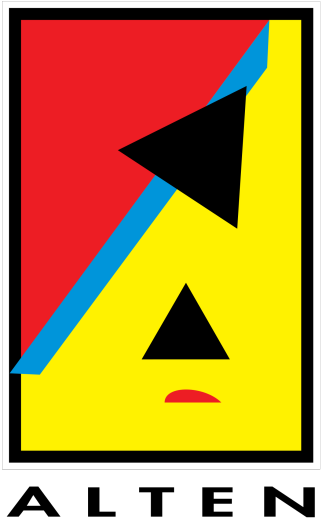


Drone meshnetwerk simulatie

Software Requirement Specificatie

Versie 2.0

Alten Nederland B.V.

Hogeschool van Arnhem en Nijmegen

HBO Technische Informatica - Embedded Software Developement

MWJ.Berentsen@student.han.nl

Studentnummer: 561399

Docent: J. Visch, MSc

Assessor: ir. C.G.R. van Uffelen

M.W.J. Berentsen

5 juni 2019

Inhoudsopgave

Begrippenlijst

Term	Beschrijving
Byte	Een samenstelling van 8 bits.
Gazebo	Simulatiesoftware met ondersteuning voor physics.
nRF24l01+	Radio transciever module werkzaam op de 2,4Ghz band.
Physics engine	Software die natuurwetten toepast op virtuele objecten.
Raspeberry Pi model 2B+	Micro computer geschikt voor prototyping.
Ros	Robot operating system. Wordt gebruikt voor de transportlaag naar zowel virtuele als gesimuleerde robots.
Router	Netwerkcomponent die berichten kan doorsturen.
Seriële verbinding	Een communicatieverbinding bekend van USB en RS232.
Simulatie software	Software die fysieke objecten nabootst in software.
Unittest	Geautomatiseerde test die functie test op resultaat.

Tabel 1: Begrippenlijst.

1 — Inleiding

1.1 Algemene beschrijving

Het volgende verslag betreft de Software Requirements Specification voor de afstudeerstage van Maurice Berentsen (hierna: student). Dit document volgt het document: *"Software Requirements Specification Template"* (Van Heesch, 2016).

Het doel van dit project is het zetten van de eerste stap in de ontwikkeling van het dronenetwerk. De eerste stap is ontwikkelen van een netwerkmodule voor het onderling verbinden van drones. Het is van belang dat het meshnetwerk van de drones snel kan reageren op uitval van netwerkpunten. Deze netwerkmodule moet zowel virtueel als fysiek gerealiseerd worden in dit project.

1.2 Doel van dit document

In dit document zullen de gebruikers, eisen en usecases van het systeem beschreven worden zodat het duidelijk is wie gebruik gaat maken van het systeem en welke functionaliteiten het systeem bevat.

1.3 Actoren en hun eigenschappen

In dit deel worden de actoren van het systeem omschreven. Elke actor wordt kort omgeschreven per paragraaf.

1.3.1 Dronecontroller

Een dronecontroller is de gebruiker die het systeem wil gebruiken om drones naar plekken toe te kunnen sturen. Hij wil hiermee een netwerk van onderling verbonden drones uitzetten.

1.3.2 Netwerkgebruiker

Een netwerkgebruiker wil het netwerk gebruiken om data te kunnen versturen naar een andere punt binnen of buiten het netwerk.

1.3.3 Algoritmetester

Een algoritmetester wil het netwerk gebruiken om de verdeling van drones te kunnen analyseren om tot een zo goed mogelijk verdeelalgoritme te komen.

1.4 Werkomgeving

Deze paragraaf omschrijft zowel de hardware- als softwareomgeving waarin dit project wordt uitgevoerd.

1.4.1 Ubuntu 18.04.2 LTS

In het project wordt gebruik gemaakt van Ubuntu 18.04.2 LTS vanwege de ondersteuning die het biedt voor ROS. Hoewel er een versie van ros opkomend is voor Windows zal hier op het moment geen rekening mee gehouden worden. De opgeleverde code wordt ontwikkeld en gecompileerd op een Ubuntu machine.

1.4.2 Raspberry Pi model B+

In het project wordt gebruik gemaakt van een Raspberry Pi als prototype board. Deze microcomputer is voorzien van een Broadcom BCM2836 SoC en heeft een 40 pin General Purpose Input Output (GPIO). Hiervan zijn 27 pinnen beschikbaar voor input, output maar ook geavanceerde technieken als PWM, SPI, I2C of een seriële verbinding. Verder biedt het twee 3,3 en twee 5 volt aansluitpunten aan.

1.4.3 NRF24

De NRF24 is gekozen door zijn prestaties ten opzichte van afstand. De mogelijkheid om een snelheid aan te kunnen bieden van 250 kbit/s op een afstand van 500 meter maakt deze module het meest geschikt voor dit project. Daarnaast kan de NRF24 tot zes adressen tegelijk onderhouden die kunnen schakelen tussen zenden en ontvangen. De NRF24 is in staat om per payload tot 32 bytes te versturen. Tenslotte is de NRF24 een transciever die werkt op een voltage van 3.3 volt waarbij de I/O pinnen 5 volt tolerant zijn wat het compatibel maakt met de Raspberry Pi model B+.

1.4.4 Drone

Hoewel in dit project drones een onmisbaar onderdeel zijn wordt er niet gesproken over een specifiek merk of type drone. De reden hiervoor is, omdat er geen focus ligt op een specifieke drone, en de student ook niet gecertificeerd is om te vliegen met zakelijke drones. Er zal gewerkt worden met gesimuleerde drones. Deze hebben een interface die in staat is om een drone te laten vliegen naar een specifiek coördinaat en de huidige locatie aan te geven.

1.4.5 ROS

ROS is middleware software die gebruikt wordt voor de aansturing en simulatie van robotica. In het geval van dit project wordt ROS gebruikt voor de communicatie naar de Drone toe, maar ook voor het simuleren van de NRF24 communicatie tussen de drones.

1.4.6 Gazebo

Gazebo is een opensource robot simulatie framework bijzonder geschikt voor het simuleren van robotica in outdoor omgevingen door de uitgebreide Physics Engine Support (Open Source Robotics Foundation, z. j.-b). In dit project wordt nu geen gebruik gemaakt van de physics engine. Doordat de netwerkonderdelen als virtuele onderdelen beschikbaar zijn in gazebo kunnen ze aangesloten worden op een gesimuleerde drones die wel realistisch vlieggedrag vertonen. Deze mogelijkheid was daarom ook een hoofdreden om Gazebo te gebruiken.

1.4.7 Catkin

Catkin is de ingebouwde standaard build tool van ROS. De tool is een combinatie van CMake en door ROS geschreven python scripts (GvdHoorn, 2017). Deze wordt gebruikt in het project om de simulatiesoftware mee te bouwen.

1.4.8 Gtest (Google unittest)

Google test wordt gebruik voor het schrijven van unit testen binnen ROS (Google, 2019). Omdat het gebruik van Gtest erg populair is binnen de ROS community maat dit het identificeren van problemen makkelijker maakt.

1.5 Ontwerp en implementatie beperkingen

De software wordt ontwikkelt in een ROS omgeving hiervoor worden de volgende eisen gesteld:

- Zie http://wiki.ros.org/ROS/Introduction#Operating_Systems voor ondersteuning van platformen. ROS draait op een Unix-based platform.
- De software dat voor het project is ontwikkelt, is op Ubuntu 18.04 gemaakt, aangeraden is dus ook om dit te gebruiken.

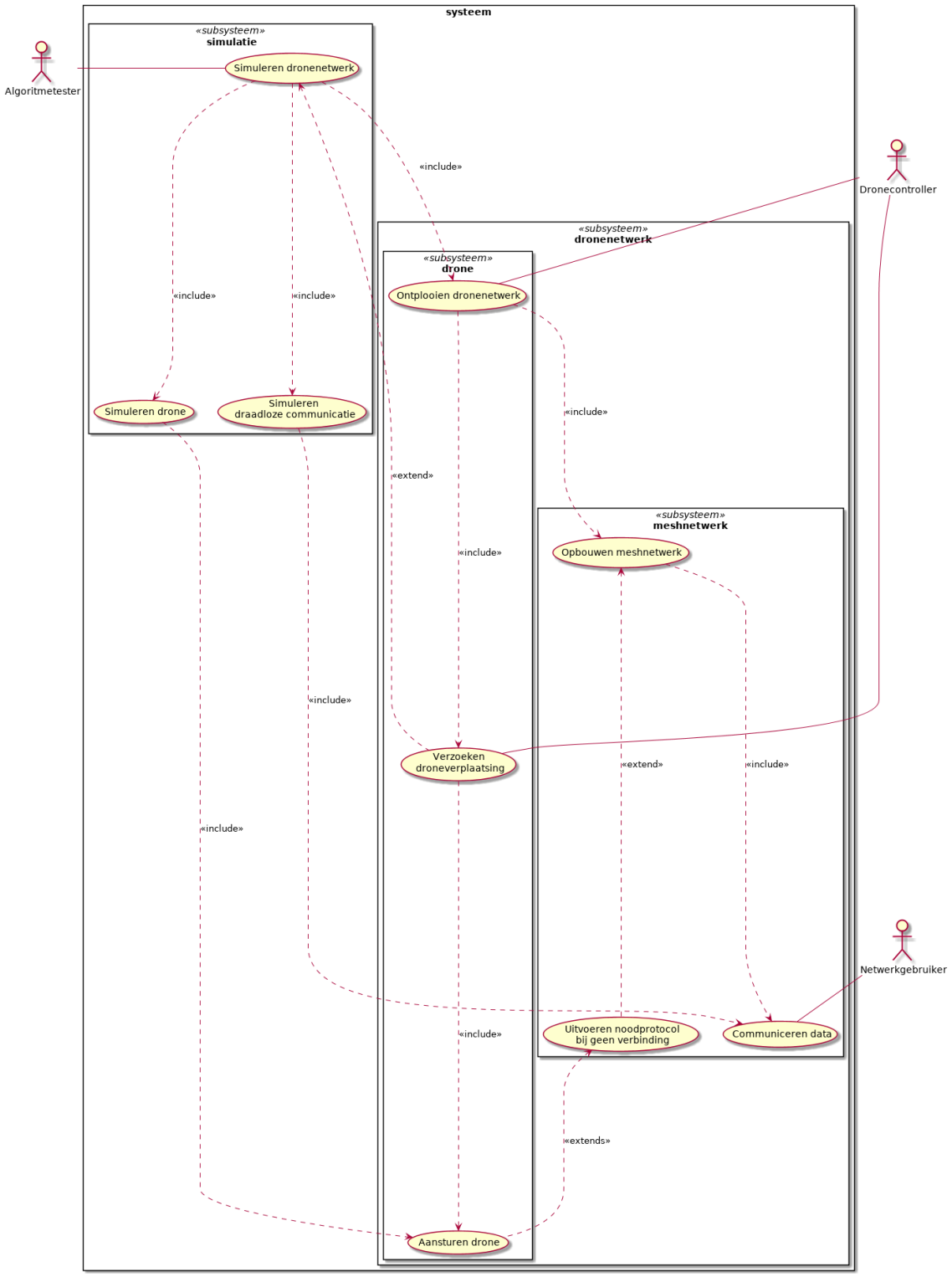
Om de drones te kunnen simuleren is er voor gekozen om gebruik te maken van Gazebo hierbij worden de volgende hardware eisen gesteld:

- Een GPU die werkt met OpenGL 3D accelerated driver.
- Een CPU welk op zijn minst een Intel i5 is of vergelijkbaar (Open Source Robotics Foundation, z. j.-a).
- Op zijn minst 500 MB vrije opslag ruimte.

1.6 Product Functies

In het onderstaande diagram 1.1 is te zien wie er betrokken is bij het systeem(actoren) en hoe ze het systeem gebruiken om hun doel te bereiken. Een netwerkgebruiker is maar geïnteresseerd in één usecase, hij wil namelijk alleen gebruik maken van het netwerk om

zijn data te versturen. De dronecontroller wil een dronenetwerk kunnen ontplooien en drones verzoeken om te verplaatsen. Een algoritmetester wil het dronenetwerk simuleren en wil daarom alleen die usecase uitvoeren. Deze usecase kan wel gebruik maken van dezelfde usecases als een dronecontroller maar doet dat dus in een gesimuleerde omgeving.



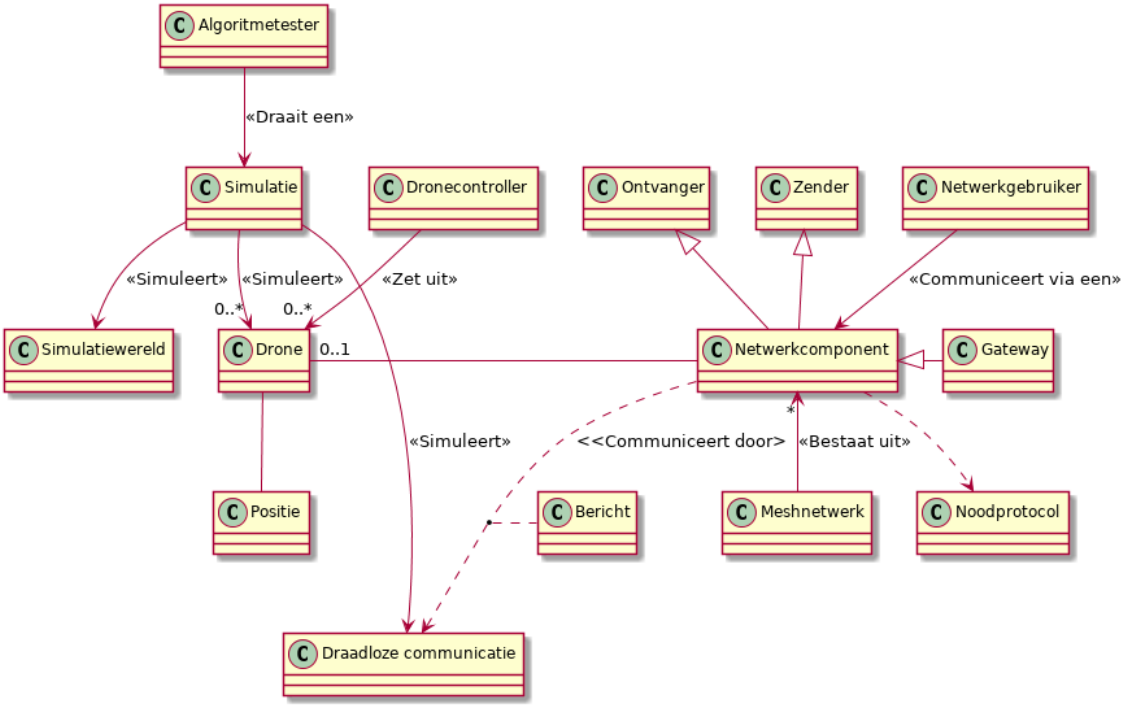
Figuur 1.1: Usecase diagram.

Usecase	Beschrijving
Simuleren dronenetwerk	Een actor wil een dronenetwerk simuleren. Hiervoor moeten drones en draadloze communicatie gesimuleerd worden.
Simuleren drone	Het systeem gaat een drone simuleren.
Simuleren draadloze communicatie	Het systeem gaat draadloze communicatie simuleren. Hiermee kan het data communiceren.
Communiceren data	Een actor geeft aan dat hij data wil communiceren via het netwerk.
Opbouwen meshnetwerk	Het systeem gaat een meshnetwerk opbouwen tussen aanwezige nodes.
Uitvoeren noodprotocol bij geen verbinding	Het systeem gaat wanneer er een bepaalde tijd geen verbinding is met een gateway een noodprotocol uitvoeren om het meshnetwerk te herstellen.
Ontplooien dronenetwerk	Een actor wil een dronenetwerk ontplooien over een gebied hiervoor worden drones aangestuurd.
Verzoeken droneverplaatsing	Een actor stuurt een verzoek tot het verplaatsen van een drone.
Aansturen drone	Het systeem stuurt een drone aan om zich te verplaatsen naar een locatie.

Tabel 1.1: Korte toelichting usecases.

2 Domeinmodel

Met een domeinmodel wordt de samenhang van de te ontwikkelen software in kaart gebracht. Hierna wordt in Beschrijving domeinmodel per class toelichting gegeven.



Figuur 2.1: Domeinmodel.

2.1 Beschrijving domeinmodel

Term	Beschrijving
Algoritmetester	Een actor die algoritmes wil testen in een simulatie.
Bericht	Een netwerkcomponent communiceert met berichten.
Draadloze communicatie	Berichten worden verstuurd door het gebruik van draadloze communicatie.
Drone	Een drone wordt ontplooit door een dronecontroller. Het beschikt altijd over een netwerkcomponent. Een drone heeft een positie.
Dronecontroller	Een dronecontroller is de actor die fysieke drones ontplooit.

Term	Beschrijving
Gateway	Een gateway is een netwerkcomponent die het netwerk van buitenaf benaderbaar maakt.
Meshnetwerk	Een meshnetwerk is een netwerktype waarin punten dynamisch met elkaar kunnen verbinden en meerdere connecties tegelijk aan kunnen gaan.
Netwerkcomponent	Een netwerkcomponent kan aangesloten worden op een drone. Het gebruikt draadloze communicatie om berichten te versturen. Het gebruikt een noodprotocol. Een netwerkcomponent is zowel een zender als een ontvanger.
Netwerkgebruiker	Een netwerkgebruiker is een actor die wil communiceren via het netwerk.
Noodprotocol	Een noodprotocol is een verzameling acties die ondernomen worden zodra een component geen verbinding meer heeft.
Ontvanger	Een ontvanger is een rol binnen het netwerk voor het ontvangen van berichten.
Positie	Een positie is een drie dimensionale plek in een ruimte.
Simulatie	Een simulatie wordt gebruik om de wereld na te bootsen. Een simulatie simuleert een wereld, drones en draadloze communicatie.
Simulatiewereld	Een simulatie wereld is een virtuele representatie van de echte wereld.
Zender	Een zender is een rol binnen het netwerk voor het zenden van berichten.

Tabel 2.1: Domeinmodel.

3 Usecase omschrijvingen

3.1 Usecase: Simuleren dronenetwerk

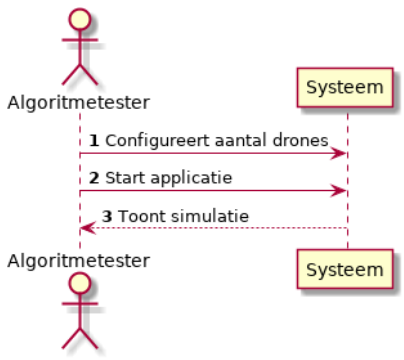
3.1.1 Fully-dressed usecase description

Usecase: Simuleren dronenetwerk.
Doel: De actor wil een dronenetwerk simuleren om algoritmes te testen zonder fysieke drones.
Beschrijving van de usecase: De usecase start een simulatie op waarin een instelbaar aantal drones gesimuleerd worden welke onderling kunnen communiceren via een gesimuleerde draadloze communicatieweg. Hiermee creëren zij een meshnetwerk.
Stakeholder: -
Primary actor: Algoritmetester.
Preconditions: Er is in een configuratie aangegeven hoeveel router- en gatewaydrones gesimuleerd moeten worden.
Postconditions: De simulatie van de drones en de communicatie is opgestart.

3.1.2 Basic Flow

Actor actie	System responsibility
1. Past simulatie parameters aan.	
2. Start de simulatie.	
	3. Toont de simulatie.

3.1.3 System Sequence Diagram



Figuur 3.1: System sequence diagram opstarten simulatie.

3.2 Usecase: Simuleren drone

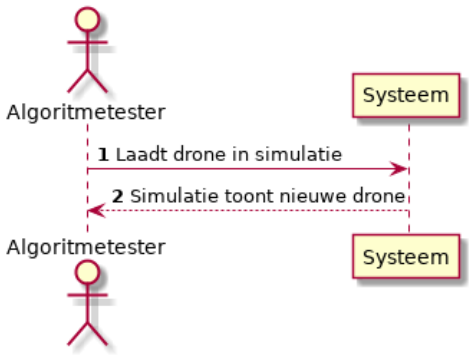
3.2.1 Fully-dressed usecase description

Usecase: Simuleren drone.
Doel: Het creëren van een virtuele representatie van een drone.
Beschrijving van de usecase: In de simulatie wil een actor een drone simuleren. Door het uitvoeren van deze usecase wordt er een drone in de simulatie geladen.
Stakeholder: Simuleren dronenetwerk.
Primary actor: Algoritmetester.
Preconditions: Er is een simulatiewereld aanwezig.
Postconditions: Een drone is ingeladen in de simulatiewereld.

3.2.2 Basic Flow

Actor action	System responsibility
1. Verzoekt een nieuwe drone.	2. Toont een nieuwe drone in de simulatie

3.2.3 System Sequence Diagram



Figuur 3.2: System sequence diagram opstarten drone simulatie.

3.3 Usecase: Simuleren draadloze communicatie

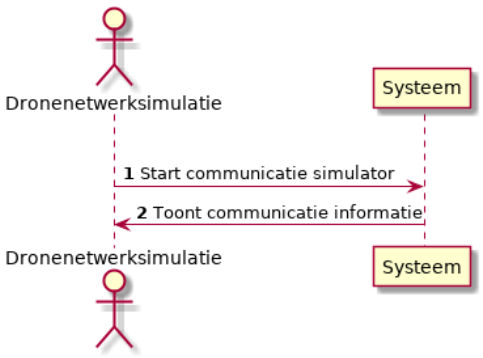
3.3.1 Fully-dressed usecase description

Usecase: Simuleren draadloze communicatie.
Doel: Het nabootsen van draadloze communicatie in een simulator.
Beschrijving van de usecase: In de simulatie wil een actor gebruik maken van draadloze communicatie. Om dit realistisch na te bootsen loopt elk bericht langs de draadloos simulator die bepaalt wat er met het bericht gebeurt.
Stakeholder: Simuleren dronenetwerk.
Primary actor: Algoritmetester.
Preconditions: -
Postconditions: Er is een simulator aanwezig die Communiceren data mogelijk maakt voor gesimuleerde NRF24's.

3.3.2 Basic Flow

Actor action	System responsibility
1. Start draadloze communicatie applicatie.	2. Start draadloze communicatie simulator.

3.3.3 System Sequence Diagram



Figuur 3.3: System sequence diagram opstarten draadloze communicatie simulatie.

3.4 Usecase: Communiceren data

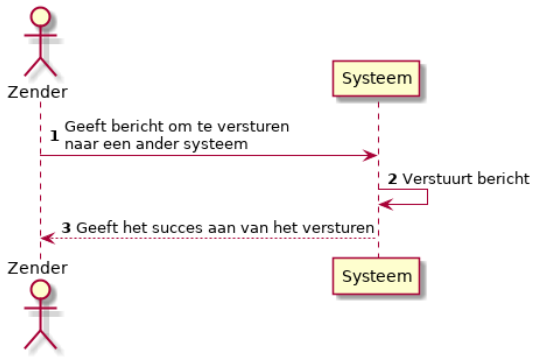
3.4.1 Fully-dressed usecase description

Usecase: Communiceren data.
Purpose: Deze usecase wordt gebruiken om data uit te wisselen tussen componenten
Beschrijving van de usecase: In het netwerk zullen componenten met elkaar willen communiceren. Deze usecase zal de data proberen te versturen.
Stakeholder: Simuleren draadloze communicatie, Opbouwen meshnetwerk.
Primary actor: Netwerkgebruiker, Algoritmetester, Dronecontroller
Preconditions: Communicatiemiddel van de zender is opgestart, de te versturen data is bekend, en het adres van de ontvanger is bekend.
Postconditions: Er is data verstuurd van een component naar een andere component.

3.4.2 Basic Flow

Actor action	System responsibility
1. Geeft bericht om te verzenden.	2. Geeft succes aan van het versturen

3.4.3 System Sequence Diagram



Figuur 3.4: System sequence diagram opstarten draadloze communicatie simulatie.

3.5 Usecase: Opbouwen meshnetwerk

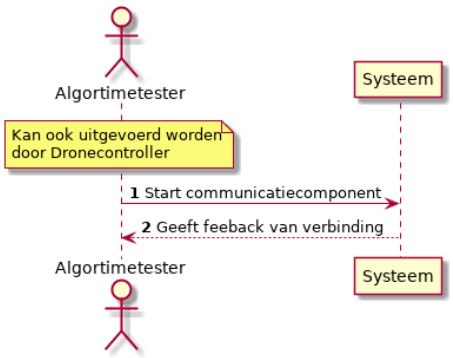
3.5.1 Fully-dressed usecase description

Usecase: Opbouwen meshnetwerk.
Doel: Het uitvoeren van de usecase zal een netwerk opzetten van onderling verbonden netwerkcomponenten.
Beschrijving van de usecase: Na het uitvoeren van de usecase moeten alle meshnetwerkcomponenten een verbinding hebben met de netwerkcomponenten die binnen hun bereik zijn. De componenten staan klaar om berichten door te sturen naar elkaar of naar een gateway voor externe adressen.
Stakeholder: Ontplooien dronenetwerk.
Primary actor: Algoritmetester, Dronecontroller.
Preconditions: Componenten zijn binnen bereik van elkaar, er is minstens één gateway aanwezig.
Postconditions: Er is een onderling verbonden netwerk opgebouwd.

3.5.2 Basic Flow

Actor action	System responsibility
1. Start netwerkcomponent.	2. Geeft feedback over verbinding met een gateway en/of andere punten.

3.5.3 System Sequence Diagram



Figuur 3.5: System sequence diagram opzetten meshnetwerk.

3.6 Usecase: Uitvoeren noodprotocol bij geen verbinding

3.6.1 Fully-dressed usecase description

Usecase: Uitvoeren noodprotocol bij geen verbinding.
Doel: Door het uitvoeren van een noodprotocol zijn één of meerdere drones in staat de verbinding te herstellen met een gateway. Hiervoor kan het gebruik maken van Aansturen drone.
Beschrijving van de usecase: Deze usecase is de uiterste stap die een netwerkcomponent onderneemt bij het verlies van een verbinding.
Stakeholder: Opbouwen meshnetwerk.
Primary actor: Algoritmetester.
Preconditions: Elke drone heeft een locatie van een gateway.
Postconditions: 1: Drone(s) is/zijn verplaatst naar een positie in verbinding met een gateway. 2: Drone(s) zijn verplaatst naar de positie van de gateway.

3.6.2 Basic Flow

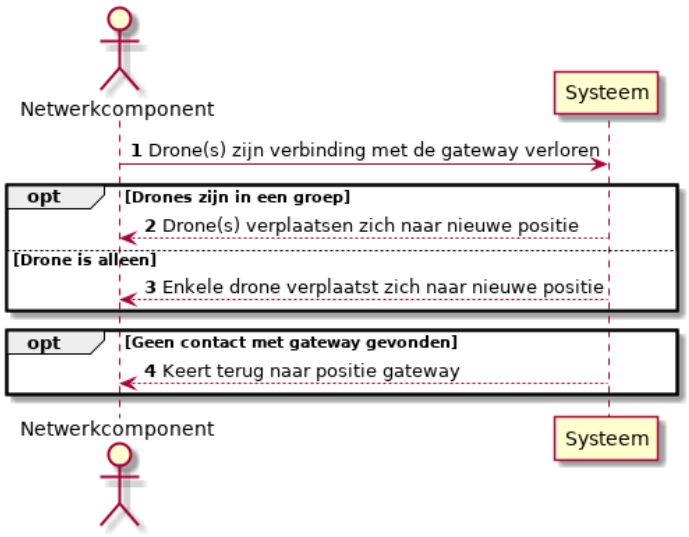
Actor action	System responsibility
1. Start noodprotocol.	2. Drones herpositioneren zich.

3.6.3 Alternative Flows

Actor action	System responsibility
	2. Drone positioneert zich op nieuwe locatie.

Actor action	System responsibility
	3. Drone(s) keren terug naar de positie van de gateway.

3.6.4 System Sequence Diagram



Figuur 3.6: System sequence diagram noodprotocol.

3.7 Usecase: Ontplooien dronenetwerk

3.7.1 Fully-dressed usecase description

Usecase: Ontplooien dronenetwerk.
Doel: Het doel van deze usecase is het ontplooien van een dronenetwerk, met een posities zal worden aangegeven waar naartoe te verplaatsen, ze moeten zelfstandig een netwerk opzetten.
Beschrijving van de usecase: Deze usecase zal alle drones aansturen om naar een vooraf bepaalde posities te vliegen. De drones zullen een meshnetwerk opbouwen waarover gecommuniceerd kan worden.
Stakeholder: Simuleren dronenetwerk.
Primary actor: Dronecontroller, Algoritmetester
Preconditions: Drones zijn voorzien van netwerkcomponenten en kunnen zich verplaatsen. Voor elke drone is een doel bekend.
Postconditions: Drones zijn verplaatst naar ingestelde positie en hebben een onderling netwerk opgebouwd.

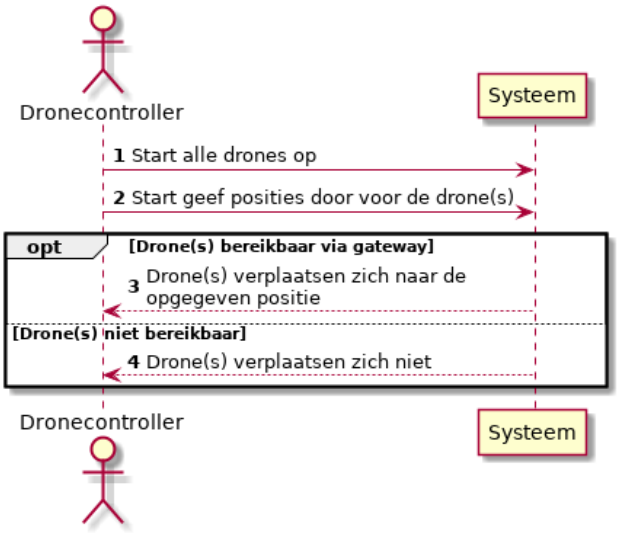
3.7.2 Basic Flow

Actor action	System responsibility
1. Start alle drones op.	
2. Geeft posities door voor de drones.	3. Stuurt elke individuele drone aan naar positie.
	4. Meshcomponenten bouwen netwerk op.

3.7.3 Alternative Flows

Actor action	System responsibility
	3. De drones die bereikbaar zijn in het netwerk verplaatsen zich.

3.7.4 System Sequence Diagram



Figuur 3.7: System sequence diagram noodprotocol.

3.8 Usecase: Verzoeken droneverplaatsing

3.8.1 Fully-dressed usecase description

Usecase: Verzoeken droneverplaatsing.
Doel: Deze usecase dient voor het verzoeken tot een verplaatsing van een drone.
Beschrijving usecase: Door deze usecase te gebruiken kan een individuele drone verzocht worden zich te verplaatsen.
Stakeholder: Ontplooien dronenetwerk, Simuleren dronenetwerk.
Primary actor: Dronecontroller, Algoritmetester.
Preconditions: Communicatieweg beschikbaar tot aan drone.
Postconditions: Drone gaat zich verplaatsen naar verzoeklocatie.

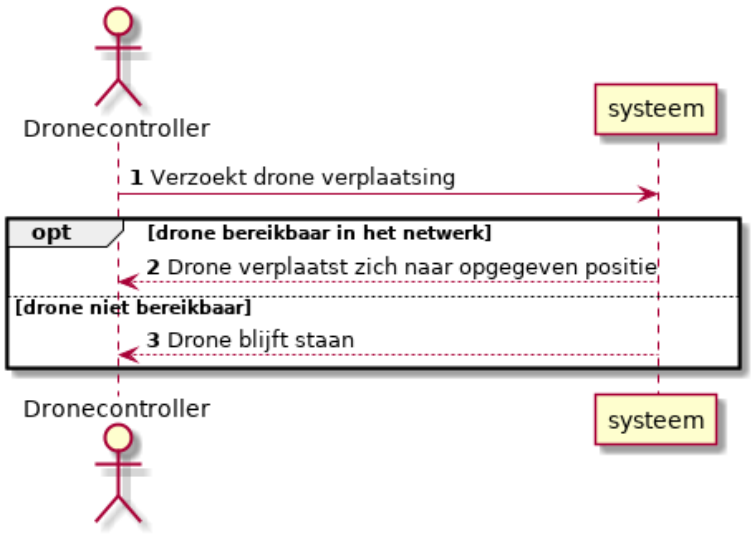
3.8.2 Basic Flow

Actor action	System responsibility
1. Verstuurt verzoek voor verplaatsing.	2. Drone verplaatst zich naar positie.

3.8.3 Alternative Flows

Actor action	System responsibility
	2. Drone blijft staan.

3.8.4 System Sequence Diagram



Figuur 3.8: System sequence verzoeken verplaatsing.

3.9 Usecase: Aansturen drone

3.9.1 Fully-dressed usecase description

Usecase: Aansturen drone.
Purpose: Het aansturen van een drone wordt gebruikt om een drone te verplaatsen.
Beschrijving van de usecase: Elke drone kan aangestuurd worden om zich te verplaatsen. Dit wordt altijd uitgevoerd vanaf het aangesloten netwerkcomponent. Daarom moet er vanuit een extern punt altijd een verzoek gestuurd worden voor een verplaatsing of vanaf de aangesloten module zelf komen.
Stakeholder: Verzoeken droneverplaatsing, Uitvoeren noodprotocol bij geen verbinding, Simuleren drone.
Primary actor: Dronecontroller, Algoritmetester.
Preconditions: Drone is in staat zich te verplaatsen.
Postconditions: Drone heeft zich verplaatst naar de verzochte positie.

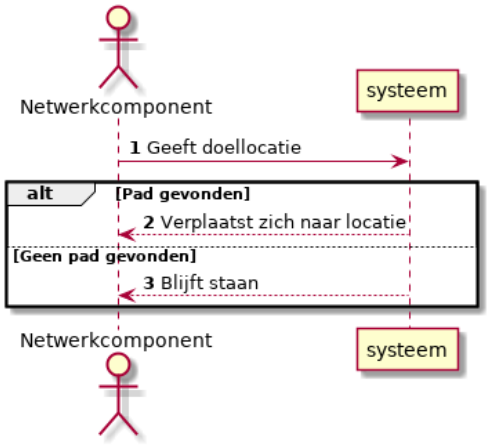
3.9.2 Basic Flow

Actor action	System responsibility
1. Stuurt doellocatie.	2. Verplaatst zich naar locatie.

3.9.3 Alternative Flows

Actor action	System responsibility
	2. Blijft op huidige locatie.

3.9.4 System Sequence Diagram



Figuur 3.9: System sequence aansturen drone.

4 Functionele requirements

Voor dit project zijn er requirements opgesteld. Deze requirements zijn opgesteld a.d.h.v. de MoSCoW methode, hiervoor is gekozen om de requirements te prioriteren. De Must requirements moeten voor het einde van het project gerealiseerd worden. De overige requirements hebben een lagere prioriteit en hieraan wordt pas gewerkt als alle Must requirements afgehandeld zijn. Onder de tabel is per requirement een toelichting te vinden.

Naam	Beschrijving	MoSCoW
SIMULATIE1	Simulatie representeert een wereld.	M
SIMULATIE2	Simulatie simuleert en visualiseert een abstracte drone.	M
SIMULATIE3	Simulatie simuleert de beweging van een drone.	M
SIMULATIE4	Simulatie simuleert tot 100 drones tegelijk.	S
SIMULATIE5	Simulatie simuleert draadloze communicatie.	M
SIMULATIE6	Simulatie simuleert een Raspberry Pi.	S
SIMULATIE7	Een gesimuleerde Raspberry Pi houdt zich aan de snelheidslimieten van een Raspberry Pi model 2B+.	W
SIMULATIE8	Simulatie kan met een configuratiebestand gestart worden.	S
DRONE1	Drone kan zich verplaatsen naar een positie.	M
DRONE2	Drone kan een route naar positie berekenen.	S
DRONE3	Drone kan een locatie aanbieden.	M
DRONE4	Drone is voorzien van een netwerkcomponent.	M
NETWERK1	Netwerk kan zelfstandig een meshnetwerk opbouwen.	M
NETWERK2	Netwerk heeft altijd minstens 1 gateway.	M
NETWERK3	Netwerk kan drones aansturen.	M
NETWERK4	Netwerk biedt een externe interface voor aansturing aan.	S
NETWERK5	Netwerk kan onderling data communiceren.	M
NETWERK6	Netwerk kan zichzelf herstellen.	M
NETWERK7	Meerdere gateways kunnen tegelijk aanwezig zijn in het netwerk.	C
ALGORITME1	Algoritmetester kan drones verdelen.	M
ALGORITME2	Algoritmetester kan een casus met een verdeling starten.	C

Tabel 4.1: Functionele requirements.

4.1 Toelichting functionele requirements

4.1.1 Simulatie

Onderstaand worden alle functionele requirements van de simulatie toegelicht.

SIMULATIE1: Simulatie representeert een wereld De simulatiecomponent moet een echte wereld nabootsen, er hoeven nog geen objecten in aanwezig te zijn. Een simulatiewereld bestaat uit een ruimte en maakt gebruiken van tijd. De simulatiewereld moet in staat zijn krachten afkomstig van zwaartekracht, botsingen en wrijving te verwerken.

SIMULATIE2: Simulatie simuleert en visualiseert een abstracte drone. De simulatie moet een abstracte versie van een drone representeren hierbij maakt de vorm niet uit zolang er maar een verschil te zien is tussen het type drone.

SIMULATIE3: Simulatie simuleert de beweging van een drone. De drone moet bewegen in de simulatie, het is belangrijk dat deze beweging visueel zichtbaar is voor analyse van het verplaatsingsgedrag.

SIMULATIE4: Simulatie simuleert tot 100 drones tegelijk. Om met meerdere drones te kunnen testen moet de simulatie tot aan honderd drones tegelijk kunnen simuleren dit hoeft niet visueel te gebeuren om de grafische kaart te ontlasten.

SIMULATIE5: Simulatie simuleert draadloze communicatie. Omdat er een simulatie van een meshnetwerk gerealisseerd wordt moet er een component aanwezig zijn die zorgt dat de draadloze communicatie volgens de juiste regels verloopt.

SIMULATIE6: Simulatie simuleert een Raspberry Pi. In het project wordt gebruik gemaakt van het Raspberry Pi model 2B+ deze moet virtueel gerepresenteerd worden door de simulatie.

SIMULATIE7: Een gesimuleerde Raspberry Pi houdt zich aan de snelheidslimieten van een Raspberry Pi model 2B+. De gesimuleerde Raspberry Pi zal zich niet houden aan de clock snelheid van een echte Raspberry Pi model 2B+.

SIMULATIE8: Simulatie kan met een configuratiebestand gestart worden. Om het starten van simulaties gemakkelijk te maken voor de algoritmetester moet het mogelijk zijn simulaties via een configuratie op te starten.

4.1.2 Drone

Onderstaand worden alle functionele requirements van de drone toegelicht.

DRONE1: Drone kan zich verplaatsen naar een positie Het netwerk rekent op de mogelijkheid van een drone om zich te kunnen verplaatsen om zo het herstellend vermogen te vergroten.

DRONE2: Drone kan een route naar positie berekenen Een drone moet op basis van een ontvangen doel zich kunnen verplaatsen naar een positie. Om deze verplaatsingen uit te kunnen voeren maakt hij gebruik van een zelf berekende route.

DRONE3: Drone kan een locatie aanbieden Een netwerkmodule is zelf niet voorzien van een component voor het ophalen van een locatie daarvoor maakt hij gebruik van de drone. Daarom moet een drone in staat zijn om zijn locatie door te geven.

DRONE4: Drone is voorzien van een meshnetwerkcomponent Een drone moet worden voorzien van een meshnetwerkcomponent om zo alle drones wanneer mogelijk met elkaar te kunnen laten communiceren.

4.1.3 Netwerk

Onderstaand worden alle functionele requirements van het netwerk toegelicht.

NETWERK1: Netwerk kan zelfstandig een meshnetwerk opbouwen De opzetter van het netwerk wil zich niet bezig hoeven houden met het onderling verbinden van de drones. De netwerkmodules moeten zelf een netwerk opzetten zonder tussenkomst van menselijke actoren.

NETWERK2: Netwerk biedt altijd minstens 1 gateway aan Om het netwerk toegang te kunnen geven tot een extern punt moet er tenminste altijd een gateway aanwezig zijn in het netwerk.

NETWERK3: Netwerk kan drones aansturen Het netwerk is zelf in staat drones aan te sturen wanneer er een verplaatsing vereist is, het netwerk kan deze beslissing zelfstandig maken.

NETWERK5: Netwerk kan onderling data communiceren De essentie van het netwerk is natuurlijk de mogelijkheid om onderling data te kunnen communiceren. Dit wordt gebruikt voor het opzetten en onderhoud van het netwerk. Daarnaast kunnen gebruikers gebruik maken van het netwerk om data te communiceren.

NETWERK6: Netwerk kan zichzelf herstellen Wanneer het netwerk zijn onderlinge verbinding verliest moet het deze zelf kunnen herstellen, dit gebeurt door een nieuwe route te creëren of door de drone(s) te herverdelen.

NETWERK7: Meerdere gateways kunnen tegelijk aanwezig zijn in het netwerk Om de stabiliteit van het netwerk te vergroten moet het mogelijk zijn het netwerk te ontplooien met meerdere gateways.

4.1.4 Algoritme testapplicatie

Onderstaand worden alle functionele requirements van de testapplicatie voor het algoritme toegelicht.

ALGORITME1: Algoritmetester kan drones verdelen. Een algoritmetester moet om zijn algoritme te kunnen testen drones kunnen verplaatsen, dit zal via een aangeboden interface gebeuren.

ALGORITME2: Algoritmetester kan een casus met een verdeling starten Een algoritmetester moet casus situaties kunnen testen deze casus zal aangeroepen worden via een API.

5 Non-functionele requirements

5.1 Performance efficiency

Naam	Beschrijving
PERF1	Simulatiesnelheid moet minimaal 50 procent van de realiteit hebben.
PERF2	Een drone kan binnen 10 seconden zijn positie bepalen.

Tabel 5.1: Non-functionele requirements performance.

5.2 Security

Naam	Beschrijving
SEC1	Een drone mag alleen door een netwerkmodule aangestuurd worden.

Tabel 5.2: Non-functionele requirements security.

5.3 Reliability

Naam	Beschrijving
REL1	Het netwerk mag maximaal 5 procent van zijn berichten verliezen.
REL2	Een router heeft altijd direct of indirect een verbinding met een gateway.

Tabel 5.3: Non-functionele requirements reliabilty.

5.4 Timeliness

Naam	Beschrijving
TIME1	Het netwerk moet binnen 30 seconden detecteren dat er een verbinding verloren is.
TIME2	Het netwerk moet minus de tijd van het verplaatsen van de drones zich herstellen binnen 2 minuten.

Tabel 5.4: Non-functionele requirements timeliness.

5.5 Quality

Naam	Beschrijving
QUA1	Het netwerk moet een minimale snelheid van 100 kbit/s ondersteunen.

Tabel 5.5: Non-functionele requirements quality.

5.6 Scalability

Naam	Beschrijving
SCALE1	Een netwerkmodule past op elke drone die voldoet aan de gevraagde interface.

Tabel 5.6: Non-functionele requirements scalability.

6 — Bewijslast gestelde requirements

Om aan te tonen dat zowel functionele als niet functionele requirements zijn behaald wordt er gebruik gemaakt van verwijzingen. Dit gebeurt door het verwijzen naar een usecase maar ook door het verwijzen naar andere documenten, demonstraties of code in de onderstaande tabel.

Requirement	Bewijs
SIMULATIE1	Er is onderzocht welke simulatiesoftware gebruikt moet worden om aan deze eis te voldoen. Dit is terug te lezen in bijlage A.1. De gekozen software is gazebo.
SIMULATIE2	In het onderzoek van bijlage A.1 is bepaald wat een abstracte drone inhoudt. Dit is gerealiseerd in de code van de class VirtualDrone te vinden in Appendix B. De werking wordt toegelicht in bijlage A.2.
SIMULATIE3	Dit is gerealiseerd in de code van de class DroneEngine te vinden in Appendix B. De werking wordt toegelicht in bijlage A.2.
SIMULATIE4	De simulatie kan met honderd drones tegelijk draaien, dit is te zien in bijlage C.1.
SIMULATIE5	In de demonstratie video's te vinden in bijlage E is te zien dat de drones met elkaar communiceren. Dit is gerealiseerd in de code van de folder drone_meshnetwork_simulation/src/ros/WirelessSimulation te vinden in Appendix B.
SIMULATIE6	Dit is gerealiseerd in de code van de folder drone_meshnetwork_sim -ulation/src/gazebo/RaspberryPi te vinden in Appendix B.
SIMULATIE8	De simulatie kan geconfigureerd worden met factory.world bestand te vinden in Appendix B.
DRONE1	Dit is gerealiseerd in de code van de class DroneEngine te vinden in Appendix B. De werking wordt toegelicht in bijlage A.2.
DRONE2	Dit is gerealiseerd in de code van de class DroneEngine te vinden in Appendix B. De werking wordt toegelicht in bijlage A.2.
DRONE3	Dit is gerealiseerd in de code van de class DroneEngine te vinden in Appendix B. De werking wordt toegelicht in bijlage A.2.
DRONE4	In het onderzoek van bijlage A.1 is bepaald wat een abstracte drone inhoudt. Dit is gerealiseerd in de code van de class VirtualDrone te vinden in Appendix B. De werking wordt toegelicht in bijlage A.2.
NETWERK1	In de demonstratie videos te vinden in bijlage E, C.1 is te zien dat de netwerkmodules zelfstandig een netwerk opbouwen in een simulatie. De werking op de raspberry is te zien in bijlage D.1
NETWERK2	In het SDD te vinden in bijlage A.2 is vinden dat de routers altijd bezig zijn met zoeken naar een gateway.
NETWERK3	In de bijlage D.1 is te zien dat er locatie via het netwerk naar een drone verstuurd wordt.

Requirement	Bewijs
NETWERK4	Dit is gerealiseerd in de code van de class InternetGateway te vinden in Appendix B. Werkend te zien in bijlage D.1
NETWERK5	In de demonstratie video's te vinden in bijlage E is te zien dat de drones met elkaar communiceren. Dit is gerealiseerd in de code van de folder drone_meshnetwork_simulation/src/ros/WirelessSimulation te vinden in Appendix B.
NETWERK6	In de demonstratie video's te vinden in bijlage E is te zien dat de netwerkmodules het netwerk zelfstandig herstellen. Dit doen ze door fysieke verplaatsing of het zoeken naar een nieuwe routeringsroute.
NETWERK7	In de video van de bijlage C.2 is te zien hoe meerdere gateways gebruikt worden in een netwerk.
ALGROITME1	Aan de hand van de API aangeboden door de dronemanager kan er een drone naar een specifieke positie gestuurd worden.
ALGROITME2	Aan de hand van de API aangeboden door de dronemanager kan de algoritmetester een casus starten. In de video van bijlage C.2 kan gezien worden hoe drone in een casus geplaatst worden.
PERF1	In de video's in de bijlage C en E worden simulaties getoond. Deze lopen allen boven de 50 procent simulatie snelheid.
PERF2	n.v.t.
SEC1	In bijlage A.2 is zichtbaar dat de interface voor aansturing alleen mogelijk is vanaf een netwerkcomponent. Bijlage D.1 laat zien dat er een aansturing wordt uitgevoerd door het gebruik van een REST request. Deze loopt via een gateway naar het te verplaatsen netwerkcomponent.
REL1	Het onderzoek in bijlage A.1 toont aan dat er op een afstand van 500 meter gemiddeld 2,8 procent van de berichten verloren gaan. Daarmee wordt de eis behaald.
REL2	In het SDD uit bijlage A.2 wordt beschreven dat een router altijd op zoek zal gaan naar een gateway.
TIME1	De software controleert elke 10 seconden of er een verbinding bestaat naar de gateway. Hiermee wordt de eis behaald.
TIME2	Zodra het netwerk 30 seconden geen verbinding heeft wordt actie ondernomen, hierbij bestaat de kans dat het huidige verplaatsingalgoritme de twee minuten niet haalt.
QUA1	De gebruikte antenne ondersteund een minimale doorvoersnelheid van 250 kbit. Het onderzoek uit bijlage A.1 toont aan dat er op de maximale afstand van 500 meter gemiddeld 87.1 kbit behaald wordt. Daarmee is de eis niet behaald.
SCALE1	n.v.t.

Tabel 6.1: Referentie tabel bewijslast requirements.

Literatuur

Google. (2019, juni). *Google test.* Op 5 juni 2019 verkregen van https://github.com/google/googletest

GvdHoorn. (2017, juli). *catkin - ros wiki.* Op 5 juni 2019 verkregen van http://wiki.ros.org/catkin

Van Heesch, U. (2016, 21 september). *Software requirements specification template.*

Open Source Robotics Foundation. (z. j.-a). *Beginner: Overview.* Op 5 juni 2019 verkregen van http://gazebosim.org/tutorials?tut=guided_b1&cat=

Open Source Robotics Foundation. (z. j.-b). *Gazebo.* Op 5 juni 2019 verkregen van http://gazebosim.org/

A Documenten

Voor het aantonen van behaalde requirements wordt gebruik gemaakt van de volgende documenten.

A.1 Onderzoeksrapport Drone meshnetwerk simulatie.pdf

A.2 SoftwareDesignDocument.pdf

B — Broncode

Voor het aantonen van behaalde requirements wordt gebruik gemaakt van code te vinden in via het pad:

../Code/

C — Videos drone simulatie

Video's in ./Bijlagen/

C.1 gateway drone en 99 routerdrones.mp4

Video van 99 drones die verbinden met een gateway drone te vinden in:

C.2 GatewayWissel.mp4

Video van drones die zich verspreiden, hier zijn twee gateways bij aanwezig. Vervolgens wordt een essentieel punt uit gezet. De daarom verbonden router wisselt van gateway.

D — Videos fysiek netwerkmodule

D.1 Fysieke router en drone.mp4

Video van fysieke router en gateway die verbinden met elkaar:
./Bijlagen/

E — Videos simulatie netwerkherstel door drone verplaatsing

De video's van deze bijlagen zijn te vinden door het volgende pad te volgen:
./Bijlagen/Videos simulatie netwerkherstel door drone verplaatsing/

E.1 enkele verloren drone situatie 1.mp4

E.2 enkele verloren drone situatie 2.mp4

E.3 groep verloren drones situatie 1.mp4

E.4 groep verloren drones situatie 2.mp4

E.5 groep verloren drones situatie 3.mp4