Plan van Aanpak

Wasmachines leven langer met Robijn

Teamnummer: 13

Teamnaam: Robijn

Naam + studentnummer:

1615865 Waila Woe

1660936 Thijs Hendrickx

1665553 Mathijs van Bremen

1657667 Yorick Schellevis

1664653 Bouke Stam

Documentdatum: 26-11-15

Versienummer: 0.1

# Index

|  |  |
| --- | --- |
| [Inleiding](#h.ursy46hijjwv) | 2 |
| [Onderzoek](#h.3hexsxifw51o) | 3 |
| [Op te leveren producten](#h.49ade1w4sxh1) | 6 |
| [Methode van kwaliteitsbewaking](#h.qxohm5aiodxz) | 9 |
| [Projectorganisatie](#h.ie9veacp3uyq) | 11 |
| [Projectactiviteiten](#h.u4sbtn9nz12b) | 12 |
| [Risico’s](#h.8ghpiext8qua) | 14 |
| [Bronvermelding](#h.g1ehvljbvpuv) | 16 |
| [Bijlagen](#h.2ytwwfjpntyk) | 17 |

# Inleiding

In deze opdracht gaan de kennis van C in een wat meer systematische omgeving gebruiken. Het doel is een besturingssysteem te maken voor een wasmachine voor de opdrachtgever op het Internet of Things. Hierdoor kan de gebruiker de wasmachine via het internet aansturen via een website, app of programma.

We komen in een tijdperk waarin steeds meer systemen bediend kunnen worden over het internet. Consumenten kunnen lichten, verwarming, televisie en veel meer allemaal bedienen met 1 druk op een knop op hun smartphone. In deze opdracht ligt de focus bij een wasmachine. Van de watertoevoer tot de rotatiesnelheid van de trommel of zelfs wasmiddel toevoeging, alles moet door middel van een ingesteld wasprogramma bediend kunnen worden via een makkelijke gebruikersinterface zoals een website, app of programma.

Dit verslag dient ervoor om een overzicht te maken van wat het project inhoudt en wat de afspraken tussen het team en de opdrachtgever zijn. Hierdoor krijgt iedereen betrokken een goed beeld van wat de uiteindelijke realisatie zal inhouden en wat er verwacht kan worden van het ontwerp proces.

Het doel is een functionerend gebruikersinterface te hebben dat verscheidene onderdelen van een wasmachine kan bedienen over het internet.

# Onderzoek

**Raspberry PI**

De raspberry pi is een goedkope en kleine computer gemaakt voor educatieve en hobby doeleinden. De versie die gebruikt wordt is de Raspberry Pi Model B rev2. Deze bestaat uit de volgende onderdelen [[1]](#footnote-0):

* SD kaart lezer en schrijver
* HDMI output
* Composite video output
* 2 USB 2.0 poorten
* 26 GPIO pinnen
* 3.5mm audio aansluiting
* Camera interface poort
* LCD display interface poort
* MicroUSB voeding aansluiting (700mA, 3.5W, 5V)
* 100mb Ethernet pport
* 512MB ram
* ARM 1176 700Mhz processor

Er zijn meerdere operating systems beschikbaar voor de raspberry pi [[2]](#footnote-1), de populairste daarvan is Raspbian. Dit is een versie van Debian linux, omgevormd voor de raspberry pi. De nieuwste versie hiervan is “Jessie”, welke ook wordt aangeraden door school, vanwege de versie van gcc.

**Webpagina server**

Er zijn veel webservers verkrijgbaar voor linux. De meest bekende is apache. Apache kan makkelijk worden geinstalleerd op linux via het apt-get commando [[3]](#footnote-2). Alle bestanden in de /var/www/html folder zijn dan beschikbaar via het netwerk.

**Socket server**

De socket server die praat met de webpagina kan vrij simpel worden gemaakt in C++. Bij de cursus Operating Systems & NetwerkProgrammeren is er gebruik gemaakt van een kleine library die sockets implementeerd in C++. Dit kan wederom worden gebruikt in dit project. Deze library is beschikbaar op de webpagina van het boek TCP/IP Sockets in C [[4]](#footnote-3).

**JavaScript server communicatie**

Om vanuit de browser te praten met de server zijn er twee makkelijke oplossingen. De eerste en meest bekende is AJAX. Met AJAX kan er een http request worden gestuurd naar de server zonder de pagina te refreshen. Het resultaat wordt dan teruggegeven via een callback methode.

De tweede optie zijn websockets. Sinds HTML5 wordt dit standaard ondersteunt door de browser en is het vrij gemakkelijk geworden om dit te implementeren. Met een websocket maakt de browser een directe verbinding met de server, die open blijft zolang als het nodig is. Berichten kunnen via deze verbinding beide kanten op worden gestuurd.

Websockets zijn in ons geval de beste keus. Dit is omdat websockets veel sneller zijn waardoor er vaker updates kunnen worden gestuurd naar de client. Ook hoeft de client niet voor elke update een request te sturen, maar kan de server dat doen wanneer hij nieuwe data heeft.

Javascript heeft goede ondersteuning voor websockets en heeft eenvoudige methodes om een socket verbinding te openen, berichten te ontvangen en te versturen. Hierdoor zal het niet veel tijd kosten om dit implementeren in de webpagina.

**RTOS**

RTOS is een library die synchronisatie objecten en een task scheduler biedt. Om dit toe te voegen moeten we de header file includen en de library linken. De library bestaat uit de volgende klassen:

* channel
* clock
* event
* flag
* mailbox
* mutex
* pool
* task
* timer
* waitable

RTOS draait in een realtime thread waardoor het de hoogste prioriteit heeft.

**Wasmachine emulator**

Omdat het niet realistisch is om echte wasmachines te gebruiken is er een wasmachine emulator gemaakt. Deze simuleert de hardware van de wasmachine [[5]](#footnote-4). De emulator draait op de lpc1114 chip met de ARM Cortex-M0 processor.

De wasmachine bestaat uit de volgende componenten:

* Wasmachine motor met instelbaar toerental van 0 t/m 1600 rpm
* Deurslot met vergrendeling
* Noodknop
* Verwarmingselement
* Kraan
* Pomp
* Zeeptoevoer
* Signaalled die brandt als het programma is afgelopen

De communicatie met de raspberry pi verloopt via een serieel interface, waarme commando’s naar de emulator kunnen worden gestuurt. De emulator reageert vervolgens met een tegenbericht. Een commando bestaat uit twee bytes, een request en een command. Het request geeft aan om welk onderdeel het gaat en het command is vaak de actie.

De reactie van de emulator bestaat ook uit twee bytes, namelijk een response en een status. De response herhaalt de request byte, met het hoogste bit aangezet. De status geeft de status van het component aan.

De emulator kan zich in 5 toestanden bevinden:

*Idle*

Dit is de begin toestand, de emulator reageert alleen het MACHINE\_REQ commando.

*Running*

De emulator reageert op alle commando’s en het indrukken van de knoppen.

*Stopped*

De deur is ontgrendeld en de watertemperatuur wordt niet meer gelijk gehouden. Het waterniveau blijft echter wel hetzelfde.

*Failed*

De failknop is ingedrukt, het vullen, verwarmen en pompen stopt. Na nog een drukt op de failknop veranderd de toestand naar stopped. De failed toestand simuleert een stroomuitval.

*Halted*

De noodknop is ingedrukt, de emulator reageert alleen nog op het opvragen van de machine status.

# Op te leveren producten

**Webpagina**

Een webpagina vanwaar de wasmachine bestuurd kan worden via websockets. Deze pagina moet een optie hebben om het ip van de wasmachine in te vullen. Ook moet er een keuze gemaakt kunnen worden uit een lijst met wasprogramma’s. Daarna moet het programma kunnen worden gestart of gestopt. De pagina laat dan zien hoe lang het programma nog duurt.

Naast de standaard wasprogramma’s moet de gebruiker ook handmatig instellingen als temperatuur, rotatie snelheid en tijdsduur kunnen instellen.

**Webserver**

Een webserver die de webpagina(s) host. Dit kan een bestaande server als apache zijn of zelf gemaakt. De server moet een http pagina kunnen versturen.

**Websocket server**

Wordt gebruikt voor de communicatie tussen de browser en het operating system van de wasmachine. De server moet berichten tussen het operating system en de webpagina kunnen sturen en moet een authenticatie mechanisme bevatten. De gebruiker moet een wachtwoord en gebruikersnaam invoeren om toegang te krijgen tot het operating system.

**Wasmachine operating system**

Het operating system houdt bij welk programma er bezig is en zorgt dat dat gevolgd wordt door de wasmachine. Het operating system geeft hoe lang het programma nog duurt en kan het programma afbreken of starten als de websocket dat bericht heeft gekregen.

**Requirements architecture**

Use case diagram met een beschrijving van elke use case en de relaties tussen de use cases. Ook een uitleg van de gemaakte keuzes bij het maken hiervan.

**Solution architecture**

* Beschrijving van de architectuur van het systeem
* Klassendiagram
* Taakstructurering
* Concurrency model
* STD van de controllers
* Beschrijving van communicatie protocol tussen browser en operating system

**Technisch verslag**

Een verslag van het technische ontwerp en de realisatie van het product. Een uitgebreide uitleg van de inhoud staat op sharepoint [[6]](#footnote-5).

# Methode van kwaliteitsbewaking

**Webpagina**

* Code moet voldoen aan bepaalde regels
  + Alle ID’s en klassen lowercase en variabelen gescheiden met ‘-’
  + Te lange regels verspreiden over meerdere regels
  + Accolades op zelfde regel
* HTML5, CSS en Javascript moeten voldoen aan de regels die bij de opmaak van deze bepaalde programmeertalen horen. Of de code voldoet aan deze regels kan gevalideerd worden met de “Markup Validation Service” van W3C.
* Functionaliteit van de webpagina(‘s) moet overeenkomen met wat er met de klant is afgesproken. Dit kan gecontroleerd worden door terugkoppeling naar de klant toe.

**Webserver**

* Testen of de webpagina bereikbaar is door middel van het browsen naar de bepaalde pagina via de website.

**Websocket server**

* Code moet voldoen aan bepaalde regels
  + Functies lowerCamelCase
  + Klassen UpperCamelCase
  + Variabelen lowerCamelCase
  + Accolades op zelfde regel
* Code moet getest worden door teamleden zelf. Mocht het deze tests met succes doorstaan dan kan er nog een test met iemand van buitenaf uitgevoerd worden.

**Wasmachine operating system**

* Code moet voldoen aan bepaalde regels
  + Functies lowerCamelCase
  + Klassen UpperCamelCase
  + Variabelen lowerCamelCase
  + Accolades op zelfde regel
* Code moet getest worden door teamleden zelf. Mocht het deze tests met succes doorstaan dan kan er nog een test met iemand van buitenaf uitgevoerd worden.

**Requirements architecture**

* Use case diagram moet voldoen aan de bijbehorende regels.
* Use case diagram moet worden gereviewd door een expert op dit gebied.

**Solution architecture**

* Diagrammen moeten worden gereviewd door één of meerdere experts op dat gebied.

**Plan van aanpak**

* Moet voldoen aan de eisen die bij een plan van aanpak gelden.
* Spel- en/of taalfouten worden door teamleden gecheckt. Later zal dit nog door iemand van buitenaf gecheckt worden.
* Het plan van aanpak moet worden gereviewd door een expert op dit gebied.

**Technisch verslag**

* Moet voldoen aan de eisen die bij een technisch verslag gelden.
* Spel- en/of taalfouten worden door teamleden gecheckt. Later zal dit nog door iemand van buitenaf gecheckt worden.
* Het technisch verslag moet worden gereviewd door een expert op dit gebied.

# Projectorganisatie

**De opdrachtnemers**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Naam | Extra taak | Email | Telefoonnummer | Woonplaats |
| Bouke Stam | Team leider | bouke.stam@gmail.com | 0643618009 | Utrecht |
| Yorick Schellevis | vergader voorzitter | y.schellevis@gmail.com | 0624371911 | Almere |
| Mathijs van Bremen | - | mathijs.van.bremen@gmail.com | 0652342107 | Rumpt |
| Thijs Hendrickx | - | thijs.hendrickx@online.nl | 0612815565 | Maarssen |
| Waila Woe | Notulist | waila.woe@student.hu.nl | 0641631887 | De Meern |

**De opdrachtgever**

De opdrachtgever is Jan Zuurbier van het bedrijf Swirl. Hij is te bereiken voor vragen per mail op [j.zuurbier@hu.nl](mailto:j.zuurbier@hu.nl)

**Begeleiding vanuit HU**

De begeleider van dit project, Marten Wensink, is te bereiken op m.wensink@hu.nl

# Projectactiviteiten

Hier onder is de lijst met taken per week te vinden. In de bijlage zit een kopie van de volledige planning en uren per taak.

Week 1

* Teamcontract opstellen
* Github aanmaken
* Voortgangsgesprek
* Globale planning maken
* Begin plan van aanpak

Week 2

* Interview vragen bedenken
* Begin requirement document
* Onderzoek
* Plan van aanpak

Week 3

* Interview opdrachtgever
* Requirements architecture af maken
* planning aanpassen aan requirements
* Beginnen aan technisch verslag

Week 4

* Technische verslag
* RTOS informatie verwerken
* Plan van aanpak inleveren

Week 5

* Solution architecture
* Technisch verslag

Week 6

* Solution architecture
* Technisch verslag

Project week 1

* Wasmachine OS maken
* Website maken
* Wasmachine protocol implementeren
* webserver maken/een volledige kiezen

Project week 2

* Website, webserver en OS aan elkaar koppelen
* Technisch verslag inleveren
* Project inleveren

# Risico’s

**Wijzigingen in project**  
Wijzigingen die gedaan worden tijdens het project is vaak een slecht teken. Wanneer dit in een vroege stadium is zal dat niet zo groot probleem zijn. Maar als dit gebeurd in een wat latere stadium dan zou de planning opnieuw gemaakt moeten worden. Sommige taken die gemaakt zijn mogelijk voor niks gemaakt en nieuwe taken komen erbij. Veel tijd gaat hierdoor verloren. Het is dus belangrijk om zo veel mogelijk contact te houden tussen de opdrachtgever en de opdrachtnemers zodat, wanneer wijzigingen ontstaan, die zo vroeg mogelijk worden behandelen.

**Slechte planning**  
Een slechte planning kan ervoor zorgen dat het project niet op tijd kan worden gerealiseerd. Dit kan bijvoorbeeld komen door een verkeerde inschatting van een onderdeel. Men kan denken dat een onderdeel van een project in korte tijd af kan worden gemaakt. Of juist het tegenovergestelde dat men denkt dat een onderdeel van een project veel tijd gaat kosten. Nog een voorbeeld is dat een onderdeel over het hoofd is gezien en dan het zelfs niet in de planning staat.   
  
**Onvoldoende kennis/onervarenheid**  
Niet iedereen in een project heeft evenveel kennis. De één heeft misschien veel meer kennis dan een ander. Dit kan misschien zijn doordat degene misschien dat al eens eerder heeft gedaan zoals op werk in een bedrijf juist thuis in vrije tijd als hobby. Bij het verdelen van taken moet hier aan gedacht worden zodat, iedereen een taak heeft om aan te werken zonder dat zij/hij te veel tijd aan verlies door het uitzoeken hoe het moet.

**Onvoldoend motivatie**  
Onvoldoende motivatie kan groot probleem zijn. Het is belangrijk dat iedereen gemotiveerd is zodat iedereen ook zo goed mogelijk gepresenteerd. Wanneer iemand niet gemotiveerd is en weinig bijdraagt aan het project kan dit een mogelijke bedreiging vormen voor het eindresultaat.   
  
**Projectleden niet samenwerken**  
Een goede samenwerking is cruciaal in een project. Mocht het zo zijn dat sommige projectleden niet met elkaar samen kunnen werken dan is het belangrijk om gaan zoeken naar oplossingen.  
Het beginnen met het zoeken wat hiervan de oorzaak of oorzaken zijn is een must.

**Ziek**  
Het gebeurd wel eens dat iemand in de projectgroep wel eens ziek word. Maar mocht het voorkomen een project lid of zelfs de gehele projectgroep ziek is, dan moet er overlegd worden bij de opdrachtgever wat hier aan gedaan kan worden en is het misschien dan ook wel nodig om de planning aan de passen.

**Projectlid niet meer aanwezig**

Wanneer een taakverdeling al is gemaakt en ieder project lid dus een taak heeft. Kan het misschien zo zijn dat iemand door een reden tijdelijk niet aan project kan werken of zelfs helemaal niet meer. De planning en taakverdeling zou dan moeten worden aangepast of juist helemaal opnieuw worden gemaakt.

# Bronvermelding

1 Raspberry pi model specifications. Retrieved from <https://www.raspberrypi.org/documentation/hardware/raspberrypi/models/specs.md>

2 Raspberry pi operating system images downloads. Retrieved from

<https://www.raspberrypi.org/downloads/>

3 Setting up an apache web server on a raspberry pi. Retrieved from <https://www.raspberrypi.org/documentation/remote-access/web-server/apache.md>

4 Michael J. Donahoo and Kenneth L. Calvert (2001). Practical C++ Sockets - Baylor. Retrieved from <http://cs.ecs.baylor.edu/~donahoo/practical/CSockets/practical/>

5 Marten Wensink (2015).Beschrijving wasmachine-emulator - Sharepoint. Retrieved from

<https://cursussen.sharepoint.hu.nl/fnt/35/TCTI-V2THO6-14/Studiemateriaal/Beschrijving%20wasmachine-emulator.pdf>

6 Marten Wensink (2015). Inhoud document ‘Technisch Verslag’ themaopdracht Domotica - Sharepoint. Retrieved from <https://cursussen.sharepoint.hu.nl/fnt/35/TCTI-V2THO6-14/Studiemateriaal/Inhoud%20Technisch%20Verslag%20themaopdracht%20Domotica.pdf>

# Bijlagen

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Themaopdracht 6** | **Team:** | **13** |  |  |  |  |  |  |  |  |  |  |
|  |  | Bouke [teamleider] | | Thijs | | Mathijs | | Yorick | | Waila | |  |  |
| Weeknr | Omschrijving activiteit | geplande  uren | bestede  uren | geplande  uren | bestede  uren | geplande  uren | bestede  uren | geplande  uren | bestede  uren | geplande  uren | bestede  uren | Totaal  geplande  uren | Totaal  bestede  uren |
| 1 | Teamcontract opstellen | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 5.0 | 5.0 |
| 1 | GitHub aanmaken | 0.2 | 0.2 | 0.3 | 0.3 |  |  |  |  |  |  | 0.5 | 0.5 |
| 1 | Voortgangsgesprek Marten Wensink |  | 0.5 |  | 0.5 |  | 0.5 |  | 0.5 |  | 0.5 | 0.0 | 2.5 |
| 1 | Vergadering 1 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 5.0 | 5.0 |
| 1 | Globale planning |  |  |  |  |  |  | 1.0 | 1.0 |  |  | 1.0 | 1.0 |
| 1 | Plan van aanpak | 2.0 | 2.0 | 2.0 | 0.0 | 2.0 | 2.0 | 2.0 | 2.0 | 2.0 | 2.0 | 10.0 | 8.0 |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 2 | Interview opdrachtgever afspraak maken | 1.0 | 1.0 |  |  |  |  |  |  |  |  | 1.0 | 1.0 |
| 2 | Vergadering 2 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 5.0 | 5.0 |
| 2 | GitHub private maken | 0.5 | 0.5 |  |  |  |  |  |  |  |  | 0.5 | 0.5 |
| 2 | Vragen bedenken voor interview | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 5.0 | 5.0 |
| 2 | Plan van aanpak | 2.0 | 2.0 | 2.0 | 2.0 | 2.0 | 2.0 | 2.0 | 1.0 | 2.0 | 1.5 | 10.0 | 8.5 |
| 2 | Requirement document |  |  |  |  | 1.0 | 1.0 |  |  |  |  | 0.0 | 1.0 |
| 2 | Vragen voor interview bij elkaar zetten | 1.0 | 1.0 |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 3 | Interview met opdrachtgever + voorbereiding | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 |  |  |
| 3 | Plan van aanpak | 4.0 |  | 4.0 |  | 4.0 |  | 4.0 | 3.5 | 4.0 |  | 20.0 | 3.5 |
| 3 | Uitwerken vragen interview | 1.0 |  | 1.0 |  | 1.0 |  | 1.0 | 1.0 | 1.0 |  |  |  |
| 3 | requirements architecture |  |  |  |  |  |  |  |  |  |  | 0.0 | 0.0 |
| 3 | Vergadering 3 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 | 1.0 | 1.0 | 1.0 | 1.0 | 3.5 | 3.5 |
| 4 | **Plan van aanpak inleveren!** |  |  |  |  |  |  |  |  |  |  | 0.0 | 0.0 |
| 4 | RTOS informatie verwerken |  |  |  |  |  |  |  |  |  |  | 0.0 | 0.0 |
| 4 | technisch verslag |  |  |  |  |  |  |  |  |  |  | 0.0 | 0.0 |
| 5 | technisch verslag |  |  |  |  |  |  |  |  |  |  | 0.0 | 0.0 |
| 5 | Solution architecture |  |  |  |  |  |  |  |  |  |  | 0.0 | 0.0 |
| 6 | Solution architecture |  |  |  |  |  |  |  |  |  |  | 0.0 | 0.0 |
| 6 | technisch verslag |  |  |  |  |  |  |  |  |  |  | 0.0 | 0.0 |
| P1 | Wasmachine OS maken |  |  |  |  |  |  |  |  |  |  | 0.0 | 0.0 |
| P1 | Website maken |  |  |  |  |  |  |  |  |  |  | 0.0 | 0.0 |
| P1 | Protocol implementeren |  |  |  |  |  |  |  |  |  |  | 0.0 | 0.0 |
| P1 | Webserver maken/kiezen |  |  |  |  |  |  |  |  |  |  | 0.0 | 0.0 |
| P1 | technische verslag |  |  |  |  |  |  |  |  |  |  | 0.0 | 0.0 |
| P2 | Website, server, OS aan elkaar koppelen | 2 |  | 2 |  | 2 |  | 2 |  | 2.0 |  | 10.0 | 0.0 |
| P2 | technisch verslag | 5 |  | 5 |  | 5 |  | 5 |  | 5.0 |  | 25.0 | 0.0 |
| P2 | Bugs in Website, server en OS fixen | 5 |  | 5 |  | 5 |  | 5 |  | 5.0 |  | 25.0 | 0.0 |
| EIND P2 | **Inleveren project** |  |  |  |  |  |  |  |  |  |  |  |  |
|  | **Totaal** | 29.2 | 12.7 | 26.8 | 8.3 | 26.5 | 11.0 | 28.0 | 15.0 | 27.0 | 10.0 | 137.5 | 57.0 |

1. Raspberry pi model specifications. Retrieved from <https://www.raspberrypi.org/documentation/hardware/raspberrypi/models/specs.md> [↑](#footnote-ref-0)
2. Raspberry pi operating system images downloads. Retrieved from

   <https://www.raspberrypi.org/downloads/> [↑](#footnote-ref-1)
3. Setting up an apache web server on a raspberry pi. Retrieved from <https://www.raspberrypi.org/documentation/remote-access/web-server/apache.md> [↑](#footnote-ref-2)
4. Michael J. Donahoo and Kenneth L. Calvert (2001). Practical C++ Sockets - Baylor. Retrieved from <http://cs.ecs.baylor.edu/~donahoo/practical/CSockets/practical/> [↑](#footnote-ref-3)
5. Marten Wensink (2015).Beschrijving wasmachine-emulator - Sharepoint. Retrieved from

   <https://cursussen.sharepoint.hu.nl/fnt/35/TCTI-V2THO6-14/Studiemateriaal/Beschrijving%20wasmachine-emulator.pdf> [↑](#footnote-ref-4)
6. Marten Wensink (2015). Inhoud document ‘Technisch Verslag’ themaopdracht Domotica - Sharepoint. Retrieved from <https://cursussen.sharepoint.hu.nl/fnt/35/TCTI-V2THO6-14/Studiemateriaal/Inhoud%20Technisch%20Verslag%20themaopdracht%20Domotica.pdf> [↑](#footnote-ref-5)