Plan van aanpak

Team 11

Versie 0.6

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 | 2-12-15 | Fasering aangepast en hoofdstuknummers aangepast. |
| V0.6 | 3-12-15 | Onderzoek aangevuld. |

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 wilt 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. Op deze web interface kan de gebruiker verschillende wasprogramma’s starten en de status van het programma zien wat 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 hier nieuwe programmeer- en software ontwerp kennis op te doen.

# 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 pool 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 uit kunnen lezen. 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 wasmachine emulator opstelling 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 tussen het schoolnetwerk en onze thuisnetwerken niet het zelfde 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 er uit ziet binnen de applicatie. Dit onderzoek moet duidelijkheid bieden hoe we makkelijk een rij aan instructies aan kunnen bieden die dan 1 voor 1 worden uitgevoerd.

## Te raadplegen literatuur

* PowerPointpresentaties colleges
* Beschrijving wasmachine pdf op SharePoint
* Pdf “2015-2016-V2TH06 notes” op SharePoint
* Apache website voor de webserver software
* Raspberry pi wiki
* Linux manual
* 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 word 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 word 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

|  |  |
| --- | --- |
| **Product** | **Prioriteit** |
| Plan van aanpak | **3** |
| Technisch verslag | **4** |
| Prototype wasmachine | **1** |
| Web interface | **1** |
| Moscow overzicht | **2** |
| 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. 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 geld dat deze moet voldoen aan de HTML5 standaard. Deze standaard is na te lezen op <http://www.w3.org/TR/html5/>. 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?

//hier moet nog een stukje geschreven worden…

## 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

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.

# 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 4 | 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