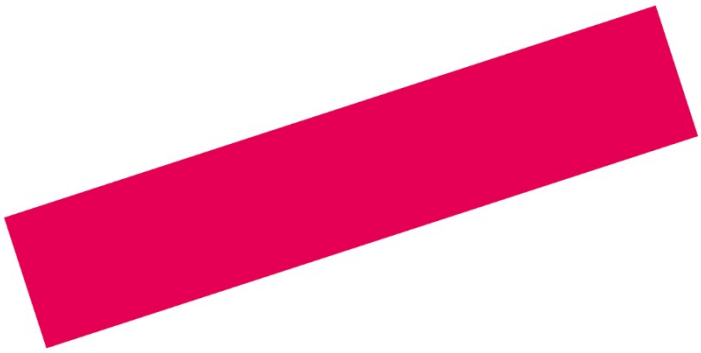


WOr Interface

8 december 2023

HAN_UNIVERSITY
OF APPLIED SCIENCES

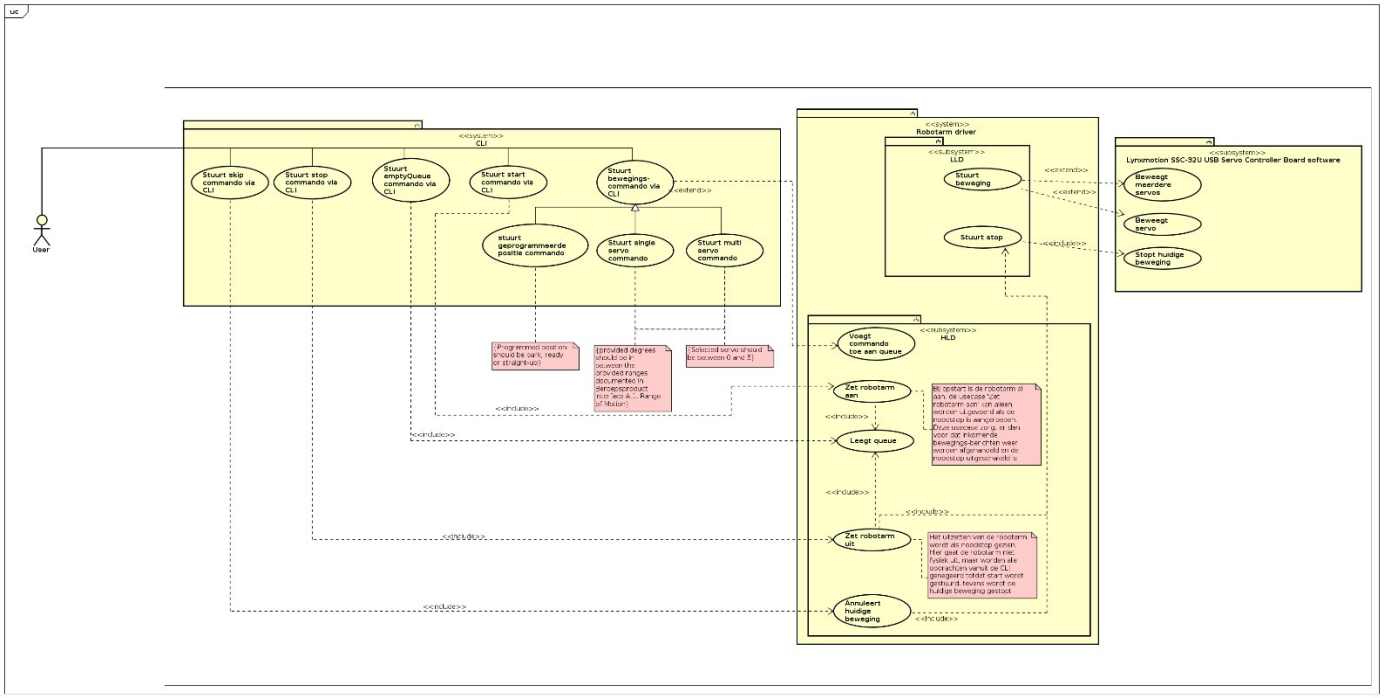
INHOUDSOPGAVE

INLEIDING

In dit document wordt een gedetailleerd overzicht gegeven van een beroepsproduct waarbij een robotarm interface is uitgeprogrammeerd. Dit document is opgesteld om te voldoen aan specifieke eisen, waaronder het beschrijven van het systeem, het definiëren van use cases en subsystemen, het presenteren van een componentendiagram, en het uitleggen van systeemgedrag. Daarnaast wordt inzicht geboden in het gebruik van het systeem, met aandacht voor de sequentie van de opstartinitialisatie en het protocol state machine.

In de verschillende onderdelen hieronder beschreven wordt ook aandacht besteedt aan het opnoemen en uitleggen van de Quality of Service in verschillende delen van het systeem, dat bijvoorbeeld betrekking heeft op veiligheid en bruikbaarheid.

1 USECASES EN SUBSYSTEMEN



Figuur 1 Usecase diagram van het gehele systeem

In het bovenstaande diagram zijn de usecases te zien die de gebruiker op het gehele systeem kan uitvoeren, verdeeld in de 4 subsystemen:

- CLI, de commandline interface waarmee de gebruiker commando's kan sturen
- Robotarm driver, het systeem die de inkomende commando's vertaald naar commando's die over de USB-poort kunnen worden gestuurd naar de robotarm
- LLD, Low Level Driver die de functionaliteit bevat om de verschillende soorten commando's te sturen naar de robotarm.
- HLD, High Level Driver die de binnenkomende commando's valideert en vertaald met behulp van de LLD.

1.1 Fully Dressed usecases

In dit onderdeel worden de belangrijkste usecases in fully dressed format beschreven. Dit zijn dan ook de usecases die direct door de gebruiker aangeroepen kunnen worden. De andere usecases spreken namelijk voor zichzelf of worden duidelijker met behulp van de onderstaande beschreven usecases

1.1.1 Use Case: Stuurt skip commando via CLI

Primary actor: User	
Stakeholders and Interests:	
Brief description: De gebruiker stuurt een skip commando via het opgestartte commandline interface programma	
Preconditions: • Het Algehele systeem is opgestart	
Postconditions (Success Guarantee): Het huidige commando wordt overgeslagen en de robotarm begint met de volgende commando in de queue, als deze aanwezig is.	
Main Success Scenario (Basic Flow):	
Actor Action	**System Responsibility**
1. De gebruiker stuurt “skip” via de CLI subsysteem	2. De HLD ontvangt dit en annuleert de huidige beweging
	3. De volgende beweging in de queue wordt gestart
Extensions (Alternative Flow):	
	3A. De queue is leeg dus stopt de robotarm met bewegen en wordt er gewacht op een nieuw commando vanuit de gebruiker

1.1.2 Use Case: Stuurt stop commando via CLI

Primary actor: User
Stakeholders and Interests:
Brief description: De gebruiker stuurt een stop commando via het opgestartte commandline interface programma
Preconditions: • Het Algehele systeem is opgestart
Postconditions (Success Guarantee):

De huidige beweging wordt geannuleerd, de queue wordt geleegd en binnenkomende bewegings-berichten worden genegeerd totdat er een start-bericht wordt ontvangen.	
Main Success Scenario (Basic Flow):	
Actor Action	**System Responsibility**
1. De gebruiker stuurt “stop” via de CLI subsysteem	2. De huidige beweging wordt geannuleerd
	3. De queue wordt geleegd
	4. Binnenkomende bewegings-berichten worden genegeerd
Extensions (Alternative Flow):	
	2A. Het systeem is al gestopt, het bericht wordt genegeerd.

1.1.3 Use Case: Stuurt emptyQueue commando via CLI

Primary actor: User	
Stakeholders and Interests:	
Brief description: De gebruiker stuurt een emptyQueue commando via het opgestartte commandline interface programma	
Preconditions: • Het Algehele systeem is opgestart	
Postconditions (Success Guarantee): De huidige beweging wordt gestopt en de queue wordt geleegd. De robotarm zal niet bewegen totdat er een nieuw bewegings-commando wordt gestuurd	
Main Success Scenario (Basic Flow):	
Actor Action	**System Responsibility**
1. De gebruiker stuurt “emptyQueue” via de CLI subsysteem	2. De HLD ontvangt dit en annuleert de huidige beweging
	3. De queue wordt geleegd
Extensions (Alternative Flow):	
	3A. De queue is al leeg dus stopt de robotarm met bewegen en wordt er gewacht op een nieuw commando vanuit de gebruiker

1.1.4 Use Case: Stuurt start commando via CLI

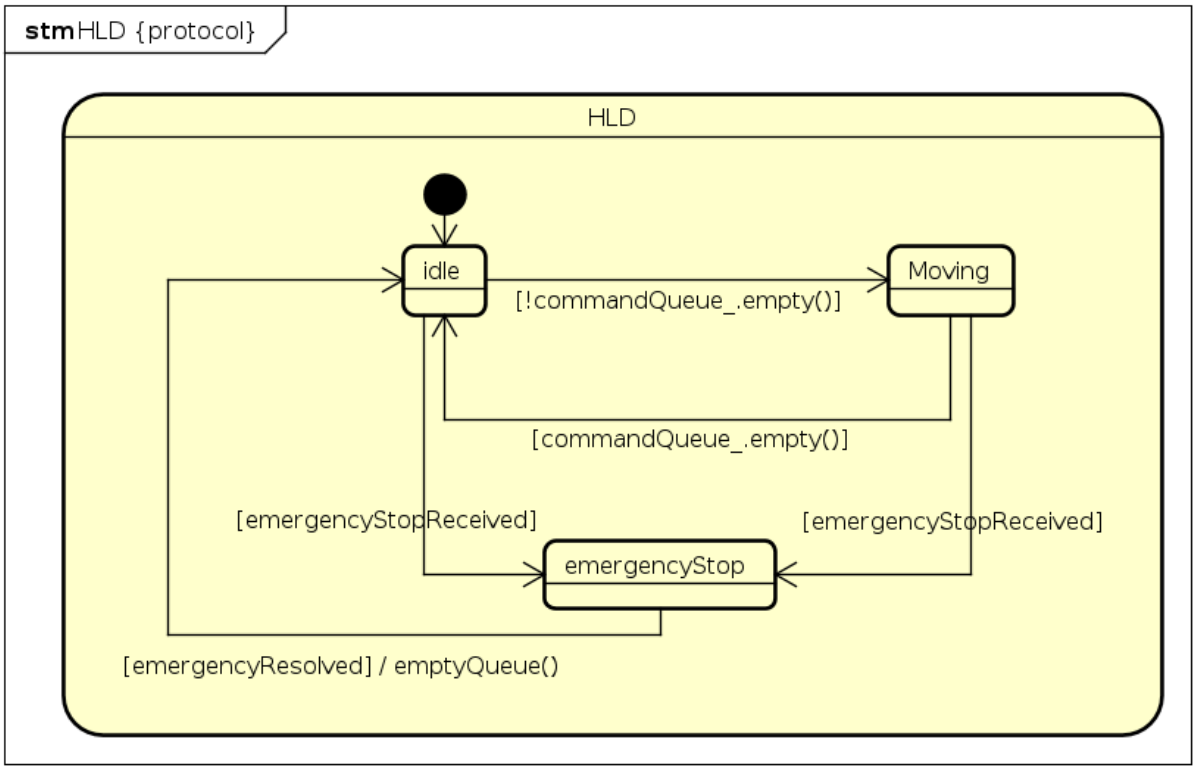
Primary actor: User	
Stakeholders and Interests: - Gebruiker: wil het systeem starten en weer bewegingscommando's kunnen verzenden.	
Brief description: De gebruiker stuurt een "start" commando via het opgestarte command-line interface programma om het systeem weer operationeel te maken.	
Preconditions: - Het algehele systeem is al eerder gestopt met behulp van de Usecase “Stuurt stop commando via CLI”	
Postconditions (Success Guarantee): - Het systeem is weer actief en gereed om bewegingscommando's te ontvangen.	
Main Success Scenario (Basic Flow):	
Actor Action	**System Responsibility**
1. De gebruiker stuurt "start" via het CLI-subsysteem.	2. Het systeem ontvangt en verwerkt het "start" commando.
	3. Het systeem wordt operationeel en is klaar om bewegingscommando's te ontvangen.
Extensions (Alternative Flow):	
	3A. Het systeem is al operationeel en actief. - Het systeem blijft actief en gereed om bewegingscommando's te ontvangen.

1.1.5 Use Case: Stuurt bewegings-commando via CLI

Primary actor: User	
Stakeholders and Interests:	Gebruiker: wil de robotarm besturen door verschillende soorten bewegingscommando's via de CLI te sturen.
Brief description:	De gebruiker stuurt een van de verschillende soorten bewegingscommando's via het command-line interface programma om de robotarm te besturen. De use case kan worden opgesplitst in drie sub-use cases: "Stuurt geprogrammeerde positie commando", "Stuurt single servo commando", en "Stuurt multi servo commando".
Preconditions:	- Het algehele systeem is operationeel.
Postconditions (Success Guarantee):	Een bewegings-commando is succesvol toegevoegd aan de queue
Main Success Scenario (Basic Flow):	
Stuurt geprogrammeerde positie commando:	
Actor action	**System responsibility**
1. De gebruiker stuurt een "programmedPosition " commando via het CLI-subsysteem met een van de drie geprogrammeerde posities (park, ready of straight-up).	2. Het systeem ontvangt en het commando
	3. Het systeem valideert of de meegegeven positie bekend is
	4. Het commando wordt toegevoegd aan de queue
Stuurt single servo commando:	
Actor action	**System responsibility**
1. De gebruiker stuurt een "singleServo" commando via het CLI-subsysteem met specificaties voor het verplaatsen van één servo naar een bepaalde positie met een snelheid of duratie	2. Het systeem ontvangt het "singleServo" commando.
	3. Het systeem valideert of de meegegeven positie te bereiken is
	4. Het commando wordt toegevoegd aan de queue
Stuurt multi servo commando:	

Actor action	**System responsibility**
1. De gebruiker stuurt een "multiServo" commando via het CLI-subsysteem met specificaties voor het verplaatsen van meerdere servos naar een bepaalde positie met een snelheid of duratie	2. Het systeem ontvangt het "multiServo" commando.
	3. Het systeem valideert of de meegegeven posities te bereiken zijn
	4. Het commando wordt toegevoegd aan de queue
Extensions (Alternative Flow):	
Probleem	**Systeem actie**
- Een meegegeven voorgeprogrammeerde positie is niet bekend	- Het commando wordt niet toegevoegd aan de queue (Zie Quality of Service)
- Een meegegeven positie is niet bereikbaar	- Het commando wordt niet toegevoegd aan de queue (Zie Quality of Service)
- Een meegegeven servo is niet bekend	- Het commando wordt niet toegevoegd aan de queue (Zie Quality of Service)

2 PROTOCOL STATEMACHINE



Figuur 2 Protocol statemachine van het HLD-systeem

In bovenstaande diagram is het protocol statemachine te zien van het HLD-subsysteem, oftewel het systeem dat gebruikersinvoer afhandelt en gebaseerd hierop de juiste commando's naar de robotarm stuurt.

2.1 Uitgebreide toelichting

Bij het opstarten van de robotarm, begint de robotarm in de idle toestand, wat aangeeft dat hij klaar is om commando's te ontvangen. Als er een bericht binnenkomt via het CLI subsysteem wordt de wachtrij van de robotarm gevuld met de beweging. Dit zorgt er dan ook voor dat de state van idle naar Moving gaat, aangezien er bij een gevulde queue de state wordt gewijzigd van idle naar Moving.

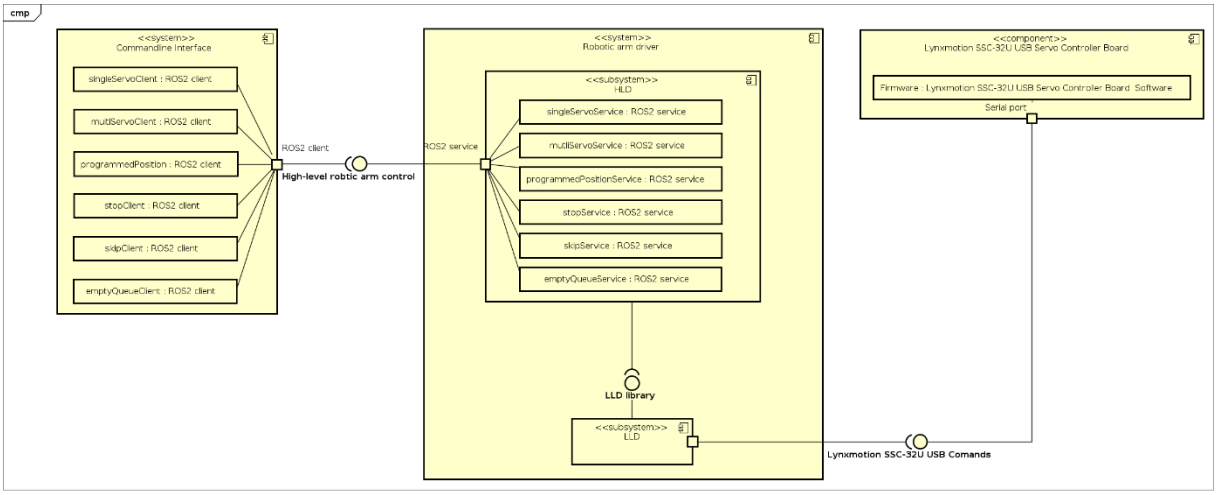
Zodra de arm zich in de moving toestand bevindt, zal deze in deze toestand blijven zolang de wachtrij met bewegingsopdrachten niet leeg is. Als er dus geen nieuwe bewegingsberichten worden gestuurd, zal de queue langzaam leeg worden, omdat er na het afhandelen van de beweging, de beweging uit de queue wordt verwijderd. Als er blijvend bewegingsberichten worden gestuurd, zal de robot dus wel in de Moving state blijven, totdat de queue leeg is.

Zowel vanuit de idle als de moving toestand kan de arm overgaan naar de emergencyStop toestand als er een noodstopbericht wordt ontvangen. Dit geeft aan dat er een noodsituatie is en de arm onmiddellijk

moet stoppen met bewegen. Ook wordt de queue geleegd, zodat er geen problemen ontstaan wanneer de emergencyStop afgehandeld is en de robotarm weer in de Idle state gaat.

Wanneer de arm zich in de emergencyStop toestand bevindt, zal deze hier blijven totdat er een start bericht wordt ontvangen. Dit bericht geeft aan dat de noodsituatie is opgelost en de arm weer operationeel is. Na ontvangst van het start bericht zal de arm terugkeren naar de idle toestand, waarin hij opnieuw commando's kan ontvangen om te bewegen. Tijdens de emergencyStop state worden ook alle berichten genegeerd, behalve het startbericht, zodat het vullen van de queue niet mogelijk is.

3 COMPONENT DIAGRAM



Figuur 3 Component diagram van de verschillende subsystemen

3.1 Uitgebreide toelichting

Zeker, het componentendiagram dat je beschrijft, vertegenwoordigt een robuust systeem voor het aansturen van een robotarm via verschillende subsystemen. Hier is een gedetailleerde toelichting op het diagram:

3.1.1 Commandline Interface Subsysteem (CLI):

Dit subsysteem fungeert als een gebruikersinterface voor het aansturen van de robotarm. Het is een ROS2 C++ programma dat gebruikersinvoer verzamelt en doorstuurt naar het HLD-subsysteem.
Het verzenden van commando's gebeurt via verschillende ROS2 clients: **singleServoClient, multiServoClient, programmedPosition, stopClient, skipClient, en emptyQueueClient**.

3.1.2 High-Level Driver (HLD):

- Het HLD-subsysteem, onderdeel van het Robotic Arm Driver-systeem, ontvangt de invoer vanuit de CLI.
- Het HLD-subsysteem houdt een statemachine bij, zoals beschreven in het eerder vermelde protocol statemachine. Dit systeem beheert de verschillende toestanden van de robotarm op basis van de invoer.
- Het HLD-subsysteem maakt gebruik van het LLD-subsysteem om daadwerkelijke bewegingen of stop-commando’s op de fysieke robotarm te laten gebeuren.

3.1.3 Low-Level Driver (LLD):

- Het LLD-subsysteem, ook onderdeel van het Robotic Arm Driver-systeem, ontvangt opdrachten van het HLD-subsysteem.
- Dit subsysteem is verantwoordelijk voor het vertalen van inkomende opdrachten naar het formaat van de Lynxmotion SSC-32U en geeft logica voor het daadwerkelijk uitvoeren van de bewegingen of het stoppen van de robotarm.
- Het communiceert met de hardware via een seriële poort, waarbij commando's worden verzonden in het formaat van de Lynxmotion SSC-32U.

3.1.4 Lynxmotion SSC-32U USB Servo Controller Board Software Component:

- Dit component is de daadwerkelijke robotarm.
- Dit component ontvangt de berichten die over de seriële poort worden verzonden vanuit het LLD-subsysteem.
- Het vertaalt deze berichten naar acties voor de robotarm, waardoor de arm daadwerkelijk beweegt of stopt met bewegen, afhankelijk van de ontvangen commando's.

Deze architectuur maakt gebruik van verschillende subsystemen om een gestructureerde en modulaire aanpak te bieden voor het aansturen van de robotarm. De CLI fungeert als een interface voor gebruikersinvoer, terwijl het HLD-subsysteem de logica van de robotarm beheert. Het LLD-subsysteem staat in voor de fysieke uitvoering van de commando's via een seriële verbinding, terwijl de Lynxmotion SSC-32U Software Component de communicatie met de hardware afhandelt voor feitelijke bewegingen van de robotarm.

OPEN UP
NEW HAN_UNIVERSITY OF APPLIED SCIENCES
HORIZONS.