System Specification
Robo Ducks

Project Name	Roboducks
Project Leader	Florentin Gewessler
Document state	In process
Version	V. 0.1

Document History

Revision Nr.	Date	Version	Chapter	Modification	Author
1		0.1	All	Initial Draft	
2	20.12.18	1.0	All	First Release	All

1 Initial Situation and Goal

1.1 Initial Situation

Our goal is to participate in the German Open Standard Platform League. The German Open Standard Platform League is a soccer league where all teams participate using the same robot, the NAO robot from SoftBank Robotics. These robots play fully autonomously and each one takes decisions separately from the others, but they still have to play as a team by using communications. The teams play on a green field with white lines and goal posts, with no other landmarks, and the ball consists in a realistic white and black soccer one. These game characteristics generate a very challenging scenario, which allows improving the league every year. Our competition are mostly teams from universities like the team B-Human which comes from the university Bremen. In this field it is very important to have sponsors because the most schools and universities can not effort too many robots. We have talked to some of the other teams and they said they would need about 50.000 Euros a year. As we mentioned earlier, our main goal is to participate in the German Open Standard Platform League but we can break this down to many sub goals. The first sub goal would be that we finish the framework, which we call “Duckburg”, till the 24.12.2018. The framework is the base of our software which will contain the basic functions of a system.

1.1.1 Application Domain

Our software will be used on the NAO-Robots. With our software it will be possible that the robots play soccer in teams of five. Our software will also be optimized for the limited resources of the NAO Robots. The robots only have a ATOM Z530 single core CPU with 1.6 Gigahertz, 1 GB of RAM and up to 10 GB of storage. Furthermore, our software has to be very modular and expandable. This is granted through a specific system architecture which we will elaborate later in the system architecture and interfaces section. All in all, we can summarize our project in three major steps. The first one would be, that the Duckburg, our basic framework, is fully implemented because that is a major requirement to start further development. The second step would be the development of the different Engines and Agents which will do specific tasks like walking or kicking a ball. The challenges are very varied depending on which task you like to develop. For example if we want to write an Engine and the associated agents we will have major problems with to balance of the robot but if we want to develop a Engine for recognising a ball we will have to face other problems. The last step would be the the communication between one robot to another one. Here we have to deal with the limitations of the wlan module of the robots which allows only a small number of kilobytes between the robots.

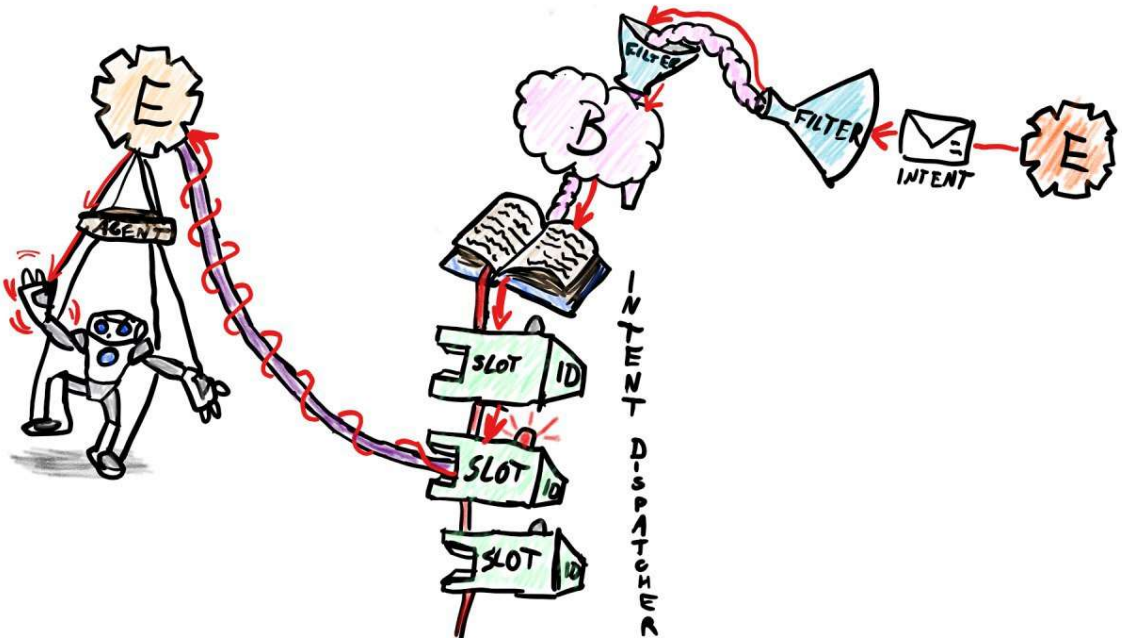
1.1.2 Glossary

Fold Concept	Description
Duckburg	Basic Framework provides the basis of our system
Engines	Manages the agents for an specific task
Agents	Providing specific function like walking
NAO-Robots	The kind of robots we are using
NAO-QI	pre installed framework

1.2 Goal Definition

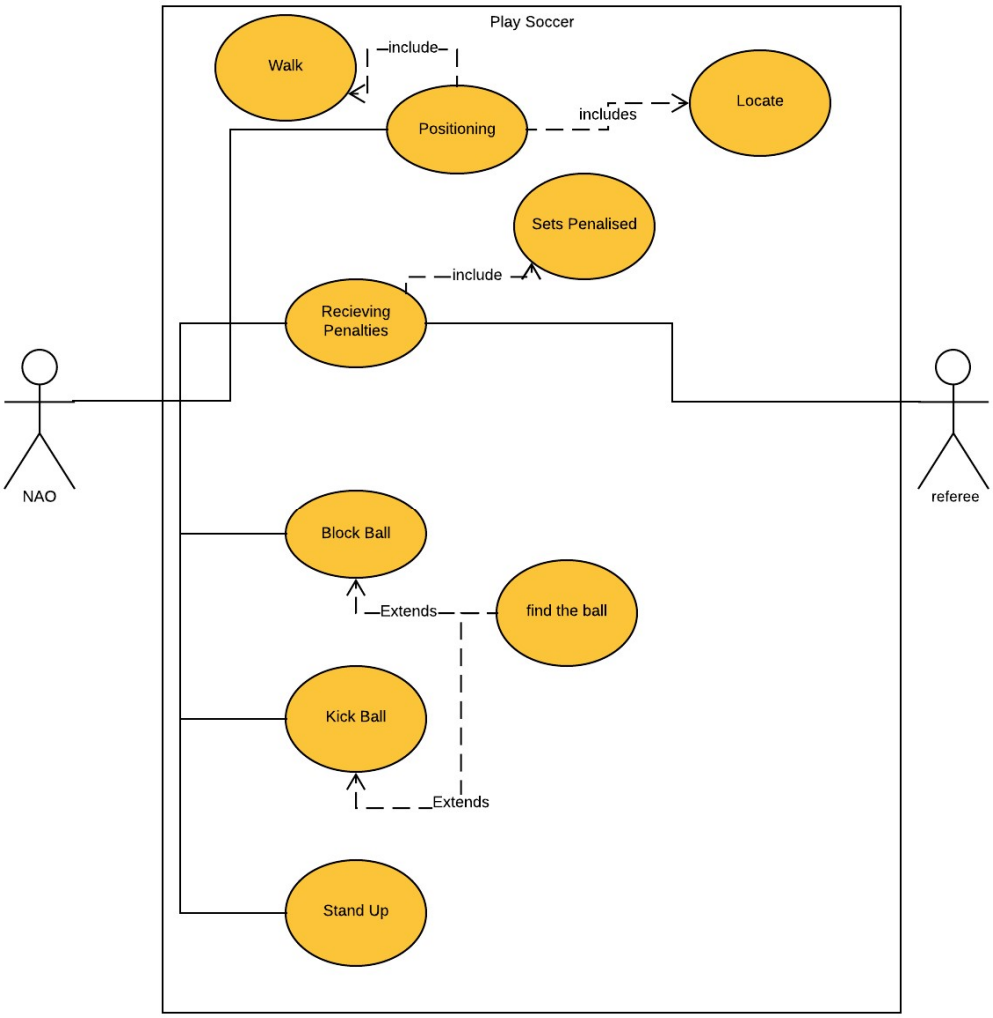
The goal of Roboducks is to participate in the German open Standard Platform League (SPL), where 5 Naos play soccer against other teams currently there is no other team from Austria and non that is from a Higher Technical School. For archiving this goal we have to program a new framework because the existing one from Aldebaran (NaoQi) is slow and in some cases not reliable. The new framework is made with C++ which provides as language a awesome performance and memory efficiency. With the new framework we can focus on our main goal to play football with the robots and program smart techniques to play out an Universitie team which would be an great phus for our schools reputation.

2 Functional Requirements



Agents: Providing specific function like walking
Engines: Manages the agents for an specific task
Intent Dispatcher: transfers the Intents to the right Slot
Brain: provides the engines the possibility to subscribe to a slot and send intents to them
Filter: filters the Intents to secure, that every Intent is valid

2.1 Use Case Diagrams



2.2 Use Case Details

Positioning

Positioning is a crucial task the robots need to perform in order to gain higher chances of winning. To avoid gaps in the formation the Naos need to be evenly distributed over the soccer field to increase the chance of finding the ball and thus being able to block or kick it more often.

Good Case Scenario

Robot Successfully positions itself.

1. Robot tries to locate itself.
    1.1. Robot gets its estimated location from its Team members
2. When the location is found the Robot moves towards its target Position

Bad Case Scenarios

Robot is not able to receive/get its Position

1. Robot can’t find it’s Position
2. Relocate

Nao walks in the Wrong direction

1. Robot exits Soccer Field
2. Nao gets Penalised according to the SPL Rules

Block the Ball

The aim of this activity is to enable the Goalkeeper robot to prevent approaching balls. Therefore it's extremely important to the Goalkeeper to Keep track of the ball.

Good Case Scenario

Goalkeeper succeeds blocking the ball.

1. Nao in the goal Keeps track of the ball
    1.1. Nao receives estimated location of the ball from other robots.
    1.2. Goalkeeper detects the ball using its cameras
2. If the ball approaches the Goal an Blocking animation is being calculated
3. Robot tries to Block the ball

Bad Case Scenarios

Goalkeeper doesn’t see the ball.

Goalkeeper fails to prevent the ball from entering the Goal.

→ Repeat Searching for ball

Kick the Ball

Good Case Scenarios

Ball successfully lands in the opponents Goal.

1. Receive position
    1.1. Nao receives estimated location of the ball from other naos.
    1.2. Nao detects the ball using its cameras
2. If the Ball is in front of the Nao an kicking animation is being calculated.
3. Nao performs calculated animation and shoots

Bad Case Scenarios

Nao does not find the ball.

1. Nao tries to get the balls position

Robots falls down.

1. Nao plays Stand up Animation

Receiving Penalties

Good Case Scenario

Nao reacts on Penalisation

1. Nao gets notified that he is penalized
2. Nao realizes he is penalized and stiffs itself and stops all prozesses
3. The robot waits for the penalisation time to pass

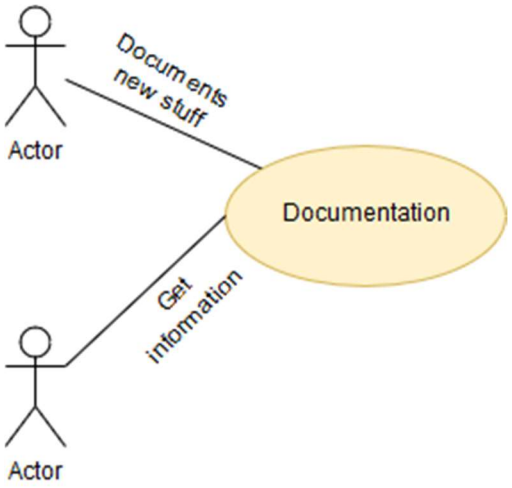
Bad Case Scenario

Nao does still move.

1. The referee has to press the robots chest button to manually penalize it
   (This guarantees that the Nao gets penalized.)

3 Non-functional Requirements

1) Documentation



The project is ongoing. Every year new people join the project. To inform these people we need to have a documentation, where is exactly shown how our framework works. This can be done by a PDF which contains all important information about our product. This PDF is on a Github Repository, so that everyone has access.

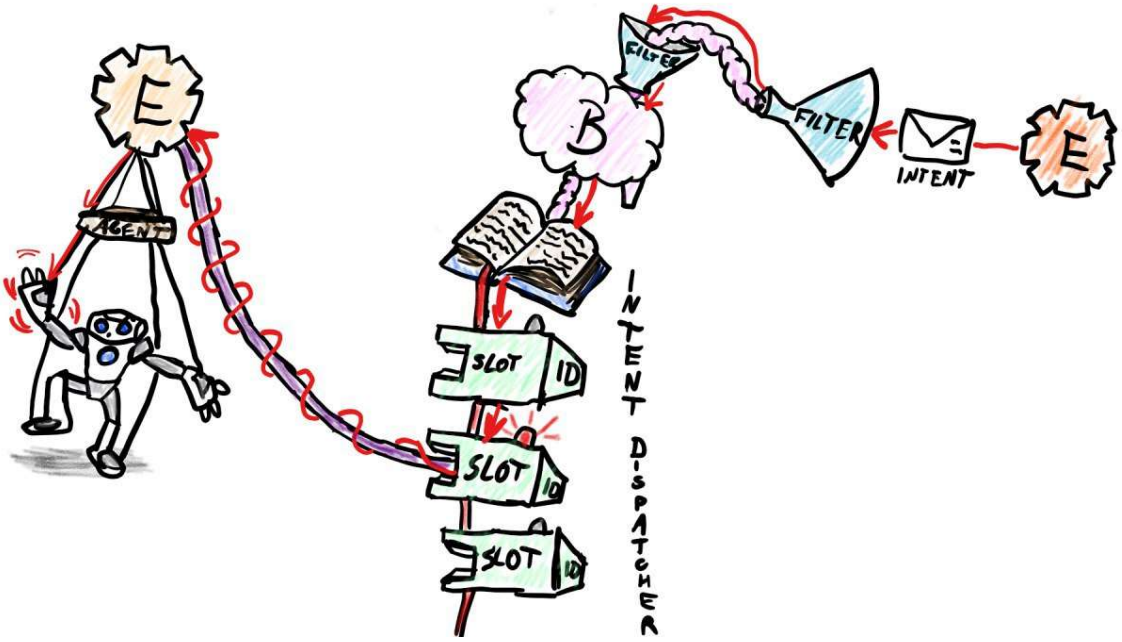
2) Useability with all versions

It is also important that the newest version of the naos(e.g.: V6) is compatible with our framework. Because if our framework would not be compatible with the older version , then we would have to few robots to participate at the German Standard Platform League.

4 Quantity Structure

4.1 System Architecture and Interfaces

System Overview:



Agents:	Providing specific function like walking
Engines:	Manages the agents for an specific task
Intent Dispatcher:	transfers the Intents to the right Slot
Brain:	provides the engines the possibility to subscribe to a slot and send intents to them
Filter:	filters the Intents to secure, that every Intent is valid

Summarizing:

Overview of ‘Duckburg Concept of a Framework’
https://github.com/Bauepete/robo-ducks-make-it-go/blob/master/docs/DuckDoc.pdf

4.2 Acceptance Criteria

AC_001 Receiving a Penalty

Test Step	Expected Behaviour	Reality
Send the Robot the penalty message	Fall into sleep mode and wait for the end of the penalty. Then the Robot should reposition itself	

AC_002 Positioning

Test Step	Expected Behaviour	Reality
Give the Robot a position where he should walt to	The Robot walks to the expected Position	

AC_003 Kick the Ball

Test Step	Expected Behaviour	Reality
Put a ball in front of a Nao	The Nao kicks the ball towards the opposite goal	

AC_004 Block the Ball

Test Step	Expected Behaviour	Reality
Let a ball roll towards our goal	The Nao blocks the ball in order to prevent the ball from rolling into our goal	

4.3 List of Abbreviations

SPL	Standard Platform League

4.4 References

Framework Documentation	https://github.com/Bauepete/robo-ducks-make-it-go/blob/master/docs/DuckDoc.pdf
SPL Rules	http://spl.robocup.org/wp-content/uploads/downloads/Rules2019.pdf