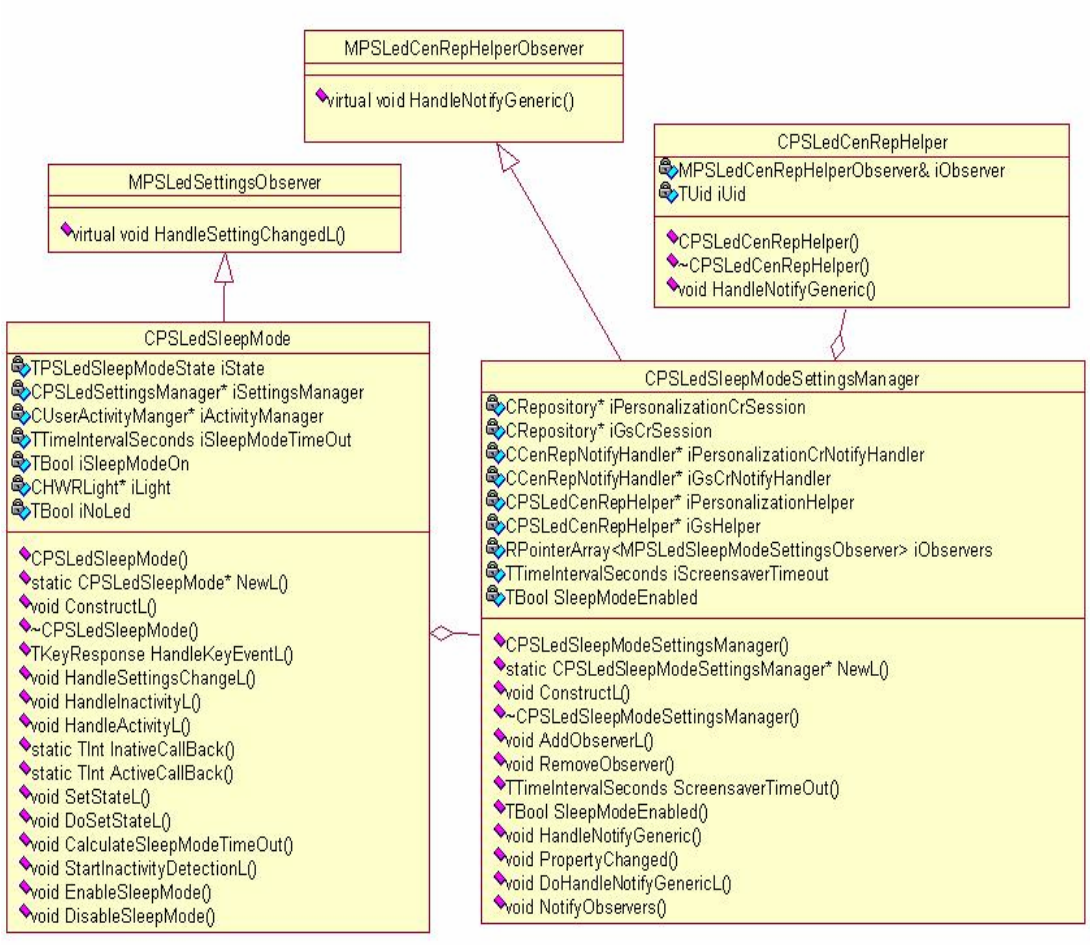


Kuva 25 Sleep moden asetusarvon vaihto näkymä

5.2 Sleep mode toiminnallisuus

Tässä kappaleessa käsitellään sovelluksen toinen moduuli, eli käyttäjältä piilossa oleva sleep mode toiminnallisuus. Moduulin tehtävänä on seurata laitteen saamia herätteitä, sekä laskea viimeisimmästä kulunutta aikaa. Kun viimeisimmästä herätteestä on kulunut käyttäjän asettama aika (esimerkiksi 5min), sammuttaa moduuli laitteen näytön ja aktivoi power save ledin. Ledin aktivointi tapahtuu tietenkin ainoastaan siinä tapauksessa, että käyttäjä on valinnut ledin käyttöön.

Luokkakaaviota tutkimalla saa kuvan moduulin luokista sekä luokkahierarkiasta (kuva 26).



Kuva 26 Sleep moden luokkakaavio

CPSLedSleepMode

CPSLedSleepMode -luokka on sovelluksen pääluokka, joka aktivoi sleep moden sammuttamalla näytön ja kytkemällä power save ledin päälle. Lisäksi luokka huolehtii sleep modesta palaamisen kytkemällä näytön päälle ja sammuttamalla ledin. Toiminnallisuudet tapahtuvat ”EnableSleepMode” - ja ”DisapleSleepMode” -metodeissa.

Luokan ”iActivityManager” -muuttuja toimii ajastimena, joka laskee viimeisimmästä herätteestä kulunutta aikaa. Ajastinta luotaessa, sille annetaan parametriksi käyttäjän asettama aika minuutteina (iSleepModeTimeOut), jonka kuluttua laitteen sleep mode kytkeytyy päälle. Tämän lisäksi ajastin saa parametreiksi InactiveCallBack()- ja ActiveCallBack() -funktiot.

Mikäli laite saa herätteen ennen kuin on kulunut ”iSleepModeTimeOut” -muuttujassa määritelty aika, kutsutaan ActiveCallBack() -funktiota, joka nollaa ajastimen ja käynnistää sen uudestaan. Laitteen oltua käyttämättä asetuksissa määritetyn ajan palataan InactiveCallBack() -funktioon, joka asettaa sleep moden aktiiviseksi ja EnableSleepMode() -funktion avulla sammuttaa näytön ja käynnistää ledin.

HandleSettingsChanged() on takaisinkutsu funktio, joka peritään MPSLedSleepModeSettingsObserver -luokasta. Asetusten muuttuessa ilmoitetaan luokalle siitä tämän funktion kautta. Funktio muuttaa luokan ”iBlink” -muuttujan arvoa sen mukaan, valitseeko käyttäjä power save ledin käyttöön vai pois käytöstä.

Tämän lisäksi luokka luo CPSLedSleepModeSettingsManager -olion, joka kuuntelee ja ilmoittaa asetusten muutoksista.

CPSLedSleepModeSettingsManager

CPSLedSleepModeSettingsManager -luokan avulla kuunnellaan repository muutoksia. Luokka periytyy MPSLedCenRepHelperObserver -luokasta. Se sisältää tiedon, minkä repositoryn avaimen arvo on muuttunut. Koska Series60 tarjoama repositoryn kuunteluun tarkoitettu rajapinta (MCenRepNotifyHandlerCallback) ei palauta muuttuneen repositoryn UID-numeroa, joudutaan jokaista kuunneltavaa repositoryä kohden luomaan olio, joka pitää sisällään repositoryn UID-numeron,

sekä muuttujan johon tieto muuttuneesta avaimesta kopioidaan. ”iPersonalizationHelper” ja ”iGsHelper” -oliot pitävät sisällään kyseiset tiedot. Ne annetaan parametreina repository muutoksia kuunteleville ”iPersonalizationCrNotifyHandler” - ja ”iGsCrNotifyHandler” -oliolle. Tämän lisäksi nämä viimeksi mainitut oliot saavat luotaessa parametreina osoittimet kuunneltaviin repositoreihin: ”iPersonalizationCrSession” ja ”iGsCrSession”. Kun ”NotifySandler” -oliot on luotu, asetetaan ne kuuntelemaan asetuksia.

Mikäli kuuntelijat huomaavat muutoksen tapahtuneen, tulee tieto siitä luokan HandleNotifyGeneric() takaisinkutsufunktion kautta. Tästä lähtee eteenpäin tieto siitä, minkä repositoryn ja minkä avaimen arvo on muuttunut. Arvoa verrataan vielä edelliseen tiedossa olevaan arvoon, ja mikäli tiedot eivät täsmää ilmoitetaan uusi arvo CPSLedSleepMode -luokalle NotidyObservers() -funktiosta kutsuttavalla HandleSettingChangeL() -funktiolla.

CPSLedCenRepHelper

Tämän luokan kautta käytetään MCenRepNotifyHandlerCallback rajapintaa, joka on Series60 tarjoama repositoryn kuuntelun hoitava rajapinta. Kun tästä luokasta luodaan olio, annetaan parametrina MPSLedCenRepHelperObserver -luokasta luotu olio, joka sisältää repositoryn UID-numeron ja muuttujan, jossa tieto muuttuneesta avaimesta voidaan palauttaa. Näin toimitaan siis sen takia, että voidaan kuunnella useamman kuin yhden repositoryn muutoksia. Pelkästään MCenRepNotifyHandlerCallback rajapintaa käyttämällä tämä ei onnistuisi, sillä se ei palauta tietoa siitä missä repositoryssä muutokset ovat tapahtuneet.

6 YHTEENVETO

Työssä läpikäytyjen matkapuhelimennäytön virransäästömenetelmien osalta suurimpana yllätyksenä tuli toimintojen monimutkaisuus. Mikä ennen näytti ainoastaan näytön sammumiselta, nostaa nykyään mieleen kymmenien sivujen mittaiset käyttäjätutkimukset sekä huomioitavat käytettävyysongelmat. Lisäksi moni asia, joita aikaisemmin on pitänyt itsestään selvyytenä, kuten esimerkiksi ledien himmennys ja sammutus, ovat tarkkaan tutkittuja ja suunniteltuja parhaan käyttömukavuuden aikaan saamiseksi.

Itse sleep mode sovelluksen yhteydessä suurinta päänvaivaa tuotti Series60 ympäristö. Sen monimutkaisuus tuntui välillä allekirjoittaneen kokemuksen huomioonottaen vaikealta ymmärtää. Monesti tuli eteen asioita, joita ei vaan osannut ottaa huomioon ja jotka nykyisin näyttävät niin selviltä, ettei niitä osaisi välttämättä toiselle heti ensi käänteessä edes neuvoa. Kaikista ongelmista kuitenkin selvittiin kysymällä, ja mikäli asiaan ei heti saatu vastausta, löytyi useimmissa tapauksissa apu joko esimerkin tai puhelinsoiton avulla.

LÄHDELUETTELO

1 Mättö Juhani, Projekti päällikkö. Luento 7.2.2007. Nokia Oyj.

2 Järventie Heli, Matkapuhelimen käyttäjälleen sallimat toiminnan tasot yhteiskuntatieteellisen teknologia tutkimuksen näkökulmasta. Pro Gradu. Tampereen Yliopisto. Sosiologian ja sosiaalipsykologian laitos. Tampere 2005. sivut 63, 70, 72, 83-84.

3 Järventie Heli, UI Asiantuntija. Luento 9.2.2007. Nokia Oyj.

4 Mättö Juhani, Projekti päällikkö. Luento 13.3.2007. Nokia Oyj.

5 Symbian OS ”Central Repository –HowTo” –Nokian sisäinen materiaali