

Automation i hemmet

En alternativ lösning på klimatproblemen, med
hjälp av Raspberry Pi & Android

Home Automation

An alternative solution to environmental problems
using Raspberry Pi and Android

ANDREAS HALLBERG

OSKAR LUNDH



**KTH Information and
Communication Technology**

Examensarbete inom Elektronik- och Datorsystem

Degree project in Electronic- and Computer Systems

Stockholm, Sweden 2013

KTH ICT

Kurs IL120X, 15hp

TRITA-ICT-EX-2013:131

Sammanfattning

Detta arbete täcker utvecklingen av ett hemautomationssystem som tillåter hushåll att sänka sin elkonsumtion och miljöpåverkan. Samtidigt som det ger den boende ett sätt att spara pengar samt ett bekvämt sätt att fjärrstyra sitt hem.

Arbetet utvärderar också vilka delar som är viktiga för att i framtiden hitta en lösning som täcker alla problem med det utvecklade systemet.

Systemet består av en nätverksenhet som vidarebefordrar signaler/meddelanden till en s.k. fjärrströmbrytare för att kunna kontrollera t.ex. lampor och värme i ett hus. Den kan också ta emot data från sensorer och skicka denna tillbaka till användaren. Användargränssnitt består av en mobilapplikation. Systemet tillåter att direkt ersätta en fjärrkontroll, men också att schemalägga händelser och utnyttja sensorer eller mobiltelefonens inbyggda funktioner i s.k. regelverk där användaren ställer upp logiska satser för hur t.ex. en fjärrströmbrytares tillstånd ska bero på en sensors värde eller en mobiltelefons GPS-koordinater.

Abstract

This paper covers the development of a home automation system which allows homes to lower their energy consumption as well as their climate influence, and at the same time allowing the residents to save money and more easily remote controlling their home.

This paper also evaluates which parts of such a system need further development to be a complete solution.

The system consists of a network gateway that relays any information from the user to remote switches using radio, allowing control of light and heat in a home. It is also capable of receiving information from sensors and relaying this back to the user. The user interface is a mobile application. The system allows it to be used as a replacement for a remote controller but also allows for scheduling and using sensors or the functionalities of modern smartphones. This by having a rule framework where the user can define logic statements where for example a remote switch' state is dependent on a temperature sensor, a scheduling event or the phones GPS-coordinates.

Innehåll

| | |
|---|----|
| Sammanfattning | 2 |
| Abstract | 3 |
| Innehåll | 4 |
| 1. Introduktion | 6 |
| 1.1 Bakgrund | 6 |
| 1.2 Problembeskrivning | 6 |
| 1.3 Problemformulering | 7 |
| 1.4 Syfte | 7 |
| 1.5 Mål med arbetet..... | 7 |
| 1.6 Avgränsningar | 7 |
| 1.7 Disposition | 7 |
| 2. Metod | 9 |
| 3. Teori | 10 |
| 3.0 Definition av termer | 10 |
| 3.1 Systemmodell | 11 |
| 3.2 Android-applikation | 12 |
| 3.3 Fjärrströmbrytare..... | 13 |
| 3.4 Raspberry Pi | 15 |
| 3.5 Sensorer | 15 |
| 3.6 Nätverkssäkerhet | 16 |
| 3.7 Miljö | 17 |
| 3.7.1 Modellering av energibesparingar..... | 17 |
| 3.8 Existerande teknik/Liknande lösningar | 19 |
| 3.8.1 KNX | 19 |

| | |
|-----------------------------------|----|
| 3.8.2 Insteon | 20 |
| 3.8.3 TellStick | 20 |
| 4. Resultat..... | 22 |
| 4.1 GreenBox | 22 |
| 4.1.1 Server | 24 |
| 4.1.2 Trådlös kommunikation | 24 |
| 4.1.3 Schemaläggare | 27 |
| 4.1.4 LCD-display | 27 |
| 4.2 GreenHouse | 29 |
| 4.2.1 GUI..... | 29 |
| 4.2.2 Datalagring | 30 |
| 4.2.3 Fjärrkontroll | 32 |
| 4.2.4 Översikt av sensorer | 33 |
| 4.2.5 Regler (koncept)..... | 33 |
| 4.2.6 Kalender (koncept) | 35 |
| 4.3 Sensor | 36 |
| 4.3.1 Temperatur | 36 |
| 5. Slutsatser | 37 |
| 6. Framtida förbättringar | 39 |
| 7. Referenser..... | 40 |

1. Introduktion

Här följer först bakgrunden till de problem som ska behandlas i denna rapport. Sedan följer en mer direkt definiering av frågorna som ställs och som i slutänden ska besvaras. Efter det förklaras syftet med det här projektet och målen som hoppas uppnås med det praktiska arbete som har utförts parallellt med skrivandet av rapporten. Sedan förklaras vilka avgränsningar som gjorts i arbetet. Sist kommer en disposition där innehållet av de olika kapitlen sammanfattas.

1.1 Bakgrund

Ett väl diskuterat ämne idag är miljöfrågan, det blir mer och mer aktuellt för privatpersoner att engagera sig och själva hjälpa till. Därför ser man exempelvis elbilar och mat med lågt koldioxidavtryck erbjudas till konsumenterna.

Utöver transport och livsmedel så har också elförbrukningen hos hushåll en viss miljöpåverkan. Idag kommer ungefär 37 procent av Sveriges elkonsumtion från bostads- och servicesektorn (1) och runt 10 procent av ett hushålls totala elkonsumtion kommer från hemelektronikens standby-el (1). Produktionen av denna el innebär utsläpp av växthusgaser som påverkar klimatet.

Ett alternativ för hushåll att sänka sin elkonsumtion är s.k. hemautomation. Med hemautomation menas ett system som tillåter en användare att kontrollera och automatisera elektroniken i sitt hem. Det kan användas för att göra det bekvämare, men också för att låta användaren effektivisera sitt elanvändande genom att exempelvis eliminera standby-el och se till att apparater som inte används automatiskt stängs av.

1.2 Problembeskrivning

Idag finns det metoder för kontroll av produkter som värme, belysning, vitvaror och annan hemelektronik. Men de vanligaste lösningarna bygger på gammal teknik, som termostater och timers. Det finns också mer moderna och effektivare system som "Smarta hus", som installeras i samband med husets uppbyggnad. Men de är ofta dyra och kräver utbildning för att installera och underhålla. Det finns även enklare lösningar som bygger på fjärrstyrning av produkter i hemmet, men deras syfte är ofta att ge ökad kontroll av apparater snarare än att automatisera användandet av dem.

För att hemautomation ska bli utbredd, behövs ett hemautomationssystem som är mer effektivt, prisvärt och användarvänligt än alternativen som finns idag.

1.3 Problemformulering

Hur kan man göra det mer attraktivt för privatpersoner att använda hemautomation för att minska sin miljöpåverkan?

Kan man utveckla ett system för hemautomation som är effektivt, prisvärt och användarvänligt?

1.4 Syfte

Projektets syfte är att ta fram ett alternativ för hushåll att sänka sin elkonsumtion. Detta genom att utveckla ett system för hemautomation som gör det möjligt för privatpersoner att kontrollera sin elkonsumtion, och således sin klimatpåverkan. I den här rapporten diskuteras nyttan och utformningen av ett sådant system, samt hur utvecklingen av det genomförs.

1.5 Mål med arbetet

Målet med arbetet är att utveckla ett system som består av en central enhet som installeras i en användares hemnätverk. Denna enhet kan kommunicera med sensorer och fjärrströmbrytare som finns i hemmet. En sensor ska designas och utvecklas medan arbetet med fjärrströmbrytare enbart består av att ta fram sätt att kommunicera med existerande teknik.

I arbetet ingår också att utveckla ett användargränssnitt i form av en mobilapplikation.

1.6 Avgränsningar

Även om systemet går att applicera på flera områden så kommer bara användningen i ett hushåll att utvärderas, detta för att det räcker för att utvärdera produktens potential.

Systemet kommer att designas och utvecklas, men kommer inte att distribueras eller bli utförligt testat. Projektet fokuserar istället på att utvärdera idén om att man som privatperson på ett enkelt och billigt sätt kan automatisera elektronik i hemmet och därmed få en positiv inverkan på klimatet.

1.7 Disposition

Kapitel 2 - Metod: Här beskrivs hur arbetet har gått till väga, vilken projektform som har använts samt vilka redskap som har använts i utvecklingsfasen.

Kapitel 3 - Teori: Här beskrivs de olika teknologier som har tillämpats i arbetet samt de definitioner som är bra att känna till för att kunna ta till sig resultaten som visas senare. Här presenteras också en modell över systemet för att ge en klar bild över dess komponenter och

tillämpningar. Sist kommer en modellering av hur ett system kan användas och vilka besparingar som kan göras.

Kapitel 4 - Resultat: Här ges först en generell beskrivning av det egenimplementerade systemet. Sedan tas de olika komponenterna som ingår i lösningen upp i detalj, både dess funktion och teknisk implementering.

Kapitel 5 - Slutsatser: Här diskuteras om projektet uppnådde de utstakade målen och om resultatet gav något användbart.

Kapitel 6 - Framtida förbättringar: Här tas de saker upp som antingen var planerade men inte hann utvecklas, eller under projektets gång har funnits vara relevanta för framtida arbete.

2. Metod

Arbetet inleddes med att en planering gjordes där alla olika komponenter delades upp i mindre tidsuppskattade arbeten. Sedan prioriterades dessa arbeten baserat på vilka delar som ansågs mest vitala för projektet. Under projektets gång hölls mötet med examinator och vissa delar prioriterades då om för att bättre passa tidsplaneringen.

För att få en överblick av existerande lösningar på problemet, samt nyttan av att lösa det, utfördes en litteraturstudie av diverse vetenskapliga rapporter och publikationer. Det sökverktyg som har använts är publikationsdatabasen DiVA (2).

Information om de protokoll för kommunikation med fjärrströmbrytare, och för att i slutänden ersätta fjärrkontrollen med projektets egna system, kom från flera källor. Detta eftersom protokollen är slutna och man måste analysera trafiken de existerande fjärrkontrollerna sänder ut. Denna data kom från vad diverse hobbyister delat med sig av på internetforum (3) (4) (5). Dock även med denna hjälp krävdes en hel del "trial and error" innan en fungerande kod togs fram.

För att ta fram en mobilapplikation för Android som uppfyller de krav vad gäller design och funktion så krävdes en grundläggande undersökning av dokumentationen för operativsystemet. Android själva tillgängliggör all den tekniska dokumentationen (6) och även en del designtips. De uppmuntrar också programmerare till att använda de senaste programmeringskonventionerna för att få ett uniformt utseende på alla Android-applikationer.

För applikationens tema användes en s.k. temagenerator (7) där man anger ett färgtema man vill ha och får tillbaka de nödvändiga konfigurationerna och filerna.

Vid utvecklingen av mjukvaran så användes den agila programmeringsmetoden Pair Programming. Detta m.h.a. programmet Eclipse (8) tillsammans med plugin:et Saros (9) som speglar två programmerares projektmappar med varandra i realtid. För att göra klassdiagram av programmen har plugin:et UMLAid (10) använts.

För att skapa kretsscheman har verktyget Circuit Lab (11) använts.

3. Teori

Här följer först en definition av de termer som är nödvändiga att känna till för förståelse av de tekniska termer som följer. Sedan presenteras och förklaras en modell över det system som har behandlats i arbetet. Detta följs av förklaringar av- och bakgrunden till de lösningar för denna modell i ordningen:

- Android applikation
- Fjärrströmbrytare
- Raspberry Pi
- Sensorer
- Nätverkssäkerhet

I slutet av kapitlet ges även en insikt i miljödiskussionen bakom ett sådant system och en modellering av hur det kan användas, för att se dess nytta.

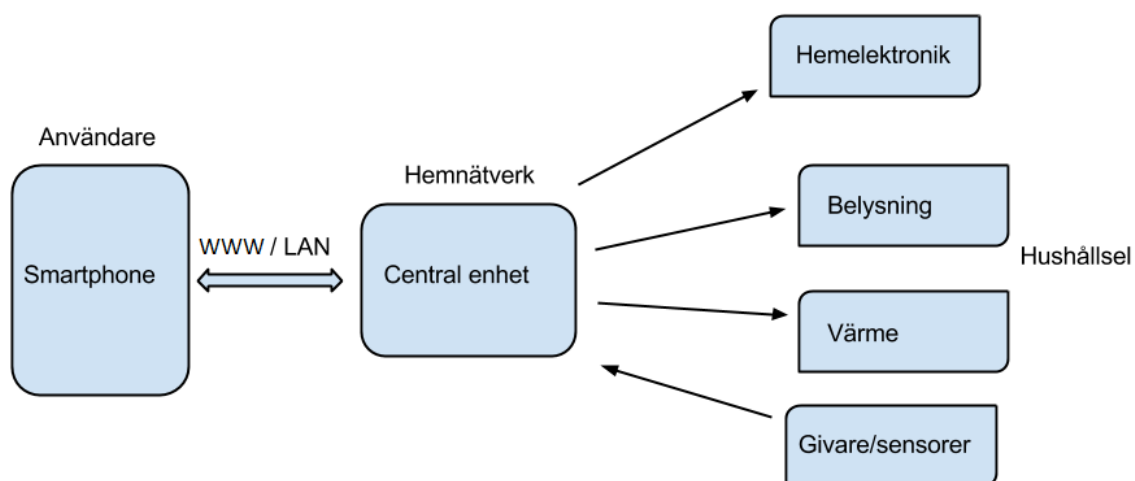
3.0 Definition av termer

- **Fjärrströmbrytare** - Ett relä som kopplas in mellan vägguttaget och en elektronikprodukt. Kan kontrolleras trådlöst.
- **Raspberry Pi** - En liten dator med I/O-gränssnitt som Ethernet, USB, HDMI, GPIO.
- **Android** - Ett Linux-baserat operativsystem anpassat för handhållna enheter, utvecklat av Google.
- **Hemkod** - En 26 bitar lång kod som gör varje strömbrytarnätverk (mer eller mindre) unikt.
- **PIC** - En familj av mikrokontroller-kretsar från Microchip Technology.
- **SBC** - Single-Board Computer, en dator med CPU, minne och I/O på samma kretskort.
- **On-/Off-Keying (OOK)** - Av/på-skiftning där en bärvågs amplitud skiftar mellan två lägen.
- **ZigBee** - En standard för trådlös kommunikation mellan enheter i t.ex. hushåll och fabriker.
- **Transciever** - En enhet som kan sända och ta emot radiosignaler.
- **SPI** - Serial Peripheral Interface, en buss för synkron seriell kommunikation.
- **UART** - Universal Asynchronous Receiver/Transmitter, en hårdvarukomponent som ger stöd för överföring av parallell data till seriell.
- **Nätverksmodul** - En enhet som har en TCP/IP-stack.
- **TCP/IP** - Protokollstack som används vid kommunikation över Internet och LAN.
- **GPIO** - General Purpose Input/Output. Port som kan ge ut/ta emot logiska binära signaler.

- **NTC-resistor** - Även känt som “Thermistor” (sv. termoresistor), vars resistans varierar med dess temperatur vilket gör det möjligt för den att användas som termometer.
- **RSA** - Asymmetrisk krypteringsalgoritm som bland annat används av SSL.
- **SSL** - Secure Sockets Layer, säkerhetsmekanism som används för att kryptera nätverkstrafik mellan två enheter.
- **TCP** - Transmission Control Protocol. Högnivåprotokoll för överföring av data i form av mindre uppdelade paket.
- **JSON** - Javascript Object Notation. En minimalistisk representation av kodobjekt i textform.
- **gson** - Ett Java-bibliotek som konverterar Java-object till JSON-strängar, och vice versa.
- **SQLite** - Structured Query Language Light(SQL-light). En lättviktsversion av standardspråket för interaktion med databaser.
- **Knudsen CC5X** - Kompilator som används vid programmering av PIC10/12/14/16.
- **Pair Programming (sv: Parprogrammering)** - En agil metod för mjukvaruutveckling. Det går ut på att en programmerare skriver kod, medan den andra observerar.
- **GreenBox** - Den centrala enheten som installeras i användarens hemnätverk. En modifierad Raspberry Pi som utgör en brygga mellan användaren och fjärrströmbrytarna.
- **GreenHouse** - Android-applikation, systemets användargränssnitt.

3.1 Systemmodell

Systemet är uppbyggt av tre komponenter; fjärrströmbrytare och sensorer som placeras runt om i hemmet, en central enhet som står för kontroll av strömbrytare och sensorer, samt ett användargränssnitt som ger användaren kontroll över hemmet.



Figur 1 visar en översikt över hur systemet ser ut.

Användaren monterar fjärrströmbrytarna mellan elnätet (dvs. vägguttagen) och diverse elektroniska produkter som ska kontrolleras, denna montering är enkel och kräver ingen

utbildning eller licens för att genomföra. Fjärrströmbrytare finns att köpa i de flesta elektronikaffärer och kostar mellan 50 och 500 kr. Systemet stödjer ett flertal olika protokoll, så vilka strömbrytare som används är upp till användaren.

Någonstans i mitten av hemmet installeras den centrala enheten som har direkt kontroll över fjärrströmbrytarna. Detta är en liten dator som har tillräckligt med minne och beräkningskraft för att uppehålla en nätverksanslutning, samt köra ett avancerat operativsystem. Enheten kopplas in på användarens hemnätverk för att kunna kommunicera över Internet eller LAN, detta utgör kopplingen mellan hemmet och användaren. All kommunikation mellan den centrala enheten och fjärrströmbrytarna sker trådlöst över ett öppet frekvensband.

Användaren kontrollerar systemet med hjälp av sin smartphone; genom en mobilapplikation kan användaren kontrollera fjärrströmbrytare över det lokala nätverket eller Internet. Detta ersätter de fjärrkontroller som går att köpa till fjärrströmbrytarna. Applikationen kan också användas för att schemalägga olika aktiviteter beroende på användarens position, tid på dygnet, rumstemperatur, strömförbrukning och andra relevanta givare.

3.2 Android-applikation

Android är ett relativt nytt operativsystem som till en början utvecklades av Android Inc, och som 2005 köptes upp av Google. Operativsystemet är avsett för mobila enheter som smartphones och surfplattor, men går även att installera på andra system då det är Linux-baserat. I maj 2013 tillkännagav Google att det fanns 900,000,000 aktiverade Android-enheter, en siffra som väntas stiga till 1,000,000,000 under samma år (12).

Android som plattform passar projektet väl då den gör det möjligt att enkelt utveckla och distribuera applikationen till så många användare som möjligt, utan att projektet behöver betala några övriga utvecklings- och distributionsavgifter, utöver de \$25 man betalar för att registrera sig som utvecklare på Google Play Store. Detta kan jämföras med kostnaden för att utveckla applikationer för Apple iOS, vilken är \$99 per år (13). Målet med projektet är att göra systemet så billigt och lättillgängligt som möjligt, och att välja den plattform som är störst och billigast är en del i arbetet att nå det (14). Därför används Android över exempelvis iOS.

Ett annat alternativ till Android är att utveckla en applikation i HTML/Javascript som kan öppnas i en webbläsare oberoende av plattform. Det kan då användas av en större mängd människor, samt kunna distribueras till ett lägre pris. Problemet med webbläsarbaserade

program är att de inte kan köras utan att användaren direkt interagerar med dem på samma sätt som en Android-applikation (service) kan. Programmet behöver exempelvis kunna schemalägga olika händelser baserat på tiden på dygnet, samt användarens position, utan att användaren har webbläsaren öppen konstant. Eftersom programmet främst ska köras på en enhet man alltid har med sig passar det bättre att använda det gränssnitt som Android erbjuder.

3.3 Fjärrströmbrytare

För att fjärrstyra sitt hem med hjälp av en fjärrkontroll finns det idag många produkter på marknaden. Fjärrströmbrytarna kommer från många olika tillverkare men har i grunden mer eller mindre samma hårdvara. Det kan egentligen brytas ner till två olika grupper av fjärrströmbrytare. En äldre version där

användaren manuellt måste ställa in adressen, och de nyare som är vad man kallar "självlärande"

där användaren aktiverar en programmeringsfas där fjärrströmbrytaren tar till sig den kod som sänds ut och ställer in sin egen adress till denna. Den äldre versionen blir allt ovanligare eftersom den kostar lika mycket som den nya och den begränsar hur många fjärrströmbrytare man kan använda. Detta eftersom den kräver ett användargränssnitt för att ställa in adressen.

Alla fjärrströmbrytare använder sig av den öppna frekvensen 433 MHz, detta ger en fjärrkontroll en räckvidd på ca 20m i fri sikt. För att undvika att man kommer åt grannens strömbrytare så har varje fjärrkontroll en unik kod (hemkod) som blir gemensam för alla som ingår i det hushållets fjärrströmbrytarnät. Denna kod ingår i adressen den använder för att adressera olika fjärrströmbrytare. Detta kan liknas med hur alla hus i ett kvarter har en gemensam postkod som associerar dem med det området.

Tekniken för att överföra data till fjärrströmbrytarna är OOK, on/off-keying. Det är en enkel form av amplitudskiftning (15) där en digital etta och nolla representeras av närvaron respektive frånvaron av en bärvåg, d.v.s. en våg som bara har amplituden noll eller något som inte är noll.



Figur 2 visar två stycken fjärrströmbrytare och en fjärrkontroll.

Andra lösningar för fjärrströmbrytares kommunikation som antingen används eller kan användas i andra implementationer är:

ZigBee - ZigBee (16) är en specifikation för kommunikation mellan flera små trådlösa och strömsnåla enheter. Alla enheter som har stöd för ZigBee kan ingå i ett s.k. “mesh-nätverk” där alla noder kan vidarebefordra meddelande mellan varandra. Detta tillåter meddelanden att skickas mellan två noder som inte är inom räckvidd för direkt anslutning och gör att ZigBee fungerar bra även i tillämpningar på stora områden. Den kommunicerar över 2.4 GHz bandet och har en teoretisk överföringshastighet på 250Kbit/s samt en räckvidd på upp till 1000 meter.

En lösning där all kommunikation sker m.h.a. ZigBee skulle ha många fördelar även utöver den höga hastigheten och långa räckvidden. Bland annat skulle det vara enkelt att koppla in många apparater tack vare stöd för adressering, som är inbyggt i protokollet. Det skulle också tillåta användaren att koppla in många andra kommersiella produkter (17) som redan använder ZigBee.

2.4 GHz Transceiver - Den billigaste lösningen som fortfarande erbjuder 2-vägs kommunikation är att använda s.k. “transceivers” som består av en mottagare och en sändare med ett gränssnitt så som SPI eller UART. Det finns transceivers med olika frekvenser, men den som är mest lämplig är 2.4 GHz som erbjuder bra räckvidd samtidigt som det är en öppen frekvens i större delen av världen. Den mest populära produkten av denna typ är nRF24l01 (18).

Nätverksmodul - En lösning hade kunnat vara att undvika all form av kommunikation utöver en anslutning till det lokala hemnätverket. Detta antingen genom en ethernetladd eller en trådlös modul. Varje sensor och fjärrströmbrytare skulle i detta fallet ha en egen nätverksmodul och vara uppkopplade på hemnätverket för att kunna kommunicera med TCP/IP. Nackdelen med denna lösning är att nätverksmoduler är dyrare än de andra lösningarna och den MCU/processor som sensorn eller fjärrströmbrytaren innehåller skulle även den bli dyrare. Detta eftersom det krävs en relativt stor del minne och beräkningskapacitet för att upprätthålla en TCP/IP-stack.

Det kan också diskuteras om den centrala enheten ens skulle vara nödvändig i detta fallet eftersom mobilapplikationen tekniskt sett skulle kunna prata direkt med sensorerna och

fjärrströmbrytarna. Det är även troligt att om de har kapacitet nog att ha en TCP/IP-stack skulle de också vara kapabla till att komma ihåg schemaläggning och själva utföra denna.

3.4 Raspberry Pi

Raspberry Pi är en liten SBC (Single Board Computer) som säljs av Raspberry Pi Foundation (19). På kretskortet finns det en ARM11-processor, tillsammans med flera olika typer av gränssnitt, t.ex. Ethernet, USB, HDMI samt 20 st. GPIO- (General Purpose Input/Output) kontakter som användaren själv kan konfigurera.

Det som utmärker den är att den har ett väldigt lågt pris i förhållande till beräkningskraft samt att den är relativt strömsnål tack vare just ARM-processor.

Detta tillåter användning av ett operativsystem som klarar av att ha en nätverksstack och flera trådar/schemaläggning av flera program. Detta i sin tur leder till att man kan skriva program med hög abstraktion i högnivåspråk och utnyttja existerande drivrutiner och färdiga nätverksimplementationer.

Det som skiljer Raspberry Pi från andra SBC:er, och MCU:er (Microcontroller) är att även om den inte har det absolut bästa priset i förhållande till hårdvara som t.ex. BeagleBone (20), CubieBoard (21) och ODroid-U2 (22) så har den ett skräddarsytt operativsystem, och det finns även en väldigt stor "community" som ofta är villig att hjälpa till och dela med sig.

Det operativsystem som används är Raspbian, vilket är baserat på Linux-distributionen Debian, och är optimerat för Raspberry Pi (23).

3.5 Sensorer

Eftersom projektet fokuserat på hushåll så togs en sensor som mäter temperatur fram.

Sensorn består av en 8-bitars MCU från Microchip som kallas PIC16F690 (24), den används för att den har stöd för UART och A/D-omvandling. A/D-omvandlingen används för att mäta ett analogt värde från en NTC-resistor (25) vars resistans varierar beroende på temperatur. Detta analoga värde representerar linjärt temperaturen och kan därför omvandlas till ett digitalt värde som sedan kan beräknas till den faktiska temperaturen. När denna omvandling är klar så skickas värdet till Raspberry Pi:n m.h.a. UART som är hårdvarustödd seriell kommunikation.

Att PIC16F690 användes berodde helt på tillgänglighet. Det finns billigare, mindre och strömsnålare MCU:er på marknaden, men leveranstiden för dem ledde till att PIC16F690 valdes.

Den NTC-resistor som används är av typen MCP9700A (26) och användes för att den var billig, vilket var ett villkor för sensorn. Den har dock bristen att den bara har en mätnoggrannhet på ca ± 2 grader Celsius.

Kommunikationen mellan sensorn och Raspberry Pi:n sker med sladd för att visa på konceptet men framtida versioner kan använda sig utav trådlös kommunikation, antingen direkt påkopplad på USART pinnarna eller så får koden till PIC:en skrivas om för eventuell förändring. Detta är en annan anledning till att PIC valdes, den har tillräcklig kapacitet för att kunna ingå i en utvecklad version av systemet, där flera sensorer ingår och kommunicerar trådlöst. Detta dels för att den har lagringsmöjlighet för en eventuell adress och dels för att den har tillräcklig beräkningskapacitet samtidigt som den är strömsnål.

3.6 Nätverkssäkerhet

För att minimera risken att någon obehörig utnyttjar systemet med fjärrströmbrytare sker all kommunikation med den centrala enheten enligt SSL-protokollet. SSL i sin tur använder genererade certifikat som innehåller RSA-nycklar. För att kunna skicka ett meddelande så måste både mottagare och sändare ha likadana certifikat för att kunna kryptera och dekryptera meddelandet.

SSL har få säkerhetshål och det enda egentliga sättet för någon obehörig att hacka sig in på denna kommunikation är att utföra en tvåvägs "man in the middle" attack. Där denna person har fått tag på certifikatet och lyckas imitera båda sidorna av kommunikationen. Det är dock inte mycket man kan göra åt detta och i grunden är inte krypteringen till för att hålla ute folk som verkligen vill hacka sig in utan mer för att hindra data från att skickas i klartext, då det hade varit enkelt att urskilja enskilda meddelanden.

Vidare så har varje Raspberry Pi:n sin egna unika hemkod, denna visas för användaren och kan när som helst ändras i applikationen. Denna hemkod används i adresseringen för alla fjärrströmbrytare i fjärrströmbrytarnätet och minimerar risken för att en granne ska kunna komma åt systemet. Även om grannen är inne på samma hemnätverk, kan nätverksadressen till Raspberry Pi:n, och har laddat ner mobilapplikationen så måste denna fortfarande veta hemkoden som står på LCD-displayen.

3.7 Miljö

Idag kommer ungefär 37 procent av Sveriges elkonsumtion från bostads- och servicesektorn (1). Detta kan ses i relation till att de totala växthusgasutsläppen till 13 procent kommer från el och värmeproduktion (27).

En källa till denna elkonsumtion är en som inte är allmänt känd. Runt 10 procent av ett hushålls totala elkonsumtion kommer från hemelektronikens standby-el. En enskild apparats elförbrukning under dess livstid kan upp till 40 procent bestå av standby-el (1).

Utöver bara standby-el då elektroniken i en användares ögon faktiskt är avstängd är den el som t.ex. en lampa använder när man inte är i rummet lika bortslösad och är något som hade kunnat lösas m.h.a. hemautomation.

Så förutom att ha en direkt påverkan på ett hushålls elkonsumtion så finns det även beteendemässiga förändringar ett hemautomationssystem kan bidra med. Om det på sikt finns möjlighet för användaren att få återkoppling på sin elanvändning så har det påvisats att det uppmuntrar till minskad elkonsumtion (28). Systemet skulle även göra det möjligt att sprida ut sin elkonsumtion på dygnet genom att t.ex. schemalägga tvättmaskinen till att tvätta på natten, vilket skulle leda till att hushållet sparar pengar men också till att på sikt på en större t.ex. nationell skala sprida ut den totala elkonsumtionen på dygnet. Detta skulle leda till mindre spikar i elanvändningen vid de vanligaste tiderna runt lunch, middag och kväll. För ett land som Sverige där mycket av vår elproduktion kommer från klimatvänlig vattenkraft men där kraftiga belastningar på elnätet kan leda till att det krävs reservkraftverk som ofta är mindre miljövänliga, skulle på sikt denna minskade belastning leda till mindre miljöpåverkan (29) (30).

3.7.1 Modellerings av energibesparingar

För att se hur stor påverkan ett hemautomationssystem kan ha på en lägenhets elförbrukning och miljöpåverkan har följande mätningar gjorts i en studentlägenhet i Kista. Lägenheten är 25kvm stor och består av ett rum. Den del som står för den största elförbrukningen, bortsett från belysning och värme, är ett s.k. underhållningssystem. I det ingår en PC, en bildskärm, en TV, två externa hårddiskar och ett högtalarsystem. Mätningar har gjorts för att jämföra systemets energiförbrukning vid olika belastning: standby, låg/inaktiv och hög/aktivt användande. Standby är då systemet inte används över huvud taget, allt står i standby-läge. Låg/Inaktiv är då allting är på, men inte används. I dagsläget är det detta tillstånd som används under större delen av dygnet då det ska gå snabbt att börja använda PC:n när man

behöver den. Hög/Aktiv är det läge då skärmen och TV:n är på, PC:n är under maximal belastning, högtalarna spelar upp ljud och de externa hårddiskarna används regelbundet.

Mätningarna gjordes med en energimätare som kopplas mellan vägguttaget och grenuttaget som all elektronik är kopplad till. Den anger den effekt (Watt) som förbrukas under de olika tillstånden. För att få en bra modellering av systemet gjordes även samma mätning av vad den centrala enheten, alltså Raspberry Pi:n och en fjärrströmbrytare har för energiförbrukning och tagit med detta i beräkningarna. Mätningarna användes sedan tillsammans med antaganden (tid för aktivt användande) och data (elpris, utsläpp) för att beräkna energiförbrukning, kostnad och koldioxidutsläpp.

| Uppmätt Energiförbrukning (W) | Standby | På/Inaktiv | Aktivt användande |
|---|-----------------------------------|--------------------------|---------------------------------|
| Underhållningssystem | 42 | 320 | 470 |
| GreenBox: | 2,4 | 2,4 | 2,4 |
| Raspberry Pi | 2,35 | 2,35 | 2,35 |
| LCD | 0,04 | 0,04 | 0,04 |
| Sändare (försömbär) | 0 | 0 | 0 |
| Temperatursensor | 0,0000792 | 0,0000792 | 0,0000792 |
| Fjärrströmbrytare | 1 | 1 | 1 |
| Beräknad Total energiförbrukning (W) | Standby/Inaktiv | Aktivt användande | |
| Underhållningssystem utan GreenBox | 42 | 470 | |
| Underhållningssystem med GreenBox | 3,3900792 | 473,3900792 | |
| Tid (h/dag) | Standby/inaktiv | Aktivt användande | |
| Underhållningssystem | 16 | 8 | |
| Variabler | | | |
| Elpris (kr/kWh) | 1,5 | | |
| Koldioxidutsläpp (kg/kWh) | 0,020 | | |
| Resultat | Energiförbrukning (kWh/år) | Kostnad (kr/år) | Koldioxidutsläpp (kg/år) |
| Utan GreenBox (utan standby) | 3 241 | 4 862 | 65 |
| Utan GreenBox (med standby) | 1 618 | 2 427 | 32 |
| Med GreenBox (Ersätter standby & inaktiv) | 1 402 | 2 103 | 28 |

Figur 3 visar data som har använts för att beräkna fram resultatet i Tabell 1.

| Resultat | Energiförbrukning (kWh/år) | Kostnad (kr/år) | Koldioxidutsläpp (kg/år) |
|---|----------------------------|-----------------|--------------------------|
| Ingen automation (ingen standby) | 3241 | 4862 | 65 |
| Ingen automation (standby) | 1618 | 2427 | 32 |
| Med automation (Ersätter standby & inaktiv) | 1402 | 2103 | 28 |

Tabell 1 visar vilka energi-, kostnads- och miljöbesparingar som kan göras med ett underhållningssystem. Dels genom att låta systemet gå i standby-läge, när det inte används, och dels genom att automatisera hanteringen av när systemet ska vara på/av.

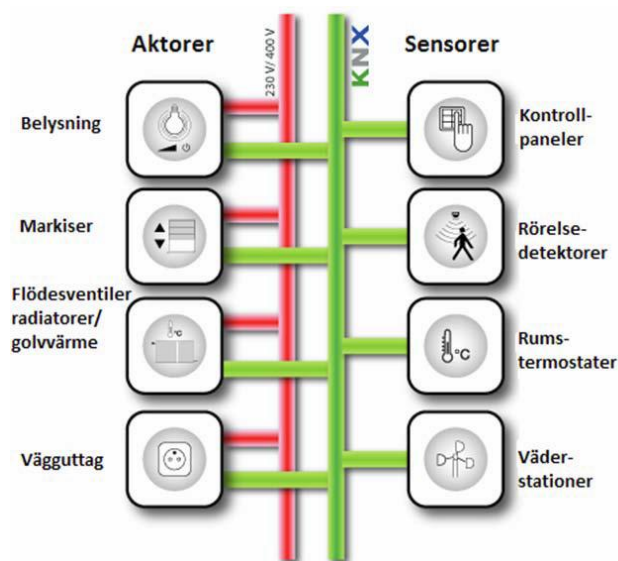
På rad tre i Tabell 1 ser man att den beräknade energiförbrukningen hos underhållningssystemet (inklusive automationssystemet) är 1423 kWh/år, vilket motsvarar en minskning med ungefär 12 % jämfört med det som förbrukas när man ställer det i standby så fort det inte används. Detta leder i sin tur till en minskning av det årliga koldioxidutsläppet från elproduktionen med 4 kg/år. Man får ha i åtanke att koldioxidutsläppen för elproduktion varierar mycket mellan olika länder. I beräkningarna har det antagits att 1 kWh motsvarar 20g koldioxid, vilket är ett medelvärde för ett normalår i Sverige. Detta kan jämföras med utsläppen från europeisk elproduktion, som ligger nära 450 g/kWh (31).

3.8 Existerande teknik/Liknande lösningar

Hemautomation är inte ett helt nytt koncept; det har länge funnits system som är ämnade för att sänka energikostnader och ge bättre kontroll över elektronik i hemmet. Ända sedan 1970-talet då X10, ett protokoll för kommunikation över elnätet, utvecklades av Pico Electronics, har nya tekniker för automation och kontroll ständigt utvecklats. Nedan presenteras några av de mest använda systemen i hushåll idag.

3.8.1 KNX

KNX är ett av de mer avancerade systemen och är det som är mest utbrett i Sverige (32). Tekniken ägs och utvecklas av KNX Association, som består av ett antal tillverkare som tillämpar systemet i byggnationer och vid utveckling av nya applikationer som belysning, värme, ventilation, bevakning, mätning, osv.



Figur 4 visar en översikt av KNX-systemet och hur det är kopplat till olika aktorer i hemmet. (32)

I ett hus som använder sig av KNX kopplas alla enheter som ska kontrolleras, och alla kontroller som skärmar, sensorer och ljusknappar till en central enhet. Denna koppling sker ofta genom ledningar som går runt i hela huset, men det finns också trådlösa alternativ, även om det sällan används. På så vis får systemet full kontroll över all elektronik, värme och belysning i hemmet vilket gör att allt kan automatiseras genom mjukvara.

KNX används främst inom industrin eller i

lyxvillor men är inte utbrett bland vanliga bostäder. Detta beror antagligen på att det kostar uppskattningsvis 1500 kr/m² att installera systemet (32).

3.8.2 Insteon

Insteon är ett hemautomationssystem där fokuset är att kunna kontrollera existerande produkter, och samtidigt använda produkter framtagna av Insteon (33) själva. Produkter så som lampor, dörrklockor, rörelsesensorer och brandvarnare. Systemet har en central enhet som användaren kan kommunicera med m.h.a. en mobilapplikation. En anledning till att systemet uppmuntrar till att använda Insteons produkter är att de har en s.k. repeaterfunktion där alla signaler som tas emot skickas vidare. Detta tillåter användning av systemet över stora ytor så länge det finns en jämn spridning mellan alla enheter. Alla av Insteons produkter har även möjlighet till tvåvägskommunikation.

Fördelar är att det är enkelt att använda och det skalar bra i ett stort hus samt små lägenheter. Nackdelar är att det är dyrt, en central hubb kostar 1000 kr och en vanlig fjärrströmbrytare kostar nästan 400 kr. Systemet finns inte heller lättillgängligt på platser utanför USA och den mobilapplikation som Insteon själva har utvecklat har fått låga betyg.

3.8.3 TellStick

Tellstick är en samling av olika typer av fjärrkontroller producerade utav Telldus Technologies (34).

De finns i flera olika format; antingen som en liten USB-dosa som kopplas till en PC och som sedan kan med hjälp av Telldus egna program eller användarens, kommunicera med enheter över 433 MHz-bandet. Det finns även i varianter som klarar av att ta emot signaler och på så sätt ha tvåvägskommunikation. En original Tellstick kostar 600 kr och en Tellstick Duo (tvåvägskommunikation) kostar 800 kr.

Fördelarna med en original Tellstick och en Tellstick Duo är att de fungerar med alla fjärrströmbrytare på marknaden. Nackdelen är att de är svåra att installera och kräver programmeringskunskaper för att kunna användas.

En annan produkt de säljer är Tellstick Net som fungerar som en förmedlingsnod på hemnätverket och som vidarebefordrar det den tar emot över TCP/IP-protokollet genom att sända ut till enheter över 433 MHz-bandet. Användarens gränssnitt består av en mobilapplikation eller Telldus hemsida.

Fördelen med en Tellstick Net är att den är enkel att installera och användaren har möjligheten att själv välja vilken mobilapplikation som den vill använda för att kontrollera sin hem. Nackdelen är att den kostar runt 1000 kr och det finns inga bra professionellt utvecklade applikationer och den Telldus själva erbjuder har väldigt lågt betyg.

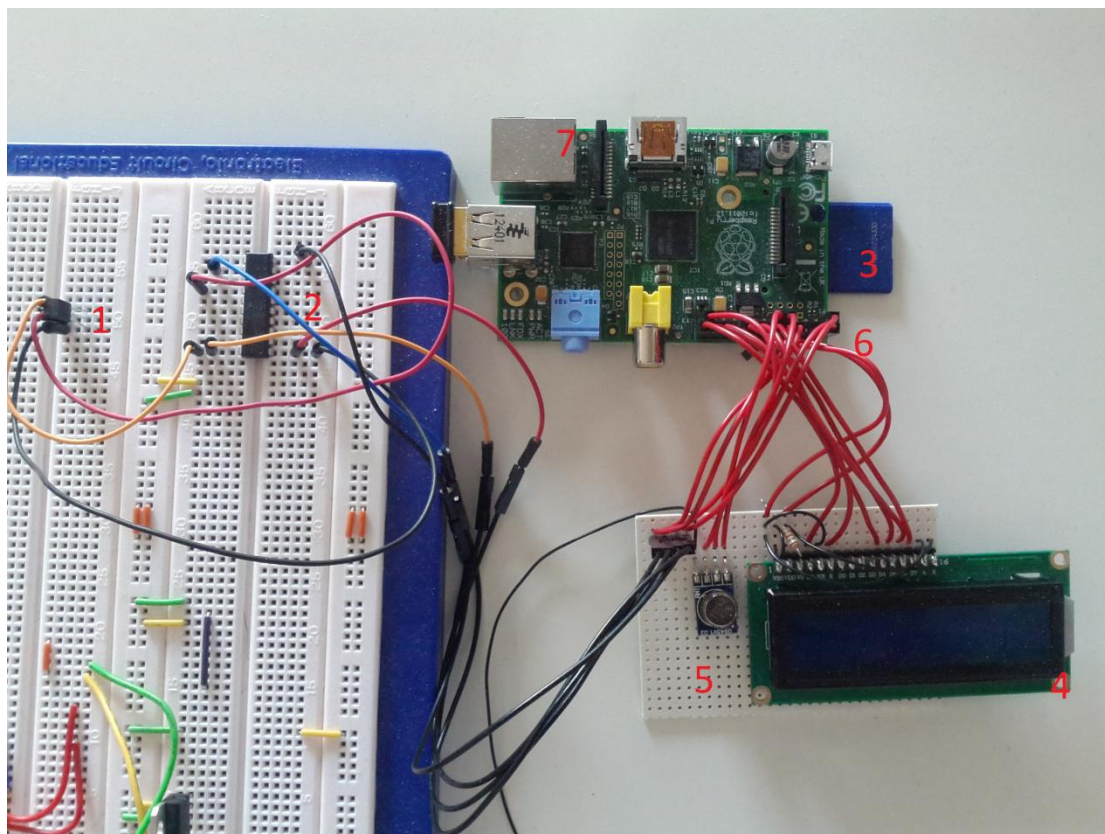
4. Resultat

Arbetet har resulterat i en produkt som är färdig att användas efter att ha följt konfigurationsguiden i manualen. Den består av tre delar som har utvecklats under projektets gång; centralenheten GreenBox, Android-applikationen GreenHouse samt en sensorkrets. Utöver dessa används två komponenter som inte utvecklats på egen hand, men som krävs för att systemet ska kunna användas; fjärrströmbrytare som installeras i hemmet och en plattform med operativsystemet Android, förslagsvis en smartphone.

Alla klassdiagram, kretsscheman och källkod finns bland bilagorna.

4.1 GreenBox

GreenBox är en nätverkskomponent i form av en låda med en ethernetanslutning och ett eluttag. Det är den centrala enhet som installeras i användarens hemnätverk. Plattformen som all mjukvara körs på är Raspberry Pi. Den kopplas in på hemnätverket och kan sedan utnyttjas utav en Android applikation som fungerar som användargränssnitt. Greenbox har även en LCD-display där den lokalt tilldelade IP:n visas tillsammans med den unika hemkoden, som tillåter användaren att ansluta sin mobil till sin Greenbox.



Figur 5 visar en bild på GreenBox och dess interna komponenter

- 1: NTC-Resistor
- 2: PIC16F690
- 3: Raspberry Pi
- 4: LCD-display
- 5: 433 MHz sändare
- 6: GPIO-pinnar
- 7: Ethernetport

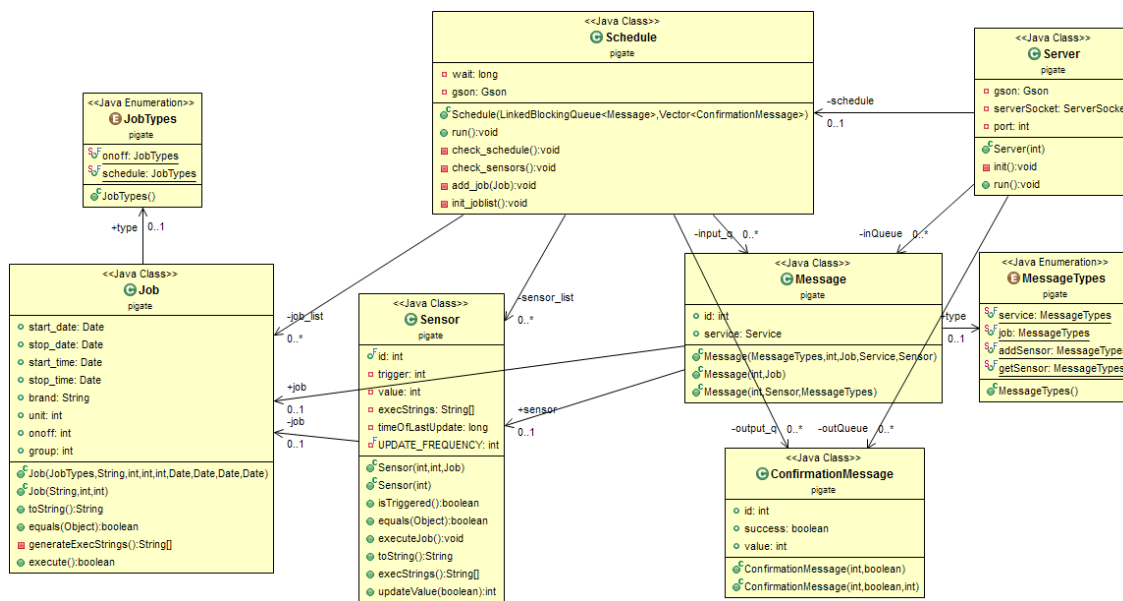
GreenBox:s uppgift är att ta emot TCP-meddelanden från användaren och utföra de eventuella kommandon som finns i meddelandet. De olika kommandona är På/Av-jobb, Schema-jobb, Sensor-jobb, och Service-jobb. På/Av-jobb är för att signalera fjärrströmbrytare huruvida de ska vara av eller på. Schema-jobb är för att lägga till ett På/Av-jobb i schemaläggaren som körs parallellt med TCP-servern. Sensor-jobb är när användaren vill få data från en sensor som är kopplad till Greenbox:en, när datan hämtats sänds den sedan tillbaka till användaren. Service-jobb är för att starta om Greenbox:en eller göra eventuella uppdateringar.

Både TCP-servern och schemaläggaren är skrivna i Java 8 Embedded version. Detta för att det är den java som har bäst prestanda och Embedded-versionen innebär att den är optimerad för ARM-processorer. All interaktion med hårdvara sker genom att starta en subprocess som exekverar ett Python-skript av Python version 2.7.3, som är standard på Rasbian OS. Skriptet interagerar med hårdvaran och skriver sedan ut resultatet. Detta kan då läsas in från Java-

programmen. Anledningen till denna konfiguration är för att Python hade mer stöd vad gäller användning av hårdvara och det ger ett abstraktionslager mellan huvudprogrammen och hårdvaran. Om man senare vill byta plattform från Raspberry Pi:n till en annan så skulle man endast behöva byta biblioteket som används för att kontrollera hårdvaran.

4.1.1 Server

Serverprogrammet är en TCP-server som väntar på nya TCP-meddelanden, utför eventuella kommandon och skickar sedan tillbaka ett konfigurationsmeddelande om hur utförandet gick.

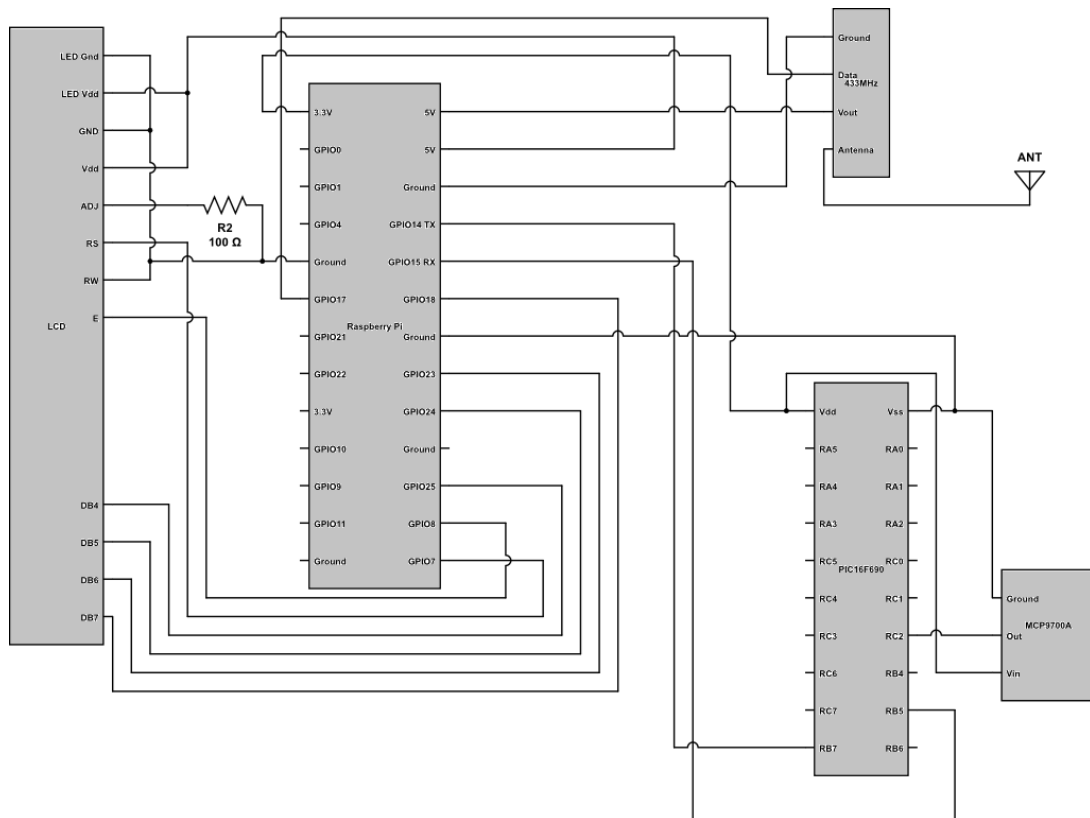


Figur 6 Klassdiagram över hur programmen TCP-server(Server) och Schemaläggare(Schedule) på Greenbox interagerar med varandra.

Ett kommandomeddelande består av en JSON-sträng som formateras m.h.a. Googles egna bibliotek gson (35), som innehåller parametrarna id och jobb, där id är till för att kunna veta om man får tillbaka ett konfigurationsmeddelande och jobb är en java-klass. Jobb-klassen innehåller allt som behövs för att GreenHouse ska veta vad för typ av kommando det är och i så fall också vilket program som ska startas och vilka parametrar som behövs för att utföra kommandot.

4.1.2 Trådlös kommunikation

Använder sig av biblioteket GPIO som är förinstallerat på Rasbian OS och tillåter en att kontrollera GPIO-pinnarna från ett eget kodat program i språket Python.



Figur 7 Kopplingsschema över GreenBox.

Tack vare att fjärrströmbrytarna använder sig av OOK (On/Off-keying) är det enkelt att skicka en binär sekvens till en GPIO-pinne. Raspberryn har en hög klockfrekvens på runt 700MHz och den minsta tidsenheten en fjärrströmbrytare kan skilja på är 250 microsekunder, vilket man kan tolka som att den teoretiskt minsta klockfrekvensen för att använda en fjärrströmbrytare är 4kHz. Detta gör att det faktum att Raspberry Pi:en är enkärnig och måste utföra "content switching", där processorn byter mellan flera trådar, inte har någon direkt påverkan. I kombination med den höga frekvensen och att Python har bra bibliotek för att mäta tid är det därför en enkel sak att generera exakta signallängder för de olika värdena i sekvensen.

För att generera ett meddelande enligt fjärrströmbrytares protokoll så krävs ett flertal parametrar: grupp, enhet, kanal, id(hemkod), och om den ska vara på eller av. Dessa kan varieras från olika modeller men är ofta lika.

Det fungerar genom att först generera en 28 tecken lång sträng bestående av 1:or och 0:or som sedan går igenom från början till slut.

```
def send_data(self, id, group, onoff, channel):
    for i in range(REPEATS):
```

```

id &= 0xFFFFFC0
id |= ((group & 1) << 5)
id |= ((onoff & 1) << 4)
id |= ((~channel) & 15)
self.shift_out(id)

```

start/stop = **röd** onoff = **blå** group = **orange** hemkod = **mörkgrön** id/kanal = **grön**

```

100100111001010011111111000000
100100111001010011111111010000
100100111001010011111111010000
100100111001010011111111011110

```

Vilket ger strängen:

```

100100111001010011111111011110

```

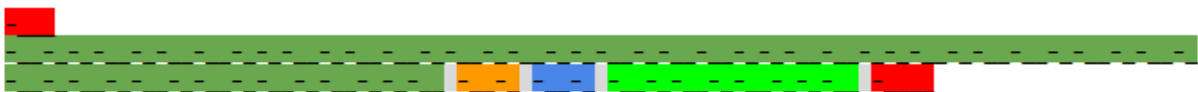
Strängen gås sedan igenom teckenvis. Om tecknet är en 1:a så skickas 1bit och sedan en 0bit, och om det är en 0:a så skickas en 0bit och sedan en 1bit.

```

10 1001011001011010100101100110010110101010101001 10 10 10101001 10

```

En 1bit består av en kort hög puls följt av en lång låg puls. En 0bit består av en kort hög puls följt av en kort låg puls, en sekvens inleds även med en kort hög och en lång låg puls och avslutas med en kort hög och en väldigt lång låg puls.



Standarden för fjärrströmbrytare idag är att utnyttja 433 MHz bandet. Anledningen till detta är främst för att frekvensen är öppen både i Europa och USA men också för att den är billig i förhållande till dess räckvidd. I öppen sikt är räckvidden ca 20 m, vilket är passande för inomhusbruk.

En nackdel med att utnyttja existerande system för fjärrstyrning är dels att dagens lösningar bara tillåter envägskommunikation, vilket leder till att man inte kan få bekräftelse om meddelanden har tagit emot. Dels för att 433 MHz tekniken begränsar om man vill ha ett system som sträcker sig längre än en lägenhet eller mindre hus.

4.1.3 Schemaläggare

Schemaläggaren är en process skriven i Java som körs parallellt med TCP-servern, med vilken den kommunicerar m.h.a köer implementerade genom Javas Queue klass. Dess uppgift är att hålla reda på tillagda fjärrströmbrytare och sensorer och se till att de aktiveras vid de tider användaren angett.

All data som kommer till schemaläggaren kommer från TCP-servern, som har läst av meddelandet och sett att det var något som schemaläggaren ska behandla. Detta meddelande läggs då i kön mellan processerna. Schemaläggarens program börjar med att kontrollera om det finns något nytt i kön, finns det så placeras det in i rätt lista beroende på om typen är en sensor eller en fjärrströmbrytare. Denna lista sparas sedan ner till minnet direkt ifall strömmen skulle gå så att ingen data går förlorad. Sedan söks båda listorna igenom efter jobb vars villkor för exekvering är sant. Detta görs genom att t.ex. jämföra sensorns nuvarande värde med ett av användaren angett tröskelvärde och om det är sant i så fall utföra det jobb som är förknippat med den sensorn. Är det ett På/Av schemajobb så kollas om tiden just nu stämmer in på den tiden som användaren angett.

Eftersom Raspberry Pi:n har begränsad kapacitet och användaren ska uppleva applikationen som snabb, måste schemaläggaren anpassas så att den inte stryper TCP-server processen, detta görs genom att den sover mellan varje iteration under en tid som är kalkylerad så att den är passiv så länge som möjligt utan att missa nästa schemahändelse.

4.1.4 LCD-display

GreenBox har en display som består av 2x16 tecken. Den används för att visa den IP och hemkod som användaren ansluter med från applikationen.

För att skriva till displayen används vanligtvis alla 8 av LCD:ns datapinnarna för att överföra en byte parallellt.

Raspberry Pi har ett begränsat antal GPIO-pinnar, därför används istället en teknik där 4 pinnar parallellt skickar de höga bitarna och sedan de låga bitarna för att bilda en byte.

Displayen använder den populära drivkretsen HD44780 som tillåter funktionalitet så som minne av text mellan skrivningar, programmering av egna tecken samt radbyten. Detta passar Greenbox eftersom det tillåter att ett enkelt Python-skript körs vid start av Greenbox och sedan bara när man vill göra ändringar. Texten tas då in i skriptet från startparametrarna som en kontinuerlig sträng med vanliga blanksteg mellan varje ord. En mening får max ha 32

tecken inklusive blanksteg och skriptet kommer försöka centrera så många ord som får plats på varje rad och fylla ut med blanksteg. Skriptet stödjer inte tecken utanför standard ascii (36).

Exempel på exekvering av skript:

“sudo python LCD.py This is an example of the lcd”

vilket skulle resultera i:

This is an examp

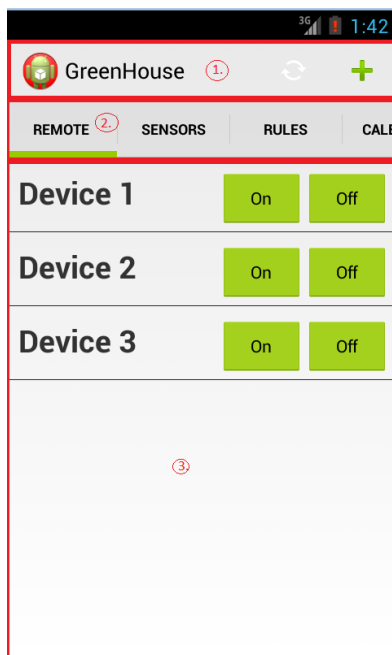
le of the lcd

4.2 GreenHouse

GreenHouse är applikationen som låter användaren interagera med hemautomationssystemet genom ett grafiskt användargränssnitt. Genom den kan användaren kontrollera fjärrströmbrytare i hemmet, läsa av värden på sensorer, sätta upp och kontrollera regler för olika strömbrytare, samt schemalägga olika händelser vid olika tidpunkter. Applikationen är kompatibel med Android API version 8 och uppåt, där API 8 motsvarar Android 2.2 Froyo.

4.2.1 GUI

Syftet med applikationen är att ge användaren kontroll över sitt hem på ett sätt som är lätt att förstå utan att ha någon teknisk kunskap om hur resten av systemet fungerar. Användargränssnittet ska vara så enkelt som möjligt samtidigt som det förser de funktioner som behövs för att användaren ska kunna reglera sin energiförbrukning och kontrollera apparater i hemmet. För att uppnå denna enkelhet och för att skapa ett GUI som är lika användbart på alla enheter oberoende av hårdvara och skärmstorlek, följer applikationen de standarder som gäller för Android (37).



Figur 8 beskriver de olika grafiska komponenterna.

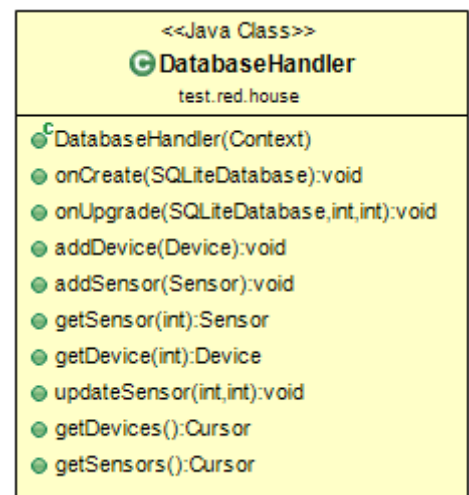
1. *ActionBar* - Här finns knappar som hjälper användaren att snabbt konfigurera det innehåll som ska visas. Applikationens namn och ikon som visas längst till vänster är där för att göra det klart för användaren vilken applikation det är som körs och hjälper den att förknippa innehållet med ett namn och en bild. Längst till höger finns en knapp som låter användaren lägga till enheter, sensorer, regler och scheman. ActionBar

introducerades med API version 11 (Android 2.3), så för att få samma design och funktion för äldre enheter används kompatibilitetsbiblioteket ActionBarSherlock (38).

2. *Tabs* - Gör det enkelt för användaren att byta mellan olika vyer, beroende på om den vill använda applikationen som en fjärrkontroll, se sensorvärden, mm. Användaren kan byta till en annan tabb genom att klicka på den eller genom att dra fingret horisontellt över skärmen i yta 3. Det finns fyra olika vyer, vars funktioner beskrivs senare i det här kapitlet. Även detta är en funktion som introducerades med API version 11, så det har implementerats med ActionBarSherlock.
3. *Tab Content/ListFragment/Huvudinnehåll* - Här visas innehållet i varje flik. I bilden visas innehållet i fliken "Remote" som visar de fjärrströmbrytare som användaren kan kontrollera. Här kan användaren hålla fingret på en enhet för att redigera/ta bort den.

4.2.2 Datalagring

För att kunna starta applikationen snabbt och för att kunna spara information mellan sessioner är all relevant data sparad i en SQLite-databas. För att underlätta hanteringen av databasen används klassen DatabaseHandler, som har metoder för skapande, insättning och radering av data. DatabaseHandler ser till att rätt information sätts in och hämtas, samt kontrollerar att inga skrivningar till databasen sker parallellt.



Figur 9 Klassdiagram (UML) för databashanteraren DatabaseHandler.

Databasen är uppbyggd av två tabeller; *devices* och *sensors*. De innehåller all information som är nödvändig för att kontrollera fjärrströmbrytare och sensorer. Där finns också information som underlättar synligheten för användaren, som namn och beskrivning på enheterna.

| | | | | |
|-------------------|--|---|--|---|
| <i>devices</i> | _id | _name | _brand | _group |
| Innehåll | INTEGER | TEXT | TEXT | INTEGER |
| Förklaring | Varje enhet har ett unikt id, användaren kan välja värdet på detta när man skapar en ny enhet. | Det namn som användaren ser i listan av enheter. Exempel: "Lampa", "PC", "Spis". | Anger vilken sorts strömbrytare det är, dvs vilket protokoll som ska användas för kommunikation. Exempel: "jula", "nexa", "proove". | För att kunna kontrollera en hel grupp av enheter med en knapptryckning. Användaren kan välja att gruppera enheter själv. |

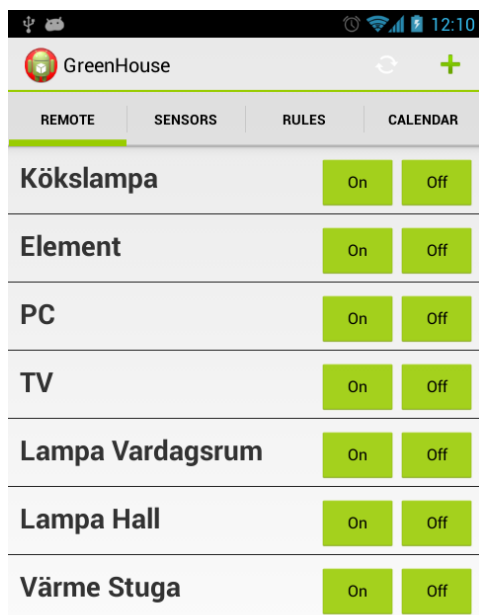
Tabell 2 - Visar hur databastabellen *devices* är uppbyggd.

| | | | | |
|-------------------|--------------------------------|--|--|---|
| <i>sensors</i> | _id | _name | _value | _type |
| Innehåll | INTEGER PRIMARY KEY | TEXT | INTEGER | TEXT |
| Förklaring | Ger varje sensor ett unikt id. | Det namn som användaren ser i listan av sensorer. Exempel: "Element", "Stuga". | Visar det senast uppmätta värdet för varje sensor. | Typ av sensor. Exempel: "temperatur", "ström", "effekt". |

Tabell 3 - Visar hur databastabellen *sensors* är uppbyggd.

4.2.3 Fjärrkontroll

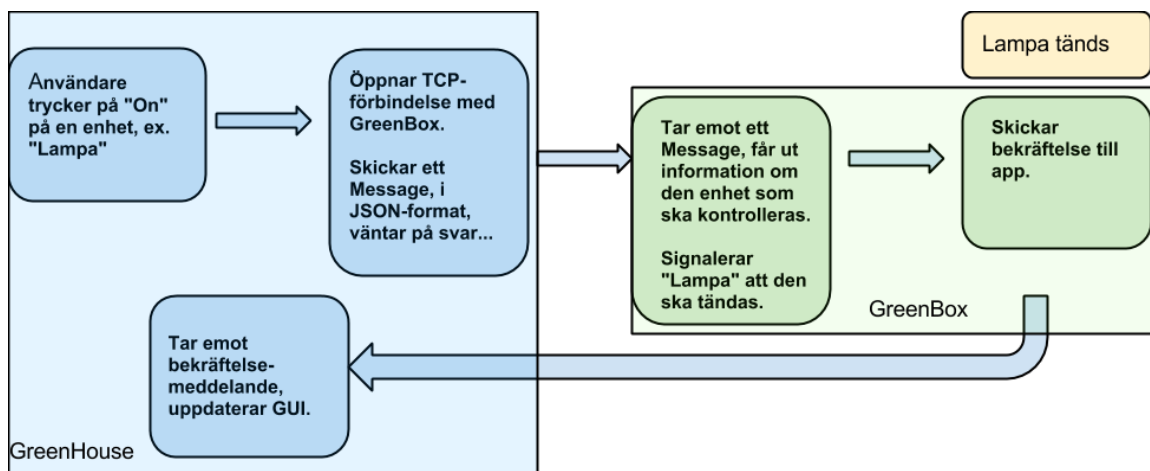
Fjärrkontrollen är det första som användaren möts av varje gång applikationen startas. Inte för att det är den viktigaste funktionen, utan för att det är den som man antagligen vill komma åt snabbast. Om man snabbt vill stänga av/sätta på en apparat i hemmet ska man bara behöva starta applikationen och klicka på rätt av/på knapp, utan att behöva gå igenom några menyer.



| REMOTE | SENSORS | RULES | CALENDAR |
|------------------|---------|-------|----------|
| Kökslampa | | On | Off |
| Element | | On | Off |
| PC | | On | Off |
| TV | | On | Off |
| Lampa Vardagsrum | | On | Off |
| Lampa Hall | | On | Off |
| Värme Stuga | | On | Off |

Att använda den är enkelt, man lägger till en enhet genom att trycka på symbolen längst upp i högra hörnet i Figur 10. Man kommer då till en meny där man anger namn, ID och typ på enheten. Efter att man har lagt till den visas den i listan. Väl där kan man trycka på On- och Off-knapparna för att signalera strömbrytarna att sluta resp. bryta strömmen. När man har tryckt på en knapp skickas ett meddelande till GreenBox, och applikationen väntar på ett bekräftelsemeddelande innan man kan trycka på knappen igen. För att redigera eller ta bort en enhet håller man fingret på den, och får då upp en meny med alternativ.

Figur 10 visar hur innehållet i tabben "Remote" ser ut.



Figur 11 beskriver händelseförloppet när användaren vill tända en lampa

4.2.4 Översikt av sensorer

Under tabben "Sensors" hittar man en lista över olika sensorer. Dessa visar det senaste uppmätta värdet hos de sensorkretsar som nämnts tidigare. Med hjälp av sensorerna får man information som kan ge användaren ökad kontroll över sin energiförbrukning, och gör det lättare att automatisera användandet av elektronik i hemmet.



| REMOTE | SENSORS | RULES | CALENDAR |
|----------------|---------|----------|----------|
| Rumstemperatur | | -9999 °C | |
| Utomhus | | 0 °C | |
| Stuga | | 0 °C | |
| Spis | | 0 °C | |

Figur 12 visar hur innehållet i "Sensors" ser ut. Värdena är bara tokenvärden.

För att uppdatera värdet på en enskild sensor trycker man på den i listan, för att uppdatera alla sensorer trycker man på symbolen uppe i högra hörnet, till vänster om plustecknet i Figur 12. Precis som i "Remote"-tabben kan man lägga till sensorer genom att trycka på plustecknet längst upp till höger. Man får då upp alternativ för att namnge sensorn, samt ange vilken enhet värdet visas i. För att redigera och/eller ta bort en sensor håller man fingret över den.

4.2.5 Regler (koncept)

Följande funktionalitet som beskriv har inte implementerats klart, utan för nuvarande finns bara backend-delen färdig på Greenbox. Det som visas här är alltså konceptet för hur användargränssnittet är planerat att se ut.

Detta är funktionen som skiljer GreenHouse från andra liknande applikationer. Här kan man kombinera alla komponenter i systemet för att automatisera användandet efter eget behov. Med hjälp av sensorkretsarna (vidare förklarade i 4.3), strömbrytarna, schemaläggningen i GreenBox samt de olika sensorerna i en smartphone kan man ställa upp regler baserade på logiska uttryck som ska hjälpa användaren att sänka sin energiförbrukning.

| REMOTE | SENSORS | RULES | CALENDAR |
|-----------------------------------|---------|---|----------|
| När jag lämnar hemmet | | PC - AV TV - AV Lampa Kök - AV Spis - AV | |
| När jag är på väg hem från jobbet | | Lampa Kök - PÅ Lampa Garage - PÅ Värme - PÅ | |
| När jag är på semester | | Allt - AV | |
| Rumstemperatur över 25 °C | | Värme - AV | |
| | | | |

Figur 13 (konceptbild) visar hur "Rules" skulle kunna se ut.

När man ska lägga till en ny regel får man upp en ruta (Figur 14) där man kan fylla i tre fält; *Description*, som motsvarar den beskrivning som visas i Figur 13, *Affected Devices* där man kan välja mellan enheter som redan har lagts till i tidigare menyer (se 4.2.3 Fjärrkontroll), samt *Conditions*. I Figur 14 är den logiska satsen "Om jag inte är hemma och om klockan är 08:30 så ska spisen och gruppen Lampor stängas av." angiven. Här kan

I Figur 13 ser man hur en översikt av reglerna skulle kunna se ut. Varje regel har en kort förklarande beskrivning på den vänstra delen av skärmen som användaren själv har angett. Till höger ser man vilka enheter som påverkas när det logiska uttryck som beskriver regeln är uppfyllt. Som vanligt kan man hålla fingret på en regel för att redigera/ta bort den, och trycka på plustecknet uppe till höger för att lägga till en ny.

För att se vilka villkor som ska uppfyllas för varje enskild regel kan man trycka på informationsfältet ("i") till vänster.

Applikationen går igenom reglerna med jämna mellanrum för att kontrollera vilka villkor som är uppfyllda, och signalerar vid behov de strömbrytare som ska påverkas genom GreenBox.

GreenHouse

Create new rule...

Description På väg till jobbet

Affected devices Spis - OFF
Lampor - OFF

Conditions My Location != HOME
AND
TIME == 08:30

Add

Figur 14 (konceptbild) visar hur det skulle kunna se ut när man lägger till en ny regel.

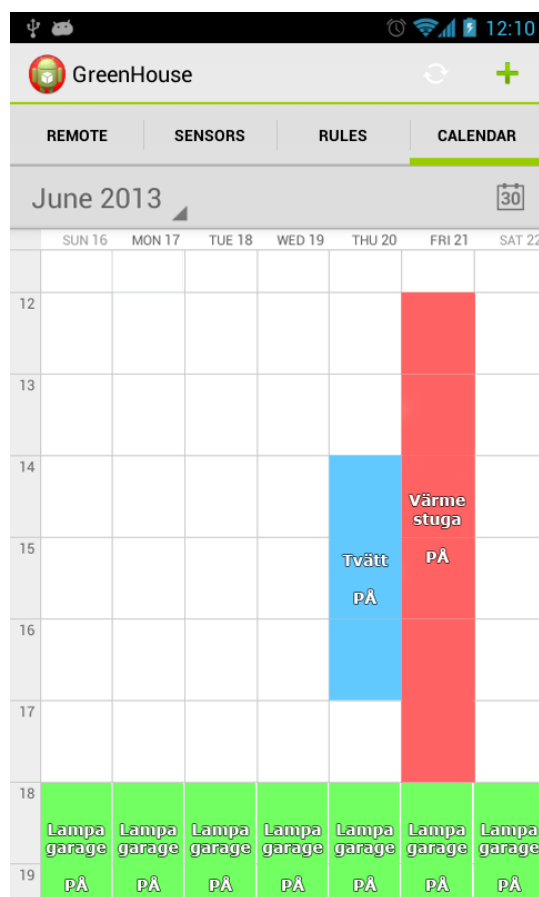
HOME exempelvis vara definierat som koordinaterna där användarens hus finns, hemadress, eller till och med hemnätverk. För att kunna sätta upp villkor för regeln krävs en viss känsla för satslogik. Då man inte kan vänta sig att alla användare har det måste gränssnittet designas på ett sätt som gör det enkelt även för mindre avancerade användare att sätta upp regler. Detta skulle kunna lösas genom att göra en steg-för-steg guide där användaren får ställa in grundinställningar som "Hem och arbetsplats", "Tider på dygnet då man är hemma/jobbar", mm, tillsammans med förinställda profiler som sätter upp regler åt en.

4.2.6 Kalender (koncept)

Följande funktionalitet som beskriv har inte implementerats klart, utan för nuvarande finns bara backend-delen färdig på Greenbox. Det som visas här är alltså konceptet för hur användargränssnittet är planerat att se ut.

Kalendern är ett simpelt verktyg för att schemalägga tidsbaserade händelser. Man anger en start- och sluttid, vilka tillstånd de påverkade enheterna ska vara i under tiden, samt huruvida händelsen ska upprepas. I Figur 15 har användaren till bland annat angett att lampan utanför garaget ska tändas varje dag kl. 18:00, och släckas vid ett senare tillfälle. Händelsen upprepas varje dag.

Kalendern är egentligen en klassisk timer, med ett mer dynamiskt och användarvänligt gränssnitt.



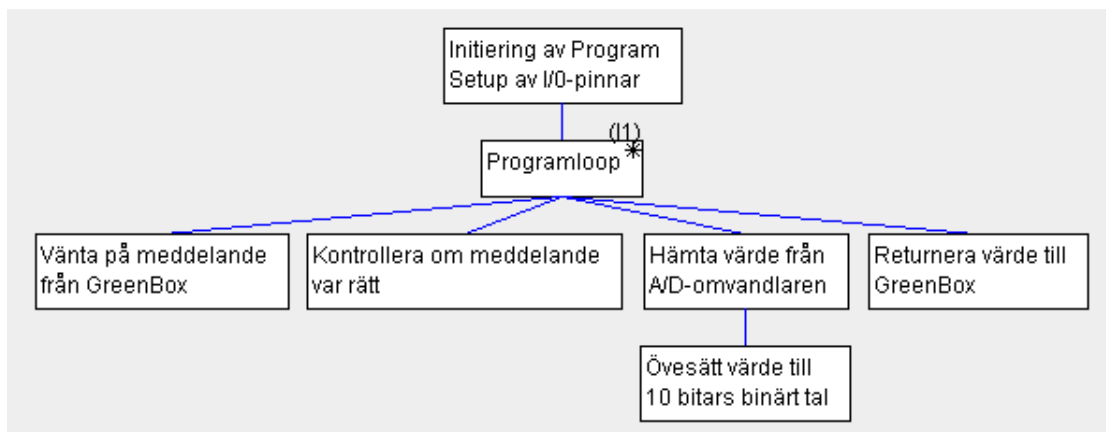
Figur 15 (konceptbild) visar hur en kalender (schema) skulle kunna se ut.

4.3 Sensor

De krav som ställts för sensorn är att den ska vara energisnål och billig, samt att den ska ha möjlighet att enkelt byta mellan trådlös och trådbunden kommunikation med GreenBox. Kommunikationen beslutades att ske genom seriell kommunikation genom sladd. Detta för att visa på ett koncept för sensorer istället för att lägga ner för mycket tid på att ta fram ett system för trådlös kommunikation mellan flera sensorer och GreenBox.

4.3.1 Temperatur

Den sensor som implementerades är en temperaturgivare som består av en NTC-resistans MCP9700a som är kopplad till en PIC16f690. Koden till PIC16f690 är skriven i C och kompileras för 8 bitars-processorn med kompilatorn Knudsen CC5X (39).



Programmet fungerar genom att vänta på att få ett meddelande från Raspberry Pi:n, när ett meddelande är mottaget som är ämnat för just denna sensorn så påbörjas temperaturmätningen.

NTC-resistorn ger ett analogt spänningsvärde som direkt proportionell mot temperaturen. PIC:en läser då av detta värde och omvandlar det till ett digitalt värde. Omvandlingen sker med 10 bitar analog till digital konvertering vilket innebär att det analoga värdet blir till ett 10 bitars digitalt värde, detta ger en noggrannhet på 5 mV. MCP9700a ger vid temperaturerna 0-70 C en mätnoggrannhet på +/- 2 C och därför tas medianen på 10 mätningar.

Detta värde skickas sedan tillbaka. Kommunikationen sker seriellt m.h.a UART.

5. Slutsatser

Ett komplett system för hemautomation som är användarvänligt och prisvärt, samtidigt som det utan ombyggnationer ska kunna installeras i ett hushåll, har många lösningar och många former. Det system framtagits i detta arbete har, förutom att vara ett exempel på implementering i sig, också avsikten att framhäva de olika problem som måste identifieras och lösas innan ett storskaligt projekt åtas. Arbetet kan därför ses som en prototyp mer än en kommersiell produkt och den har konstruerats med tid och begränsade resurser i åtanke.

Många av de problem som har identifierats och som måste lösas är relaterade till kommunikationen mellan GreenBox och fjärrströmbrytarna:

Tvåvägskommunikation - För att kunna bygga ett säkert och återkopplande system där användaren kan vara säker på att den information som ges faktiskt stämmer, krävs att en fjärrströmbrytare kan svara och bekräfta den har bytt tillstånd.

Räckvidd - För att kunna bygga ett system som har möjlighet att fungera även i större bostäder, så behövs antingen kommunikation som klarar långa avstånd eller någon form av nodnät där signaler/meddelanden skickas vidare genom nätet och på så sätt når noder som är utom direkt räckvidd.

Pris - För att vara attraktivt för så bred mängd personer som möjligt krävs att priset för ett sådant här system inte rusar iväg bara för att man går över till ett system med tvåvägskommunikation och bra räckvidd. Problemet med många av de system som existerar idag är att utvecklings- och produktionskostnaderna för avancerade kommunikationsteknik höjer priset till en nivå som är långt över vad de flesta är villiga att betala.

För att svara på frågan: ”Går det att utveckla ett system för hemautomation som är prisvärt, effektivt och användarvänligt?”. Så kan vi dra slutsatsen att om ett system likt det i detta arbete utvecklat fungerar i både teori och praktik, och som har påvisats både med modellering och ett faktiskt installerat system i en av författarnas lägenhet. Så är det med all sannolikhet möjligt att utveckla en produkt som uppfyller kraven. Genom att utnyttja existerande teknik för bl.a. systemet med fjärrströmbrytare och plattformen som mjukvaran körs på har det varit möjligt att uppnå de krav som ställdes i arbetet.

Om arbetet skulle fortsättas och en eventuell uppdatering av kommunikationen mellan fjärrströmbrytare och GreenBox skulle introduceras så är priset en sak som skulle behövas ha

i åtanke. Kostnaderna för utveckling av mjukvaran, hårdvaran, och all testning av systemet samt produktions- och distributionskostnader behöver tas med i beräkningen. Dock så anser vi att om två studenter som delat på 15 hp för att utvecklat detta lyckats med ett faktiskt funktionellt system så borde eventuella utvecklingskostnader kunna hållas nere.

En annan fråga vi ställde oss i problemformuleringen var "Hur kan man göra det mer attraktivt för privatpersoner att använda hemautomation för att minska sin miljöpåverkan?". Vi tror att man göra det bland annat genom att uppmärksamma människor om vilka följder ett hemautomationssystem har på både miljön och ekonomin. I kapitel 3.7.1 *Modellering av energibesparingar* visades ett exempel, som förvisso byggde på grova förenklingar och antaganden, där man sänkte den årliga elkostnaden och koldioxidutsläppen. Dessa besparingar gjordes på ett sätt som inte påverkade användarens vardagsliv över huvud taget. Om man applicerade systemet på en större del av huset, och framför allt i flera hushåll, tror vi att man skulle kunna göra en stor positiv inverkan på miljön, i varje fall i förhållande till hur stor del av ett lands totala miljöpåverkan orsakas av bostadssektorn.

En ytterligare frågeställning som är viktig är: Är ett system likt detta realistiskt? Är hemautomation för allmänheten något som är värt att utveckla? - I takt med att fler och fler personer har smartphones och den allmänna kunskapen kring datorer och elektronik växer, bör det kunna ses som naturligt att också personers hem är mer integrerade med elektronik. Detta tillsammans med att det idag finns en drivande kraft hos individer att faktiskt värna om miljön, gör att folk har en vilja att ta till sig denna typen av teknik. Därför är hemautomation nästa steg och helt i tiden.

6. Framtida förbättringar

Efter att projektet slutförts finns det en del förbättringar och ändringar som behöver göras i systemet för att det ska vara användbart och för att uppfylla dess syfte. Problem som måste lösas är de som uppstått under utvecklingen, och de som planerades men inte hann lösas på grund av tids- och resursbrist.

Den punkten som projektets egna implementaion på hemautomationssystem visat vara den viktigaste är kommunikationen mellan GreenBox och fjärrströmbrytarna. För att kunna ha någon form av garanti krävs tvåvägskommunikation och denna borde ske med något form av nodnät-lösning så som ZigBee. Detta dels för att kunna skicka tillbaks att fjärrströmbrytaren faktiskt har bytt tillstånd och dels för att det skulle erbjuda möjligheten att skicka tillbaka mer data så som energianvändning till användaren.

För att nå så många personer som möjligt skulle även fler användargränssnitt behöva tas fram. Då helst i form av en webbläsarklient eller en applikation för fler operativsystem.

Även om Greenbox i dagsläget inte drar mer än några få watt så finns det enkla ändringar som hade kunnat sänka elkonsumtionen ytterligare. Hade man konfigurerat operativsystemet för att tillåta manipulering av processorns klockfrekvens så hade det varit möjligt att systematiskt sänka processorns klockfrekvens när GreenBox:en inte används vilket hade lett till minskad elkonsumtion.

Till den Android-applikation som utvecklats fanns det flera funktioner som inte hann implementeras. Bland annat ett sätt för användaren att definiera ett regelverk där dels mobilens gps och dels sensorer skulle kunna kontrollera vissa delar av systemet. Om t.ex. användaren kom inom 100 m av sin lägenhet skulle en regel kunna säga att dörrlampan skulle tändas. En annan funktion skulle bli att kunna ha en kalender, dels i applikationen och dels integrera med användarens standardkalender för schemaläggning av systemet. Dessa två funktioner blev implementerade på serversidan, alltså i själva GreenBox:en men hann aldrig få ett användargränssnitt.

7. Referenser

1. **Borgström, Håkan (Capio AB), Frank, Harry. Kasemo, Bengt. Lagerträd, Henrik (red.).** *Energi, möjligheter och dilemman.* Stockholm : Stockholm: Kungl. Ingenjörsvetenskapsakademien (IVA)., 2009. 978-91-7082-815-7.
2. Digitala Vetenskapliga Arkivet. [Online] Uppsala Universitet. [Citat: den 19 Maj 2013.] <http://diva-portal.org/smash/search.jsf>.
3. **joakim.** A Tech Blog. [Online] 2012. [Citat: den 19 Maj 2013.] <http://tech.jolowe.se/home-automation-rf-protocols/>.
4. RF-protokoll JULA Anslut. [Online] Elektronikforumet. [Citat: den 1 April 2013.] http://www.elektronikforumet.com/wiki/index.php?title=RF_Protokoll_-_JULA-Anslut.
5. NexaProtocol. [Online] Telldus. [Citat: den 19 Maj 2013.] http://svn.telldus.com/svn/rf_ctrl/nexa_2_6_driver/trunk/NexaProtocol.txt.
6. Android Developers. [Online] Google Inc. [Citat: den 19 05 2013.] <http://developer.android.com/index.html>.
7. **Gilfelt, Jeff.** Android Action Bar Style Generator. [Online] GitHub. [Citat: den 19 Maj 2013.] <http://jgilfelt.github.io/android-actionbarstylegenerator/>.
8. Eclipse Home. [Online] The Eclipse Foundation. [Citat: den 19 Maj 2013.] <http://eclipse.org/>.
9. Saros Project. [Online] Saros. [Citat: den 19 Maj 2013.] <http://www.saros-project.org/>.
10. ObjectAid Home. [Online] <http://www.objectaid.com/>.
11. Circuit Lab. [Online] CircuitLab Inc. <https://www.circuitlab.com/>.
12. **Bort, Julie.** Google: There Are 900 Million Android Devices Activated. *Business Insider*. den 15 Maj 2013, ss. <http://www.businessinsider.com/900-million-android-devices-in-2013-2013-5>.
13. iOS Developer Program. [Online] Apple Inc. [Citat: den 19 Maj 2013.] <https://developer.apple.com/programs/ios/>.

14. **McCracken, Harry.** Who's winning, iOS or Android? [Online] TIME, den 16 April 2013. [Citat: den 19 Maj 2013.] <http://techland.time.com/2013/04/16/ios-vs-android/>.
15. **Holenarsipur, Prashanth.** I'm OOK. You're OOK? [Online] den 8 April 2009. [Citat: den 19 Maj 2013.] <http://pdfserv.maximintegrated.com/en/an/AN4439.pdf>.
16. ZigBee Technology. [Online] ZigBee Alliance. [Citat: den 19 Maj 2013.] <http://www.zigbee.org/About/AboutTechnology/ZigBeeTechnology.aspx>.
17. Motion Sensor NCSZ-3041. [Online] NYCE Control Inc. [Citat: den 19 Maj 2013.] <http://nycecontrol.com/products/motion-sensor/>.
18. Nordic Semiconductor. [Online] Nordic Semiconductor. [Citat: den 31 Maj 2013.] <http://www.nordicsemi.com/eng>.
19. Raspberry Pi FAQ. [Online] Raspberry Pi Foundation. [Citat: den 19 Maj 2013.] <http://www.raspberrypi.org/faqs>.
20. What is BeagleBone? [Online] BeagleBoard.org. [Citat: den 19 Maj 2013.] <http://beagleboard.org/Products/BeagleBone>.
21. CubieBoard Home. [Online] CubieBoard. [Citat: den 19 Maj 2013.] <http://cubieboard.org/>.
22. ODROID-U2 Details. [Online] Hardkernel Co. [Citat: den 19 Maj 2013.] http://www.hardkernel.com/renewal_2011/products/prdt_info.php?g_code=G135235611947.
23. **Rasbian.** Home. *Rasbian.org*. [Online] [Citat: den 20 05 2013.] <http://www.raspbian.org/>.
24. PIC 16F690 Data Sheet. [Online] [Citat: den 20 Maj 2013.] <http://ww1.microchip.com/downloads/en/DeviceDoc/41262E.pdf>.
25. **Wikipedia.** Thermistor. *wikipedia.org*. [Online] [Citat: den 20 05 2013.] <http://en.wikipedia.org/wiki/Thermistor>.
26. **Microchip.** MCP9700A. *Microchip.com*. [Online] [Citat: den 20 05 2013.] <http://ww1.microchip.com/downloads/en/DeviceDoc/21942e.pdf>.
27. **Björklund, Anna.** Avfall. *Avfall*. u.o. : AG1815 Hållbar utveckling, 2013.
28. **Boverket.** *Boendets Miljöpåverkan*. u.o. : Boverket, 2011. 978-91-86827-93-9.

29. **Boveri), ABB (Asea Brown.** Smarta elnät. *http://www.abb.se*. [Online] 2013. [Citat: den 22 02 2013.] <http://www.abb.se/cawp/seabb361/b823cb445895db5fc12575a5003b1edb.aspx>.
30. **Augustsson, Tomas.** Smartare el-nät ljus idé. *svd.se*. [Online] den 14 06 2011. [Citat: den 22 02 2013.] http://www.svd.se/naringsliv/smartare-el-nat-ljus-ide_6043811.svd.
31. Klimatpåverkan och växthusgaser. [Online] Svensk Energi, 2012. [Citat: den 29 Maj 2013.] <http://www.svenskenergi.se/Elfakta/Miljo-och-klimat/Klimatpaverkan/>.
32. **Väljements, Erik och Yngvesson, Magnus.** Det Intelligent Huset - En studie i hemautomation. *DiVA*. [Online] den 10 Maj 2012. [Citat: den 28 Maj 2013.] <http://kth.diva-portal.org/smash/record.jsf?searchId=1&pid=diva2:552623>.
33. Insteon - Home. [Online] INSTEON, 2013. [Citat: den 20 Maj 2013.] <http://www.insteon.com/>.
34. TellStick Basics. [Online] Telldus Technologies. [Citat: den 19 Maj 2013.] <http://www.telldus.se/products/tellstick>.
35. **inder123, Joel leitch, limpbizkit.** gson. *code.google.com*. [Online] [Citat: den 20 05 2013.] <https://code.google.com/p/google-gson/>.
36. asciitable. *asciitable.com*. [Online] [Citat: den 20 05 2013.] <http://www.asciitable.com/>.
37. **Android.** design. *developer.android.com/design*. [Online] Google. [Citat: den 20 05 2013.] <http://developer.android.com/design/index.html>.
38. **Wharton, Jake.** ActionBarSherlock. [Online] [Citat: den 1 Maj 2013.] <http://actionbarsherlock.com/>.
39. **Knudsen, B.** CC5X. *bknd.com/cc5x*. [Online] [Citat: den 20 05 2013.] <http://www.bknd.com/cc5x/>.
40. Användarmanual för Nexa Gateway. [Online] [Citat: den 19 Maj 2013.] <http://www.nexa.se/res/pdf/Nexa-Gateway-manual-v1.9.pdf>.