Timo Haapio

ThereHome-käyttöliittymän suunnittelu ja toteutus

Metropolia Ammattikorkeakoulu

Insinööri (AMK)

Tietotekniikka

Insinöörityö

6.5.2014

Metropolia

Tekijä(t) Otsikko Sivumäärä Aika	Timo Haapio ThereHome-käyttöliittymän suunnittelu ja toteutus 52 sivua + 2 liitettä 6.5.2013
Tutkinto	Insinööri (AMK)
Koulutusohjelma	Tietotekniikka
Suuntautumisvaihtoehto	Ohjelmistotekniikka
Ohjaaja(t)	COO Toni Sormunen Lehtori Ilpo Kuivanen

Tämä ohjelmistotekniikan insinöörityö tehtiin There Corporation Oy:lle. Työn tavoitteena on ollut kehittää JavaScript-pohjainen käyttöliittymä, jossa käyttäjät pystyvät itse määrittelemään osan käyttöliittymästä dynaamisesti ja luomaan itse omanlaisensa näkymät.

Työssä suunnittelimme, mitä toiminnallisuuksia käyttöliittymässä tulisi olla ja miltä sen tulisi näyttää. Aluksi toteutimme ThereHome Mobilen, missä oli perustoiminnallisuus Theren mökkipakettiin. Tämän jälkeen tein käyttäjäpalautetutkimuksen, ja sen pohjalta teimme muutoksia ja parannuksia ThereHome-käyttöliittymään.

Tällä hetkellä ThereHome toimii pääkäyttöliittymänä kaikissa Theren tuotteissa. Käyttöliittymässä on toiminnallisuus kaikille Theren kuudelle tuotteella sekä sisältää teemat eri asiakkaille.

Keskeneräisiä toiminnallisuuksia ovat laitteiden hallinta, widgettien hallinta ja palvelut-osio. Näiden valmistuttua kaikki suunniteltu toiminnallisuus on toteutettu ThereHome-käyttöliittymään.

Työn lähdeaineisto perustuu Theren itse tuottamiin materiaaleihin.

Avainsanat	There Corporation, JavaScript, käyttöliittymä, käyttäjäpalaute tutkimus

Author(s) Title Number of Pages Date	Timo Haapio Planning and implementing ThereHome user interface 52 pages + 2 appendices 6 May 2014
Degree	Bachelor of Engineering
Degree Programme	Information Technology
Specialisation option	Software engineering
Instructor(s)	Toni Sormunen, CDO Ilpo Kuivanen, Senior Lecturer

This thesis was created for There Corporation Oy. The purpose of the thesis was to create a JavaScript based user-interface, where user can modify part of it dynamically and create unique views.

In this thesis we designed what functionalities the user-interface should have and look like. First we created a ThereHome Mobile with basic functionalities to the Holiday home package. After that i organised usabily tests and based of that, we made changes and improvements to the ThereHome user-interface.

Currently ThereHome is the main user-interface in all There's products. User-interface has funtionalities to all There's products and themes for all customers.

Unfinished functionalities are device management, widget editor and services section. When those are ready, all planned functionalities are implemented in ThereHome.

The source material is based on There's self-produced materials.

Keywords	There Corporation, JavaScript, User-interface, Usability test

Sisällys

Lyhenteet

ThereGate	ThereGate on keskusyksikkö, joka mahdollistaa mm. erilaisten laitteiden ohjaamisen ja tietojen lukemisen muutamien eri teknologioiden avulla.
THM	ThereHome Mobile. Ensimmäinen versio JavaScript-pohjaisesta käyttöliittymästä.
TH	ThereHome. JavaScript-pohjainen käyttöliittymä.
THF	ThereHome Framework. Javascript-pohjainen alusta, jonka toiminnallisuuksia TH käyttää.
Mini	Yksinkertainen mobiilikäyttöliittymä.
TG-Pages	PHP-pohjainen käyttöliittymä.
UIDB	Käyttäjäkohtainen käyttöliittymätietokanta, jonka perusteella koti-osion widgetit esitetään.
P-Device	Fyysinen laite, joka on liitetty jollain tuetulla teknologialla keskusyksikköön.
L-Device	Esittää fyysisen laitteen toiminnallisuudet sekä toimii rajapintana palvelun ja fyysisen laitteen välillä.
I-Device	Palvelu sisältää fyysisen laitteen toiminnallisuudet, tai vaihtoehtoisesti se voi olla virtuaalinen palvelu.
PB-http	Rajapinta, jonka avulla pystyy autentikoitumaan ja käyttämään järjestelmää.
TD	Teknologia-ajuri. Teknologia, jolla fyysiset laitteet ovat yhteydessä keskusyksikköön.
Z-Wave	Z-Wave on langaton teknologia, joka on suunniteltu kotiautomaatioon ja erityisesti laitteiden ohjaamiseen ja niiden tietojen lukemiseen.

ZigBee	ZigBee on langaton teknologia, joka on suunniteltu kotiautomaatioon ja erityisesti laitteiden ohjaamiseen ja niiden tietojen lukemiseen.
M-Bus	Meter-Bus on langallinen teknologia, joka mahdollistaa laitteiden ohjaamisen ja niiden tietojen lukemisen.
MVC	Model View Controller. Ohjelmointiarkkitehtuuri.
JSON	JavaScript Object Notation. Tiedonsiirtomuoto.
HTML	Hypertext Markup Language. Hypertekstin merkintäkieli.
AJAX	Asynchronous JavaScript And XML. JavaScript-tekniikka web-sivujen päivittämiseen.

1 Johdanto

ThereHome-projektin tavoitteena on ollut kehittää uuden sukupolven käyttöliittymä, jossa käyttäjät pystyvät itse määrittelemään ison osan käyttöliittymästä dynaamisesti ja luomaan itse omanlaisensa näkymät. Tavoitteena on myös saada mahdollisimman suuri yhteensopivuus eri päätelaitteiden kesken siten, että samaa järjestelmää voitaisiin käyttää mahdollisimman suuresta päätelaitejoukosta (iPhone, iPad, Android, MeeGo, Windows) puhelimista, tableteista ja PC-laitteista.

Aloitimme kesällä 2011 ThereHome Mobile -käyttöliittymän kehittämisen, johon toteutin termostaatit-widgetin, viestit-osion sekä muita erilaisia toiminnallisuuksia. Joulukuussa 2011 järjestin käyttöliittymästä käytettävyystutkimuksen, jonka pohjalta aloimme suunnittelemaan ThereHome-käyttöliittymää. Itse olin mukana sen suunnittelussa ja toteuttamassa erilaisia toiminnallisuuksia.

2 Kotiautomaatio

Kotiautomaatio tarkoittaa kodin automatisointia, eli normaalisti kohteeseen asennetaan laitteita, joiden avulla voi seurata ja ohjata kodin eri asioita, kuten ilmalämpöpumppuja, valoja tai vaikka ohjata kodin lämpötilaa. Laitteiden ohjaaminen ja arvojen seuraaminen on yleensä mahdollista tietokoneella, puhelimella, tabletilla tai kotiin asennetun näytön avulla.

Kotiautomaation tarkoitus on tarjota kodin omistajalle lisämukavuutta ja mahdollisuus säästää esim. lämmityskustannuksissa.

There Corporation on keskittynyt kotiautomaatiossa seuraaviin asioihin [4, s. 8.]:

- tuntihinnoiteltuihin sähköratkaisuihin
- kodin lämmityksen tarpeen hoitamiseen, perustuen sähkön tuntihintoihin, lämpötilaan ja kodin tarpeisiin
- monipuolisiin tehtäviin, joiden avulla voi esim. ohjata pientuotantoa haluamiin asioihin tai mahdollisesti myydä ylijäämän takaisin verkkoon
- erilaisten tietojen keräämisen ja niiden näyttämiseen.

3 There Corporation

Uudenlaisten energiaratkaisujen ensimmäiset ideat juontavat juurensa vuoden 2007 alkuun, jolloin tutkimus- ja kehitystyö tällä alueella alkoi Nokia Oy:n tutkimuskeskuksessa. Mahdollisuus ohjata kodin monia laitteita matkapuhelimella nähtiin Nokian kannalta merkittäväksi asiaksi. Smart Home -liiketoimintaohjelma perustettiin vuoden 2008 alussa, ja se kasvoi kolmen hengen tiimistä 50 ihmisen hankkeeksi sisältäen yli 300 ulkopuolisen yrityksen partneriohjelman. [2.]

There Corporation perustettiin toukokuussa 2009 Nokia Smart Home-liiketoimintaohjelman johdon toimesta kaupallistamaan Nokialla vuosien aikana kehitetty teknologia. Kaikki siihen liittyvän kehitystyön tulokset, materiaalit, sopimukset ja tietotaito siirrettiin Nokialta There:lle yhtiöiden välisellä lisenssisopimuksella. Samalla yritykseen liittyivät Comsel System -nimisen yrityksen perustajat tuoden mukanaan osaamisensa älykkäistä mittareista ja energiatehokkuudesta. [2.]

Therellä on tällä hetkellä 16 työntekijää ja toimistot Helsingissä, Vaasassa ja Tampereella. Theren pääasiakkaita ja yhteistyökumppaneita ovat energiayhtiöt ja palveluntarjoajat Skandinaviassa, Italiassa ja Saksassa.

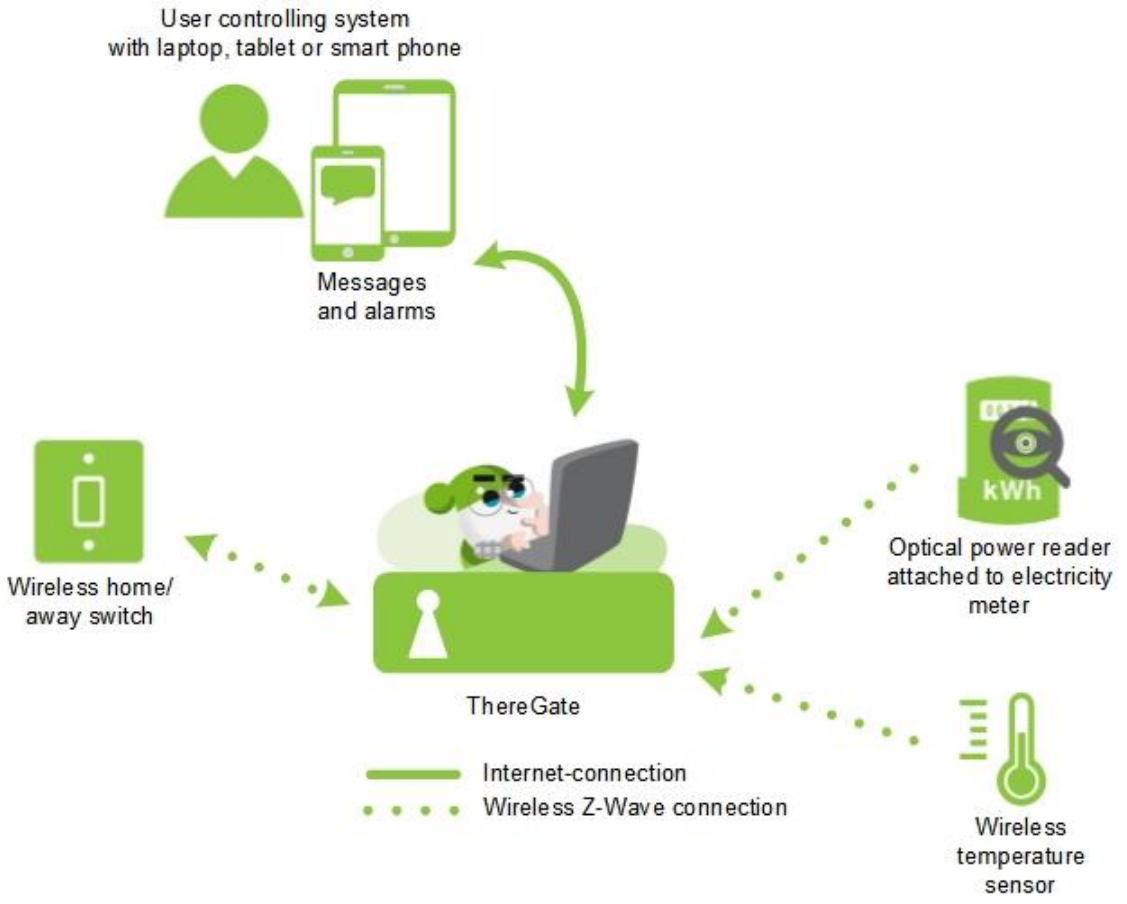
3.1 Ratkaisut

Therellä on tällä hetkellä kuusi eri energianhallintaratkaisua. Kaikkiin ratkaisuihin kuuluu ohjausyksikkö ja yksilöity internetosoite, minkä kautta pääsee internet-selaimella kirjautumaan omaan ohjausyksikköön. Kaikki ratkaisut edellyttävät internetyhteyttä ja niihin on mahdollista ostaa lisälaitteita, jotka voivat olla esim. kamera, liiketunnistin, ovi- ja ikkunasensori, kytkin, energiamittari, co2-sensori, lämpötila- ja kosteusmittari, himmennin, savutunnistin, valoisuussensori, vesivuotosensori, vesimittari, termostaatti, kaukolämpömittari, kaihtimen säädin tai ilmalämpöpumpun ohjain.

3.1.1 Mökkipaketti

Mökkipaketti on tarkoitettu vapaa-ajan asuintoihin tai vuokrattavana oleviin huviloihin. Kohteeseen asennetaan kotona/poissa-kytkin, jonka avulla voi tiputtaa asunnon lämpö-

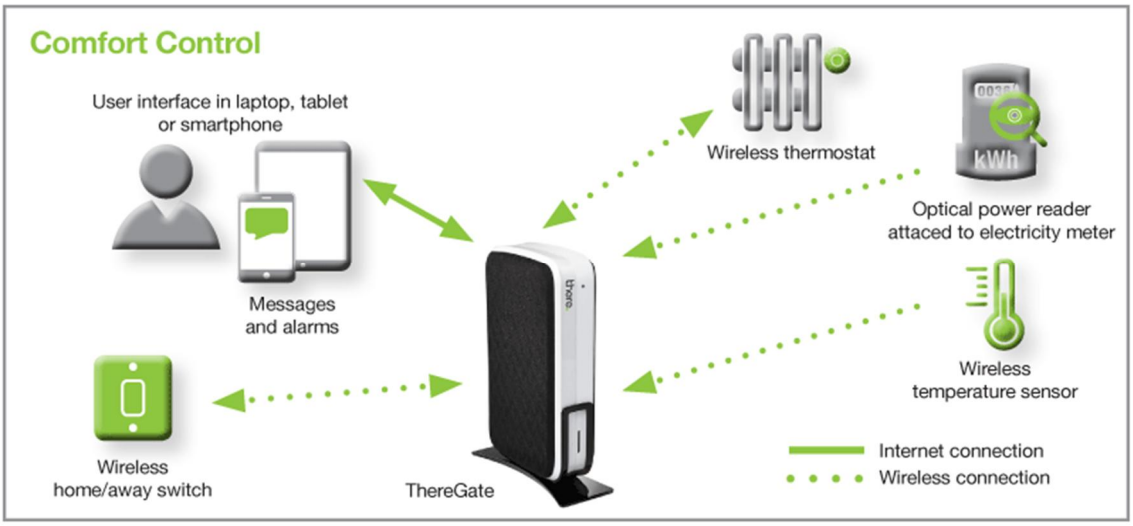
tilaa silloin, kun mökki on tyhjillään. Lisäksi on mahdollista asettaa lämpötila- tai kosteusrajoja ja saada hälytykset sähköpostitse tai tekstiviestinä.



Kuva 1. Mökkipaketti. Käyttäjä ottaa mobiililaitteella yhteyden ThereGateen, jonka avulla hän voi ohjata kotona/poissa-kytkintä ja tarkkailla asunnon lämpötilaa sekä energiankulutusta [4, s. 9].

3.1.2 Termostaattipaketti

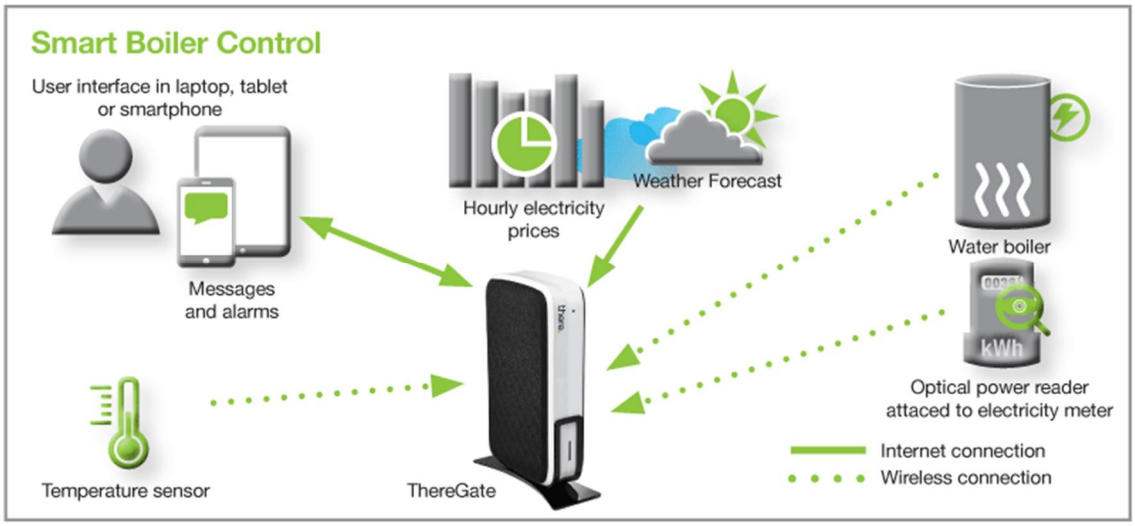
Termostaattipaketti on tarkoitettu asuntoihin, jonka lämmitys hoidetaan pattereilla. Asunnon pattereissa olevat vanhat termostaatit korvataan Danfoss- tai Stella-termostaateilla. Nämä mahdollistavat lämpötilan säädön yksittäisille pattereille, huonekohtaisesti tai vaikkapa kerralla kaikkien talon patterien ohjauksen yhdellä kertaa. Termostaattipaketin avulla voidaan helposti tiputtaa talon lämpötilaa silloin, kun on poissa, ja näin säästää lämmityskustannuksissa jopa 20 %.



Kuva 2. Termostaattipaketti. Käyttäjä ottaa mobiililaitteella yhteyden ThereGateen, jonka avulla hän voi ohjata kotona/poissa-kytkintä, termostaatteja ja tarkkailla asunnon lämpötilaa sekä energiankulutusta [3].

3.1.3 Spotti-paketti sähkölämmittäjälle

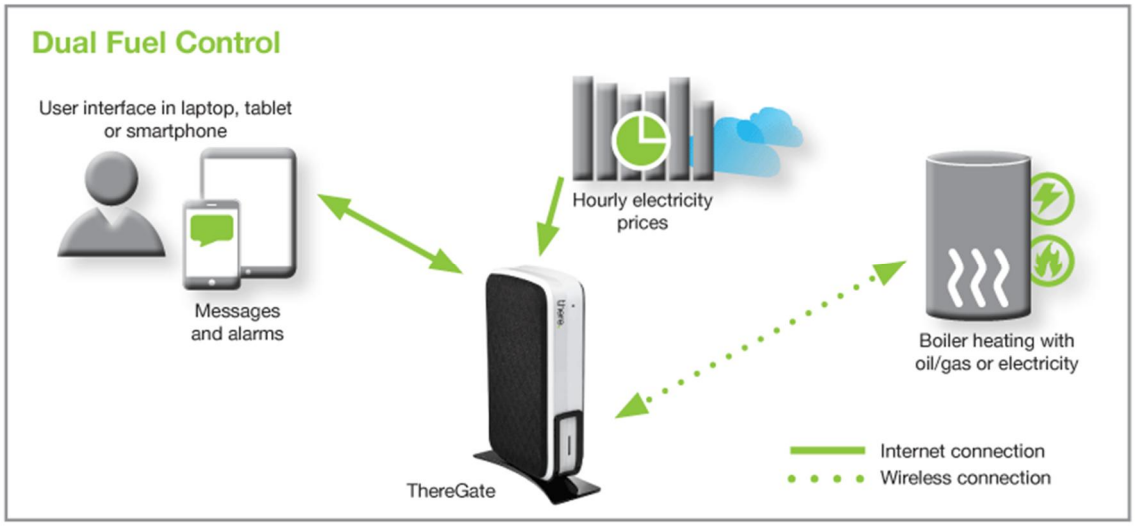
Spotti-paketti sähkönlämmittäjille on suunniteltu yhdessä Fortumin kanssa. Fortumin kutsuma Fortum Fiksu sähkönlämmittäjälle on tarkoitettu rakennuksiin, jossa on lämminvesivaraaja ja tuntihinnoiteltu sähkösopimus. Ohjausjärjestelmä hakee sähköpörssin tiedoista hinnan vuorokauden jokaiselle tunnille sekä sääennusteen ja määrittää niiden avulla kotisi lämmitystarpeen ja ajankohdan. Järjestelmä valitsee vuorokauden edullisimmat tunnit vesivaraajan lämmitykseen ja siihen on mahdollista ajastaa lisälämmityksiä, mikäli jonain hetkenä on kova vedentarve. Lisäksi on mahdollista asettaa hintarajoja, ja mikäli jonkun tunnin sähkönhinta ylittää sen saat hälytyksen sähköpostitse tai tekstiviestinä.



Kuva 3. Spotti-paketti sähkölämmittäjälle. Käyttäjä ottaa mobiililaitteella yhteyden ThereGateen, josta hän näkee lämminvesivaraajan lämmitysaikataulun, sääennusteen ja tuntikohtaisen sähkönhinnan [3].

3.1.4 Spotti-paketti öljylämmittäjälle

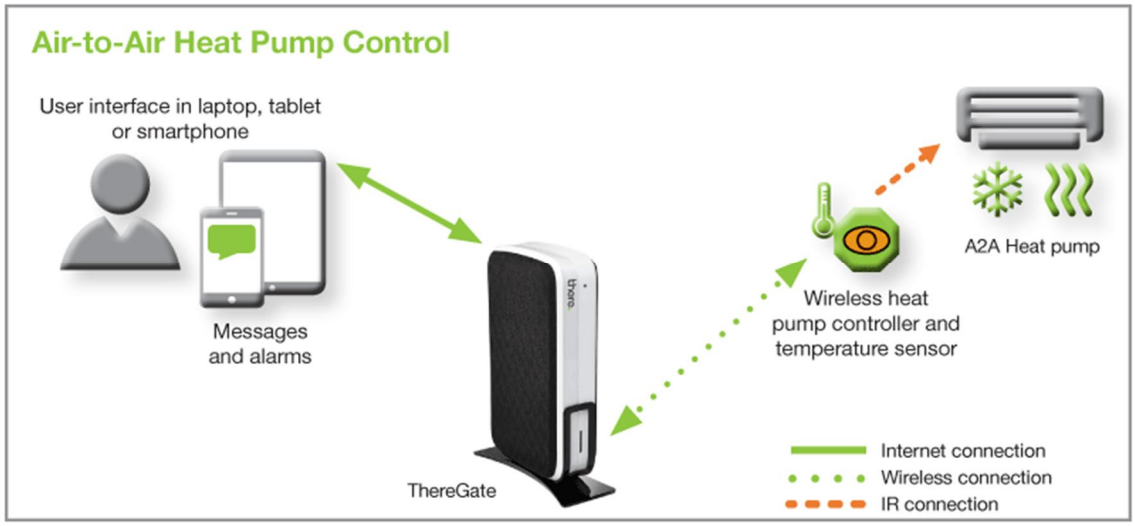
Spotti-paketti öljylämmittäjille on myös suunniteltu yhdessä Fortumin kanssa. Tämä on tarkoitettu asuntoihin, joissa on öljylämmitteinen lämminvesivaraaja. Periaate on muuten sama kuin sähkölämmitteisessä spotti-paketissa, mutta tässä ohjausjärjestelmä valitsee edullisemman lämmitysvaihtoehdon öljyn ja sähkön väliltä.



Kuva 4. Spotti-paketti öljylämmittäjälle. Käyttäjä ottaa mobiililaitteella yhteyden ThereGateen, josta hän näkee lämminvesivaraajan lämmitysaikataulun, sääennusteen, tuntikohtaisen sähkönhinnan ja lämmitetäänkö sähköllä vai öljyllä [3].

3.1.5 Ilmalämpöpumppupaketti

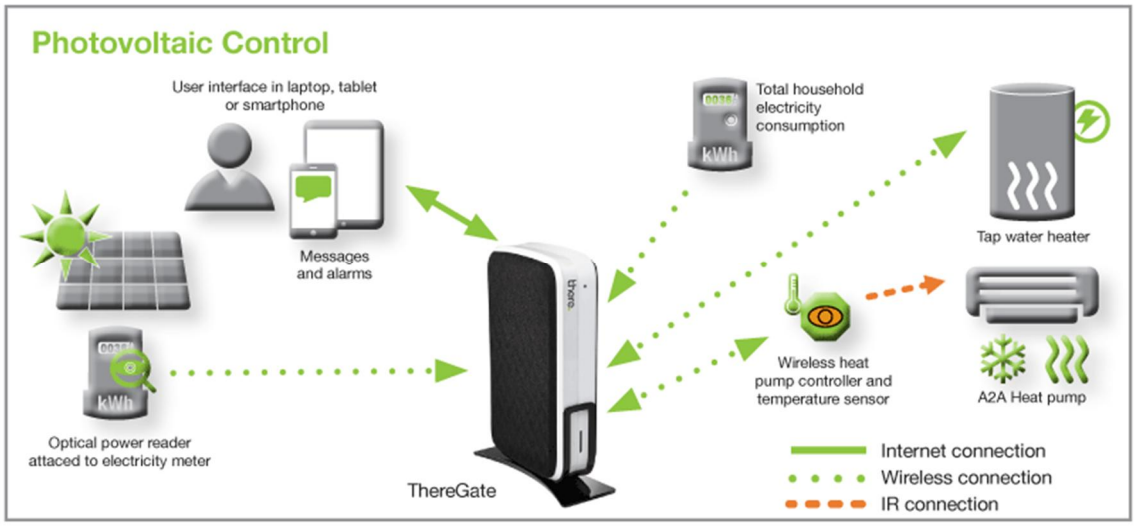
Ilmalämpöpumppupaketin avulla käyttäjät voivat ohjata kodin ilmalämpöpumppuja etänä. Ratkaisun avulla voi helposti säästää rahaa ja lisätä mukavuutta. Ratkaisun voi myös liittää osaksi muita ratkaisuja.



Kuva 5. Ilmalämpöpumppu-paketti. Käyttäjä ottaa mobiililaitteella yhteyden ThereGateen, jonka avulla hän voi ohjata kotona olevaa ilmalämpöpumppua sekä tarkkailla kohteen lämpötilaa [3].

3.1.6 Virtuoso Photo Voltaic -paketti

Virtuoso Photo Voltaic -paketti on tarkoitettu kodin energiankulutuksen hallintaan ja sen avulla voi optimoida aurinkopaneelien tuottaman energian parhaan mahdollisen käytön kotitaloudessa. Päivisin aurinkopaneelilla tuotettu energia voidaan ohjata ilmastointilaitteelle, pattereihin, lämminvesivaraajaan tai vaikkapa uima-altaan lämmittämiseen.



Kuva 6. Virtuoso Photo Voltaic -paketti. Käyttäjä ottaa mobiililaitteella yhteyden ThereGateen, josta hän näkee energiankulutuksen, aurinkopaneelin tuottaman sähkön ja paljonko joudutaan ostamaan sähköä tai myydäänkö ylijäämä takaisin verkkoon. Aurinkopaneelin tuottama sähkö voidaan ohjata asukkaan määrittämiin laitteisiin, laite voi olla mm. ilmalämpöpumppu [3].

3.2 ThereGate

ThereGate on Nokia (HCC) Home Control Centerin kehittyneempi versio. Se on Linux-pohjainen kehitysalusta, missä on avoimet rajapinnat ja ohjelmistopaketti (ThereCore™), joka mahdollistaa sovellusten tekemisen sekä erilaisten laitteiden, teknologioiden ja protokollien lisäämisen. Alustaan kehitetyt peruskomponentit ja -palvelut (ThereWare™) ovat myös kaikkien seuraavien ThereGateen tehtävien sovellusten hyödynnettävissä. [1.]

ThereGaten avulla pystymme integroimaan kohteiden monenlaiset laitteet sekä järjestelmät osiksi tiedon keräystä, kontrollia ja automaatiota. ThereGate toimii käyttäen joko kohteessa olevaa internetyhteyttä tai sisäänrakennettua 3G-modeemia. [1.]

Kuva 7. ThereGate-ohjausyksikkö.

ThereGaten tarkemmat tekniset tiedot (liite 1).

3.2.1 Teknologiat

Teknologia-ajurit ovat ThereWare-komponentteja, jotka tarjoavat L-devicen, eli rajapinnan fyysisen laitteen ja palvelun välille, sekä toteuttaa ThereCoren vaatimat ominaisuudet, kuten laitteen lisäämisen, poistamisen ja niiden konfiguroimisen.

Z-Wave ja M-Bus ovat Theren yleisemmin käytetyt teknologiat, mutta meillä on myös tuki ZigBee- ja 1-Wire-teknologioille.

Z-Wave on vähän virtaa käyttävä langaton teknologia, joka on suunniteltu erityisesti kodin energialaitteiden hallintaan. Z-Wave toimii 868 megahertsin taajuudella ja yhteen Z-Wave-verkkoon on mahdollista liittää jopa 232 eri laitetta. Z-Wave-signaali kantaa sisätiloissa noin 30 metriä ja ulkona noin 100 metriä. [7.]

Tällä hetkellä on satoja Z-Wave-laitteita valmistavia yrityksiä. Laitteet voivat olla patterikäyttöisiä tai verkkovirralla toimivia. Langattoman teknologian ansiosta ne ovat helposti asennettavissa.

M-Bus on yksinkertainen langallinen teknologia, jonka avulla pystyy hallitsemaan kotiautomaatiolaitteita. Langallisen teknologian takia M-Bus-laitteet ja niiden asennus on hieman kalliimpaa kuin esim. Z-Wave-laitteiden. Langallisen teknologian ansiosta se on kuitenkin aina varmempi kuin langaton.

3.2.2 Laitteet ja palvelut

P-device eli fyysinen kolmannen osapuolen valmistama laite. There on testannut ja tukee useita erityyppisiä ja eri valmistajien laitteita. Laite voi olla esim. kytkin-energiamittari, jolloin laitteella on kaksi L-deviceä eli toiminnallisuutta.

L-device tarjoaa kyseisen toiminnallisuuden rajapinnan fyysisen laitteen ja I-devicen eli palvelun välille. L-deviceen on mahdollista liittää erityyppisiä palveluita. Esim. BinarySwitch-tyyppiseen L-deviceen on mahdollista liittää BinarySwitch- tai FailOnSwitch-tyyppinen palvelu. EnergyMeter-tyyppiseen L-deviceen on mahdollista liittää EnergyMeter- tai ExportEnergyMeter-tyyppinen palvelu. Palveluissa on pieniä eroja, mutta perusperiaatteeltaan ne ovat samanlaisia.

Palvelu sisältävää mm. tiedon kytkimen tiloista, ja mikäli esim. kytkimen tilaa muutetaan fyysisestä laitteesta, laite ilmoittaa muutoksesta palvelulle, jonka järjestelmä signaloi, eli kertoo muutoksesta eteenpäin. Tämä mahdollistaa lähes reaaliaikaista tiedon näyttämisen käyttöliittymissä. Kaikki palvelujen muuttujien muutokset signaloidaan ja muutkin järjestelmän kannalta oleelliset asiat kuten uuden laitteen lisääminen ja poistaminen.

Palvelun on mahdollista siirtää laitteesta toiseen, mikäli esim. fyysinen laite hajoaa. Tämän avulla historiatiedot säilyvät ja laite näyttää käyttäjien kannalta samalta.

Therellä on myös virtuaalisia palveluja, jotka toimivat ilman laitetta. Virtuaalisia palveluja ovat SummaryAem, TaskManager ja RateManager.

3.2.3 Ohjelmistopaketit

Theren ohjelmisto rakentuu eri ipkg-paketeista. Bundleen valitsemme ne toiminnallisuudet, jotka toimitamme asiakkaalle. Seuraava bundle ei esim. sisällä mbus-teknologia-ajuria.

Bundlen nimi: tonttu-1.2.2

Taulukko 1. Bundleen kuuluvat paketit.

Paketin nimi	**Versio**
Base Platform	1.1.4
ThereCore	1.4.12
Z-wave TD	1.2.13
AEM i-device	1.1.11
Binary Switch i-device	1.2.5
CO2 sensor i-device	1.0.1
District Heat Meter i-device	1.1.4
Door/window sensor i-device	1.2.2
Fail-to-on Switch i-device	1.1.3
Humidity Sensor i-device	1.2.1
Motion Detector i-device	1.1.3
Temperature Sensor i-device	1.2.1
Thermostat i-device	1.2.1
Water Meter i-device	1.1.4
TaskManager VI-device	1.7.0
HTTP PB (HTTP API)	1.11.0
ThereGate (web) Pages	1.4.11
ThereHome Mobile	0.1.1
TG Mini (Widget)	1.7.0
Ddns-update-client	0.1.8
RA Relay	1.1.6

3.2.4 PB-http

PB-http API on REST-tyyppinen rajapinta, jonka avulla on mahdollista autentikoitua järjestelmään. Autentikoiduttua apin yli on mahdollista mm. ohjata palveluita, kuunnella palveluiden ja järjestelmän lähettämiä signaaleja, liittää sekä poistaa laitteita, luoda tehtäviä ja nähdä järjestelmän lähettämiä hälytyksiä.

Laitteen 5 tiedot palautetaan json-formaatissa kutsumalla ”/api/devices/5”.

```
{
    BatteryStatus: -1,
    PDeviceId: 5,
    Description: "BinarySwitch9",
    ActivePolling: false,
    SerialNumber: "Undefined",
    CreationTime: 1342252953,
    Location: "",
    tobj: "device",
    Model: "ZW ZA 3500",
    Manufacturer: "Duewi",
    Technology: "zwave",
    Availability: 0,
  - LDeviceList: [
      - {
            LDeviceType: "BinarySwitch:1",
            PDeviceId: 5,
          - p: {
                ActualSwitchState: false,
              - HistoryUpdated: [
                    "na",
                    0,
                    0
                ],
                Name: "Lamppu",
              - HistoryAttributes: [
                  - [
                        "SwitchState",
                      + { … }
                    ],
                  - [
                        "ActualSwitchState",
                      + { … }
                    ]
                ],
                SwitchState: false,
                ReasonedState: "initially_off",
                Availability: 0
            },
            LDeviceId: 7,
            FuncType: "BinarySwitch:1",
            CreationTime: 1342252966,
            IDeviceType: 0,
            IDeviceId: 28,
            x_controlled_by: [ ],
            x_monitored_by: [ ],
            x_permissions: 15
        }
    ]
}
```


Kuva 8. Laitteen 5 tiedot. Duewi ZW ZA 3500 -laitteella on yksi L-device ja siihen on liitetty BinarySwitch-tyyppinen palvelu. SwitchState on false, eli kytkin on kytketty pois päältä.

3.2.5 Tehtävät ja hälytykset

Järjestelmään on mahdollista luoda erilaisia tehtäviä. Tehtävä voi olla esim. lämmin-vesivaraajan ohjaus. Tehtävä voi olla yksinkertaisempikin, eli esim. ajastettu tehtävä,

mikä voi ohjata jotain kytkintä tiettynä aikana. Esim. vaikka arkisin klo. 7.30-16.30 kytkin on pois päältä ja muina aikoina päällä.

Tehtävä voi myös olla hälytys, eli mikäli esim. lämpömittari alittaa jonkun raja-arvon, järjestelmä lähettää sähköpostitse tai tekstiviestillä ilmoituksen. Järjestelmä lähettää myös yleisiä ilmoituksia, esim. mikäli jonkun ThereGateen liitetyn pattereilla toimivan laitteen patterit ovat vähissä.

3.3 Käyttöliittymät

3.3.1 TG-Pages

TG-Pages on PHP-pohjainen käyttöliittymä, joka on rakennettu Fitzgerald-alustan päälle. Käyttöliittymässä on tukemillemme palveluille yksilöidyt näkymät, jossa pystyy ohjaamaan palvelua sekä näkemään sen historiaa. Lisäksi käyttöliittymässä on mm. laitteiden hallintanäkymä sekä käyttäjänhallinnan ja erilaisten järjestelmän vaatimien asetusten näkymiä.

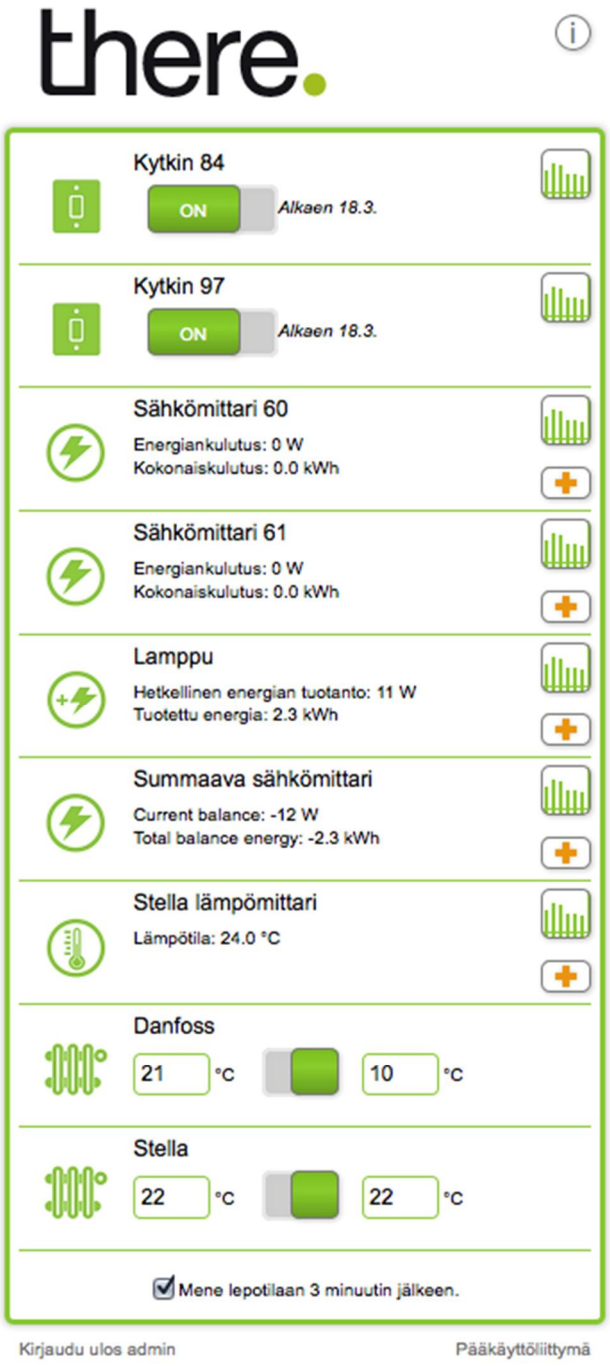
Käyttöliittymä on käännetty neljälle kielelle: suomeksi, englanniksi, ruotsiksi ja italiaksi.

TG-Pageissa on tuki teemoille, eli käyttöliittymästä oli mahdollista luoda hieman erinäköinen eri asiakkaille. CSS -tiedostot, kuvat ja tekstit on mahdollista ylikirjoittaa teemakohtaisesti.

Kuva 9. TG-Pages-käyttöliittymän järjestelmät-osio.

3.3.2 Mini (Mobiilikäyttöliittymä)

Mini on Theren ensimmäinen mobiilikäyttöliittymä. Käyttöliittymässä on tukemillemme palveluille yksilöidyt näkymät, joissa pystyy ohjaamaan palvelua sekä näkemään historiatietoja. Käyttöliittymästä näkee myös järjestelmän lähettämät viestit. Käyttöliittymä on kevytrakenteinen, jonka ansiosta se toimii hyvin vanhemmillakin puhelinmalleilla.



Kuva 10. Mini-käyttöliittymän kotinäkymä.

3.3.3 THM (ThereHome Mobile)

THM oli alun perin kehitetty Theren mökkipakettia varten, joten sen päätarkoituksena oli pystyä tiputtamaan asunnon lämpötilaa silloin, kun mökki oli tyhjillään. Lisäksi oli pystyttävä näkemään järjestelmän lähettämiä viestejä ja mm. asunnon lämpötila.

THM oli Theren ensimmäinen versio JavaScript-pohjaisesta käyttöliittymästä. Käyttöliittymän ulkoasu oli suunniteltu puhelimille ja tablet-laitteille. THM ei ollut skaalautuva, eli käyttöliittymä näytti suurin piirtein samalta, käyttää sitä puhelimella, tablet-laitteella tai tietokoneella.

THM:ssä oli tuki teemoille, eli käyttöliittymästä oli mahdollista luoda hieman erinäköinen eri asiakkaille. CSS-tiedostot, kuvat ja tekstit oli mahdollista ylikirjoittaa teemakohtaisesti. THM oli käännetty kolmelle eri kielelle: suomeksi, englanniksi ja ruotsiksi. Käyttöliittymän kieli näytettiin käyttäjälle asetetun kielen perusteella.

THM:ssä oli kirjautuminen, koti-, viestit- ja tietoa-osiot ja tuki neljälle widgetti-tyypille, jotka oli kotona/poissa, termostaatit, kytkimet ja mittarit. Koti-osio näytti käyttäjän UIDB-tietokantaan valitsemat widgetit. UIDB:n sisältöä pystyi muokkaamaan TG-Pages-käyttöliittymän kautta.

Kuva 11. THM-käyttöliittymän koti-osio, näkyvillä kytkimet- ja termostaatti-widgetit. Kytkin-widgetissä on 3 kytkintä. Ylhäällä on ryhmäkytkin ja sen alapuolella jokaiselle kytkimelle omat kytkimet. Järjestelmässä on 5 lukematonta viestiä [6, s. 15.].

4 Käyttäjäpalautetutkimus

4.1 Johdanto

Käyttäjäpalautetutkimus toteutettiin osana (DiYSE) Do-it-Yourself Smart Experiences -projektia ja käyttäjäpalautetutkimuksen tarkoituksena oli löytää parhaimmat ja heikoimmat kohdat THM-käyttöliittymästä sekä selvittää asioita, joita käyttäjät toivoisivat tehtäväksi ja mahdollisesti löytämään turhat ominaisuudet. Näin pystyisimme kehittämään käyttöliittymää monipuolisemmaksi ja käytettävämmäksi. [6, s. 4.]

4.2 Menetelmät

Testi järjestettiin niin, että kutsuin sähköpostitse ajat, joihin testikäyttäjät vastasivat. Testin oli tarkoitus kattaa There Corporationin kaikki työntekijät, mutta viisi ei pystynyt osallistumaan, joten testi pidettiin 18 henkilölle There Corporationin tiloissa Helsingissä sekä Vaasassa.

Jokaisen testitapahtuma kesti keskimäärin 50 minuuttia. Testin alussa selvitin testitapahtuman ja testin taustat. Osallistuja luki testin tehtävät ja yritti suoriutua testistä testinpitäjän ohjeiden mukaan. Pidin kirjaa testaajien käyttäytymisestä, kommenteista ja kuinka hän suoriutui tehtävistä [6, s. 4-5.].

Ensimmäisen tehtävän jälkeen kysyin testaajalta ensivaikutelman. Muiden tehtävien jälkeen kysyin testaajalta, oliko tehtävä helppo ja mikä teki tehtävästä helpon tai vaikean.

Viimeisen tehtävän jälkeen kysyin testaajan mielipidettä käyttöliittymästä, muuttuiko mielipide ensivaikutelmasta ja oliko THM-käyttöliittymä käytettävä ja hyödyllinen. Toivoiko joidenkin asioiden olevan toisin tai puuttuiko käyttöliittymästä jotakin oleellista.

Testit järjestettiin 14.12.2011-20.12.2011 välisenä aikana. Kaikki osallistujat olivat tietoisia, mitä käyttöliittymällä pystyi hallitsemaan ja olivat käyttäneet jo muita olemassa olevia käyttöliittymiä. Testi sisälsi 14 yleistä tehtävää, mitä käyttöliittymällä voi tehdä.

4.3 Laitteet, palvelut ja widgetit

Testissä käytettiin ThereGaten mallia TG800GZ v.1.0 B5. Asennetut ohjelmistopaketit ja niiden versiot on nähtävillä taulukosta 1.

Z-Wave-laitteet: 3 kpl. Aeon Labs Smart Energy Switch -kytkin- ja energiamittaria, 3 kpl Danfoss-termostaattia, Home Managealables -liiketunnistin ja Everspring ST814 -lämpö- ja kosteusmittari.

Palvelut: 3 kp kytkimiä, energiamittari, lämpömittari, kosteusmittari, liiketunnistin ja 3 kpl termostaatteja.

Kotinäkymään oli valittu kotona/poissa-, mittarit-, kytkinryhmä- ja termostaattiryhmä-widgetit.

Testit suoritettiin Apple iPad 16GB Wi-Fi first generation tabletilla ja Safari-internetselaimella.

4.4 Tehtävät

Tehtävät esitettiin siten, että ne sisälsivät seuraavanlaisen taustatarinan:

Testaaja omistaisi kesämökin, johon on asennettu Theren mökkipaketti. Testaaja on naapurissa vierailemassa ja esittelee naapurin iPad-tabletilla, mitä kaikkea hän voi ohjata ja nähdä kesämökistään. Tehtävät lomake (liite 2).

4.5 Yhteenveto

Käyttöliittymän yleiskuva oli hyvä, käyttöliittymällä pystyi tekemään kaikki tarpeelliset asiat ja se oli yleisesti helppokäyttöinen. Testaajat pitivät testistä ja kokivat, että testi oli sopivan laaja.

Käyttäjät toivoivat lisätoiminnallisuuksia kuten kameraa, ja että olisi mahdollista nähdä sähkönkulutuksia ja hintoja tietyiltä aikaväleiltä ja että voisivat vertailla eri aikavälejä. Säästönäkymää toivottiin, sekä sähkön tuntihintoja ja säätietoja. Lisäksi widgetteihin toivottiin lisää kuvaustekstejä ja että niitä pystyisi pienentämään.

Käyttöliittymä latautui liian hitaasti ja jumittui välillä. Liisäksi käyttöliittymän kieli oli välillä englanniksi, vaikka käyttöliittymän piti olla suomeksi. Käyttöliittymän takaisin-painike toimi välillä hieman oudosti. Kytkimien ja termostaattien tilapäivitykset kestivät liian kauan ja mittarit-näkymä oli liian värikäs. [6, s. 10.]

4.5.1 Kirjautuminen

Sivu latautui osan mielestä liian hitaasti, ja testaajat toivoivat, että virheilmoitus olisi selkeämpi. Osa ei huomannut virheilmoitusta ollenkaan. Virheilmoitus esitettiin punaisena reunuksena tekstikentissä.

4.5.2 Kotona / Poissa sekä mittarit

Ensivaikutelma oli vähän levottoman tuntuinen. Widgettien tarkoituksella tehdyt värikkäät taustavärit oli monen mielestä liian värikkäitä. Taustavärit on käyttäjän itse muutettavissa. Testaajat eivät erottaneet painikkeita, ja lähes jokainen yritti päästä palvelun viesteihin painamalla pienoismittaria.

Kuva 12. Koti-osion kotona/poissa- ja mittarit-widgetit [6, s. 12.].

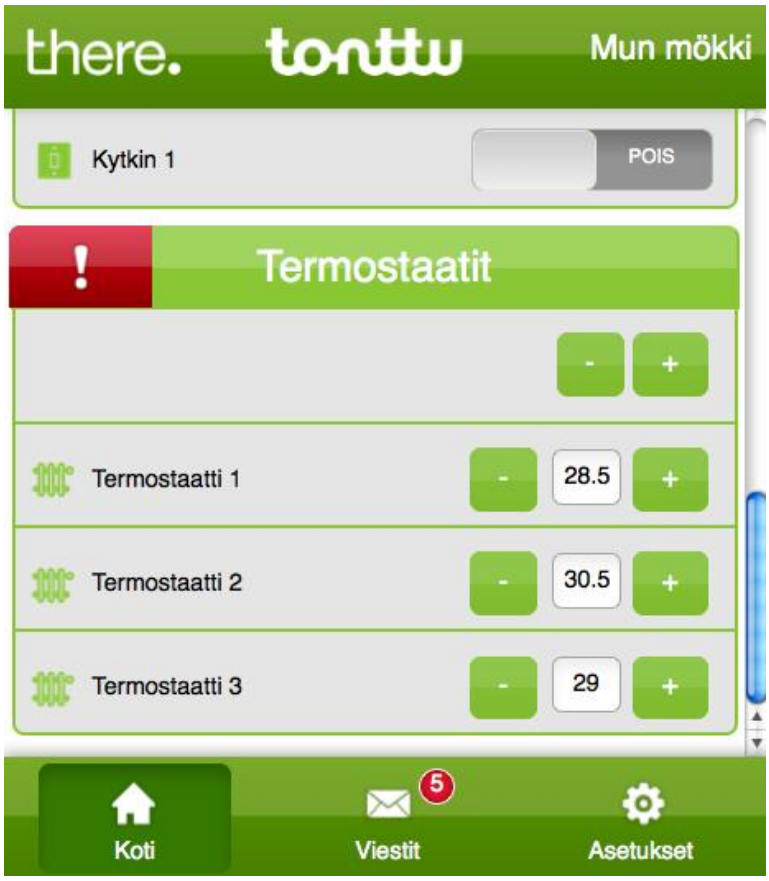
4.5.3 Kytkinryhmä

Testaajat pitivät kytkimiä helppokäyttöisenä ja riittävän selkeinä. Osa testaajista toivoi kuvausta ryhmäpainikkeille ja osa käyttäjistä ei ollut varma, onko kytkin päällä vai ei.

Kuva 13. Koti-osion kytkinryhmä [6, s. 14.].

4.5.4 Termostaattiryhmä

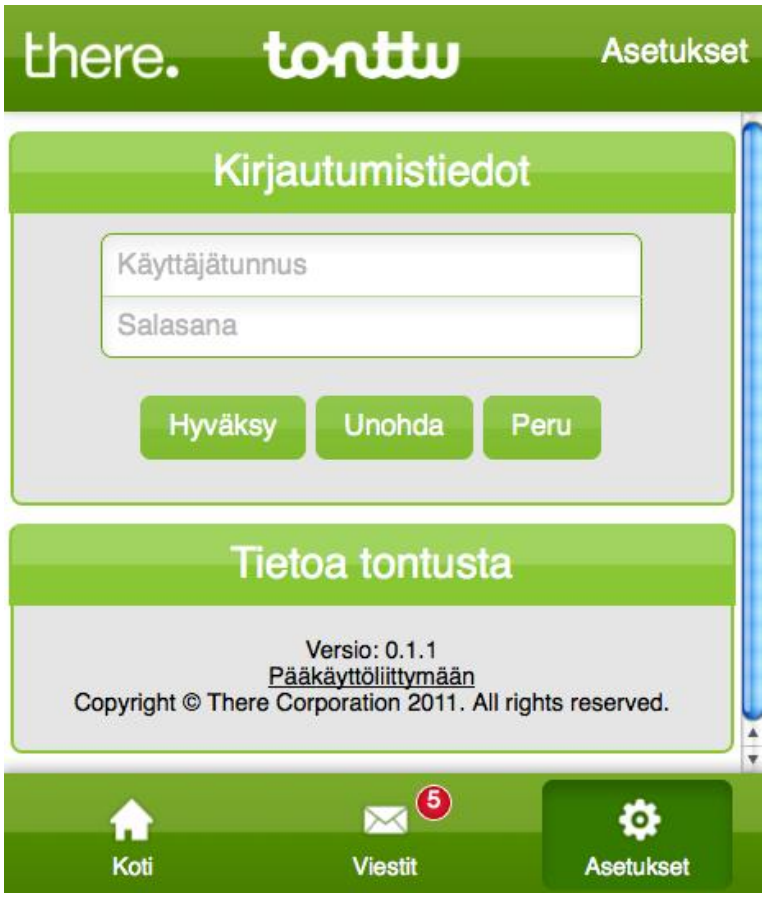
Termostaatit olivat selkeitä käyttää, vaikkakin osa ihmetteli, kuinka arvot päivittyivät, kun muuttivat arvoja nopeasti suuremmaksi tai pienemmäksi. Osa testaajista toivoi kuvausta ryhmäpainikkeille.



Kuva 14. Koti-osion termostaattiryhmä [6, s. 15.].

4.5.5 Kirjaudu ulos

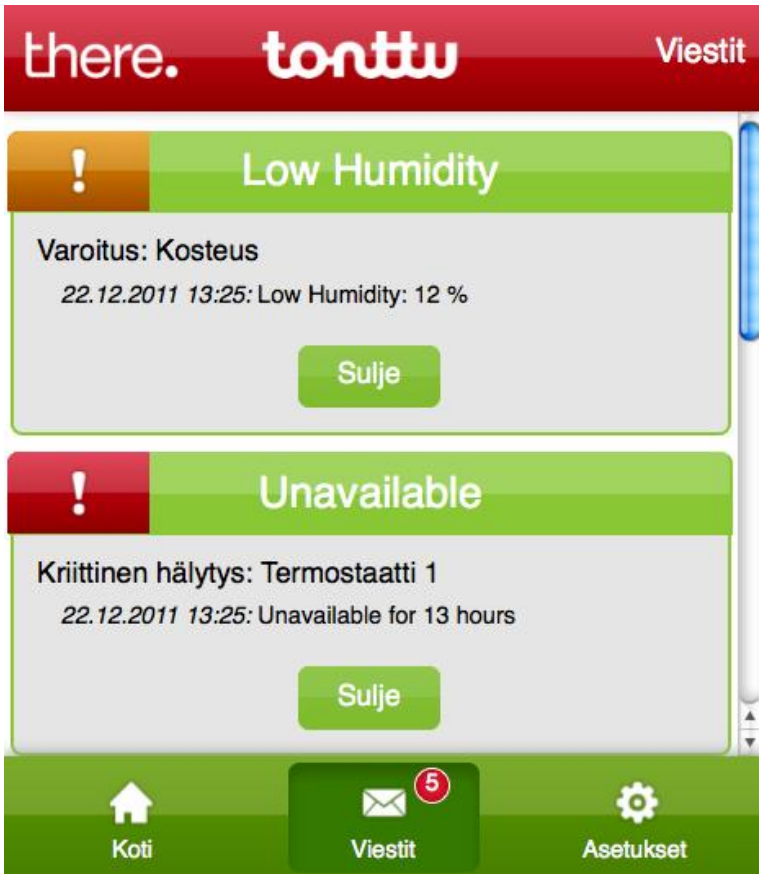
Kirjaudu ulos oli liian hankala. Testaajat eivät osanneet etsiä sitä asetukset-sivulta eikä unohda-painikketta osattu yhdistää uloskirjautumiseen.



Kuva 15. Asetukset-osio [6, s. 13].

4.5.6 Viestit

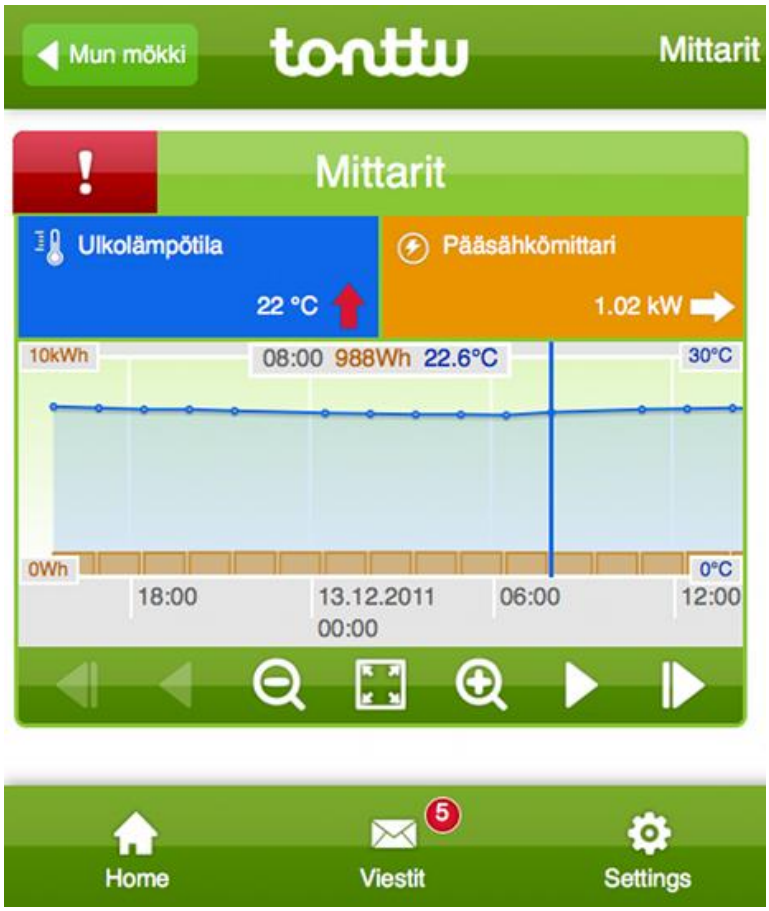
Viestit olivat selkeitä ja käyttäjät pitivät niitä riittävän yksinkertaisena. Osa tosin olisi toivonut lisätietoa viestistä.



Kuva 16. Viestit-osio [6, s. 16.].

4.5.7 Kuvaajat

Testaajat pitivät kuvaajasta ja löysivät siitä helposti kysytyt tiedot. Osa testaajista toivoi väliarvoja akseleihin. Osan mielestä skaalaus olisi voinut olla täsmällisempi.



Kuva 17. Kuvaajasta näkee ulkolämpötilan nykyisen lämpötilan, pääsähkömittarin nykyisen arvon sekä historiatiedot niistä [6, s. 17.].

4.6 Ehdotukset

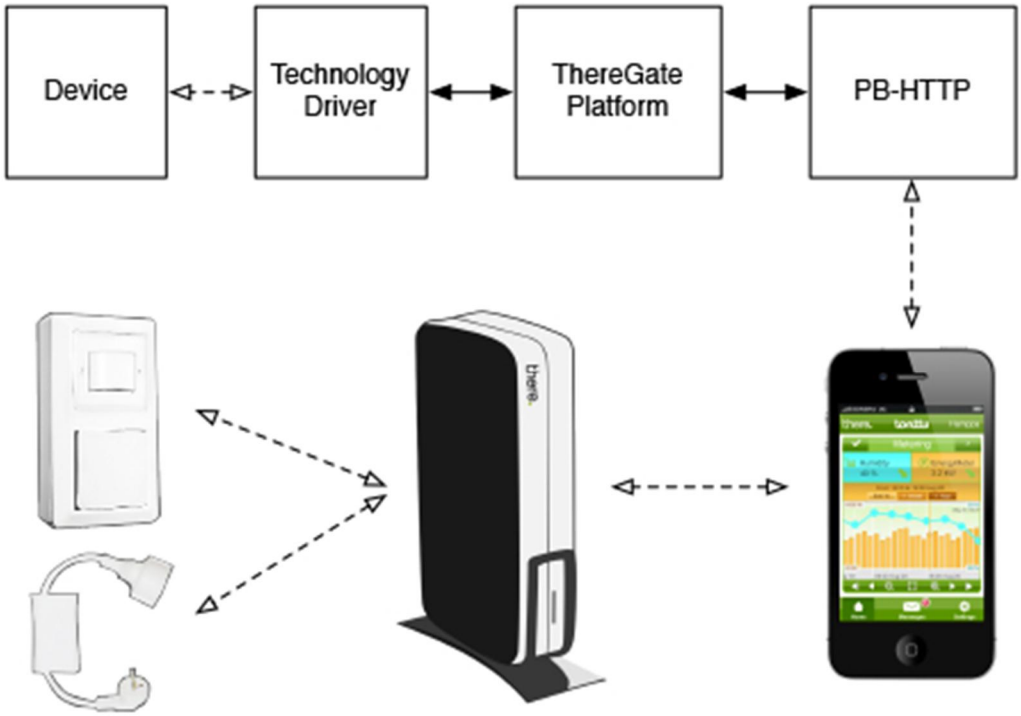
Perustuen testaajien palautteisiin ja testin tuloksiin olemme miettineet seuraavia muutoksia.

- Mietimme pienoismittarien taustavärejä ja kuinka käyttäjä pääsisi niistä helpommin viesteihin ja kuvaajaan.
- Tilamuutokset ja virheilmoitukset tulee erottua selvemmin ja olla kuvaavampia.
- Uloskirjautuminen tulisi olla helpommin löydettävissä ja ”Unohda” oli useille epäselvä.
- Termostaatin arvojen päivittymisen tulee olla sulavampaa.
- Sivun tulee latautua nopeammin. Lisäksi jumittumisista tulee päästä eroon.

5 Käyttöliittymän suunnittelu

5.1 There-arkkitehtuuri

Käyttöliittymän tulee toimia osana There-arkkitehtuuria ja käyttää PB-HTTP:n tarjoamaa rajapintaa kommunikoidessaan ThereGaten ja sitä kautta erilaisten palveluiden ja järjestelmään liitettyjen laitteiden kanssa [5, s. 5].



Kuva 18. There-arkkitehtuuri [5, s. 5.].

5.2 Teknologiat

TH:n tulee toimia THF:n kanssa kuin (SPA) single-page application. Valitsimme käytettäviksi teknologioiksi JavaScriptin, (CSS) cascading style sheetsin, (HTML) hypertext markup languagen, (Ajax) asynchronous JavaScript XML requestsin, jQueryn, HTML canvasin, Web-socketsin, Web-Storagen, PHP:n ja browser cookiesin.

5.3 ThereHome Framework

Tutkimme yleisesti käytetyt JavaScript-alustat ja selvitimme, kävisikö jokin meidän tarkoitukseemme. Emme aikanaan löytäneet sopivaa, tarpeeksi vakaata ja valmista vaihtoehtoa, joten päätimme, että kehitämme oman.

THF:n tulee perustua (MVC) malli-näkymä-käsittelijä-arkkitehtuuriin, jossa tieto on tallennettu malleihin, toiminnallisuus on käsittelijöissä ja tiedon esittäminen tapahtuu näkymissä.

THF:n tulee sisältää yleiset perustoiminnallisuudet, kuten kommunikoinnin apin yli ThereGateen, signaalien kuuntelun sekä komponentit, joita TH laajentaa tarkempiin tarkoituksiinsa.

5.4 Vaatimusmäärittely

TH:n tulee korvata yrityksemme nykyiset käyttöliittymät ja sisältää kaikki toiminnallisuus, joita THM- ja TG-Pages-käyttöliittymät sisältävät. Toiminnallisuudet teemme kokonaisuuksina, ja näin voimme julkaista uusia versioita mahdollisimman usein. Minikäyttöliittymä, joka on tarkoitettu vanhemmille puhelinmalleille, jää edelleen käyttöön. TH:n tulee myös sisältää tuotteillemme räätälöidyt toiminnallisuudet, olla helppo käytettävä, mahdollisimman nopea, monikielinen, mahdollista kustomoida näyttämään osittain erinäköiseltä eri asiakkaille ja se tulee rakentaa THF:n päälle.

TH:n tulee olla skaalautuva, eli joidenkin komponenttien koot määräytyvät sen perusteella käytetäänkö käyttöliittymää kännykällä, tabletilla vai tietokoneella. TH on suunnattu nykyaikaisille puhelimille, tableteille sekä uusimmille internet-selaimille.

Tuetut selaimet [5, s. 6-7.]:

- Safari 7
- Opera 12
- IE 10
- Chrome
- Firefox
- IE8 ja IE9 (Toissijainen tuki, pitää latautua ja toimia pääsääntöisesti, mutta esim. ulkonäössä saa olla eroja).

Mobiililaitteissa tuotetut selaimet:

- Apple IOS7: Safari
- Android 4.x: Chrome, Firefox
- WP8: IE10.

Oliot

Käyttöliittymän tulee sisältää seuraavat oliot, joita ovat widgetti, käyttäjä, teknologia-ajuri, laite, palvelu, viesti, tehtävä ja hälytys.

Sivupohjat

Käyttöliittymän tulee käyttää sivupohjia, eli ulkoasut on oltava omissa tiedostoissaan.

Signaalit

Käyttöliittymän tulee kuunnella järjestelmän lähettämiä viestejä ja päivittää käyttöliittymää muutosten perusteella. Muutokset voivat mm. olla:

- widgetteihin liittyvä muutos
- kielen vaihtuminen
- palvelujen tilojen/arvojen muuttuminen

- järjestelmään liitetty/poistettu uusi laite
- järjestelmään on luotu/poistettu uusi palvelu tai palvelu on liitetty tai irrotettu fyysisestä laitteesta
- teknologia-ajuriin liittyvä muutos
- viesteihin liittyvä muutos
- tehtäviin tai hälytyksiin liittyvä muutos.

Monikielisyys

Käyttöliittymän tulee tukea useita eri kieliä sekä käyttäjän on pystyttävä itse muuttamaan käyttöliittymän kieltä. Käyttöliittymän tulee olla käännetty suomeksi, englanniksi, ruotsiksi ja italiaksi. Käyttöliittymän tekstit tulee olla po- ja json-tiedostoissa.

Teemoitus

Käyttöliittymästä on pystyttävä luomaan osittain erinäköisiä eri asiakkaille. CSS-tiedostot, kuvat ja tekstit on pystyttävä ylikirjoittamaan teemakohtaisesti.

ThereHome ja ThereHome Framework

TH ja THF tulee olla erillisiä paketteja, että jatkossa olisi mahdollista toteuttaa muitakin käyttöliittymiä, jotka käyttävät THF:n ominaisuuksia.

5.5 Käyttöliittymät

Käyttöliittymän tulee sisältää eri osioita kuten koti, viestit, palvelut ja asetukset. Niiden toiminnot tulee olla widgeteissä.

THF:n tulee selvittää laitetyyppi, millä käyttöliittymää käytetään, ja luoda widgettien koot sen perusteella. Puhelimella käytettäessä kaikki widgetit tulee olla 29 em:n levyisiä, tabletilla ne voivat olla joko 29 em:n tai 59,5 em:n levyisiä ja tietokoneella käytettynä joko 29 em:n, 59.5 em:n tai 90.9 em:n levyisiä.

Käyttöliittymässä tulee käyttää Theren omia tai Font Awesomen ikoneja. Käyttöliittymän fontti tulee olla Arimo ja tekstin koko tulee olla väliltä 0.8em - 2em. Käyttöliittymäelementtien värit tulee määrittää ThereHome:n config-tiedostossa tai moduulien config-tiedostoissa.

Widgetit

Etusivun tulee näyttää käyttäjän UIDB-tietokantaan valitsemat widgetit. Osa widgeteistä on valittu käyttäjälle jo toimitusvaiheessa, eli käyttäjän ei tarvitse tehdä niitä itse. Esim. Spotti-paketti sähkönkäyttäjä-asiakkaille tulee olla esiasetettuina kyseinen widgetti. Käyttäjä voi vapaasti valita muitakin widgettejä, mikäli haluaa nähdä esim. lisätietoa sähkönkulutuksesta tai lämpötiloista.

Viestit

Viestit-osion tulee sisältää järjestelmän lähettämät viestit. Käyttöliittymässä on pystyttävä merkitsemään viestit luetuiksi tai mahdollisesti poistamaan viestejä. Oletuksena viestit osion tulee näyttää lukemattomat viestit. Luettuja viestejä on myös pystyttävä näkemään. Kaikki viestit on myös pystyttävä merkitsemään luetuiksi helposti.

Palvelut

Palvelut-osion tulee sisältää listattuna kaikki järjestelmässä olevat palvelut. Palvelun tulee näyttää tämän hetkiset tilat/arvot ja mahdollisuus muuttaa tiloja, mikäli palvelu tukee tilojen muutoksia. On myös pystyttävä näkemään palvelun historiatietoja.

Asetukset

Asetukset-osion tulee sisältää seuraavat asiat:

- Tietoa-näkymän, jossa tulee olla järjestelmään asennettujen pakettien versionumerot sekä linkki käyttöohjeeseen.
- Käyttäjän asetukset -näkymän, jossa on pystyttävä asettamaan mm. käyttöliittymän kielen sekä lähettääkö järjestelmä ilmoituksista tekstiviestejä ja/tai sähköposteja.
- Yleiset asetukset -näkymän, jossa hinta- ja sää-palvelimen asetukset sekä pientuotannon asetukset.

- Hälytykset-näkymän, jossa tulee olla mahdollista luoda ja muokata TaskManagerin tarjoamia hälytyksiä.
- Tehtävät-näkymän, jonka tulee toimia samaan tapaan kuin hälytykset, mutta kuitenkin niin, että tässä osiossa näytetään TaskManagerin tarjoamat tehtävät.
- Laitteiden hallinta -näkymän, jossa on mahdollista hallita laitteita ja palveluita.

6 Käyttöliittymän toteutus

6.1 Menetelmät

THF on JavaScript-pohjainen MVC-malliin perustuva alusta, jonka luokat ja luokkien perintä perustuu John Resigin yksinkertaiseen JavaScript-luokkien perintään.

MVC

THF:ssä mallit sisältävät sen tiedot, kuuntelevat siihen kohdistuvia muutoksia, sekä sisältävät funktiot, joiden avulla on mahdollista tehdä API-kyselyjä, esim. kytkimen tilan muuttamiseen [5, s. 9-10.]. THF sisältää seuraavat mallit:

- widgetti
- laite
- palvelu
- viesti
- tehtävä ja hälytys
- teknologia-ajuri.

THF:ssa komponentit kuvaavat MVC-mallin käsittelijää, joka kuuntelee mallin muutoksia ja päivittää niiden perusteella näkymiään.

Näkymät on liitetty käsittelijöihin. Käsittelijässä määritetään, mitä sivupohjaa komponentti käyttää ja initTemplateData-funktiossa muodostetaan json-objekti, joka välitetään sivupohjalle.

Lazy loading

TH perustuu Lazy loading -tekniikkaan. THF hakee käyttöliittymän alustusvaiheessa ensiksi TH:n kannalta tärkeimmät tiedot, ja TH luo niiden saatuaan käyttöliittymän ja sen näkymät. Näkymät päivittävät itseään myöhemmin, kun kaikki sen vaatimat tiedot on saatu. [5, s. 10.]

6.2 Toiminta

Sivunlatauksessa ohjaudutaan ensiksi Index.php-tiedostoon, jonka tehtävänä on luoda staattinen html-tiedosto, joka käyttää minimoituja css- ja js-tiedostoja. Lisäksi sen tulee käydä TH:n ja THF:n moduulit-kansiot läpi ja luoda niiden config-tiedostojen perusteella moduulit-taulukko.

Mikäli sivunlatauksessa index.html-tiedosto löytyy, tarkistaa index.php-tiedosto ainoastaan, onko staattinen tiedosto ajan tasalla. Jos ei ole niin luodaan tiedosto uudestaan ja käynnistetään sivu.

Index.html luo client-olion, joka kysyy api:n kautta statuksen ja saa vastauksena mm. tiedon teemasta. Client-olio luo lisäksi thereHomeClient-, modules-, translation-, api-listener-, system- ja user-oliot.

ThereHomeClient-olio luo yläosan, valikon sekä seuraavat osiot, joita ovat koti, viestit, palvelut ja asetukset.

Modules-olio käy index.php:n luoman moduulit-taulukon läpi ja luo niistä oliot. Moduuli voi olla mm. osio, osioon haluttu widgetti, teknologia-ajuri tai palvelu.

Translation-olion tehtävä on palauttaa käännöstiedoista pyydetty teksti käyttäjän kielen perusteella.

Api-listener-olion tehtävänä on kuunnella järjestelmän lähettämiä signaaleja sekä kertoa niistä system-oliolle.

User-olio sisältää käyttäjän tiedon sekä hoitaa järjestelmään kirjautumisen. Kun käyttäjä on kirjautunut, haetaan käyttäjän UIDB ja kutsutaan system-olion loadSystem-funktiota.

LoadSystem kysyy api:n kautta palvelut, laitteet ja viestit sekä luo niistä oliot. Kun tarvittavat tiedot on ladattu kerrotaan client-oliolle, että järjestelmä on valmis ja client-avaa halutun osion osoitteen perusteella, esim. koti-osion, ja koti-osio luo käyttäjän UIDB:n perusteella halutut widgetit oikeassa järjestyksessä.

Osiot osaavat itse huolehtia alasivulle ohjaamisen, mikäli osoite on esim. ”/#/settings/alarms/”, asetukset-osio avaa hälytykset näkymän.

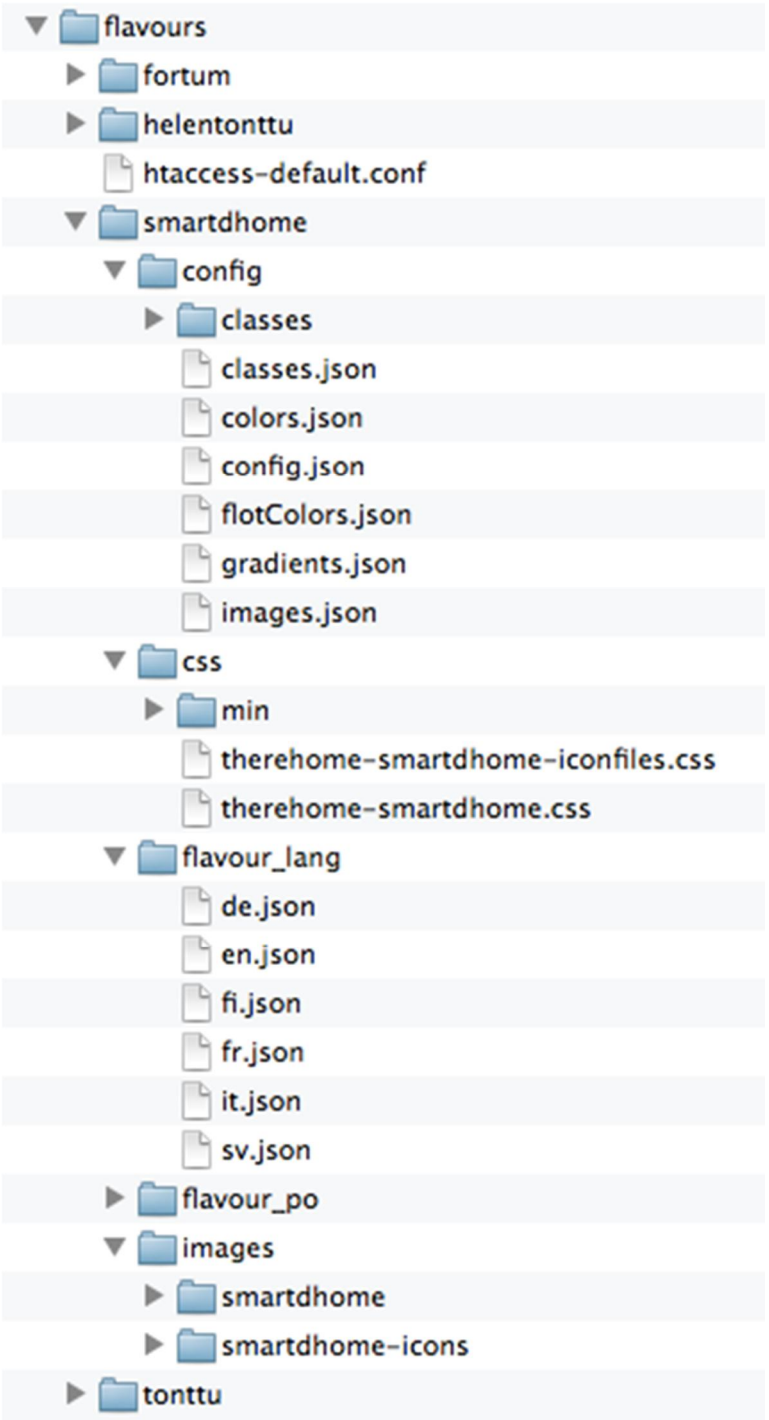
6.3 Parannukset

- Yleinen widgetti, jota hyödynnämme myös palvelut-osiossa. Yhdistimme kytkinryhmän, termostaattiryhmän, mittarit- ja ilmalämpöpumppu-widgetit yleiseen widgettiin. Yleinen widgetti tukee kaikkia palvelutyyppejä, sekä mahdollistaa useiden arvojen näyttämisen.
- Yhdistimme MVC-mallin näkymät komponentteihin, THM:ssä näkymät oli omissa tiedostoissa.
- Menuun on rakennettu ns. More menu, mikä on nähtävillä kuvan 25 oikeassa reunassa.
- Tällä hetkellä TH:ssa on 5 teemaa, sekä uusien teemojen toteuttaminen on huomattavasti yksinkertaisempaa kuin THM:ssä.
- Olemme toteuttaneet uusia toiminnallisuuksia kuten uudet widgetit, laitteiden hallinnan, palvelut-osion, tehtävät ja hälytykset. Käyttöliittymä on lisäksi skaalautuva, eli soveltuu paremmin tietokoneilla käytettäväksi.

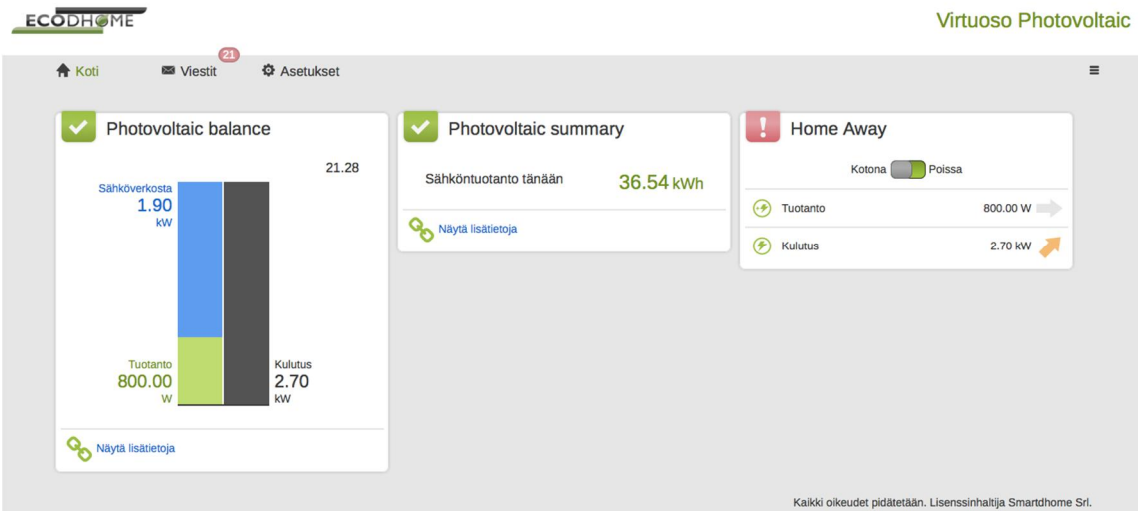
6.4 Teemat

ThereHomeen on mahdollista luoda erilaisia teemoja. Teemoissa on mahdollista ylikirjoittaa kuvat, CSS- ja kielitiedostot sekä config.json. Haluttu teema määritellään there-

gate.config-tiedostossa, joka haetaan käyttöliittymää ladattaessa. SmartDHome-teema nähtävillä kuvassa 20 ja Fortum-teema kuvassa 26.



Kuva 19. TH:n SmartDHome-teeman kansiorakenne.



Kuva 20. TH:n koti-osio SmartDHome-teemalla, johon on valittu seuraavat widgetit: energiatuotannon tase, energiatuotannon yhteenveto ja kotona/poissa.

6.5 Toiminnallisuudet

ThereHome esittää koti-osiossa UIDB-käyttöliitymätietokantaan valitut widgetit. THF luo tietokannassa olevista widgeteistä oliot, jotka kuuntelevat muutoksiaan sekä siihen liitettyjen palvelujen muutoksia ja päivittää näkymäänsä muutosten perusteella. Koti-osio näyttää widgetit käyttäjän valitsemassa järjestyksessä sekä päivittää widgettejä UIDB:n muutosten perusteella.

```
{
  - Global: {
      - Widgets: {
          - 10: {
                Name: "Mittarit",
              - Items: [
                  - {
                        Attribute: "ActiveExportPower",
                        Type: "ExportEnergyMeter",
                        Service: 480
                    }
                ],
                Id: 10,
                Disabled: false,
                Method: "follow",
                Type: "DefaultWidget",
              - Options: {
                    Stack: [ ]
                }
            }
        },
      - Tags: [
          - {
                TagOrder: [ ],
              - WidgetOrder: [
                    10
                ],
                Id: 1,
                Name: "Koti"
            }
        ]
    },
    tobj: "settings"
}
```

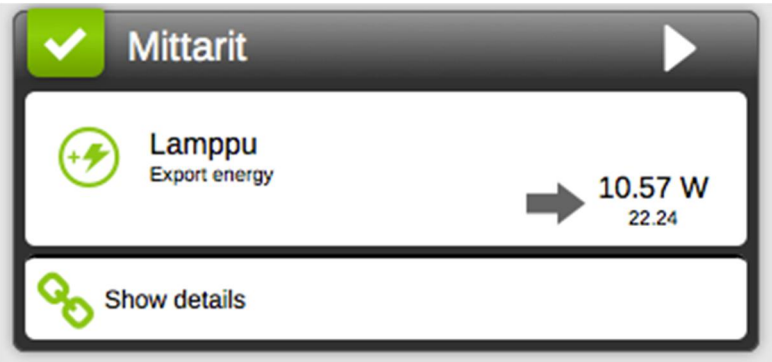

Kuva 21. Esimerkki UIDB:stä. Koti-osioon on valittu yksi yleinen widgetti, johon on valittu ExportEnergyMeter-palvelu sekä sen on haluttu näyttävän ActiveExportPowerin arvoa.

6.5.1 Kotona/poissa

Kotona/poissa-widgettiin on mahdollista valita kytkin sekä 2 haluamaansa minimittaria mihin voi valita esim. lämpö- ja energiankulutus-mittarin. Kuvassa 20 on nähtävillä kotona/poissa-widgetti.

6.5.2 Yleinen widgetti

Yleinen widgetti on mahdollista asettaa joko ohjaus- tai seuranta-tyyppiseksi ja siihen on mahdollista liittää erityyppisiä palveluita. Yleinen widgetti on korvannut vanhat kytkinryhmä-, termostaattirymä-, mittarit ja ilmalämpöpumpunohjain-widgetit. Yleisellä widgetillä on myös sisäsivu, jossa on kuvaaja, joka näyttää valittujen arvojen historiatietoja.



Kuva 22. Kuva Mittariksi nimetystä yleisestä-widgetistä. Widgettiin on valittu palvelu nimeltä Lamppu ja sen on haluttu näyttävän Export energyn arvoa.

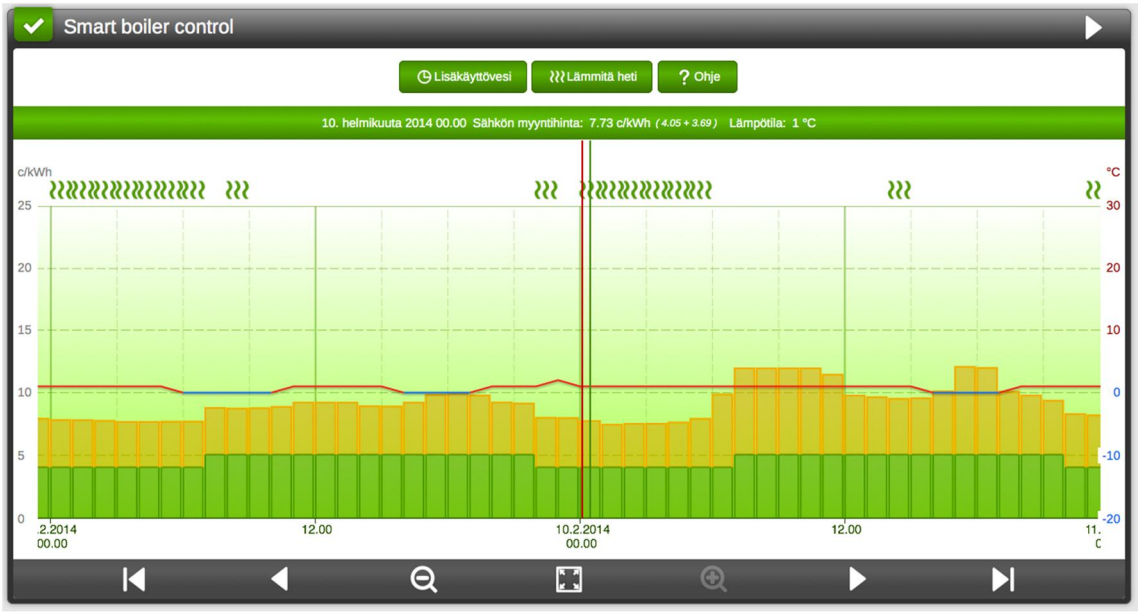
6.5.3 Energiantuotanto

Energian tuotanto koostuu energiatuotannon tase- ja energiatuotannon yhteenveto -widgeteistä. Energiatuotannon asetukset sivulla valitaan mitä summaavaa energiamittari käytetään. Summaavaan energiamittariin valitaan tuotetunenergian mittari ja kokonaiskulutuksen mittari. Kuvassa 20 nähtävillä molemmat widgetit. Tase näyttää, että tämän hetkinen tuotanto on 800 W, kokonaiskulutus 2.7 kW ja sähköä ostetaan 1.9 kW. Yhteenvedosta on nähtävillä tarkemmat tiedot tuotannosta, kulutuksesta ja taseen sisäsivulla kuvaaja, jossa on nähtävillä historiatietoja.

6.5.4 Spotti-paketti sähkölämmittäjälle

Spotti-paketti sähkölämmittäjälle näyttää sähköpörssin hinnat, ulkolämpötilan sekä hetket, jolloin lämminvesivaraajaa on lämmitetty. Mikäli järjestelmään on luotu sähkönhintahälytys ja sähkönhinta ylittää sen, niin kuvaaja värittää punaiseksi ns. kalliit tunnit.

Kuvaajan tiedot saadaan TaskManagerilta, kun järjestelmään on luotu kyseinen tehtävä. Tehtävään tulee asettaa sen vaatimia tietoja kuten lämminvesivaraajan koko. Järjestelmän päättämien lämmitystuntien lisäksi on myös mahdollista luoda lisälämmityksiä, mikäli jonain hetkenä on normaalia kovempi tarve lämpimälle vedelle. Sisäsivulta on mahdollista ajastaa ja poistaa lisälämmityksiä.



Kuva 23. Kuvaaja spotti-paketti sähkölämmittäjille. Kuvasta näkee sähkön hinnan, vesivaraajan lämmityshetket sekä sääennusteen.

6.5.5 Spotti-paketti öljylämmittäjälle

Spotti-paketti öljylämmittäjälle näyttää sähköpörssin hinnat, milloin öljyllä lämmittäminen on edullisempaa sekä milloin on lämmitetty öljyllä ja milloin sähköllä.

Kuvaajan tiedot saadaan TaskManagerilta, kun järjestelmään on luotu kyseinen tehtävä.

Kuva 24. Kuvaaja spotti-paketti öljylämmittäjille. Kuvasta näkee sähkön hinnan, milloin öljyllä lämmittäminen on edullisempaa sekä milloin on lämmitetty öljyllä ja milloin sähköllä.

6.5.6 Viestit

THF pyytää järjestelmältä lukemattomat viestit käyttäjän kielen perusteella, ja viestit-osio näyttää viestit. Viestin tila merkitään luetuksi, kun siirrytään viestin-näkymään. Viestin tilan voi myös muuttaa lukemattomaksi ja poistetuksi.

Luettuja viestejä on mahdollista hakea napista ”hae lisää viestejä”. Viestejä haetaan 10 kpl kerralla, ja mikäli järjestelmän kaikki viestit on haettu, nappi muuttuu tekstiksi ”Ei enempää viestejä saatavilla”. Viestit on mahdollista merkitä yhdellä kertaa luetuksi painamalla nappia ”Merkitse kaikki viestit luetuiksi”.

Viestit sivu kuuntelee järjestelmän lähettämiä signaaleja ja päivittää viestit-listaa niiden perusteella.

Viestit-osion yläosan taustaväri määrittyy korkeimman hälytystason perusteella. Kuvassa 23 käyttäjällä on kolme lukematonta viestiä, ja kaikki viestit ovat varoituksia, joten sivun yläosan taustaväri asetetaan keltaiseksi. Muut värit ja hälytystasot ovat sininen eli info ja punainen eli hälytys.

Kuva 25. Kuva viestit-osiosta.

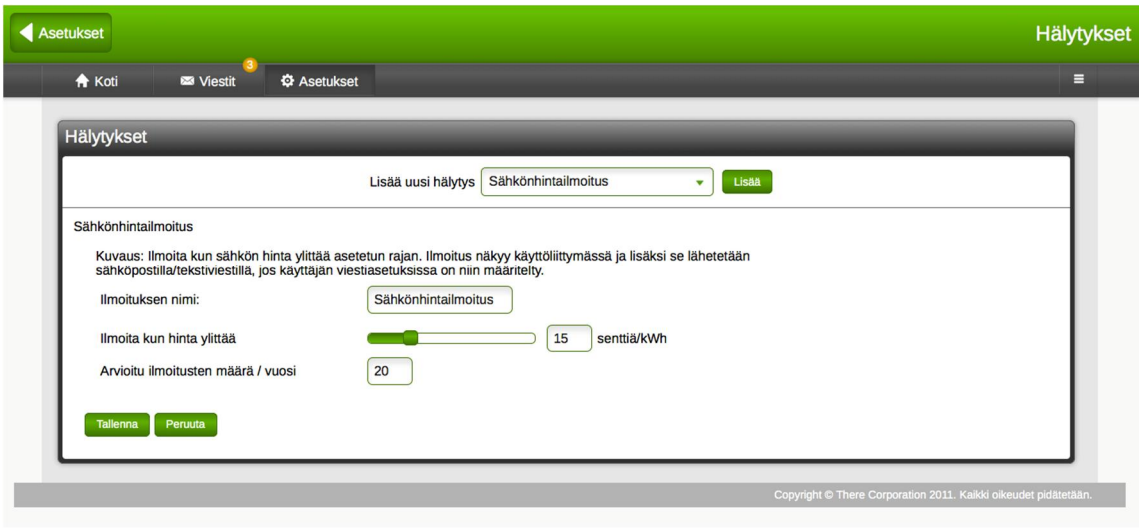
6.5.7 Palvelut

Pelvelut-osio näyttää järjestelmän kaikki palvelut. Jokainen palvelu käyttää yleisen widgetin toiminnallisuuksia hyväkseen ja luo seuranta-näkymän sekä ohjaus-näkymän mikäli palvelu tukee arvojen muuttamista. Palvelut-osio päivittää palvelut-listaa järjestelmän lähettämien signaalien perusteella, ja palvelut kuuntelevat niiden muutoksia sekä päivittävät näkymiään.

6.5.8 Asetukset

Asetukset-osio sisältää monenlaisia asetuksia, laitehallinnan, tehtävien ja hälytysten näkymät sekä järjestelmän tiedot näkymän.

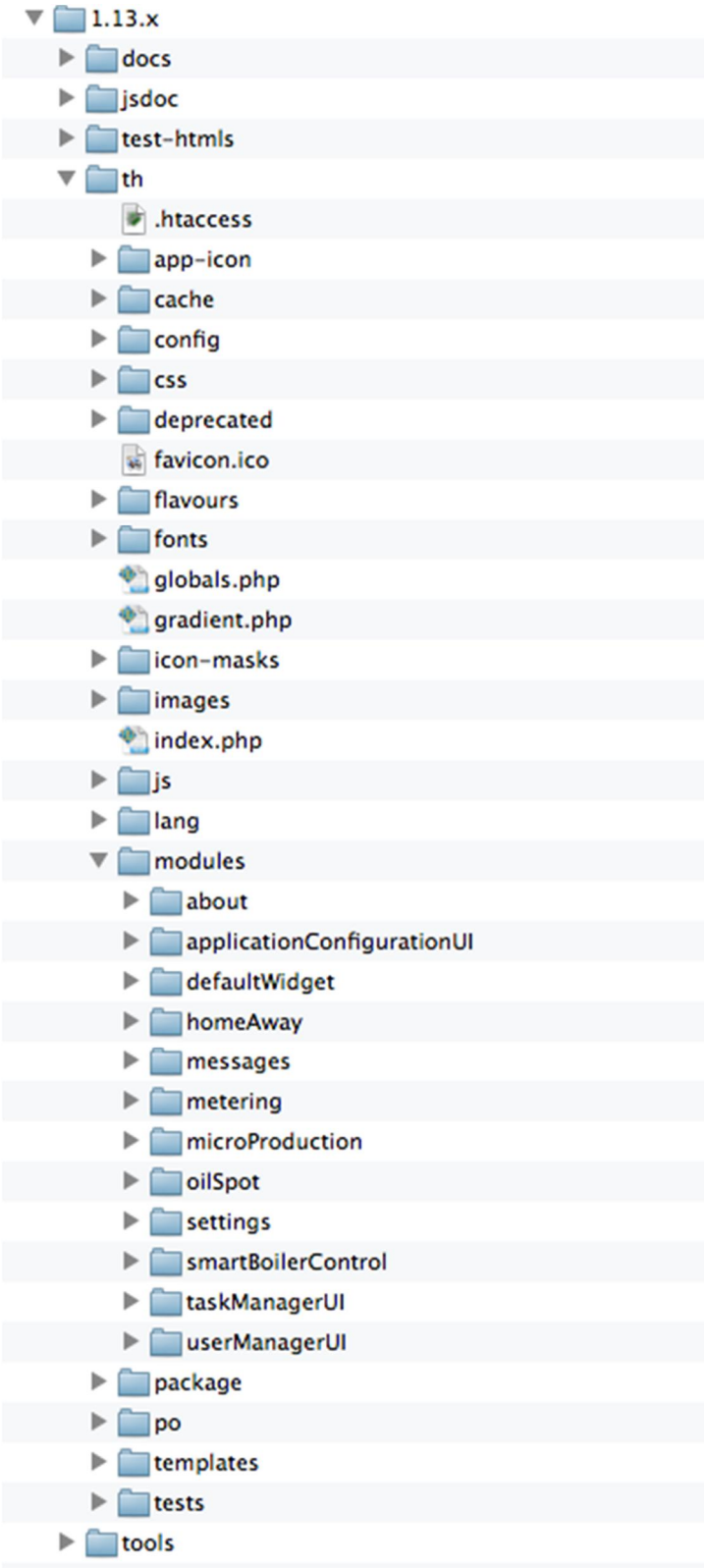
- tehtävien ja hälytyksien hallinnan
- käyttäjän asetukset
- laitteiden hallinnan
- palveluryhmien hallinnan
- laskentamittarien hallinnan
- tietoa sovelluksesta
- pientuotannon asetukset
- hintapalvelun asetukset.



Kuva 26. Kuva-asetukset-osion hälytykset widgetistä. Kuvassa ollaan luomassa sähkönhintahälytystä.

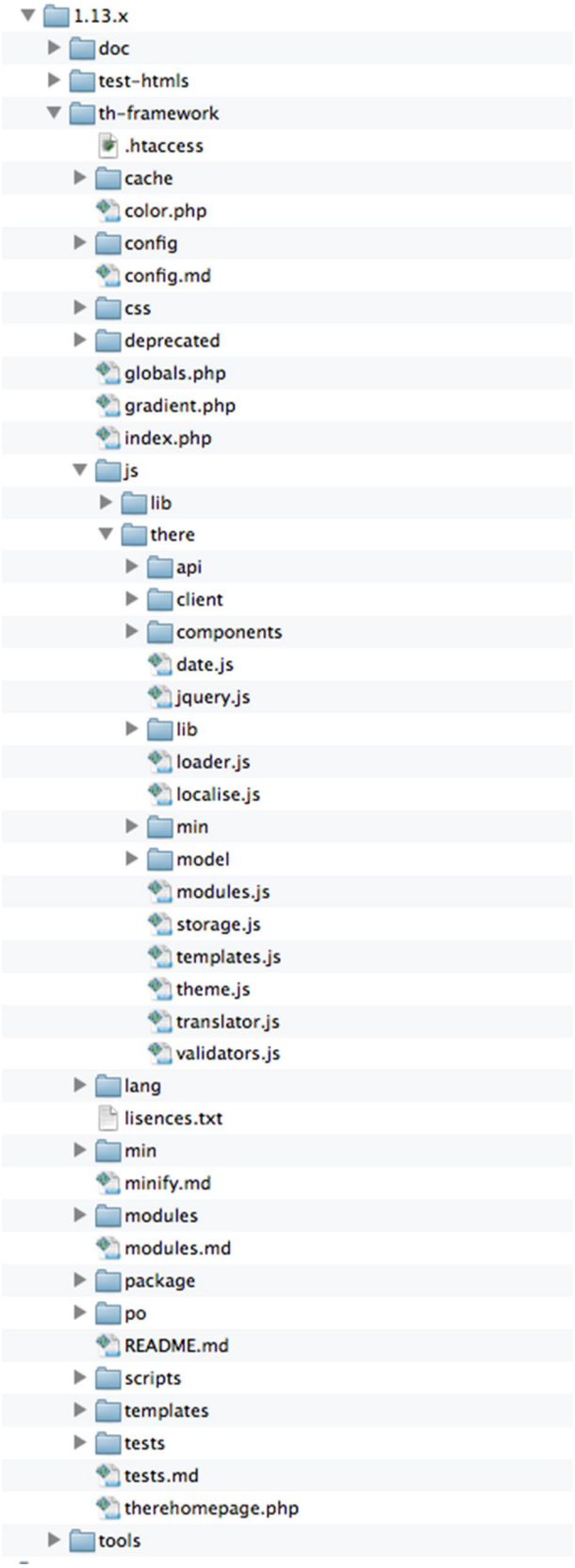
6.6 Kansiorakenne

TH:ssa ja THF:ssä tiedostot on jaoteltu omiin kansioihin. Seuraava kuva 27 esittää TH:n kansiorakenteen.



Kuva 27. TH:n 1.13-haaran kansiorakenne.

THF:n mallit sijaitsevat js-kansion sisällä olevassa model-kansiossa. Seuraava kuva 28 esittää THF:n kansiorakenteen.



Kuva 28. THF:n 1.13-haaran kansiorakenne.

6.7 Esimerkkikomponentti

Esimerkkikomponenttina esittelen _widgetPrototype-komponentin. Komponenttia laajennetaan mm. TH:n koti-osion yleisessä widgetissä. Yleinen widgetti on nähtävillä kuvassa 22.

Kuvan 29 riveillä 10-14 on nähtävillä, mitä tiedostoja komponentti vaatii ja init-funktiossa määritetään mm. this.widget, eli komponentin malli.

Kuvassa 30 määritetään sivupohjiin liittyviä tietoja sekä ajetaan super-funktio, eli ajetaan perityn komponentin init-funktio. Rivillä 98 kutsutaan update-funktio, mikäli selaimen tyyppi muuttuu. Update-funktiota ei ole laajennettu tässä komponentissa, joten tässä tapauksessa kutsutaan _prototype:n update-funktiota.

Kuvan 31 rivillä 135 on widgetAddService-funktio, jota kutsutaan kun widgettiin on lisätty palvelu. Funktiossa kutsutaan kuvan 32 updateAlarms-funktiota, eli päivitetään widgetin status-ikoni kaikkien widgetin palveluiden suurimman hälytystason perusteella.

Kuvassa 33 on InitTemplateData-funktio, joka toimii meidän MVC-mallin näkymänä yhdessä sivupohjien kanssa. Funktiossa luodaan this.tmplData, joka välitetään sivupohjalle. InitTemplateData-funktio ajetaan aina, kun kutsutaan update- ja open-funktioita.

```
/*jshint evil: true*/

// # ThereHome Framework Prototype classes
// Copyright (c) There Corporation 2010-14. All rights reserved.

// Komponentin nimi
define("framework.there.components._widgetPrototype",
    // Mitä tiedostoja vaaditaan
    [
        "framework.there.lib.utils",
        "framework.there.components._prototype",
        "framework.there.model.uidb.widget",
        "framework.there.model.system._alarmSupport"
    ], function (utils, _prototype, uidbWidget, alarmSupport) {
    "use strict";
    // _widgetPrototype laajentaa framework.there.components._prototypeä
    return _prototype.extend({
        init : function (parentComponent, container, options) {
            var hasChildContainer = typeof this.childContainerDomSelector !== "undefined";

            // Komponentin luonnissa annetut parametrit
            this.options = this.options || options || {};
            // Määritetään widgetti
            this.widget = this.widget || this.options.widget || parentComponent.widget || {};
            this.name = this.name || this.widget.type || this.options.name || "WidgetContainer";
            this.title = this.title || this.widget.name || this.options.title;
            this.type = this.type || this.widget.type || this.options.type || "";
            this.hasStatus = this.hasStatus || this.widget.hasAlarms || this.options.hasStatus || false;
            this.hasFullview = this.hasFullview || this.widget.hasFullview || this.options.hasEnlarge || false;
            this.getFullviewUrl = this.getFullviewUrl || this.options.getFullviewUrl;
            this.style = this.style || this.options.style;
            // Määritetään mitä sivupohjia käytetään
            this.template = this.template || this.widget.template || "Basic.UIContainer.container";
            this.headerTemplate = this.headerTemplate || this.widget.headerTemplate || "Basic.Widget.header";
            this.footerTemplate = this.footerTemplate || this.widget.footerTemplate || "Basic.UIContainer.footer";

            this.showHeader = (typeof this.showHeader === "boolean") ?
                (this.showHeader) :
                ((typeof this.options.showHeader === "boolean") ?
                    (this.options.showHeader) :
                    (true));

            this.containerOnly = (typeof this.containerOnly === "boolean") ?
                (this.containerOnly) : (
                    (typeof this.options.containerOnly === "boolean") ? (this.options.containerOnly) : (false)
                );

            this.useServiceList = (typeof this.useServiceList === "undefined") ? (false) : (this.useServiceList);
            this.useCustomTemplate = (typeof this.useCustomTemplate === "undefined") ?
```

Kuva 29. _widgetPrototype.js osa 1/5.

```
50                  (false) : (this.useCustomTemplate);
51
52              this.templateList = $.extend({
53                  "template" : {
54                      "template" : this.template
55                  },
56                  "header" : {
57                      "template" : this.headerTemplate
58                  },
59                  "footer" : {
60                      "template" : this.footerTemplate
61                  }
62              }, this.templateList || {});
63
64              if (this.useCustomTemplate && this.customTemplate) {
65                  this.templateList.custom = this.templateList.customTemplate || {
66                      "template" : this.customTemplate
67                  };
68              }
69
70              if (this.useServiceList) {
71                  this.serviceListTemplate = this.serviceListTemplate || "Basic.Widget.serviceList";
72                  this.templateList.serviceList = this.templateList.serviceList || {
73                      "template" : this.serviceListTemplate
74                  };
75                  this.serviceTemplate = this.serviceTemplate || "Basic.Widget.service";
76                  this.templateList.service = this.templateList.service || {
77                      "template" : this.serviceTemplate
78                  };
79              }
80
81              // Ajetaan perityn luokan init-funktio
82              this._super(parentComponent, container, options);
83
84              if (this.widget.connectComponent) {
85                  // Liittää itsensä malliin
86                  this.widget.connectComponent(this, "widget");
87              }
88
89              this.childContainerDomSelector = (hasChildContainer) ?
90                  (this.childContainerDomSelector) : (this.domSelector + " > div.container");
91
92              if (typeof this.widget.changes !== "undefined") {
93                  // Kuuntelee widgetin muutoksi, esim. mikäli nimi muuttuu ajetaan tämä: widgetUpdateName
94                  this.bindModel("widget", this.widget);
95              }
96
97              // Päivittää itsensä mikäli clientType muuttuu. Client typet ovat: mobile, tablet ja desktop
98              this.client.changes().listen(this, "clientType", this.update);
99
```

Kuva 30. _widgetPrototype.js osa 2/5.

```
            // kuuntelee milloin html on luotu
            this.changes().listen(this, "renderQueue", function () {
                var that = this,
                    $enlarge = _t.$("header > a.enlarge", this.domSelector),
                    $footer = _t.$("footer > a.title > span.icon", this.domSelector);

                _t.$("header > a, footer > a", this.domSelector).on("click", function () {
                    if (_t.$(this).attr("href")) {
                        var def = $.Deferred();
                        that.client.changes().once(that, "urlChange", def.resolve);
                        if (_t.$(this).is(".status")) {
                            _t.$(this).thereProcessing(def);
                        } else if ($enlarge.is(":visible")) {
                            $enlarge.thereProcessing(def);
                        } else if ($footer.is(":visible")) {
                            $footer.addClass("contra");
                            $footer.thereProcessing(def);
                            def.done(function () { $footer.removeClass("contra"); });
                        }
                    }
                });
            // Kuuntelee ensimmäisen kerran kun komponentti avataan
            }).once(this, "completedOpen", function () {
                if (typeof this.sids !== "undefined" && this.showHeader) {
                    for (var i in this.services) {
                        if (this.services.hasOwnProperty(i)) {
                            if (typeof this.services[i] === "object" && typeof this.services[i].changes !== "undefined") {
                                this.services[i].changes().listen(this, "newAlarm remove alarm status", this.updateAlarms);
                            }
                        }
                    }
                    this.updateAlarms();
                }
            });
        },
        // Kun widgettiin luodaan palvelu, liitetään se palvelut listaan
        widgetAddService : function (widget, service) {
            if (typeof this.services === "undefined" || typeof this.services[service.id] === "undefined") {
                this.connectService(service);
                if (this.showHeader) {
                    service.changes().listen(this, "newAlarm remove alarm status", this.updateAlarms);
                    this.updateAlarms();
                }
            }
        },
        // Kun widgetin palvelu poistetaan, pelvelu poistetaan palvelut listalta
        widgetRemoveService : function (widget, service) {
            if (typeof this.sids === "undefined") {
                this.sids = _.keys(this.services);
            }
```

Kuva 31. _widgetPrototype.js osa 3/5.

```
            var sidIndex = _.indexOf(this.sids, service.id);
            if (sidIndex !== -1) {
                this.disconnectService(service);
                if (this.showHeader) {
                    service.changes().deafen(this, "newAlarm remove alarm status", this.updateAlarms);
                    this.updateAlarms();
                }
            }
        },
        // Päivittää widgetin nimen
        widgetUpdateName : function (widget, change) {
            this.title = change;
            _t.$("header > .title h2, header > h2", this.domSelector).html(this.title.toString());
        },
        // Päivittää viestit-ikonia suurimman hälytystason perusteella
        updateAlarms : function () {
            var aStatus = this.domSelector + " a.status";

            this.alarms = this.system.getAlarmsByServices(this.sids);
            this.highestAlarm = this.system.getHighestAlarm("service", this.sids);

            if (typeof this.highestAlarm !== "undefined") {
                this.renderQueue(aStatus, _t.renderMethods.exists("removeClass"), alarmSupport.severities.join(" "));
                this.renderQueue(aStatus, _t.renderMethods.exists("addClass"),
                    alarmSupport.severities[this.highestAlarm.severity] || alarmSupport.severities[0]);
                _t.$(aStatus).attr("href", "#/messages/widget/" + this.widget.id);
            } else {
                this.renderQueue(aStatus, _t.renderMethods.exists("removeClass"), alarmSupport.severities.join(" "));
                this.renderQueue(aStatus, _t.renderMethods.exists("addClass"), "none");
                this.renderQueue(aStatus, _t.renderMethods.exists("removeAttr"), "href");
            }
        },
```

Kuva 32. _widgetPrototype.js osa 4/5.

```
182         // Muodostetaan json-objekti joka välitetään sivupohjalle
183         initTemplateData : function () {
184             var statusLink = "#";
185 
186             // Viestit sivu näyttää myös widgettikohtaisesti viestit
187             if (this.widget.id !== undefined) {
188                 statusLink = "#/messages/widget/" + this.widget.id;
189             }
190 
191             var customCss;
192 
193             if (this.domReady()) {
194                 customCss = this.$domSelector.attr("style");
195             }
196 
197             this.tmplData.statusLink = undefined;
198             this.tmplData = $.extend({
199                 "title" : this.title,
200                 "notWidget" : this.containerOnly,
201                 "headerClass" : this.headerStyle,
202                 "showHeader" : this.showHeader,
203                 "showFullview" : this.hasFullview,
204                 "fullviewText" : this.fullviewText || s("Show details", "thf"),
205                 "fullviewIcon" : this.fullviewIcon || "fa link",
206                 "fullviewLink" : (typeof this.getFullviewUrl === "function") ?
207                     (this.getFullviewUrl()) : ("/widget/" + this.widget.id + "/fullview/"),
208                 "type" : this.type,
209                 "iconClass" : "widget",
210                 "iconType" : this.widget.type,
211                 "statusText" : Gettext.strargs(s("Alarm level %1").toString(), (typeof this.highestAlarm !== "undefined") ?
212                     (alarmSupport.severityStrings[this.highestAlarm.severity]) :
213                     (alarmSupport.severityStrings[0])
214                 ),
215                 "statusClass" : (typeof this.highestAlarm !== "undefined") ?
216                     (alarmSupport.severities[this.highestAlarm.severity]) : (alarmSupport.severities[0]),
217                 "useServiceList" : this.useServiceList,
218                 "statusLink" : statusLink,
219                 "width" : this.width,
220                 "customCss" : customCss
221             }, this.tmplData);
222             this._super();
223         }
224     });
225 });
```

Kuva 33. _widgetPrototype.js osa 5/5.

6.8 Valmiit komponentit

THF ja TH hyödyntää useita valmiita komponentteja. Kuvaajat hyödyntävät esim. flot-komponenttia ja MVC-mallin näkymät jsrender-sivupohjamoottoria. Seuraavana luetelma sisältää kaikki käyttöliittymässä käytetyt valmiit komponentit.

- Simple JavaScript Inheritance (http://ejohn.org/blog/simple-javascript-inheritance)
- Localization library (http://jsgettext.berlios.de/)
- Template rendering library (http://github.com/BorisMoore/jsrender)
- Dynamic script loading library (http://headjs.com)
- A javascript date library for parsing, validating, manipulating, and formatting dates. (http://momentjs.com)
- Library for checking browser support for advanced JavaScript and CSS features (http://www.modernizr.com)
- Advanced scroll support for mobile devices (http://cubiq.org/iscroll)

- json2 (https://github.com/douglascrockford/JSON-js)
- jQuery plugin for Web-Storage (http://www.jstorage.info)
- Plotting library for jQuery (http://code.google.com/p/flot/)
- In-Field Labels jQuery Plugin (http://fuelyourcoding.com/scripts/infield/)
- Adding a Timepicker to jQuery UI Datepicker (http://trentrichardson.com/examples/timepicker)
- HTML5 Placeholder jQuery Plugin (http://mths.be/placeholder)
- Freetile (http://www.yconst.com/web/freetile)
- jQuery UI (https://jqueryui.com)
- Extends jquery-ui slider widget to include multiple ranges within a single slider (https://github.com/ladrower/jquery-ui-intervals)
- jQuery.color (http://jquery.com)
- jQuery.transit (https://github.com/rstacruz/jquery.transit)

6.9 Validointi, minimointi ja käännöstiedostot

Aina koodimuutosten jälkeen takistamme, että tiedostot ovat valideja käyttäen jshint-, csslint- ja jsonlint-komponentteja. Minimoimme lisäksi JavaScript- ja CSS-tiedostot käyttäen uglifyjs- ja cleancss-komponentteja.

Jokaisen tekstimuutoksen jälkeen luemme PO-käännöstiedostoista tekstit JSON-tiedostoihin käyttäen po2json-komponenttia.

7 Yhteenveto

Tällä hetkellä ThereHome on versiossa 1.14.0 ja suurimmalle osalle asiakkaista on asennettu 1.12-haaran versiot.

1.14.0-versiossa keskeneräisiä ominaisuuksia ovat laitteiden hallinta, palvelujen asetukset, palvelut-osio, teknologia-ajurien näkymät ja widgettien hallinta. Näiden valmistuttua kaikki TG-Pages-käyttöliittymän toiminnallisuudet on toteutettu ThereHome-käyttöliittymässä.

Olemme lisäksi miettineet seuraavia jatkokehityksiä:

- Tutkimme, voisiko Facebookin React-kirjasto olla sopiva korvaamaan nykyiset näkymät ja niiden päivittämisen.
- CSS-tiedostot tulisi kirjoittaa LESS:llä tyylitiedostojen selkiyttämiseksi.
- Yhdistämme TH:n ja THF:n samaksi paketiksi. Emme näe enää tarvetta erillisille paketeille.

Lähteet

1 Alusta. Verkkodokumentti. There Corporation. < http://smarthomepartnering.com/fi/alusta> Luettu 27.3.2014.

2 Historia. Verkkodokumentti. There Corporation. < http://smarthomepartnering.com/fi>. Luettu 27.3.2014.

3 Solutions. Verkkodokumentti. There Corporation. <http://www.therecorporation.com/solutions>. Luettu 20.3.2014.

4 ThereGate Reference Guide. Sisäinendokumentti. There Corporation. < There-Gate_Reference_Manual.pdf>. Luettu 27.3.2014.

5 ThereHome Implementation Specification. Sisäinendokumentti. There Corporation. < ThereHome-Implementation-Specification.pdf>. Luettu 27.3.2014.

6 ThereHome Mobile Usability Test. Sisäinendokumentti. There Corporation. < ThereHome-Mobile-Usability-Test.pdf>. Luettu 27.3.2014.

7 Z-Wave. Verkkodokumentti. Wikipedia. <http://en.wikipedia.org/wiki/Z-Wave>. Luettu 27.4.2014.

ThereGate800Z ja ThereGate800GZ tekniset tiedot

Ethernet Ports	4 x Gbps LAN Ports. 1 x Gbps WAN Port.
VLAN	802.1 Q.
Wireless network	802.11 b/g/n, 300 Mbps transfer rate. 2 internal antennas.
Sensitivity	-70dBm (11g @54Mbps).
Security	TPM (Trusted Platform Module), v. 1.2 specified by Trusted Computing Group.
HAN Radio	Integrated Z-Wave controller. Additional radio interfaces may be connected to USB expansion ports.
USB Ports	4 x USB 2.0 full speed host. 1 of the USB ports is powered running ThereGate on battery power.
Memory	2 GB internal storage standard, may be ordered with up to 16 Gb. End user accessible SD card reader for extension. 256 MB DDR2 RAM.
CPU	533 Mhz Broadcom 4718 SOC with integrated MIPS74k CPU.
Power Adapter	Wall Mount. AC input: 90V .. 264 V, 47 ~ 63 Hz. DC output: 5 V, 4 A max current.
Battery	8 x AA batteries, no recharging in ThereGate. Battery backup possibilyty for ThereGate operation during power failures.
RTC (Real Time Clock)	72 hours minimum run time without mains power or batteries.
Cooling	Passive cooling (no fan).
Indicators	14 LED:s.
Size	250mm x 139mm x 50mm (without canvas panels).
Weight	~500 g.
Operating Environment	0 ~ +40 oC. 10 ~ 90 % Relative Humidity, Non Condensing.
Storage Environment	-20 ~ +60 oC. 10 ~ 90 % Relative Humidity, Non Condensing.

Network security	Kernel firewall, Network Address Translation (NAT), Port forwarding, MAC address filtering
Operating system	Linux, Kernel 2.6
Antenna	Internal multi band antenna.
UMTS Module (G-model only)	BandRich M250V. SIM holder under battery cage.
WCDMA (G-model only)	HSUPA 2Mbps uplink modem operation (SW upgradable to 5.7Mbps). HSDPA 7.2Mbps downlink modem operation
E-GPRS (G-model only)	850/900/1800/1900 MHz.

Käyttäjäpalautetutkimuksen tehtävät

1. Kirjautuminen

Testaajalle oli annettu internetosoite, käyttäjätunnus, salasana ja käskettiin kirjautumaan.

2. Ensivaikutelma

Testaajaa pyydettiin tutustumaan hetken käyttöliittymään ja selvittämään mitä käyttöliittymällä on mahdollista tehdä.

3. Mitä tietoa liiketunnistimesta on saatavilla

Testaajaa pyydettiin kertomaan mitä tietoa liiketunnistimesta on saatavilla.

4. Missä tilassa kytkinryhmän kytkimet ovat

Testaajalta kysyttiin missä tilassa kytkinryhmän kytkimet ovat.

5. Mihin arvoon termostaatit on asetettu

Testaajalta kysyttiin mihin arvoon termostaattiryhmän termostaatit on asetettu.

6. Lämpötila ja sähkönkulutus

Testaajalta kysyttiin mikä on ulkolämpötila ja energiamittarin sähkönkulutus tällä hetkellä ja mitkä ne oli 13. Tammikuuta 2011 klo. 8.00.

7. Aseta kaikki termostaatit arvoon 30 °C

Testaajaa pyydettiin asettamaan termostaattiryhmän kaikki termostaatit arvoon 30 °C.

8. Aseta termostaatti 3 arvoon 27 °C

Testaajaa pyydettiin asettamaan termostaatti 3 arvoon 27 °C.

9. Kytke kaikki kytkimet päälle

Testaajaa pyydettiin asettamaan kytkinryhmän kaikki kytkimet päälle.

10. Kytke kytkin 1 pois päältä

Testaajaa pyydettiin asettamaan kytkimen 1 pois päältä.

11. Mikä hälytys liiketunnistimelle on tullut?

Testaajalle kerrottiin että liiketunnistimelle on tullut hälytys ja tehtävänä oli selvittää mistä hälytys oli tullut.

12. Kuinka monta viestiä järjestelmässä on

Testaajalta kysyttiin monta hälytystä järjestelmässä on, ovatko ne helposti ymmärrettävissä ja onko viestien merkitseminen luetuksi helppoa.

13. ThereHome mobiilikäyttöliittymän versionumero

Testaajalta kysyttiin mitä versiota mobiilikäyttöliittymästä hän käyttää.

14. Kirjaudu ulos

Testaajaa pyydettiin kirjautumaan ulos järjestelmästä.