Plan van aanpak

Team: 4

Wouter van den Heuvel	1564952
Wilco Louwerse	1655993
Robin Noten	1671668
Daniel Klomp	1661521

Datum: 4-12-2015

Versie: 1.0

Index

1. Inleiding

In dit hoofdstuk zal beschreven worden waar dit document over gaat en wat de aanleiding was tot het maken van dit project/dit product. De opdracht beschreven in dit document is verkregen van de Hogeschool Utrecht faculteit Natuur en Techniek binnen de studie Technische Informatica.

De opdracht houd in om een webinterface te maken die via een besturingssysteem een wasmachine kan bedienen. Dit project (wat dus deel is van de studie Technische Informatica) heeft het doel om het team te leren hoe een webserver te laten communiceren met hardware. Ook word er geleerd hoe een product correct volgens de eisen van de opdrachtgever gemaakt kan worden door middel van een interview met de opdrachtgever. Uiteraard speelt het leren om beter samen te werken in teamverband ook een grote rol bij dit project.

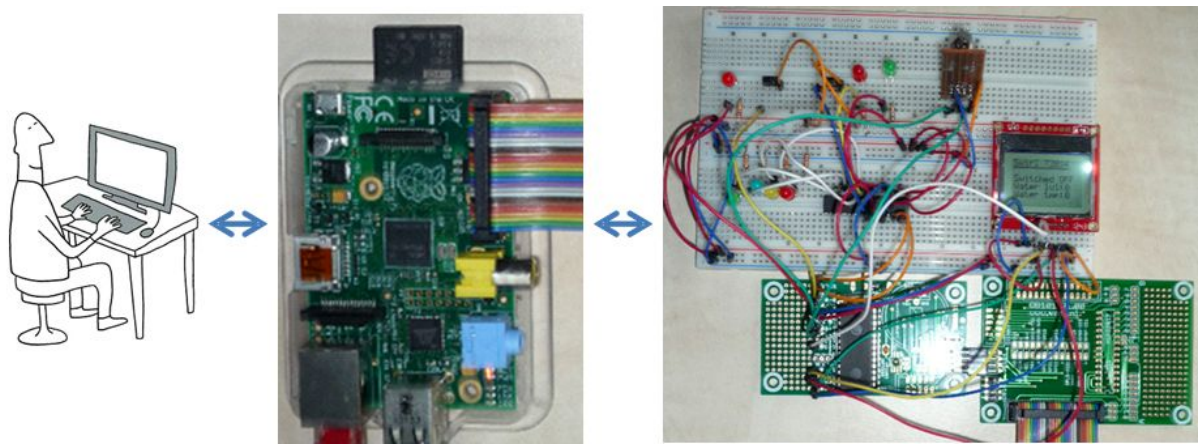
Binnen deze opdracht vervult een van de docenten binnen de studie technische informatica de rol als opdrachtgever. Uit het onderzoek met de opdrachtgever is gebleken dat deze een nieuwe revolutionaire soort wasmachine op de markt wilt brengen. Deze wasmachine is naast voor de gewone huisvrouw ook bedoeld voor ondernemingen die speciale of grote wasopdrachten hebben voor bijvoorbeeld een specifiek soort werkkleding.

In dit document zal beschreven worden hoe de doelstellingen voor deze opdracht gaan worden gerealiseerd. Het doel van deze opdracht is zoals eerder al genoemd: het maken van een webinterface die via een besturingssysteem een wasmachine kan bedienen. Er word hierbij gebruik gemaakt van een emulator voor de hardware van de wasmachine en een Raspberry Pi om commando’s naar deze emulator te kunnen sturen.

2. Requirements

In dit hoofdstuk word beschreven wat de eisen zijn om het eindproduct te kunnen realiseren. Dit is onderverdeeld in een aantal subgroepen, volgens de MoSCoW methode. Deze informatie is verkregen door een interview met de klant (zie hoofdstuk 3 Onderzoek).

Voordat deze subgroepen beschreven worden word eerst nog kort uitgelegd hoe het systeem er uit ziet zodat de lezer de subgroepen die hierna volgen beter begrijpt.
Hier onder volgt een afbeelding van het systeem, bestaande uit (van links naar rechts) de gebruiker achter zijn device dat toegang heeft tot internet en het webinterface van dit systeem, de Raspberry Pi waar de webserver op draait die weer in verbinding staat met het laatste onderdeel: de wasmachine zelf.



Must haves
Onder de Must haves wordt verstaan wat eisen zijn van het systeem om een werkend product te kunnen realiseren. Een werkend product houd dus in dat het voldoet aan de meest belangrijkste functionaliteiten. Deze functionaliteiten zijn:

- De gebruiker moet moet via de server verbinding kunnen maken met de machine.
- De gebruiker moet een wasprogramma kunnen kiezen.
- De gebruiker moet een wasprogramma kunnen aanzetten/pauzeren/stoppen.
- De machine moet een wasprogramma kunnen draaien.
- De fabrikant moet een wasprogramma kunnen samenstellen.
    - De fabrikant moet minstens 1 wasprogramma meeleveren.
    - Een wasprogramma moet uit minstens 1 fase bestaan.
- De machine moet beveiligd worden tegen de onkunde van de gebruiker.
    - De machine moet niet geopend kunnen worden wanneer een wasprogramma draait.
    - Beveiliging om het verwarming's element aan te zetten als er geen water in de machine aanwezig is.
    - Beveiliging om het pompen te laten stoppen als er geen water meer aanwezig is.

Should haves
Onder de Should haves wordt verstaan welke eisen het product nog meer zou moeten hebben naast de must haves om een volledig product te zijn. Een volledig product functioneert dus volledig zoals dat vooraf ontworpen is. In geval van tijdsgebrek of een andere verhindering kan een van deze eisen worden geschrapt.

- De gebruiker zou een wasprogramma moeten kunnen samenstellen en opslaan.
    - De gebruiker zou een eerder zelf gemaakt wasprogramma moeten kunnen wijzigen.
- De gebruiker zou feedback in de vorm van een bericht of geluid kunnen krijgen door de webserver, bijvoorbeeld op het moment dat een wasprogramma klaar is.
- De gebruiker zou een visuele weergave van het huidige wasprogramma en de status daarvan kunnen zien.
- De gebruiker zou de eindtijd en de tijdsduur moeten kunnen zien op het web interface.
- De gebruiker zou met een schuifbalk door het wasprogramma moeten kunnen lopen.
- Er zouden een aantal standaard wasprogramma's aanwezig moeten zijn: Witte was, wol en kwetsbare was.
- Een wasprogramma zou de volgende fases moeten bevatten: water naar de juiste temperatuur brengen, een van de soorten zeep toevoegen, (afval)water wegpompen, draaien en spoelen.

Could haves
Onder de Could haves wordt verstaan wat het product nog meer zou kunnen hebben om het product te perfectioneren. Dit wordt alleen uitgevoerd in het geval van tijdsoverschot.

- Om verbinding te maken met de server van de wasmachine kan een wachtwoord worden ingesteld.
- De gebruiker zou een tijdlijn kunnen zien waarop zichtbaar is in welk stadium het wasprogramma zich bevind, wat al klaar is en wat er nog gaat gebeuren.
- De gebruiker zou de optie kunnen hebben om het wasmachine systeem terug naar factory settings te resetten. (zelf gemaakte wasprogramma's verwijderen en standaard houden)
- De gebruiker zou niet essentiële informatie zoals bijvoorbeeld de temperatuur kunnen zien.

Will-not haves
Onder de Will-not haves wordt verstaan welke functionaliteiten het product in elk geval niet zal gaan hebben. Hier wordt dus geen tijd voor ingepland of aan besteedt.

- Er zal geen detectie zijn voor teveel was in de trommel.
- Er zal geen domeinnaam voor de webinterface geregistreerd worden.
- Er zullen geen thema's voor de webinterface zijn. Dit houd in dat het eindproduct één vaste opmaak zal hebben, zonder optie deze te veranderen.
- Er zal geen functionaliteit worden geïmplementeerd om vanuit één webinterface meerdere wasmachines aan te sturen.

3. Onderzoek

In dit hoofdstuk wordt besproken welke informatie moet worden verzameld om dit product zo goed mogelijk aan de gevraagde eisen te late voldoen en op welke manier deze informatie moet worden verkregen. Verder worden de experimenten die moeten worden uitgevoerd om dit te realiseren nader uitgelegd.

Om dit product goed in elkaar te kunnen zetten met de juiste functionaliteiten, is het nodig om bepaald informatie te verzamelen. Zo is het bijvoorbeeld belangrijk om te weten wat de functionaliteiten zijn van de wasmachine, maar het is ook om te weten wat de klant belangrijk vindt.
Om de functionaliteiten en de klant behoefden duidelijk te krijgen, is een interview gehouden met de klant. Voor dit interview zijn van te voren een aantal vragen opgesteld, waaruit duidelijk moest worden wat de klant wilt. Vervolgens zijn deze vragen aan de klant gesteld, en is er op de antwoorden van deze vragen verder ingegaan. Hierdoor is het een levendig gesprek gebleven tussen ontwikkelaar en klant. De uiteindelijk verkregen informatie is in het MoSCoW-model uitgewerkt. Hierdoor zijn de eisen voor het product duidelijk geworden.

Verder moet het team weten hoe deze eisen moeten wordt geïmplementeerd. De meeste informatie hiervoor wordt door de Hogeschool Utrecht geleverd door middel van een aantal cursussen. In deze cursussen worden een aantal hoorcollege's gegeven en moeten een aantal practica worden uitgevoerd.
Wat niet in de cursussen wordt behandel is hoe gewerkt moet worden met Java-script. Om Java-script toch onder de knie te krijgen wordt gebruikt gemaakt van externe bronnen zoals CodeAcadamy en Youtube.

Naast dit alles moeten om dit product te realiseren een aantal onderzoeken worden uitgevoerd. Zo zal het volgende moeten worden getest:

Welk gebruikers interface is het meest efficiënt en overzichtelijk voor de gebruiker?
Bij dit onderzoek wordt gekeken welke mogelijkheden bestaan voor het indelen van alle menu's binnen het webinterface, zodat daarna het meest efficiënte en overzichtelijke webinterface kan worden gebruikt in het product. Dit houd dus in de lay-out van het webinterface, maar ook de benaming van de buttons en welk volgend menu er na elke button geopend word. Zo wordt dus getest op de volgende aspecten van het webinterface: de overzichtelijkheid van de webpagina (Lay-out) en op de efficiëntie/duidelijkheid van de knoppen en menu's binnen de webpagina. Word het menu geopend dat een gebruiker zou verwachten als hij/zij op een knop drukt dat naar dat menu toe gaat?

Zoals eerder genoemd gaat er bij dit onderzoek gekeken worden naar de mogelijkheden die er bestaan. Een aantal bestaande mogelijkheid zijn:
-Een interface in het midden van het scherm, het controle paneel, waarin het huidige wasprogramma wordt weergegeven. Met aan de zijkant van het scherm een grote knop om een nieuw was programma aan te maken of om je profiel te wijzigen
-Een interface waarin de opties om naar het controle paneel en om je profiel aan te passen allemaal aan de zijkant in een net sub menu zitten.
Dit wordt getest door de mogelijke interfaces te realiseren en vervolgens door meerdere gebruikers te laten beoordelen. Op deze manier wordt de efficiëntie en de overzichtelijkheid

van het interface beoordeelt. Zo komt het meest gebruiksvriendelijke interface naar voren.

Wat is de meest gebruiksvriendelijke manier waarop een gebruiker zijn wasprogramma kan instellen?

Bij dit onderzoek wordt gekeken welke mogelijkheden er zijn voor het instellen / aanmaken van een nieuw wasprogramma zodat de beste mogelijkheid in dit product kan toegepast worden. Zo moet worden onderzocht hoe de de lay-out van het aanmaken van een wasprogramma efficiënt is en tegelijkertijd ook overzichtelijk voor de gebruiker. Maar ook welke stappen een gebruiker moet doorgaan om een wasprogramma samen te stellen. En dus in welke volgorde deze stappen voor het aanmaken van een nieuw wasprogramma het best kunnen worden uitgevoerd.

De gebruiker moet zoals eerder genoemd een wasprogramma samenstellen, wat gedaan word in een aantal verschillende stappen. Een aantal bestaande mogelijkheden voor het instellen van het wasprogramma zijn:

-De gebruiker moet eerst kiezen welke fases het wasprogramma gaat bevatten en daarna in welke volgorde deze fases moeten worden uitgevoerd.

-De gebruiker kan tegelijk fases toevoegen en de volgorde van deze fases veranderen.

Dit onderzoek kan getest worden door eerst de mogelijke manieren waarop een wasprogramma kan worden samengesteld uit te werken in verschillenden interfaces. Zodat deze getest kunnen worden door meerdere gebruikers. Hierdoor word duidelijk welke mogelijkheid het meest efficiënt en gebruiksvriendelijk is.

Welk interface voor het veranderen van het wasprogramma terwijl het draait is het meest efficiënt en overzichtelijk voor de gebruiker?

In dit onderzoek wordt getest wat de meest efficiënte en overzichtelijke manier is om het wasprogramma te wijzigen terwijl het al aan het draaien is of het draaiende wasprogramma gepauzeerd is. Het programma moet een aantal fases verder of terug kunnen worden gestuurd, om vervolgens daar in die fase weer verder te gaan. Hiervoor moet bijvoorbeeld worden onderzocht welke lay-out voor het wijzigen van deze fases het meest gebruiksvriendelijk is.

Om het wasprogramma te kunnen controleren zijn een aantal manieren mogelijk, zoals een schuifbalk, waarmee de gebruiker door de fases van het wasprogramma kan lopen, en tegelijkertijd dus kan zien hoever het wasprogramma is.

Een andere mogelijkheid is om het wasprogramma te controleren door middel van knoppen. Op deze manier kan de gebruiker door de verschillende fases lopen door middel van de knoppen terug en vooruit.

Dit onderzoek kan getest worden door eerst de mogelijke manieren waarop een wasprogramma kan worden samengesteld uit te werken in het interface. Zodat deze getest kunnen worden door meerdere gebruikers. Hierdoor word duidelijk welke mogelijkheid het meest efficiënt en gebruiksvriendelijk is.

Beveiliging

Ook moet de beveiliging nog worden getest. Zo is een van de beveiligingen dat de wasmachine niet geopend moet kunnen worden wanneer een wasprogramma draait. En ook moet het verwarming’s element altijd uitstaan als er geen water in de machine aanwezig is. Deze beveiligingen moeten worden getest door de wasmachine aan deze conditie’s bloot te stellen en te controleren of het systeem naar wens reageert op deze situaties.

4. Op te leveren te producten

In dit hoofdstuk wordt uitgelegd welke producten aan het eind van het project moeten worden opgeleverd. Deze producten zijn onderdeel van het functioneel ontwerp, oftewel de documentatie waarin uit wordt gelegd hoe het systeem in elkaar steekt.

Technisch verslag
Het belangrijkste product naast het prototype is het technisch verslag. In het technisch verslag wordt verslag gelegd over wat bereikt is in het project. Ook wordt in dit verslag behandel hoe het systeem is opgebouwd en hoe systeem functioneert. Hoe het systeem funcioneert wordt vooral behandelt in de Requirements Architecture en de Solution Architecture.

Requirements Architecture
Als eerste de Requirements Architecture, wat bestaat uit een use-case diagram en voor elke Use-case een eigen Activity diagram. Ook staat in de Requirements Architecture het Constraint model. Hier onder word nog even kort verteld wat deze diagrammen inhouden.
Het Use-case diagram is een diagram dat laat zien ‘wie’ ‘wat’ met het systeem kan doen. ‘Wie’ staat hierbij voor een actor buiten het systeem en ‘wat’ staat voor een use-case/actie binnen het systeem.
Een Activity diagram is een diagram die de zogeheten ‘flow’ van het syteem laat zien. Hierin is dus te zien in welke volgorde het systeem bepaalde activiteiten uitvoert.
Als laatste het Constraint model, dit is een model dat bestaat uit een tabel met alle criteria waar het systeem zich aan moet voldoen.

Solution Architecture
Verder wordt het Solution Architecture opgeleverd, wat bestaat uit het klassendiagram, een Concurrency diagram en voor elke taak van het systeem een State Transition Diagram.
In het Klassen diagram wordt de structuur van de code omschreven. Uit dit diagram kan ook veel van de functionaliteiten van het systeem worden gehaald.
In het Concurrency diagram wordt de interactie tussen de taken in het systeem vastgelegd. Hierin wordt dus duidelijk zichtbaar hoe de taken met elkaar communiceren.
Als laatste wordt voor elke taak in het systeem een State Transition Diagram gemaakt. Uit deze diagrammen wordt duidelijk hoe een taak in elkaar zit en welke functionaliteiten deze uitvoert. Ook wordt duidelijk waarmee deze taak nog verder communiceert.

5. Methode van kwaliteitsbewaking

In dit hoofdstuk word besproken hoe wij als team de kwaliteit van onze producten zullen waarborgen.

Om te kunnen garanderen dat al onze op te leveren producten aan de gestelde kwaliteit voldoen, worden de volgende checklijst gehanteerd:

- Het klassendiagram (zoals beschreven in het Technische verslag) dient actueel te zijn met de code.
- Alle code dient met behulp van Doxygen voorzien te zijn van commentaar.
- Alle code dient te voldoen aan de richtlijnen zoals vermeld in het document *TI C++ software rules* van W. van Ooijen.
- De code dient zonder problemen (warnings en errors) te compileren.
- Het programma dient zich geheel zoals beschreven in het functioneel ontwerp te gedragen.

Deze checklijst dient voor elk op te leveren product door minstens twee teamleden te worden uitgevoerd.

6. Projectorganisatie

In dit hoofdstuk word beschreven welke rollen er in ons team zijn en door wie deze worden uitgevoerd.

Voorzitter *(Daniel Klomp)*
De belangrijkste taken van de voorzitter zijn om orde te houden in het team door onder anderen te weten en te controleren of de samenwerking en communicatie goed verloopt binnen het team, ook is het van belang dat er goed contact word gehouden met de opdrachtgever door de voorzitter zodat het team goed gestuurd kan worden in de juiste richting dat de opdrachtgever wil dat het project verloopt.
Verder heeft de voorzitter nog een paar andere taken, namelijk het bijhouden van de urenverantwoording en het opstellen van de agenda, de agenda bevat de dagen en tijden waarop er bijeenkomsten of vergaderingen zijn voor het hele team .

Notulist *(Robin Noten)*
De notulist heeft de verantwoordelijkheid om tijdens of na vergaderingen de notulen te schrijven, deze moeten de belangrijkste dingen bevatten die er besproken/afgesproken zijn tijdens de vergadering.

Hardware verantwoordelijke *(Robin Noten)*
De hardware verantwoordelijke heeft de taak om alle benodigde hardware die de wasmachine simuleert te bewaren en mee te nemen wanneer nodig. Ook zal deze persoon de kwaliteit van de hardware in de gaten houden, dus checken op kleine beschadigingen en dergelijken.

Raspberry pi verantwoordelijke *(Wilco Louwerse)*
De Raspberry pi verantwoordelijke heeft de taak om de Raspberry pi en bijbehorende hardware elementen te bewaren en mee te nemen wanneer nodig. Ook zal deze persoon in de gaten houden hoe het zit met de kwaliteit van deze hardware, dit houd in het checken op kleine beschadigingen en defecten.

Teamlid *(Alle teamleden)*
Elk teamlid inclusief de notulist en voorzitter heeft de verantwoordelijkheid om voor zichzelf bij te houden hoeveel uur en waaraan hij per week werkt voor dit project. Dit is zodat de voorzitter tijdens de vergaderingen de urenverantwoording kan bijwerken.
Ook word van elk teamlid verwacht dat hij op tijd aanwezig zal zijn bij gezamenlijke activiteiten met betrekking tot dit project. Indien een teamlid niet aanwezig kan zijn zal hij dit zo snel mogelijk aan de teamleider/voorzitter en de rest van het team vermelden zodat deze daar rekening mee kunnen houden.

7. Projectactiviteiten

Voor het voltooien van het product moeten er uiteraard werkzaamheden worden verricht om de eind- en tussenproducten tijdig op te leveren.

Voor het project dat tot het bedoelde product leid moeten de volgende mijlpalen behaald worden:

Teamcontract

Er moet worden vastgelegd hoe we als team samenwerken, en hoe wij met elkaar contact houden. Het document waarin dit wordt gedaan is het Teamcontract, en dit document wordt doorgegeven aan de opdrachtgever om ervoor te zorgen dat de opdrachtgever kennis heeft van de gemaakte afspraken, en eventueel hierop kan inhaken.

Plan van aanpak

Er moet worden vastgelegd wat de planning is voor het verloop van het project, hoe er met risico's wordt omgegaan en welke speciale verantwoordelijkheden bij welke teamleden liggen. Hiervoor wordt het Plan van Aanpak opgesteld, en ter verantwoording aan de opdrachtgever opgeleverd.

Solution architecture

Er moet worden vastgelegd voor welk probleem de opdrachtgever een oplossing zoekt, hoe het eindproduct de eisen van de opdrachtgever vervult, en hoe het eindproduct ontworpen is. Hiervoor wordt de Solution Architecture opgesteld, waarin door middel van UML-schema's het ontwerp van het eindproduct wordt vastgelegd, en verslag wordt gedaan van de belangrijkste gebeurtenissen tijdens het project.

Demonstratie & oplevering eindproducten

Aan het einde van het project moet er een demonstratie worden gegeven van de opgeleverde producten en verslagen, mede om te bewijzen dat het opgeleverde eindproduct inderdaad doet wat de opdrachtgever verwacht.

Eindbeoordeling

Als laatste wordt de eindbeoordeling gegeven. Hierin wordt alles beoordeelt wat het team heeft opgeleverd.

8. Projectplanning

In dit hoofdstuk wordt uitgelegd hoe we onze planning week voor week hebben ingedeeld.

Project flow:
Om dit project goed te laten lopen moet eerst een MoSCoW Model worden gemaakt. In dit model wordt duidelijk wat de eisen zijn voor het eindproduct. Verder wordt duidelijk wat in het eindproduct moet zitten als functionaliteit, en wat eventueel bij tijdstekort kan worden geschrapt.
Vervolgens kan de Requirements Architecture worden gemaakt. Dit bestaat uit een Use-case model, voor elke Use-case een Activity Diagram en het Constraints model. Hierin wordt duidelijk wat het systeem allemaal moet kunnen en hoe de systeem zichzelf instant houd.
Aansluitend hierop kan de Solution Arcitecture worden gemaakt. Hierin wordt een klassendiagram, een Concurrency model en een aantal State Transition Diagram's gemaakt. Hierdoor wordt precies duidelijk hoe het systeem in elkaar zal gaan zitten. Tot in de details worden de functionaliteiten van het systeem uitgewerkt.

Nadat alles op papier is ontworpen, in het tijd om de hardware in elkaar te zetten, en om de html website te bouwen. Om dit te doen moeten een aantal onderzoeken worden uitgevoerd. Zo wordt eerst onderzocht wat de meest efficiënte en gebruiksvriendelijke interface is voor de html website. Vervolgens wordt getest wat de beste manier is om was programma's in te voeren op de website.

Nadat het eindproduct is gerealiseerd, wordt een technisch verslag gemaakt, waarin alle benodigde informatie en modellen worden verwerkt.

Exacte planning:
Hieronder staat de exacte planning in een tabel. De planning bestaat uit zes werk weken, twee vakantie weken en twee project weken. Elke taak is toegewezen aan een van de teamgenoten. De tijd waarmee hij verwacht wordt bezig te zijn aan die taak staat onder zijn naam genoteerd.

		Daniel	Wilco	Robin	Wouter	
Weeknr	Omschrijving activiteit	geplande uren	geplande uren	geplande uren	geplande uren	Totaal
1	Maandag hoorcollege week 1	2,0	2,0	2,0	2,0	8,0
1	Maandag vergadering, maken team contract	2,0	2,0	2,0	2,0	8,0
1	Donderdag werkbijeenkomst, maken Plan v. Aanpak	3,0	3,0	3,0	3,0	12,0

1						0,0
2	Maandag hoorcollege	2,0	2,0	2,0	0,0	6,0
2	Maandag werkbijeenkomst Opstellen Interview	2,0	2,0	2,0	0,0	6,0
2	Woensdag 20:00 skype meeting	1,0	1,0	1,0	1,0	4,0
2	Donderdag Interviewen klant	0,5	0,5	0,5	0,5	2,0
2	Donderdag Vergadering en werkbijeenkomst	1,0	1,0	1,0	1,0	4,0
2	Interview uitwerken	1,0	1,0	1,0	1,0	4,0
3	Maandag hoorcollege	2,0	2,0	2,0	2,0	8,0
3	Maandag Maken MoSCoW	1,0	1,0	1,0	1,0	4,0
3	Maandag Afmaken Plan van Aanpak	0,5	0,5	0,5	0,5	2,0
3	Donderdag Vergadering	1,0	1,0	1,0	1,0	4,0
3	Donderdag Maken Usecase diagram	2,0	2,0	2,0	2,0	
4	Maandag Maken Activity Diagram Control washing cycle	2,0	0,0	0,0	0,0	2,0
4	Maandag Maken Activity Diagram Provide access	0,0	1,0	0,0	0,0	1,0
4	Maandag Maken Activity Diagram Manage user profile	0,0	1,0	0,0	0,0	1,0
4	Maandag Maken Activity Diagram Read machine state	0,0	0,0	2,0	0,0	2,0
4	Maandag Maken Activity Diagram Display machine state	0,0	0,0	0,0	2,0	2,0
4	Maandag Maken Constraints	1,5	1,5	1,5	1,5	6,0
4	Woensdag Maken Activity Diagram Manage user profile	0,0	1,0	0,0	0,0	1,0
4	Donderdag werkbijeenkomst	2,0	2,0	2,0	2,0	8,0
4	Donderdag Feedback Plan v. Aanpak .Ovink	1,0	1,0	1,0	1,0	4,0
4	Deadline inleveren plan van aanpak voor 23:59					

4	Deadline Requirement voor 23:59					
5	Maandag Maken Taakstructurering	2,0	2,0	2,0	0,0	6,0
5	Maandag Maken Klassendiagram	0,0	0,0	1,5	1,5	3,0
5	Maandag Maken Concurrency model	1,5	1,5	0,0	0,0	3,0
5	Donderdag vergadering om 11:00	0,5	0,5	0,5	0,5	2,0
5	Donderdag Maken Klassendiagram	0,0	0,0	2,5	2,5	5,0
5	Donderdag Maken Concurrency model	2,5	2,5	0,0	0,0	5,0
						0,0
6	Maandag Maken STD	3,5	3,5	0	3,5	
6				0		
6	Donderdag feedback SA om 11:00	2	2	0	2	
6	Donderdag maken STD	1	1	0	1	3,0
6	Deadline alle Modellen moeten af (Solution Architecture)					0,0
Vac. 1	Hardware in elkaar zetten	3	3	0	0	
Vac. 1	Html pagina maken	0	0	3	3	
P1	Dag 1 Onderzoek gebruikers interface	1	0	0	0	1,0
P1	Dag 1 Maken gebruikers interface	7	8	8	8	31,0
P1	Dag 2 Verbindings protecol opstellen	2	2	2	2	
P1	Dag 2 Verbinding maken hardware en interface	6	6	6	6	
P1	Dag 3 Onderzoek was programma instellen	0	1	0	0	1,0
P1	Dag 3 Maken was programma instellen en verbinding maken	8	7	8	8	
P1	Dag 4 Onderzoek Wasprogramma wijzigen	0	0	1	0	1,0
P1	Dag 4 Maken wasprogramma wijzigen	8	8	7	8	31,0

P1	Dag 5 Maken wasprogramma uitvoeren	8	8	8	8	32,0
P1	Dag 5 Code optimaliseren	0	0	0	0	0,0
P2	Dag 1 Code optimaliseren	8	8	8	8	32,0
P2	Dag 2 Code afronden	8	8	8	7	31,0
P2	Dag 2 Doxygen testen	0	0	0	1	1,0
P2	Dag 3 Technisch verslag beginnen	8	8	8	8	32,0
P2	Dag 4 Technisch verslag bijwerken	8	8	8	8	32,0
P2	Dag 5 Technisch verslag afronden	8	8	8	8	32,0
P2	Deadline Donderdag 22-1 Inleveren eindproducten.					0,0

9. Risico’s

In dit hoofdstuk worden de mogelijke risico’s behandeld. Dit kunnen hardware, software of samenwerking’s problemen zijn. Verder worden de maatregelen genoemd om deze problemen te voorkomen.

Hardware
Als eerste is kan veel fout gaan bij het gebruik van hardware. Doordat de hardware die gebruikt wordt niet van hoge kwaliteit is, kan het makkelijker kapot gaan. Verder kan, doordat de hardware erg klein is, kleine onderdelen makkelijk kwijtraken.
Dit kan kan voorkomen worden door voorzichtig om te gaan met de hardware, en duidelijke afspraken te maken binnen het team over het hardware gebruik. Ook is het verstandig om een back-up te maken van de hardware, en in ieder geval voor alle onderdelen een reserve te hebben.

Samenwerking
Ook op de samenwerking kan het snel mis gaan. Teamgenoten kunnen hun taken niet uitvoeren of de communicatie kan niet soepel verlopen. Dit zijn problemen die in het begin van het project snel moeten worden opgepakt, omdat ze problemen kunnen veroorzaken voor de rest van het project.
Dit kan allemaal worden voorkomen door elke week een vergadering te houden, waarin wordt gekeken hoe het gaat in het team en met de teamgenoten individueel. Dit helpt om het probleem zo snel mogelijk het herkennen.

Software
Als laatste kunnen problemen voordoen met de software. Zo kan het medium wat gebruikt wordt om de code te delen defect worden of niet correct worden gebruikt door de teamleden. Hierdoor kan veel werkt dat als is verricht teniet worden gedaan.
Dit kan worden voorkomen door regelmatig lokaal een back-up te maken van de code en de documenten. Verder kan het nog helpen om met de teamleden duidelijke afspraken te maken over wanneer code mag worden samengevoegd.

10. Bronnen

Ooijen, W. *“TI C++ software rules”* (2014) Verkregen van:
https://cursussen.sharepoint.hu.nl/fnt/35/TCTI-V2THO6-14/Studiemateriaal/2015-2016-TI-C++-software-rules-1-0.pdf

Wensink, M. *“Themaopdracht 6: Domotica“ (2015)* Verkregen van:
https://cursussen.sharepoint.hu.nl/fnt/35/TCTI-V2THO6-14/Studiemateriaal/Moduleb eschrijving%20TCTI-V2THO6-14%20versie%207-11-15.pdf

Louwerse, W. “*Interview Klant*”(2015) Verkregen van:
https://drive.google.com/open?id=1p_5uGTc_Oi_MP93zYm7UUE18Rljm5MZr2WML k8QGJCQ