INF2

**Projectleden**

**Tedjan Hulshof 198110**

**Dennis Hartmann 197335**

**Justin Bakker 197068**

**Roy Heidotting 213497**

**Melvin Bos 199427**

**Tim Wennekes 204617**



**Eindverslag INF2C**

**Social Greenhouse**

Eindverslag

INF2C

**Projectleden**NaamStudentnummer E-mail

Tedjan Hulshof 198110 tedjan.hulshof@student.stenden.com

Dennis Hartmann 197335 dennis.hartmann@student.stenden.com

Justin Bakker 197068 justin.bakker@student.stenden.com

Roy Heidotting 213497 roy.heidotting@student.stenden.com

Melvin Bos 199427 melvin.bos@student.stenden.com

Tim Wennekes 204617 tim.wennekes@student.stenden.com

**In opdracht van**

Stenden Hogeschool

**Begeleiders**

Jeroen Pijpker jeroen.pijpker@stenden.com

Henk Bakker henk.bakker@stenden.com

**Module**

IPR 5.3

Datum: 16-04-2013, Emmen.

# Engelse samenvatting

In the school year of 2011 to 2012, a group of students at Stenden Hogeschool, located in Emmen, developed a largely automated greenhouse with social media integration. This school year, 2012 to 2013, a different group of students was tasked with coming up with an improved version of the product. For the improved version, the focus shifted from automation, to building a universal, accessible, open source and environmentally friendly product. This product is the *Social Greenhouse*.

The Social Greenhouse system can connect any greenhouse to Twitter to keep the owner up to date on data like temperatures and soil moisture levels. At the root of the system are the so-called *modules*: small devices which measure data, like temperatures, and wirelessly transmit it to the greenhouse’s *main unit*. The main unit, which is connected to the internet via Wi-Fi, will pass on these measurements to the *server*, where they will then be stored in a database.

Each module is powered by a solar panel. Thanks to the use of low-power components, one day of charging is enough for the module to work for approximately 4 days. For their casing, a material named *polymorph* was used. While this doesn’t yield the prettiest results without proper molding equipment, it’s a very recyclable and accessible material.

The user can control the system using a device running on Google’s open source Android operating system. The team built an application which provides the following features:

* Connect to the main unit, via USB, to set up its connection to the server
* Register modules to the server by holding a module to the back of the tablet
* View the latest measurement of each module
* Link a Twitter account to the greenhouse
* Configure, per module, when the system should post messages to Twitter; the so-called triggers. An example might be, “if the soil moisture is below 70%, post ‘The tomatoes need water!’”

The user may install the application on different devices, e.g. all modules present on their tablet will be synchronized with their phone automatically.

Rather than attempting to pack as many features as possible into the initial release, the team aimed to build a solid product that is easy for the open source community to build upon. The initial release only supports Twitter for social media integration, and temperature and soil moisture for modules. However, a solid base has been built for integrating additional social media and different types of modules.

# Versiebeheer

|  |  |  |
| --- | --- | --- |
| **Versie** | **Revisiedatum** | **Toelichting** |
| 0.1 | 16-04-2013 | Opzet |
| 0.5 | 19-06-2013 | Concept |
| 0.8 | 20-06-2013 | Revisie concept #1 |
| 1.0 | 21-06-2013 | Revisie concept #2 |

Inhoudsopgave

[0. Engelse samenvatting 3](#_Toc359587145)

[1. Versiebeheer 4](#_Toc359587146)

[2. Algemene inleiding 6](#_Toc359587147)

[2.1 Opdrachtomschrijving 6](#_Toc359587148)

[2.2 Projectgroep en werksituatie 6](#_Toc359587149)

[2.3 Organisatiestructuur 7](#_Toc359587150)

[2.4 Samenstelling van de projectorganisatie 7](#_Toc359587153)

[3 Productdoelstelling 8](#_Toc359587154)

[4 Methoden, technieken en werkzaamheden 9](#_Toc359587155)

[5 Het product 10](#_Toc359587156)

[5.1 Samenvatting 10](#_Toc359587157)

[5.2 Modules 11](#_Toc359587158)

[5.2.1 Het stuurprogramma 11](#_Toc359587159)

[5.2.2 Draadloze communicatie 12](#_Toc359587160)

[5.2.3 Energie 12](#_Toc359587161)

[5.2.4 NFC tag 13](#_Toc359587162)

[5.2.5 Printplaat 13](#_Toc359587163)

[5.2.6 Behuizing 16](#_Toc359587164)

[5.3 Hoofdunit 17](#_Toc359587165)

[5.3.1 Stuurprogramma 17](#_Toc359587166)

[5.3.2 Behuizing 18](#_Toc359587167)

[5.4 Server 19](#_Toc359587168)

[5.4.1 Java service 19](#_Toc359587169)

[5.4.2 Database 19](#_Toc359587170)

[5.4.2 Triggers 20](#_Toc359587171)

[5.4.3 Twitteren 21](#_Toc359587172)

[5.4.4 Webservice 22](#_Toc359587173)

[5.4.5 Specificatie webservice 23](#_Toc359587174)

[5.5 Android applicatie 25](#_Toc359587175)

[5.5.1 Dynamische layout 25](#_Toc359587176)

[5.5.2 Synchronisatie modules 25](#_Toc359587177)

[5.5.3 NFC verbinding 26](#_Toc359587178)

[5.5.4 Uitbreiden triggers en soorten modules 26](#_Toc359587179)

[6 Discussie en conclusies 27](#_Toc359587180)

[7 Suggesties voor vervolg onderzoek / project 28](#_Toc359587181)

[7.1 Behuizing modules 28](#_Toc359587182)

[7.2 Stroomverbruik 28](#_Toc359587183)

[7.3 Meerdere hoofdunits 29](#_Toc359587184)

[7.4 Meer soorten modules 29](#_Toc359587185)

[7.5 Efficiëntere synchronisatie Android applicatie 29](#_Toc359587186)

[8 Bijlagen 30](#_Toc359587187)

# Algemene inleiding

## 2.1 Opdrachtomschrijving

In het kader van een leerproject gaat de projectgroep een universeel en duurzaam systeem ontwikkelen dat een kas integreert met sociale media; project Social Greenhouse. Het gewenste eindproduct is een kas die via sociale media updates geeft over bijvoorbeeld grondvochtigheid en weersomstandigheden.

Dit project is een nieuwe kijk op een project dat een jaar eerder is uitgevoerd. In dat project lag de focus met name op de automatisering van een kas, die daarnaast met sociale media geïntegreerd werd. In het nieuwe project bleef het aspect van sociale media, maar is de focus verlegd van automatisering naar monitoring; zo kon er een toegankelijk product ontwikkeld worden dat toegepast kan worden op elke bestaande kas. Daarnaast was er de vraag om aandacht te schenken aan duurzaamheid en de open source community.

## 2.2 Projectgroep en werksituatie

Aan dit project hebben zes projectleden gewerkt, elk minimaal 20 uur per week. De projectgroep bestaat uit een projectleider, planner, controller, kwaliteitsborger, archiefbeheerder/notulist, lokaalbeheerder en een woordvoerder. Ieder project lid heeft ook een eigen taak binnen het project zelf.

## 2.3 Organisatiestructuur

Opdrachtgever

Projectleider

## 

## 

Projectteam

Archief beheerder

Planner

Kwaliteitsborger

## 2.4 Samenstelling van de projectorganisatie

De opdrachtgever:

|  |  |  |  |
| --- | --- | --- | --- |
| Naam | Rol | E-mail | Beschikbaarheid |
| Jeroen Pijpker | Opdrachtgever | jeroen.pijpker@stenden.com | Ma, di, do, vr van 12:30 tot 13:15 |

Het projectteam bestaat uit:

|  |  |  |  |
| --- | --- | --- | --- |
| Naam | Rol | E-mail | Beschikbaarheid |
| Roy Heidotting | Projectleider | royheidotting13@msn.com | Ma,Di,Wo,Do, Vrij van 8:30 tot 16:00 |
| Tedjan Hulshof | Notulist / archiefbeheer / projectlid | tedjan\_3@hotmail.com | Ma,Di,Wo,Do, Vrij van 8:30 tot 16:00 |
| Dennis Hartmann | Timemanager / planner / projectlid | dennishartmann31@hotmail.com | Ma,Di,Wo,Do, Vrij van 8:30 tot 16:00 |
| Justin Bakker | Woordvoerder / projectlid | justin\_1993@live.nl | Ma,Di,Wo,Do, Vrij van 8:30 tot 16:00 |
| Tim Wennekes | Lokaalmanager / projectlid | timwennekes@gmail.com | Ma,Di,Wo,Do, Vrij van 8:30 tot 16:00 |
| Melvin Bos | Kwaliteitscontroleur / projectlid | melvin.m.bos@gmail.com | Ma,Di,Wo,Do, Vrij van 8:30 tot 16:00 |

# Productdoelstelling

Het Social Greenhouse moet de gebruiker ondersteunen met het onderhouden van gewassen in een (kweek)kas. Daarnaast is het Social Greenhouse toegankelijk voor de opensource community, dit zorgt ervoor dat het project makkelijk opnieuw kan worden uitgevoerd en makkelijk uit te breiden is. Duurzaamheid was ook een eis van het Social Greenhouse, daarom maakt het Social Greenhouse gebruik van een duurzaam materiaal genaamd polyform. Het Social Greenhouse is daarnaast ook modulair opgebouwd, ook dit was een eis. Ook de stroomvoorziening moest aan gedacht worden, daarom beschikt het Social Greenhouse over een eigen zonnepaneel zodat het zichzelf van stroom kan voorzien. Eén van de belangrijkste doelen was om het Social Greenhouse te integreren met social media, hiervoor maakt het Social Greenhouse gebruik van de social media website Twitter. Tot slot moest er een applicatie worden ontwikkeld die het gebruik van het Social Greenhouse vergemakkelijkt, dit moest zowel draadloos als bekabeld kunnen.

# Methoden, technieken en werkzaamheden

Binnen dit project is er gewerkt met scrum. Dit heeft als voordeel:

* Het verhogen van de effectiviteit van het team
* Het bewaken van de vooruitgang van het team
* Het oplossen van blokkades
* Het bewaken van de projectvoortgang
* Het in kaart brengen en minimaliseren van risico's

[](http://commons.wikimedia.org/wiki/File:NL-scrum.jpg)

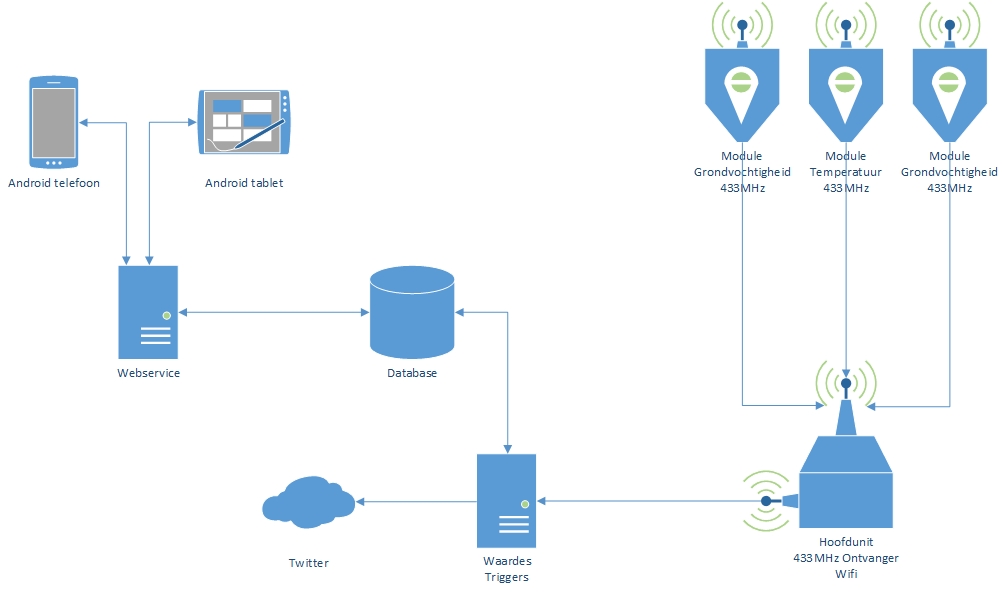
Scrum is een mooie methode om taken duidelijk te krijgen. Hiermee kun je taken heel mooi afscheiden en afbaken. En als iets werkt hoef je er ook niet meer naar om te kijken. Het was in het begin wel even wennen om met scrum te werken, aangezien er hier nog nooit mee gewerkt is. Wel was het moeilijk om een tijd te plannen voor een bepaalde taak en dan met name de software. Er is toch gebleken dat hier meer tijd in ging zitten dan in het begin gepland was. Misschien dat er volgende keer een kleine vergadering kunnen houden met een uurtje onderzoek naar de duur van een specifieke taak.

# Het product

## 5.1 Samenvatting

In een kas kunnen meerdere **modules** worden geplaatst. Elke module heeft een sensor en staat draadloos in verbinding met de **hoofdunit**. De hoofdunit ontvangt draadloos metingen van de modules en stuurt deze via Wi-Fi door naar de **Java server**. Op de Java server worden die metingen opgeslagen in de **database** en, als een meting aan een bepaalde voorwaarde voldoet, berichten naar sociale media verstuurd.

Met de **Android applicatie** kan de gebruiker de meest recente sensormetingen inzien en de voorwaarden voor het plaatsen van berichten op sociale media instellen. Daarnaast wordt de Android applicatie gebruikt om sociale media accounts te koppelen aan de server, en de hoofdunit via USB te voorzien van netwerkinstellingen .

****

## 5.2 Modules

### 5.2.1 Het stuurprogramma

Het stuurprogramma van een modules vervult drie taken; het uitlezen van een sensor, het verwerken van de ruwe data naar een concrete waarde, en het opsturen van deze waarde. Dit gebeurt één keer per minuut.   
  
Zodra het programma start initialiseert het alle registers[[1]](#footnote-1). De juiste ADC poort wordt geselecteerd en aangezet. Vervolgens wordt de timer aangezet; eerst wordt de prescaler[[2]](#footnote-2)gezet, gevolgd door de startpositie van de timer. En uiteindelijk wordt er een interrupt[[3]](#footnote-3) gezet op de timer overflow[[4]](#footnote-4).

ADMUX = 0x00;

ADCSRA = \_BV(ADPS1)|\_BV(ADPS0)|\_BV(ADEN);

TCCR0B |= (1<<CS01); // de prescaling is 8

TCNT0 = 0x9F; // start timer met 204

TIMSK0 |=1<<TOIE0; // interrupt timer0 overflow

Daarna komt het programma terecht in een oneindige while lus, die er als volgt uitziet (voor de volledige code, zie Module\_Grondvochtigheid\_v1.0.C).

while(1)

{

// read ADC value

// calculate value

// send the data

// wait for some time

}

Als eerste stap in de lus wordt de sensor waarde uitgelezen. Voor dit uitlezen wordt gebruik gemaakt van de ADC (Analog Digital Converter). Deze zet een analoog signaal om in een digitaal signaal. Op dit digitale signaal is een meting uit te voeren.

Deze meting wordt uitgevoerd op de sensor van de module. Een bepaald voltage van de invoerspanning wordt niet doorgelaten door de sensor. Het voltage dat wel wordt doorgelaten, wordt gemeten. Daarover wordt een berekening uitgevoerd, zodat de juiste waarde verstuurd kan worden.

### 5.2.2 Draadloze communicatie

Dit oversturen gebeurt via een 433 MHz zender. Hiervoor is gekozen omdat dit een vrij te gebruiken golflengte is. Hiervoor is het volgende protocol gedefinieerd:

<checksum><moduleID>:<waarde>-<checksum><checksum>

De eerste checksum bestaat uit “JAB”. Dit is het begin van iedere bericht. Daarna komt de ID van de betreffende module en na de dubbele punt komt de gemeten waarde. Na het streepje komt de checksum die berekend wordt over de module ID en de waarde (zie bijlage ?? voor uitleg over de checksum). En als laatste wordt er een “z” achter het bericht gezet.

Als het bijvoorbeeld 22.7 graden is wordt het volgende bericht verstuurd: “JAB10000002:22.7-!z”.

Voor het daadwerkelijke versturen wordt gebruik gemaakt van drie functies. Een voor het versturen van de eerste checksum (“JAB”). Hierover hoeft de tweede checksum niet worden berekend. De volgende functie wordt gebruikt voor het versturen van de data waarover de tweede chekcum wordt berekend. De data wordt om de 15 seconden verstuurd. Hierdoor is het zeker dat een module een juiste waarde over zendt naar de hoofdunit.

De gegevens moeten worden overgestuurd op een baudrate van 1200. Met deze baudrate wordt het bericht verstuurd met de minste ruis. 1200 Baud komt overeen met 775us. Dit is gemeten met een scope. Vanuit hier is verder gerekend om de timer juist in te stellen voor het oversturen van de gegevens.

Uit onderstaande berekening volgt de waarde die ingesteld moet worden om de timer overflow op het juist moment te laten uitvoeren.

256-(1000000/(775\*8))=96

Hieruit volgt dat de prescaler moet worden ingesteld op 8. De standaard overflow van de timer is 256.

### 5.2.3 Energie

Doordat de module zelfvoorzienend moet zijn qua energie is het van belang om zuinig en efficiënt om te gaan met de stroom. Zo krijgt de 433MHz transmitter alleen stroom tijdens het verzenden en wordt deze daarna weer uitgezet. Dit gebeurt door middel van een transistorschakeling. Ook de ADC krijgt alleen stroom als deze actief is, maar dit gebeurt door het rechtstreeks aansturen van een bepaalde poort. Dit is gedaan omdat de transistor ook stroom trekt, waardoor de ADC meting niet juist zou verlopen.

### 5.2.4 NFC tag

Elke module is voorzien van een NFC tag waarin de ID en het type van de sensor staan opgeslagen, als volgt:

<ID>:<SENSOR\_TYPE>  
Bijvoorbeeld: “10000002:soil\_moisture”

Momenteel zijn er twee sensortypen gedefinieerd:

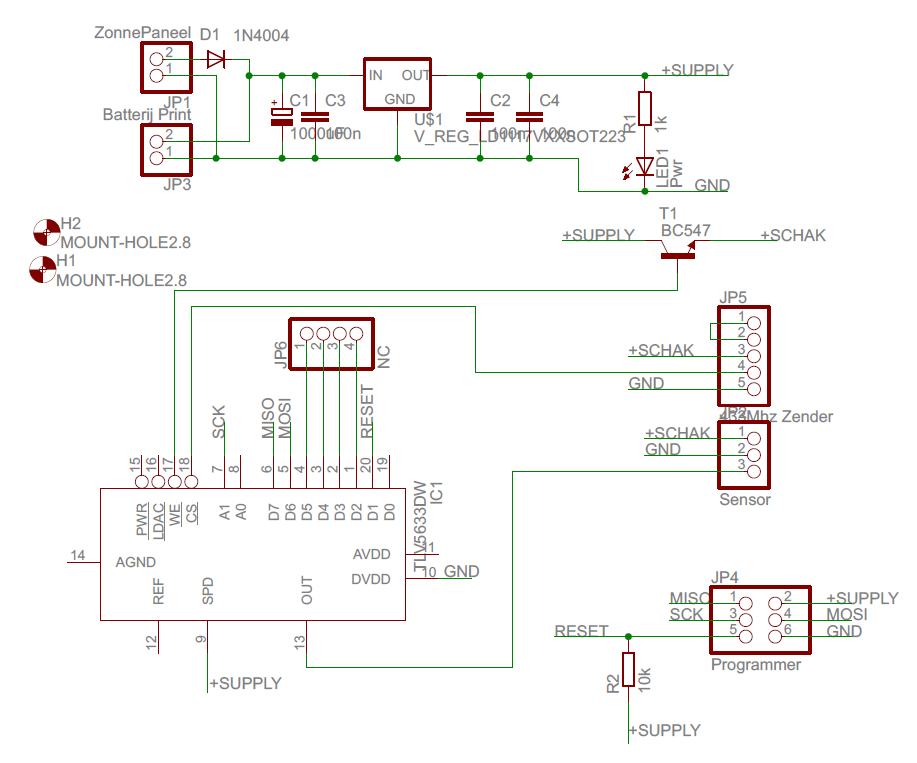
* soil\_moisture
* temperature

Daarnaast is de NFC tag van een MIME-type en naam voorzien. Hiermee kunnen apparaten het juiste onderdeel van het Social Greenhouse applicatie starten zodra een tag wordt gedetecteerd. Zie voor meer informatie de specificatie van het NFC Data Exchange Format (NDEF)[[5]](#footnote-5) en een voorbeeld[[6]](#footnote-6) van een toepassing in Android.

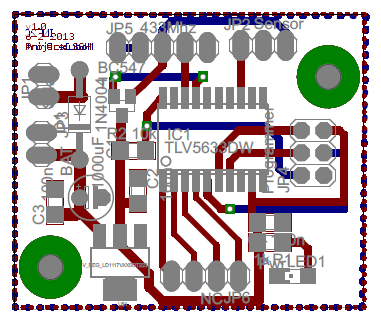
### 5.2.5 Printplaat

De module moet voldoen aan speciale wensen met betrekking tot het stroomverbruik. Hiervoor zijn er specifieke onderdelen gekozen, voor verdere toelichting van de keuze wordt verwezen naar ‘Bijlage 1’. Om alle onderdelen van stroom te voorzien is er gekozen voor een spanningsregulator van 3.3V en alle componenten kunnen met deze spanning overweg.

Daarom is er voor gekozen om een print te ontwikkelen. Dit heeft als voordeel dat de print naar eigen wensen gemaakt kan worden om zo onnodige stroom verbruikers uit te sluiten. Er is eerst begonnen met het tekenen van een schema. Hier zijn alle onderdelen stap voor stap uitgewerkt.

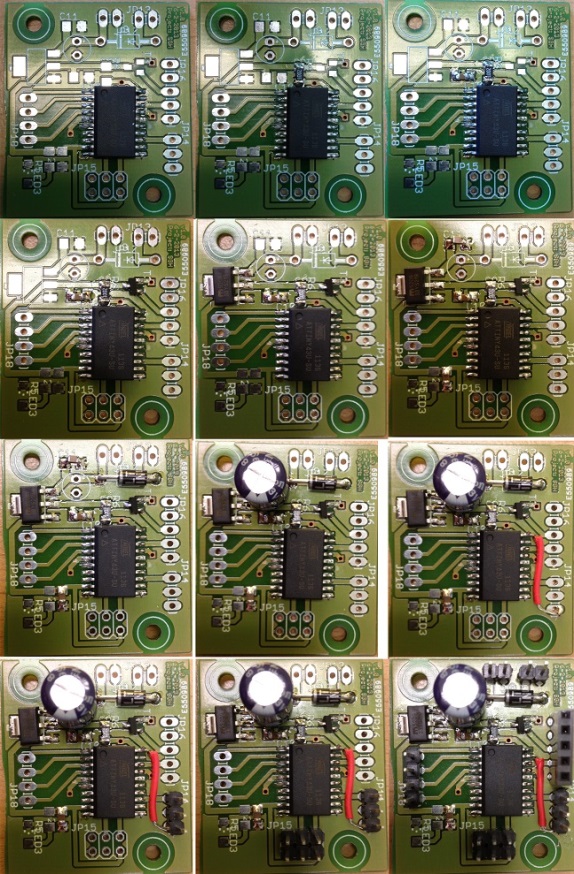


Vervolgens is dit schema uitgewerkt naar een print. Er is begonnen met het tekenen van de voeding om zo alle secties bij elkaar te houden.

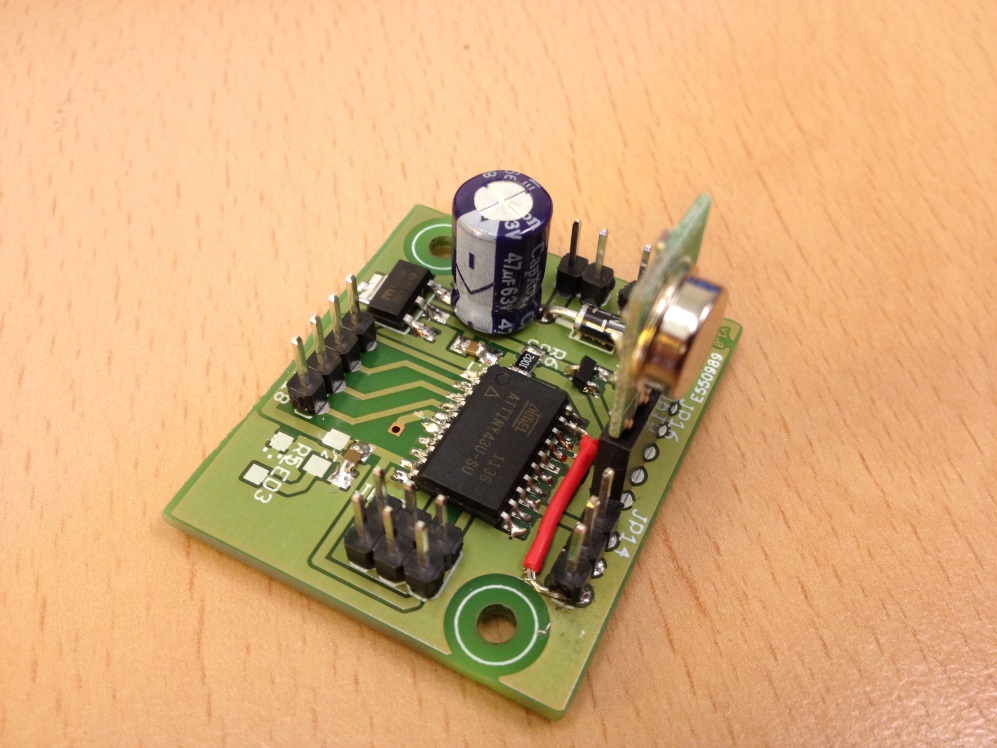


Het linker gedeelte van deze print is de voeding. In het midden is de microcontroller te zien en rechts daarvan zitten 6 pinnen om de software te uploaden. Boven aan bevinden zich de aansluitingen voor de sensor en de zender en onderaan zitten 4 pinnen voor test doeleinden of toekomstig gebruik.

Nadat deze print is gemaakt door het bedrijf euro-circuits, is er begonnen met het solderen er van. Hieronder zie je een afbeelding van het solderen er van.



Na dit solderen is de print stap voor stap getest op functies. Na dat dit allemaal goed was bevonden is de functionele software er op gezet.



### C:\Users\Tedjan\Dropbox\Social Greenhouse\Periode 4 - Testen Afronden\Project\Project documentatie\foto\20130604_105211.jpg5.2.6 Behuizing

Voor de behuizing is er gebruik gemaakt van polymorph[[7]](#footnote-7) plastic. Dit soort plastic is niet giftig, bio recyclebaar en heeft zijn smeltpunt heeft rond 60°C. Als het in kokend water wordt gehouden smelt de polymorph waardoor deze zacht wordt en te bewerken is.

De behuizing is gemaakt door het polymorph om een mal heen te drukken. Voor een handleiding betreffende het maken van de behuizing wordt verwezen naar de bijlage “Handleiding module”.

## 5.3 Hoofdunit

De hoofdunit dient als simpel doorgeefluik tussen de modules en de server. De waardes die de modules versturen via 433Mhz, worden ontvangen en doorgestuurd naar de server via Wi-Fi. Door de modules niet direct met het internet te verbinden worden stroom en kosten bespaard en gebruikersgemak verhoogd; de verbinding hoeft zo slechts eenmalig te worden ingesteld.

### 5.3.1 Stuurprogramma

Het stuurprogramma van de server heeft twee hoofdtaken: het ontvangen van inkomende waarden van modules, en het versturen van deze waardes naar de server.

Waardes worden tijdelijk opgeslagen in het geheugen. Elke minuut worden de waardes in het geheugen via Wi-Fi verstuurd naar de server (mits er een Wi-Fi verbinding actief is), waarna ze uit het geheugen worden verwijderd. Als een module een tweede bericht stuurt voordat het eerste bericht is verstuurd naar de server, wordt het eerste bericht in het geheugen vervangen.

#### 5.3.1.1 Ontvangen 433Mhz

De 433Mhz ontvanger zit aangesloten op een seriële poort, welke constant wordt uitgelezen op inkomende gegevens.

Zoals te lezen was in het hoofdstuk module wordt er een bericht uitgezonden, maar omdat 433Mhz een vrije frequentie is met een hoge kans op storing, wordt er een drietal controles uitgevoerd op de integriteit van het bericht. Als één van deze controles faalt, wordt het bericht gezien als corrupt en dus vergeten.

**Controle 1: lengte**  
Het bericht moet precies 12 karakters lang zijn  
  
**Controle 2: verplichte karakters**  
Het bericht moet de karakters “:” en “-” bevatten.  
  
**Controle 3: checksum**  
Het laatste karakter, de zogenaamde checksum, is het resultaat van een optelsom van de decimale waarde van elk karakter in het bericht. Deze wordt door de module berekend en meegestuurd, waarna de hoofdunit deze opnieuw berekent en vergelijkt: als het resultaat verschilt, faalt de controle.

#### 5.3.1.2 Wi-Fi

De hoofdunit heeft een Wi-Fi chip om deze via een draadloos netwerk met de server te verbinden. Het instellen van het Wi-Fi netwerk en het IP adres van de server wordt gedaan via de seriële verbinding. Deze instellingen worden opgeslagen en blijven ook na een herstart bewaard.

Voor het instellen bestaan twee commando’s:  
  
**Connect**connect <SSID>(+<WPA>+<SLEUTEL>)   
Dit commando wordt gebruikt om te verbinden met een Wi-Fi netwerk. Voorbeelden:  
“connect Stenden+OPEN+”(verbindt met een open netwerk)  
“connect Thuis+WPA+Wachtwoord” (verbindt met een netwerk beveiligd met WPA)  
  
**Server**server <IP>:<PORT>  
Dit commando wordt gebruikt om te verbinden met een server. Voorbeeld:  
“server 123.456.7.8:5000”  
  
Mocht er tijdens het versturen geen actieve Wi-Fi verbinding zijn, dan probeert de hoofdunit eerst opnieuw te verbinden. Lukt dit niet, dan wordt dit pas de volgende minuut opnieuw geprobeerd.

### 5.3.2 Behuizing

De behuizing van de hoofdunit in uitgesneden met een lasersnijder. Deze snijdt een plastic plaat in losse fragmenten, welke daarna in model geplakt kunnen worden. Dit uitsnijden is gedaan bij Stenden MyConcept Emmen.

De behuizing is opgebouwd uit verschillende delen. Om te beginnen is er een bak gemaakt waarin alle componenten zich bevinden. Hierin is een dubbele bodem gemaakt waarop alle elektronica vastgemaakt zit. Onder deze dubbele bodem is een antenne weggewerkt voor het ontvangen van de data.

Verder is er een deksel gemaakt, welke spatwaterdicht is gemaakt doormiddel van een rubber strip. De naden zijn gecontroleerd op lekkages; waar deze zijn gevonden, zijn de naden extra afgekit.



## 5.4 Server

De server is het hart van het systeem en heeft de volgende verantwoordelijkheden:

* Het opslaan van binnenkomende gegevens van de hoofdunit
* Het aanbieden van de module gegevens via een webservice
* Het ontvangen van wijzigingen in het systeem,van bijvoorbeeld de Android applicatie
* De uitvoering van triggers

### 5.4.1 Java service

Voor toelichting op de keuze van VertX[[8]](#footnote-8), verdere onderbouwing kan worden gevonden in Bijlage 1. Er wordt gebruik gemaakt van een asynchrone socket server. Dit maakt de server stabiel en schaalbaar.

Zoals al in het hoofdstuk Hoofdunit beschreven werd ontvangt de hoofdunit een opslag commando op data op te slaan. Dit gaat om onderstaande commando:

**#save|10000002,23.2$  
#<Command>|<SensorID>,<SensorValue>$**  
De server zal dit commando verwerken en controleren of de opgegeven module wel geregistreerd is in de database. Mocht dit het geval zijn dat update de server zijn waarden.

### 5.4.2 Database

Als databasesysteem is er gekozen voor MySQL, met de volgende redenen:

* MySQL voldoet aan de eisen voor de database en biedt voldoende functionaliteit voor eventuele doorgroei
* MySQL is gratis en cross-platform
* Er is veel documentatie voor beschikbaar

De database ziet er als volgt uit:



Het tabel module is verantwoordelijk voor het bijhouden van de modules gekoppeld aan het Social Greenhouse. Om een logboek van waarden bij te houden wordt er gebruik gemaakt van de tabel State. Deze houdt de status van een sensor op een bepaald tijdstip bij.

Ook zijn er een aantal instellingen voor het Social Greenhouse. Deze worden los bijgehouden in de Settings tabel. Hierin staan bijvoorbeeld de Twitter gegevens.

### 5.4.2 Triggers

Om een bepaalde actie uit te voeren wordt gebruik gemaakt van een trigger. Bij het Social Greenhouse server wordt de trigger opgebouwd uit een aantal lagen conform Decorated Command Pattern.**[[9]](#footnote-9)** Dit pattern wordt vooral bij commando gebaseerde programma’s, zoals de server. Ook is het door dit design patroon mogelijk om bepaalde acties te beveiligen. Dit wordt op de server gebruikt om pas te Twitteren wanneer de waarde door een aantal poortjes komt.



// "tweet "Turn up the heat!""

Observer observer = new Tweet("Turn up the heat!");

// "every 3 hours"

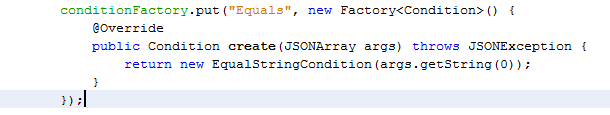
observer = new CooldownDecorator(3 \* 60 \* 60, observer);

// "when temperature is below 20"

Condition condition = new LessThanCondition(20);

observer = new FilterDecorator(condition, observer);

Deze condities en acties worden door middel van het factory pattern aangemaakt. Hierdoor is het mogelijk om gemakkelijk extra condities en acties toe te voegen. Zo’n pattern ziet er als volgt uit:



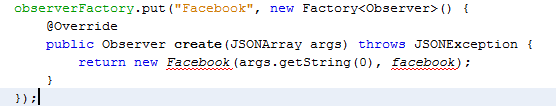
Deze maakt gebruik van de volgende interface



Met deze factories kan de parser gemakkelijk worden uitgebreid met nieuwe condities of acties. Hierdoor hoeft dan niet meer in de parser zelf worden gewerkt.

#### 5.3.2.1 Uitbreiden

Mocht de server later bijvoorbeeld worden uitgebreid met Facebook, kan dit door een nieuwe action factory toe te voegen, dit ziet er dan als volgt uit:



Nu kan er verwezen worden naar een Facebook object die de mogelijkheid geeft om een bericht te plaatsen.

### 5.4.3 Twitteren

Wanneer een ontvangen waarde voldoet aan een waarde van een geïnstalleerde trigger zal de server, mits de server langer dan een uur draait, een Twitter bericht verspreiden. Dit doet hij doormiddel van de tritter4j bibliotheek voor Java. Deze bibliotheek kan door middel van de ontwikkelaars API van Twitter berichten plaatsen namens een gebruiker.

Deze berichten zijn via de Android applicatie in te stellen. Het bericht zal gestuurd bijvoorbeeld elk uur gestuurd worden wanneer de waarde nog voldoet aan de trigger. Het bericht zal standaard worden voorzien van een tijd omdat de Twitter bibliotheek niet toestaat dat hetzelfde bericht binnen 48 uur opnieuw gestuurd wordt.

### 5.4.4 Webservice

Voor de synchronisatie tussen tablet en server wordt voor het ophalen en aanpassen van gegevens een webservice gebruikt. Deze webservice is geschreven in PHP met een open source bibliotheek genaamd Silex[[10]](#footnote-10). De keuze voor Silex is gemaakt doordat het een bijzonder simpele API gebouwd op een robuste kern biedt.

Voor de gegevensuitwisseling wordt gebruik gemaakt van JSON[[11]](#footnote-11). De keuze voor JSON is gebaseerd op zijn goede ‘mapping’ met moderne programmeertalen. In tegenstelling tot bijvoorbeeld XML heeft JSON datastructuren als strings, arrays, booleans en null ingebouwd. Daarnaast zijn er veel bibliotheken voor beschikbaar.

### 5.4.5 Specificatie webservice

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Actie | Method | URL | Response code (goed/fout) | Content-Type |
| Ophalen modules | GET | {webservice\_root}/index.php/modules | 200/500 |  |
| Opslaan alle modules | POST | {webservice\_root}/index.php/modules | 201/500 | Application/json |
| Verwijderen module | DELETE | {webservice\_root}/index.php/module/{SerialNo} | 204/500 |  |
| Ophalen instellingen | GET | {webservice\_root}/index.php/settings | 200/500 |  |
| Opslaan instellingen | POST | {webservice\_root}/index.php/settings | 200/500 | Application/json |
| Opslaan triggerdata | POST | {webservice\_root}/index.php/triggers | 201/500 | Application/json |

Alle GET methodes zullen een JSON array retourneren. De opbouw van deze array is op te maken in de code van de webservice. Zo ziet het antwoord van de /settings er als volgt uit:

[

{

“Name”:”AccessToken”,

“Value”:”392cm0230cm823098c0928m0cm3”

},

{

“Name”:”AccessTokenSectret”,

“Value”:”cm5o234m924092mu4c092”

},

{

“Name”:”ConsumerKey”,

“Value”:”c2n4782m3o423osjbajdm4o2”

},

{

“Name”:”ConsumerSecret”,

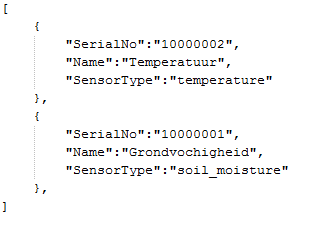
“Value”:”i3p2u23u9u23ou23oo”

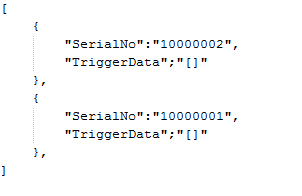
}

]

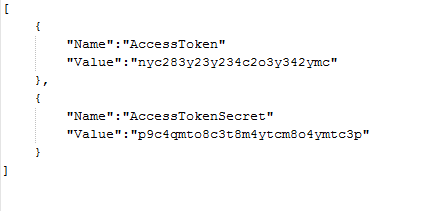
Bij het gebruiken van een POST is het van belang dat de Content-Type staat op application/json. Deze acties zullen dan om een stukje JSON vragen.

**Opslaan alle modules**

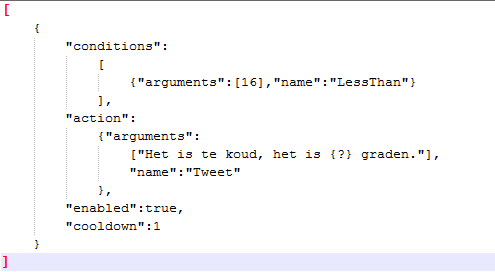
 **Opslaan triggerdata**



**Opslaan settings**



**TriggerData**

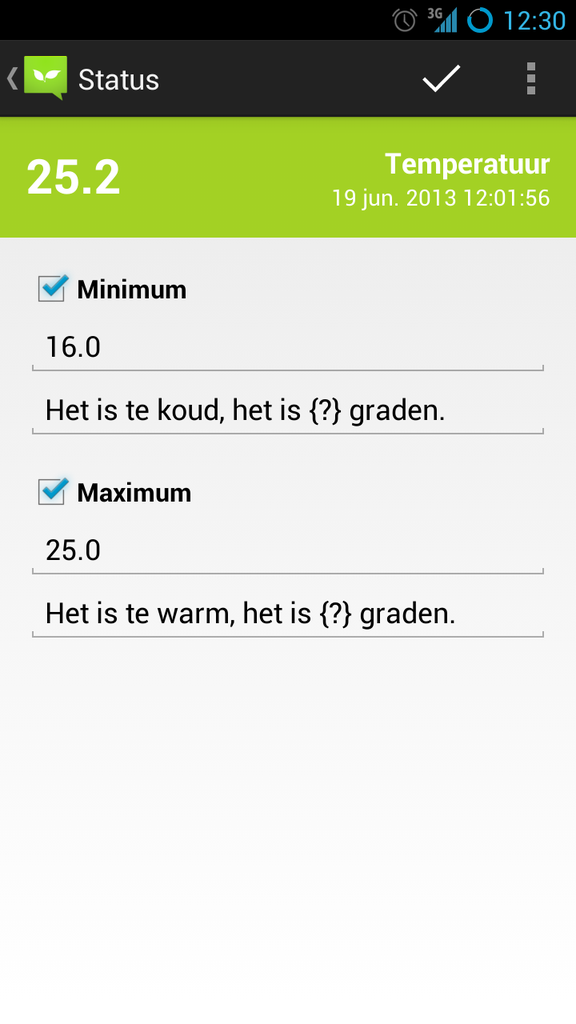
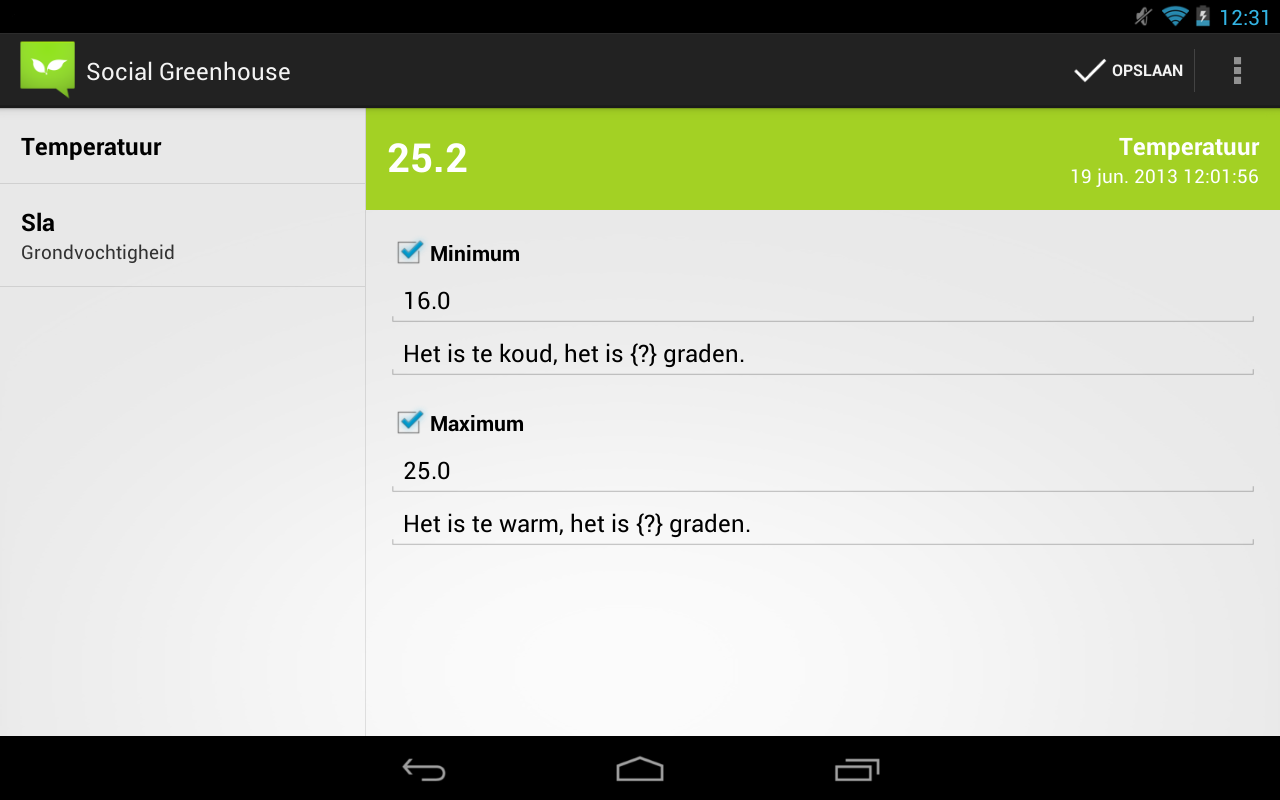


## 5.5 Android applicatie

### 5.5.1 Dynamische layout

Om de applicatie op zowel grote als kleine schermformaten goed weer te geven, is er gebruikt gemaakt van de volgende technologieën:

* **Fragments**<http://developer.android.com/guide/components/fragments.html>  
  Het opdelen van de user interface in losse stukken die zowel los van elkaar als samen kunnen worden weergegeven.
* **Flexibele layouts**

<http://developer.android.com/guide/topics/ui/layout/relative.html>  
<http://developer.android.com/guide/topics/ui/layout/linear.html>  
UI-elementen worden dynamisch gepositioneerd op basis van de randen van het scherm en/of andere elementen.

### 5.5.2 Synchronisatie modules

Alle relevante gegevens worden lokaal opgeslagen in een SQLite tabel die er als volgt uitziet:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Modules** | | | | | |
| **ID** | **SensorType** | **Name** | **Value** | **ValueTimestamp** | **TriggerData** |
| INT (PK) | TEXT | TEXT | TEKST | INT | TEKST |

Deze gegevens worden beschikbaar gemaakt via een ContentProvider[[12]](#footnote-12). Gecombineerd met de CursorLoader[[13]](#footnote-13) maakt dit het makkelijk om live (automatisch bijgewerkte) waardes vanuit de database weer te geven. De tabel wordt, als het scherm van het apparaat aanstaat, om de minuut bijgewerkt met de meest recente gegevens van de server.

Voor meer details wordt verwezen naar de klasse ModuleContract in de namespace com.socialgreenhouse.database.

Bij het toevoegen van modules, het verwijderen van modules, en het aanpassen van triggers, wordt er data naar de server verstuurd. De resulterende veranderingen worden niet direct lokaal gemaakt; in plaats daarvan wordt de gehele tabel weer bijgewerkt met de nieuwe gegevens van de server. Dit versimpelt de code aanzienlijk.

### 5.5.3 NFC verbinding

Als een module wordt gedetecteerd via NFC, komt er automatisch een scherm naar voren waarin de gebruiker optioneel een aangepaste naam aan de trigger toe kan wijzen. De gebruiker kan vervolgens kiezen om de module op te slaan; deze wordt dan geregistreerd aan de server. Als de module al is geregistreerd wordt deze simpelweg overschreven.  
  
Zie 5.3.1 voor meer informatie over de NFC tags.

### 5.5.4 Uitbreiden triggers en soorten modules

Voor het toevoegen van een nieuw soort module moet het type geregistreerd worden in de klasse com.socialgreenhouse.database.SensorType.

 Per sensortype kan een aparte GUI voor het instellen van triggers worden weergegeven, middels de TriggerDataAdapter interface in de namespace com.socialgreenhouse.triggers.

Per sensortype moet er in ModuleDetailFragment in de namespace com.socialgreenhouse een adapter worden geregistreerd. Deze adapter zet de in JSON opgeslagen triggergegevens om naar een GUI, en later, de inhoud van de GUI weer om naar een JSON array.

Dit concept is vergelijkbaar met de SimpleCursorAdapter[[14]](#footnote-14) van Android.

Zie voor meer details het commentaar in de source code.

Als er een SensorType en TriggerDataAdapter zijn geregistreerd, kan de nieuwe soort module met de applicatie gebruikt worden.

# Discussie en conclusies

Het project heeft over het algemeen goed uitgepakt. De huidige opbouw maakt het mogelijk om een willekeurige kas te voorzien van dit monitor systeem. Doordat alle units draadloos zijn kost dit relatief weinig werk.

Aan de opgegeven opdracht is in ruime mate voldaan. Ik het kader van Sociale media is het Social Greenhouse een volwaardig product. Er is een actieve koppeling met het Twitter-account van de gebruiker; de gebruiker kan zelf instellen wat de kas op Twitter plaatst, en wanneer.

Duurzaamheid is goed terug te zien in het product. In elk aspect van het product is rekening gehouden met duurzaamheid. Zo zijn de microcontrollers binnen de modules voorzien van enkel RoHS[[15]](#footnote-15) onderdelen. De voeding van deze controllers is verzorgd door een li-ion accu; dit is duurzamer dan een standaard lood accu. De behuizingen van de modules zijn bijzonder goed te recyclen. De modules zijn door het gebruik van een zonnepaneel zelfvoorzienend. De hoofdunit zou in het kader van duurzaamheid echter nog verbeterd kunnen worden, door eenzelfde soort laadcircuit als de modules te gebruiken.

Het Social Greenhouse is modulair, open source, gebouwd op open source libraries – kortom, ontworpen om door te groeien. Het complete project wordt overgedragen aan de open source community.

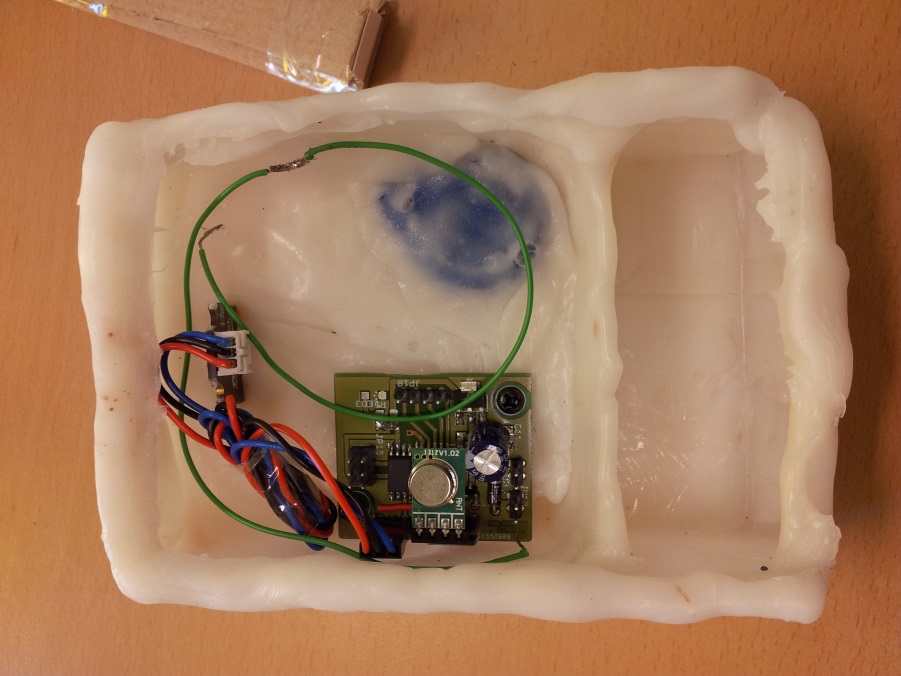
Het doel was een duurzaam, open source systeem te ontwikkelen om een kas met sociale media te integreren, en dit is geslaagd. Het project mag dan ook als een succes gezien worden.

# Suggesties voor vervolg onderzoek / project

Tijdens dit project zijn er verschillende werkzaamheden uitgevoerd. Naar aanleiding van deze werkzaamheden zijn er bepaalde delen die extra onderzoek vereisen, of welke verder uitgebreid kunnen worden.

## 7.1 Behuizing modules

Er is geprobeerd de module behuizing te maken, door de polymorph om een mal de drukken. Het resultaat hiervan is onderstaande behuizing.



Zoals te zien is de behuizing niet strak af te werken als het met de hand wordt gemaakt. Hieruit is gebleken dat er voor een volgende module de polymorph onder druk moet worden verwerkt. Dit kan bijvoorbeeld door het persen tussen verschillende panelen, of door het onder druk in de mal te stoppen. Een andere mogelijkheid is het gebruik maken van een andere soort plastic.

## 7.2 Stroomverbruik

Het stroomverbruik vereist nog verder onderzoek. Er is onderzoek gedaan naar het gebruik van een Li-Ion accu in combinatie met een zonnepaneel. Deze combinatie is getest onder beperkte omstandigheden( zie bijlage 1). Zo zijn de modules niet getest in de buitenlucht waar ze meer zon kunnen vangen. Hierdoor zijn de behaalde resultaten van de accu in combinatie met het zonnepaneel niet compleet betrouwbaar.

In een vervolg op dit project zou een uitgebreider, betrouwbaarder onderzoek naar het stroomverbruik kunnen worden uitgevoerd.

## 7.3 Meerdere hoofdunits

Om het project binnen de gestelde tijd goed te kunnen afronden, is ervoor gekozen om aan elke server slechts één hoofdunit te koppelen. Bij een uitbreiding naar ondersteuning voor meerdere hoofdunits zal elke hoofdunit aan een gebruikersaccount moeten worden gekoppeld. Hierbij komen aanpassingen aan de database, de cliënt en de hoofdunit zelf kijken:

* Elke hoofdunit moet worden voorzien van een unieke code voor identificatie.
* In de database moeten gebruikersaccounts worden toegevoegd
* De tabel voor koppeling tussen hoofdunit en gebruikersaccounts moet worden toegevoegd.

## 7.4 Meer soorten modules

Binnen het project bestaat de mogelijkheid om meerdere modules toe te voegen. Zo kan er bijvoorbeeld een module gemaakt worden voor het meten van de zuurgraad. Om dit te doen moeten er verschillende aanpassingen worden gemaakt.

In de code voor de module moet een functie worden toegevoegd voor het berekenen van de waarde van de nieuwe sensor.

Voor het wijzigen van de Android applicatie wordt verwezen naar hoofdstuk ‘5.4.4 Uitbreiden triggers en soorten modules’ in dit verslag.

## 7.5 Efficiëntere synchronisatie Android applicatie

Voor de synchronisatie van de modules kan in plaats van de huidige DataManager klasse een implementatie van de efficiëntere en robustere SyncAdapter[[16]](#footnote-16) worden gebruikt.

# Bijlagen

**Bijlage 1:** Onderzoeksrapport V1.2

**Bijlage 2:** Plan van aanpak V1.2

**Bijlage 3:** SRS v1.1

**Bijlage 4:** Testrapport v1.0

1. Een geheugenlocatie voor het opslaan van bepaalde waarden [↑](#footnote-ref-1)
2. Deze deelt de kloksnelheid van de microprocessor voor het vertragen van de timer. [↑](#footnote-ref-2)
3. De normale processor werking wordt onderbroken om een bepaalde actie uit te voeren. [↑](#footnote-ref-3)
4. Wanneer een timer zijn maximale waarde (‘count’) overschrijdt wordt er een interne interrupt uitgevoerd [↑](#footnote-ref-4)
5. <http://www.nfc-forum.org/specs/spec_list/> [↑](#footnote-ref-5)
6. <http://developer.android.com/guide/topics/connectivity/nfc/nfc.html> [↑](#footnote-ref-6)
7. <http://protoplast.nl/> [↑](#footnote-ref-7)
8. <http://vertx.io/core_manual_java.html#writing-verticles> [↑](#footnote-ref-8)
9. <http://martinfowler.com/bliki/DecoratedCommand.html> [↑](#footnote-ref-9)
10. <http://silex.sensiolabs.org/> [↑](#footnote-ref-10)
11. <http://json.org/> [↑](#footnote-ref-11)
12. <http://developer.android.com/guide/topics/providers/content-providers.html> [↑](#footnote-ref-12)
13. <http://developer.android.com/reference/android/content/CursorLoader.html> [↑](#footnote-ref-13)
14. <http://developer.android.com/reference/android/support/v4/widget/SimpleCursorAdapter.html> [↑](#footnote-ref-14)
15. <http://www.rohscompliancedefinition.com/> [↑](#footnote-ref-15)
16. <http://developer.android.com/reference/android/content/AbstractThreadedSyncAdapter.html> [↑](#footnote-ref-16)