



**Amsterdam University
of Applied Sciences**

ROVER RESCUE SYSTEM

Documentation

Author:

Christiaan van Arum

Raphaël Bunck

Nino van Galen

Martijn Vegter

Student Number:

500778983

500774349

500790589

500775388

January 14, 2019

Contents

1	Introduction	2
2	Use Cases	3
3	Definition of Ready	4
4	Definition of Done	5
4.1	Feature	5
4.2	Sprint	5
5	Domain model	6
6	System sequence diagram	7
7	Class Diagrams	8
7.1	Project Overview	8
7.2	Controller - Steam Controller	9
7.3	Controller - DPad	10
7.4	Controller - Dual Axis	10
7.5	Controller - Simple Button	11
7.6	Controller - Touch	12
7.7	Controller - Single Axis	12
7.8	Motor	13
7.9	Nervi - Camera Mount	14
7.10	Nervi - Rotary Encoder	14
7.11	Nervi - Servo	15
7.12	Nervi - Ultrasonic	16

1 Introduction

In the (near) future robots will be more and more part of our daily life. Even more than they are already part of society today. Industrial robots are nowadays very common, the use of drones by military and civilians triggers discussion and science is creating robots to take care of people who need help. Functioning more or less autonomous requires that such machines have to be robust, take their own decisions and operate in a safe way.

Thus we created a robot that helps to rescue people trapped in a collapsed building.

The user is able to control the robot using the combination of a controller, specifically a Steam Controller, and a mobile phone as display used by the Google Cardboard. Of course, being a robot, the robot itself wants to do as much as possible. The degree of autonomy of the robot has been focused on assisting the user as much as possible.

2 Use Cases

Table 1: Connect to Rover

Name	Connect to rover
Brief	The user connects to the rover and enters the token
Actors	User, Rover and control device.
Assumption	The rover is installed and activated
Description	To use the rover the user must first connect to the rover.
Primary scenario	1. Connect all needed devices to the internet. 2. Enter IP in the application on the control device. 3. Enter Token on controller.
Exception	
Result	The user is connected to the rover and is able to operate it.

Table 2: Drive the Rover

Name	Drive Rover
Brief	The user connects to the rover and enters the token.
Actors	User, Rover and control device.
Assumption	The rover is installed, activated and connected.
Description	To use the rover the user must first connect to the rover.
Primary scenario	1. Press the button corresponding to your movement.
Exception	
Result	The rover moves in the chosen direction.

3 Definition of Ready

1. Must have clear description.
2. Must have a milestone.
3. Must have a estimate for the burndown chart.
4. Issues from the same type must be organised in epic's.
5. Relevant issues must be done before continuing.
6. Previous sprint issues must be done before new sprint issues. (Except icebox issues)
7. Issues can't be large and vague, should be split in multiple issues.

4 Definition of Done

4.1 Feature

1. DoD of each single User story, included in the Sprint are met
2. “To do’s“ are completed
3. All unit tests passing
4. Product backlog updated
5. Project deployed on the test environment identical to production platform
6. Tests on devices/browsers listed in documentation passed
7. Tests of backward compatibility passed
8. The performance tests passed
9. All bugs fixed
10. Sprint marked as ready for the production deployment by the Product Owner

4.2 Sprint

1. Code Complete
2. Environments are prepared for release
3. All unit & functional tests are green
4. All the acceptance criterias are met
5. QA is done & all issues resolved
6. All “To Do“ annotations must have been resolved
7. OK from the team: UX designer, developer, software architect, project manager, product owner, QA, etc.
8. Check that no unintegrated work in progress has been left in any development or staging environment.
9. Check that TDD and continuous integration is verified and working

5 Domain model

TODO

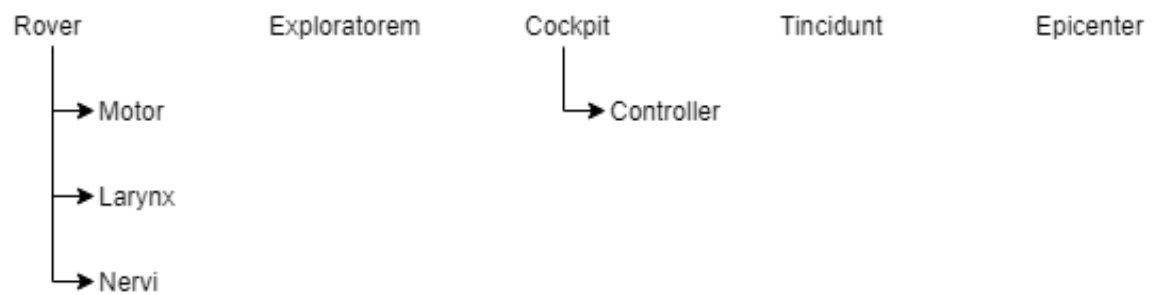
6 System sequence diagram

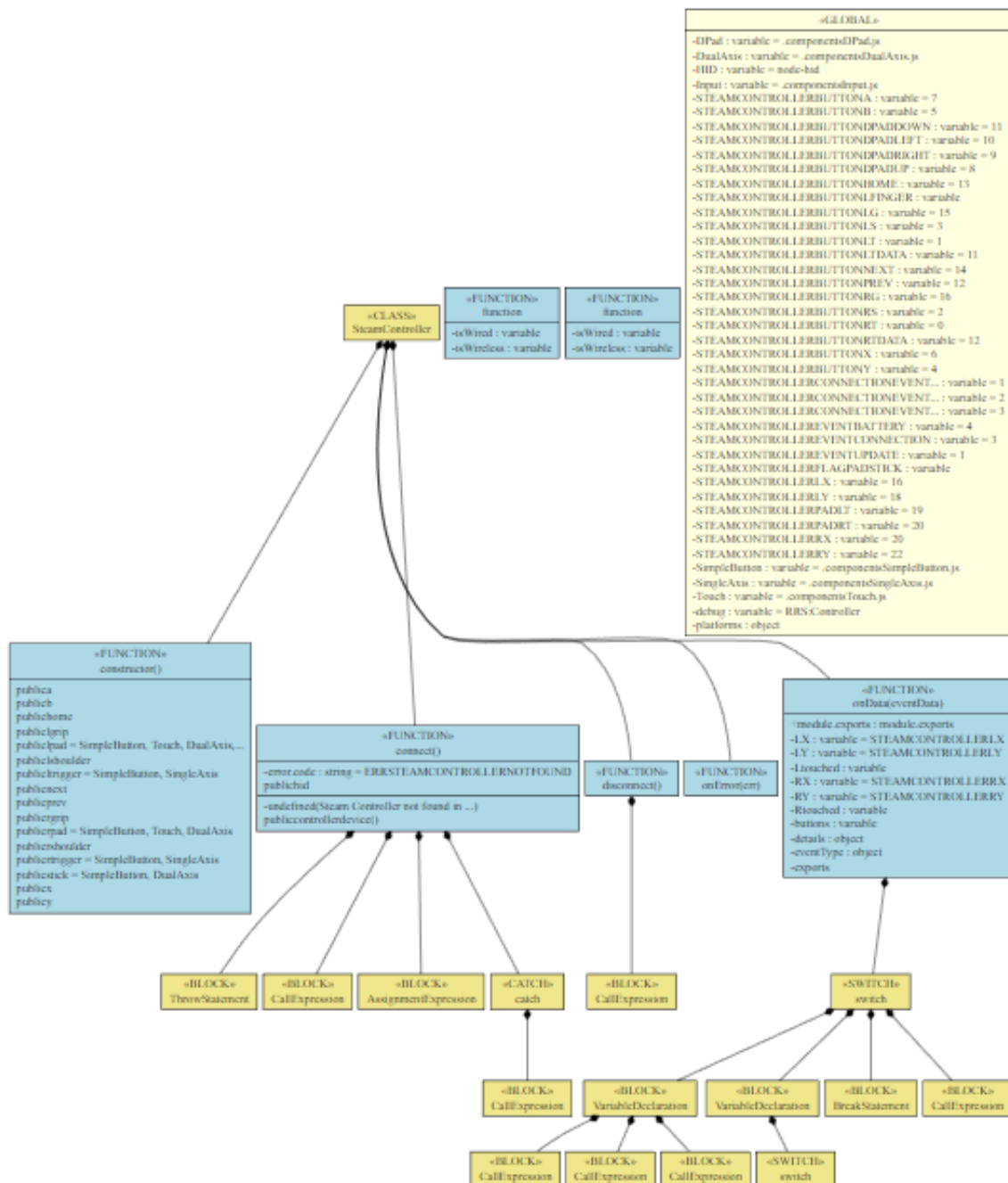
TODO

7 Class Diagrams

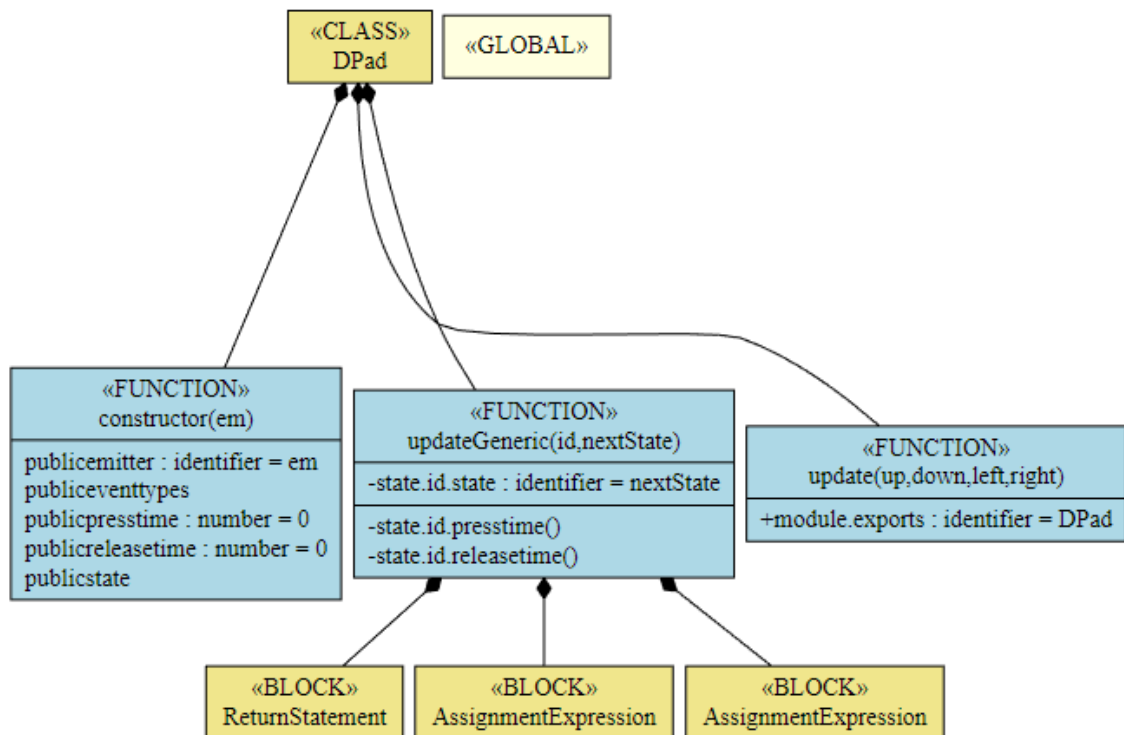
The class diagrams are a visual representation on how the RRS has been structured. The project overview given below shows which modules are submodules from the overhanging modules. In the UML diagrams below there has been given a precise and complete overview of the class diagrams per module. These diagrams have been made using a generating tool in a command line interface.

7.1 Project Overview

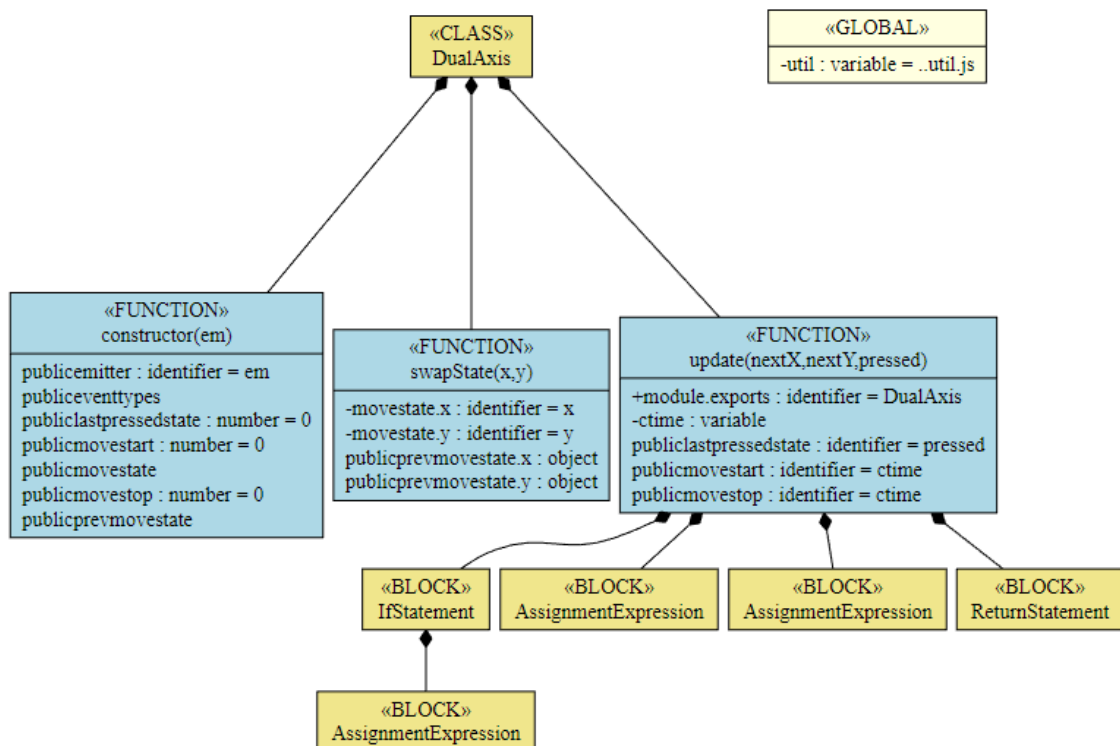




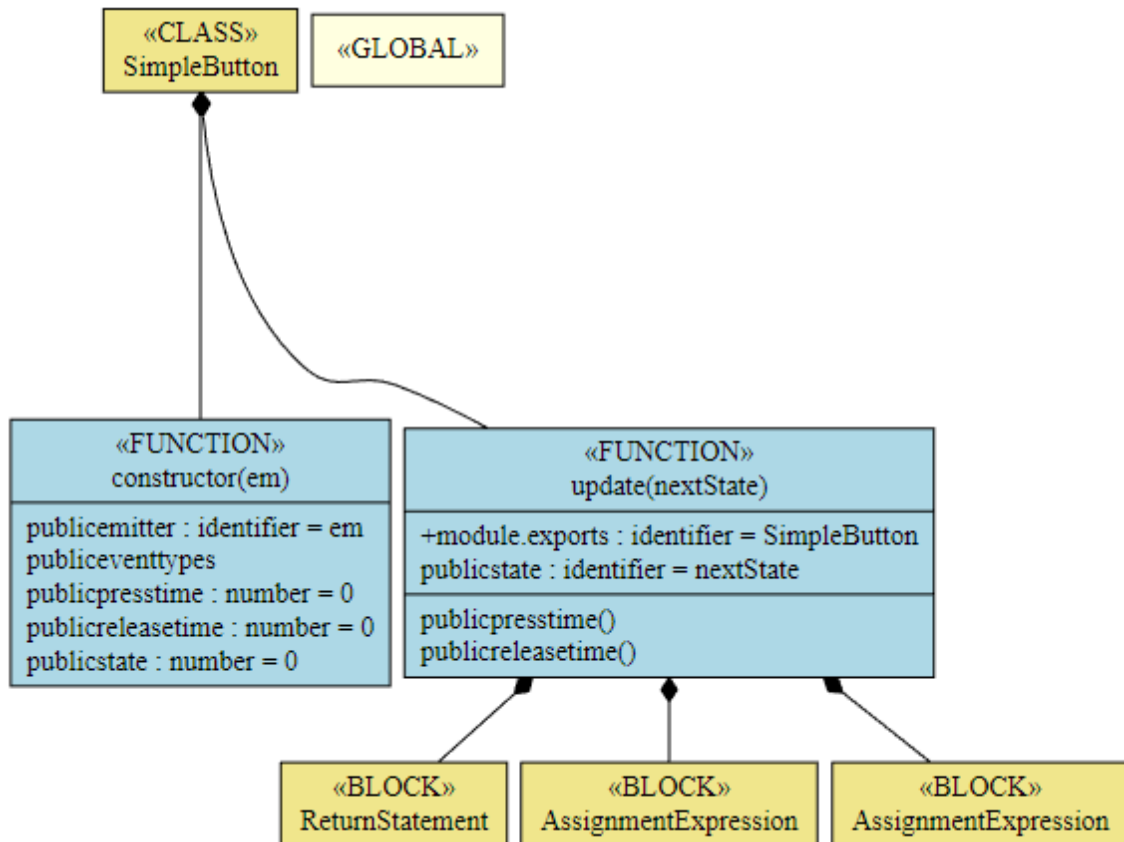
7.3 Controller - DPad



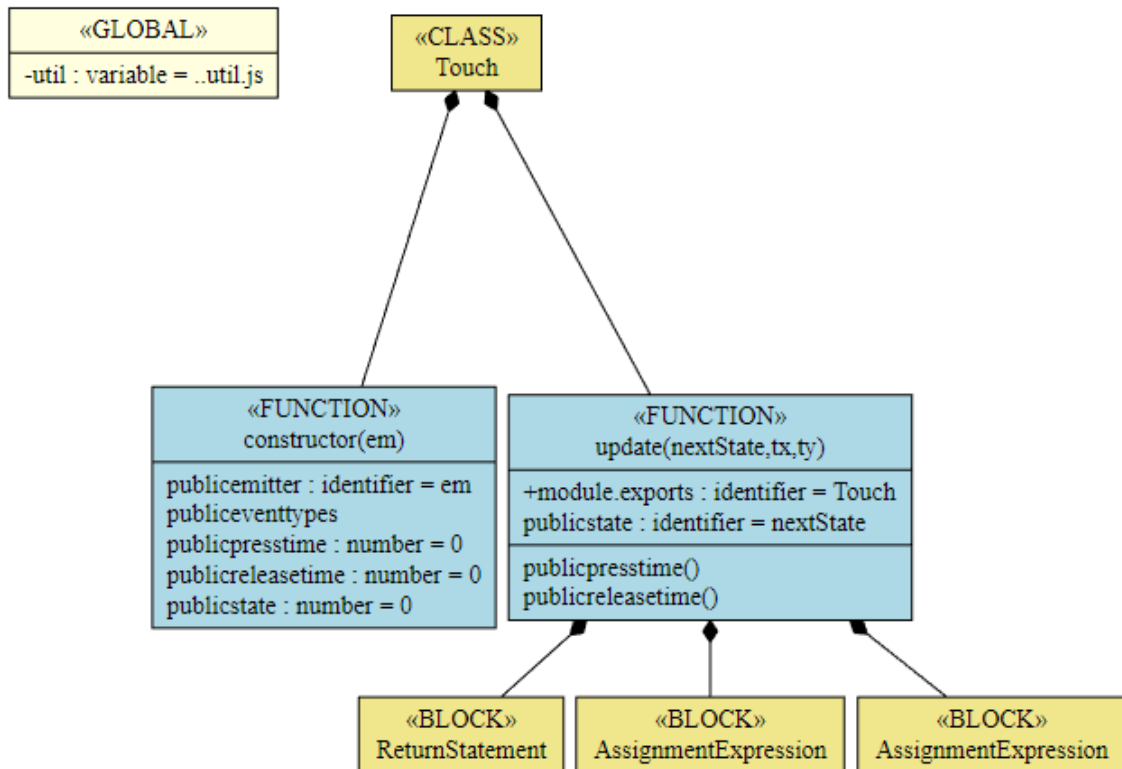
7.4 Controller - Dual Axis



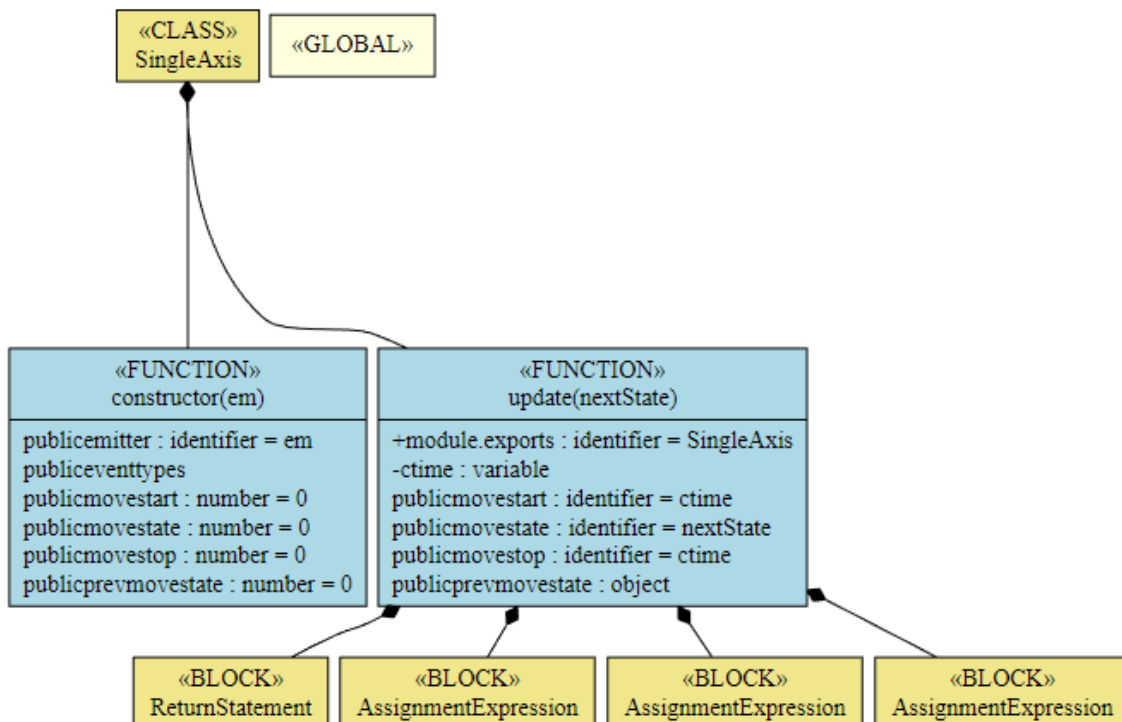
7.5 Controller - Simple Button



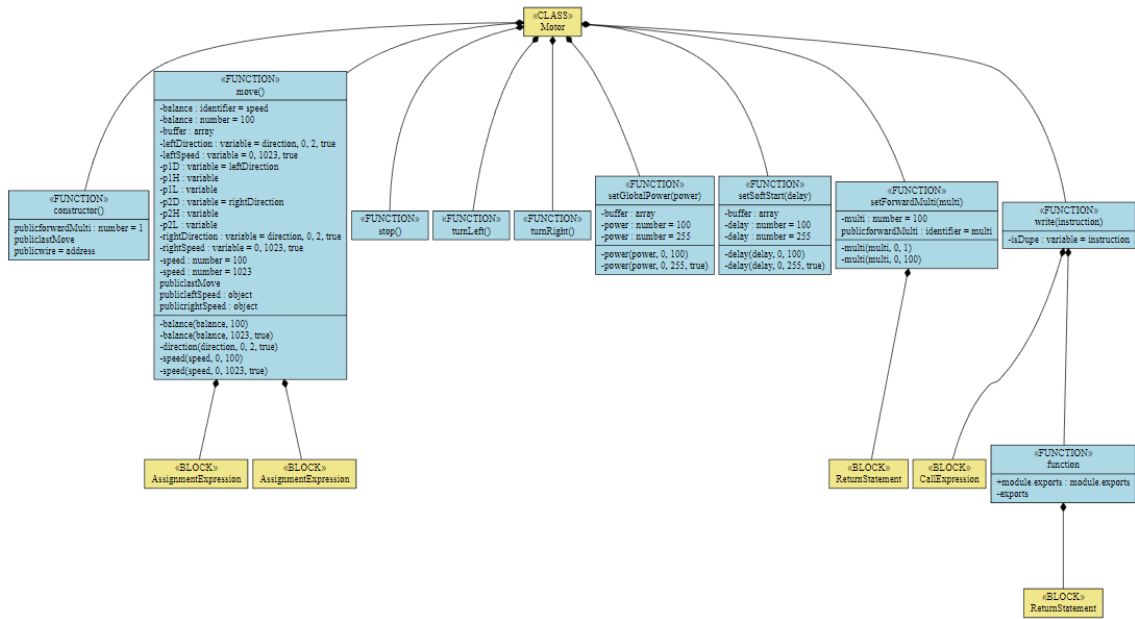
7.6 Controller - Touch



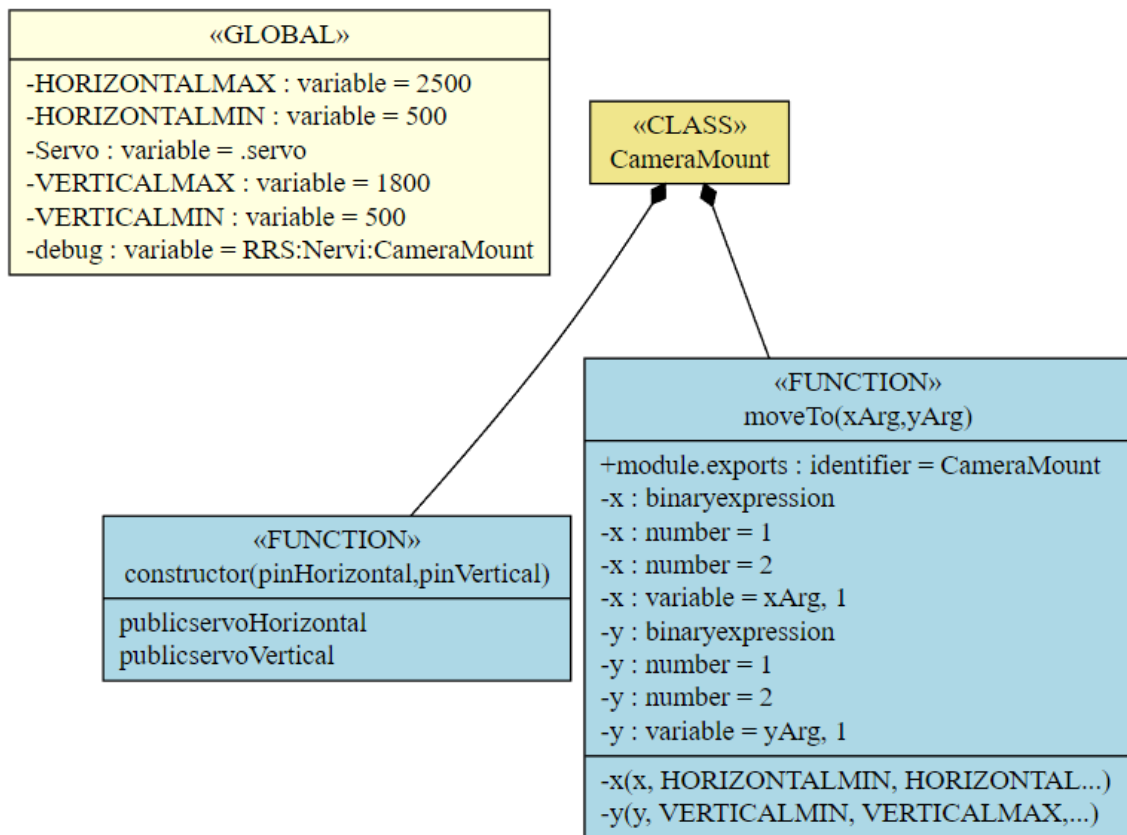
7.7 Controller - Single Axis



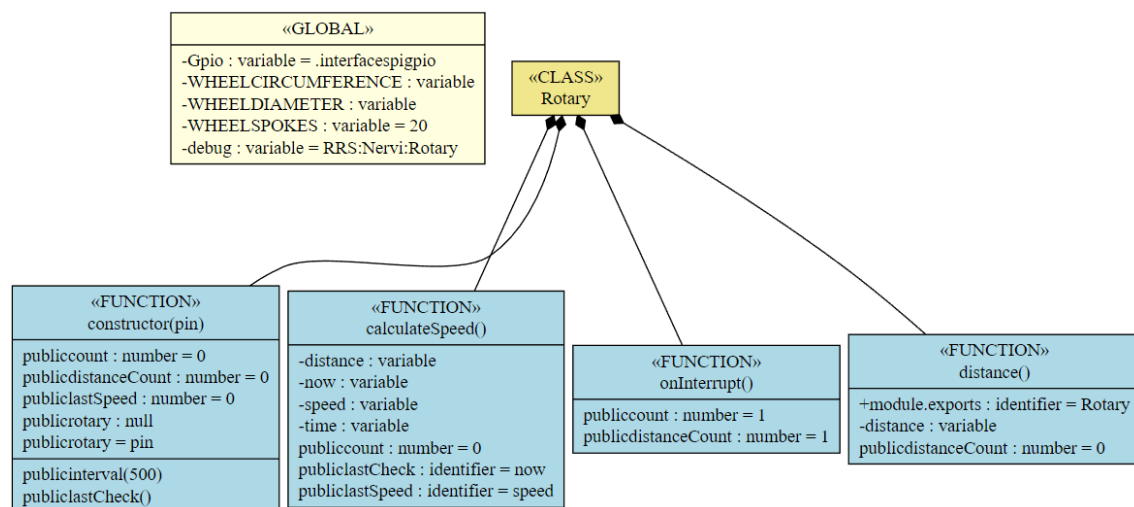
7.8 Motor



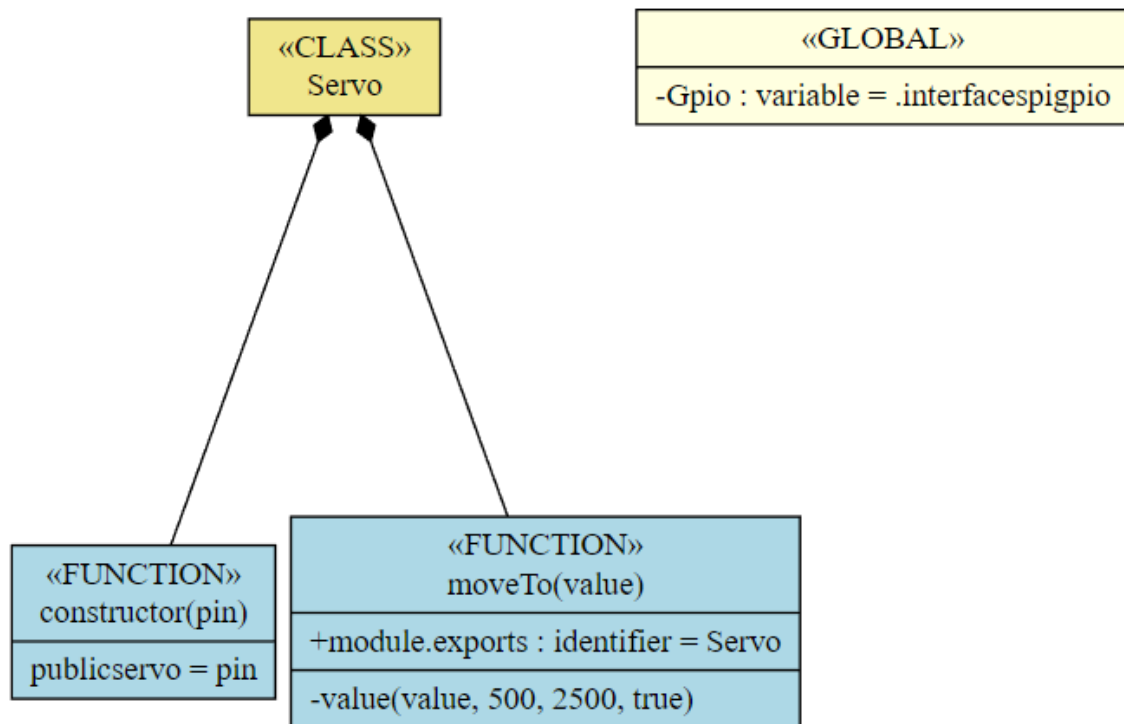
7.9 Nervi - Camera Mount



7.10 Nervi - Rotary Encoder



7.11 Nervi - Servo



7.12 Nervi - Ultrasonic

