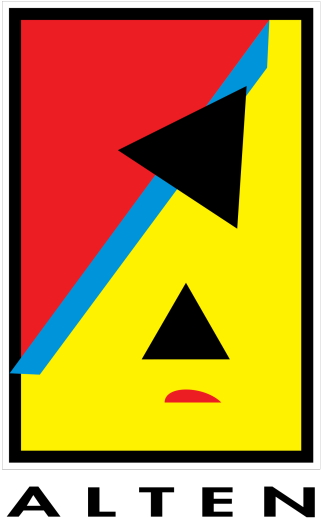


Drone meshnetwerk simulatie

Software Requirement Specificatie

Versie 1.0

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

17 mei 2019

Inhoudsopgave

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)

1.2 Doel van dit document

Het doel is dat de lezer aan het einde van dit document begrijpt hoe de functionaliteit geïmplementeerd zal worden in het op te leveren product. Het document laat zien hoe de flow van het programma loopt en welke classes in de software worden gebruikt. Een lezer van dit document zou met de juiste technische kennis de volledige applicatie moeten kunnen maken zoals het bedacht is door de ontwerper.

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 drone controller 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 kunnen ontplooien

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 optimaal verdeel algoritme 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 dit omdat er ondersteuning nodig is 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 bord. 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 of output of geavanceerde technieken als PWM, SPI, I2C en of een seriële verbinding. Verder biedt het stroompunten aan van 3,3 of 5 volt.

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 minimaal 250 kbitps 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 essentieel 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 dus gewerkt worden met gesimuleerde drones. Deze hebben een interface die in staat is om een drone te laten vliegen naar een specifiek coördinaat en kan de huidige locatie aangeven.

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. In dit project wordt momenteel geen gebruik gemaakt van de physics engine maar doordat de onderdelen als virtuele onderdelen beschikbaar zijn in gazebo kunnen ze zo aangesloten worden op een beter gesimuleerde drone. 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 scripst. 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. Dit omdat het gebruikt van Gtest er populair is binnen de ROS community en wat vinden van oplossingen van problemen makkelijker maakt.

1.5 Ontwerp en implementatie beperkingen

De software wordt ontwikkelt in ROS omgeving hiervoor zijn de volgende eisen:

- 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.

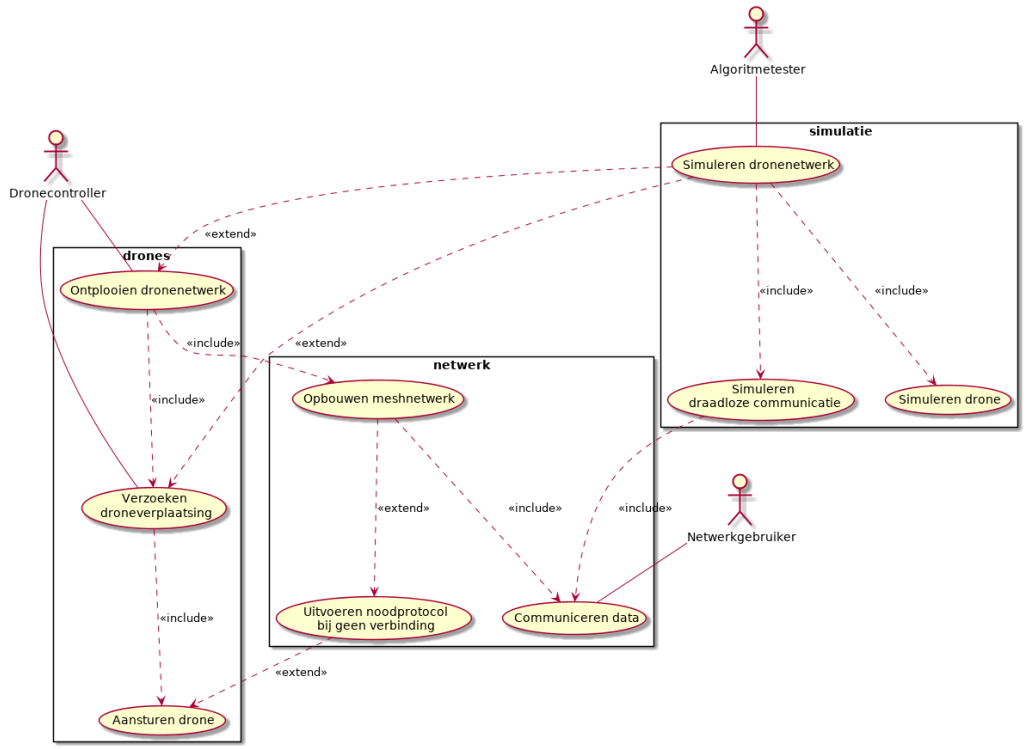
Om de drones te kunnen simuleren is er voor gekozen om gebruik te maken van Gazebo hierbij worden de volgende hardware eisen gesteld

- Een GPU die werkt met OpenGL 3D accelerated driver.
- Een CPU dat op z'n minst Intel i5 is of vergelijkbaar. (bron)
- Op z'n 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 zij het systeem gebruiken om hun doel te gebruiken. 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 drone verzoeken om te verplaatsen. Een algoritme tester wil het dronenetwerk simuleren

en wil daarom alleen die usecase uitvoeren deze usecase kan wel gebruik maken van dezelfde usecases als een dronecontroller.



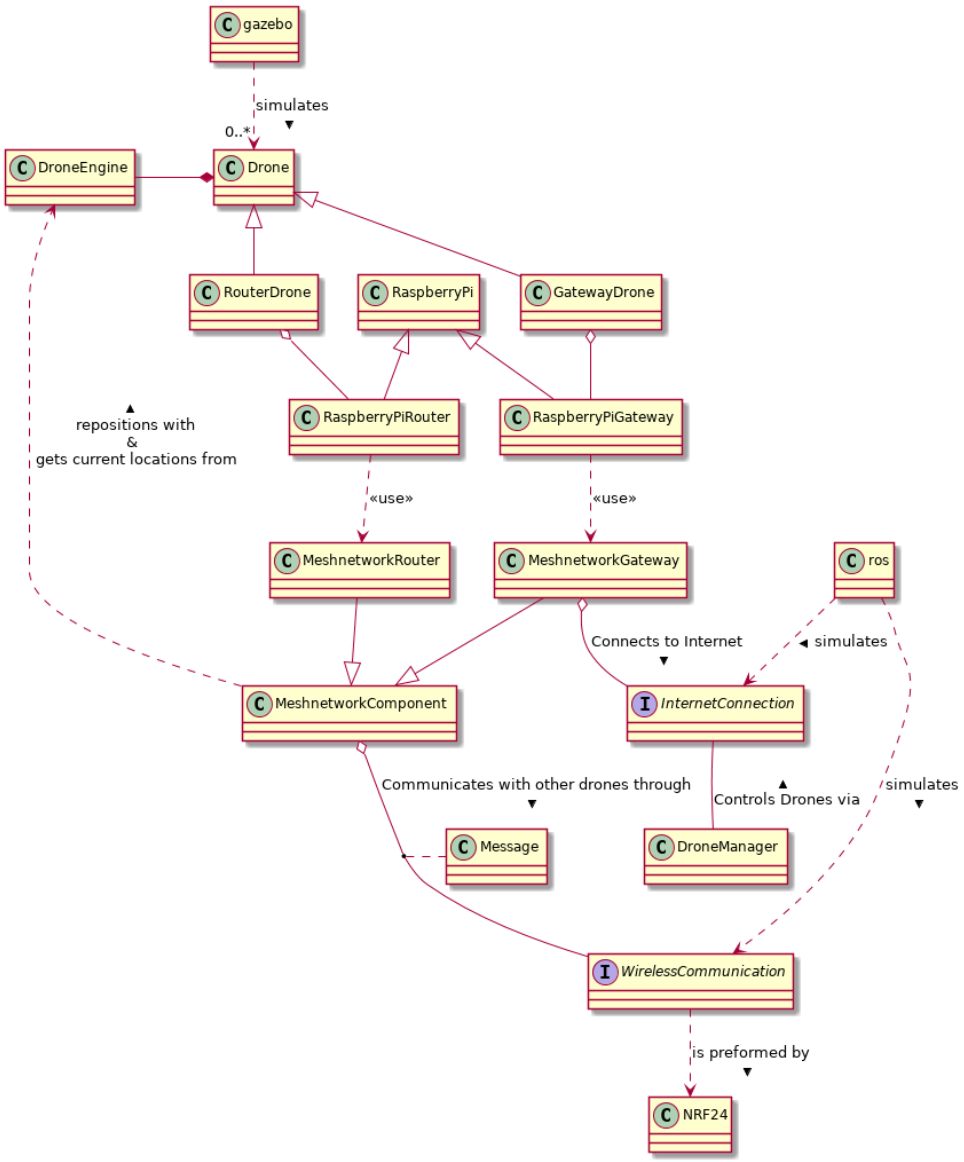
Figuur 1.1: Usecase diagram

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 Domein Model

Aan de hand van een domeinmodel wordt de samenhang van de te ontwikkelen software in kaart gebracht. In de hier opvolgende Beschrijving domeinmodel wordt per class toelichting gegeven.



Figuur 2.1: Domeinmodel

2.1 Beschrijving domeinmodel

Naam	Beschrijving
gazebo	Gazebo is de simulatie omgeving voor physics.
Drone	Een drone kan zichzelf verplaatsen in een ruimte met een DroneEngine.
DroneEngine	Een drone engine wordt gebruikt door de drone voor verplaatsing.
RouterDrone	Een RouterDrone is een drone voorzien van een RaspberryPiRouter.
GatewayDrone	Een RouterGateway is een drone voorzien van een RaspberryPiGateway.
RaspberryPi	Een RaspberryPi is een mircocomputer.
RaspberryPiRouter	Is een RaspberryPi voorzien van MeshnetworkRouter. software
RaspberryPiGateway	Is een RaspberryPi voorzien van MeshnetworkGateway. software
MeshnetworkComponent	Is een onderdeel binnen een meshnetwerk.
MeshnetworkRouter	Is een meshcomponent in staat berichten binnen het netwerk door te sturen.
MeshnetworkGateway	Is een meshcomponent met een verbinding extern van het netwerk.
InternetConnection	Is een verbinding naar het wereldwijde web.
DroneManager	Een manager voor het aansturen van drones.
ros	Middleware voor het aansturen van fysieke of gesimuleerde drones.
Message	Een bericht gebruikt voor communicatie.
WirelessCommunication	Communicatie type die door de lucht verloopt.
NRF24	Een radio transceiver bruikbaar voor communicatie.

Tabel 2.1: Toelichting domeinmodel

3 Use-case omschrijvingen

3.1 Usecase: Simuleren dronenetwerk

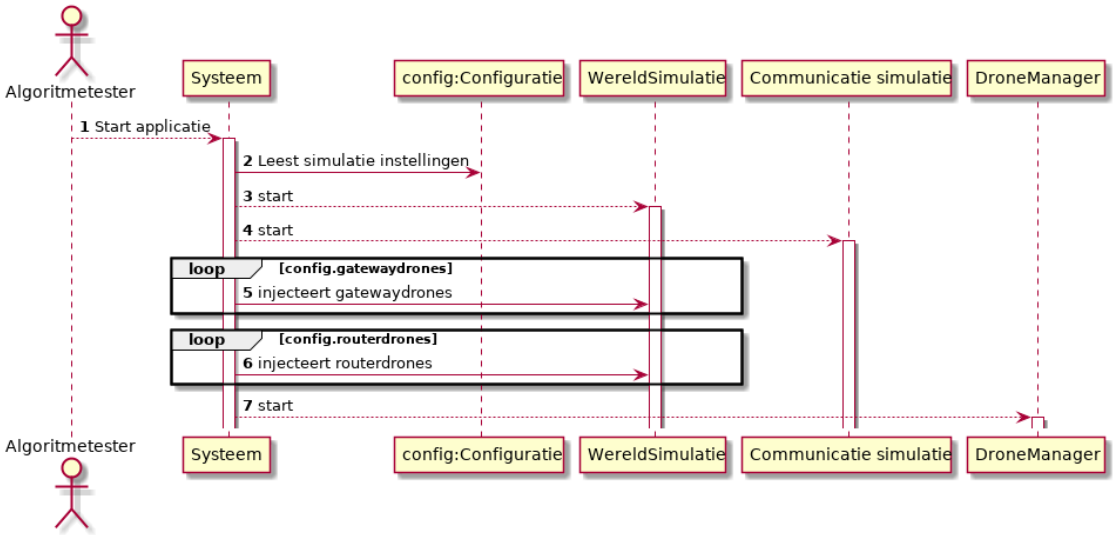
3.1.1 Fully-dressed usecase description

Usecase: Simuleren dronenetwerk
Doel: De actor wil een dronenetwerk simuleren om algritmes 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.
Primary actor: Algoritmetester
Preconditions: Er is aangegeven hoeveel routers en gateway gesimuleerd moeten worden.
Postconditions: De simulatie van de drones en de communicatie is opgestart.

3.1.2 Basic Flow

Actor actie	System responsibility
1. Start de applicatie.	2. Leest het configuratie bestand uit.
	3. Start de wereld simulatie.
	4. Start de communicatie simulatie.
	5. Laadt het opgegeven aantal gatewaydrones de simulatie in.
	6. Laadt opgegeven aantal routerdrones de simulatie in.
	7. Start de dronemanager.

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 welke de rol van gateway of router kan hebben.
Primary actor: Simuleren dronenetwerk
Preconditions: Er is een simulatiewereld aanwezig.
Postconditions: Een drone is ingeladen in de simulatiewereld.

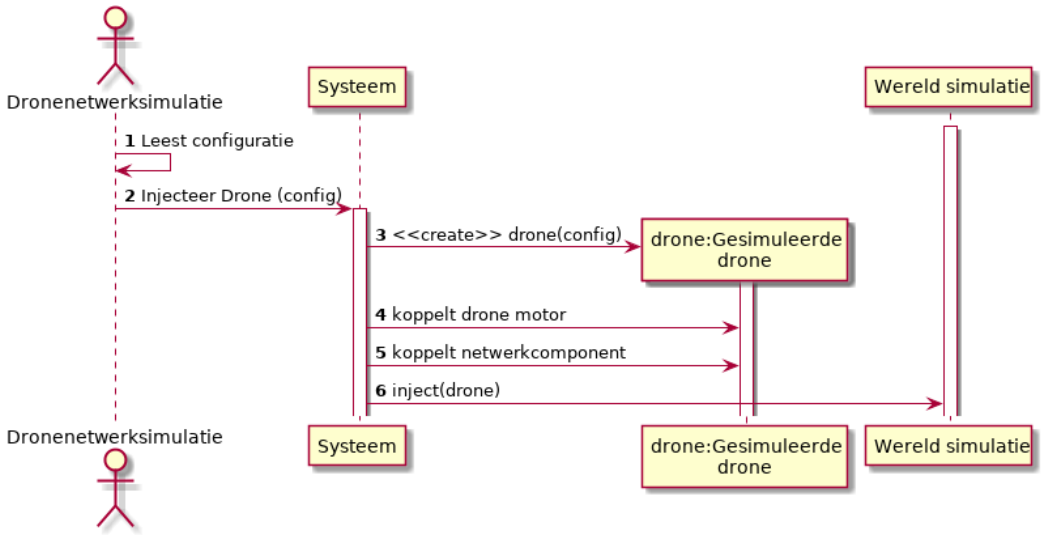
3.2.2 Basic Flow

Actor action	System responsibility
1. Leest configuratie uit.	
2. Verzoekt het injecteren van een gatewaydrone.	3. Leest de drone parameters uit.
	4. Koppelt een motor aan de gesimuleerde drone.
	5. Koppelt een gatewaycomponent aan de gesimuleerde drone.
	6. Injecteert de drone de gesimuleerde wereld in.

3.2.3 Alternative Flows

Actor action	System responsibility
2. Verzoekt het injecteren van een routerdrone.	3. Leest de drone parameters uit.
	5. Koppelt een router component aan de gesimuleerde drone.

3.2.4 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

Use Case: 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 gebeurd.
Primary actor: Simuleren dronenetwerk.
Preconditions:
Postconditions: Er is een simulator aanwezig die Communiceren data mogelijk maakt voor gesimuleerde NRF24's.

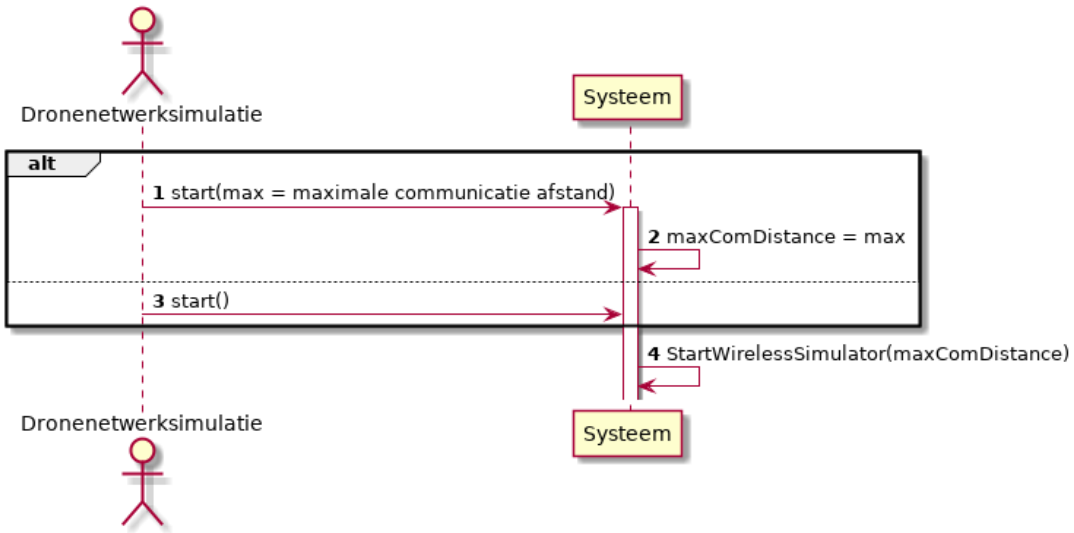
3.3.2 Basic Flow

Actor action	System responsibility
1. Start applicatie met maximale afstand parameter.	2. Stelt de meegegeven waarde in voor de maximale communicatie afstand.
	3. Start draadloze simulator.

3.3.3 Alternative Flows

Actor action	System responsibility
1. Start applicatie zonder parameters.	2. Stelt de standaard waarde in voor de maximale communicatie afstand.

3.3.4 System Sequence Diagram



Figuur 3.3: System sequence diagram opstarten draadloze communicatie simulatie

3.4 Use case: Communiceren data

3.4.1 Fully-dressed usecase description

Use Case: 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.
Primary actor: Netwerkgebruiker, Simuleren draadloze communicatie, Opbouwen meshnetwerk
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.

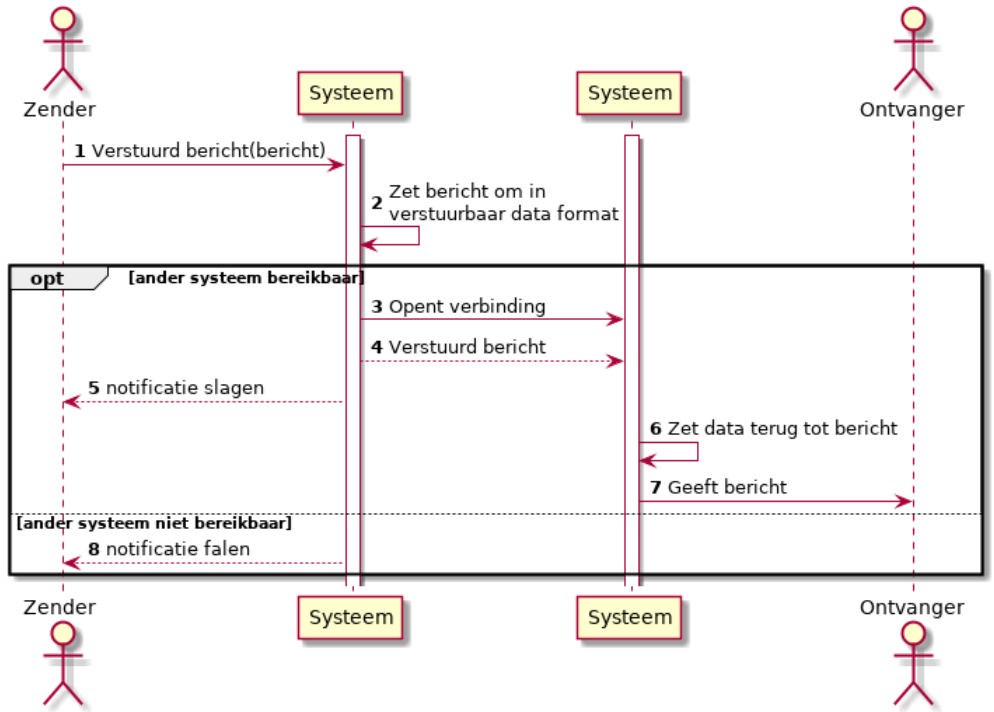
3.4.2 Basic Flow

Actor action	System responsibility
1. Geeft bericht om te verzenden.	2. Zet bericht om in overdraagbare data.
	3. Opent verbinding met ontvanger.
	4. Verstuurd bericht naar ontvanger.
5. Krijgt notificatie van slagen.	6. Ontvangend syteem zet data terug naar bericht.
	7. Ontvanger ontvangt bericht.

3.4.3 Alternative Flows

Actor action	System responsibility
	3. Kan geen verbinding openen met ontvanger.
4. Krijgt notificatie van het niet slagen.	

3.4.4 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 meshnetwerkcomponenten.
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.
Primary actor: Ontplooien dronenetwerk
Preconditions: Componenten zijn binnen bereik van elkaar, er is minstens één gateway aanwezig.
Postconditions: Er is een onderling verbonden netwerk opgebouwd.

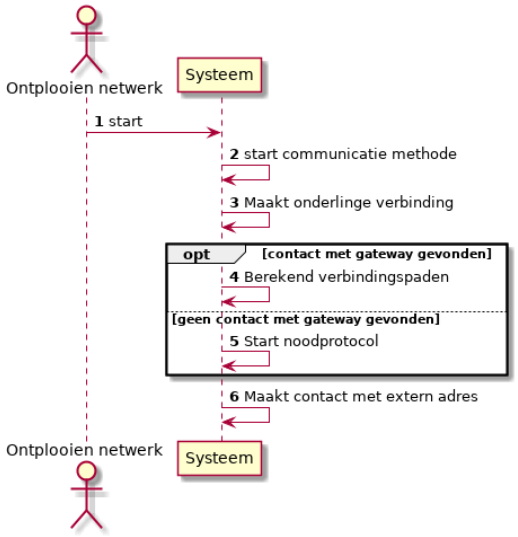
3.5.2 Basic Flow

Actor action	System responsibility
1. Start applicatie.	2. Start communicatie methode.
	3. Componenten maken verbinding met elkaar.
	4. Componenten berekenen verbindingspaden
	5. Maak contact met extern adres.

3.5.3 Alternative Flows

Actor action	System responsibility
	3. Componenten maken geen directe of indirecte verbinding met een gateway.
	4. Componenten starten Uitvoeren noodprotocol bij geen verbinding.

3.5.4 System Sequence Diagram



Figuur 3.5: System sequence diagram opzetten meshnetwerk.

3.6 Use case: Uitvoeren noodprotocol bij geen verbinding

3.6.1 Fully-dressed usecase description

Use Case: 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 meshcomponent.
Primary actor: Opbouwen meshnetwerk
Preconditions: Elke drone heeft ooit direct of indirect verbinding gehad met een gateway.
Postconditions: Drone(s) is/zijn herplaatst naar een positie in verbinding met een gateway.

3.6.2 Basic Flow

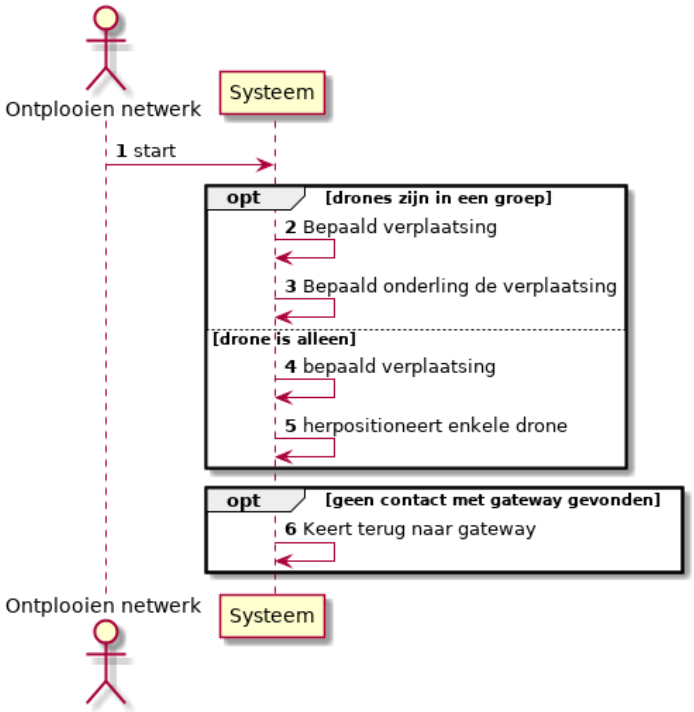
Actor action	System responsibility
1. Start noodprotocol.	2. Kijkt of node alleen is of samen.
	3. Drones bepalen verplaatsing.
	4. Drones herpositioneren zich.
	5. Alle drones hebben direct of indirect verbinding met een gateway.

3.6.3 Alternative Flows

Actor action	System responsibility
	3. Enkele drone bepaalt verplaatsing.
	4. Enkele drones herpositioneert zich.

Actor action	System responsibility
	5. Drone(s) heeft/hebben nog geen verbinding met gateway.
	6. Drone(s) verplaatst/verplaatsen zich terug naar positie van 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

Use Case:Ontplooien dronenetwerk
Doel: Het doel van deze usecase is het ontplooien van een dronenetwerk, met een casus 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.
Primary actor: Dronecontroller, Simuleren dronenetwerk
Preconditions: Drones zijn voorzien van meshcomponenten en kunnen zich verplaatsen. Er zijn genoeg drones aanwezig voor de casus.
Postconditions: Drones zijn verplaatst naar ingestelde positie en hebben een onderling netwerk opgebouwd.

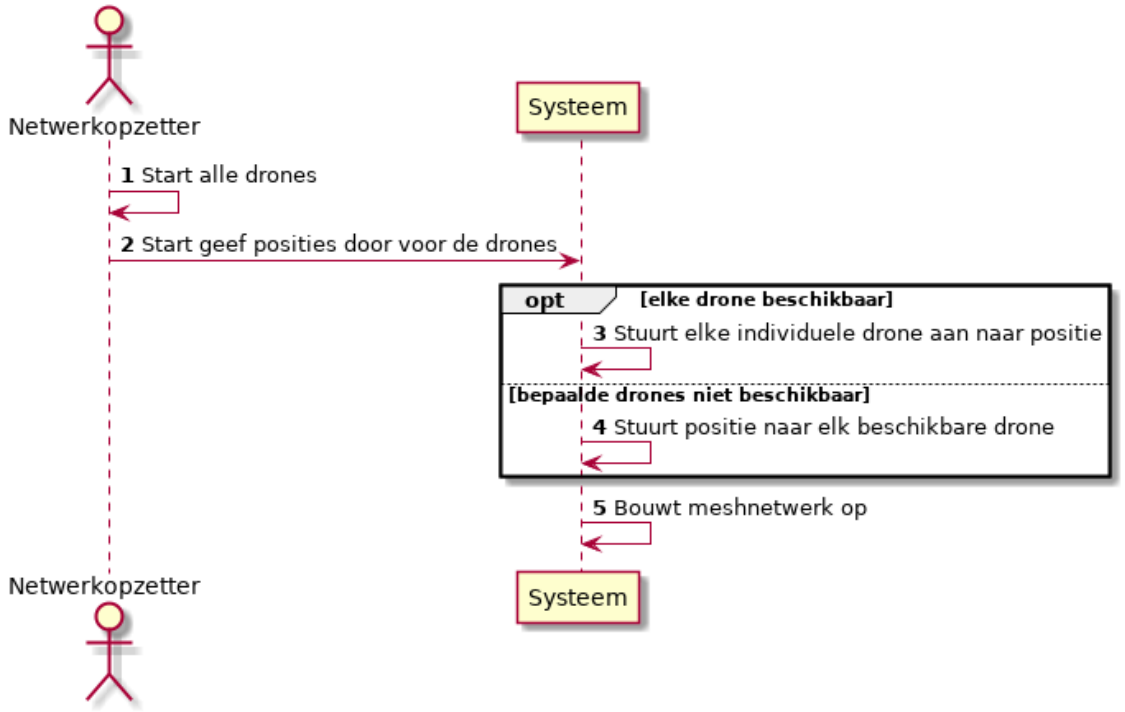
3.7.2 Basic Flow

Actor action	System responsibility
1. Start alle netwerkdrones 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. Er zijn niet genoeg drones voor de casus aanwezig of binnen bereik
	4. De drones die wel bereikbaar zijn 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.
Primary actor: Dronecontroller, Simuleren dronenetwerk, Ontplooien dronenetwerk.
Preconditions: Communicatieweg beschikbaar tot aan drone.
Postconditions: Drone gaat zich verplaatsen naar verzoeklocatie.

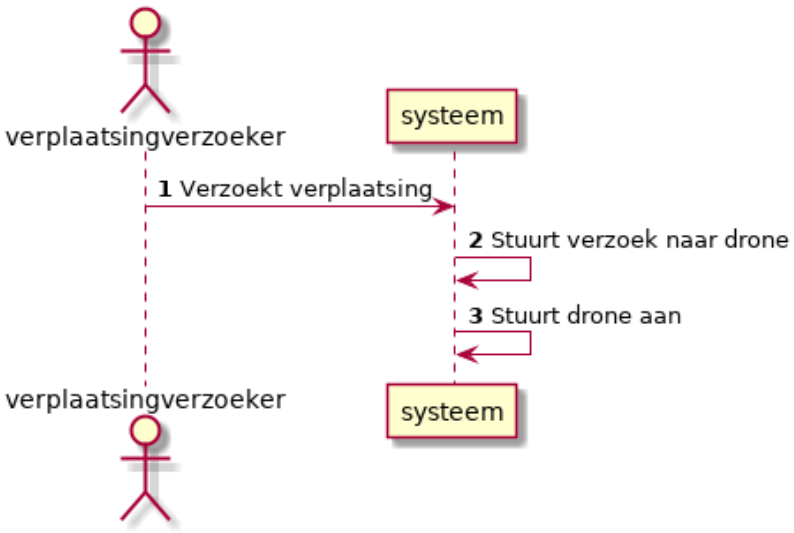
3.8.2 Basic Flow

Actor action	System responsibility
1. Verstuurt verzoek voor verplaatsing.	2. Verstuurt verzoek naar drone.
	3. Stuurt drone aan.

3.8.3 Alternative Flows

Actor action	System responsibility
	2. Drone is niet bereikbaar.

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

Use Case: 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 zicht te verplaatsen. Die 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.
Primary actor: Verzoeken droneverplaatsing, Uitvoeren noodprotocol bij geen verbinding.
Preconditions: Drone is in staat zich te verplaatsen.
Postconditions: Drone heeft zich verplaatst naar de verzochte positie.

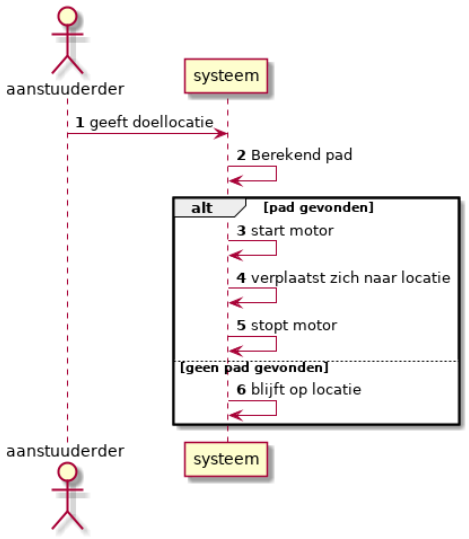
3.9.2 Basic Flow

Actor action	System responsibility
1. Stuurt doellocatie	2. Berekend pad naar locatie
	3. Start motor
	4. Verplaatst zich naar locatie
	5. Stopt motor

3.9.3 Alternative Flows

Actor action	System responsibility
	3. Geen pas mogelijk naar locatie
	4. 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.

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 het draait om een simulatie van een meshnetwerk 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 bais 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. Zij 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 mag 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 test applicatie

Onderstaand worden alle functionele requirements van de algoritme testapplicatie toegelicht.

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

ALGORITME2: Algoritmetester kan een casus met een verdeling starten Een algoritme tester moet casus situaties kunnen testen deze casus mag 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 router moet "snelëen pad berekenen.
PERF3	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 netwerkwerk moet binnen 30 seconden detecteren dat er een verbinding verloren is.
TIME2	Het netwerkwerk 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 kbitps 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

Literatuur

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