SAVONIA

OPINNÄYTETYÖ - **AMMATTIKORKEAKOULUTUTKINTO**
LUONNONTIETEIDEN ALA

MEEGO–SOVELLUKSEN SUUNNITTELU JA OHJEL-MOINTI

CASE: EKG-pitkäaikaisrekisteröinnin päiväkirja

TEKIJÄ/T: Tomi Häkkinen

SAVONIA-AMMATTIKORKEAKOULU

OPINNÄYTETYÖ
Tiivistelmä

Koulutusala Luonnontieteiden ala	
Koulutusohjelma Tietojenkäsittelyn koulutusohjelma	
Työn tekijä(t) Tomi Häkkinen	
Työn nimi Meego-sovelluksen suunnittelu ja ohjelmointi. CASE: EKG-pitkäaikaisrekisteröinnin päiväkirja	
Päiväys 10.10.2013	Sivumäärä/Liitteet 43
Ohjaaja(t) Marja-Riitta Kivi	
Toimeksiantaja/Yhteistyökumppani(t)	

Tiivistelmä

Opinnäytetyön tavoitteena oli suunnitella ja luoda matkapuhelimeen päiväkirjasovellus EKG-pitkäaikaisrekisteröinnin avuksi. Työssä käytettiin Linux-käyttöjärjestelmään pohjautuvaa matkapuhelinta.

Opinnäytetyössä kuvataan ohjelmistokehityksen eri vaiheita ja valintoja sekä käytettyjä työvälineitä Meego-käyttöjärjestelmälle ohjelmoitaessa. Työssä kuvataan myös käyttöliittymän rakentamista QML-ohjelmointikielellä. Lisäksi pyritään ottamaan huomioon käytettävyys sovelluksen käyttötarkoitusta ajatellen sekä matkapuhelimeen liittyvät erityispiirteet ohjelmaa suunniteltaessa ja ohjelmoitaessa.

Sovellus ohjelmoitiin Meego-käyttöjärjestelmälle käyttäen Qt Quick -kehitysympäristöä. Ohjelmointikielinä käytettiin QML- ja Javascript-ohjelmointikieliä. Työtä voidaan hyödyntää erityisesti QML-kielisessä ohjelmistoprojekteissa, mutta myös yleisesti mobiilisovellusten suunnittelussa.

Avainsanat
päiväkirjasovellus, EKG-pitkäaikaisrekisteröinti, Meego, QML

SAVONIA UNIVERSITY OF APPLIED SCIENCES

THESIS
Abstract

Field of Study Natural Sciences			
Degree Programme Degree Programme in Information Technology			
Author(s) Tomi Häkkinen			
Title of Thesis Design and development of a Meego-application. CASE: Diary for ECG long term registration.			
Date	10.10.2013	Pages/Appendices	43
Supervisor(s) Marja-Riitta Kivi			
Client Organisation /Partners			

Abstract

The purpose of thesis was to design and create diary application to help long-term ECG registration on a Linux-based mobile phone.

This thesis describes the different stages of and choices made during software development and the development tools that were used while programming for a Meego operating system. This thesis also describes user interface programming with QML -programming language. Usability from the viewpoint of the use of the application and special characteristics of the mobile phone were also consider in development.

The application was programmed for Meego operating system by using Qt Quick framework and QML and Java-script languages. This thesis can be applied in particular in QML software development but also generally for de-signing mobile applications.

Keywords
diary-application, long-term ECG registration, Meego, QML

SISÄLTÖ

1 JOHDANTO

Opinnäytetyössä toteutetaan Meego-käyttöjärjestelmälle suunniteltu ja ohjelmoitu EKG-pitkäaikaisrekisteröinnin avuksi tarkoitettu päiväkirjasovellus.

Päiväkirjasovelluksen tarkoitus on helpottaa päiväkirjamerkintöjen tekemistä seurannan aikana. Sovellus kulkee esimerkiksi luontevasti käyttäjän mukana matkapuhelimeen asennettuna. Se myös laskee toiminnon keston automaattisesti ja helpottaa käyttäjää tallentamaan toiminnon aikana mahdollisesti ilmenneet oireet.

Opinnäytetyössä käydään läpi sovelluskehitykseen liittyviä vaiheita ja huomioidaan erityispiirteitä, jotka liittyvät matkapuhelinsovellusten kehittämiseen ja Linux-käyttöjärjestelmään. Näitä ovat esimerkiksi ristiinkääntäminen ja paketointi. Työssä on myös esitelty sovelluksen käyttämä ratkaisu tietojen tallentamiseen.

Näiden lisäksi opinnäytetyössä perehdytään käyttöliittymän suunnitteluun ja rakentamiseen QML-ohjelmointikielellä. Tässä osuudessa otetaan huomioon sovelluksen käyttötarkoitus ja erilaiset käyttötilanteet. Näiden lisäksi tarkastellaan muutoksia, jotka ovat käyttöliittymässä tapahtuvia tai erityisesti käyttäjää ohjaavia. Työssä käytetyt esimerkit on esitetty niin näyttökaappauksina kuin myös kooditasolla tehtävinä määrityksinä.

2 MEEGO

Meego on Linuxiin perustuva käyttöjärjestelmä, joka oli Nokian ja Intelin perustama yhteistyöprojekti. Meego oli suunniteltu käytettäväksi matkapuhelimissa, mutta myös erilaisissa kuluttajaelektroniikan laitteissa kuten digibokseissa, tablettitietokoneissa sekä sulautetuissa järjestelmissä. (Wikipedia 2013a.)

2.1 Meego-projektin käynnistyminen

Vuonna 2005 Nokia esitteli Linux World Summit -tapahtumassa Internet tablettinsa 770 (Sharma 2005). Laite julkaistiin Maemo-kehitysalustalle, joka oli erityisesti suunniteltu Nokian Internet tableteille. Nokia julkaisi Maemosta yhteensä viisi versiota vuosien 2005 – 2010 välisenä aikana.

Vuonna 2007 – 2008 Intel julkisti mobiililaitteille tarkoitetun käyttöjärjestelmän Moblinin. Moblin, joka oli lyhennys sanoista Mobile Linux, oli Intelin Atom-prosessoreiden ympärille luotu Linux-pohjainen käyttöjärjestelmä. Se oli suunniteltu erityisesti mobiililaitteille, tableteille ja kannettaville tietokoneille. (Wikipedia 2013b.)

Helmikuussa 2010 Mobile World Conferenssissa ilmoitettiin, että Nokian Maemo ja Intelin Moblin aiotaan yhdistää yhteiseksi projektiksi MeeGoksi (Wikipedia 2013b). Yhteistyön tarkoituksena oli saada yhteen perustajat, avoimen lähdekoodin yhteisö sekä kaupalliset ja ei-kaupalliset toimijat ja lisätä Linuxin laajentumista mobiililaitealustana (Haddad 2011).

2.2 Harmattan

Ennen Intel yhteistyötä Nokia kehitti Maemo-käyttöjärjestelmää vuosina 2005 – 2010. Viimeisen version, Maemo 6, version koodinimenä käytettiin Harmattania. Ennen Intel yhteistyötä aloitettu Maemo 6:n kehitystä päätettiin jatkaa ja yhteistyön julkistamisen jälkeen se pyrittiin rakentamaan mahdollisimman hyväksi yhteensopivuudeltaan Meegon kanssa. (Kurri 2012.)

Maemo 6 -brändistä kuitenkin päätettiin luopua, sillä markkinointi tapahtuisi jatkossa Meego-nimen alla. (Quim 2010.) Harmattan oli tarkoitettu eräänlaiseksi yhteenliittäjäksi Maemon ja Meegon kanssa, jonka ohjelmointirajapinta olisi yhteensopiva MeeGo 1.2 version kanssa. (Kurri 2012.) Ari Jaaksi, Nokian Meego-osaston varajohtaja, on pitänytkin Harmattania pikemminkin Meegon ilmentymänä kuin käyttöjärjestelmänä. (Quim 2010.)

Nokia käytti kahdessa matkapuhelimessaan N9 ja N950 Meego -käyttöjärjestelmää, joista vain ensimmäinen julkaistiin kuluttajille. Näissä puhelimissa siis käytettiin Nokian jatkokehittämää Harmattan-kehityshaaraa Meego:sta. (Pitkänen 2011.)

2.3 Meegon alasajo

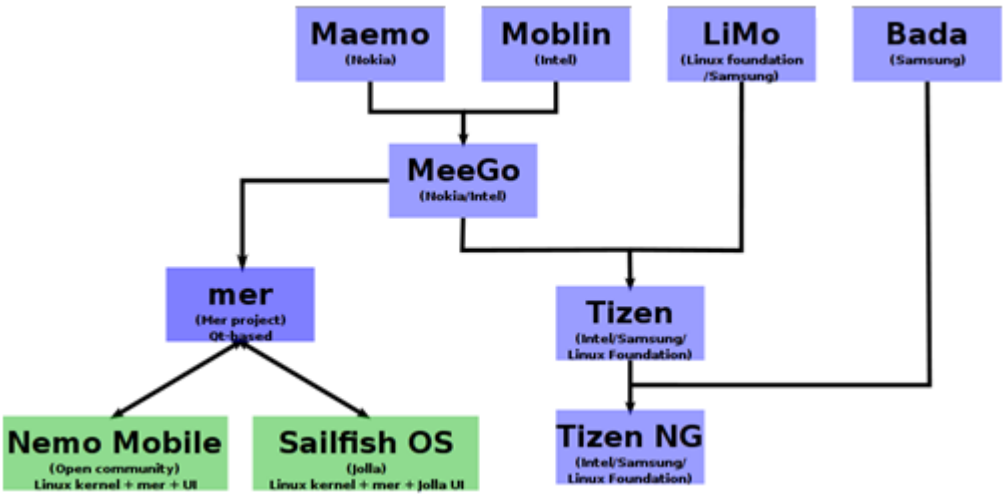
11.2.2011 Nokia ilmoitti strategiamuutoksestaan. Yhteistyössä Intelin kanssa syntynyt Meego – käyttöjärjestelmä tullaan korvaamaan Windows Phone 7 -käyttöjärjestelmällä älypuhelinalustana. Samalla Meegon kehityksestä vastannut kehitysjohtaja Alberto Torres irtisanottiin. (Orlowski 2011.) Yhtenä syynä on pidetty sitä, että Nokia olisi saanut ainoastaan kolme Meego-puhelinta tuotua markkinoille ennen vuotta 2014 (Burrows 2011.). Vuoden 2011 strategiassaan Nokia ilmoitti, että Meego-käyttöjärjestelmän kehitys jatkuu lähinnä tutkimusmielessä. (Nokia 2011.)

2.4 Jatkokehitetyt järjestelmät

2.4.1 Sailfish

14.10.2011 perustettiin Jolla Oy -niminen ohjelmistojen suunnitteluun ja kehittämiseen keskittynyt yritys. (YTJ 2013.) Yrityksen takana oli joukko entisiä Nokialaisia, jotka olivat keskeisiä toimijoita Nokian Meego-osastolla. (Kwang 2012.)

Jolla käyttää puhelimissaan kehittämäänsä Sailfish-käyttöjärjestelmää, joka perustuu Meego-käyttöjärjestelmään. Sailfish ei kuitenkaan käytä suoraan Nokian Harmattania Meegosta, vaan Mer-kehityshaaraa (Kaavio 1). (Saadebra 2012.) Lisäksi yhtenäisyyksiä löytyy mm. Qt-kehitysympäristön käyttämisestä. Ensimmäinen Sailfish-käyttöjärjestelmää käyttävä puhelin on tarkoitus julkaista loppuvuonna 2013 (Sajari 2013.).



KAAVIO 1: Mobiililinuxin kehityshaarat. (Wikipedia 2013c.)

2.4.2 Tizen

Intel päätti jatkaa yhdessä Samsungin kanssa avoimeen lähdekoodiin perustuvan Linux-käyttöjärjestelmällä pyörivän matkapuhelimeen suunnatun käyttöjärjestelmän kehittämistä. Poikkeavuutena Meego/Harmattaniin ja Sailfishin nähden on mm. Qt-kehitysympäristön tuen puute. Tizen puolestaan tukee erityisesti HTML5-rajapintaa. (Wikipedia 2013d.)

3 QT, QT QUICK, QML

3.1 QT

Qt on Trolltechin kehittämä C++:n pohjautuva alustariippumaton kehitysympäristö, joka on saatavilla useille eri alustoille. Työpöytäalustojen lisäksi Qt on saatavilla useille mobiilialustoille ja sulautetuille Linux-järjestelmille. (Digia 2013a.)

Alustariippumaton Qt mahdollistaa sen, että sovelluksen ohjelmakoodia voidaan käyttää pohjana myös muihin alustoihin ohjelmoitaessa. (Digia 2013b.) Qt suunniteltiin alun perin C++:n laajennukseen, mutta virallinen tuki löytyi myös mm. Java -kielelle (Thelin 2007.).

Qt:n tarina alkoi vuonna 1990, jolloin kaksi norjalaista, Haavard Nord ja Eirik Chambe-Eng, työskentelivät yhdessä C++ -pohjaisen tietokannan parissa, joka oli tarkoitettu ultraäänien tallennusta varten. Järjestelmän täytyi toimia graafisella käyttöliittymällä, joka taas toimisi niin Macintoshilla, Unixilla kuin Windowsillakin. Tarve ja siitä syntynyt keskustelu loivat lähtökohdan oliopohjaiselle alustariippumattomalle graafiselle kehitysympäristölle, jonka he toteuttaisivat. (Blanchette & Summerfield 2006.)

Vuonna 1994 kaksikko perusti yrityksen nimeltä Quasar Technologies, josta myöhemmin tuli Troll Tech ja myöhemmin nimi vaihdettiin vielä muotoon TrollTech. Vuonna 1995 julkistettiin ensimmäinen Qt-versio 0.97. Vuosi 1997 oli merkittävä Qt:n kannalta, sillä se otettiin osaksi KDE -ohjelmistokokonaisuutta. Samalla siitä tuli myös ”yleinen käsite”, de facto, C++ -graafisten käyttöliittymien kehityksessä Linux-ympäristössä. Vuoteen 2005 mennessä Qt oli edennyt jo versioon 4.0 sisältäen luokkia noin 500 ja funktioita yli 9000. (Blanchette & Summerfield 2006.)

Vuoden 2008 alussa Nokia osti Trolltechin 153 Miljoonalla dollarilla (Sayer 2008.). Yhtiön nimi myös muutettiin oston jälkeen Qt Softwareksi. (Dr Dobbs 2008.) Seuraavana vuonna Nokia lopetti Qt kehitysympäristön virallisen tukemisen Java -kielelle. (Paul 2008.)

Nokian ohjelmistostrategian muutoksista johtuen, se myi vuonna 2012 Digialle koko Qt:n ohjelmistoympäristön. (Digia 2012.)

3.2 Qt Quick

Qt Quick on Qt:stä laajennettu kehitysympäristö. Se on suunniteltu ohjelmistokehittäjien käyttöön suunniteltaessa käyttöliittymiä erityisesti matkapuhelimiin, mutta myös muihin kannettaviin laitteisiin. (Digia 2013c.)

Qt Quick -moduuli on standardikirjasto, joka tarjoaa kaikki välttämättömät komponentit graafisten käyttöliittymien luomiseen. Se sisältää mm. erilaisia tapoja luoda ja animoida komponentteja, vastaanottaa käyttäjän syötteitä ja luoda tietomalleja. (Digia 2013d.)

Qt Quick -moduuli sisältää QtQuick QML ja QtQuick C++ -moduulit. QtQuick QML -moduuli mahdollistaa käyttöliittymien luomisen QML-kielellä ja QtQuick C++ tarjoama C++ ohjelmistorajapintojen integroimiseen käyttöliittymään. (Digia 2013d.)

Qt Quick sai virallisen tuen ensimmäisen kerran Qt 4.7 versiossa, joka julkaistiin 21.7.2010. (Holwerda 2010.)

3.3 QML

QML (Qt Modeling Language tai Qt Meta-Object Language) on Javascript-pohjainen deklaratiivinen ohjelmointikieli ja se on osa Qt Quick -kokonaisuutta. (Nokia 2013a)

QML on erityisesti käyttöliittymien suunnitteluun luotu kieli. Se sisältää valmiiksi huomattavan määrän eri elementtejä, joita voidaan hyödyntää käyttöliittymän rakentamisessa. Koska kieli on deklaratiivinen, ohjelmointikieli hoitaa varsinaisen ohjelmointiosuuden ja ohjelmoijalle jää lähinnä komponenttien sijoittelu (Knuutila 1998.).

QML:n kehitys on alkoi vuosien 2007 ja 2008 vaihteessa. Projektista käytettiin aluksi nimeä Qt Kinectic ja sen ajatuksena oli tehdä animoitavien käyttöliittymien teko helpoksi ilman, että kehittäjän tarvitsisi hallita monimutkaisia tietojärjestelmiä, mikä oli tyypillistä animoiduille käyttöliittymille. Vuonna 2010 QML liitettiin osaksi Qt IDE kokonaisuutta. (Van Donderen 2010, 11.)

QML:n koodi muodostuu monista erilaisista objekteista, joiden sisään määritellään objektin asetukset, kuten sijainti, koko, väri jne. Objekteihin voidaan määritellä objekteja, jotka sisältyvät ylemmän tason objektiin. Alla olevassa esimerkissä neliön sisään sijoitetaan teksti ”Hyvää huomenta!” (Koodi 1). Jotta käyttäjä voi ottaa QML:n käyttöönsä, on hänen tuotava Qt Quick laajennus käyttöönsä.

```
Import QtQuick 1.1
Rectangle {
             Id: tausta
             Width: 400
             Height: 400
             Color: “red”
             Text {id: teksti
                          text: “Hyvää huomenta!”
                          Color: “white”
             }
    }
```

KOODI 1: Esimerkki QML-koodista.

4 SOVELLUSKEHITYS

Tässä opinnäytetyössä suunnitellaan ja toteutetaan päiväkirjasovellus Meego Harmattan -käyttöjärjestelmälle. Työssä esitellään käytetyt työvälineet ja niiden merkitykset sovelluskehityksessä, sekä käytetty tietokanta ja sen käyttöönottaminen. Pääpainopiste on selvittää käyttöliittymän rakentamista QML-ohjelmointikielellä Nokia N9 -matkapuhelimelle.

4.1 Opinnäytetyössä toteutettu sovellus

HolterMeemo on matkapuhelimella käytettävä EKG-pitkäaikaisrekisteröintiä avustava päiväkirjasovellus. Pitkäaikaisrekisteröinnin aikana tutkittava potilas joutuu kirjaamaan rekisteröinnin aikana tehtävät askareet ja toiminnot kuten heräämisen, liikunnan, lääkkeiden otot, ruokailut, levon jne. Näiden lisäksi seurataan mahdollisia oireita, joita potilas päivän aikana kokee. Niin ikään oireet ja niiden kestot kirjataan ylös.

Ohjelman tarkoituksena on helpottaa rekisteröinnissä tarvittavien päiväkirjamerkintöjen tekemistä yksinkertaisella käyttöliittymällä, toiminnallisuutta kuvaavien symbolein ja automaattisen laskennan avulla toimintojen kestoissa. Käyttäjä voi halutessaan myöhemmin täydentää oirekuvauksiaan mm arvioimalla oireen vakavuuden numeroasteikolla.

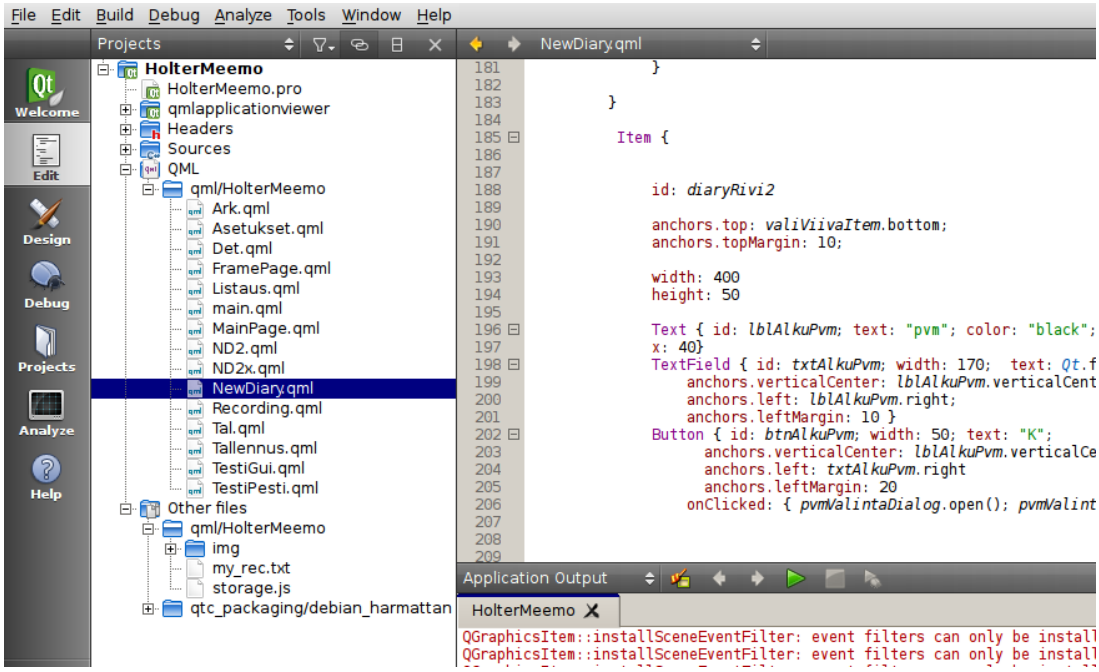
Ohjelmaa voidaan käyttää myös ilman potilaalle tehtävää rekisteröintitoimenpidettä, jolloin sovellus toimii eräänlaisena oirepäiväkirjana. Potilas voi esimerkiksi kerätä tietoa oireistaan muinakin aikoina kuin EKG-rekisteröintien yhteydessä ja raportoida niistä tarvittaessa seuraavan käynnin yhteydessä hoitohenkilökunnalle tai hoitavalle lääkärille.

4.2 Käytetyt työvälineet

4.2.1 QtCreator

QtCreator (QtC) on C++ -ohjelmistokehitin, joka sisältyy Qt SDK -sovelluskehityspakettiin (Kuva 1). QtC sisältää mm. mahdollisuuden ohjelmoida Qt Quick -sovelluksia, jollaiseksi tämän työn sovellus luokitellaan. QtC on ns. ”monialusta” kehitin, joka mahdollistaa sovelluksen kehittämisen eri laitteistotyypeille. (Digia 2013e)

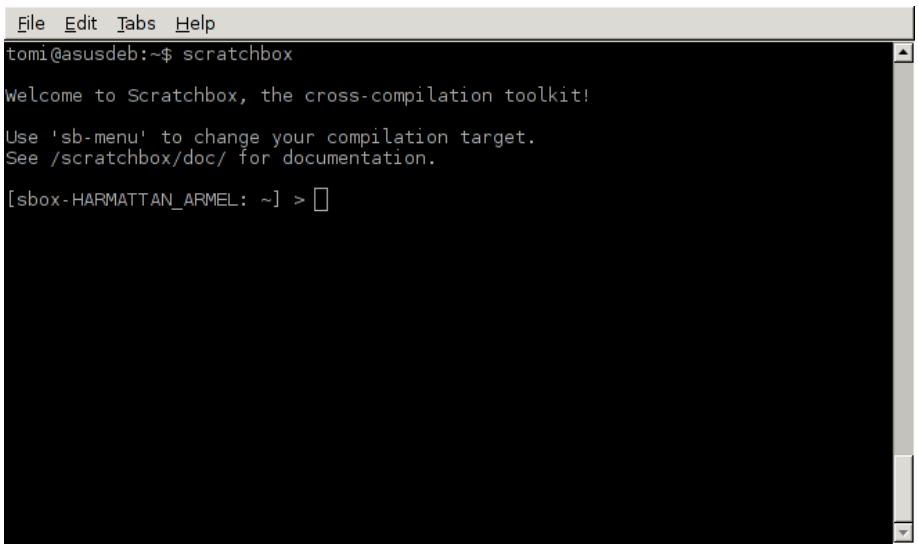
Käytetty kehittimen versio 2.4.1 perustuu Qt:n 4.7.4 versioon. Nokia julkaisi viimeisen version QtCreatorista 1.2.2012. (Nokia 2013b). Digia on julkaissut sekä kehittimestä, että Qt:sta uudemmat versiot (Knoll 2013-07-03), mutta niitä ei tässä työssä käytetä. Tällä pyritään välttämään mahdolliset yhteensopivuusongelmat.



KUVA 1: QtCreator-ohjelmistokehitin.

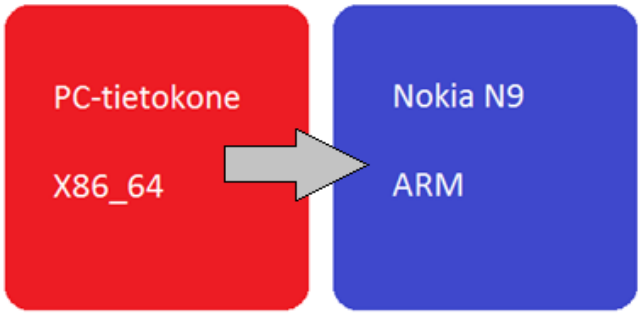
4.2.2 Scratchbox

Scratchbox on ohjelmatyökalu, joka mahdollistaa mm. sovelluksen ristiinkääntämisen ja paketoinnin. (Scratchbox). (Kuva 2)



KUVA 2: Scratchbox-työkalu.

Ristiinkääntämistä tarvitaan silloin, kun alusta johon sovellusta ollaan ohjelmoimassa, sisältää eri prosessoriarkkitehtuurin kuin laite, jolla varsinainen ohjelmointityö tapahtuu. Esimerkiksi tässä työssä varsinainen ohjelmointi tapahtuu tietokoneella, jossa on X86-käskykantaa noudattava prosessori ja kohdelaite, eli puhelin, käyttää ARM-arkkitehtuurin prosessoria (Kaavio 1). Ohjelmistoa ei voida ajaa laitteessa ilman että se käännetään sopivaksi kyseiselle laitteelle.



KAAVIO 1: Käytetyt laitteet, ”ohjelmointisuunta” ja arkkitehtuurit.

Ohjelmisto on mahdollista kääntää myös kohdelaitteistolla, eli tässä tapauksessa matkapuhelimella, mutta tapa on huomattavasti hitaampi mitä nykyaikaisella tietokoneella ajettu Scratchbox -ristiinkääntäjä. (Mankinen & Rahkonen 2005.)

4.3 Kohdelaite

Laite, mihin HolterMeemo sovellusta ohjelmoidaan, on Nokia N9 matkapuhelin (Kuva 3). Laitteessa on 1GHz ARM -prosessori sekä erillinen grafiikkaprosessori. Laitteessa on käyttömuistia 1GB sekä 16 tai 64GB massamuistia mallista riippuen. Näyttötarkkuus laitteessa on 854x480 pikseliä. (GSMArena 2013.)

Näyttötarkkuus on työn kannalta oleellisin, sillä se määrittää hyvin pitkälti minkä kokoisia komponentteja käyttöliittymään voidaan tehdä. Prosessorilla, grafiikkaprosessorilla tai muistin määrällä ei ole merkitystä, koska työ ei sisällä tai luo suurta prosessori-, grafiikkaprosessoritehoa tai suurta muistia vaativia toimintoja.

KUVA 3: Nokia N9 matkapuhelin. (Nokia 2013c.)

Puhelin on valittu työhön siksi, että se oli ainoa Suomen markkinoilla oleva ei-symbian puhelin joka tuki Qt -ohjelmointikieltä. Nokian N950 -mallia ei koskaan julkistettu kaupallisesti vaan sitä annettiin rajoitettu määrä sovelluskehittäjille. Blackberry puhelimia vastaavasti on Suomessa ollut saatavilla hyvin rajoitetusti vasta viime vuosina (Sirkiä 2012.).

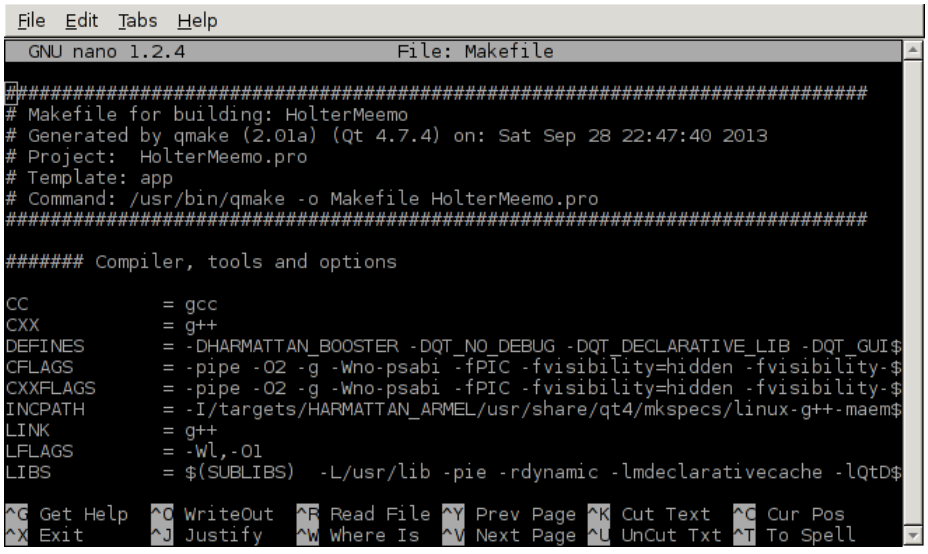
4.4 Kääntäminen ja paketointi

HolterMeemon kääntäminen ja paketointi ajettavaan muotoon tapahtuu Scratchbox -ristiinkääntötyökalun avulla.

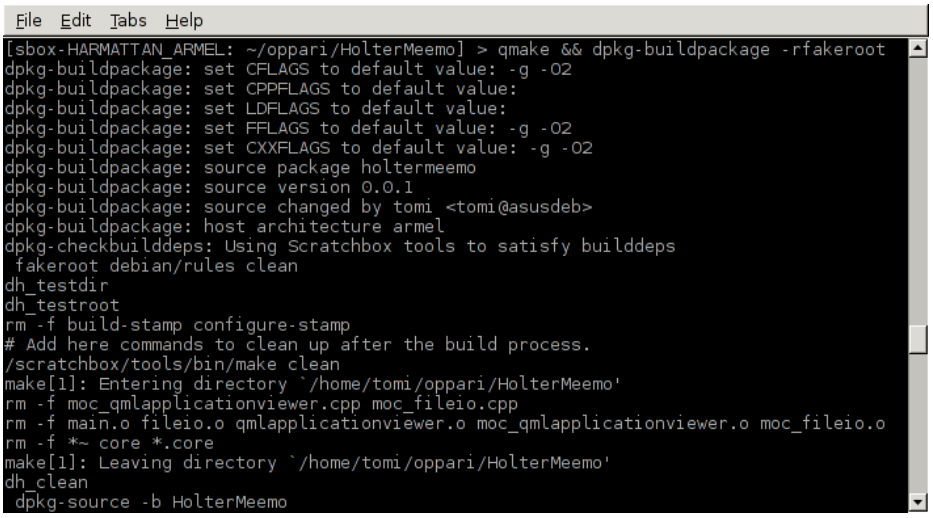
Paketoinnissa kerätään sovelluksen tiedot kuten tiedostot, kuvat yms. yhteen, pakataan tiedostot pienempään tilaan ja yhteen pakettiin sekä määritetään minne tiedostot asennetaan. Jotta ohjelmisto voidaan hallitusti asentaa, on paketointi erittäin suositeltava toimenpide. Paketti sisältää nimittäin tiedot mm. riippuvuuksista, joita ohjelmisto tarvitsee toimiakseen oikein (Wikipedia 2013e). Myös ohjelmistoa poistettaessa paketoinnin mukana tulleen tiedostolistauksen avulla voidaan ohjelman tiedostot poistaa turvallisesti ja hallitusti. Paketointi mahdollistaa myös sovelluksen päivittämisen ilman, että käyttäjän erikseen tarvitsisi poistaa vanhaa versiota ja asentaa uutta.

Paketointijärjestelmiä on useita, mutta Meego/Harmattan käyttää paketoinnissa Debianin deb-paketointijärjestelmää. Järjestelmä on sama mitä käytettiin Meegoa edeltäneessä Maemo-kehitysalustassa. Alkuperäinen Meego puolestaan käytti RedHatin kehittämää rpm-paketointijärjestelmää (The Linux Foundation 2013.).

Kääntäminen aloitetaan qmake-komennolla, joka generoi Makefile-tiedoston (Kuva 4). Tätä tiedostoa tarvitaan varsinaiseen kääntämiseen, sillä se sisältää tiedot käännettävistä tiedostoista sekä määritellyt säännöt, jotka liittyvät kääntämiseen. Itse kääntäminen tapahtuu samassa yhteydessä paketoinnin kanssa komennolla ”dbkg-buildpackage -rfakeroot” käännettävän sovelluksen juurihakemistossa (Kuva 5). Optiolla ”-rfakeroot” ainoastaan ohitetaan vaatimus käyttää pääkäyttäjää paketin luomiseen.



KUVA 4: Qmake-työkalun luoma Makefile-tiedosto.



KUVA 5: Qmake ja dpkg-buildpackage on käynnistetty ”putkikomentona”.

Paketointi edellyttää, että käyttäjä on luonut debian-alahakemiston tiedostoineen, joka sisältää erilaisia sääntöjä paketointiin liittyen. Hakemiston ja tiedoston luominen tapahtuu QtCreatorissa uuden projektin luomisen yhteydessä.

Tiedostot, jotka luodaan, määrittävät seuraavia asioita:

debian/changelog = sisältää versionmuutokset sovelluksesta.
debian/compat = määrittelee numerolla 7-9 yhteensopivuuden debhelper-työkalulle.
debian/control = sisältää tietoa paketista kuten paketin nimen, prosessoriarkkitehtuurin, riippuvuudet sekä paketin kuvauksen. Tietoja paketinhallintatyökalut käyttävät käsitellessään pakettia.
debian/copyright = sisältää tietoa ohjelmiston tekijänoikeudesta.
debian/rules = määrittelee säännöt, joita dpkg-buildpackage käyttää luodessaan paketin.
(Rodin, Aoki, Small ja Hertzog. 1998.)

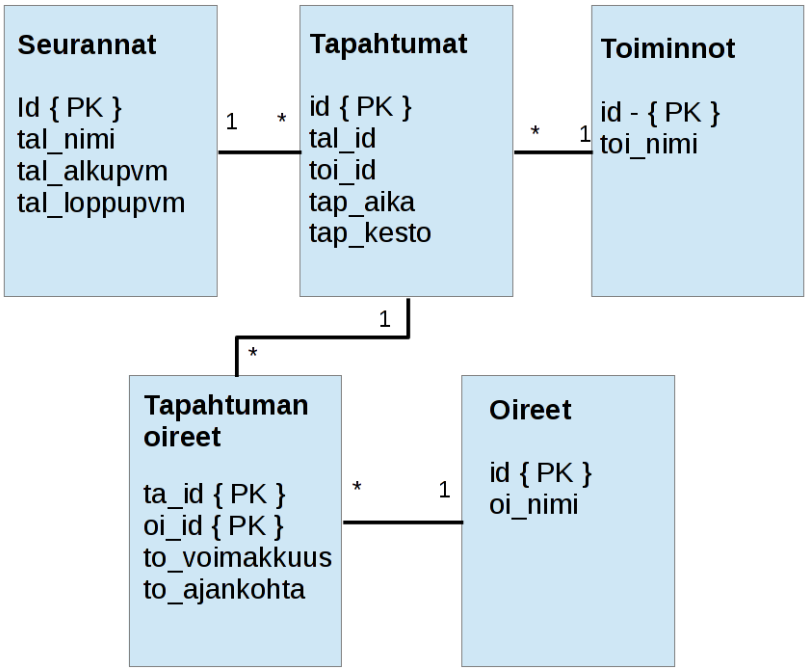
Valmis paketti löytyy sovelluksen juuritason ”yläpuolelta”, muodossa holtermeemo_0.0.1_armel.deb. Tiedoston nimestä löytyy sovelluksen nimi (holtermeemo), sovelluksen versio (0.0.1) sekä alusta (armel), jolle sovellus on käännetty.

4.5 Tallennettavien tietojen hallinta

Sovellus tallentaa rekisteröinnistä nimen ja alku- sekä loppuajan. Rekisteröintiin liittyviin tapahtumiin tallennetaan kellonaika, kesto ja tehty toiminto. Mahdollisista oireista puolestaan tallennetaan oireen voimakkuus ja kellonaika.

Meego/Harmattan, kuten myös monet muutkin älypuhelimet, sisältää SQLite-tietokantajärjestelmän. Se mahdollistaa tietojen tallentamisen paikallisesti muiden SQL-92 standardin mukaisesti, muutamia poikkeuksia lukuun ottamatta. (Wikipedia 2013f.)

Sovelluksessa siis käytetään tietojen tallentamiseen paikallista SQL-tietokantaa. Se on käytettävissä ainoastaan lokaalisesti, eikä siihen siis voida saada yhteyttä verkon yli. Tietokanta muodostuu viidestä taulusta: Seurannat, Tapahtumat, Toiminnot, Oireet ja Tapahtuman oireet. (Kaavio 2)



KAAVIO 2: HolterMeemo sovelluksen tietokantakaavio.

Kooditasolla tietokantaa voidaan käyttää Javascriptin avulla. Tietokannan ohjaus on sijoitettu omaan Javascript-tiedostoon, josta sitä kutsutaan tarvittaessa QML-sivulla.

Tietokanta otetaan käyttöön objektilla *openDatabaseSync*(), jolla määritellään tietokannan nimi, versio, kuvaus tietokannasta sekä käytettävä muistin määrä tavuina. (Digia 2013f.)

```
function haeTietokanta() {
     return openDatabaseSync("HolterMeemoTestiFinal2", "1.0", "StorageDatabaseTesti-
Final2", 100000);
}
```

KOODI 2: Tietokannan tiedot palautetaan haeTietokanta funktiosta.

Haettu tietokantamäärittely siirretään muuttujaan db. Tietokantaan luodaan taulut, jos se on ensimmäistä kertaa käytössä. (Koodi 3)

```
function maaritteleKanta() {
    var db = haeTietokanta();
    db.transaction(function(tx) {
            tx.executeSql('CREATE TABLE IF NOT EXISTS Seurannat(talNro INT PRIMARY
            KEY, talNimi TEXT, talAlkuPvm DATE, talLoppuPvm DATE)');
            tx.executeSql('CREATE TABLE IF NOT EXISTS Tapahtumat(tpNro INT PRIMARY KEY,
            tal_tp_Nro INT, toi_tp_Nro INT, oi_id_Nro INT, talAika_tp DATE)');
            tx.executeSql('CREATE TABLE IF NOT EXISTS Toiminnot(toNro INT PRIMARY KEY,
            toNimi TEXT)');
            tx.executeSql('CREATE TABLE IF NOT EXISTS Oireet(oiNro INT PRIMARY KEY, oiNi
            mi TEXT)');
            tx.executeSql('CREATE TABLE IF NOT EXISTS TapahtumanOireet(ta_id INT, oi_id
            INT, voim INT, PRIMARY KEY (ta_id, oi_id))');
         });
    }
```

KOODI 3: Tietokannan taulujen luominen.

Metodin db.transaction(callback(tx)) avulla on toteutettu varsinainen tietokannan hallinta. Metodi sisältää transactionin ja *callback* -funktion, jossa kutsumalla metodia *tx.executeSql()* voidaan antaa tietokantaa hallitsevia komentoja. Jos tietokannan syötössä on virhe, callback palauttaa virheen, exceptionin, jolloin transaction palautetaan muutosta edeltäneeseen tilaan, eikä muutosta näin ollen hyväksytä. (Digia 2013g.) Sijoittamalla tämä try-catch -haaraan, voidaan poikkeuksia hallita ja palauttaa funktioista esimerkiksi tietty arvo. (Koodi 4)

```
function maaritaTallennus(talNro, talNimi, talAlkuPvm, talLoppuPvm) {

   var db = haeTietokanta();
   var res = "";

    try {
        db.transaction(function(tx) {
        tx.executeSql('INSERT OR REPLACE INTO Tallennukset VALUES (?,?,?,?)',
        [talNro,talNimi,talAlkuPvm,talLoppuPvm]);});
        res = "OK";
    }
    catch (err) {
        res = "Virhe";
    }
    finally { return res; }
}
```

KOODI 4: Tallennusfunktio Javascript-koodissa.

4.6 Jatkokehitys

4.6.1 Siirtäminen toiseen laitteeseen

HolterMeemo sovelluksen siirtäminen toiseen laiteympäristöön on mahdollista suhteellisen pienin toimenpitein. Esimerkiksi Symbianin puhelinmallit Anna ja Belle, tukevat myös Qt Quick -kehitysympäristön sovelluksia. (Digia 2013h.)

Myös tulevaan Sailfish -käyttöjärjestelmää käyttävään Jolla -puhelimeen siirto olisi toteutettavissa. Osa käytettävistä QML-komponenteista on vaihtunut Qt Quick 2.0:ssa, mutta niihin on olemassa korvaavat komponentit. (Sailfish 2013.)

4.6.2 Sovelluksen muokkaus tarpeiden mukaan

Sovelluksen muokkaaminen paremmin omiin tarpeisiin olisi myös toteutettavissa. Käyttäjä voisi lisätä omia toimintoja kuten ”Sauvakävely” tai ”Ratsastus”, sekä näille omat ikonit. Järjestelmän tietokantaan ei tarvitsisi tehdä kovinkaan suuria muutoksia. Muutokset koskisivat lähinnä käyttöliittymää, johon tulisi uuden toiminnon lisäävä toiminto. Tämän voisi sijoittaa esimerkiksi Asetukset-sivun alaisuuteen.

4.6.3 Tiedon siirtäminen laitteesta

Tallennetut tiedot sijaitsevat laitteen muistissa, mutta tällä hetkellä niitä ei voida siirtää sähköisesti. Tietojen siirtämisen laitteesta voisi toteuttaa NFC tai Bluetooth -tekniikoilla, sillä nämä molemmat tekniikat löytyvät laitteesta (GSMArena 2013). Siirtämisessä tulisi ottaa huomioon järkevä siirtomuoto. Tämä voisi olla esimerkiksi PDF-dokumentti, johon ohjelmaan lisätty toiminto kirjoittaisi raportin rekisteröinnistä ja tapahtumatallennuksista. Siirrossa tulisi ottaa huomioon myös salaus. Salausta voisi myös harkita käytettäväksi dokumenttiin.

4.6.4 Useamman käyttäjän mahdollisuus

Jos sovellus kehittyisi enemmän yleiseksi terveyden seurantaan tarkoitetuksi päiväkirjaksi tai se olisi käytössä esimerkiksi poliklinikoilla potilasseurannan yhteydessä, olisi käyttäjätili hyvin perusteltu ratkaisu. Käyttäjätunnuksen ja salasanan taakse tallentuva tieto pysyisi henkilökohtaisena. Myös tallennetun tiedon salaaminen olisi välttämätöntä, etenkin potilasseurannassa.

5 KÄYTTÖLIITTYMÄ

5.1 Käytettävyys

Sinkkonen, Kuoppala, Parkkinen ja Vastamäki (2002) määrittelee kirjassaan ”Käytettävyyden psykologia” käytettävyyden teoria- ja menetelmäkentäksi, jonka avulla pyritään käyttäjän ja laitteen yhteistyötä tehostamaan sekä tekemään siitä käyttäjän kannalta miellyttävämpi. Kuutti (2003, 13) on samoilla linjoilla ja yksinkertaistaa ajatuksen käytettävyydestä ihmisen ja laitteen väliseksi kanssakäymiseksi.

Nielsen (2012) määrittelee käytettävyyden viidellä laatutekijällä: opittavuudella, tehokkuudella, muistettavuudella, virheettömyydellä ja mielekkyydellä. Näitä tekijöitä on myös pyritty ottamaan huomioon HolterMeemo-sovellusta suunniteltaessa mm. käyttämällä yhtenäisiä sivupohjia, luomalla käyttäjälle visuaalinen efekti liikkumisesta tai pyrkiä estämään virheellinen käyttö.

5.2 Käyttö

Ohjelma avaa käynnistyessään käynnistysruudun (Kuva 6), josta selviää ohjelman nimi sekä versio. Käynnistysruutu näkyy käyttäjälle kolme sekuntia, jonka jälkeen avautuu päävalikko. Käynnistysruutu on suunniteltu vaakatasoon.

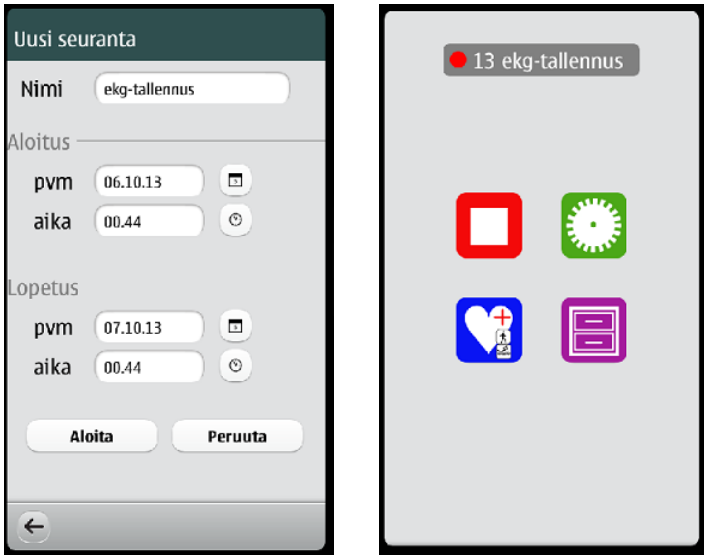
KUVA 6: HolterMeemo 1.0 -sovelluksen käynnistysruutu.

Päävalikko koostuu neljästä eri toiminnosta, symbolista, joiden avulla käyttö tapahtuu. Plus avaa näkymän, josta uusi seuranta luodaan. Sydän avaa näkymän, josta oireet ja toiminnot tallennetaan rekisteröintiin. Ratas avaa näkymän, josta hallitaan ohjelman käyttöasetuksia. Laatikosto mahdollistaa vanhojen rekisteröintien selaamisen ja katselun. Jos seuranta ei ole käynnissä, on tapahtumien tallentaminen estetty, ja näin ollen ikoni näkyy himmennettynä. (Kuva 7)

KUVA 7: Päävalikko Meego-teemalla käynnistyksen jälkeen puhelimen ollessa vaakatasossa.

5.2.1 Seurannan aloittaminen

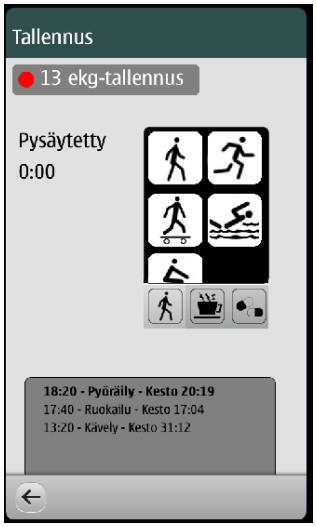
Käyttäjä aloittaa käytön valitsemalla (+), jolloin hänelle avautuu näyttö Uusi Seuranta uutta rekisteröintiä varten (Kuva 8). Käyttäjää pyydetään määrittelemään seurannalle nimi sekä seurannan aloituksen ja lopetuksen päivämäärät ja kellonajat. Päivämäärien ja kellonaikojen määritys tapahtuu painikkeiden avulla, jolloin käyttäjälle avautuu päivän ja kellonajan valintanäkymä. Sovelluksessa oletuksena on yhden vuorokauden seuranta aloitushetkestä. Kun käyttäjä on määritellyt tarvittavat seurantatiedot, hän voi aloittaa rekisteröinnin painamalla Aloita-painiketta. Sovellus palauttaa käyttäjän takaisin päänäyttöön, jossa symbolien lisäksi näkyy seurannan nimi ja tallennussymboli.



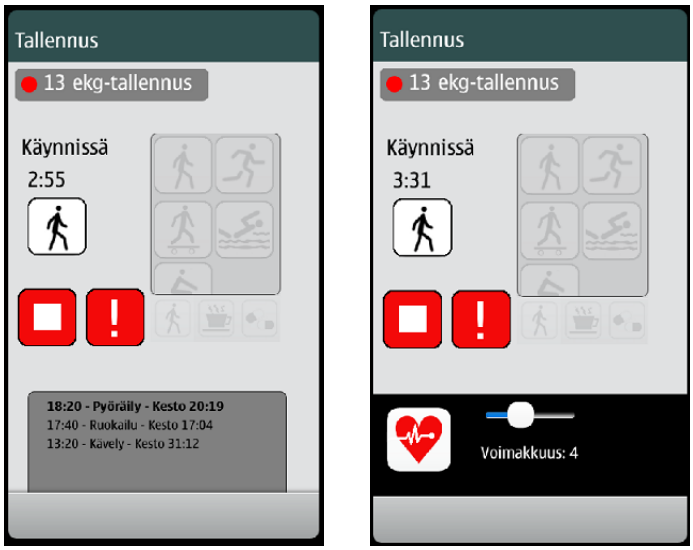
KUVA 8: Uuden seurannan aloitus ja näkymä pääsivulla aloituksen jälkeen.

5.2.2 Toimintojen tallennus

Toimintojen ja oireiden tallennus tapahtuu Tallennus -näytöltä (Kuva 9). Näytön oikeaan laitaan on sijoitettu eri symbolein kuvattuja toimintoja. Käyttäjä valitsee haluamansa toiminnon, jolloin näytön vasemmassa laidassa käynnistyy laskuri. Laskuri laskee sekunnin tarkkuudella toiminnon kestoa. Samalla näyttöön ilmestyvät stop- ja (!) -oiresymbolit, joita painamalla näyttöön avautuu pieni oire-laatikko, josta käyttäjä voi tallentaa oireen sekä sen voimakkuuden vieritystyökalulla (Kuva 10). Käyttäjä pysäyttää toiminnon haluamanaan ajankohtana painamalla stop-painiketta.



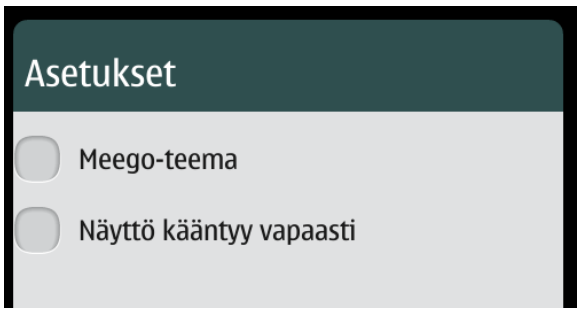
KUVA 9: Tallennus -näyttö.



KUVA 10: Tallennus-näyttö tallennuksen ollessa käynnissä ja oirepainike painettuna.

5.2.3 Asetusten muuttaminen

Asetusten muuttaminen tapahtuu Asetukset-sivulta (Kuva 11). Käyttäjällä on mahdollisuus muuttaa näytön kääntymisen reagointia ja näyttöjen värimaailmaa, sekä poistaa tietokantaan tallennetut tiedot.



KUVA 11: HolterMeemo-sovelluksen valittavat asetukset.

Näyttö voidaan käyttäjän niin halutessa kääntää aina oikein päin, olipa puhelin hänellä kädessä miten päin vain (Kuva 12). Oletuksena näyttö kääntyy vain ns. normaalitiloihin, yhteen pystysuuntaan ja yhteen vaakatasoon.

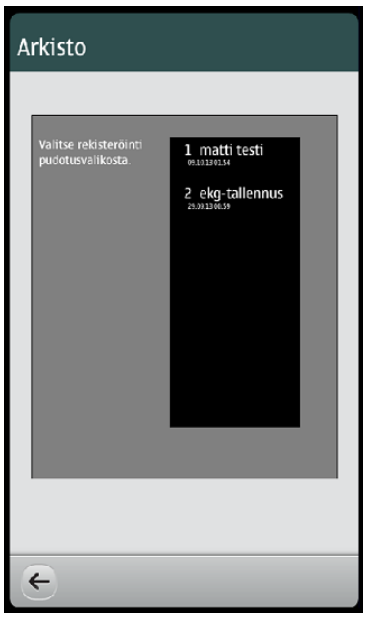
KUVA 12: Käänteinen vaakatasonäkymä.

Värimaailma voidaan käyttäjän halutessa muuntaa muun Meego/Harmattan-käyttöjärjestelmän tavoin käänteiseksi normaalitilasta. Tämä luo näyttöön suuremman kontrastin värien välille, mikä saattaa helpottaa näytön lukemista tietyissä tiloissa puhelinta käytettäessä.

5.2.4 Seurantojen hallinta

Tallennettuja seurantoja voi hallita erilliseltä Arkisto-näytöltä. Näyttöön luetteloidaan tallennetut seurannat, jotka voidaan avata selaamista tai muokkausta varten omaan näkymäänsä.

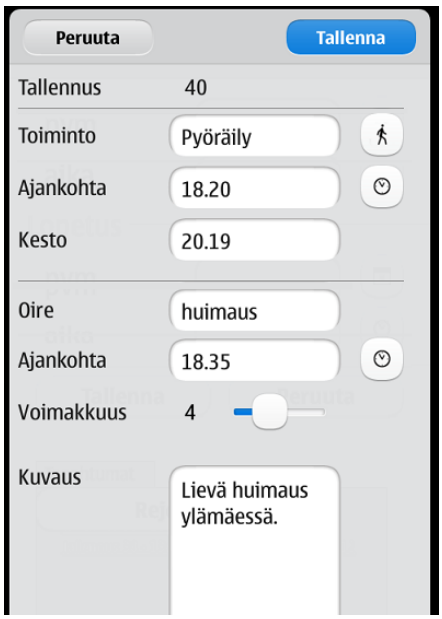
Käyttäjä valitsee haluamansa seurannan ”pudotusvalikosta” nimen ja päivämäärän perusteella (Kuva 13). Haluttu seuranta avautuu omaan ikkunaansa, jossa on seurannan perustiedot (Kuva 14). Avatessa kenttien tiedot on lukittu, mutta käyttäjä voi halutessaan avata tiedot muokattaviksi yläreunassa näkyvän kytkimen avulla.



KUVA 13: Arkisto-sivun näkymä.

Seurannan yhteydessä tapahtuneet toimintotallennukset näkyvät omasta listasta (Kuva 15). Halutun tapahtumatallennuksen voi muokata valitsemalla sen listasta, jolloin tämä avautuu uuteen näkymään. Tapahtuman tiedot on eritelty ja niihin voi tehdä muutoksia painikkeiden sekä valintaikkunoiden avulla.

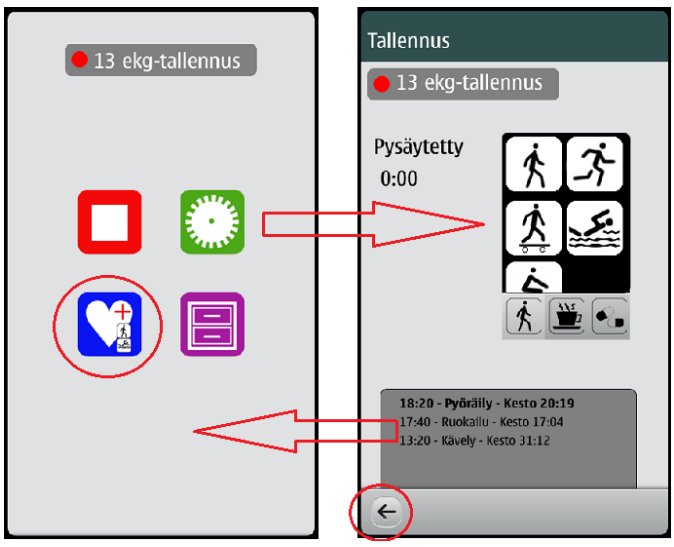
KUVA 14: Seurannan tarkasteluikkuna muokkaustilassa.



KUVA 15: Yksittäisen tapahtuman tarkasteluikkuna muokkaustilassa.

5.3 Sivut ja Liikkuminen

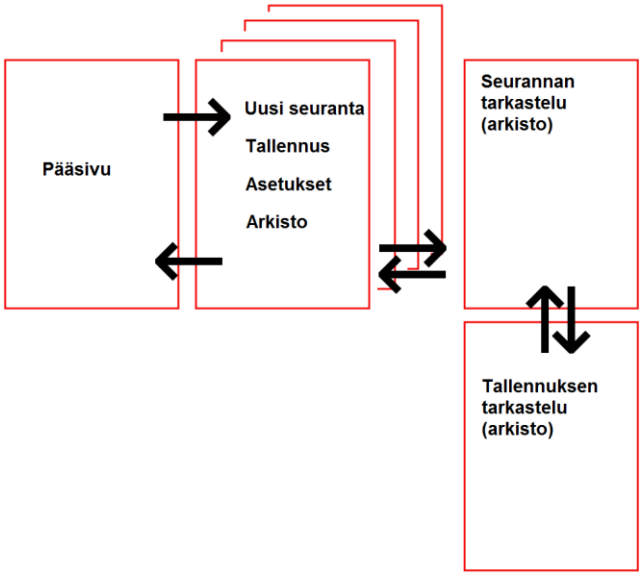
Sovelluksessa liikkuminen tapahtuu päävalikon ja toimintosivujen välillä. Päävalikosta löytyy toimintosivujen ikonit ja toimintosivuilta paluupainike päävalikkoon. Päävalikon ikoni avaa halutun toimintosivun vierittämällä kuvan oikealle (Kuva 16). Palaaminen onnistuu työkaluriviltä löytyvän nuoli vasemmalle -painikkeen avulla. Painikkeen painaminen vierittää kuvan vastaavalla vasemmalle palaten päävalikkoon. Liikkuminen sovelluksessa muistuttaakin länsimaista lukusuuntaa, vasemmalta oikealle. Tämä liike viestii myös lähtemistä, kun taas oikealta vasemmalle kertoo jonkin loppumisesta tai kotiinpaluusta (Huovila 2006, 53).



KUVA 16: Näytön liikkumisen suunta päävalikosta Tallennus-sivulle ja päinvastoin.

Kuvassa 17 on kuvattu sivujen sijainti ja liikkumissuunta. Rekisteröinti, tallennus, asetukset ja arkisto ovat käytettävissä pääsivun kautta, kun taas aikaisemmin tallennetut tiedot ainoastaan arkiston

kautta. Liikkumisen tunnetta lisää animaation kaltainen liike, joka tapahtuu sivujen vaihdon välillä. Liikkuminen on samansuuntainen, pidettiin laitetta vaaka- tai pystytasossa.



KUVA 17: HolterMeemo-sovelluksen sivukartta.

Kooditasolla edellä mainittu liikkuminen perustuu PageStackiin, eli ”sivupinoon”. Pinoon, (Stack), ”stäkkiin”, voidaan avata sivu päällimmäiseksi PageStackin metodilla push(”haluttu_sivu.qml”). Päällimmäinen sivu voidaan taas poistaa metodilla pop() (Koodi 7). Koska päävalikossa on useampi avattava sivu, on järkevää käyttää yhtä funktiota sivujen avaamiseen (Koodi 5 & 6). Samalla tarkastetaan sivun olemassaolo ennen avaamista, mikä estää sovelluksen kaatumisen, jos avattavaa sivua ei ole. (Nokia 2013d.)

```
Button {
        x: 200
        y: 50
        height: 100
        width: 100
        iconSource: "img/ratas2.png"
        onClicked: {
            openFile("Asetukset.qml");
        }
```

KOODI 5: Painikenappi kutsuu funktion openFile, joka avaa Asetukset.qml -sivun.

```
    function openFile(file) {
        var component = Qt.createComponent(file)

        if (component.status === Component.Ready)
            pageStack.push(component);
        else
            console.log("Error loading component:", component.errorString());

        /* openFile function from http://harmattan-dev.nokia.com/docs/library/html/qt-
components/qt-components-meego-pagestack.html */
    }
```

KOODI 6: Funktio QML-tiedoston avaamiseen.

```
ToolBarLayout {
        id: perustools
        visible: true
        ToolIcon { iconId: "toolbar-back"; onClicked: pageStack.pop();  }
    }
```

KOODI 7: Työkalurivin nuoli vasempaan -painike, käynnistää PageStackin metodin pop().

Tallennetun rekisteröinnin tapahtumien muokkaustila avautuu edellä mainituista poikkeavalla tavalla. Tässä sivu nousee alhaalta ylös peittäen mukanaan myös toimintorivin. Poikkeavalla tietojen avaustavalla halutaan korostaa sitä, että kyse on kyseiseen rekisteröintiin liittyvistä tiedoista, ei pelkästään uudesta sivusta. Poikkeava avaus on toteutettu koodissa Sheet-elementin avulla.

5.4 Symbolit

5.4.1 Päävalikon symbolit

Päävalikko koostuu neljästä pääsymbolista, joista jokaisesta avautuu oma toimintosivunsa (Kuva 18). Symbolit kuvaavat yhdellä-kahdella objektilla avautuvan sivun toimintoa. Kaikki neljä symbolia ovat taustaltaan eri värisiä, jolla puolestaan pyritään korostaa niiden erottavuutta, mutta myös muistettavuutta sekä länsimaista väriassosiaatiota. Myös symbolien erottuvuus on osa niiden estetiikkaa (Sinkkonen ym. 2002, 179). Toimintoa kuvaava väri on puolestaan valkoinen. Muodoltaan symbolit ovat neliöitä joissa on pyöristetyt kulmat.

KUVA 18: Päävalikon symbolit.

Uuden rekisteröinnin symboli on pohjaväriltään punainen jonka sisässä on + -merkki. Sinkkosen ym. (2002) mukaan punainen luo tehokkaan ja toimeliaan tunteen ja on huomiota herättävä väri. Värin valinta punaiseksi on perusteltua, sillä käytännössä ohjelman käyttäminen alkaa kyseisen painikkeen kautta. Plus-merkki puolestaan on matemaattinen symboli, joka kuvastaa lisäystä ja positiivista (Wikipedia 2013g.). Kun tallennus on käynnissä, muuttuu kyseinen ikoni medialaitteista tunnetuksi ”STOP”-merkiksi. (Kuva 19)

KUVA 19: Rekisteröinnin pysäytysikoni.

Toiminnon tallennuksen symboli on sydän. Sen sisään on lisätty + -merkki ja yhteyteen toimintoikoneja, jotka auttavat käyttäjää tunnistamaan symbolin. Sydän on symbolissa valkoinen muiden päävalikon symbolien mukaan. Pohjaväri symbolissa on sininen, joka länsimaalaisen väriassosiaation mukaan kuvastaa mm. suoritusta (Sinkkonen ym. 2002, 152).

Asetusten symbolissa on objektina ratas (Kuva 18). Ratas on asetuksista yleisesti käytetty symboli. Esimerkiksi Nokia N9 -puhelimen asetukset on myös kuvattu ratas-symbolilla (Kuva 20).

Kuva 20: Nokia N9 -puhelimen asetussymboli. (Nokia 2013e.)

Arkiston symboli on laatikosto, jolla halutaan kuvata säilytettävyyttä. Pohjaväriltään se on violetti. (Kuva 18)

Edellä mainittujen lisäksi päävalikossa näkyy seurannan ollessa käynnissä tallennusta symbolisoiva punainen pallo (Kuva 21). Kyseistä symbolia käytetään yleisesti tallennuksen kuvaamisessa, kuten ääni ja kuvatallentimissa. Samassa näkyy seurannan numero ja käyttäjän määrittelemä nimi.

KUVA 21: Ilmoitus seurannasta.

5.4.2 Toimintosymbolit

Toimintasymboli esittää toimintoa, tapahtumaa tai aktiviteettia, joka liittyy tallennukseen (Kuva 22). Toimintasymbolit on ennalta määritelty järjestelmään. Tällaisia ovat mm. pyöräily, juoksu, kahvitauko ja lääkkeen otto. Symboleja käytetään, jotta ne olisivat helpommin ja nopeammin huomattavissa kun käyttäjä on tekemässä tallennusta.

Sovelluksessa käytettävät toimintasymbolit ovat siluettikuvia. Siluetilla tarkoitetaan kuvaa, jossa kuvattavaan kohteeseen on piirretty uloimmat ääriviivat. Käytetyt toimintasymbolit ovat ns. puhtaita siluetteja, sillä niissä käytetään ainoastaan mustaa ja valkoista väriä. (Wikipedia 2013h.)

KUVA 22: Toimintasymbolit ja osastot.

Toiminnot ja niiden symbolit on jaettu eri osastoihin sen mukaan ovatko ne aktiviteetteja vai muita toimintoja. Osastojen valintasymbolit ovat: kävelijä, kahvikuppi ja tabletit. (Kuva 22)

5.5 Sivujen asettelu

5.5.1 Yhtenäisyys

Pääsivua lukuun ottamatta kaikkien sivujen asettelu on pyritty tekemään yhtenäiseksi. Yläreuna on väriltään harmaa ja sivun otsikkoteksti valkoinen (Kuva 23). Länsimaissa harmaa väri mielletään mm. yhtenäiseksi, arkiseksi ja turvalliseksi (Sinkkonen jne. 2002 s.154). Sivujen alareunassa on palkki, jossa sivusta riippuen on ainoastaan nuoli vasempaan -painike (Kuva 24).

Toimintosivujen otsikkotausta on tehty Rectangle-objektilla, johon on määritelty nimi, sijoittumiskoordinaatit, läpinäkyvyys, leveys sekä korkeus ja väri (Koodi 8). Rectangle-objektiin on sisällytetty Text-objekti, johon luettava teksti sijoitetaan. Harmattan versio 1.2 käyttää oletuskirjasintyyppinä Nokia Pure Text -fonttia, jolloin sitä ei tarvitse erikseen määritellä. (Nokia 2013f.)

Koodissa näkymä yhtenäiseksi muiden Meego/Harmattan 1.2 -sovellusten kanssa saadaan käyttämällä PageStackWindow -objektia. PageStackWindow tarjoaa mm. navigoinnin ja työkalupalkin. Työkalupalkki määritellään ToolBarLayoutilla (Koodi 9). Työkalupalkin vasempaan reunaan on sijoitettu nuolivasemmalle painike, joka palauttaa näkymän päävalikkoon. (Nokia 2013g.)

KUVA 23: Otsikkopalkki.

```
Rectangle {
        id: idotsikko
        x:0
        y:0
        z: 100
        width: 856
        height: 80
        color: "#2f4f4f"
        Text {
            y: 25
            x: 10
            text: "Uusi seuranta"
            color: "#ffffff"
            font.pixelSize: 32
        }
}
```

KOODI 8: Otsikkopalkin luominen sivulle koodissa.

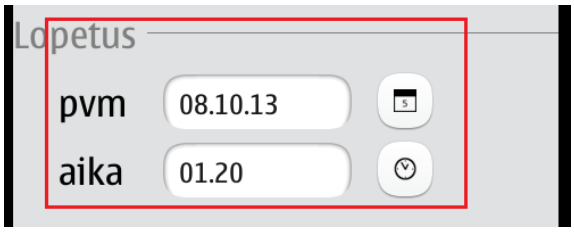
```
ToolBarLayout {
        id: commonTools
        visible: true
       ToolIcon { iconId: "toolbar-back"; onClicked: pageStack.pop();  }
    }
```

KOODI 9: Työkalupalkin määrittäminen koodissa.

KUVA 24: Työkalupalkki.

5.5.2 Komponenttien sijoittelu

Sovelluksen eri sivuilla olevien komponenttien sijoittelu on tehty hahmolakeja kunnioittaen. Esimerkiksi Uusi seuranta -sivun komponenttien, kuten otsikkotekstin, tekstikenttä ja painonappi sijoittelussa on otettu huomioon läheisyys, jolloin ne ymmärretään yhteenkuuluviksi läheisyyden hahmolain mukaan (Sinkkonen ym. 2002, 102). Edellä mainitut kentät on sijoitettu keskitetysti otsikkotekstin ja erotusviivan läheisyyteen, jolloin ne noudattavat myös sulkeutuvuuden lakia (Sinkkonen ym. 2002, 104). (Kuva 25)



KUVA 25: Hahmolaki: läheisyys

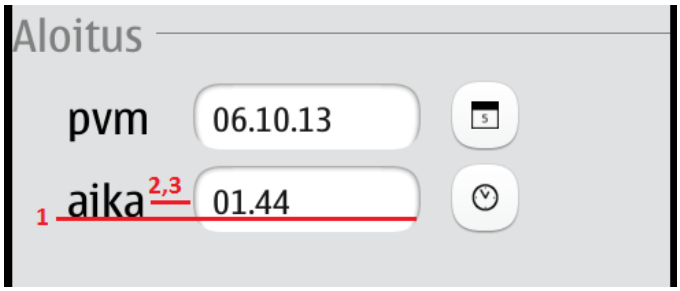
Kooditasolla asettelu perustuu ”ankkurointiin” (eng. anchor). Ankkuroinnilla määritellään komponentin sijoittuminen muihin komponentteihin nähden. Esimerkiksi Uusi seuranta -sivulla (Koodi 4) otsikko ”Kello” ja kellonaika tekstikenttä on ankkuroitu samalle tasolle vertikaalisesti toisiinsa nähden [1]. Tämän lisäksi on määritelty, että tekstikenttä on otsikkokentän oikealla puolella [2], sekä se että tekstikentän vasemmalla puolella on 10 pikseliä marginaalia [3]. Vastaavalla tavalla painike ”btnAl-

kuAika” on tasattu otsikon kanssa ja sijoitettu 20 pikselin marginaalilla tekstikentän oikealle puolelle. Kuvassa 26 näkyy miten komponentit sijoittuvat käyttöliittymässä.

Kuten koodissa 10 näkyy, on määritellyt objektit sijoitettu Item diaryRivi3 -objektin sisään. Item -objekti kerää komponentit yhden komponentin alle ilman, että se tekisi mitään graafisia muutoksia käyttöliittymään. Tämä helpottaa käyttöliittymän rakentamista, sillä ankkuroinnit voidaan tehdä suurempien kokonaisuuksien kesken. Tämä auttaa erityisesti silloin, kun halutaan vaihdella useiden komponenttien sijaintia.

```
Item {
            id: diaryRivi3
            width: 400
            height: 50
            Text { id: lblAlkuAika; text: "aika"; color: "black"; font.pixelSize: 32;
                  width: 160 x:20 }
            TextField { id: txtAlkuAika; width: 170;  text: Qt.formatTime(new Date(),
                 "hh.mm");
                anchors.verticalCenter: lblAlkuAika.verticalCenter; [1]
                anchors.left: lblAlkuAika.right; [2]
                anchors.leftMargin: 10 [3] }
            Button { id: btnAlkuAika; width: 50; iconSource: “img/kelloimage.png”;
                  anchors.verticalCenter: lblAlkuAika.verticalCenter;
                  anchors.left: txtAlkuAika.right
                  anchors.leftMargin: 20
                onClicked: { aikaValintaDialog.open(); aikaValintaDialog.visible = true
          }
        }
      }
```

KOODI 10: Otsikon ja tekstikentän ankkurointi koodissa.



KUVA 26: Otsikon ja tekstikentän ankkurointi näytöllä.

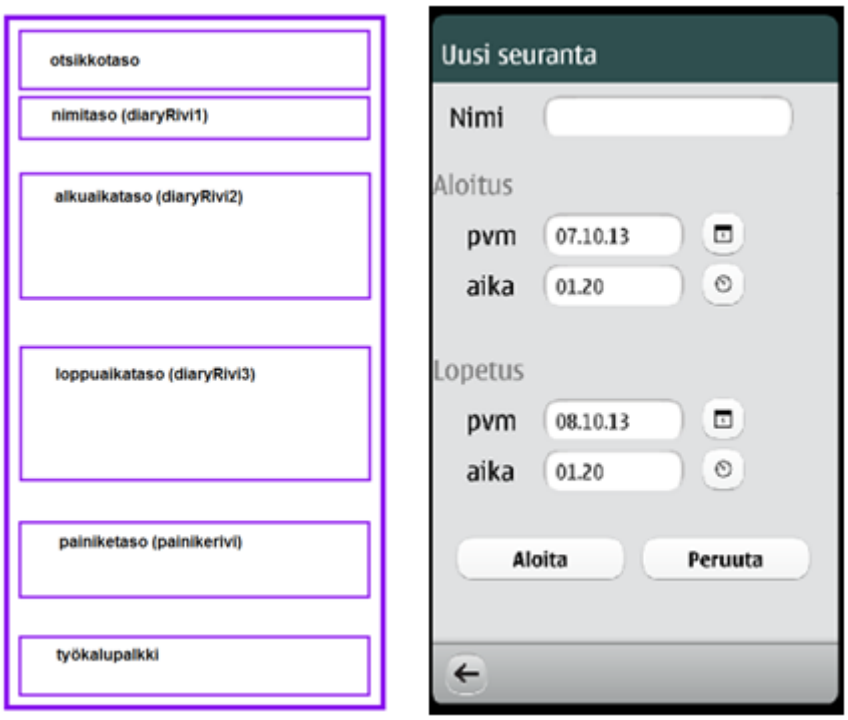
5.5.3 Komponenttien sijoittelun muuttaminen

Monet nykyaikaiset matkapuhelimet mahdollistavat puhelimen käytön niin vaaka- kuin pystytasossakin. Tiettyjen toimintojen, kuten esimerkiksi normaalien Internet-sivujen selaaminen, on vaakatasossa helpompaa. Sivut suunnitellaan yleensä käytettäväksi tietokoneilla joissa näytöt ovat leveämpiä.

HolterMeemo sovelluksessa käyttäjä voi halutessaan käyttää sovellusta myös puhelimen ollessa vaakatasossa. Tällöin sovellus muuttaa asetteluaan näytölle sopivaksi ilman, että käyttäjä esimerkiksi menettäisi tietojaan. Näyttö siis mukautuu aina sen mukaisesti, kuinka käyttäjä pitää laitettaan kädessään. Jos uudelleenasettelua ei tehtäisi, se vaikuttaisi negatiivisesti käytettävyyteen. Näytön

komponentit sijaitsisivat ainoastaan toisessa reunassa, jolloin mm. asettelun tasapaino kärsisi (Sinkkonen ym. 2002, 177.) Käyttäjä myös joutuisi vierittämään entistä kapeammaksi käynyttä näyttöä entistä enemmän.

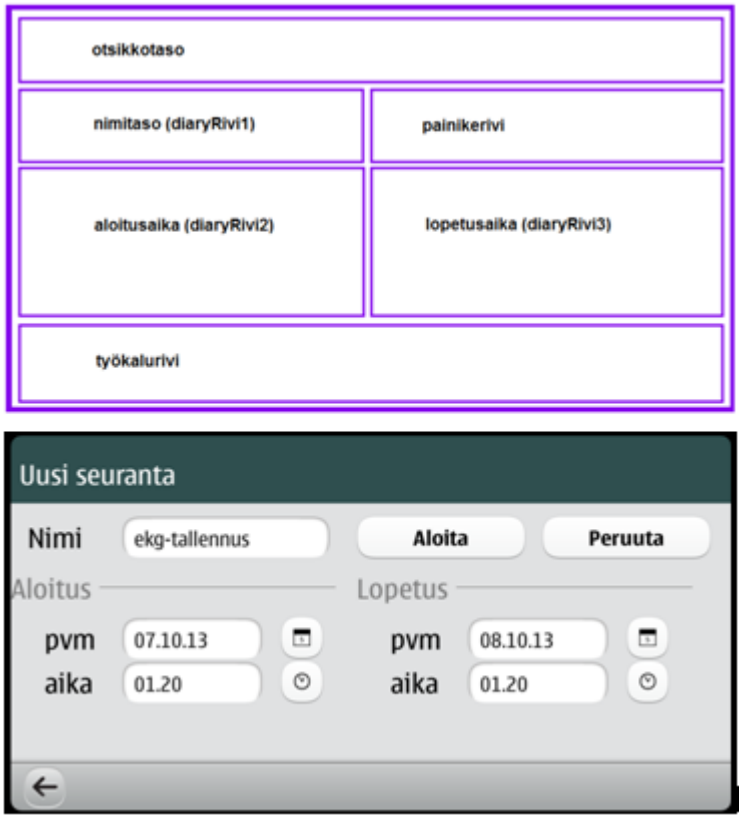
Kuvassa 27 näkyy kaavakuva ja näyttökuva Uusi seuranta -sivulta. Kaavakuvasta selviää, että suurin osa sivun komponenteista on sijoitettu muutamaan suurempaan Itemiin. Ainoastaan otsikkokenttä ja työkalukenttä ovat itsenäisiä komponentteja. Kun näyttöä käännetään, painikerivi ja rekisteröinnin lopetus Item vaihtavat paikkaa. Lopetustietojen tekstikenttää ja painikerivin painikkeita kavennetaan. Näin käyttäjän ei tarvitse myöskään vaakanäkymässä rullata näyttöä.



KUVA 27: Kaavakuva ja käyttöliittymän näkymä Uusi seuranta -sivulta pystytasossa.

Meego/Harmattan 1.2 mahdollistaa vaakatason käytön State-objektin avulla. (Nokia 2013h.) Koodissa asetusmuutos määritellään PropertyChanges -objektilla State -objektiin. PropertyChanges -objektiin määritellään kohde (target) eli mitä komponenttia halutaan muuttaa ja tämän jälkeen määritellään mitkä ovat kyseisen komponentin uudet ominaisuudet uudessa tilassa. (Koodi 11)

Koska vaakasuora näyttö muuttaa näytön kokoa leveämmäksi, mutta myös lyhyemmäksi, on järkevää tehdä komponenttien sijoitusmuutoksia, mutta myös ominaisuusmuutoksia (Kuva 28). Sijoitusmuutos tehdään AnchorChanges -objektilla. Myös AnchorChanges -objektin kohdalla tarvitaan kohde, jonka sijoittamista ollaan muuttamassa sekä uudet määritettävät tiedot. Ankkurointi ei kuitenkaan onnistu QML-rakenteessa sellaisten komponenttien kesken, jotka eivät ole samalla tasolla, ts. sisaruksia (siblings) tai periytyviä (parent - child) (KDE TechBase 2013). Tällainen on esimerkiksi painikerivi, jota ei voida ankkuroida otsikkotasoon, mutta voidaan ankkuroida nimitason itemin ja lopetusaika itemin. Ankkurointia voidaan siis käyttää myös käänteisesti määrittelemällä marginaaliarvo negatiiviseksi, joka siirtää komponentteja oletetusta suunnasta poispäin. (Koodi 11)



KUVA 28: Kaavakuva ja käyttöliittymän näyttö Uusi seuranta -sivulta vaakatasossa.

```
state: (screen.currentOrientation === Screen.Portrait) ? "portrait" : "landscape"
    states: [
            State {
            name: "landscape"
                PropertyChanges {
                    target: diaryRivi1
                    height: 30
                    width: 300
                }
                AnchorChanges {
                    target: painikeRivi
                    anchors.left: diaryRivi4.left
                    anchors.top: diaryRivi4.top
                }
                PropertyChanges {
                    target: painikeRivi
                    anchors.leftMargin: 0
                    anchors.topMargin: -125
                }
                AnchorChanges {
                    target: diaryRivi4; anchors.left: diaryRivi2.right;
                }
                PropertyChanges {
                    target: diaryRivi4
                   anchors.leftMargin: 10
                }
                AnchorChanges {
                    target: diaryRivi5; anchors.left: diaryRivi3.right
                    anchors.top: diaryRivi4.bottom
                }
                PropertyChanges {
                    target: diaryRivi5; anchors.topMargin: 2;
                    anchors.leftMargin: 10
                }
      }
```

KOODI 11: Komponenttien ankkurointi- ja ominaisuusmuutoksia vaakatasossa.

5.6 Käyttäjän huomion kiinnittäminen

5.6.1 Ilmoitukset

Sovelluksessa käytetään kahdentyyppisiä ilmoituksia, joilla käyttäjälle kerrotaan toiminnoista ja tapahtumista. Eri toimintojen yhteydessä käytetään ns. InfoBanner-ilmoituksia ja käyttäjän hyväksymiseen dialog-ilmoituksia.

InfoBanner–ilmoituksia käytetään tilanteissa, joissa käyttäjää autetaan muistamaan jo tehty toiminto. Ilmoitukset eivät ole tyyliltään varoituksia tai virheilmoituksia, vaan kyseessä on enemmänkin huomautus. Esimerkiksi käyttäjä pysäyttää toiminnan tallentamisen ja hän saa vielä ilmoituksen ”Tallennus on pysäytetty”. InfoBanner-ilmoitus näkyy ruudulla muutaman sekunnin ja häviää itsestään, eikä siis vaadi erityisesti käyttäjältä hyväksyntää.

Kooditasolla InfoBanner määritellään omalla objektillaan. Myös InfoBanner-objekti sisältää mahdollisuuden siirtää se sopivalle paikalle ruudulla (kuva 29).

KUVA 29: InfoBanner-ilmoitus.

```
InfoBanner { id: ibTalPys
             text: "Tallennus on pysäytetty"
             x: 0
             y: 20 }
```

KOODI 12: InfoBanner määritelty koodissa.

Dialog-ilmoituksia käytetään virheen ilmoittamiseen tai merkittävän toiminnon varmistamiseen. Tällä pyritään siihen, että käyttäjän on reagoitava ilmoitukseen. Ilmoitusta tehostetaan symbolilla. Dialog-ilmoitus on koko näyttöpinnan kokoinen ja se estää alla olevien toimintakomponenttien käytön. Kuvassa 30 on esimerkki virheilmoituksesta.

Virheilmoituksen dialogissa koodiin (Koodi 13) määritellään viestien, otsikon ja hyväksymispainikkeen lisäksi myös ikoni. Koodissa 14 on puolestaan esimerkki dialogissa tapahtuvasta funktion kutsumisesta. Kyseisessä dialogissa (koodi 14) päivämäärän määrittämisen hyväksyminen käynnistää lisaaPvm() -funktion, joka puolestaan sijoittaa arvot tekstikentille. Kuvassa 31 on päivämäärän valintadialog-ruutu. Päivämäärän valintadialog–komponentti kuuluu com.nokia.extras 1.1 kirjastoon.

KUVA 30: Ilmoitus tietokannan tallennuksessa ilmestyneestä virheestä.

```
QueryDialog {
        id: tallennusKeskeytysDialog
        icon: "image://theme/icon-l-error"
        titleText: "Virhe tallennuksessa!"
        message: "Tietokantaan tallennuksessa on virhe."
        acceptButtonText: "OK"
    }
```

KOODI 13: Virheilmoitusdialogin koodi.

KUVA 31: Päivämäärän lisäys tapahtuu dialogina.

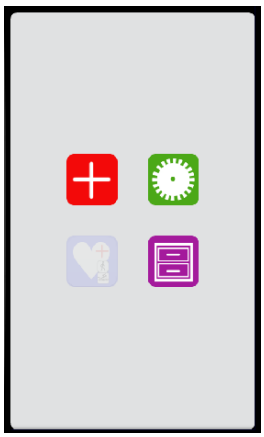
```
    DatePickerDialog {
        id: pvmValintaDialog
        visible: false
        titleText: "Valitse päivämäärä"
        onAccepted: lisaaPvm();
        acceptButtonText: "Lisää"
        rejectButtonText: "Peruuta"
    }
```

KOODI 14: Päivämäärälisäys

5.6.2 Komponenttien aktiivisuuden muutokset

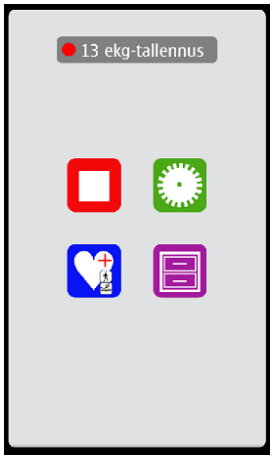
Komponenttien aktiivisuuden muutoksilla voidaan käyttäjää ohjata tekemään asioita oikein. Käyttäjältä voidaan hävittää tarpeettomat komponentit näkyvistä tai komponentit voidaan lukita ja visuaalista ilmettä muuttaa.

Ohjelman käynnistämisen jälkeen, ennen seurannan aloittamista Tallennus-toiminto on poissa käytöstä (Kuva 32). Himmeällä näkyvä painike on lukittu, ja seurannan aloittamisen jälkeen se kirkastuu ja on käytettävissä (Kuva 33). Kun rekisteröinti on lopetettu, palaa Tallennus-painike lukittuun tilaan.



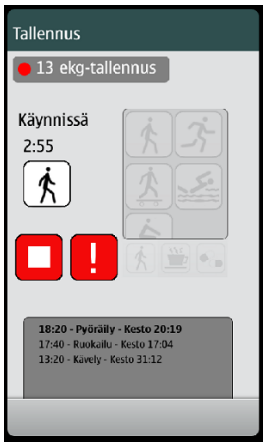
KUVA 32: Tallennus-toiminnon käyttäminen on estetty ennen seurannan aloittamista.

Seurannan aloittamisen jälkeen uuden seurannan -painike on muuttunut STOP-painikkeeksi (Kuva 33). Näin estetään käyttäjää aloittamasta useampaa seurantaa päällekkäin.



KUVA 33: Päänäytön muutokset seurannan aloittamisen jälkeen.

Tapahtuman tallennuksen ollessa käytössä työkalurivin nuolipainike hävitetään. Samalla muut toimintopainikkeet, mukaan lukien toimintojen osastopainikkeet lukitaan ja himmennetään. (Kuva 34)



KUVA 34: Sovellus on tallennustilassa.

Kooditasolla tämä toteutetaan muuttamalla kahta arvoa ”pohjalevylle”, johon toimintopainikkeet on sijoitettu. Enabled-arvolla määritetään painikkeiden toimintavalmius ”pohjalevyllä” ja opacityllä puo-

lestaan läpinäkyvyys. Läpinäkyvyyden pienentäminen näyttää käyttäjälle painikkeiden himmentämisenä. (Koodi 15)

```
Rectangle {  id: ikonitPohja
             anchors.fill: parent
             color: "black"
             radius: 10
             opacity: 0.1
             enabled: false
…
```

KOODI 15: Toimintopainikkeiden himmennys koodissa.

Tallennus-sivun valitun osaston aktiivisuus osoitetaan vaaleammalla taustalla kuin muiden painikkeiden. (Kuva 35)

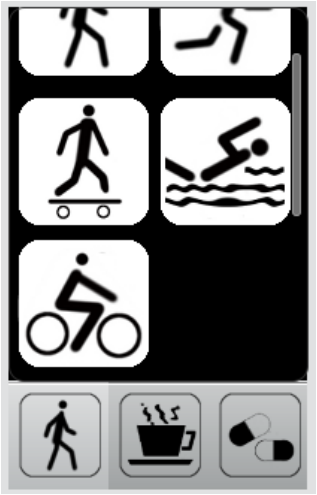
KUVA 35: Valitun osaston osoittaminen vaaleammalla taustavärillä.

5.7 Muut matkapuhelimeen liittyvät erityispiirteet ja niiden huomiointi

5.7.1 Yhdellä kädellä käyttäminen

Pääsivun ”nelikenttä” on sijoitettu keskelle näyttöä. Symbolit ovat suhteellisen lähellä toisiaan, jolla on pyritty yhtenäisyyteen, mutta myös siihen, että käyttäjän on helppo tehdä valinta kosketusnäytöltä yhdellä kädellä esimerkiksi peukaloa käyttäen. Peukalon liike on suhteellisen pieni verrattuna siihen mitä se olisi, jos symbolit olisivat kauempana toisistaan.

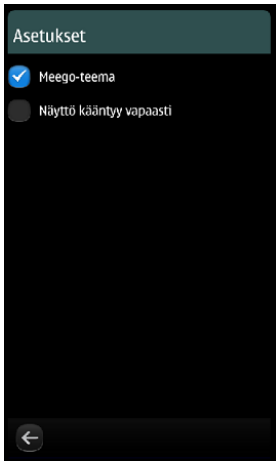
Tallennus-sivulla olevat symbolit on sijoitettu kahteen riviin sivuun oikeaan laitaan. Symboleja voidaan kuitenkin liikutella ylösalas tarpeen mukaan toimintoa vastaavan symbolin löytämiseksi. Sijoittelussa on huomioitu puhelimen käyttäminen yhdellä kädellä ja yhdellä sormella, jolloin sopiva symboli voidaan ”rullata” esiin valittavaksi (Kuva 36).



KUVA 36: Toimintasymbolien ”rullaus” ja sitä hahmottava vaalea sijaintipalkki.

5.7.2 Käyttö eri valaistuksissa

Sovellus sisältää kaksi eri väri teemaa, jotka poikkeavat toisistaan. Oletuksena käytettävä harmaa-taustainen teema ei kuitenkaan välttämättä erotu näytöstä kaikissa tilanteissa. Myöskään puhelimen mahdollistama näytön valaistuksen säätäminen ei välttämättä korjaa tilannetta. Käyttäjä voikin vaihtaa teemaa tarvittaessa suurempikontrastiseen Meego-teemaan (Kuva 36). Näin pohjaväri muuttuu mustaksi ja oletusteksti valkoiseksi.



KUVA 36: Meego-teeman valinta Asetukset-sivulta.

6 POHDINTA

Lähtökohtana opinnäytetyössä oli luoda sovellus Meego-käyttöjärjestelmää käyttävälle matkapuhelimelle. Tarkoituksena oli selvittää sovelluskehityksen prosessia ja eri vaiheita sekä käyttöliittymän rakentamista Linux -pohjaiselle matkapuhelimelle.

Opinnäytetyön suunnittelu alkoi syksyllä 2012, jolloin aloin tutustua Meego-sovelluksen sovelluskehitykseen ja käytettäviin työvälineisiin. Ideaa millainen sovellus olisi tai mitä se tekisi, ei vielä tuossa vaiheessa ollut. Varsinainen työ alkoi keväällä 2013, jolloin myös ajatus sovelluksesta konkretisoitui EKG-seurannan päiväkirjasovellukseksi.

Ohjelmointiprosessi Meego-alustalle oli suhteellisen haastava. Nokian ilmoitus jättää Meego ja siirtyä Windows-käyttöjärjestelmään vaikeutti omalta osaltaan opinnäytetyön tekemistä, joka näkyi erityisesti sovellusten ja palvelujen ylläpitoprioriteetissa. Esimerkiksi vuoden 2013 aikana Nokian Harmattan Developer Library -sivusto oli useita kertoja poissa käytöstä. Sovelluskehityksessä tarvittavaa ristiinkääntötyökalua ei voinut enää asentaa normaalilla tavalla suoraan Internetistä. Myöskään uusinta versiota sovelluskehittimestä ei voinut käyttää mahdollisten yhteensopivuusongelmien vuoksi. Toisaalta myös itselle tuli yllätyksenä hyvien ohjelmointiratkaisujen löytäminen, esimerkiksi sovelluksessa käytettävä tietojen tallennustapa.

Opinnäytetyöprosessin aikana opin paljon yleisesti sovelluskehityksestä. Ohjelmointikielistä mainittakoon QML ja Javascript, joista ainoastaan Javascriptiä olin käyttänyt aikaisemmin hyvin vähän. Myös opinnäytetyössä käytetyt työvälineet tulivat hyvin tutuksi. Opinnäytetyöprosessin aikana opittuja taitoja voi varmasti hyödyntää jatkossa erilaisissa sovelluskehitysprojekteissa.

LÄHTEET

BLANCHETTE, Jasmin ja SUMMERFIELD, Mark. 2006. C++ Gui Programming with Qt 4. Stoghton, Massachussetts: Prentice Hall.

BURROWS, Peter. 2011-06-02. Stephen Elop's Nokia Adventure. Bloomberg Businessweek. [viitattu 2013-10-08]. Saatavissa: http://www.businessweek.com/magazine/content/11_24/b4232056703101.htm

DIGIA 2012-08-09. Digia ostaa koko Qt-kehitysympäristön Nokialta. [tiedote] Saatavissa: http://www.epressi.com/tiedotteet/tietokoneet/digia-ostaa-koko-qt-kehitysympariston-nokialta.html

DIGIA. 2013a. Qt About us. [viitattu 2013-04-20]. Saatavissa: http://qt.digia.com/About-us/

DIGIA. 2013b. Qt Code Features - Functions. [viitattu 2013-09-07]. Saatavissa: http://qt.digia.com/Product/Qt-Core-Features--Functions/

DIGIA. 2013c. Introduction of Qt Quick. [viitattu 2013-08-24]. Saatavissa: http://doc.qt.digia.com/4.7/qml-intro.html

DIGIA. 2013d. Qt Quick. [viitattu 2013-08-10]. Saatavissa: http://qt-project.org/doc/qt-5.0/qtquick/qtquick-index.html

DIGIA. 2013e. Qt Core Features & Functions - Developer tools. [viitattu 2013-08-30]. Saatavissa: http://qt.digia.com/Product/Qt-Core-Features--Functions/Developer-Tools/

DIGIA. 2013f. QML Global Object. [viitattu 2013-08-04]. Saatavissa: http://doc.qt.digia.com/4.7/qdeclarativeglobalobject.html#offline-storage-api

DIGIA. 2013g. Qt Quick Local Storage QML Types. [viitattu 2013-07-19]. Saatavissa: http://qt-project.org/doc/qt-5.0/qtquick/qmlmodule-qtquick-localstorage2-qtquick-localstorage-2.html

DIGIA. 2013h. Connecting Symbian Devices. [viitattu 2013-10-07]. Saatavissa: http://doc.qt.digia.com/sdk-1.2/creator-developing-symbian.html

DR. DOBBS. Trolltech Now 'Qt Software'; Launches Qt Extended. [viitattu: 2013-08-13]. Saatavissa: http://www.drdobbs.com/tools/trolltech-now-qt-software-launches-qt-ex/210604690

GSMARENA. Nokia N9 Full Specification. [viitattu: 2013-07-13]. Saatavissa: http://www.gsmarena.com/nokia_n9-3398.php

HADDAD, Ibrahim. 2011. Happy 1 Year Anniversary, MeeGo! [viitattu: 2013-10-06]. Saatavissa: http://meego.com/sites/all/files/users/u19961/meego_anniversary_article.pdf

HOLWERDA, Thom. 2010-10-21. Qt 4.7.0 Released. OSnews. [viitattu: 2013-09-20]. Saatavissa: http://www.osnews.com/story/23828

HUOVILA, Tapani. 2006. ”Look” - visualisoi viestisi. Helsinki: Inforviestintä Oy.

KDE TECHBASE. QML Getting Started. [viitattu 2013-09-07]. Saatavissa: http://techbase.kde.org/Development/Tutorials/Plasma/QML/GettingStarted

KNOLL, Lars. 2013-07-03. Qt 5.1 Released [blogi] [viitattu 2013-09-08]. Saatavissa: http://blog.qt.digia.com/blog/2013/07/03/qt-5-1-released/

KNUUTILA, Timo 1998-01-21. Deklaratiivinen ohjelmointi. [viitattu 2013-09-14]. Saatavissa: http://www.cs.utu.fi/knuutila/okp98/node130.html

KURRI, Sampsa. 2012-10-10. Nokia Meegon Tarina. [viitattu 2013-08-10]. Saatavissa: http://taskumuro.com/artikkelit/nokia-meegon-tarina

KUUTTI, Wille. 2003. Käytettävyys, suunnittelu ja arviointi. Saarijärvi: Talentum.

KWANG, Kevin. 2012-07-09. Finnish startup breathes new life into Meego. ZDNet. [viitattu 2013-08-19]. Saatavissa: http://www.zdnet.com/finnish-startup-breathes-new-life-into-meego-7000000446/

LINUX FOUNDATION, THE. Meego FAQ. [viitattu 2013-10-04]. Saatavissa: https://meego.com/about/faq

MANKINEN, Veli ja RAHKONEN, Valtteri. 2005. Cross-Compiling tutorial with Scratchbox [viitattu 2013-07-10] Saatavissa: http://www.scratchbox.org/documentation/user/scratchbox-1.0/html/tutorial.html

NIELSEN, Jakob. Usability 101: Introduction to Usability. 2012-04-01. [viitattu 2013-10-08]. Saatavissa: http://www.nngroup.com/articles/usability-101-introduction-to-usability/

NOKIA. 2011a. Nokia strategy 2011. [viitattu 2013-08-19]. Saatavissa: http://conversations.nokia.com/nokia-strategy-2011/

NOKIA. 2013a. Introduction to QML for Web developers. [viitattu 2013-09-29]. Saatavissa: http://developer.nokia.com/Community/Wiki/Introduction_to_QML_for_Web_developers

NOKIA. 2013b. Qt SDK. [viitattu 2013-08-14] Saatavissa: http://developer.nokia.com/Community/Wiki/Qt_SDK

NOKIA. 2013c. Nokia N9 -puhelin [digikuva]. Saatavissa: http://developer.nokia.com/Devices/MeeGo/

NOKIA. 2013d. QML PageStack Element. [viitattu: 2013-06-20]. Saatavissa: http://harmattan-dev.nokia.com/docs/library/html/qt-components/qt-components-meego-pagestack.html

NOKIA. 2013e. Nokia N9 -puhelimen asetukset ikoni. [kuva]. Saatavissa: Qt SDK 1.2.1.

NOKIA. 2013f. Font Sizes in different Harmattan releases. [viitattu: 2013-06-30]. Saatavissa: http://harmattan-dev.nokia.com/docs/library/html/guide/html/Developer_Library_Best_practices_for_application_development_Font_sizes_in_different_Harmattan_releases.html

NOKIA. 2013g. Simple Tutorial. [viitattu: 2013-06-19]. Saatavissa: http://harmattan-dev.nokia.com/docs/library/html/qt-components/qt-components-meego-simpletutorial.html

NOKIA. 2013h. Controlling rotation. [viitattu: 2013-08-22]. Saatavissa: http://harmattan-dev.nokia.com/docs/library/html/guide/html/Developer_Library_Developing_for_Harmattan_Controlling_rotation.html

ORLOWSKI, Andrew. 2011-02-11. It's official: Nokia bets on Microsoft for Smartphones. The Register. [viitattu 2013-08-23]. Saatavissa: http://www.theregister.co.uk/2011/02/11/nokia_microsoft_smartphone_agreement/

QUIM, Gil. 2010-02-16. Renaming "Maemo 6" to MeeGo / Harmattan. [keskusteluviesti] [viitattu 2013-08-23]. Saatavissa: http://talk.maemo.org/showpost.php?p=529073&postcount=14

RODIN, Josip, AOKI, Osamu, SMALL, Graig ja HERTZOG, Raphaël. 1998. Debian New Maintainers' Guide. version 1.2.32-svn. [viitattu: 2013-08-10]. Saatavissa: http://www.debian.org/doc/manuals/maint-guide/dreq.en.html#rules

PAUL, Ryan. 2009-02-22. Nokia ends official support for Qt Java port. Ars Technica. [viitattu 2013-08-13]. Saatavissa: http://arstechnica.com/information-technology/2009/02/nokia-ends-official-support-for-qt-java-port/

PITKÄNEN, Manu. 2011-06-21. Nokia julkaisi toisen Meego-puhelimen - Nokia N950:n. Puhelinvertailu. [viitattu 2013-09-11]. Saatavissa: http://www.puhelinvertailu.com/uutiset.cfm/2011/06/21/nokia_julkaisi_toisen_meego-puhelimen_nokia_n950_n

SAADEBRA, Humberto. 2012-07-07. Core Nokia Meego Team Annouces New Company in Jolla. PhoneNews. [viitattu [2013-09-02]. Saatavissa: http://www.phonenews.com/core-nokia-meego-team-announces-new-company-in-jolla-20662/

SAYER, Peter. 2008-01-28. Nokia to Buy Software Developer Trolltech for $153M. TechHive. [viitattu 2013-08-13]. Saatavissa: http://www.techhive.com/article/141871/article.html

SAILFISH. Porting from Meego Harmattan to Sailfish OS. [viitattu 2013-09-01]. Saatavissa: https://sailfishos.org/wiki/Porting/Harmattan

SAJARI, Petri. 2013-05-20. Jollan puhelin tulee myyntiin loppuvuonna. Helsingin Sanomat. [viitattu 2013-10-03]. Saatavissa: http://www.hs.fi/talous/a1369013989643

SCRATCHBOX. [viitattu 2013-07-10] Saatavissa: http://www.scratchbox.org/

SHARMA, Dinesh C. 2005. Nokia debuts Linux-based Web device. Cnet. [viitattu 2013-04-21]. Saatavissa: http://archive.is/heJh

SINKKONEN, Irmeli, KUOPPALA, Hannu, PARKKINEN, Jarmo, VASTAMÄKI, Raino. 2002. Käytettävyyden psykologia. Helsinki: Edita Oyj/IT Press.

SIRKIÄ, Sami. 2012-02-20. BlackBerryn voi poimia nyt myös Suomesta. MBNet. [viitattu 2013-06-29]. Saatavissa: http://www.mbnet.fi/artikkeli/ajankohtaiset/uudet_tuotteet/blackberryn_voi_poimia_nyt_myos_suomesta

SWEET, David et al. 2001. KDE 2.0 Development. Sams Publishing. [viitattu 2013-09-01]. Saatavissa: http://openbooks.sourceforge.net/books/kde20devel/ch03lev1sec4.html

THELIN, Johan. 2007. Foundations of Qt Development. APress. New York.

VAN DONDEREN, Casper 2010. QCOMPARE(HTML5, QML) A comparison of UI development technologies. [thesis]. Hogeschool Utrect.

YTJ. Jolla Oy. Yritysrekisteritieto. [viitattu 2013-10-06]. Saatavissa: http://www.ytj.fi/yritystiedot.aspx?yavain=2319714&tarkiste=09432FE8912F6D266B1530B16BEE9BD166AE2AD2&path=1547;1631;1678

WIKIPEDIA. 2013a. Meego. [viitattu 2013-10-03]. Saatavissa: http://en.wikipedia.org/wiki/MeeGo

WIKIPEDIA. 2013b. Moblin. [viitattu 2013-04-20]. Saatavissa: http://en.wikipedia.org/wiki/Moblin

WIKIPEDIA. 2013c. Mer and mobile operating sysystem. [kaavio] Saatavissa: http://en.wikipedia.org/wiki/File:Mer_and_mobile_operating_systems.svg

WIKIPEDIA. 2013d. Tizen. [viitattu 2013-07-24]. Saatavissa: http://en.wikipedia.org/wiki/Tizen

WIKIPEDIA. 2013e. Package management system. [viitattu 2013-10-05]. Saatavissa: http://en.wikipedia.org/wiki/Package_management_system

WIKIPEDIA. 2013f. SQLite. [viitattu 2013-07-29]. Saatavissa: http://en.wikipedia.org/wiki/SQLite

WIKIPEDIA. 2013g. Plus and minus signs. [viitattu 2013-07-14]. Saatavissa: http://en.wikipedia.org/wiki/Plus_and_minus_signs

WIKIPEDIA. 2013h. Siluetti. [viitattu 2013-07-28]. Saatavissa: http://fi.wikipedia.org/wiki/Siluetti