Plan van aanpak

Team 11

Versie 0.8

25-11-2015

Joost Wagensveld

Zehna van den Berg

Jessy Visch

Koen de Groot

# Versiebeheer

|  |  |  |
| --- | --- | --- |
| **Versienummer** | **Datum** | **Wijzigingen** |
| V0.1 | 11-11-2015 | Kopjes toegevoegd en mijlpalen in het document gezet. |
| V0.2 | 22-11-2015 | (opzet) Inleiding toegevoegd |
| V0.3 | 25-11-2015 | Onderzoek, op te leveren producten en methode van kwaliteitsbewaking toegevoegd. |
| V0.4 | 25-11-2015 | Projectorganisatie, Projectactiviteiten en Risico’s toegevoegd |
| V0.5 | 2-12-2015 | Fasering aangepast en hoofdstuknummers aangepast. |
| V0.6 | 3-12-2015 | Onderzoek aangevuld. |
| V0.7 | 3-12-2015 | Lay-out aangepast en Bronvermelding/referenties aangepast |
| V0.8 | 4-12-15 | fasering verder ingevuld bij 5.2 en de samenvatting van het interview nog apart kopje gegeven. |

Inhoudsopgave

[Versiebeheer 2](#_Toc436833786)

[1. Inleiding 4](#_Toc436833787)

[1.1. Achtergrond van de opdracht 4](#_Toc436833788)

[1.2. Doel van het document 4](#_Toc436833789)

[1.3. Doel van de opdracht 4](#_Toc436833790)

[2. Onderzoek 5](#_Toc436833791)

[2.1. Verzamelen van informatie 5](#_Toc436833792)

[2.2. Te raadplegen literatuur 5](#_Toc436833793)

[2.3. Experimenten die moeten worden uitgevoerd 5](#_Toc436833794)

[3. Op te leveren producten 6](#_Toc436833795)

[3.1. Prioritering van op te leveren producten 6](#_Toc436833796)

[3.2. Kwaliteitseisen waaraan de op te leveren producten moeten voldoen 6](#_Toc436833797)

[4. Methode van kwaliteitsbewaking 7](#_Toc436833798)

[4.1. Hoe wordt vastgesteld hoe of aan de kwaliteitseisen voldaan wordt? 7](#_Toc436833799)

[4.2. Tests die zullen worden uitgevoerd 7](#_Toc436833800)

[5. Projectorganisatie 8](#_Toc436833801)

[5.1. Verantwoordelijkheden van de teamleden 8](#_Toc436833802)

[6. Projectactiviteiten 10](#_Toc436833803)

[6.1. Mijlpalen 10](#_Toc436833804)

[6.2. Fasering in tijd & schatting in uren 10](#_Toc436833805)

[6.3. Wie wat en wanneer doet 10](#_Toc436833806)

[7. Risico’s 12](#_Toc436833807)

[7.1. Potentiele vertragingen 12](#_Toc436833808)

[7.2. Maatregelen om risico’s te voorkomen. 12](#_Toc436833809)

[8. Bronvermelding en referenties 14](#_Toc436833810)

# Inleiding

In dit hoofdstuk wordt een korte samenvatting gegeven van het doel en de achtergrond van deze opdracht.

## Achtergrond van de opdracht

De opdracht wordt gegeven door Jan Swirl van Swirl wasmachines B.V. Swirl wasmachines wil een wasmachine die via het internet te besturen is. De opdracht wordt uitgevoerd door Team Calgon, een team van 4 programmeurs. De wasmachine moet via een web interface te besturen zijn. Via deze web interface kan de gebruiker verschillende wasprogramma’s starten. Ook kun je de status zien van het programma welke op dat moment bezig is.

## Doel van het document

Het doel van dit document is het vastleggen van alles met betrekking tot de organisatie van dit project. In dit document is onder andere een globale planning, een overzicht van onderzoeken die moeten worden gedaan, verantwoordelijkheden van de teamleden en de kwaliteitseisen van het product te vinden.

## Doel van de opdracht

Het doel van de opdracht is niet alleen een innovatieve wasmachine maken voor Jan Swirl. De opdracht is tevens bedoelt voor Team Calgon om nieuwe programmeer- en software ontwerp kennis op te doen.

# Requirements interview

## Samenvatting van het interview met Jan Swirl van Swirl Industries

Het gaat om de ontwikkeling van een nieuw type wasmachine. De wasmachine heeft een interface die alleen vanaf een web interface te bereiken is. Op de wasmachine is alleen een noodknop aanwezig. In de toekomst wil Swirl Industries zelfs meerdere andere apparaten via hetzelfde type interface gaan bedienen. De interface moet zo ontwikkeld worden dat deze specifiek geschikt is voor gebruik op Smartphones en tablets.

De wasmachine moet alles kunnen wat een echte wasmachine ook kan. Via de web interface moet de gebruiker het wasprogramma kunnen kiezen. Per wasprogramma moet de gebruiker de temperatuur in kunnen stellen. Dit zullen de volgende programma’s zijn:

* Witte was
* Bonte was
* Fijne was

In de toekomst moet er vanuit Swirl Industries extra wasprogramma’s toegevoegd kunnen worden. De gebruiker hoeft dit niet te kunnen. Het updaten van de wasprogramma’s en de software moet via het internet kunnen of via USB. Als er updates zijn voor de wasmachine moet de gebruiker een notificatie krijgen. De gebruiker moet de update goedkeuren voordat deze geïnstalleerd wordt op de wasmachine.

De gebruiker moet wanneer de wasmachine een programma draait de volgende gegevens via de web interface kunnen bekijken. Dit is om de gebruiker van op de hoogte te stellen van de voortgang van de was.

* Het waterniveau
* De temperatuur
* De fase van het wasprogramma

De gegevens van de sensoren worden (bij verandering) actief naar de web interface gestuurd als de waarden veranderen. Dit gebeurt met een interval van maximaal 1 keer per seconde.

Als er ergens in het programma een fout optreedt dan moet de gebruiker hier ook onmiddellijk van op de hoogte gesteld worden. Ook wordt dit in een logbestand opgeslagen.

Voor de deurvergrendeling geldt de aanname dat als Lock is ingedrukt de deur ook echt vergrendeld is. Via de web interface worden behalve de gegevens van de sensoren ook de verschillende stadia en de resterende tijd van het wasprogramma doorgestuurd. Op deze manier weet de gebruiker hoe ver de was is.

Het is mogelijk om een timer in te stellen zodat een wasprogramma later begint. Het is ook mogelijk om het wasprogramma te pauzeren en af te breken (noodstop). Dit alles gebeurt vanuit de web interface. De web interface is alleen bereikbaar vanuit het netwerk waar de wasmachine zich in bevind. Als het wasprogramma is gepauzeerd moet later gekozen kunnen worden tussen afbreken van programma (afpompen), of het voortzetten.

Het koppelen vanuit de interface naar de wasmachine gebeurt d.m.v. een broadcast aanroep. Hier reageert de wasmachine op zodat er een directe verbinding ontstaat. Hierdoor hoeft het IP-adres ook niet bekend te zijn om toch de eerste verbinding op gang te brengen.

Alles wordt in een log bijgehouden. Bijvoorbeeld het aantal wasprogramma’s dat is gedraaid. Welke wasprogramma’s er gebruikt worden. Welke fouten er zijn opgetreden. Deze logbestanden moeten worden uitgelezen via de web interface. Ook kan de gebruiker uitlezen hoe vaak hij was draait en hoeveel energie en water verbruikt word.

Als het netwerk wegvalt, dient dat duidelijk gemeld te worden en hoort het wasprogramma zelfstandig door te lopen tot het klaar is.

Het verloop van een standaard wasprogramma is als volgt:

* Voorwas
* Spoelen kwartier (niet al te hoog temperatuur)
* Spoelen met schoon water
* Hoofdwas (3 kwartier)
* Spoelen (bont-was wel 6 keer fijne was 3 keer)
* Centrifugeren (fijne was niet) 1400 toeren
* Deur niet gelijk openen (deur 1 minuut gesloten houden na beëindigen programma)

Swirl Industries maakt geen gebruik van antikreuk. Dit hoeft dus ook niet ontwikkeld te worden. Na afloop van het programma wordt er een Mail/SMS verzonden of een geluid afgespeeld.

De interface kan door meerdere gebruikers bereikt worden en is door middel van een wachtwoord beveiligd. Eventueel moet er vanuit de interface het wachtwoord aangepast kunnen worden.

De interface moet er als volgt uit komen te zien. Op de eerste pagina moet de gebruiker het wasprogramma kiezen met behulp van grote knoppen. Na het kiezen van een wasprogramma krijgt de gebruiker een vervolgscherm te zien. Op dit scherm staan de instellingen van het wasprogramma, de temperatuur, de timer en dergelijke, ook is er een startknop. Daarna krijgt de gebruiker een pagina te zien met de huidige temperatuur, het waterniveau, fase weergave hoever hij is. Op deze pagina staat ook een stop-, noodknop. Voor de schetsen zie de bijlage.

# Onderzoek

In dit hoofdstuk wordt een kort overzicht gegeven van alle onderzoeken die moeten worden gedaan voordat het project daadwerkelijk word ontworpen en gebouwd.

## Onderzoeken die gedaan moeten worden

### 2.1.1 Welke webserver softwarepakket past het best bij onze wensen?

Er moet onderzoek worden gedaan naar het softwarepakket waar de web interface op gaat draaien.

In het onderzoek gaan we kijken welke webserver software we het best kunnen gebruiken op de Raspberry Pi. De focus van dit onderzoek gaat liggen op het resource verbruik van het pakket omdat we een zo zuinig mogelijke webserver willen gebruiken zodat er genoeg resources overblijven voor de aansturing van de wasmachine zelf.

### 2.1.2 Hoe functioneert een Linux queue?

Er moet onderzoek worden gedaan naar een queue op Linux. Deze queue moet gaan functioneren als communicatie tussen de web interface en het RTOS. Allereerst moet er worden uitgezocht of een queue voldoet aan de eisen. Dit onderzoek is belangrijk omdat de webserver niet direct met het RTOS mag communiceren omdat de schedulers van Linux en het RTOS dan door elkaar gaan lopen. Vervolgens moet er een klein prototype worden gemaakt van deze queue, hier meer over in hoofdstuk 2.3 experimenten.

### 2.1.3 Hoe functioneert het RTOS?

Er moet een onderzoek worden gedaan naar de exacte werking van het RTOS. Primair is uitvinden hoe we vanuit het RTOS de Linux queue kunnen uitlezen. Vervolgens moet er worden onderzocht

hoe we vanuit het RTOS instructies naar de wasmachine kunnen sturen en wat deze instructies voor data terug geven. Vervolgens moet er worden gekeken hoe we deze gegevens weer terug kunnen sturen naar de web interface. Dit is belangrijk om te weten zodat we dit mee kunnen nemen bij het ontwerp van onze Solution architecture.

### 2.1.4 Hoe ziet de opstelling van de wasmachine emulator eruit?

Er moet een onderzoek worden uitgevoerd naar de opbouw van de opstelling van de wasmachine. Het belangrijkst is uitzoeken hoe we de LPC met het scherm en de verschillende onderdelen moeten verbinden. Vervolgens moeten we uitzoeken hoe we de LPC met de PI verbinden.

### 2.1.5 Hoe kan de Raspberry Pi zijn IP adres bekend maken?

Er moet een onderzoek worden gedaan naar manieren om de Raspberry Pi zijn IP adres bekend te laten maken indien dit veranderd is. Dit is nodig omdat we op het schoolnetwerk en onze thuisnetwerken niet het hetzelfde statische IP adres kunnen gebruiken. Het is handig om het IP-adres te hebben zodat we via SSH de pi kunnen benaderen zodat we niet bij elk gebruik van de pi een scherm en een toetsenbord nodig hebben.

### 2.1.6 Hoe ziet een wasprogramma eruit?

Er moet worden onderzocht hoe een wasprogramma eruit ziet binnen de applicatie. Dit onderzoek moet duidelijk maken hoe we een rij van instructies kunnen aanbieden die 1 voor 1 worden uitgevoerd.

## Te raadplegen literatuur

* PowerPointpresentaties colleges.
* Beschrijving wasmachine pdf op SharePoint.
* Pdf “2015-2016-V2TH06 notes” op SharePoint([Wensink et al., 2015](#_ENREF_7)).
* Apache website voor de webserver software([The\_Apache\_Software\_Foundation, 2015](#_ENREF_5)).
* Raspberry Pi wiki([Raspberry\_Pi\_Foundation, 2015](#_ENREF_4)).
* Linux manual([die.net, 2015](#_ENREF_2)).
* Git Documentatie website([Git, 2015](#_ENREF_3)).
* Het Git Boek([Chacon, 2009](#_ENREF_1)).
* Verschillende technische websites (voor het webserver onderzoek).

## Experimenten die moeten worden uitgevoerd

Om sommige onderzoeken die genoemd zijn in hoofdstuk 2.1 kracht bij te zetten moet met de gevonden informatie een aantal experimenten worden uitgevoerd. De resultaten van de experimenten worden in een kort rapport geschreven.

### 2.3.1 Experiment met de Linux queue

Voor dit experiment wordt er een Linux queue geprogrammeerd waarin de web interface een bericht kan schrijven. Dit bericht moet vervolgens worden gelezen door het RTOS. Deze queue moet 2 kanten op werken, het RTOS moet dus ook een bericht naar de web interface kunnen sturen. Indien er tijd over is, is het wenselijk dat deze queues worden beveiligd met een mutex zodat er niet op het zelfde moment gelezen en geschreven wordt in de queue.

### 2.3.2 Experiment met de wasmachine emulator

Om te testen of de wasmachine emulator correct in elkaar is gezet moet er een experiment uitgevoerd worden. Dit experiment bestaat uit een directe call vanaf een PC naar de LPC maken en kijken of het resultaat het gewenste is.

### 2.3.3 Experiment Raspberry Pi IP adres

In dit experiment worden de bevindingen van het beschreven onderzoek in hoofdstuk 2.1.5 uitgevoerd om te kijken welke oplossing het meest wenselijk en het meest haalbaar is.

# Op te leveren producten

In dit hoofdstuk wordt een overzicht gegeven van alle producten die moeten worden gerealiseerd met een bijbehorende prioriteit. Ook wordt er beschreven aan welke kwaliteitseisen deze producten moeten doen.

## Prioritering van op te leveren producten

Hieronder zal besproken worden welke producten er aan het eind van het project opgeleverd moeten worden. Dit zal gedaan worden op volgorde van prioriteit waarbij het belangrijkste voor de klant eerst behandeld wordt.

Allereerst is het belangrijkste dat de wasmachine werkt. Daarom staat het prototype van het wasmachinegedeelte bovenaan onze lijst. Dit prototype bevat de basisfunctionaliteit van de hardware op de wasmachine om wassen te kunnen draaien. De hieraan opgedragen eisen zijn afkomstig uit het interview en het MoSCoW overzicht. Dit product staat bovenaan aangezien er zonder de software voor de wasmachine niet gewassen kan worden.

Even belangrijk voor de goede werking van de wasmachine is de web interface. De wasmachine wordt qua hardware interface zo klein mogelijk gehouden met alleen een noodknop en deurvergrendeling. Het kiezen van de wasprogramma’s en verdere instellingen om aan te passen zoals temperatuur en tijdsduur zal allemaal geregeld worden via de web interface. Hierdoor zal de wasmachine maar weinig daadwerkelijke functionaliteit geven voor de klant indien er niet mee gecommuniceerd kan worden.

Er wordt een MoSCoW overzicht gemaakt om er voor te zorgen dat de functionaliteit van de wasmachine overeenkomt met de wensen van de klant. Dit MoSCoW overzicht is een schema waar alle features worden ingedeeld naar hun prioriteit. Zo worden alle *musts* en *shoulds* in de planning opgenomen die aan het eind van het project af moeten zijn. De minimale functionaliteit die het product moet bieden om het project voldoende af te ronden is dat alle *musts* af moeten zijn. Alle *shoulds* worden geïmplementeerd als alles volgens schema verloopt. Indien het voorspoedig gaat zullen eventueel ook nog enkele *coulds* meegenomen.

Voor de klant zal het plan van aanpak in het begin essentieel zijn om op papier duidelijk te krijgen waarvoor ons team ingehuurd is. In dit document worden alle activiteiten van het team vastgesteld die uitgevoerd zullen moeten worden om het project goed af te ronden. Zo worden ook de verantwoordelijkheden verdeeld onder de teamleden en benoemd. Daarnaast zullen alle proeven beschreven worden die voor het testen van de hardware nodig zijn en het nut van hun uitvoering. Als laatste worden ook mogelijke risico’s aangekaart om er extra op voorbereid te zijn en indien ze toch voorkomen, welke gevolgen ze zullen hebben op het verloop van het project.

Wanneer het product voor deze opdracht afgerond is wordt er een technisch verslag opgesteld. In dit verslag worden alle geïmplementeerde functies van het product beschreven. Ook wordt beschreven hoe deze functies tot stand zijn gekomen. Dit zal van pas komen wanneer de klant zijn product uit wil breiden maar het vervolgproject bij een ander team terecht zal komen. Dit zal hun het uitzoeken van de werking van het programma besparen en zullen ze zich meer kunnen richten op het toevoegen van nieuwe features.

Als laatste prioriteit komen de testrapporten waarin de testen die uitgevoerd zijn ook tekstueel beschreven worden. Dit zal helpen bij het opzoeken hoe het systeem reageert op bepaalde acties en hoe ze verholpen zijn. Als vergelijkbare problemen later optreden kan makkelijk opgezocht worden hoe ze eerder opgelost waren. Deze testrapporten dienen ook als ondersteuning hoe bepaalde functionaliteit is geïmplementeerd of indien de tests falen, wat de oorzaak hiervan is.

|  |  |
| --- | --- |
| **Product** | **Prioriteit** |
| Prototype wasmachine | **1** |
| Web interface | **1** |
| Moscow overzicht | **2** |
| Plan van aanpak | **3** |
| Technisch verslag | **4** |
| Testrapporten | **5** |

## Kwaliteitseisen waaraan de op te leveren producten moeten voldoen

Alle code gerelateerde producten dienen te voldoen aan de C++ code standaard zoals deze te vinden is op SharePoint([Wensink et al., 2015](#_ENREF_7)). De verslagen dienen ten minste te voldoen aan de eisen die gesteld zijn in hun bijbehorende pdf op SharePoint. Voor het plan van aanpak is dit “inhoud plan van aanpak themaopdracht domotica.pdf” en voor het technisch verslag is dit “inhoud technisch verslag themaopdracht domotica.pdf”. Tevens dienen alle verslagen in correct Nederlands te zijn geschreven. Voor de web interface geldt dat deze moet voldoen aan de HTML5 standaard. Deze standaard is na te lezen op de site van w3.org([W3C, 2014](#_ENREF_6)). De web interface wordt onderworpen aan een zogeheten validator om te checken op HTML 5 fouten.

# Methode van kwaliteitsbewaking

## Hoe wordt vastgesteld hoe of aan de kwaliteitseisen voldaan wordt?

Er zal per op te leveren product anders gekeken worden of deze aan de kwaliteitseisen die eraan gesteld zijn voldoet.

De code die geschreven wordt voor het maken van de software moet aan de standaard op SharePoint voldoen. Dit zal gecontroleerd worden door ieder teamlid eerst zorgvuldig de code standaard door te laten lezen. Vervolgens worden er gewone code of eventueel speciale *Fagan inspecties* gedaan op de code van elkaar en hunzelf om zo gezamenlijk alles wat niet volgens de standaard is te verbeteren.

De verslagen ieder voldoen aan hun eigen stel van eisen. Om ervoor te zorgen dat alles soepel verloopt wordt eerst persoonlijk gecontroleerd door de teamleden als er nieuwe stukken geschreven zijn. Hierdoor zal er tijdens de algemene controle minder onnodige fouten voordoen. Als het concept af is worden ze nog even globaal doorgelezen door alle teamleden voordat het voor feedback naar een daarvoor gespecialiseerde docent wordt gestuurd. Deze zal dan feedback geven hoe het document verbeterd kan worden om te voldoen aan de richtlijnen. Het team zal deze feedback verwerken in het maken van de laatste versie van het document. Als alle feedback verwerkt is zal het document worden ingeleverd bij de opdrachtgever.

Voor de controle van de webserver wordt de code eerst doorgekeken door de teamleden om er zeker van te zijn dat het voor ieder duidelijk is wat er gebeurd. Vervolgens maken we gebruik van een online validator([W3C, 2014](#_ENREF_6)) die de code syntactisch test voor fouten. Als het aan beide voldoet is dit goedgekeurd.

## Tests die zullen worden uitgevoerd

Het grootste deel van de testen wordt gedaan in zogeheten unit tests. In deze unit tests wordt tijdens het programmeren het zojuist geprogrammeerde deel door de programmeur zelf getest. Verder doen de programmeurs zelf ook algehele systeem testen. Bij deze testen worden alle functionaliteiten van het systeem uitgevoerd en gekeken of het resultaat van deze actie de gewenste is. Als laatst wordt het systeem door externe mensen die niets met het project te maken hebben getest. Dit wordt gedaan om te kijken hoe een normale gebruiker met het systeem omgaat, op deze manier komen er altijd weer andere fouten en situaties aan het licht. Ook word met deze test direct bepaalt of de UI van de web interface intuïtief genoeg is door te kijken of de gebruiker zijn weg door de applicatie kan vinden zonder hulp te vragen.

# Projectorganisatie

## Verantwoordelijkheden van de teamleden

De teamleden hebben ieder rollen toegedeeld gekregen waar zij zelf ook verantwoordelijkheid voor willen nemen. Zij behouden deze rol voor zolang het project duurt om voor consistentie te zorgen binnen het gemaakte werk. Hierdoor ligt de verantwoordelijkheid voor bepaald werk altijd bij een persoon en kan dit niet op andere afgeschoven worden zonder overleg.

De volgende rollen zijn vastgesteld gedurende het project:

* Teamleider: Dit is de persoon die zorgt dat het project in goede baan verloopt en alles op tijd ingeleverd wordt. De initiële agenda wordt de avond voor een vergadering online gezet om een leidraad te creëren. Deze persoon verstuurt ook de mails voor reviews en afspraken met de opdrachtgever of externe personen. Deze rol wordt uitgevoerd door Joost Wagensveld.
* Notulist: Dit is de persoon die de notulen bijhoudt wat er in een vergadering gezegd wordt en welke besluiten hieruit zijn voorgekomen. Tijdens de vergadering wordt er kort opgeschreven per agendapunt wat hierover besproken is. Deze aantekeningen worden dan later verwerkt in de officiële notulen waar nog de goedkeuring van de overige teamleden over wordt gevraagd. Deze rol wordt uitgevoerd door Zehna van den Berg.
* Hardware bewaarder: Deze persoon neemt de Raspberry Pi tussen schooldagen mee naar huis en voert de initialisatie van de software op de hardware uit zodat hier later mee gewerkt kan worden. Hieronder valt ook de verantwoordelijkheid om de hardware weer mee naar school te nemen als hieraan voor het project gewerkt moet worden. Deze rol wordt uitgevoerd door Jessy Visch.
* “Git meester”: Deze persoon is verantwoordelijk voor het opzetten en het onderhouden van de git repository. Deze rol wordt uitgevoerd door Jessy Visch.

## Fasering

In dit hoofdstuk wordt behandeld wanneer aan de verschillende op te leveren producten gewerkt wordt.

Het eerste wat gedaan wordt na de vorming van het team is het opstellen van het teamcontract. Hier wordt duidelijk gemaakt wanneer iedereen beschikbaar is. Hoe en wanneer ieder bereikbaar is en welke sancties er volgen bij het niet naleven van de afspraken. Als dit opgesteld is zijn er duidelijke afspraken gemaakt die nageleefd dienen te worden en de verdere samenwerking bevorderen.  
Verder is er in de eerste week zo snel mogelijk een git-repository aangemaakt door Jessy zodat alle bestanden op een centrale plek staan. De docenten zijn hiertoe uitgenodigd. Zo kunnen ze alle producten makkelijk vinden.  
  
In week 2 wordt een interview gehouden met de opdrachtgever Jan Swirl. De uitslag van dit interview zal de functionaliteit van de wasmachine en de webinterface bepalen. Het interview wordt voorbereid door alle teamleden die samen over de vragen nadenken. Tijdens het interview zijn de rollen globaal verdeeld tussen notulisten (Zehna en Joost) en interviewers (Koen en Jessy). Dit interview is nodig om alle eisen vast te leggen. Deze eisen worden gebruikt voor het opstellen van de Requirements architecture en het Plan van aanpak.

Er wordt door alle teamleden aan het plan van aanpak gewerkt. In week 1 en 2 is hier al een globale opzet van gemaakt. Aan de hand van het interview worden er extra punten bijgevoegd die dan aan het licht komen. Nadat alle informatie is verzameld worden alle lege punten ingevuld. Het plan van aanpak wordt vervolgens in het begin van week 4 gereviewed door Gerald Ovink. Met de feedback die uit deze review is gekregen wordt het document aangepast. Bij onduidelijkheid over zaken worden er nog vragen gesteld aan de opdrachtgevers. Als er naar inzien geen verbeterpunten meer zijn wordt er een laatste spellingcheck gedaan en ingeleverd bij de opdrachtgever.

Nadat het interview in week 2 gehouden is kan er al gewerkt worden aan de requirement architecture van het product. Dit legt de eisen van de wasmachine en de webinterface vast waaraan ze moeten voldoen. Hier kan gelijktijdig met het plan van aanpak aan gewerkt worden aangezien ze beide alleen afhankelijk zijn van de informatie uit het interview. De concept versie van de requirement architecture wordt eerst gereviewed door docent Joost Schalken-Pinkster. De feedback die uit deze review komt wordt gelijk verwerkt. Dit is nodig omdat de requirement architecture volledig af moet zijn voordat aan de solution architecture gewerkt kan worden.

In week 5 en 6 wordt er aan de solution architecture gewerkt. Hier wordt aan begonnen zodra de requirement architecture is ingeleverd. De deadline voor de solution architecure is aan het eind van week 6. Hierom willen we graag dat we een conceptversie klaar hebben liggen aan het begin van week 6 zodat hier feedback op gegeven kan worden. Net als bij het plan van aanpak wordt hier dan de feedback op verwerkt zodat er een definitieve versie aan het eind van de week klaarligt.

Tijdens het maken van de solution architecture worden er al kleine testjes uitgevoerd op de hardware om te kijken welke software communicatie er mogelijk is tussen de verschillende interfaces. Zo moet er gekeken worden hoe er commands tussen het RTOS en de webinterface verwisseld worden. De resultaten van deze experimenten zullen direct invloed hebben op het ontwerpen van de solution architecture. Dit komt omdat er misschien speciale methoden ontworpen en ontwikkeld moeten worden om de software vloeiend te laten communiceren. De definitieve versie van de solution architecture moet ingeleverd worden voor de kerstvakantie.

Na de kerstvakantie wordt er volledig gestort op het coderen van alle software. Dit is het laatste op te leveren product van het project. Hier zal gedurende beide projectweken de hele week aan gewerkt worden. De code wordt tussendoor door teamleden nagekeken om ervoor te zorgen dat het netjes blijft. Als er problemen optreden die het team zelf niet kan oplossen qua codeerkennis wordt er om hulp gevraagd. Het programmeren zal onderverdeeld worden in 2 groepjes die zich ieder richten op de wasmachine danwel de webinterface. Op deze manier willen we zorgen dat er zo veel mogelijk onafhankelijk geprogrammeerd kan worden zonder op andere te hoeven wachten.

Als het project helemaal is afgerond zal er na inlevering van het product een evaluatie plaatsvinden. Hierin kan ieder teamlid zelfstandig zijn evaluatie van het project invullen. Deze evaluatie zal betrekking hebben op het verloop van het project in het algemeen, danwel de persoonlijke inzet van de teamleden.

# Projectactiviteiten

## Mijlpalen

|  |  |
| --- | --- |
| **Datum / weeknummer** | **Mijlpaal** |
| Week 1 | Inleveren teamcontract |
| Week 1 | Aanmaken git repository |
| Week 1 | Uitnodigen docent git repository |
| Week 2 | Interview houden met klant |
| Week 4 | Reviewen plan van aanpak |
| Week 4 | Inleveren plan van aanpak |
| Week 5 | Reviewen requirement architectuur |
| Week 5/6 | Reviewen solution architectuur |
| Week 6 | Inleveren solution architectuur |
| Week 2 | Inleveren programmacode |
| Week 3 / 22-1-2016 | Inleveren peer review (per e-mail) |

## Fasering in tijd & schatting in uren

Er wordt per mijlpaal ingeschat hoeveel tijd dit ongeveer in manuren in beslag zal nemen. Daarbij zal ook het tijdsbestek worden gemeld over hoeveel dagen dit verdeeld kan worden.

* Opstellen teamcontract: 6 manuur, binnen 1 dag
* Aanmaken git repository: 1 manuur, binnen 1 dag
* Uitnodigen docent git repository: 1 manuur, binnen 1 week aangezien docent adressen niet voorhanden waren.
* Interview houden met klant: 20 manuur, week 2 houden van interview, begin week 3 samenvatten notulen.
* Plan van aanpak: 50 manuur
* Reviewen plan van aanpak: 1 manuur voor het opsturen, 6 manuur voor het verwerken van de feedback.
* Inleveren plan van aanpak: 1 manuur
* Concept requirement architectuur: 50 manuur
* Definitieve requirement architectuur: 16 manuur
* Concept solution architectuur: 60 manuur
* Definitieve solution architectuur: 16 manuur
* Programmeren server: 60 manuur
* Programmeren wasmachine: 120 manuur
* Peer review: 6 manuur

## Wie wat en wanneer doet

* Het opstellen van het teamcontract wordt gezamenlijk gedaan.
* De git repository wordt aangemaakt door Jessy, hij nodigt ook de teamleden en docenten uit voor toegang tot de directory en eventueel andere zaken hierbij.
* Interview met klant wordt door alle teamleden aan gewerkt. Zo worden de vragen samen bedacht. Tijdens het interview wordt er wel een kleine afsplitsing gemaakt tussen notulisten, Zehna en Joost, en de interviewers, Jessy en Koen. Alhoewel bij doorvragen deze ook door de notulisten gevraagd worden.
* Plan van aanpak: De opzet wordt door Jessy en Koen gemaakt. De punten worden later voornamelijk ingevuld door Zehna en Jessy. Waarna er nog een algehele controle door alle teamleden over gedaan wordt.
* Requirement architecture: Deze wordt opgezet door Joost en Koen. Als de andere teamleden klaar zijn met hun bezigheden wordt hiermee geholpen. Aan het eind wordt ook hier alles door alle teamleden nagekeken voordat het wordt ingeleverd.
* Solution architectuur: Hier wordt door alle teamleden aan gewerkt.
* Programmeren: Aan het programmeren van de software draagt iedereen zijn stuk bij. Hoe de daadwerkelijke verdeling van de taken zal lopen wordt alvorens er gecodeerd wordt bepaald. Er zal waarschijnlijk wel een splitsing vinden waarbij de hoofdtaak zich op de server respectievelijk de wasmachine gericht zal zijn.
* Peer review: Dit wordt door ieder teamlid zelfstandig gemaakt en zal naar de daarvoor bestemde docent gemaild worden.

# Risico’s

Hier worden de mogelijke risico’s besproken die tijdens het verloop van het project op kunnen treden. Erna worden ook enkele maatregelen genoemd die genomen kunnen worden om mogelijke problemen zoveel mogelijk te voorkomen.

## Potentiele vertragingen

* Teamlid kan vanwege omstandigheden niet naar school komen
* Teamlid valt weg gedurende onbepaalde tijd
* De hardware is niet voorhanden als eraan gewerkt moet worden
* De hardware gaat stuk.
* Het schrijven van de code voor het project duurt langer dan verwacht.
* De docent heeft nog geen tijd om te reviewen.

## Maatregelen om risico’s te voorkomen.

* Om potentiële vertragingen door het ontbreken van een teamlid te voorkomen wordt er gestreefd om alles zoveel mogelijk volgens de planning af te maken. De planning is er zo op ingesteld dat alle *must* en *should* *haves* in de standaard planning zijn meegenomen als iedereen er vol aan kan werken. Zo kan er indien er tijdgebrek ontstaat vanwege gebrek aan mankracht of capaciteit, ingeperkt worden op de *shoulds* van het systeem. Hierdoor zal de oorspronkelijke functionaliteit van het systeem nog wel afgemaakt kunnen worden.
* Als de hardware niet aanwezig is zal er gekeken worden wat er voor de rest op dat moment gedaan kan worden. Om te voorkomen dat de hardware niet aanwezig is hebben we specifieke mensen aangewezen die elke keer de hardware mee moeten nemen zodat het duidelijk is wie het meeneemt.
* Als de hardware stuk gaat wordt er bij Marten om vervanging of reparatie gevraagd en zo spoedig mogelijk opgehaald.
* Bij problemen die zich tijdens het programmeren van de software voordoen kan er om hulp gevraagd worden via mail bij Wouter dat dan via response via mail of persoonlijk contact bekeken kan worden hoe het opgelost kan worden.
* De docenten worden, zodra het te reviewen product afgemaakt is of maar enkele aanpassingen nog wenselijk zijn, tijdig gemaild of ze tijd willen vrijmaken voor het reviewen van ons product. Hierna hebben we een ruim tijdsbestek in de planning opgenomen waarin ze feedback terug kunnen geven.

# Bronvermelding en referenties

Chacon, S. (2009). *Pro git*: Apress.

die.net. (2015). Linux Documentation. from <http://linux.die.net/>

Git. (2015). Git Documentation. from https://git-scm.com/documentation

Raspberry\_Pi\_Foundation. (2015). Raspberry Pi Documentation. Retrieved 26-11-2015, from https://[www.raspberrypi.org/documentation/](http://www.raspberrypi.org/documentation/)

The\_Apache\_Software\_Foundation. (2015). The Apache Software Foundation. from <http://www.apache.org/>

W3C. (2014). HTML5 - A vocabulary and associated APIs for HTML and XHTML. Retrieved 30-11-2015, from <http://www.w3.org/TR/html5/>

Wensink, M., Zuurbier, J., van Ooijen, W., Ovink, G., Schalken-Pinkster, J., & van Doesburg, A. (2015). SharePoint. Retrieved 9-11-2015, from https://cursussen.sharepoint.hu.nl/fnt/35/TCTI-V2THO6-14/default.aspx

# Bijlage:

