Marco Schriek, Wouter Goyen, Dominic Voets,

Eveline Ververgaert en Joey Nieuwkoop

Abstract

Dit document bevat de uitwerkingen van opdracht 3 over het onderwerp CM voor het vak SD6.

CM plan en SC plan

Source Control & Configuration Management

# Inleiding

In dit document wordt de opdracht CM plan en SC plan voor de proftaak uitgewerkt. De volgende onderdelen worden behandeld:

* Een Configuration Managment plan
* Een Source Control plan

Tabel van Contents

[1. Inleiding 1](#_Toc477519943)

[1. Configuration management fouten beheersing 3](#_Toc477519944)

[1.1. Code conflicten 3](#_Toc477519945)

[1.2. (Te) Late commits 3](#_Toc477519946)

[1.3. Slechte afbakening tests 3](#_Toc477519947)

[1.4. Onduidelijke release kandidaat 3](#_Toc477519948)

[1.5. Ongedefinieerde fouten oplossing 4](#_Toc477519949)

[2. Plan SC voor proftaak 5](#_Toc477519950)

# 1. Configuration management fouten beheersing

In dit hoofdstuk worden de mogelijke problemen uitgelegd die in de proftaak voor kunnen komen. Deze problemen worden uitgelegd en vervolgens wordt er ook de oplossing voor gegeven.

## 1.1. Code conflicten

De proftaak beslaat het schrijven en bouwen van autonomische sturing voor een transportvoertuig. Er zullen verschillende onderdelen zijn waar verschillende teamleden aan werken. Het is mogelijk dat door veranderingen en overlap een conflict zal ontstaan in deze onderdelen. Deze conflicten moeten dan opgelost worden door de programmeurs.

Versiebeheer maakt het mogelijk om terug te kunnen naar de voorgaande versie of werkende versie. Veranderingen worden doorgevoerd naar de main branch wanneer een set afspraken nagekomen is.

De belangrijkste afspraak is dat functionaliteiten die niet door de testen komen niet naar de main branche doorgevoerd mogen worden.

## 1.2. (Te) Late commits

De teamleden voor deze proftaak, wij de uitvoerders, zijn nog niet ervaren in continuous integration. Het is belangrijk dat we commits maken terwijl dat niet altijd correct aanvoelt. Dit zorgt ervoor dat iedereen de taak kan oppakken en er geen afhankelijkheid ontstaat op een lokale werkruimte/kopie.

Commits maken als een feature nog niet volledig is voelt niet goed aan. Echter kan dit opgelost worden door branches te maken waar deze features dan in komen. Andere teamleden hebben dan de meest recente code voor deze feature. Dit is zodat de feature niet afhankelijk is van het teamlid dat er mee bezig was. Dit voorkomt mogelijkheden waar niet aan een feature gewerkt kan worden op het moment dat dit teamlid niet aanwezig kan zijn.

## 1.3. Slechte afbakening tests

Releases zijn bedoeld voor de product owner/klant. Wanneer de testen niet goed geschreven worden (lees: geen hoge test coverage), dan kunnen veranderingen ervoor zorgen dat releases niet op geplande momenten uitgegeven kan worden. Dit heeft invloed op de geplande demo-momenten.

Wanneer testen niet de functionaliteiten goed afbakenen, dan zal de release niet meer uitgebracht kunnen worden over verloop van tijd. De testen worden ontworpen om de eisen van de klant te kunnen vervullen. De gewenste features worden op een rijtje gezet en als een checklist in de storyboard gehangen. Deze features leveren direct aan de hand van de beschrijving een test op.

## 1.4. Onduidelijke release kandidaat

Een release is een product dat gedeeld wordt met de klanten/product owners. Er moet wel besloten worden wat de volgende release toegevoegd zal worden. De release moet ook daadwerkelijk toegevoegde waarde hebben voor de klant. Het is zeer belangrijk dat er duidelijk afgesproken wordt wat er opgeleverd zal worden.

Elke sprint gaat het team in discussie met de product owner. De gewenste functionaliteiten worden geprioriteerd en in volgorde ontwikkeld. Zo zal de product owner elke release weten wat er veranderd is aan het software product.

## 1.5. Ongedefinieerde fouten oplossing

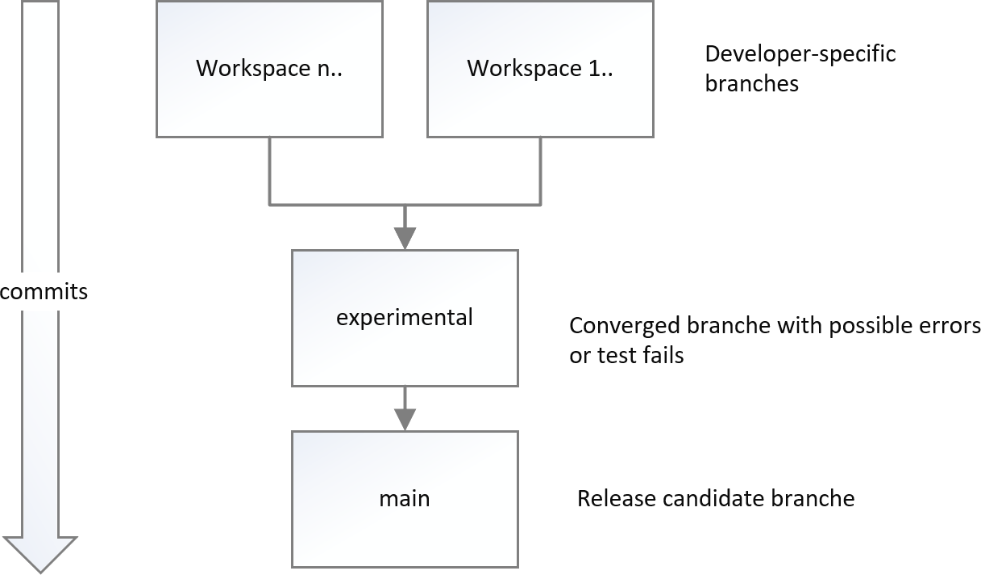
Fouten zijn onvermijdelijk in de ontwikkeling van software. Oplossen is zeer belangrijk. Er moet een systeem bepaald worden zodat fouten een prioriteit kunnen krijgen en verwerkt worden in de planning. Als het niet gedefinieerd wordt, zullen fouten zich opstapelen of worden er problemen opgelost die niet relevant zijn voor de volgende release.

Om dit tegen te gaan is zal documentatie bijgehouden moeten worden. Deze documentatie bestaat uit een handleiding of zelfs een bug report. Eventueel kan in een logboek of bij de tickets op het scrumbord commentaar geleverd worden als toevoeging. Wanneer opstapelende fouten voorkomen kan met de product owner overlegd worden desbetreffende onderdelen de hoogste prioriteit te geven.

# 2. Plan SC voor proftaak

Het software programma bestaat uit handgeschreven codes die uiteindelijk gecompileerd worden voor een microcontroller. Deze codes worden beheert door een versie beheer.

Onze versiebeheer zal gebruik maken van verschillende branches. Elk teamlid heeft een eigen branche om zijn taak in te verwerken. Een experimentele branche bevat de laatste versie van het gehele project, waar naar gecommit kan worden zonder problemen. De main branche dient als release code; alleen werkende code mogen naar de main branche toe gecommit te worden. Een  De componenten worden getest en daarna gecommit. De main branche mag bugs bevatten, maar mogen niet falen voor de geschreven testen.



De main branche kan ten alle tijden direct gebruikt worden als release kandidaat.

Bugs en nieuwe functionaliteiten mogen alleen verwerkt worden via de experimentele branche. Vervolgens mag de experimentele branche gecommit worden als deze voldoet aan alle eisen, waaronder het succesvol zijn van alle test cases.

Aangezien we in de proftaak hoogstwaarschijnlijk niet te maken krijgen met verschillende klanten met verschillende wensen, zal er niet gekeken worden naar speciale versies. Wel wordt er gedocumenteerd welke versies gereleaset zijn en wie deze hebben ontvangen. Versies die gereleaset zijn moeten ten alle tijden door de tests heen komen. Verder mag een versie die gereleaset is geen bugs meer bevatten.

Verder als de opdrachtgever en/of product owner een verandering wil doorvoeren, zal dit gedaan worden aan de hand van een Change Request. Dit formulier bevat alle benodigde informatie over de gewenste verandering, o.a. de prioriteit, eventuele deadline, wat de verandering inhoudt en waar dit effect op zal hebben. Uiteindelijk zal gezamenlijk besloten worden of deze veranderingen haalbaar zijn en op welke termijn ze toegevoegd kunnen worden.