CHALMERS

JunkSpy – Den Smarta Soptunnan

Ett system för övervakning av sopnivåer och ruttoptimering

Kandidatarbete inom Data- och informationsteknik

Anton Johansson
Fredrik Fröst
Johannes Blomquist
Amanda Strömsten
Gustav Sundin
Johannes Sjöberg

Chalmers tekniska högskola
Göteborgs universitet
Institutionen för Data- och Informationsteknik
Göteborg, Sverige, Juni 2013

JunkSpy – The Smart Trash Can
Fill level monitoring and route optimization system

A. Johansson, A. Strömsten, F. Fröst, G. Sundin, J. Blomquist, J. Sjöberg.

Examiner: A. Linde

Chalmers University of Technology
University of Gothenburg
Department of Computer Science and Engineering
SE-412 96 Göteborg
Sweden
Telephone + 46 (0)31-772 1000

Department of Computer Science and Engineering
Göteborg, Sweden June 2013

Sammanfattning

I dagens samhälle är sophantering en miljömässig utmaning. Dels för att själva sophanteringen är energikrävande, men också för att det med dagens sophanteringssystem är svårt att bedöma på förhand om en soptunna behöver tömmas. Personen som ska tömma en specifik soptunna måste först uppsöka soptunnan för att avgöra om den behöver tömmas eller ej.

Projektets huvudmål är att utveckla ett system som kan effektivisera sophämtningen genom att göra soptunnor smarta. Detta genomförs med hjälp av en kompakt, integrerad enhet som mäter mängden skräp i soptunnor och som enkelt kan monteras på en soptunna. Hårdvaran kommunicerar med en webbapplikation via ZigBee där användaren i realtid ser vilka soptunnor som behöver tömmas från en dator eller smartphone. Vidare kan systemet beräkna nya, effektiva soptömningsrutter och föreslå dessa för användaren.

I rapporten återfinns ett teoriavsnitt av de olika komponenterna som används: en Arduino-plattform, sköldar för nätverkskommunikation och en ultraljudssensor. Mjukvaran som består av en webbapplikation är skriven i HTML5 och har en databas som hanteras av MySQL. Kartorna som syns i applikationen kommer från Google Maps.

Läsaren får även följa hela utvecklingsprocessen av hårdvaran som mäter sopmängden, tillvägagångssättet och motiveringar av olika komponentval. Utvecklingen sker i en iterativ process. Avslutningsvis redovisas det grafiska användargränssnitt som tagits fram genom förklarande bilder tagna direkt från applikationen.

Abstract

In today's society, garbage disposal is an environmental challenge. Partly because the waste disposal system is energy consuming, but also because it is difficult to estimate in advance when a garbage bin needs emptying. The person responsible for emptying a specific garbage bin therefore needs to visit it to tell if it is full or not.

The main goal of the project have been to design a system that can make garbage collection more effective by making garbage bins smart. This is done with a compact, integrated hardware unit that can easily be installed into a garbage bin and that continuously measures the amount of garbage in the bin. The hardware communicates with a web application via ZigBee where the user in real time can see which garbage bins needs to be emptied from a computer or a smartphone. Further, the system is able to calculate new, efficient garbage collection routes, also in real time.

Found in the report is a description of the different components used: an Arduino, shields for network communication and an ultrasonic distance sensor. The software is written in HTML5 and uses a MySQL database. The maps in the application comes from Google Maps.

The reader may also follow the entire development of the hardware that measures the bin's waste level, complete with motivations for different choices of components. The development is done in an iterative process. Finally, the graphical user interface is demonstrated through screenshots of the application.

Författarnas tack

Projektgruppen vill tacka handledare Lennart Hansson för hans hjälp och konsultation under projektets gång och kandidatansvarig Arne Linde för rådgivning och hans stöd vid införskaffning av hårdvara. Stort tack till Sia Sheikholeslamzadeh på Svensk Markservice och Petter Backlund på räddningstjänsten i Storgöteborg för att de ställt upp på att bli intervjuade. Stort tack även till Love Yregård för hans insikter och råd kring utvecklingen av ruttberäkningsalgoritmen och korrektur samt Lars Agnemar för den ursprungliga idén och inspiration.

Innehåll

1 Inledning **1**
- 1.1 Bakgrund . . . 1
- 1.2 Syfte . . . 1
- 1.3 Problembeskrivning . . . 1
- 1.4 Projektupplägg . . . 1
- 1.5 Avgränsningar . . . 2
- 1.6 Metod . . . 2

2 Förstudier **3**
- 2.1 Liknande arbeten . . . 3
- 2.2 Kravinsamling . . . 3
  - 2.2.1 Intervju 1: Svensk Markservice . . . 4
  - 2.2.2 Intervju 2: Räddningstjänsten . . . 4
- 2.3 Krav- och testsammanfattning . . . 5

3 Teori **6**
- 3.1 Hårdvara . . . 6
  - 3.1.1 Arduino . . . 6
  - 3.1.2 Sköldar . . . 6
  - 3.1.3 Ethernet Shield . . . 6
  - 3.1.4 Wireless Proto Shield . . . 7
  - 3.1.5 XBee S2 . . . 7
  - 3.1.6 Parallax Ping))) avståndssensor . . . 7
  - 3.1.7 Sharp GP2Y0D02YK avståndssensor . . . 8
  - 3.1.8 Energisparläge . . . 8
  - 3.1.9 Strömförsörjning . . . 8
  - 3.1.10 Programmering av mikrokontrollern . . . 9
  - 3.1.11 Inkapsling . . . 10
- 3.2 Mjukvara . . . 10
  - 3.2.1 Webbapplikationen . . . 10
  - 3.2.2 Responsiv webbdesign . . . 10
  - 3.2.3 HTML5 . . . 11
  - 3.2.4 Lagring av mätdata . . . 11
  - 3.2.5 MySQL . . . 11
  - 3.2.6 Google Maps . . . 12
- 3.3 Kommunikation . . . 12
  - 3.3.1 ZigBee . . . 12
- 3.4 Säkerhet . . . 13
- 3.5 Att beräkna den optimala soprutten . . . 13
- 3.6 Miljöaspekter . . . 15

4 Genomförande **17**
- 4.1 Iteration 0 - Sensor och grundläggande kommunikation . . . 17
  - 4.1.1 Kravlista för prototyp 0 . . . 17
  - 4.1.2 Tillvägagångssätt . . . 17
- 4.2 Iteration 1 - Förfining och grundläggande mjukvara . . . 17
  - 4.2.1 Kravlista för prototyp 1 . . . 17
  - 4.2.2 Tillvägagångssätt . . . 17
- 4.3 Iteration 2 - Kommunikation med ZigBee . . . 18
  - 4.3.1 Kravlista för prototyp 2 . . . 18

4.3.2 Brygga . . . 18
4.3.3 Tillvägagångssätt . . . 18
4.4 Iteration 3 - Fullständigt energisparläge samt inkapsling . . . 18
4.4.1 Kravlista för prototyp 3 . . . 19
4.4.2 Tillvägagångssätt . . . 19

5 Resultat **20**
5.1 Hårdvara . . . 20
5.1.1 Spion . . . 20
5.1.2 Brygga . . . 21
5.2 Kommunikation . . . 21
5.3 Mjukvara . . . 22

6 Diskussion **23**
6.1 Utvärdering av metod . . . 23
6.2 Utvärdering av systemet . . . 23
6.2.1 Utveckla hårdvara kapabel att bedöma sopnivå . . . 23
6.2.2 Utveckla energieffektiv hårdvara . . . 24
6.2.3 Få hårdvaran att skicka mätdata . . . 25
6.2.4 Lagra sopnivåer och avgöra soptunnors sopstatus . . . 25
6.2.5 Beräkna effektiva sophämtningsrutter . . . 25
6.2.6 Presentera data för slutanvändaren på ett tydligt sätt . . . 26
6.3 Framtida utveckling . . . 26

7 Slutsats **28**

Bilagor

A Krav- och testspecifikation **I**

B Skiss av användargränssnitt för webbapplikation **VI**

C Databasstruktur **IX**

D Detaljerad hårdvara **X**

E Applikationsförklaring **XI**

Ordlista

Nedan listas ord av mer teknisk karaktär, förkortningar av ämnesspecifika termer och andra ordförklaringar.

API Application Programming Interface. En beskrivning av funktionaliteten hos ett mjukvarubibliotek.

Arduino Programmerbar prototypplattform för styrning av elektroniska komponenter.

Back-end Ett systems bakomliggande mjukvara, som tillhandahåller all funktionalitet som visas upp i användargränssnittet.

Brygga Länken mellan **Spionerna** i det interna ZigBee-nätverket och Internet.

C/C++ Hårdvarunära programspråksfamilj. Arduino programmeras i dessa språk.

CPU Central Processing Unit. Digital beräkningskrets som utför instruktioner enligt en instruktionsuppsättning.

CSS Cascading Style Sheets. Stilmall för att anpassa presentation av HTML-element i en webbläsare.

Energisparläge (eng. **Sleep mode**) Ett sätt för en elektronisk krets att förbruka mindre ström genom att stänga av delar av sin funktionalitet.

Ethernet En IEEE-standard för kommunikation mellan datorer via kabel i ett lokalt nätverk

Givare Se **sensor**.

GPRS General Packet Radio Services. En standard av protokoll för datanätverkstjänster i GSM-nät.

GSM Global System for Mobile Communication. En standard av protokoll för mobilkommunikation.

Hash *-ning* En datorfunktion som associerar ett värde av variabel längd (till exempel ett namn) till ett värde av fast längd (till exempel ett heltal). Kan användas i kryptografiska sammanhang.

HTML Hypertext Markup Language. Ett märkspråk som ger struktur till hemsidor och dokument.

IEEE 802.15.4 En standard för trådlös kommunikation vilken ZigBee bygger på.

Interrupt Sv. Avbrott. En signal som tvingar processorn att avbryta nuvarande läge.

IR-sensor Givare som med hjälp av att mäta reflektionen av det infraröda ljuset den sänder kan avgöra avståndet till närmsta föremål.

I/O Fr. eng. **Input/Output**. Ungefär Indata/Utdata och syftar på maskinnära kommunikation mellan maskin och annan part.

JavaScript Ett skriptspråk som körs på klientsidan i webbsammanhang.

JunkSpy Systemets namn.

Kontaktdon Även vardagligt kallat kontakt. Den utrustning vilken sammankopplar elektriska ledare.

Mesh-nätverk En nätverksstruktur där alla noder i nätverket kan vidarebefordra meddelanden till varandra. Ett meddelande kan alltså ta olika vägar genom nätverket.

MySQL Structured Query Language. En databashanterare.

Nod En punkt i ett nätverk som kan sända och/eller ta emot data.

PHP Hypertext Preprocessor. Ett skriptspråk som körs på serversidan i webbsammanhang.

RJ45 En modularkontakt som används vid datakommunikation över Ethernet.

Sensor En apparat eller anläggning som insamlar, konverterar och i vissa fall distribuerar någon form av signal eller stimuli eller data.

Shield Ett kretskort som monteras på en Arduino för att ge utökad funktionalitet av något slag.

Sköld Se **shield**.

Sleep mode Se **energisparläge**.

Spion En Arduino med tillhörande **sensorer** som sitter monterad i en soptunna och regelbundet skickar mätdata till en central databas. Varje Spion utgör en nod i ZigBee-nätverket.

TCP/IP Transmission Control Protocol/Internet Protocol. Är en samling av protokoll för IP-nätverk.

Ultraljudssensor Givare som mäter avstånd till närmsta föremål genom att skicka ut en ljudvåg och räkna tiden tills ekot från det föremålet.

WLAN Wireless Local Area Network. En standard för trådlös kommunikation, oftast mellan router och trådlösa enheter som laptops, skrivare, mobiltelefoner etc. i ett hem- eller kontorsnätverk. WLAN bygger på standarden IEEE 802.11 -familjen

ZigBee En standard för trådlös styrning och övervakning av utrustning i bland annat hem, fastigheter och industrier. ZigBee bygger på radiostandarden IEEE 802.15.4.

1 Inledning

I första kapitlet presenteras bakgrunden till projektet som handlar om att övervaka sopnivåer och ruttoptimering, dess syfte samt en formulering av problemet. Därefter beskrivs projektets upplägg som består av tre olika delområden - hårdvara, kommunikation och mjukvara. Slutligen beskrivs den arbetsmetod som har tillämpas i projektet.

1.1 Bakgrund

I dagens samhälle är sophantering en miljömässig utmaning. Dels för att själva sophanteringen är energikrävande, men också för att det med dagens sophanteringssystem inte går att bedöma på förhand om en soptunna behöver tömmas. Personen som ska tömma en specifik soptunna behöver alltså uppsöka soptunnan för att avgöra om den behöver tömmas eller ej.

Smarta system är ett aktuellt ämne och diskuteras inom alla typer av verksamheter vare sig det gäller att reglera energiförbrukningen i hemmet, e-böcker eller självgående bilar. Smarta system går ut på att byta ut analoga system mot en digital motsvarighet som i större utsträckning är självskötande och underlättar användarens vardag.

Soptunnan är ett bra exempel på en analog produkt som går att förbättra med hjälp av dagens teknik. Genom att digitalisera soptunnan och på så sätt göra den smart, öppnas möjligheterna att skapa ett mer miljömedvetet och effektivt sophanteringssystem. Med en smart soptunna som kan hjälpa till med planering, kalkylering och notifiering kan miljövårdsarbete förenklas och översikten för verksamheten tydliggöras.

1.2 Syfte

Syftet med projektet är att skapa ett pålitligt system – benämnt JunkSpy – som kan hjälpa till att effektivisera sophämtningen. Målsättningen för projektet är att skapa en prototyp vid namn Spionen som ska mäta mängden sopor, skicka mätdata till en databas och som slutligen presenterar mätningarna för användaren. Ur en miljöaspekt är JunkSpy ett verktyg som skulle kunna bidra till ett mer energieffektivt sophanteringssystem eftersom onödiga turer till soptunnor som inte behöver tömmas kan undvikas.

1.3 Problembeskrivning

Projektet har tre avgränsade delar - hårdvara, mjukvara och kommunikation. Hårdvarumässigt ligger problemet i att undersöka vilken eller vilka typer av sensorer som ger pålitliga mätvärden och är lämpade för att monteras i en soptunna. Vidare gäller att på ett realistiskt sätt kunna modellera mängden sopor i en soptunna. Ur en miljöaspekt är målet att denna hårdvara ska vara så energisnål som möjligt. Mjukvarudelen består av att insamlad mätdata ska kunna användas för att beräkna effektiva sophämtningsrutter. Dessa ska sedan presenteras för användaren på ett användarvänligt sätt. Slutligen ska hårdvaran kunna skicka mätvärden till en databas med jämna tidsintervall, vilket behandlas i ett kommunikationsavsnitt.

1.4 Projektupplägg

Huvudproblemen i projektet är följande:

- Utveckla hårdvara kapabel att bedöma sopnivå
- Utveckla energieffektiv hårdvara
- Utveckla hårdvara kapabel att skicka mätdata
- Lagra sopnivåer och avgöra soptunnors sopstatus
- Beräkna effektiva sophämtningsrutter
- Presentera data för slutanvändaren på ett tydligt sätt

1.5 Avgränsningar

Eftersom projektets syfte inte är att utveckla en färdig kommersiell produkt kommer endast en prototyp tas fram. På grund av den begränsade tiden syftar projektet till att i så stor utsträckning som möjligt utnyttja färdiga protokoll, algoritmer, komponenter och hårdvaruenheter för att utveckla en så användbar prototyp som möjligt. Därför kommer det endast krav på att sensorn ska gå att placera i soptunnan av Göteborgsmodellen, en bild av soptunnan återfinns i bilaga A. Vidare ska sensorn som utför mätningen på soporna ska klara av lättare stötar. Däremot ställs inga krav på att prototypen ska klara det svenska utomhusklimatet. JunkSpy antas också vara utformad för en stadsmiljö där soptunnorna är tätt utplacerade. Detta kommer sammantaget att spela in i valet på kommunikationsform som väljs mellan enheterna i systemet.

1.6 Metod

Arbetet kommer i ett tidigt stadium innefatta identifikation av befintliga komponenter, färdiga system, standardiserade protokoll samt genomförandet av intervjuer. Tanken med detta är att på så sätt kunna testa olika lösningars lämplighet för den prototyp som ska modelleras utefter problembeskrivningen. Initialt kräver detta en undersökning för vilka komponenter eller system som finns att tillhandahålla, införskaffa dessa och genom experiment anpassa dessa efter våra kravspecifikationer, parallellt med fördjupad inläsning.

Projektet har som nämnts ovan delats upp i tre avgränsade områden, nämligen:

1. Hårdvara i form av en prototyp med mikrokontroller, sensorer samt dess implementering (kallad Spion).
2. Mjukvara i form av en webbapplikation.
3. Kommunikationen mellan ovanstående hårdvara och mjukvara.

Arbetet för att ta fram prototyper kommer att ske i form av iterationer, där varje ny prototyp innehåller en uppsättning av mer exakta krav än den tidigare versionen. Tanken med det iterativa arbetssättet handlar om att gruppen är övertygad om att ett bra resultat uppnås genom att löpa igenom arbetsprocessen i flera etapper. På så vis är det lättare att i ett tidigt stadie upptäcka onödiga fel och istället lägga fokus på att förbättra och vidareutveckla Spion-prototypen.

Stor vikt kommer att läggas på att ta fram olika utkast av hårdvaru- och mjukvaruprototyper rent praktiskt och låta utvärderingen efter varje iteration styra utvecklingen av nästkommande prototyp. Vid inköp av hårdvara och utvärdering av resultat bör inläsning ske för att besvara frågor som kan uteslutas rent teoretiskt.

2 Förstudier

Detta kapitel inleds med att behandla två liknande projekt; BigBelly och Smartbin. Vidare har två intervjuer genomförts under kravinsamlingsarbetet, den första med en potentiell användare och intervju nummer två med en person från räddningstjänsten för att ta del om information kring bränder i soptunnor. Slutligen återfinns en sammanfattning av projektets krav- och testspecifikation.

2.1 Liknande arbeten

Under projektets start har en litteraturstudie genomförts med syftet att öka kunskapen kring hur andra löst liknande problem. Fokus under studien har varit att utöka kunskapen kring användbara komponenter som passar projektets syfte, systemets framställning mot användaren samt på vilket sätt dessa produkter effektiviserar sophämtningen.

BigBelly[23] och Smartbin[22] är två produkter som liknar JunkSpy. Syftet för båda projekten är att effektivisera sophämtningen så att en soptunna alltid ska vara nästintill full när den töms men aldrig överfylld. Detta uppnås i båda produkterna genom att använda sig av ultraljudssensorer för mätning och en webbapplikation för att presentera data för användaren. Olika typer av information som kartvyer, varningar, rapporter och information från avfalls- och återvinningsstationer tillhandahålls och visas i webbapplikationen.

BigBelly har utvecklad två olika helhetslösningar där kunden får köpa en färdig soptunna utrustad med sensorer och kommunikationsutrustning. Den ena lösningen har även utrustats med en kompressor som ska pressa ihop soporna för att ytterligare minska tömningsbehovet. BigBelly kommunicerar med GPRS (*General Packet Radio Services*), en teknik som gör det möjligt att placera en soptunna var som helst där det finns mottagning för mobiltelefoner. Då BigBelly drivs av batterier och laddas upp med hjälp av solceller behöver inte heller någon el kopplas in till soptunnan. En nackdel kan dock vara att den endast har batterikapacitet för att klara upp till 72 timmar utan direkt solljus, på grund av hög elförbrukning. Det kan bli ett problem i vissa delar av världen som inte har tillräckligt mycket sol.

Smartbins lösning är mer lik JunkSpy, med en kompakt enhet som monteras i locket på valfri soptunna. Smartbin använder sig av en ultraljudssensor för att mäta sopnivån och den skickar mätdata över det mobila nätverket, precis som BigBelly. Smartbin drivs också med batterier och är därför enkel att installera och underhålla då batterierna uppges hålla i minst fem år.

I Malmö finns det 3300 återvinningsbehållare som utrustats med IR-sensorer (*infraröd-sensor*) för att kunna mäta mängden sopor i dem [10]. Syftet är dels för att kunna kontrollera hur tömningen sköts av de ansvariga entreprenörföretagen och dels för att entreprenörerna själva ska kunna använda sig av realtidsinformation från soptunnorna för att effektivisera sitt arbete. En studie från Lunds universitet visar att ett dynamiskt sophämtningsschema kan minska kostnaderna med 10-20% jämfört med att ha ett statiskt schema [10].

Sammanfattningsvis pekar allt detta på att det är fullt möjligt att konstruera ett system som kontinuerligt mäter mängden sopor i ett antal soptunnor, och att det dessutom verkar kunna göra nytta i praktiken.

2.2 Kravinsamling

I ett projekt likt detta är det viktigt att i ett tidigt stadium definiera projektets syfte. Detta görs lämpligen genom att ställa krav kring hur sophämtningen ska förenklas, vad som krävs rent tekniskt

för att uppnå en effektivisering av dagens sophämtning samt hur den slutliga webbapplikationen ska upplevas av användaren.

I kravinsamlingsarbetet har fokus legat på brainstorming, både individuellt och i grupp, och observationer av liknande tekniker genom litteraturstudier. Då det också är viktigt att knyta an krav från olika synvinklar har två stycken intervjuer genomförts med syfte att förankra projektet till verkligheten och på så vis utveckla den produkt användaren efterfrågar. Den första intervjun hölls med en potentiell användare, Sia Sheikholeslamzadeh, som (bland annat) arbetar med att tömma soptunnor på Chalmers. Den andra intervjun hölls med Petter Backlund på räddningstjänsten där fokus låg på att undersöka nyttan av att inkludera en branddetektor i Spionen.

2.2.1 Intervju 1: Svensk Markservice

Sia Sheikholeslamzadeh[1] är anställd på Svensk Markservice. Han är en potentiell brukare av systemet och ansvarar bland annat för tömningen av de drygt åttio soptunnor som är stationerade på Chalmersområdet. Varje morgon åker han en runda runt området och tömmer de soptunnor som behöver tömmas. Är en soptunna inte full låter han den vara, vilket innebär att alla soptunnor inte behöver tömmas varje dag. Andra soptunnor som används mer flitigt kan behöva tömmas 2-3 gånger per dag.

Eftersom Sia har jobbat på Chalmers i fyra år har han fått in rutinerna väl och vet hur ofta och när varje soptunna behöver tömmas. Av den anledningen anser Sia att han personligen inte skulle dra stor nytta av JunkSpy-systemet, däremot kan han tänka sig andra sammanhang där systemet skulle komma till nytta. Ett exempel på detta kan vara när en nyanställd ska lära sig rutinerna. Enligt Sia brukar detta ta omkring två veckor, men med en applikation som i realtid visar vilka soptunnor som behöver tömmas tror han inlärningstiden kan kortas ner. För en organisation som ansvarar för fler soptunnor än vad som finns på Chalmers och/eller har fler anställda som sköter soptömningen kan det enligt Sia också vara mer motiverat att strukturera upp arbetet med hjälp av ett system likt JunkSpy.

2.2.2 Intervju 2: Räddningstjänsten

En idé som brainstormingen utmynnade i var att installera en branddetektor i Spionen med syfte att tidigt upptäcka bränder som startar i soptunnor. Därför kontaktades Petter Backlund[2] för en intervju som arbetar med olycksuppföljning och analys på räddningstjänsten i Storgöteborg. Enligt honom finns det statistik från räddningstjänsten som säger att de åker på cirka 100 utryckningar per år där elden startat i antingen en soptunna eller i en papperskorg utomhus. Han tror dock att det verkliga antalet kan vara ännu fler, då statistiken beror på hur olika utryckningar har kategoriserats. Räddningstjänsten åker även på cirka 30 larm där branden startat i ett soprum, vilket anses ännu farligare då spridningsrisken till bostäder är större.

Enligt honom är bränderna oftast anlagda men kan även bero på faktorer som att folk varit ansvarslösa med fimpar, aska och engångsgrillar. Ett problem han ser med de gröna flyttbara plastsoptunnorna är att när en viss temperatur uppnås smälter tunnan ner och blir en flytande eldpöl. Lutar marken då åt fel håll kan branden sprida sig till intilliggande fastigheter.

Innan samtalet med Petter Backlund hade två förslag kring hur en brand kan upptäckas diskuterats fram. Den första idén var med hjälp av en rökdetektor, den andra med hjälp av en temperatursensor. Backlunds resonemang kring detta var att rökdetektering upptäcker brand fortare, men är osäkrare

[1] Intervjuad 2013-02-28.
[2] Intervjuad 2013-02-26.

och leder därmed ofta till falskt alarm. Av den anledningen används metoden generellt sett enbart då det finns risk för människoliv. Då bränder i soptunnor sällan är akuta ansåg Backlund att den mer pålitliga temperatursensorn är att föredra. Genom att sätta ett gränsvärde över den maximala temperaturen som kan uppnås i en soptunna bör bränderna kunna upptäckas i ett tidigt stadium, en temperatur som behöver vara ungefär 80 grader enligt Backlund.

Den sista frågan handlade om vad en tidigare upptäckt av en brand kan innebära. Svaret blev att det varierar. Bränder i soptunnor är oftast ganska små och ofarliga. Det kan till och med vara så att de inte upptäcks förrän några dagar senare då elden redan brunnit ut. Som nämnts ovan blir det så klart farligare om soptunnan står i närheten av en byggnad. Räddningstjänsten rycker generellt sett inte ut på larm som inte konstaterats då det alltid innebär en risk med utryckning i hög hastighet. Därför bör denna typ av system använda sig av ett vaktbolag eller annan servicepersonal som får åka ut till en soptunna som larmat. Eftersom bränderna oftast är små, till en början i alla fall, skulle den personen kunna ha med sig enklare släckningsutrustning. Är branden för stor kan vaktbolaget snabbt larma räddningstjänsten.

2.3 Krav- och testsammanfattning

För att uppnå en god kvalitet på Spionen har en krav- och testspecifikation utarbetats från kravinsamlingen, vilken återfinns i bilaga A. Kraven är uppdelade enligt Wohlins modell [28] för att skapa en tydlig kravstruktur innehållandes användarkrav och systemkrav. Genom att i varje enskilt krav lägga till ett test samt testets resultat blir det enkelt för läsaren att följa varje delkrav från början till slut. Nedan följer en sammanfattning av krav- och testspecifikationen.

JunkSpy-systemet ska kunna samla in och lagra mätdata från samtliga soptunnor i ett område som gjorts smarta genom att utrustas med en Spion. Via ett webbaserat användargränssnitt ska användaren kunna få tillgång till all data från soptunnorna i systemet samt kunna se den mest effektiva rutten för att tömma de soptunnor som behöver tömmas. Själva Spionen ska gå att montera i en soptunna med lock utan att förhindra användandet av soptunnan. Baserat på att projektet avgränsats till soptunnor av Göteborgsmodellen är ett rimlig krav att Spionen ska klara av att mäta avstånd på mellan 10 och 150 centimeter från sensorn. Det är också rimligt att felmarginalen i mätningen är under 5 % av det maximala mätavståndet. Därför krävs det också att de mätvärden som levereras av sensorn har minst 5 centimeters precision.

3 Teori

I följande avsnitt behandlas samtliga komponenter som används i projektet. Först behandlas projektets hårdvara och därefter mjukvara. Sist behandlas de mer generella avsnitten om säkerhet, ruttoptimering och miljöaspekter.

3.1 Hårdvara

Med hårdvaruutveckling avses utvecklingen och konstruktionen av Spionen som ska monteras i soptunnorna. Den består till huvudsak av en Arduino med tillhörande sköldar (se 3.1.2) och givare. Även programmering av mikrokontrollern tas upp i detta avsnitt. Hårdvaran har utvecklats iterativt genom laborering efter de krav som satts för hårdvaruiterationen. De tidiga iterationerna fokuserade på funktion för att sedan flytta fokus till effektivitet.

3.1.1 Arduino

Mikroprocessor ATMEL ATMega328

Klockfrekvens 16 Mhz

CPU 8-bit

Arduino är en prototypplattform och genom att läsa/skriva insignaler respektive utsignaler kan plattformen interagera med omvärlden. Plattformen sköter I/O (*input/output*) via stiftlister som sitter på kretsen, och det är till dessa som komponenter och givare kopplas [26]. Arduino och dess utvecklingsmiljö är licensierad under Creative Commons Share-Alike. De öppna licenserna var en bidragande faktor för valet av Arduino som utvecklingsplattform i projektet, och var det som avgjorde att den valdes framför andra lösningar, så som BASIC Stamp. Andra faktorer som spelade in var bland annat att den har en enkel utvecklingsmiljö samt att den enkelt kan byggas ut med extra funktionalitet genom sköldar (se avsnitt 3.1.2) för bland annat kommunikation via Ethernet, GSM och Bluetooth. Samt att delar av gruppen har tidigare erfarenhet av plattformen. Det finns även mycket erfarenhet och kompetens gällande Arduino på Chalmers då det är ett populärt val i kandidatarbeterna.

3.1.2 Sköldar

Sköldar (eng. *shields*) kan ses som en utökning av plattformen och monteras lödfritt via stiftlisterna på Arduino-kortet (se figur 1). Vanligtvis adderar sköldarna nya kretsar med speciell funktionalitet, exempelvis Ethernet-kommunikation eller motorstyrning.

3.1.3 Ethernet Shield

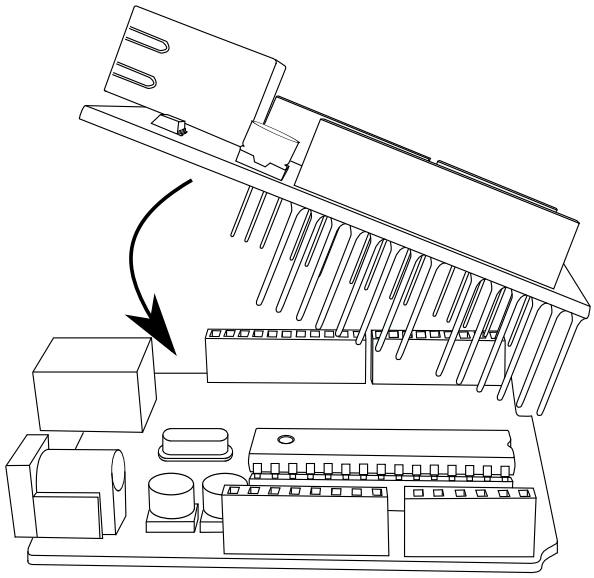
Ethernet-kontroller Wiznet W5100

Kommunikationsprotokoll TCP/IP

Kommunikationskontaktdon RJ45

Kommunikationstyp Trådad

Ethernet shield är en kommunikationsmodul som gör det möjligt för Arduino att kommunicera via Ethernet. I projektet används Ethernet för kommunikation över Internet eftersom det är en lättanvänd och välbeprövad teknik.



Figur 1: En Ethernet-sköld monteras lödfritt på Arduino.

3.1.4 Wireless Proto Shield

Wireless Proto Shield fungerar som en brygga mellan trådlös kommunikation och Arduino. Skölden tar emot data från den trådlösa XBee-modulen (se avsnitt 3.1.5) och skickar det till Arduino via plattformens seriella gränssnitt. Wireless Proto Shield valdes för att det är den officiella skölden för XBee-kretsarna och bedömdes då vara enklast att arbeta med för att uppnå snabbt resultat.

3.1.5 XBee S2

Kommunikationsprotokoll ZigBee 802.15.4

Kommunikationstyp Trådlöst

En liten krets som monteras på Wireless Proto Shield. Datautbytet mellan mikrokontrollern och XBee sköts via plattformens seriella gränssnitt. XBee sköter hela nätverksstacken vid trådlös kommunikation, vilket gör kommunikationstypen transparent för plattformen. XBee valdes för protokollet ZigBees (avhandlas i avsnitt 3.3.1) många fördelar för trådlös kommunikation mellan små, integrerade kretsar med begränsad datatrafik. WLAN (*Wireless Local Area Network*) övervägdes som kommunikationsform i projektets början, men fördelar gällande kostnadseffektivitet och strömsnålhet gjorde ZigBee till det slutgiltiga valet[27].

3.1.6 Parallax Ping))) avståndssensor

Mätintervall 2-300 cm

Mätteknik Ultraljud

Ping))) är en avståndsgivare från Parallax. Den uppskattar avståndet genom att skicka ut en kort puls av högfrekvent ultraljud [18] och sedan mäta tiden tills ekot återvänder. Genom att mäta hur lång tid detta tar går det att räkna ut avståndet till det närmaste objektet framför sensorn. Ping är

lätt att arbeta med och kräver lite förarbete. Av den anledningen valdes Ping framför att konstruera en liknande krets från grunden.

3.1.7 Sharp GP2Y0D02YK avståndssensor

Mätintervall 20-150 cm

Mätteknik IR-ljus

Sharp GP2Y0D02YK är en sensor som avläser huruvida dess IR-stråle har brutits [20]. Den kan därmed enbart avgöra huruvida sopnivån överstigit gränsen för dess avsökningsområde. Efter laboration kom gruppen fram till att sensorn inte var tillräcklig för projektet, och Ping (se avsnitt 3.1.6) valdes istället.

3.1.8 Energisparläge

Energisparläge (eng. *sleep mode*) innebär att Arduino kopplar från strömförsörjningen till de delar av plattformen som drar mycket ström [4]. Med detta kommer ett antal nya faktorer att ta i beaktande, framförallt att inga beräkningar kan utföras av mikroprocessorn under tiden Spionen sover. Spionen behöver då få ett avbrott (eng. *interrupt*) som väcker kretsen. Avbrottet kan exempelvis komma från en extern klockkrets, en intern klocka eller från ZigBee-chippet. När ett avbrott sker under energisparläget startar Arduino igen och kan då utföra beräkningar.

Även ZigBee-skölden kan sättas i energisparläge. Den innehåller en egen mikrokontroller, samt är utrustad med en radiosändare/-mottagare vilka vars strömförbrukning är nästan lika hög som Arduino. Uppmätningar av strömförbrukning visade att en ur energisynpunkt icke-optimerad Spion använder 100-110 mA. När Arduino sattes i energisparläge sänktes strömförbrukningen med cirka 10 mA, och när även ZigBee-chippet sattes i energisparläge sänktes strömförbrukningen ytterligare med cirka 40 mA.

Tiden som Spionen utför mätningar är mycket liten i förhållande till den övriga tiden. Av den anledningen är det mycket användbart att försätta enheten i energisparläge under tiden som inga mätningar utförs. Arduino är utrustad med en spänningsregulator för att skydda mikrokontrollern, och den förbrukar ström även i energisparläge. Genom mätningar framkom det att spänningsregulatorn förbrukar omkring 40-50 mA när alla komponenter var försatta i energisparläge. Genom att montera av mikrokontrollern och direktkoppla den till ZigBee-skölden sänktes strömförbrukningen till cirka 0,5 mA när alla komponenter ligger i energisparläge.

3.1.9 Strömförsörjning

Arduino har en rekommenderad driftspänning på 7-12 V, av vilket den klarar att leverera 5 V till periferienheter [26]. Atmega processorn som nu monterats på prototypskölden kräver en driftspänning på 1,8-5,5 V. Då till exempel avståndmätaren har en driftspänning på 5 V krävs därmed en strömkälla som levererar mellan 5 och 5,5 V. Som strömkälla valdes batteri, eftersom det inte ansågs praktiskt att dra matningsledning till det stora antal Spioner som skulle installeras i en konkret tillämpning.

För att nå den efterfrågade spänningen seriekopplas ett antal batterier vilket resulterar i en högre total spänning [17]. Den genomsnittliga strömförbrukningen beräknades på följande sätt:

Antag två lägen för spionen: Dag respektive Natt. Dag innebär 1 mätning per timma och under Natt genomförs 1 mätning var 5:e timma. Genom mätning med multimeter uppmättes ≈70 mA strömför-

brukning under de tre sekunder en mätning genomfördes. På samma sätt uppmättes en genomsnittlig strömförbrukning på 0,5 mA när inga mätningar genomfördes. Det gav följande uppställning för dag (14h):

- vaken: 3s med $\approx$ 70 mA
- sover: 3597s med $\approx$ 0,5 mA

Som ger $\frac{3 \cdot 70 + 3597 \cdot 0{,}5}{3600} = 0,56\text{mA/h}$.

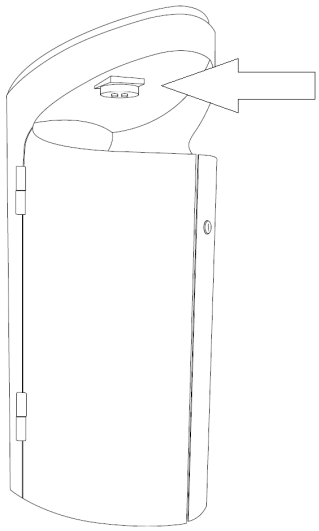
Natt (10h) ställdes upp i 5-timmarsintervall på följande sätt:

- Vaken: 3s med $\approx$ 70 mA
- Sover: 17997s med $\approx$ 0,5 mA

som ger $\frac{\frac{3 \cdot 70 + 17997 \cdot 0{,}5}{5}}{3600} = 0,51\text{mA/h}$ och det ger $0,56 \cdot 14 + 0,51 \cdot 10 = 12,95\text{mA/dag} \rightarrow 12,95 \cdot 365 = 4727,6\text{mA/år}$. Seriekopplas två stycken 3-volts batterier uppnås tillräcklig spänning. Parallellkopplas sedan två sådana serier kan omkring 5000 mAh levereras beroende på battterityp. Spionen kan då vara aktiv i 12 månader under ideala förhållanden.

3.1.10 Programmering av mikrokontrollern

Arduino programmeras i programspråken C/C++ med speciella bibliotek för Arduino-specifika funktioner [13]. Under projektets början användes bara enklare funktioner, så som att läsa och skriva från stiftlisterna eller det seriella gränssnittet. Allt eftersom projektet fortskred utökades programbiblioteken för de olika enheterna beroende på deras användningsområde. Mycket av programkoden är rena funktionsanrop till de färdiga biblioteken, kompletterat med egenutvecklade algoritmer och kod för databehandling.



Figur 2: Skiss på monterad Spion i kapsling.

3.1.11 Inkapsling

Eftersom Spionen ska monteras i en soptunna kräver den någon form av inkapsling. I soptunnan riskerar Spionen att utsättas för bland annat regn, kladdiga sopor och törnar från föremål som slängs. För att skydda den känsliga hårdvaran är det ett krav att inkapslingen ska uppfylla kapslingsklass IP65 [1]. Detta innebär att inkapslingen ska sluta tätt mot finkornigt damm samt tåla att spolas med vatten utan att någon fukt tränger igenom. Av ovanstående skäl valdes en inkapsling av glasfiberförstärkt polyester. Exempelskiss på hur inkapslad spion är monterad kan ses i figur 2.

3.2 Mjukvara

Mjukvarudelen av projektet består av en databas och en webbapplikation. Databasen lagrar alla mätningar som görs av Spionerna som sedan webbapplikationen ordnar och presenterar i ett webbgränssnitt. Med en webbapplikation kan användaren snabbt komma åt viktig information oberoende av vilken enhet eller plattform som denne använder för tillfället. Eftersom webbapplikationer körs i molnet behöver användare på klientsidan inte installera någon ytterligare proprietär programvara utan kan använda sig av de redan befintliga webbläsare som finns installerade på läsplattor, datorer och mobiltelefoner. Det blir istället servern som får ansvara för att leverera en applikation som passar för användarens enhet, något som kan göras på flera olika sätt med hjälp av responsiv webbdesign (se 3.2.2).

3.2.1 Webbapplikationen

Webbapplikationen har ett användargränssnittet för att presentera insamlad data från Spionen. Första mjukvaruiterationen resulterade tre i skisser, vilka återfinns i bilaga B. En plattformsoberoende webbapplikationen har därför utvecklats i HTML5, CSS3 och JavaScript. Nedan beskrivs de slutgiltiga vyerna, som det även finns skärmdumpar av i bilaga E.

Stadsvy I den första vyn som användaren möter illustreras en översiktskarta över samtliga soptömningsområden. På kartan är varje unik soptunna markerad med en färgkodning där grön innebär att soptunnan är tom, gul att tunnan är mindre än halvfull, orange att soptunnan är mer än halvfull samt röd som innebär att soptunnan behöver tömmas.

Områdesvy Samma som stadsvyn men zoomat på ett enskilt område.

Sensorvy Här visas all information om en enskild soptunna. Som standard visas mätdata för det senaste dygnet.

3.2.2 Responsiv webbdesign

För att uppfylla kravet att upprätta en plattformsoberoende webbapplikation används responsiv webbdesign. Responsiv webbdesign innebär att innehållet som presenteras på en webbapplikation anpassas unikt för varje enhet/webbläsare som läser in innehållet. Anpassningen sker beroende på enhetens hårdvaruegenskaper så som skärmstorlek, upplösning och andra I/O-komponenter tillhörande enheten.

CSS3 är ett populärt språk, eller stilmall, som anpassar utseendet för hemsidor och kan användas för responsiv webbdesign [15]. Webbapplikationen använder sig därför av CSS3 och utvecklas initialt för att kunna läsas in på en dator eller smartphone. CSS Media queries[16] är de regler som CSS3 använder sig av för att anpassa stilmallarna och gör det enkelt att utvidga support för flera enheter vid framtida utveckling.

3.2.3 HTML5

HTML5 är ett sidformateringsspråk som ger struktur för information som presenteras på webben. Språket tolkas av webbläsare som har i uppgift att presentera innehållet för användaren. HTML5 är den senaste HTML-standarden som rekommenderas vid nyproduktion av hemsidor av W3C [5], en världsomspännande organisation med representanter från regeringar, forskningsinstitut och näringslivet. Den stora fördelen med att använda HTML5 är införandet av tydligare instruktioner, regler och direktiv för att olika webbläsare ska kunna rendera innehåll konsekvent oberoende av webbläsare.

JunkSpy ska fungera som en webbapplikation och erbjuda stöd för flera olika typer av enheter och webbläsare, företrädesvis utan att flera olika implementationer av applikationer behöver kodas. HTML5 erbjuder just detta och stöds av de mest populära webbläsarna [21].

3.2.4 Lagring av mätdata

Varje gång en Spion utför en mätning skickas resultatet för lagring i en central databas. Varje Spion förknippas med ett unikt ID, och detta ID sparas i databasen tillsammans med mätvärdet och det klockslag och datum som mätningen utfördes. När användaren vill få ut data som finns lagrad i databasen sker detta via ett anrop från webbapplikationen till databasen.

Det går att använda informationen i databasen för att räkna ut annat som kan vara av intresse för användaren, så som genomsnittliga mätvärden med mera. Samtliga sådana beräkningar utförs vid behov av webbapplikationen och resultatet sparas inte i databasen. Detta görs för att undvika redundans i databasen, det vill säga att samma data upprepas på flera ställen. Redundans ger nämligen upphov till svårigheter med att undvika inkonsekvent data och andra problem [14]. Exakt hur databasen är organiserad visas i bilaga C.

Databasen är en relationsdatabas, där den data som lagras organiseras i form av kolumner och rader, vilket kallas för tabeller. Informationen i tabellerna binds samman av relationer och restriktioner som sätts upp med hjälp av nycklar och regler.

En relationsdatabas gör den initiala utvecklingsfasen mer komplicerad än om ett mer primitivt lagringssätt skulle ha använts. Detta beror inte minst på att många databaser inte kan användas direkt utan kräver licenser och installation/hosting. Ett mer primitivt lagringssätt skulle till exempel kunna vara att information lagras i en enkel textfil. Ur skalbarhetsperspektiv blir dock detta omständligt och opålitligt eftersom hantering av säkerhetskopiering, utvidgning av nya funktioner och konflikthantering blir svårhanterligt.

3.2.5 MySQL

Databasen hanteras av MySQL som är världens näst mest använda relationsdatabashanteringsystem [9]. MySQL bygger på öppen källkod och körs som en server vilken tillåter flera användare tillgång till ett flertal databaser. Frasen SQL står för Structual Query Language vilket är det språk MySQL använder för att hantera data. Det medföljer inget administrativt användargränssnitt i MySQL. Användaren kan välja mellan att använda inkluderade kommandotolksinstruktioner eller ett av många fristående MySQL-gränssnitt för att skapa och hantera databasen.

MySQL använder som standard InnoDB som lagringsmotor, vilken ger ACID[3]-kompatibla transaktionsfunktioner och möjlighet att använda främmande nycklar.

[3] ACID är en akronym för Atomicity, Consistency, Isolation och Durability. Dessa egenskaper är mål för att en databas ska garantera fullständig strukturell integritet.

MySQL har en stor användarbas vilket har bidragit till att en stor mängd publicerat material finns att tillgå om hur tjänsten kan användas på bästa sätt och anpassas för olika användningsområden. Vidare så underhålls databasen ofta och frekvent av ansvariga utgivare vilket säkerställer att nya funktioner tillkommer ofta och att säkerheten blir hög. Enligt analysföretaget 451 Group pekar mycket på att MySQL kommer att vara välanvänt inom en överskådlig framtid [3].

3.2.6 Google Maps

Mjukvarans kartor använder sig av Google Maps[8], en karttjänst som tillhandahålls av Google och som kan användas genom flera olika API (Application Programming Interface). Google Maps är fritt att använda om antalet anrop till servern är mindre än 25 000 per dag, därefter tillkommer en abonnemangsavgift. För att hålla reda på detta ges varje utvecklare en API-nyckel som är kopplat till en webbapplikation där användarstatistik kan överblickas.

Google Maps tillhandahåller en mängd olika inbyggda funktioner för att hitta kortaste vägen mellan två mål, placera ut markörer, rita ut vägar och stöd för olika plattformar. Stöd ges till exempel för JavaScript vilket är användbart för utveckling av webbapplikationer.

Google Maps är fritt att använda och gör det enkelt att snabbt kunna utnyttja tjänsten samtidigt som skalbarheten för Google är pålitlig. Användarbasen är stor vilket innebär att omfattande instruktioner och dokumentation för tjänsten finns lättillgängliga. Detta underlättar dels den initiala utvecklingsfasen och allt eftersom nya funktioner adderas finns möjligheten att lägga till dessa för att på så sätt förbättra webbapplikationen.

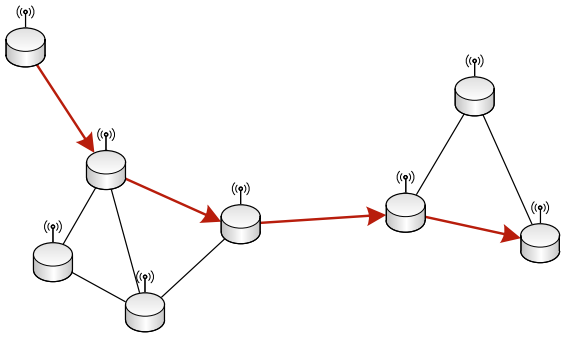
3.3 Kommunikation

Under projektets gång har flera olika former av kommunikationsmedium övervägts. Tidigt fastslogs att någon form av trådlös kommunikation är nödvändig - även om de tekniska utmaningarna blir något större är det enklare än att behöva dra kablar till varenda soptunna som ska göras intelligent. Av tre olika lösningar som undersökts - WLAN, GSM (Global System for Mobile communications) och ZigBee - valdes den sistnämnda av flera olika skäl. ZigBee är enkelt att integrera med Arduino. Internt inom ZigBee-nätverket är dessutom säkerhetskraven som satts upp för projektet (se sektion 3.4) redan uppfyllda per automatik. Så den enda säkerhetsaspekten som är kvar att implementera är den som rör överföringen från slutnoden i ZigBee-nätet till databasen.

3.3.1 ZigBee

ZigBee är en specifikation för trådlös kommunikation mellan två eller flera noder. ZigBee bygger på standarden IEEE 802.15.4 och är främst tänkt att användas tillsammans med 8-bitars styrkretsar i applikationer där pålitlighet, strömsnålhet och kostnadseffektivitet är viktigt [7]. Dessa är samtliga faktorer som det är viktigt att den färdiga prototypen uppfyller. ZigBee stödjer olika former av nätverkstopologier, bland annat *mesh*-strukturen som har använts i det här projektet. Begreppet mesh (sv. nät) innebär att när en nod skickar ett meddelande behöver detta nödvändigtvis inte ske direkt till mottagaren. Istället letar sig meddelandet fram via mellanliggande noder som i sin tur vidarebefordrar meddelandet i riktning mot slutdestinationen (se figur 3) [7]. Då Spionen är utvecklad för en stadsmiljö med mycket soptunnor kommer dessa att direkt kunna kommunicera med slutnoden i nätverket. Därför är detta en väl lämpad lösning.

ZigBee har en begränsad överföringshastighet - den teoretiska maxgränsen är 250 kbps, enligt [7] ligger hastigheten i praktiken omkring 25 kbps i ett mesh-nätverk. Detta är fullt tillräckligt då varje sändning endast innehåller ett ID och ett mätvärde.



Figur 3: Exempel på topologin över ett mesh-nätverk.

3.4 Säkerhet

Även om säkerhet inte är arbetets huvudfokus är det ett viktigt område att åtminstone ha tänkt igenom så att inga onödiga säkerhetsluckor lämnas öppna.

Eftersom data som Spionerna skickar in till databasen inte är känslig i sig behövs ingen avancerad kryptering för att dölja meddelandets innehåll. Däremot måste det centrala systemet kunna verifiera att rätt enhet har skickat meddelandet och att det inte har blivit manipulerat på vägen.

Den optimala lösningen utifrån de här kraven är att använda en hash-funktion. Då uppfylls dessa säkerhetskrav utan att onödigt mycket prestanda krävs för kryptering. Hash-algoritmen SHA-1 (*Secure Hash Algorithm*) är tillräckligt säker för dessa behov [24]. ZigBee implementerar dock redan krypteringsalgoritmen AES-128 (*Advanced Encryption Standard*) [6]. Detta innebär mer säkerhet än vad som behövs enligt ovanstående krav, men fördelen är att tid och arbete slipper läggas på säkerhetsfaktorn. Den enda svaga punkten som återstår är överföringen från bryggan till databasen, alltså när ett meddelande lämnar det interna ZigBee-nätverket och fortsätter ut via internet. Säkerheten i detta sista led har inte implementerats i projektet, men måste beaktas innan en färdig produkt kan släppas.

3.5 Att beräkna den optimala soprutten

Projektets back-end har bland annat i uppgift att beräkna den effektivaste soprutten för tömning. Att beräkna den effektivaste vägen där ett antal punkter ska passeras är ett klassiskt problem inom datalogi. Det benämns ofta *Traveling Salesman Problem*, eller *TSP*, och är ett beräkningsmässigt svårt problem (NP-Komplett) [11]. På grund av delproblemets stora omfattning och komplexitet valdes Google Maps API (se 3.2.6) för att lösa de TSP-problem som uppstår. För att bestämma de *noder* (i vårt fall soptunnor) som skall tas med i beräkningen delas soptunnorna upp i olika prioritet:

Prioritet 0 Sopnivå $\geq 80\%$, soptunnan måste tömmas.

Prioritet 1 Sopnivå $\geq 60\%$, soptunnan bör tömmas.

Prioritet 2 Sopnivå $\geq 30\%$, soptunnan kan tömmas.

Prioritet 3 Sopnivå $< 30\%$, soptunnan behöver inte tömmas.

där samtliga soptunnor av prioritet 0 (**P0**) alltid läggs in som noder i Google Maps. Detta tar dock inte hänsyn till eventuella soptunnor av **P1** eller **P2** som ligger i närheten av rutten. Det löses genom att först beräkna avståndet δ_{P} mellan två soptunnor av prioriteten **Pv** (lägre än **P0**) och **P0**. Därefter

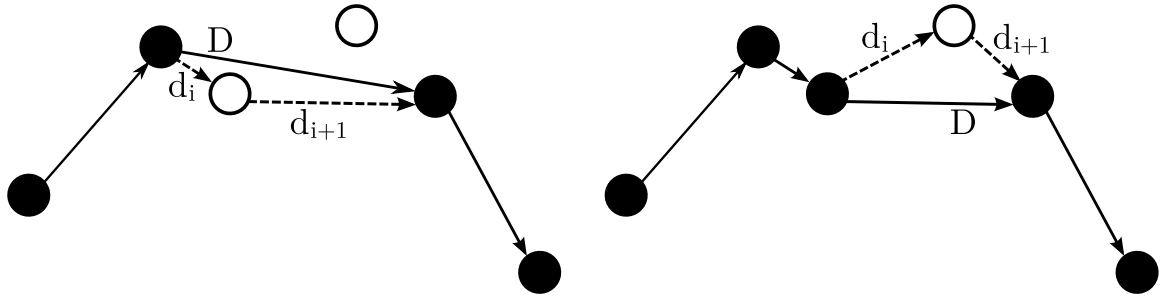
Pv	c_v
P1	0.2
P2	0.4
P3	0.7

Tabell 1: Antagna värden för c_v. Värdena kräver justering utifrån insamlade data under en längre tid, något som projektet saknar.

multipliceras avståndet med en konstant c_v som är given för alla prioriteter (se tabell 1) och på så sätt erhålls tömningskostnaden $d_i = \delta_{\mathrm{P}} \cdot c_v$. Tömningskostnaden d_i möjliggör beräkning av en effektiv lösning, där fler kilo sopor kan tömmas under en soptömningsrutt. Den alternativa rutten benämns r och dess totala sträcka $|r|$ och optimeringen kan beskrivas som

$$|r| = d_i + d_{i+1} \leq D$$

där D är det tidigare avståndet mellan två noder som ska besökas. Värden för c_v är givet i tabell 1. En körning av algoritmen illustreras i figur 4 där svart nod representerar **P1**, och vita noder lägre



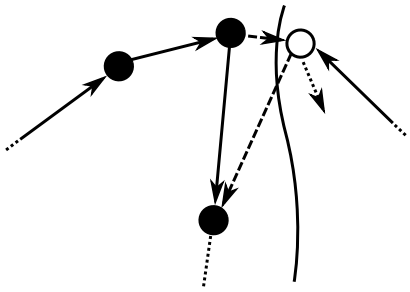
(a) Första iterationen

(b) Andra iterationen

Figur 4

prioritet. Problemet blir än mer komplext när hänsyn skall tas till att flera soptömningsbilar ska ut samtidigt. Ett sätt att lösa problemet är att dela upp soptunnorna i olika zoner, och tilldela varje bil en zon z. Det skapar dock ytterligare ett nytt problem som illustreras i figur 5. Omdefiniera d_i till $d_i = \delta_{\mathrm{P}} \cdot c_v \cdot c_v^+$ där $c_v^+ = ((\delta_{\mathrm{Pz}}) \,/\, (\delta_{\mathrm{Pk}})) + \Gamma$ där (δ_{Pz}) är sträckan till en soptunna $w \in z_n$ från z_m, och (δ_{Pk}) är sträckan mellan w och tidigare soptunnan i z_n och Γ är en konstant ökande proportionell mot avståndet från sopbilens tilldelade zon. Då kan bilar även designeras att tömma soptunnor utanför deras zon med en kostnad proportionell mot avståndet justerad med en konstant Γ för att hindra sopbilar att prioritera tunnor i andra zoner och strunta i sina egna. Genom att justera Γ till att skala linjärt, exponentiellt eller annan skala kan olika tolerans åstadkommas gällande avstånd in i annan zon. Därefter kommer många parametrar som krävs för att realisera algoritmen för en verklig situation, bland annat:

- Fördela soptunnorna i zoner, troligtvis geografiskt.
- Hur långt olika soptömningsbilar kan färdas.



Figur 5: Antag att svarta noder tillhör en zon z_m och vita z_n. Låt $|z_n| = 1$ och $|z_m| \gg |z_n|$. Att låta bilen i z_m även tömma den enda soptunnan i z_n är effektivare användning av sopbilarna än att skicka en bil för endast en soptunna.

- Hur lång tid ett soptömningspass får vara.
- Hur trafik och andra miljömässiga parametrar påverkar rutten.
- Etc.

Eftersom projektet saknar data för dessa parametrar abstraheras dem till en metod

```
boolean is_visit_possible_today(node v)
```

Givet en nod v applicerar metoden parametrarna och returnerar ett booleskt värde huruvida det är möjligt att besöka noden. Denna abstraktion möjliggör uppställningen av följande algoritm:

Data: Path l_0 of all nodes which has to be visited, List l_v of all nodes that can be visited
Input: l_0, l_v
Result: Path p
Boolean *done* is false;
while *not done* **do**
  set *done* to true;
  for *all pairs of nodes* n_i *and* $n_i + 1$ *in* l_0 **do**
    if $d_i \leq D$ *for any node* n_i^0 *in* l_v *&* $is\_visit\_possible\_today(n_i^0)$ **then**
      move n_i^0 to l_0;
      set *done* to false;
      break;
    end
  end
end
build and return path p from l_0;

3.6 Miljöaspekter

I rapportens inledning står det att en av utmaningarna med projektet JunkSpy är att utveckla ett system som leder till att personen som ska tömma en specifik soptunna slipper att besöka soptunnan för att avgöra om en soptunna behöver tömmas eller ej. Detta för att minska energiåtgången vid sophämtning.

Hårdvarumässigt har energisnålhet och hållbarhet varit två ledord under hela utvecklingsprocessen. Förutom att vara positivt för miljön är detta måsten för att produkten ska vara praktiskt användbar - om underhållet av Spionenheterna kräver mer arbete än vad som sparas in av att använda systemet förlorar det helt sitt syfte.

Eftersom produkten inte kräver att några soptunnor byts ut kommer dess miljöpåverkan inte vara särskilt stor. Av komponenterna som används är åtminstone Arduino dessutom certifierad med miljöcertifikatet Zero Impact [2]. Den största miljövinsten kommer dock att vara i form av insparat bränsle för sopbilarna i och med att antalet turer kommer att kunna minimeras.

4 Genomförande

I detta kapitel behandlas de fyra iterationerna som gjorts under projektet samt en beskrivning av dessa. Varje ny iteration är tänkt att producera en prototyp som bygger på lärdomar ifrån de tidigare. På så sätt tas en produkt fram som gradvis blir allt mer välarbetad.

4.1 Iteration 0 - Sensor och grundläggande kommunikation

Prototyp 0 utvecklades främst för att projektgruppen skulle bekanta sig med Arduino-hårdvaran och dess tillhörande mjukvara. Därför följdes ingen särskilt strukturerad arbetsmetod i det här stadiet. Med andra ord var det ett sätt att lära sig grunderna inom Arduino-utveckling genom praktisk tillämpning.

4.1.1 Kravlista för prototyp 0

Prototypen ska kunna mäta djup/avstånd med någon form av sensor. Den ska även sända dessa mätvärden till en dator som kan presentera dem.

4.1.2 Tillvägagångssätt

Eftersom syftet med prototyp 0 var att projektgruppen skulle bekanta sig med hårdvaran. Hårdvaruppsättningen ändrades under utvecklingen allt eftersom mer lättarbetade komponenter fanns att tillgå. Två olika typer av avståndsgivare utvärderades, SHARP IR-givare och Parallax ultraljudsgivare Ping))). Ping))) presterade bäst av de två och blev den givare som användes i nästkommande iterationer. Prototyp 0 använde Telnet som kommunikationsprotokoll.

4.2 Iteration 1 - Förfining och grundläggande mjukvara

Prototyp 1 är en vidareutveckling av prototyp 0. Denna prototyp ska klara av att göra mätningar som i sin tur skickas vidare med hjälp av Ethernet till ett databassystem där avstånd vid olika tidpunkter sedan kan läsas av.

4.2.1 Kravlista för prototyp 1

Hårdvaran ska klara av att mäta avståndet till objekt som ligger mellan 10 och 150 centimeter ifrån sensorn samt klara av kommunikation via Ethernet. Mjukvaran ska klara av att ta emot, lagra och presentera mätdata från prototypen.

4.2.2 Tillvägagångssätt

Iteration 1 utvecklades med kommunikation och presentation som huvudfokus. Arbetet skedde utifrån nätverkskommunikation via en Ethernet-sköld monterad på Arduino. Under utvecklingen av iteration 1 etablerades databasstrukturen samt protokollet för hur en mätning ska representeras. Även vilken data som skulle skickas fastslogs. Prototyp 1 anslöt till en lokal MySQL-databas för att skriva mätvärden samt ID. Valet av MySQL som databas var för att det är enkelt och gratis att få tag på vilket underlättade för testning och debugging hos prototypen. Iteration 1 använde det grafiska gränssnitt som verktyget phpMyAdmin erbjuder, därför konstruerades för inget eget gränssnitt för att presentera

mätdata för användaren. Sensorn Ping klarade av kraven som var uppsatta för prototypen, och var installerad och i bruk kort efter att iterationen påbörjats.

4.3 Iteration 2 - Kommunikation med ZigBee

Den största skillnaden mellan prototyp 1 och 2 är bytet av kommunikationsprotokoll från Ethernet till ZigBee.

4.3.1 Kravlista för prototyp 2

Förutom kraven från prototyp 1 ska prototyp 2 även klara av att skicka mätdata via ZigBee. Spionen ska också kunna utföra flera mätningar för att urskilja mätfel där till exempel en person kastar något i soptunnan i samma ögonblick som Spionen ska mäta. Enheten utför därför tre mätningar med en sekunds intervall, jämför resultaten och sänder det högsta värdet. Enheten ska även automatiskt gå in i energisparläge efter att ha gjort en sändning, och därefter kunna väckas igen via en avbrottssignal.

4.3.2 Brygga

För att testa ZigBee-kommunikationen behövs även en mottagarenhet. Denna enhet ska ha som ansvar för att ta emot mätdata från alla noder i ZigBee-nätverket och därefter vidarebefordra detta in till databasen. Mottagarenheten agerar alltså som en *brygga* mellan Spionerna och databasen.

För att kunna utföra detta måste alltså bryggan kunna ta emot meddelanden via ZigBee och sedan skicka vidare dessa till en dator via Ethernet. Eftersom bryggan inte kommer att vara monterad i någon soptunna utan sitta inkopplad till en extern strömkälla behöver den inte drivas på batteri. Den kommer heller inte att ha någon mätsensor som Spionenheten.

4.3.3 Tillvägagångssätt

Radiomodulen som användes i både Spion-prototypen och bryggan heter XBee S2 (se avsnitt 3.1.5). Anledningen till att denna modul valdes är att den stöder så kallade *mesh-nätverk*, det vill säga att alla noder i nätverket kommer att kunna utnyttjas för att skicka vidare meddelanden till vilken annan nod som helst som ligger inom räckvidd. Det har som fördel att någon ytterligare nätverksinfrastruktur inte kommer behövas. XBee S2 har en specificerad räckvidd på 40 meter (120 meter vid fri siktlinje), vilket är tillräckligt - åtminstone i prototypstadiet [12]. För att testa räckvidden i praktiken placerades bryggan inne i en byggnad på Chalmers och Spionen utanför. Sikten mellan enheterna skymdes av husväggar och annat, men de klarade ändå av att kommunicera med varandra utan problem på ett avstånd omkring 50 meter. Skulle det visa sig senare att en färdig produkt behöver ha längre räckvidd kan detta åtgärdas genom att byta radiomodul eller genom att montera en kraftfullare antenn.

4.4 Iteration 3 - Fullständigt energisparläge samt inkapsling

Detta är den slutgiltiga prototypen, som är tänkt att kunna testas praktiskt i fält. Den behöver därför ha någon form av inkapsling som går att montera i en riktig soptunna samt ha alla nödvändiga funktioner implementerade och fungerande. I den här iterationen färdigställdes även webbapplikationen, vilken visas i bilaga E.

4.4.1 Kravlista för prototyp 3

Prototyp 3 ska uppfylla alla krav som prototyp 2 uppfyller. Den ska även inneslutas i ett skal som går att montera i en riktig soptunna samt ha en fullt implementerat energisparläge, det vill säga att den automatiskt ska gå in och ur energisparläge med jämna mellanrum. Ett enklare flödesschema över händelseförloppet i prototyp 3 finns i bilaga D.

4.4.2 Tillvägagångssätt

Prototyp 3 utvecklades utifrån den hårdvara som tidigare var monterad. Vid mätning av dess strömförbrukning togs beslutet av optimera hård- och mjukvara för att minska strömförbrukningen. Efter flertalet försök valdes det att montera av mikrokontrollern från Arduino på grund av plattformens ineffektiva spänningsregulator. Efter att mikrokontrollern monterats på prototypkortsdelen på wirelessskölden (se avsnitt 3.1.4) var strömförbrukningen nere i acceptabla nivåer (0,5 mA). Energisparläge blev automatiserat på prototyp 3 och den klarade av att gå ner i energisparläge samt vakna upp och genomföra mätningar.

5 Resultat

Kapitlet behandlar det resultat och produkten som projektgruppen utvecklat och kommit fram till under projektet. Kapitlet är strukturerat så att de olika projektkomponenterna behandlas i följande ordning: hårdvara, kommunikation och mjukvara.

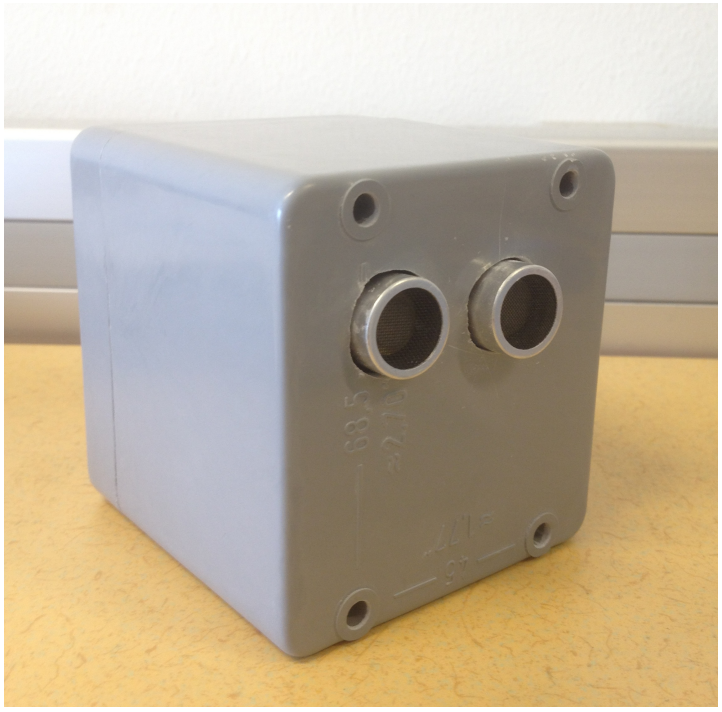
5.1 Hårdvara

5.1.1 Spion

Spionen består av:

- ATMEL ATMega328 Mikroprocessor (se 3.1.1).
- Parallax PING))) avståndssensor (se 3.1.6).
- Wireless Proto Shield (se 3.1.4).
- XBee S2 (se 3.1.5).
- 4 st AA Batterier.

I den slutgiltiga Spion-prototypen har mikrokontrollern ATMEL ATMega328 monterats direkt på prototypskölden med ZigBee-modul. Den har implementerats med ett energisparläge som sänkt energiförbrukningen. Mätningar utförs med hjälp av Parallax Ping))) och mätresultatet skickas sedan till Bryggan. Hela Spionen drivs av fyra stycken batterier och har en uppskattat batterilivslängd på strax över 12 månader.



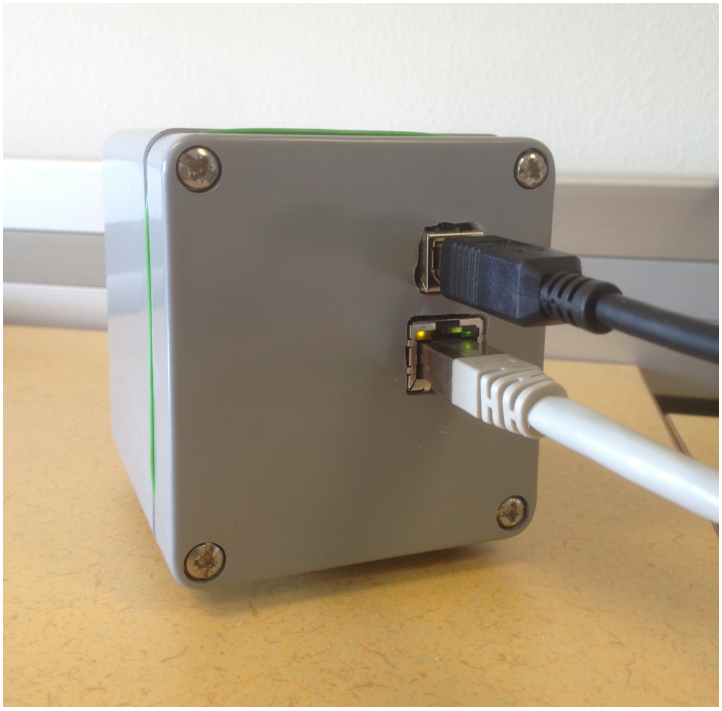
Figur 6: Inkapsling för Spionen.

5.1.2 Brygga

Bryggan består av:

- Arduino UNO (se 3.1.1).
- Ethernet Shield (se 3.1.3).
- Wireless Proto Shield (se 3.1.4).
- XBee S2 (se 3.1.5).

Bryggan består av en Arduino med två monterade sköldar. En prototypsköld med ZigBee-modul och en Ethernetsköld. Bryggans uppgift är att vidarebefordra mätningarna från Spionerna till webbapplikationen. Inget energisparläge finns implementerat på Bryggan, därmed är den alltid redo för att ta emot nya mätningar. Bryggan kräver tillgång till fast strömförsörjning samt en Ethernetport uppkopplad mot internet.



Figur 7: Inkapsling för bryggan.

5.2 Kommunikation

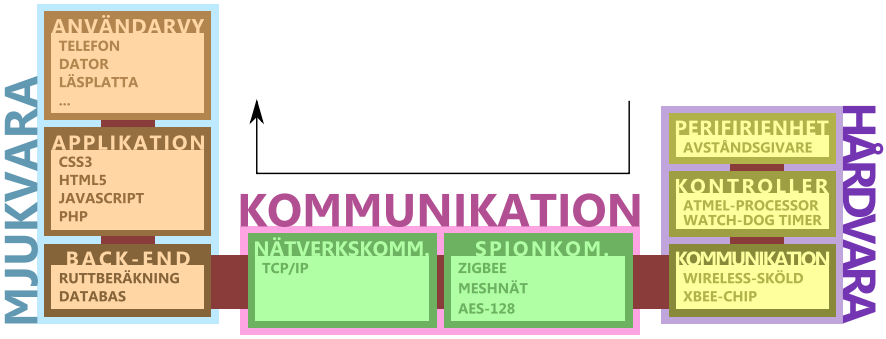
Projektet har behandlat två kommunikationsflöden. Mellan Spionen och Bryggan samt mellan Bryggan och webbservern. Mätvärden skickas från Spionen via Zigbee som är en radiostandard för trådlös kommunikation. Här värdesätts den snabba uppstartstiden och det låga energibehovet. Bryggan mottar sedan meddelandet från Spionen och sänder det vidare till en webbserver via Ethernet, en standard för datorkommunikationen via kabel. Mätningen kan sedan presenteras för användaren genom en webbapplikation.

5.3 Mjukvara

Mjukvaran är uppbyggd av:

- En MySQL-databas (se 3.2.5).
- En webbapplikation (se 3.2.1).
- Google Maps karttjänster (se 3.2.6).

Mjukvarudelen består av en databas och en webbapplikation. Databasen lagrar alla mätningar som utförts av Spionen i en relationsdatabas. Relationsdatabasen används i sin tur av webbapplikationen som presenterar data för användaren i tre olika vyer; stadsvyn, områdesvyn och sensorvyn. En detaljerad förklaring av de olika vyerna återfinns i bilaga E. Eftersom applikationen är webbaserad och stödjer responsiv webbdesign (se 3.2.2) kan den läsas in ifrån smarta telefoner, datorer och läsplattor.



Figur 8: Beskrivning av systemarkitekturen

Ruttberäkningen (se 3.5) har behandlats teoretiskt men är inte implementerad i sin helhet i webbapplikationen. Istället beror applikationen på Google Maps karttjänster för att optimera sophämtningsrutterna.

Figur 8 sammanfattar hur systemet i sin helhet fungerar.

6 Diskussion

I denna del av rapporten diskuteras arbetets resultat med syftet som utgångspunkt. Vidare kommer valet av arbetsmetod att utvärderas kring för och nackdelar med att arbeta iterativt samt utmaningar projektet stått inför. Avslutningsvis kommer framtida utvecklingsidéer tas upp och resoneras kring.

6.1 Utvärdering av metod

Den metod som använts under arbetets gång är den iterativa arbetsmetoden. Huvudsyftet med att arbeta iterativt är att för varje ny iteration producera en ny prototyp som bygger på lärdomar ifrån den tidigare.

Totalt har fyra iterationer utarbetats. En betydande fördel med denna typ av iterativa process är just att först börja i en liten skala för att sedan bygga ut och lägga till fler funktioner. Metoden har både fördelar och nackdelar. Exempelvis användes en Ethernet-sköld och Telnet-protokollet för kommunikation i prototyp 0, istället för ZigBee som i den färdiga prototypen. Därför fick tid läggas på något som inte kom med i slutprototypen, men samtidigt gjorde det att den första enkla prototypen kom att fungera tidigare. En positiv effekt av arbetsmetoden är att projektet ständigt hade hårdvara som gick att arbeta på vilket uppmanade till laborering och testning i samtliga delar av utvecklingsfasen.

Gruppen har till största del arbetat tillsammans, även om uppgifterna mellan gruppmedlemmarna varit uppdelat inom olika områden. Det värdefulla med att arbeta tillsammans är att det alltid funnits någon att diskutera och bolla tankar med, onödiga missuppfattar har kunnat undvikas och att gruppen tack vare varandra sällan hamnat på villovägar och arbetat åt fel håll. Sammantaget har den goda gruppdynamiken lett till att den har skapat en bra grund att stå på under utvecklingen av JunkSpy.

6.2 Utvärdering av systemet

I utvärderingen av systemet utgår diskussionen som tidigare nämnts från de sex huvudproblem som formulerades i inledningen. Detta för att på ett tydligt och strukturerat sätt beskriva olika svårigheter och utmaningar kring utvecklingen av hårdvara och mjukvara på detaljnivå.

6.2.1 Utveckla hårdvara kapabel att bedöma sopnivå

Det finns en mängd olika typer av sensorer som kan uppskatta sopnivån. Exempel på mättekniker är ljus, ljud eller vikt. En optimal lösning vore att använda flera sensorer av olika typer för att samla in så mycket data som möjligt för varje uppskattning.

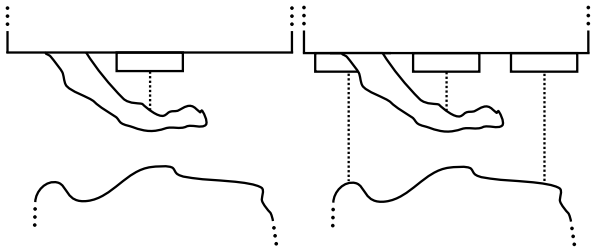
En viktsensor har i projektet behandlats teoretiskt och aldrig testats på en prototyp. Beroende på hur en full soptunna definieras får en viktsensor olika betydelse. Anses det att en tung soppåse är full blir en viktsensor, som alltså placeras under soptunnan, högst relevant. Projektet behandlar dock inte tyngden av soppåsar utan låter istället volymen sopor bestämma hur full soptunnan är. En viktsensor ger då inte tillräcklig med information om sopnivån. Om däremot sopavfallets densitet är känd, till exempel ett kärl för glasåtervinning, skulle endast en viktsensor kunna ge tillfredställande resultat.

Ett annat alternativ är att använda sig av en IR-sensor av den typen som kan avgöra avstånd till ett föremål. Däremot ansågs IR-sensorn inte vara tillräckligt pålitlig, bland annat för att mätningarnas

noggrannhet beror mycket på föremålets material och färg. Något som är svårt att förutse i en offentlig soptunna.

Ultraljudssensorn Ping))) kan med en centimeters precision bedöma sopnivån i soptunnan under förutsättning att skräpet i den är någorlunda jämnt fördelat. Ljudvågor har till skillnad mot IR-ljus en vidare utbredning och kan därmed underlätta detektering av föremål i soptunnans utkant[18]. Mot denna bakgrund ter sig Ping))) som ett utmärkt val av sensor till projektet.

Den allra bästa lösningen skulle antagligen vara en kombination av flera sensorer. Då går det till exempel att säkerställa att soptunnan varken är för tung eller för fylld. En annan nackdel med att bara använda en sensor är att Spionen då saknar sensorredundans (se figur 9). En sensor kan bli blockerad, skadad eller på annat sätt försatt i ett tillstånd som levererar ett felaktigt resultat. Ytterligare en eller flera sensorer kan då göra kontrollmätningar.



Figur 9: Illustration av problemet med avsaknad av sensorredundans. Ett föremål har fastnat i närheten av Spionen. Spionen till höger är utrustad med tre sensorer och klarar då av att mäta den korrekta sopnivån.

Att utöka Spionen med fler sensorer av samma eller olika typ skulle öka dess tillverkningskostnad samt strömförbrukning. Det i kombination med att en ensam sensor presterat tillfredsställande var skäl till att inga ytterligare sensorer använts till Spionen.

6.2.2 Utveckla energieffektiv hårdvara

Det är kostsamt och opraktiskt att dra fasta elledningar till varje soptunna som ska utrustas med Spionen. Därför behövs en annan strömkälla. Batterier är praktiska för ändamålet och finns i många former och prisklasser. Underhållsbehovet på Spionerna ska även hållas så litet som möjligt och eftersom batterier är en begränsad energikälla ställs höga krav på energiförbrukningen.

Generellt sett finns det två metoder för att öka batterilivslängden. Strömförbrukningen kan minskas genom till exempel energisparläge eller så kan mer energi tillföras med hjälp av en extern strömkälla. till exempel solceller.

Dock anses det här stadiet av utvecklingen vara tillfredställande med en effektiv energisparfunktion som gör att Spionen har mycket liten strömförbrukning när den inte utför mätningar och sänder data. Batterierna kommer att behöva bytas, men tack vare den låga strömförbrukningen kommer de att klara sig strax över ett år med befintlig batterikonfiguration. Däremot är solceller i kombination med energisparläge en intressant funktion att beakta i en eventuell vidareutveckling av produkten (se avsnitt 6.3).

6.2.3 Få hårdvaran att skicka mätdata

Valmöjligheterna angående kommunikation är stor. Likt strömförsörjning är det dyrt och omständigt att dra kablar till Spionerna. Därav utesluts alla former som inte är trådlösa. Alternativ som kan anses lämpliga måste vara säkra, pålitliga och ha möjligheten att enkelt kunna ansluta en ny Spion. Efter att ha undersökt vilka alternativ som uppfyller detta samt stöds av Arduino kvarstod WLAN, ZigBee och GSM.

Den största praktiska skillnaden mellan de tre är att WLAN och ZigBee behöver en närliggande brygga (med internetuppkoppling) för att vidarebefordra information till backend medan GSM kan operera självständigt. Räckvidd för WLAN och ZigBee är upp till 120 meter[12],[27]. ZigBee har möjligheten att använda mesh-nätverk vilket leder till ett mindre antal nödvändiga bryggor.

För att bibehålla strömsnålhet krävs snabb uppstart och kort sändningstid, samt att enheten i sig drar lite ström. ZigBee är i detta avseende överlägset med uppstart samt sändningstid under 1,5 sekund och strömförbrukning på 45 mA. WLAN och GSM drar uppemot 150 mA[27] respektive 500 mA[25], båda med betydligt längre uppstartstid än ZigBee.

Eftersom GSM har en hög strömförbrukning och JunkSpy relativt svag strömkälla var inte användandet av GSM ett alternativ Därför valdes ZigBee som både kräver mindre ström och ett mindre antal bryggor än WLAN.

I den slutgiltiga kommunikationslösningen skickar Spionen en mätning och sitt ID i form av en 8 siffror lång sträng som sedan avkodas och skickas vidare till back-end av bryggan. Detta uppfyller kommunikationskraven och även kraven för strömsnålhet. Den färdiga prototypen har en räckvidd på omkring 50 meter vid skymd sikt vilket är tillräckligt för prototypstadiet. Räckvidden kan ökas genom att montera en kraftfullare antenn, vilket kan behövas hos en Spion monterad i områden med glesare placerade soptunnor.

6.2.4 Lagra sopnivåer och avgöra soptunnors sopstatus

Webbapplikationen klarar av att ta emot mätningar ifrån ett stort antal Spioner och lagra mätdata i en databas. Databasen kan sedan tillhandahålla webbapplikationen med information om soptunnors djup och mätpunkter för att på så sätt kunna beräkna hur många procent av en viss soptunna är fylld. För att avgöra vilken tidpunkt som mätningen togs stämplas varje nytt mätvärde med aktuell tid, detta görs först när mätningen når webbservern. Nackdelen med att ge mätningen med en tidstämpel först hos webbservern är att den registrerade och faktiska tiden kan skilja sig med ett par sekunder. Detta har dock ansetts vara av mindre besvär med tanke på att intervallet för varje mätning är satt till 1 timme. Vidare skulle tidstämpling redan hos Spionen innebära att mer kod skulle behöva köras för att synkronisera klockorna och därmed minska batteritiden. Provkörningar av systemet har visat att de flesta mätningarna kommer in på servern inom ett par millisekunder.

Under utvecklingen av Spionen gjordes valet att hela tiden mäta den högsta soppunkten i soptunnan, vilket sensorn mäter med god precision. Detta visade sig dock inte vara helt optimalt vid alla typer av mätningar. Vid de mätningarna som utfördes med Spionen fylldes en soptunna med olika typer av material för att avgöra hur korrekta mätningar sensorn kunde utföra. Mätningarna utfördes med följande material och resultat.

6.2.5 Beräkna effektiva sophämtningsrutter

I dagsläget går det att beräkna den effektivaste soptömningsrutten på de hårdkodade soptunnorna med hjälp av en utarbetad algoritm för detta ändamål (se 3.5). Däremot är inte algoritmen implementerad i applikationen. Därmed uppfylls inte kravet 1.01 Hitta smartaste soptömningsrutten.

Algoritmen implementerades i en enkel version, men av flera anledningar som tas upp avsnitt 3.5 var det inte av vidare intresse att i detta skede implementera algoritmen fullt ut. Detta eftersom algoritmens resultat skulle vara ytterst approximativa som bäst.

6.2.6 Presentera data för slutanvändaren på ett tydligt sätt

Slutanvändaren av JunkSpy-systemet är tänkt att ansvara för ett stort antal soptunnor som sträcker sig över ett större geografiskt område i storleksordning av en stad eller en stadsdel. Detta ledde utvecklingen av tre olika vyer för webbapplikationen som var för sig tillhandahåller data om en viss sensor, en grupp av sensorer eller en överblick av en hel stads sensorer.

Under den första intervjun som hölls med Svensk Markservice (se avsnitt 2.2.1) framgick det att sophanterare är ute i fält under en stor del av arbetstiden vilket var anledningen till att JunkSpy utvecklades till en webbapplikation. Genom att distribuera applikationen över webben tillåts användare tillgång till systemet ute i fält så länge som det finns tillgång till internetuppkoppling. Vidare är webbapplikationen plattformsoberoende så länge det finns stöd för en modern webbläsare hos enheten. Nackdelen med att låta applikationen vara plattformsoberoende är att kompatibilitetstestning blir mer komplext att utföra, dock ansågs detta vara av mindre vikt eftersom plattformsoberoendet öppnar upp applikationen till flera användare.

Webbapplikationen har idag stöd för att anpassa presentationen av data beroende på den skärmstorlek som användaren hämtar hemsidan på. Detta bidrar till att data presenteras på ett tydligt och överskådligt sätt för användaren på så väl mobila och stationära enheter.

6.3 Framtida utveckling

I dagsläget innehåller Spionen de basala funktioner som krävs för att utgöra en mätning, men det finns stor utvecklingspotential. Nedan listas nya funktioner som kunde vara bra att införa om utvecklingen hade pågått under en längre tid och förklaringar av dessa.

Ruttberäkningsalgoritm Först och främst hade fokus legat på att implementera ruttberäkningsalgoritmen (se 3.5) fullt ut. Vid ruttberäkningen går det även att ta hänsyn till fler faktorer, exempelvis avståndet mellan soptunnorna och hur lång tid det var sedan soptunnan tömdes senast. En annan faktor som bör tas hänsyn till är att sopbilen behöver åka och tömmas då och då. Detta borde vara enkelt att schemalägga eftersom både mängden skräp och sopbilens volym är två kända variabler.

Förutse soptömning Under projektet har realtidsdata använts för att visa den aktuella sopstatusen och skapa soptömningsscheman. Som ett komplement till detta skulle en framtida version av systemet även kunna göra kvalificerade uppskattningar kring när en soptunna beräknas bli full, baserat på historisk data. Med en sådan teknik skulle smartare soptömningsrutter kunna beräknas.

Brandindikator Mot bakgrund av intervjun med räddningstjänsten i Storgöteborg prioriterades vikten av en brandindikator för att upptäcka bränder ned i JunkSpy. Däremot skulle lösningen vara intressant på vissa platser, exempelvis på skolor. Dock bör beslut kring huruvida en brandindikator ska inkluderas tas i samråd med varje specifik kund. Det då kunden är den som måste tillhandahålla personal som rycker ut för att släcka alternativt larma räddningstjänst då en eventuell brand indikerats.

Solceller För att öka Spionens batteritid ytterligare skulle enheten kunna utrustas med solceller. Solceller medför dock vissa utmaningar eftersom monteringen av Spionen i soptunnan skulle försvåras. Detta, tillsammans med priset på solceller, medför att tillverkningskostnaden stiger. Den högre kostnaden för solceller måste därför ställas i relation till kostnaden att manuellt byta ut batterierna när de blir dåliga. Hur lång livslängd som soptunnan beräknas ha är en avgörande faktor i den här beräkningen.

Fälttester För att produkten ska kunna massproduceras och användas i praktiken behöver JunkSpy utsättas för omfattande fälttester för att säkerställa att systemet lever upp till de satta kraven. Detta gäller samtliga delar av systemet, såväl själva Spionerna som den bakomliggande mjukvaran. Därefter måste kostnaderna för serietillverkning undersökas och om möjligt pressas, förslagsvis genom att byta ut diverse komponenter mot billigare men likvärdiga alternativ.

Säkerhet Slutligen skulle även säkerheten i överföringen mellan bryggan och databasen (se 3.4) ses över, då detta led inte skyddas av de inbyggda säkerhetsfunktionerna i ZigBee.

7 Slutsats

JunkSpy fungerar till stor del som planerat. Fungerande prototyper för både Bryggan och Spionen har framställts och dessa kan kommunicera med webbapplikationen. Det mesta av kravspecifikationen är implementerad med undantag för ruttoptimeringen i webbapplikationen.

Trots att ambitionen fanns att utveckla flera fungerande Spion-prototyper uteblev detta med anledning av tidsbrist och finansiella begränsningar.

Prototyperna och projektet i sin helhet planeras inte att vidareutvecklas trots stor utvecklingspotential, men lämnas öppet för den som är intresserad.

Referenser

[1] Henrik Andersson. *IP-klassning*. SP Sveriges Tekniska Forskningsinstitut. 2009. URL: http://www.sp.se/sv/index/services/ip/sidor/default.aspx (hämtad 2013-04-25).

[2] Arduino. *Arduino Manufacturing and Carbon Neutrality*. 2009. URL: http://blog.arduino.cc/2009/10/13/arduino-manufacturing-and-carbon-neutrality/ (hämtad 2013-04-25).

[3] Matt Aslett. *451 Research database survey points to MySQL gravitational tug of war*. 451 Research. 2013. URL: https://451research.com/report-short?entityId=76835 (hämtad 2013-05-09).

[4] Atmel. *ATmega48PA/88PA/168PA/328P*. Data sheet.

[5] World Wide Web Consortium. *ABOUT W3C*. World Wide Web Consortium. 2013. URL: http://www.w3.org/Consortium/ (hämtad 2013-05-09).

[6] Shahin Farahani. *ZigBee wireless networks and transceivers*. Newnes, 2008. ISBN: 9780750683937.

[7] Drew Gislason. *Zigbee Wireless Networking*. Newnes, 2008. ISBN: 9780750685979.

[8] Google. *Google Developers*. Google. 2013. URL: https://developers.google.com/maps/ (hämtad 2013-05-09).

[9] solid ITy. *DB-Engines Ranking of Relational DBMS*. solid IT. 2013. URL: http://db-engines.com/en/ranking/relational+dbms (hämtad 2013-04-17).

[10] Ola Johansson. "The effect of dynamic scheduling and routing in a solid waste management system". I: *Waste Management* 26.8 (2006), s. 875–885.

[11] Jon Kleinberg och Éva Tardos. *Algorithm Design*. Pearson International, 2006, s. 474. ISBN: 0-321-37291-3.

[12] MaxStream. *XBee Series 2 OEM RF Modules*. Product Manual v1.x.1x - ZigBee Protocol.

[13] David A. Mellis. *Arduino/Processing Language Comparison*. Arduino. Juni 2007. URL: http://arduino.cc/en/Reference/Comparison (hämtad 2013-04-17).

[14] Microsoft. *Grundläggande databasnormalisering (Artikel-id: 283878)*. Microsoft. 2008. URL: http://support.microsoft.com (hämtad 2013-05-09).

[15] Microsoft. *Responsive Web Design*. Google. 2011. URL: http://msdn.microsoft.com/ (hämtad 2013-05-09).

[16] Mozilla Developer Network. *CSS media queries*. Mozilla Developer Network. 2013. URL: https://developer.mozilla.org/en-US/docs/Web/Guide/CSS/Media_queries (hämtad 2013-05-09).

[17] "Numerical simulation for the discharge behaviors of batteries in series and/or parallel-connected battery pack". I: *Electrochimica Acta* 52.3 (2006), s. 1349 –1357.

[18] Parallax. *PING))) Ultrasonic Distance Sensor (#28015)*. Data sheet.

[19] RZ Riboverken. *Ett helt affärsområde i Papperskorgar*. RZ Riboverken. 2013. URL: http://www.rzg.se/referenser/ett-helt-affarsomrade-i-papperskorgen/ (hämtad 2013-04-25).

[20] Sharp. *Sharp GP2Y0D02YK IR-Sensor Optoelectronic Device*. Data sheet.

[21] Deep Blue Sky. *HTML5 & CSS3 Support*. Deep Blue Sky. 2013. URL: http://fmbip.com/litmus (hämtad 2013-05-09).

[22] Smartbin. *Smartbin*. Smartbin. 2013. URL: http://www.smartbin.com (hämtad 2013-04-24).

[23] BigBelly Solar. *BigBelly Solar*. BigBelly Solar. 2013. URL: http://www.bigbelly.com/solutions/stations/smartbelly/ (hämtad 2013-04-22).

[24] Timothy Stapko. *Cryptography for embedded systems - Part 1: Security level categories & hashing*. EE Times. Juni 2010. URL: http://www.eetimes.com/ (hämtad 2013-04-01).

[25] Itead Studio. *IComSat v1.1*. Schematic.

[26] Dale Wheat. *Arduino Internals*. Apress, 2011. ISBN: 9781430238829.

[27] WizNet. *WizFi210 User Manual*. Data sheet.

[28] Claes Wohlin. *Introduktion till programvaruutveckling*. Studentlitteratur AB, 2005. ISBN: 9789144028613.

A Krav- och testspecifikation

I den här bilagan listas samtliga krav som den slutgiltiga prototypen ska uppfylla. Intressenten *administratör* är den person som styr systemet och har särskild access för att exempelvis lägga till eller ta bort soptunnor ur systemet.

Funktionella Användarkrav

ID : 1.01	Räkna ut den effektivaste soptömningsrutten
Intressenter	Användare
Utlösare	Användaren vill se den mest effektiva soptömningsrutten för ett specifikt område.
Förhandsvillkor	Minst en soptunna finns inom soptömningsområdet som behöver tömmas.
Huvudhändelseförlopp	1. Användaren navigerar till områdesvyn. 2. Användaren klickar på knappen som beräknar den optimerade soptömningsrutten. 3. Systemet räknar ut den effektivaste vägen där alla soptunnor som behöver tömmas i ett område besöks. 4. Användaren får upp en lista där den mest effektiva soptömningsvägen listas samt en karta över resvägen.
Alternativa händelseförlopp	-
Eftervillkor	-
Test	Användaren klickar på knappen som beräknar den bäst optimerade soptömningsrutten och får upp en lista med den smartaste soptömningsvägen.
Resultat	Webbapplikationen presenterar den optimerade sophämtningsrutten för soptunnor som behöver tömmas.

ID : 1.02	Visa sopstatus för varje unik soptunna
Intressenter	Användare
Utlösare	Användaren vill kunna se varje soptunnas sopstatus för att avgöra om soptunnan behöver tömmas eller ej.
Förhandsvillkor	-
Huvudhändelseförlopp	1. Användaren väljer soptömningsområde. 2. Användaren får upp en vy där alla soptunnor i området listas tillsammans med deras status.
Alternativa händelseförlopp	3. Om användaren klickar på en specifik soptunna visas mer detaljerad information.
Eftervillkor	-
Test	Användaren navigerar till områdesvyn där den aktuella statusen för soptunnorna visas.
Resultat	Områdesvyn visas med samtliga soptunnor för området.

ID : 1.03	Visa tidigare mätningar för en soptunna
Intressenter	Användare
Utlösare	Användaren vill se tidigare mätningar.
Förhandsvillkor	-
Huvudhändelseförlopp	1. Användaren väljer soptömningsområde. 2. Användaren får upp en vy där alla soptunnor i området listas tillsammans med deras status. 3. Användaren klickar på en specifik soptunna för att visa mer detaljerad information.
Alternativa händelseförlopp	-
Eftervillkor	-
Test	Användaren klickar först på knappen 'city region' och därefter på en sensor där de tidigare mätningarna visas i ett diagram.
Resultat	Tidigare mätningar illustreras i applikationen.

Funktionella Systemkrav

ID : 2.01	Hantera ny mätning
Intressenter	Administratör
Utlösare	Spionenheten skickar in en mätning till databasen.
Förhandsvillkor	Spionen som rapporterar finns registrerad i systemet & meddelandet med mätresultatet är korrekt formaterat och innehåller inga uppenbara felaktigheter.
Huvudhändelseförlopp	1. En Spion rapporterar automatiskt in ett mätresultat. 2. Mätdata lagras i databasen. 3. Systemet uppdaterar all statistik och den aktuella soptunnans status.
Alternativa händelseförlopp	-
Eftervillkor	-
Test	Gör en mätning och se så att mätningen läggs in i databasen.
Resultat	Varje mätning sker under 3 sekunder en gång i timmen och skickas därefter direkt till databasen.

ID : 2.03	Om kontakt med en nod förloras underrättas användaren.
Intressenter	Användare
Utlösare	En nod i nätverket har inte skickat någon mätdata de senaste 24 timmarna.
Förhandsvillkor	-
Huvudhändelseförlopp	1. Nästa gång användaren använder webbapplikation visas ett meddelande som informerar användaren om vilken nod som har tappat kontakten med nätverket.
Alternativa händelseförlopp	
Eftervillkor	
Test	Koppla bort en nod och se så att ett varningsmeddelande skickas.
Resultat	Ett varningsmeddelande dyker upp om sensorn inte har rapporterat något mätvärde de senaste 24 timmarna.

ID : 2.04	Systemet sköter kommentar kring en soptunna
Intressenter	Användare
Utlösare	Användaren vill skriva eller redigera en kommentar om en soptunna.
Förhandsvillkor	-
Huvudhändelseförlopp	1. Användaren klickar på en soptunna. 2. Användaren lägger till en beskrivande text.
Alternativa händelseförlopp	- 2a) Det finns redan en kommentar om soptunnan. Användaren redigerar kommentaren istället.
Eftervillkor	- Kommentaren sparas i systemet.
Test	Testa så att det går att skriva en kommentar och att den sparas.
Resultat	Det går att skriva kommentarer angående en soptunna i databasen och denna visas upp i webbapplikationen.

Ickefunktionella Systemkrav

- ID: 3.01. JunkSpy-systemet ska vara webbaserat.

Funktionella Hårdvarukrav

- ID: 4.01. Spionen ska kunna mäta sopnivån i en soptunna.

  Test: Montera Spionen i en soptunna och låt den rapportera sopnivå.

  Resultat: Spionen rapporterar sopnivån.

- ID: 4.02. Spionen utför en mätning.

  Test:

  Hopknycklade A4-papper Fungerade bra, samma höjd på skräpet.

Plastpåsar och ett tungt föremål Fungerade dåligt eftersom soptunnan fort fylldes men så fort ett tungt föremål slängdes på dessa pressades sopmängden ihop.

Paraply Fungerade dåligt eftersom ett paraply lutandes mot ena kanten ser ut att vara en fylld soptunna enligt mätningarna även om det i verkligheten finns plats kvar att slänga sopor på.

- ID: 4.02. Spionen ska kunna skicka sina mätvärden till back-end.

  Test: Montera Spionen i en soptunna och låt den trådlöst rapportera sopnivån till bryggan.

  Resultat: Spionen rapporterar sopnivån till bryggan. Mätningen dyker slutligen upp i webbapplikationens databas.

- ID: 4.03. Spionen ska gå att montera i Göteborg stads egen soptunnemodell (se figur 10) och liknande soptunnor.

  Test: Mät storleken av inkapslingen för Spionen.

  Resultat: Storleken av inkapslingen är 75 mm x 75 mm x 80 mm. Inkapslingen är tillräckligt liten för att kunna monteras i en soptunna likt den som används i Göteborg.

Figur 10: En soptunna av Göteborgsmodellen. 1800 soptunnor av denna modell finns i Göteborg [19] (Egen bild).

Ickefunktionella Hårdvarukrav

- ID: 5.01. Spionen ska klara av att mäta ett minavstånd på 10 centimeter.

  Test: Mät om avståndet stämmer när det är större än 10 cm.

  Resultat: Spionen klarar av ett minavstånd på 2 cm.

- ID: 5.02. Spionen ska klara av ett maxavstånd på 150 centimeter.

  Test: Mät om avståndet stämmer när det är mindre än 150 cm.

  Resultat: Spionen klarar av ett maxavstånd på 370 cm.

- ID: 5.03. Spionen ska ha en mätprecision som är bättre än 5 centimeter.

  Test: Låt Spionen rapportera flera mätningar och kontrollera mätningen manuellt Resultat: Spionen har en mätprecision som är bättre än 5 centimeter mellan 2 till 370 cm.

- ID: 5.04. Spionen ska ha konstant strömförsörjning i minst 1 år.

  Test: Räkna ut hur stor energiåtgång Spionen har och hur länge batteriet håller.

  Resultat: Spionens strömförsörjning räcker nästan exakt ett år under optimala förhållanden.

- ID: 5.05. Spionen ska tåla temperaturer mellan $-40\,^{\circ}\mathrm{C}$ och $70\,^{\circ}\mathrm{C}$

  Test: Utsätt Spionen för de specificerade temperaturerna.

  Resultat: Ej verifierat.

- ID: 5.06. Spionen ska uppfylla kapslingsklass IP65[4].

  Test: Kontrollera specifikationen för Spionen.

  Resultat: Den omodifierade inkapslingen uppfyller kapslingsklass IP65, dock är kapslingsklassen för den färdiga Spionen ej verifierad.

- ID: 5.07. Spionen bör kunna kommunicera trådlöst.

  Test: Kontrollera att trådlöskommunikation sker.

  Resultat: Spionen kommunicerar via ett ZigBee-nätverk.

- ID: 5.8. Spionen ska gå att installera på olika vanliga typer av soptunnor på under 15 minuter.

  Test: Utför installation av Spionen i en soptunna och mät tiden

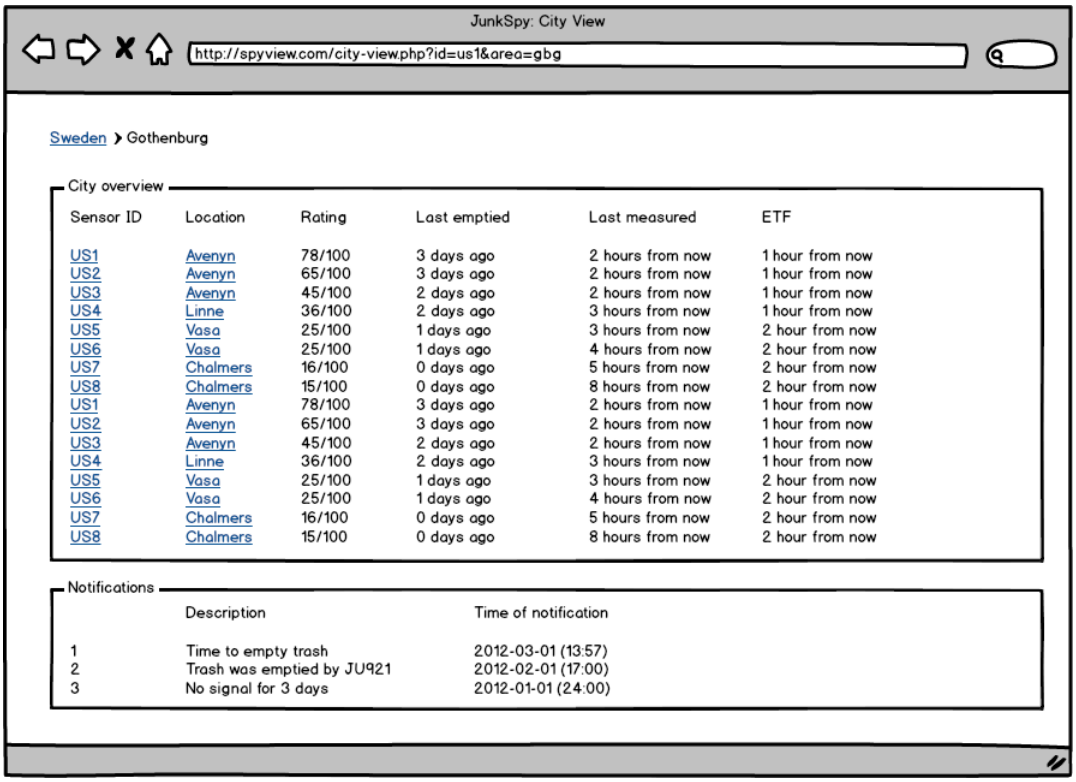
  Resultat: Ej verifierat.

- ID: 6.01. När en ny soptunna installeras måste den sättas upp inom 50 meter från en annan smart soptunna eller en ZigBee-router.

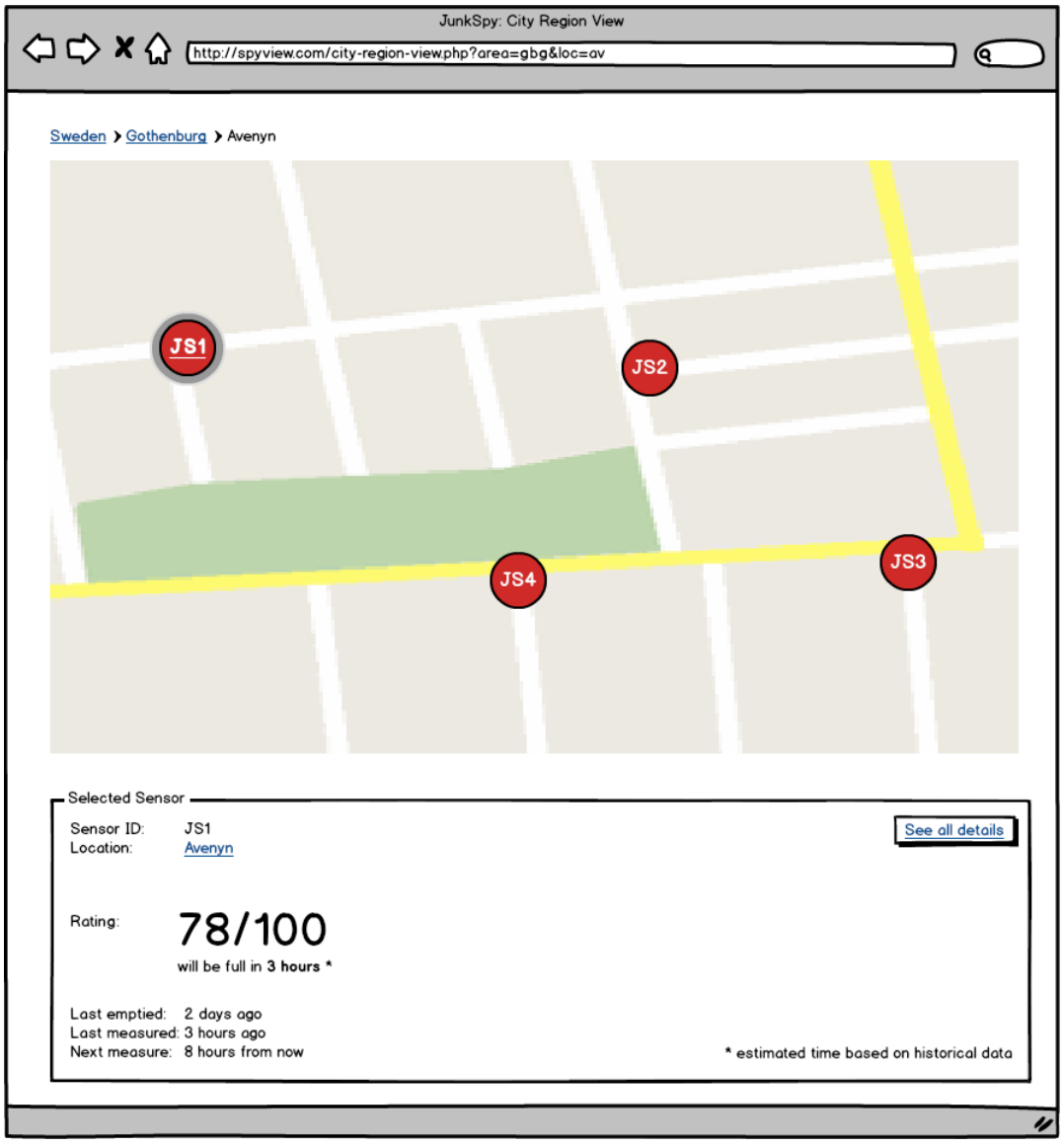
[4]Dammtät och spolningssäker - se [1].

B Skiss av användargränssnitt för webbapplikation

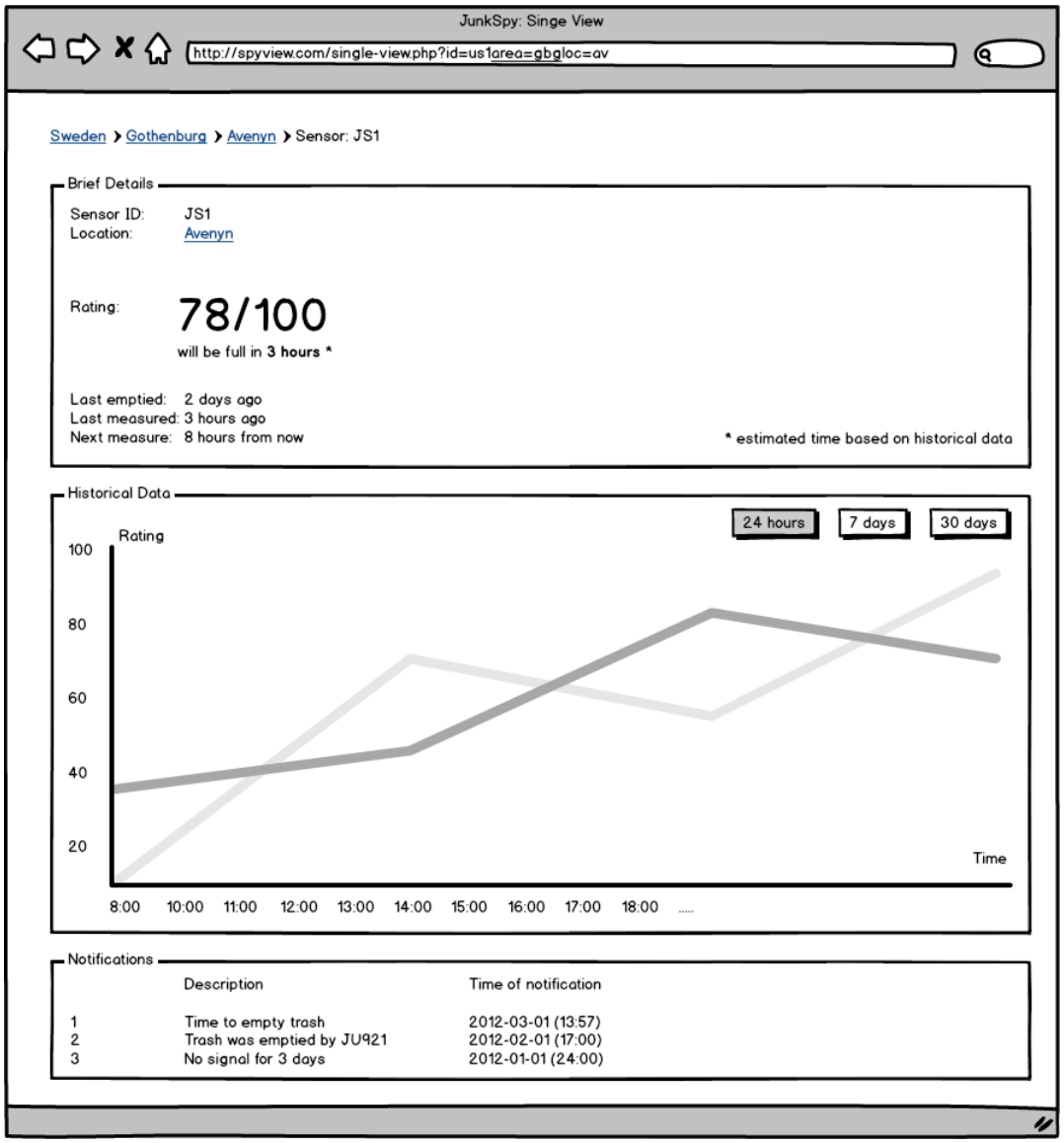
Stadsvyn



Områdesvyn


JunkSpy: City Region View
http://spyview.com/city-region-view.php?area=gbg&loc=av
Sweden › Gothenburg › Avenyn
JS1
JS2
JS3
JS4
Selected Sensor
Sensor ID: JS1
Location: Avenyn
See all details
Rating:
78/100
will be full in 3 hours *
Last emptied: 2 days ago
Last measured: 3 hours ago
Next measure: 8 hours from now
* estimated time based on historical data

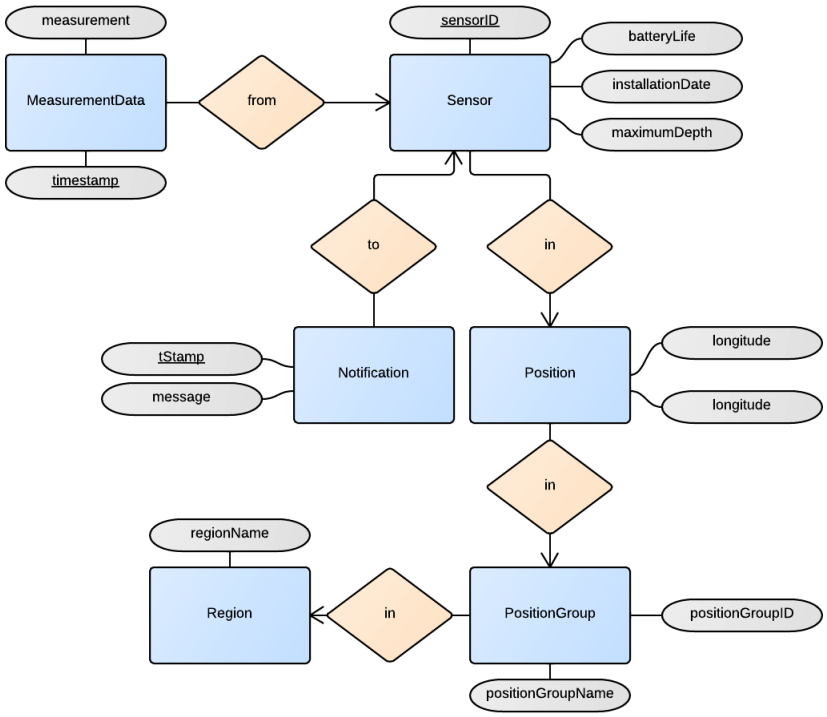
Sensorvyn


JunkSpy: Singe View
http://spyview.com/single-view.php?id=us1area=gbgloc=av
Sweden › Gothenburg › Avenyn › Sensor: JS1
Brief Details
Sensor ID: JS1
Location: Avenyn
Rating: 78/100
will be full in 3 hours *
Last emptied: 2 days ago
Last measured: 3 hours ago
Next measure: 8 hours from now
* estimated time based on historical data
Historical Data
24 hours
7 days
30 days
Rating
100
80
60
40
20
Time
8:00 10:00 11:00 12:00 13:00 14:00 15:00 16:00 17:00 18:00 .....
Notifications
Description
Time of notification
1 Time to empty trash 2012-03-01 (13:57)
2 Trash was emptied by JU921 2012-02-01 (17:00)
3 No signal for 3 days 2012-01-01 (24:00)

C Databasstruktur

Relationsdiagram

För att på ett strukturerat sätt utforma en databasdesign ritades ett *Entity-Relationship Diagram*, vilket är en konceptuell beskrivning av verkligheten. Relationsdiagrammet innehåller olika entiteter som i sin tur erhåller olika egenskaper, *attributes* som kopplas samman med hjälp av relationer. På så vis framträder en tydlig bild över hur alla kopplingar hänger ihop.



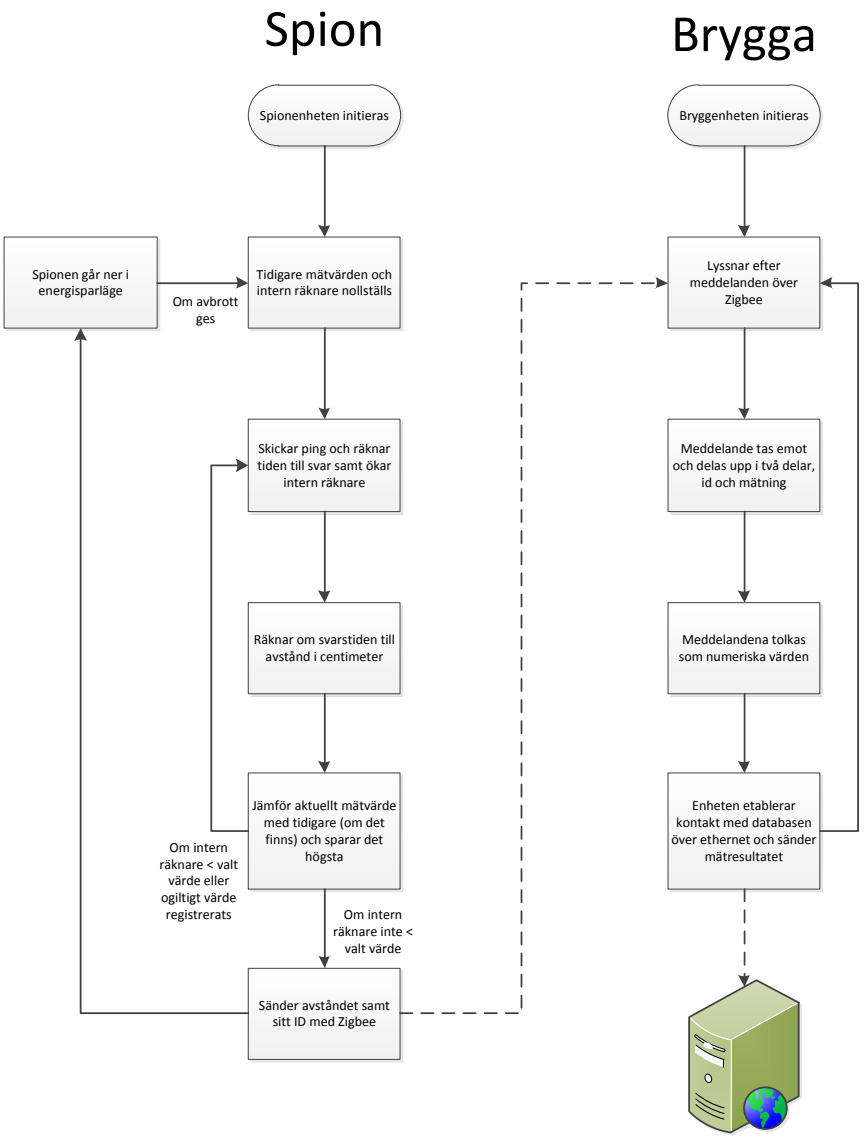
Databasschema

Relationsdiagrammet går i sin tur att översätta till ett databasschema vilket illustreras nedan.

```
MeasurementData(sensorID, tStamp, measurement)
Sensor(sensorID, positionID, maximumDepth, installationDate)
Region(regionID, regionName)
Position(positionID, positionGroupID, latitude, longitude)
PositionGroup(positionGroupID, regionID, positionName)
Notifications(sensorID, tStamp, message)
```

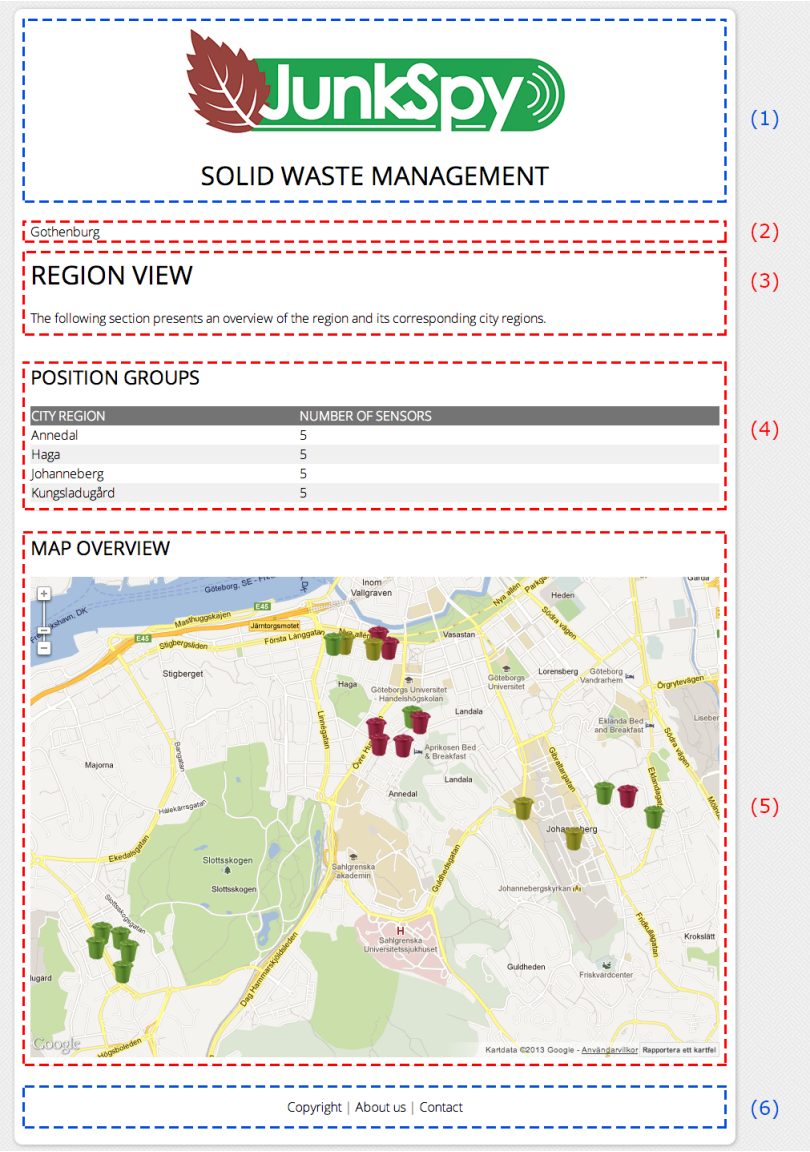
D Detaljerad hårdvara

Flödesschema för prototyp 3



E Applikationsförklaring

Stadsvyn



1: Rubrik och logotyp för webbapplikationen. Denna är statisk oberoende av vilken sida användaren befinner sig på.

2: Navigeringslänkar som byggs upp beroende på vilken vy som användaren befinner sig på.

3: Rubrik och beskrivning av vyn som användaren befinner sig på.

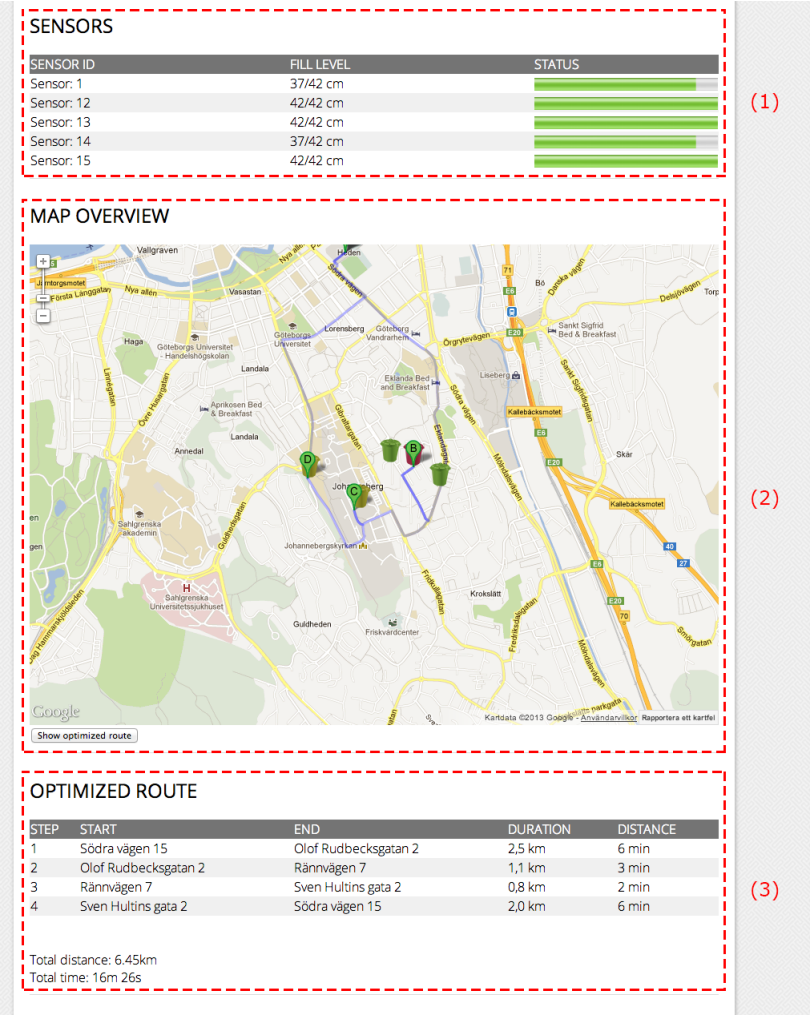
4: Lista av soptömningsområden inom den aktuella stadsvyn. Raderna är länkar som tar användaren vidare till vald områdesvy.

5: En karta som visar den geografiska representationen av stadsvyn och placering av dess soptunnor.

Soptunnorna är klickbara vilket och visar ett fönster med aktuell sopnivå och låter användaren gå vidare till sensorvyn.

6: Sidfot med länkar till tre statiska sidor om upphovsrätt, om oss samt kontaktinformation.

Områdesvyn

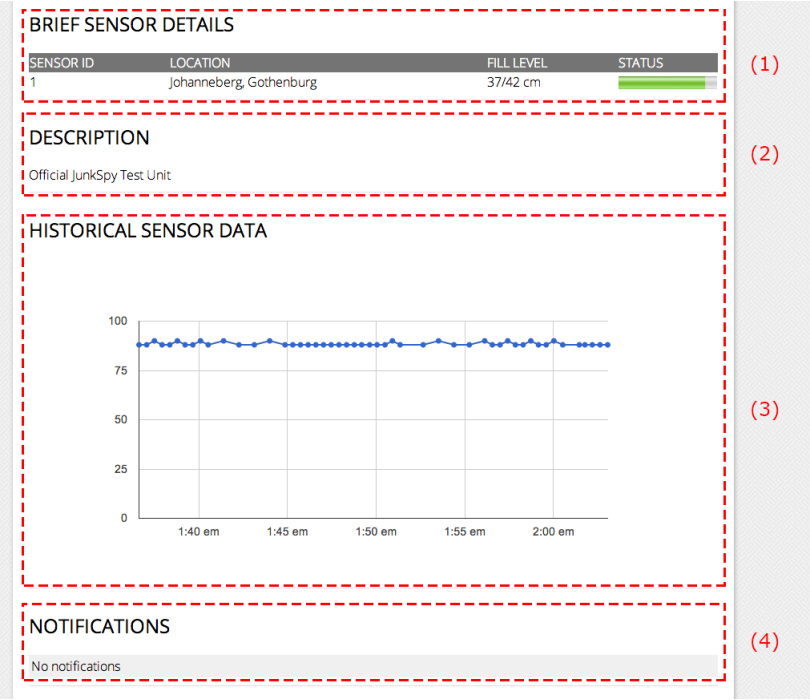


1: Lista av samtliga sensorer inom soptömningsområdet samt senast uppmätt sopnivå hos respektive sensor. Raderna är länkar som tar användaren vidare till sensorvyn.

2: En karta som visar den geografiska representationen av områdesvyn och placering av dess soptunnor. Soptunnorna är klickbara vilket och visar ett fönster med aktuell sopnivå och låter användaren gå vidare till sensorvyn.

3: Instruktioner för den optimerade resvägen med adresser, distans och restid.

Sensorvyn



1: Detaljer som berör den aktuella sensorn med senast uppmätt sopnivå.

2: Beskrivning av den aktuella sensorn.

3: Ett diagram som visar de senaste sopnivåerna det senaste dygnet.

4: Meddelanden som berör den aktuella sensorn. Innehåller varningsmeddelanden utifall soptunnan har förlorat uppkoppling mot webbapplikationen.