1 Introduction

This document describes the architecture instantiation of RefSORS (Reference Architecture for Service Oriented Robotic Systems)[1], which is a reference architecture based on Service-Oriented Architecture (SOA) [2] specifically for indoor, grounded mobile robotic system. The instatiated architecture will be design following the ArchSORS process [3] and will responsible for controlling two robots transporting products on an industrial shop floor. The specifications used in this project are equivalent to ROBAFIS[4]. Comparing this system quality attributes with systems from RobAFIS competition enables evaluation of it and impact of modeling an architecture using RefSORS.

2 Step RA-2: Capability Indentification

Since functional requirements from ROBAFIS competition already fulfill the first step of the ArchSORS process, it can be the input for step RSA-2. In this step the information from the previous is used to model the application flow represented by BPNM diagrams. Then capabilities are identified by mapping these functional requirements with RefSORS requirements and analyzing the application flow. Finally these capabilities are assessed based on their functionalities to decide which are going to be exposed as services and which are going to be provided as components that support these services.

2.1 Model Application Flow

From functional requirements and concepts from the reference architecture BPNM diagrams were modeled to represent the application flow. BPNM diagrams are unique representations of a problem the application solves, since an representation of it in the reference architecture would be to general to add any useful information, it could only contribute with concepts. The modeled application flow was divided in six BPNM diagrams and seven polls shown in order of occurrence, each lane represents a possible capability and tasks are inspired by the robotic system required functionalities.

Figure 1 shows the most abstracted flow, the flow of the application. The application starts by subscribing to all robots sensor, then mapping the ambient with both robots cooperatively. The system will then wait for a transport order, when it happens the nearest robot is chosen to transport it. It continues to stand by waiting or product orders until its finished.

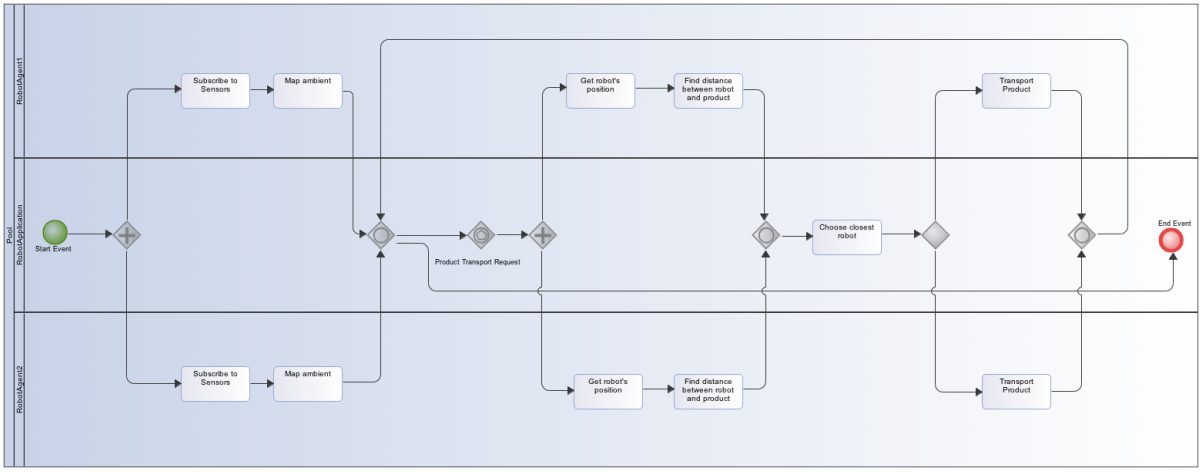

Figure 1: Application Diagram

Figure 2 shows the system's flow for subscribing to the robots sensors. When subscribing to sensor data will be sent periodically, the system will handle this new data while continuing to receive updates, this applies to all sensors used.

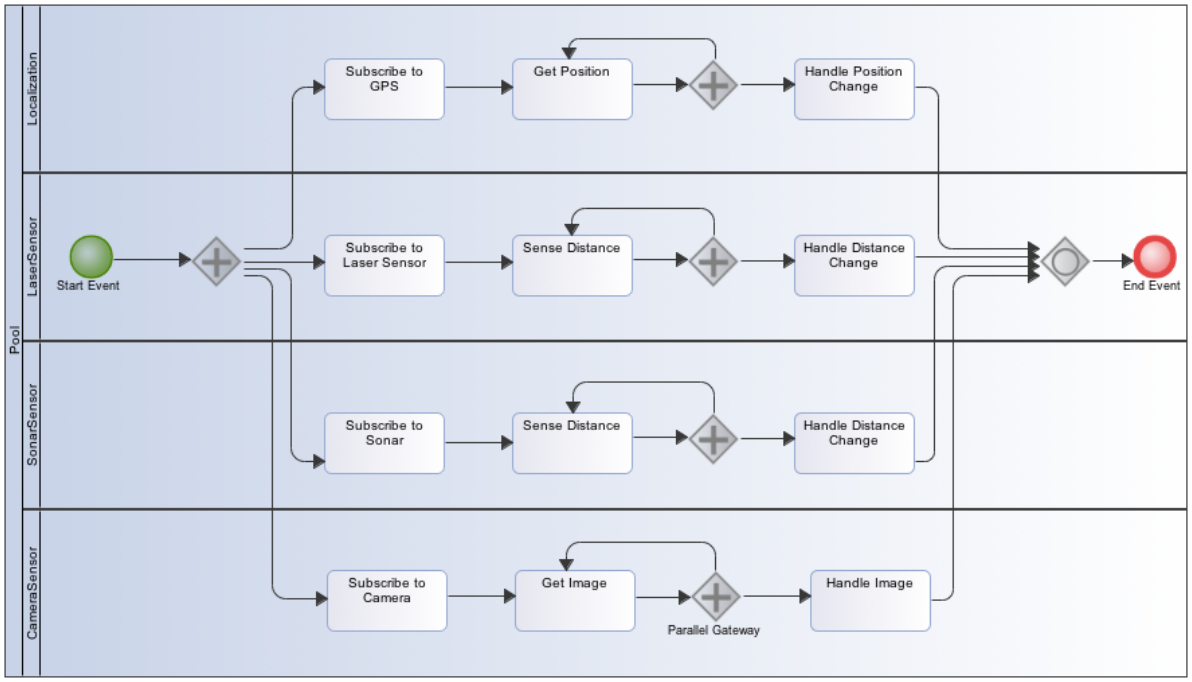

Figure 2: Subscribe to Sensors

Figure 3 shows the flow of one robotic agent when mapping the ambient at the beginning of every day. Each robotic agent follows a wall and uses the laser sensor to compose parts of the map. The composed part of the map is then send to a map shared by both robots. When they each reach the other side of the map they stop at a resting area.

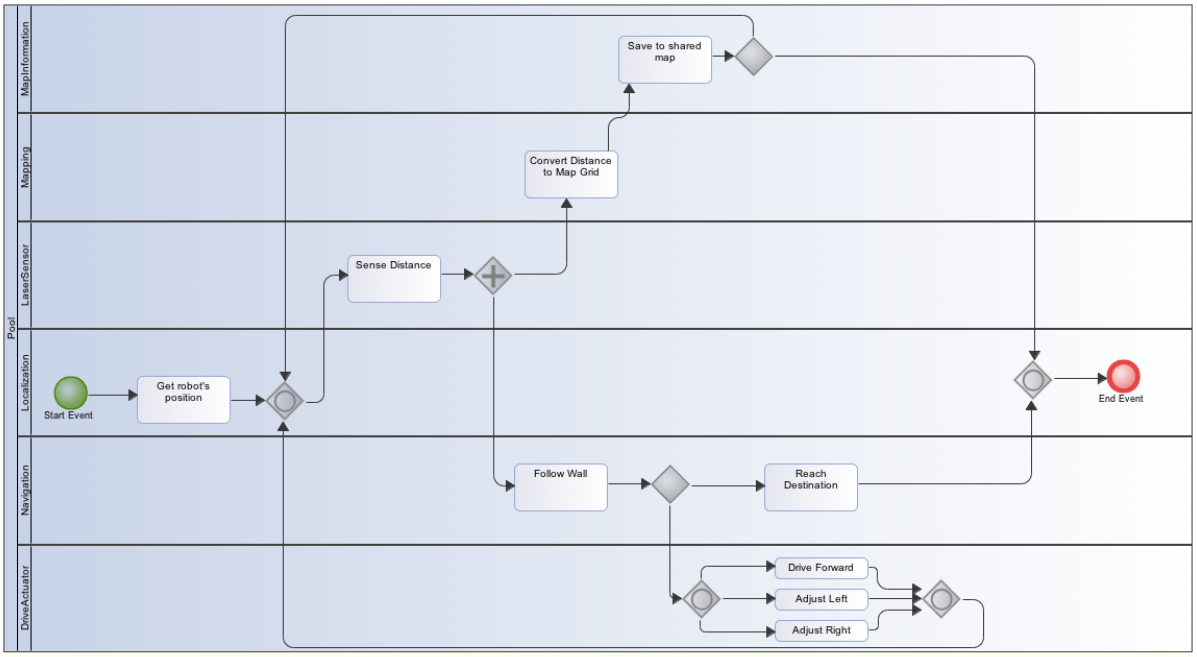

Figure 3: Mapping

Figure 4 shows the flow for driving a robot from its location to its destination by the shortest path. Starts by locating the robot and positioning it on the map. With the map position of the robot and its destination it finds the shortest path between them. Finally driving following the path until it reaches its destination.

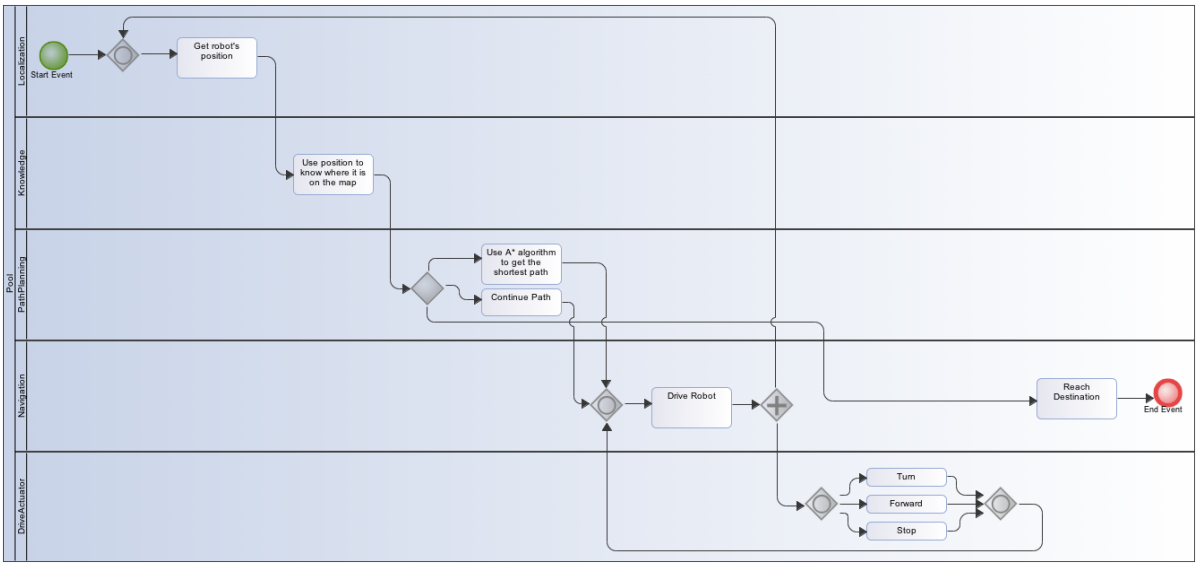

Figure 4: Move to Destination

Figure 5 shows main activities for transporting a product. From the resting area it should first drive to the product unit where the product will be located. After picking up the product

it should carry it to its respective storage unit. Upon releasing the product robot will go back to a available resting area.

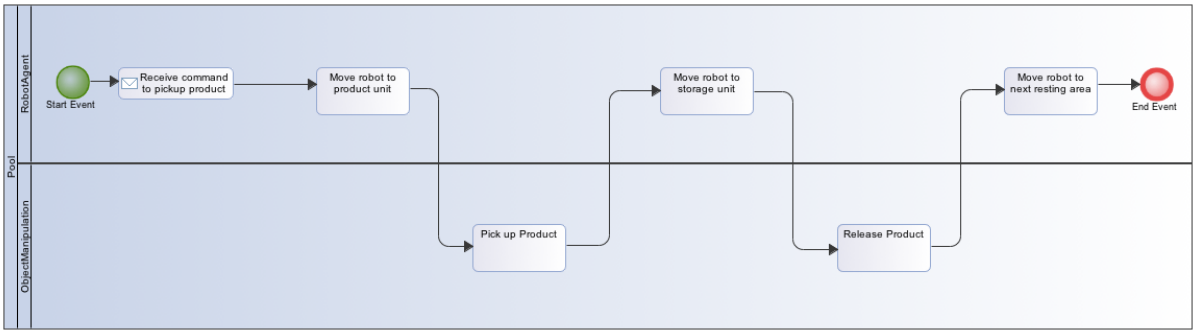

Figure 5: Move Product to storage unit

Figure 6 shows the flows of both picking and releasing the product divided in two distinct pools. For picking up a product we expect the robot to be near it in the product unit. A picture is taken and the exact position of the product is identified by image processing. The robot is then adjusted to the product position until its align to pick it up without flipping it. The gripper will lower its arms, close them around the product and raise them again to start transporting. For releasing the product, when in the storage unit, the robot will stop moving. Standing still grippers arms are going to lower, then opening, leaving the product on the floor. The gripper will then raise its arms and the robot will drive backwards a little, before start going back to a resting area.

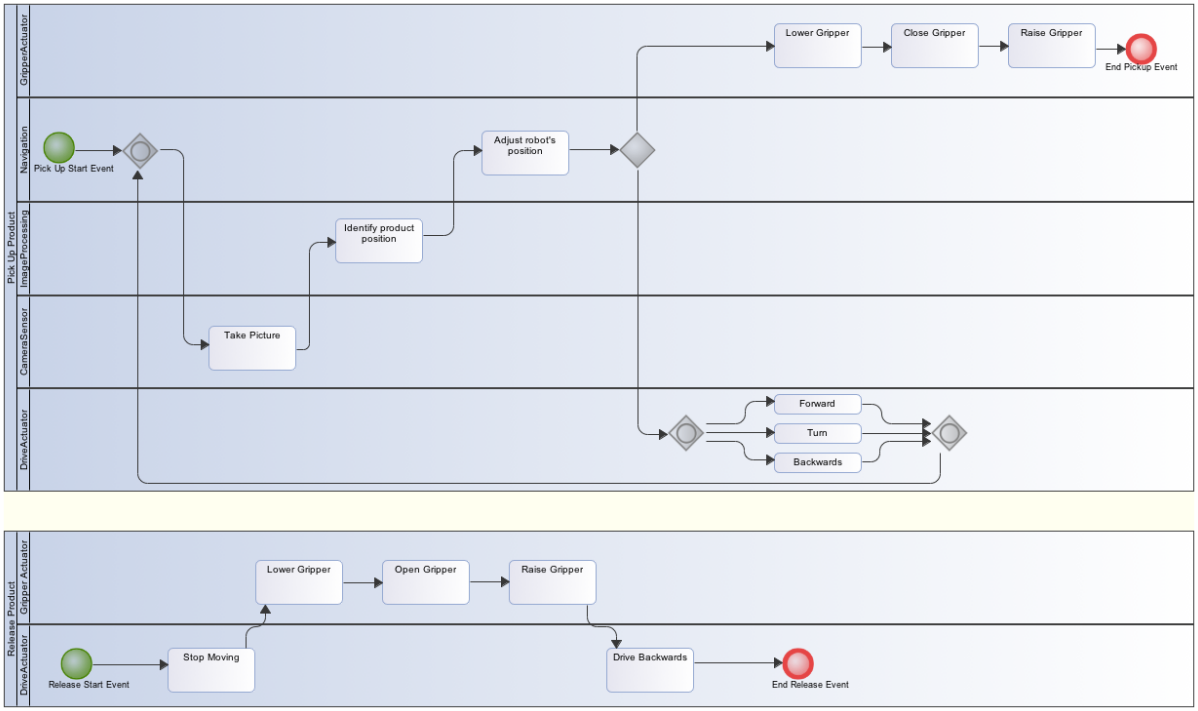

Figure 6: Interact with Product

2.2 Decompose the Robotic Application

Capabilities are identified based on the analysis of BPNM diagrams, and instantiated from the reference architecture. BPNM lanes are usually modeled in a way that one or more are potential capabilities. A relationship between functional requirements of the project and requirements from the reference architecture implies in a relationship between BPNM lanes and domain concepts form the taxonomy. Using concepts to classify BPNM lanes narrows down the possible capabilities they could be an instance of from the reference architecture, thus making it easier to identify the correct capabilities. Figure 7 illustrates this process for identifying capabilities.

Figure 7: Process to identify capabilities

Table 1 presents the relationship between RefSORS requirements, the project functional requirements, domain concepts defined by the taxonomy and identified capabilities. Note that not all of RefSORS requirements were relatable to the project. Some of them are related to non-functional requirements, like quality of service, or aspects not addressed in it. Although control is identified similar to navigation, since it presented different functionalities, when related to requirements it was classified closer to a robotic agent.

Functional Requirements	RefSORS Requirement	Domain Concept	Capability
FR1, FR2, FR4, FR34	R-R1	Application	Application
FR2, FR3	R-R2		
FR1, FR4, FR8, FR9	R-R3		
FR1, FR2, FR3	R-R4		
FR4, FR18, FR34	R-R5		
FR5, FR6, FR22, FR23, FR29	R-R7	Robotic Agent	Robotic Agent
FR14, FR27, FR28			Control
FR5, FR6, FR29, FR3	R-R8	Robotic Agent	Robotic Agent
FR14, FR27, FR28			Control
FR19, FR20, FR33, FR35	R-R12	Task	Operation
FR25, FR26	R-R13	Task	Object Manipulation
FR7	R-R14	Task	Localization
FR30, FR31, FR32	R-R15	Task	Navigation
FR10, FR16, FR17	R-R16	Task	Path Planning
FR11, FR12, FR13	R-R17	Task	Mapping
FR13	R-R18		
FR21, FR24, FR37	R-R19	Task	Image Processing
FR15	R-R20	Knowledge	Map Information
FR15	R-R22		
FR36	R-R24	Device Driver	Camera Sensor
FR40			Laser Sensor
FR41			Sonar Sensor
FR42			GPS Sensor
FR38	R-R29	Device Driver	Drive Actuator
FR39			Gripper Actuator

Table 1: Mapping requirements, domain concepts and capabilities

Figure 8 presents a SoaML[5] Capability Diagram modeled based on identified capabilities, BPNM diagrams and RefSORS capabilities as a template. This diagram represents capabilities, functionalities and their relationship of dependency.

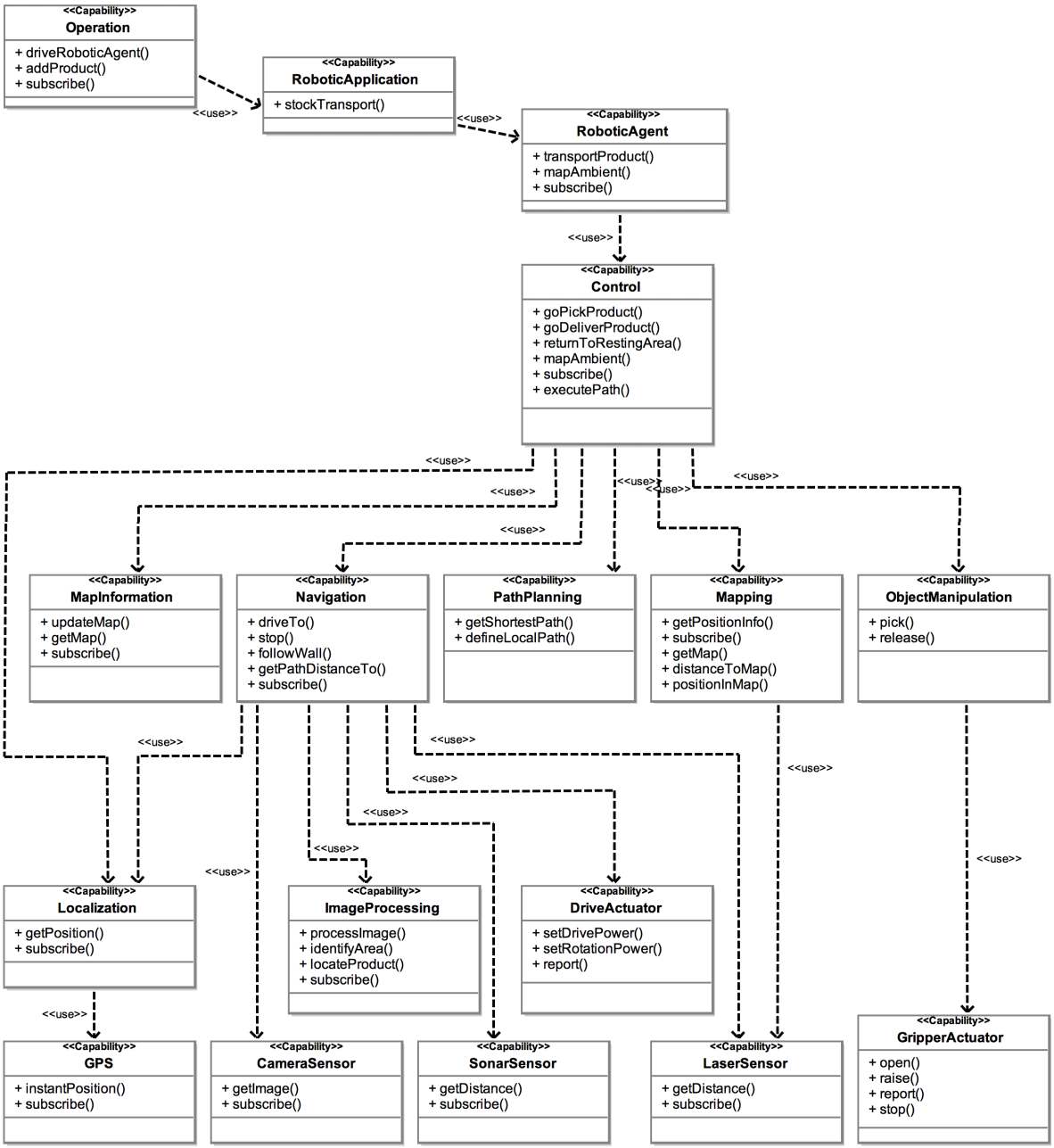

Figure 8: Capability Diagram

2.3 Rationalize Capabilities

After identifying capabilities it is important to verify for each of them if it can be implemented as a service. If a capability is too complex it can be divided into multiple services. If it is to simple it can be part of another service. Table 2.3 describes for each capability if and why it should be a service. Capabilities for controlling sensors and actuators were grouped since they all from the device driver concept and control only one device.

Capability	Description	Service
All Sensors and Actuators	Coherent and cohesive, needed for controlling sensor or actuator devices. Also each deals with only one device.	yes
Image Processing	Cohesive, deals only with extraction of information from images. Used to identify objects and marked areas on the floor.	yes
Localization	Only translates gps coordinates, needed for better localization of robots.	yes
Object Manipulation	An abstraction of picking and releasing, deals only with product interaction.	yes
Operation	Deals only with user interface, needed when the autonomy of a robot fails.	yes
Mapping	Cohesive, dealing only with map representation.	yes
Path Planning	Cohesive, needed for navigation, only plans paths.	yes
Navigation	Coherent, coordinates capabilities for a safe navigation.	yes
Map Information	Coherent, stores the map for shared access from robots.	yes
Control	Deals only with logical decisions, required to coordinate other capabilities.	yes
Robotic Agent	Coherent, deals only with abstraction of robots functionalities.	yes
Robotic Application	Represents the application, deals only with controlling robotic agents.	yes

Table 2: Rationalizing Capabilities

3 Step RA-3 Robotic Architecture Modeling

In this step identified services are described and related to the functional requirements they fulfill. Their interface, contract and protocol are also described and represented by SoaML diagrams. Relationship among services, based on the obligations of provider and consumer defined by services contracts, are presented in a service architecture diagram.

3.1 Specify Robotic Services

Gripper Actuator Service

Its an instance of actuator driver service that exposes the capability for controlling a gripper device. More detail about gripper interface at subsection 3.2.1.

Related requirements: FR39.

Drive Actuator Service

Its an instance of actuator driver service that exposes the capability for controlling a differential drive. More detail about the driving interface check subsection 3.2.2.

Related requirements: FR38.

Laser Sensor Service

An instance of sensor driver service that provides distances from a laser range finder. Can provide periodic distance updates or instant distance information from a selected angle.

Related requirements: FR40.

Camera Sensor Service

An optical sensor, instance of sensor driver service, that provides functionalities for capturing images from a camera device. Can provide frequent updates or instant capture of an image.

Related requirements: FR36.

Sonar Sensor Service

An instance of sensor driver that provides distance information from sonar sensor. Can provide frequent updates or instant distance measure.

Related requirements: FR41.

GPS Sensor Service

A position sensor service, which provides position data from a GPS device. Can provide frequent updates or instant position information.

Related requirements: FR42.

Operation Service

An instance of supporting service, provides the operator with an interface to register a transport order, monitor the application and control robots when they fail to navigate autonomously.

Related requirements: FR19, FR20, FR33, FR35.

Image Processing Service

Also an instance of supporting service, it provides useful information from images that come from the camera. Can Identify reference areas on the floor, that represents resting areas, product units or storage units. Identifies product position relative to the robot helping dock the robot more precisely when picking a product up.

Related requirements: FR21, FR24, FR37.

Localization Service

Uses absolute position data from GPS sensor service to provide localization of the robot related to the ambient.

Related requirements: FR7.

Mapping Service

Provides position of the robot on the map and information about any space on it. Uses laser sensor service distances to build a 2D map representation of the ambient.

Related requirements: FR11, FR12, FR13.

Path Planning Service

Provides global and local shortest paths between two points. Its used by control service to guide the navigation service. Uses map information from mapping service and robot position from localization services to search for the shortest path.

Related requirements: FR10, FR16, FR17.

Object Manipulation Service

Provides interaction with the product in the vertical position as an abstraction to gripper actuator service. Coordinates the gripper to pick or release a product without letting it fall or flipping it over.

Related requirements: FR25, FR26.

Navigation Service

Provides navigation for the robot following paths provided by control and avoiding collision. Uses laser sensor service to avoid frontal collisions and sonar sensor services to avoid rear-end collisions. Uses image processing service for providing identification of marked areas and combine with localization service for better orientation.

Related requirements: FR30, FR31, FR32.

Map Information Service

Its a instance of knowledge service that provides a way for one or more robots to access

and update map information about their environment.

Related requirements: FR15.

Control Service

Controls logical decisions of the robot. Coordinates mapping, map information, path planning, navigation and object manipulation services. Controls information access and update from mapping to map information. Uses localization and mapping to request a path from path planning. Defines a destination for navigation using the path from path planning. Provides feedback from its tasks to the robotic agent.

Related requirements: FR14, FR27, FR28.

Robotic Agent Service

Represents a robot, provides information about it for the application and delegate tasks to the control service.

Related requirements: FR5, FR6, FR22,FR23, FR29.

Robotic Application Service

Represents the application itself, controlling two robotic agents cooperatively to achieve its goal. Provides information about its progress and means to receive product orders.

Related requirements: FR1, FR2, FR3, FR4, FR8, FR9, FR18, FR34.

All functional requirements are related to a service, meaning all necessary functionalities are fulfilled by them.

3.2 Model Robotic Services

3.2.1 Gripper Actuator Service

Figure 9 describes gripper actuator service interface, contract and protocol. Four main functionalities are provided from the service interface: (i) open, that changes the distance between gripper arms; (ii) raise, that changes the height of the gripper arms; (iii) report informs the current state of the actuator; (iv) stop, which is used for stopping an on going action. Open, raise and stop functionalities are instances of execute functionality.

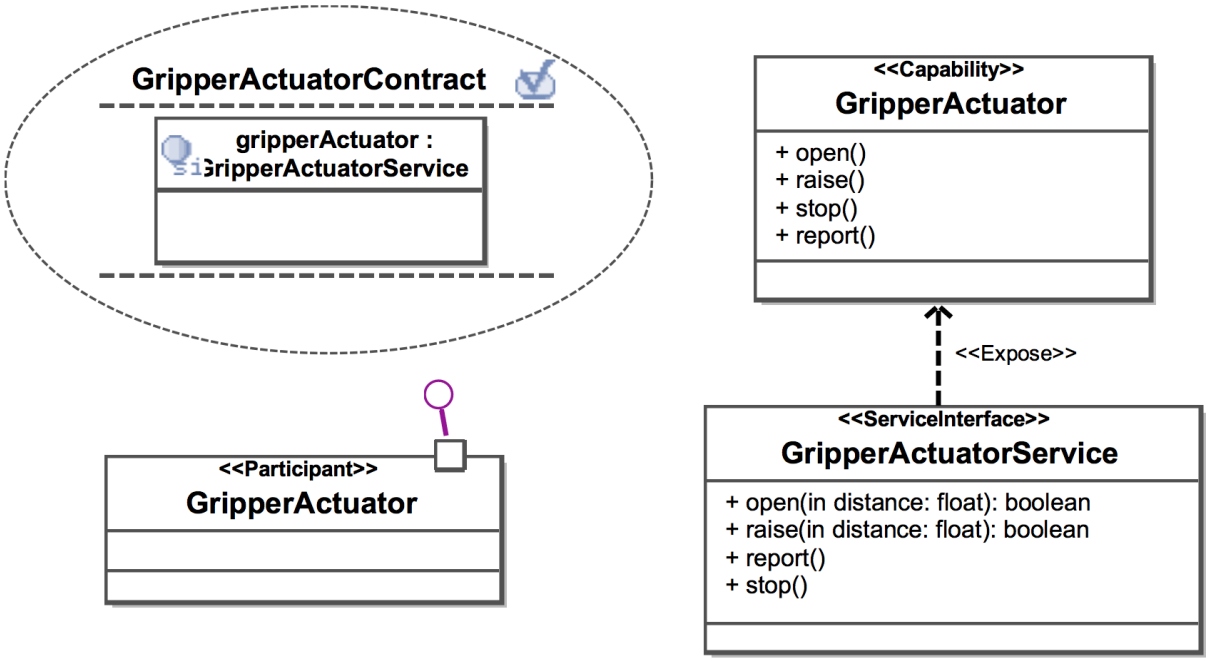

Figure 9: Service interface, contract, and protocol of Gripper Actuator service

3.2.2 Drive Actuator Service

Figure 10 describes drive actuator service interface, contract and protocol. Its service interface offers three functionalities: (i) setDriverPower, which changes the power of the motor moving both wheels; (ii) setRotatePower, that changes the power of each wheel differently to turn the robot by a specified angle; (iii) report, which informs the current state of the actuator. SetDriverPower and setRotatePower functionalities are both an instance of execute.

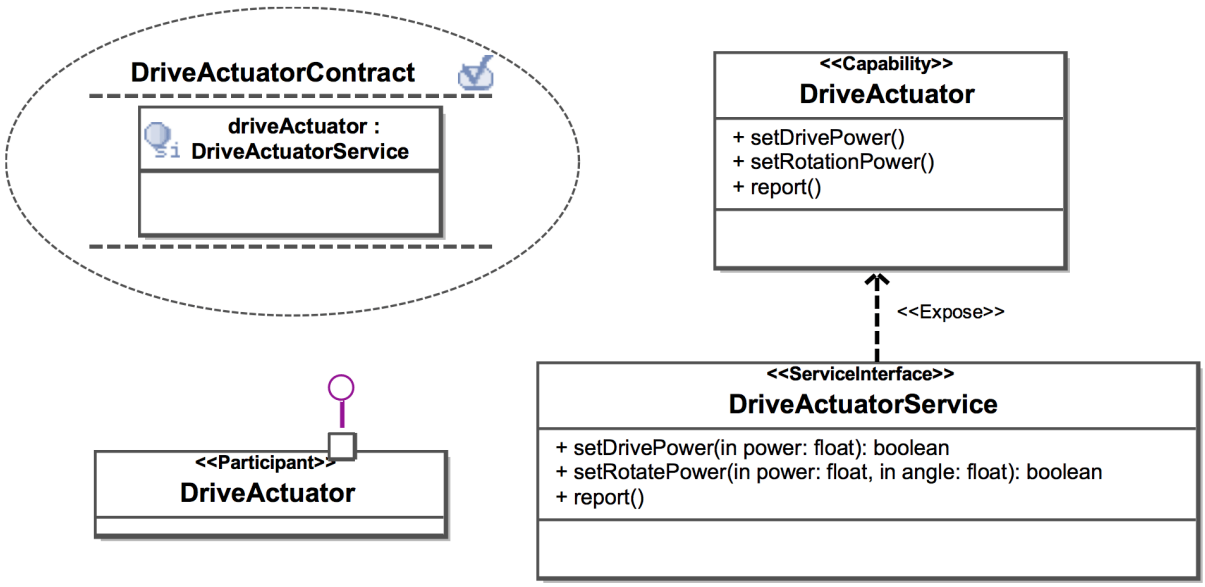

Figure 10: Service interface, contract, and protocol of Drive Actuator service

3.2.3 Laser Sensor Service

Figure 11 presents laser sensor, service interface, contract and protocol, which provides two main functionalities: (i) getDistance, an instance of measure, that provides instant value of distance for a desired angle; (ii) subscribe, that send period updates of distance information. Subscribe requires the implementation of functionalities from LaserSensorHandler interface from service consumer. It also implies upon requesting a subscription that all responses will be sent to handleDistanceChange functionality.

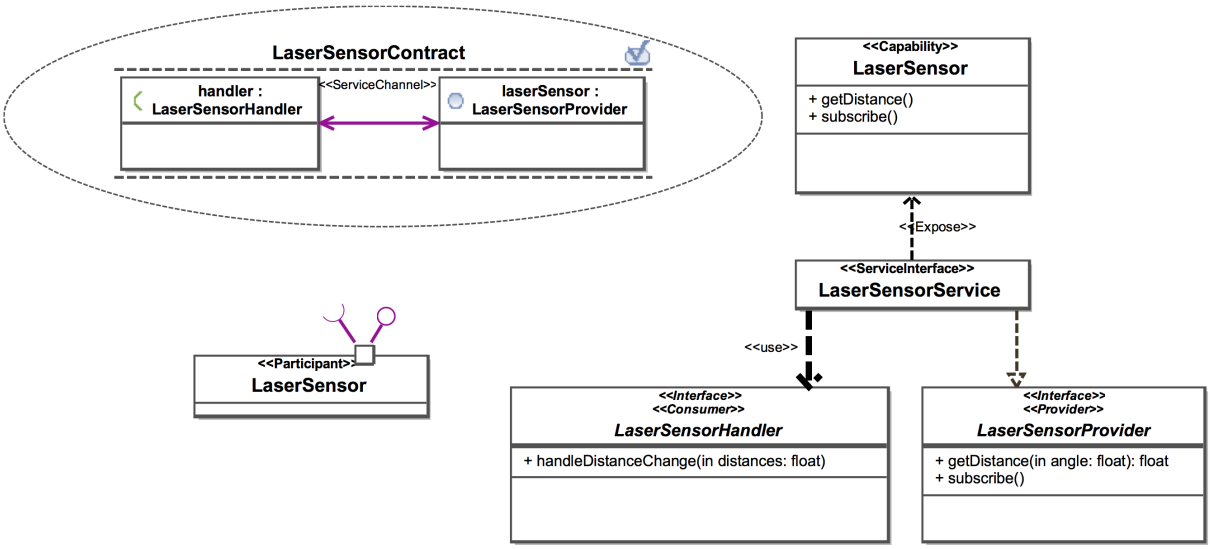

Figure 11: Service interface, contract, and protocol of Laser Sensor service

3.2.4 Camera Sensor Service

Figure 12 describes camera sensor service interface, contract and protocol. Service Interface presents two functionalities: (i) getImage, an instance of measure, which provides an instant image from the camera; (ii) subscribe, which send periodic images from the camera. Much like laser sensor service, the subscription functionality requests a required interface and protocol.

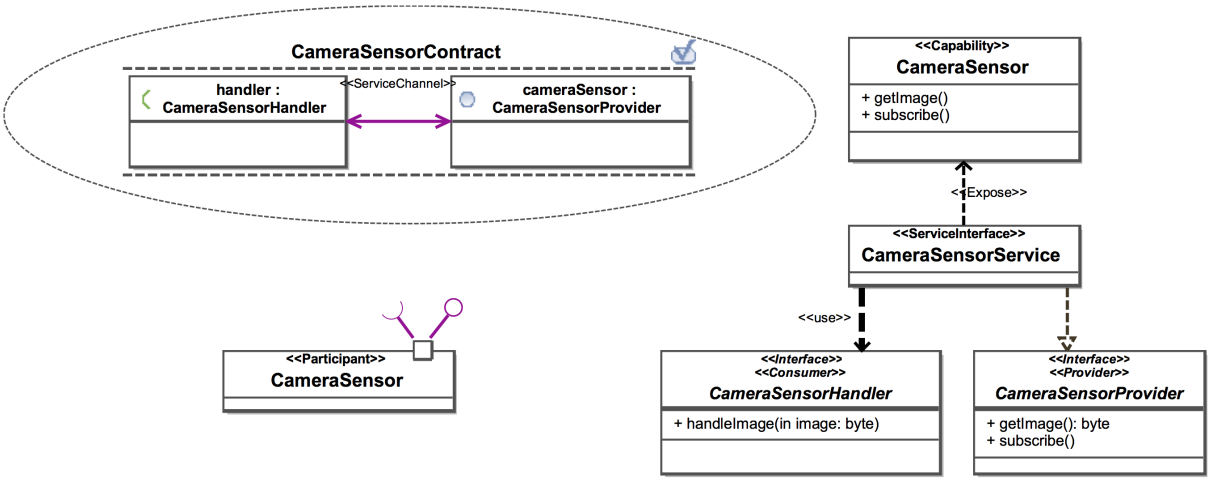

Figure 12: Service interface, contract, and protocol of Camera Sensor service

3.2.5 Sonar Sensor Service

Figure 13 describes sonar sensor service interface, contract and protocol. Two main functionalities are provided by the service interface: (i) getDistance, an instance of measure, that provides instant value of distance for desired angle; (ii) subscribe, which sends periodic information of distances change. Upon subscribing the protocol requires that service consumer provides the proper implementation of the SonarSensorHandler required interface. Also implying that later on response messages will be sent to handleDistanceChange defined by the protocol in its contract.

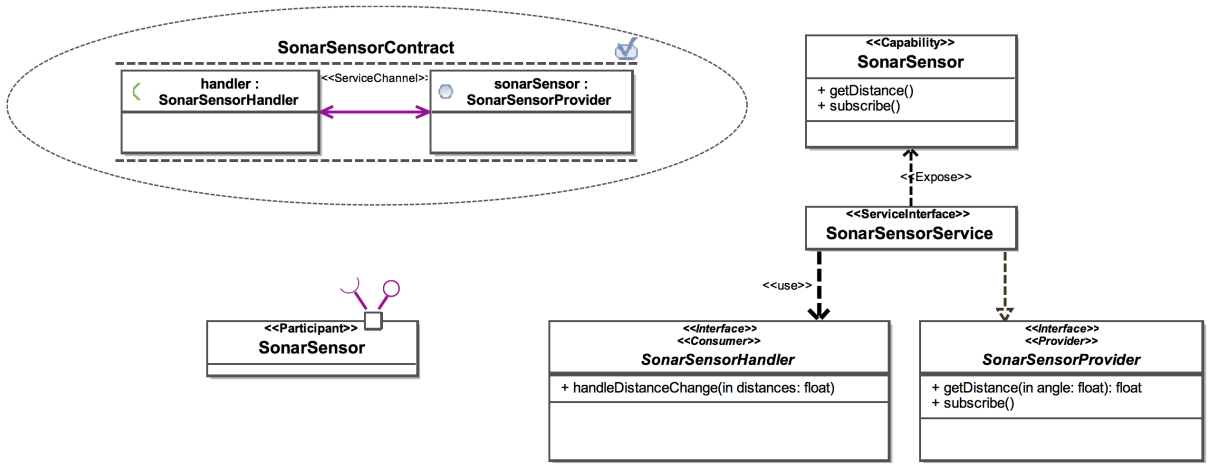

Figure 13: Service interface, contract, and protocol of Sonar Sensor service

3.2.6 GPS Service

Figure 14 describes gps service interface, contract and protocol. Service Interface presents two functionalities: (i) instantPosition, an instance of measure, that provides an instant value of

the robot's position; (ii) subscribe, that send period position information. As in other sensor services, subscribe functionality demands the use of a compatible required interface and protocol.

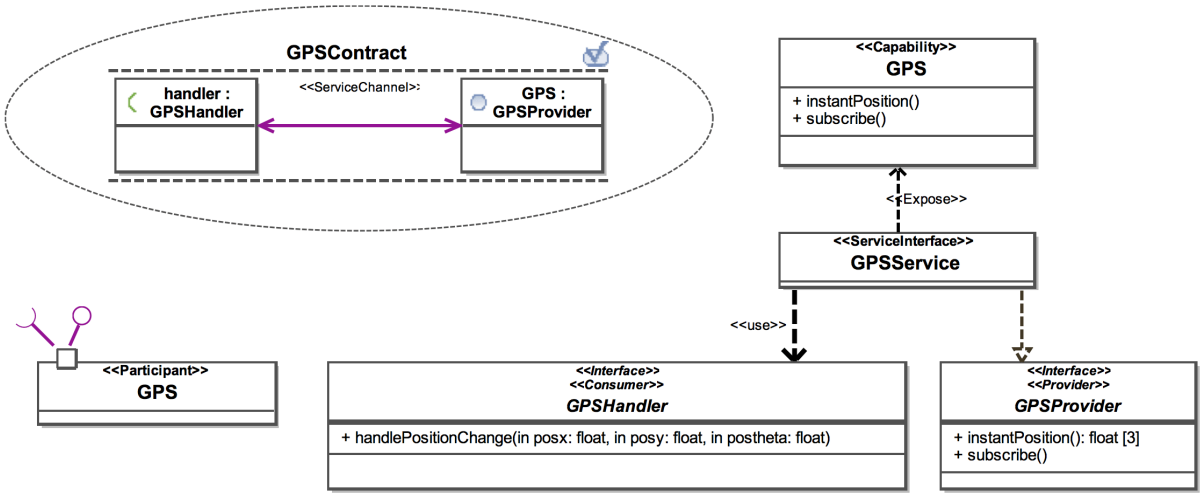

Figure 14: Service interface, contract, and protocol of GPS service

3.2.7 Operation Service

Figure 15 describes operation service interface, contract and protocol. Service Interface offers three main functionalities: (i) driveRoboticAgent, which allows controlling a robotic agent when it fails; (ii) addProduct, adds a product transport order giving a products position; (iii) subscribe, provides period information about the system. Support services have only execute functionality, therefore all these functionalities are an instance of execute. Also the subscription protocol implies that later on service consumer will be notified through handleApplicationChange.

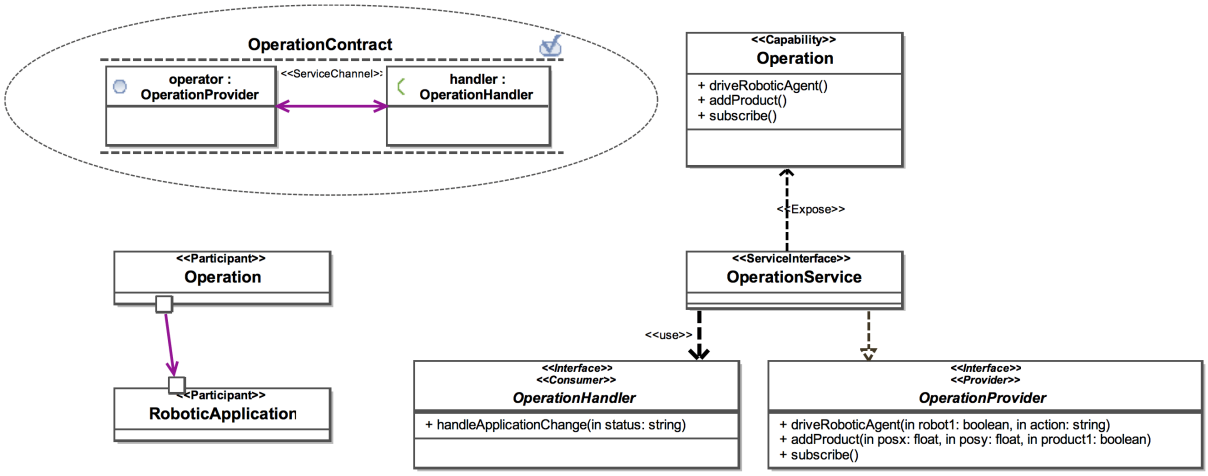

Figure 15: Service interface, contract, and protocol of Operation service

3.2.8 Image Processing Service

Figure 16 describes image processing service interface, contract and protocol. The functionality processImage, will receive an image and provide information about any marked area on the floor or recognized product using both of its other functionalities. IdentifyArea functionality will search for colored marks on the floor to locate special areas. LocateProduct functionality will calculate the distance and angle between robot and product. As the operation service, all functionalities are an instance of execute.

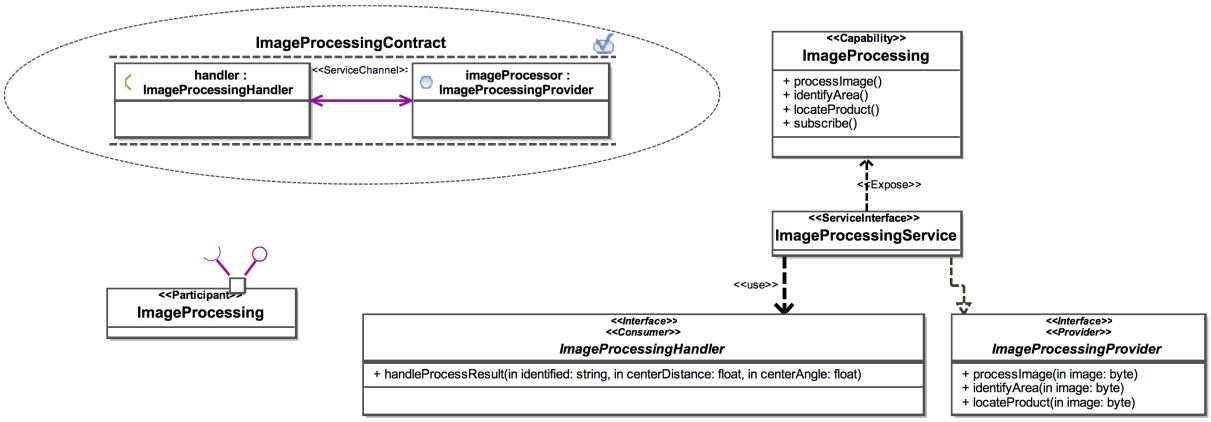

Figure 16: Service interface, contract, and protocol of Image Processing service

3.2.9 Localization Service

Figure 17 describes localization service interface, contract and protocol. Two main functionalities are presented by service interface: (i) getPosition, an instance of getLocalization, which provides a instant position of the robot in the environment; (ii) subscribe, that provides periodic position values of the robot. Service subscription also demands service consumer to provide handler functionalities for its contract.

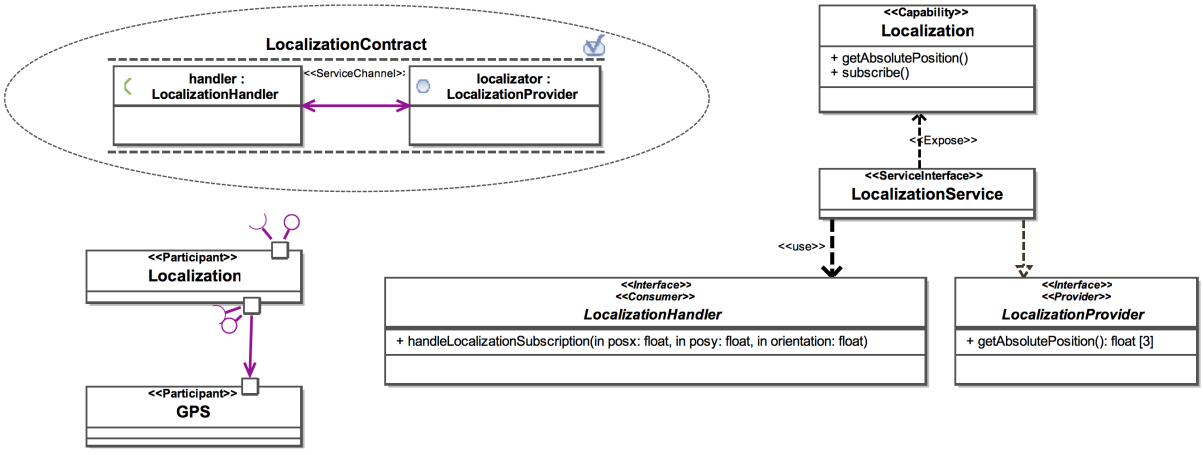

Figure 17: Service interface, contract, and protocol of Localization service

3.2.10 Mapping Service

Figure 18 describes mapping service interface, contract and protocol. Service Interface presents five functionalities: (i) getMap, which provides the current representation of the environment; (ii) getPositionInfo, which provides the value of a given space in the map, whether the space is empty or not; (iii) subscribe, which sends periodic information about the map; (iv) distanceToMap, which uses laser sensor data to construct the map representation; (v) positionInMap, which receives a position and returns a position on the map. Subscription demands service consumer to implement MappingHandler required interface.

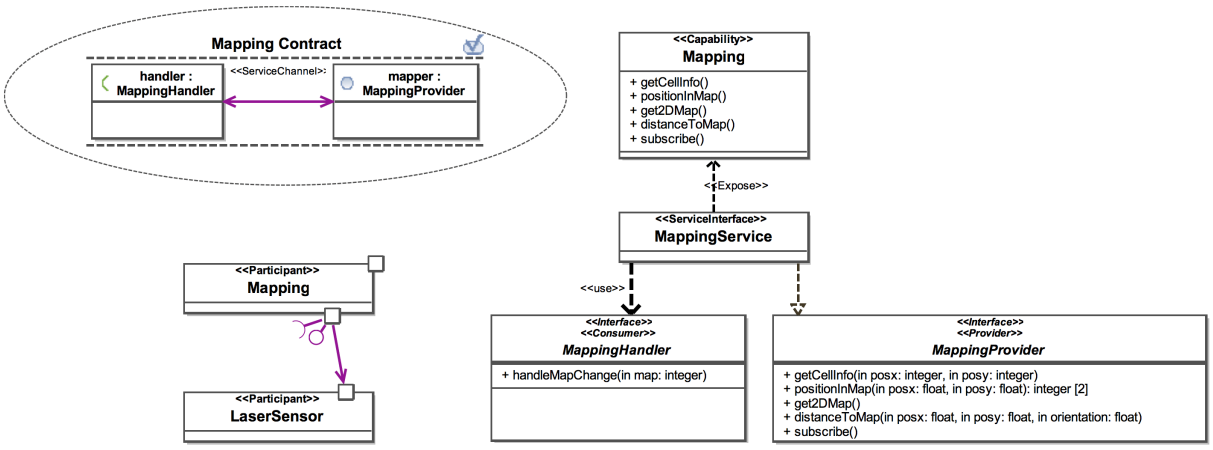

Figure 18: Service interface, contract, and protocol of Mapping service

3.2.11 Path Planning Service

Figure 19 describes path planning service interface, contract and protocol. Service Interface presents two functionalities: (i) getShortestPath, that provides the shortest global path between the robot and its goal; (ii) defineLocalPath, which provides the shortest local path between the robot and its next destination in the global path.

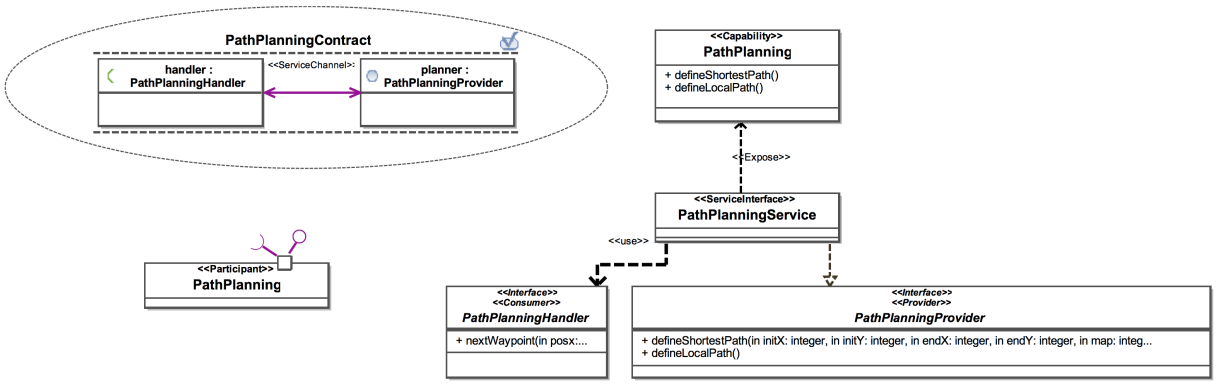

Figure 19: Service interface, contract, and protocol of Path Planning service

3.2.12 Object Manipulation Service

Figure 20 describes object manipulation service, service interface, contract and protocol. Service Interface provides two functionalities: (i) pick, that picks a product; (ii) release, that releases a object in the vertical position.

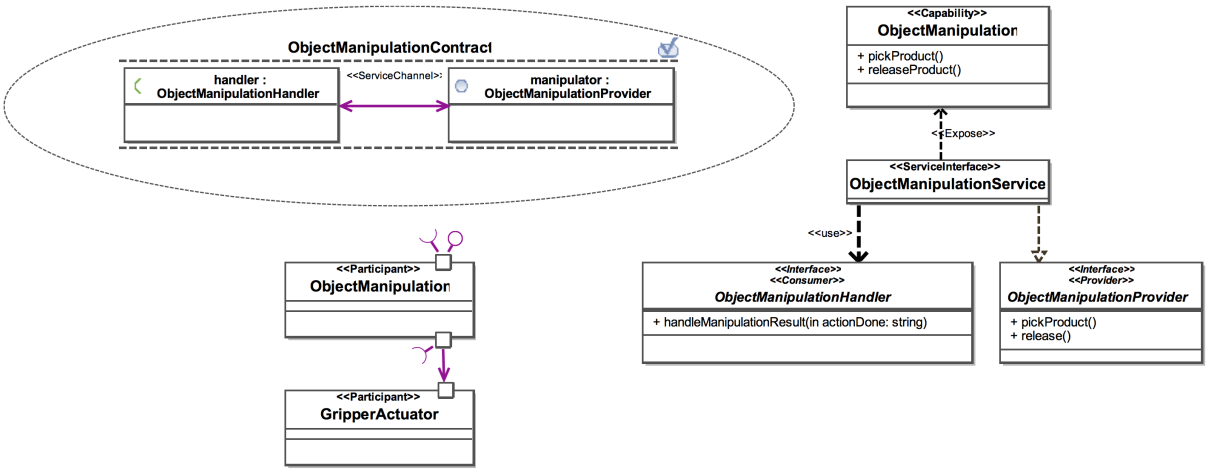

Figure 20: Service interface, contract, and protocol of Object Manipulation service

3.2.13 Navigation Service

Figure 21 describes navigation service interface, contract and protocol. Service Interface presents four main functionalities: (i) driveTo, which drives the robot to desired position; (ii) stop, that reduces robots velocity to zero, can be used to avoid collisions or in case of emergency; (iiI) followWall, which uses the laser sensor distances to follow a wall; (iv) subscribe, that send periodic information about navigation. Subscribe functionality demands the use of a compatible required interface and protocol.

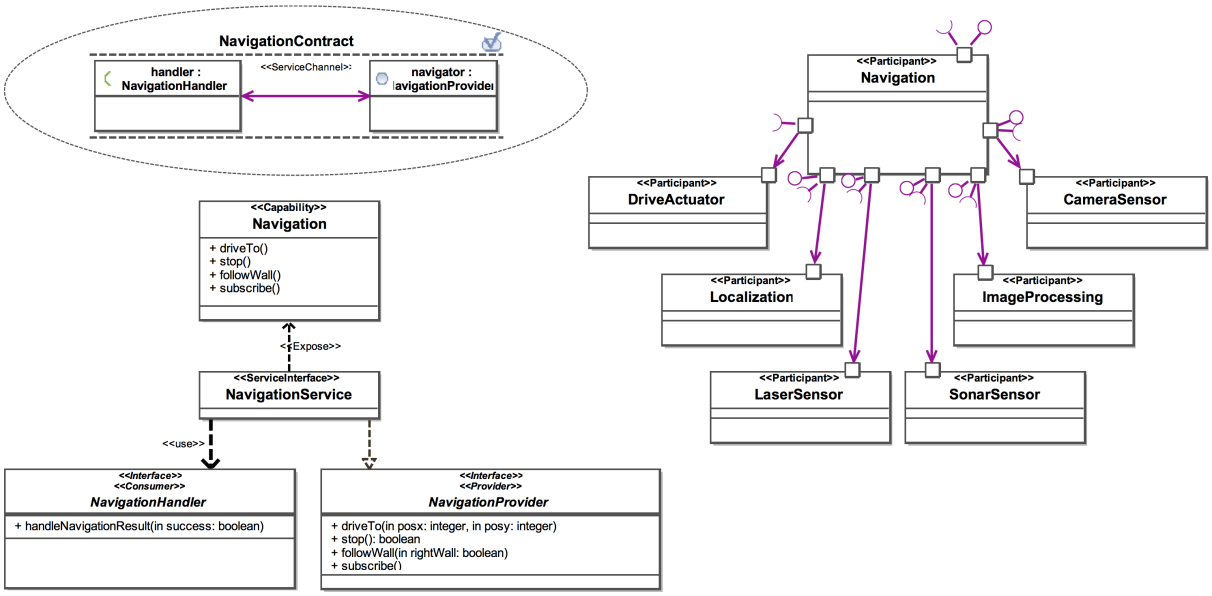

Figure 21: Service interface, contract, and protocol of Navigation service

3.2.14 Map Information Service

Figure 22 describes map information service interface, contract and protocol. The following functionalities are presented by its service interface: (i) updateMap, an instance of updateKnowledge, which changes the value of a position in map; (ii) getMap, an instance of getKnowledge, that provides the whole map; (iii) subscribe, which sends periodic information about map updates. As in many other services, subscription requires the proper implementation of its interface and protocol.

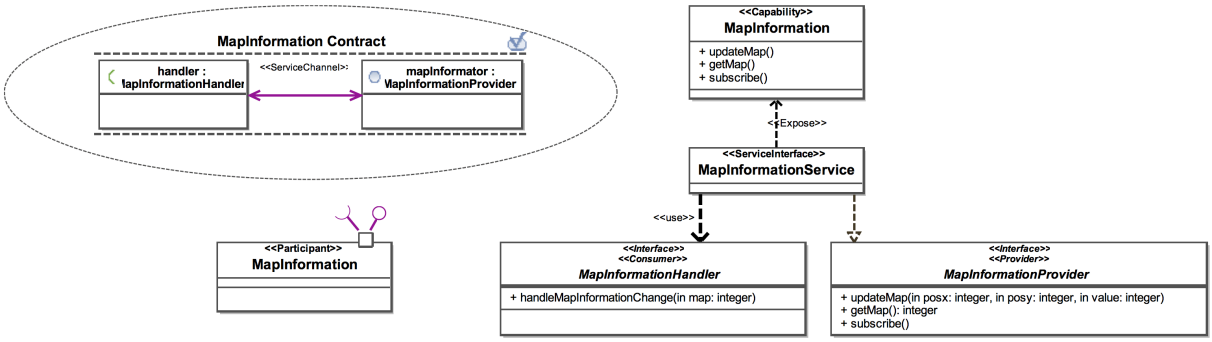

Figure 22: Service interface, contract, and protocol of Map Information service

3.2.15 Control Service

Figure 23 describes control service interface, contract and protocol. Service Interface presents six functionalities: (i) goPickProduct, which instruct services to make the robot get to the product and pick it; (ii) goDeliverProduct, which instruct services to carry the product to its

storage unit and release it; (iii) returnToRestingArea, that finds a free resting area and makes the robot go there; (iv) mapAmbient, which commands services to map the environment; (v) subscribe, that sends periodic information about the tasks progression; (vi) executePath, that handles path planning responses. Subscribe functionality also requires from service consumer a handler functionality named handleTaskResult.

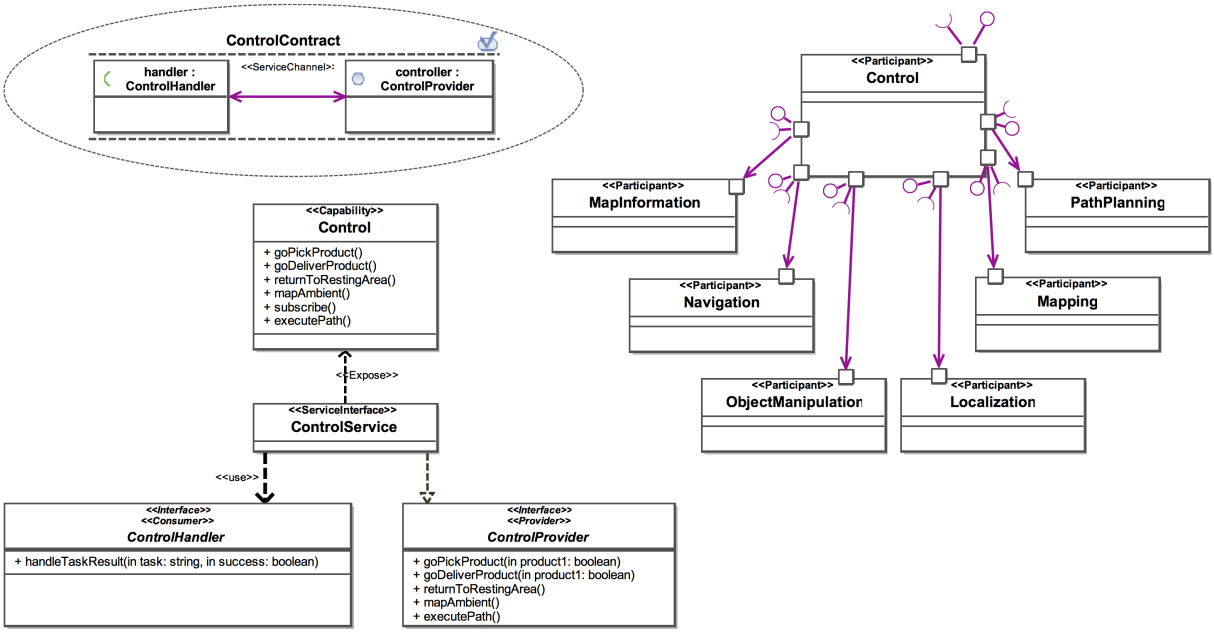

Figure 23: Service interface, contract, and protocol of Control service

3.2.16 Robotic Agent Service

Figure 24 describes robotic agent service interface, contract and protocol. Service Interface describe three functionalities: (i) transportProduct, that carries a product from its product unit to its storage unit then moves the robot back to a resting area; (ii) mapAmbient, that maps the environment following a wall then makes the robot go back to a resting area; (iii) subscribe, which reports progress on the other two functionalities. Subscription demands service consumer to implement required interface of RoboticAgentHandler.

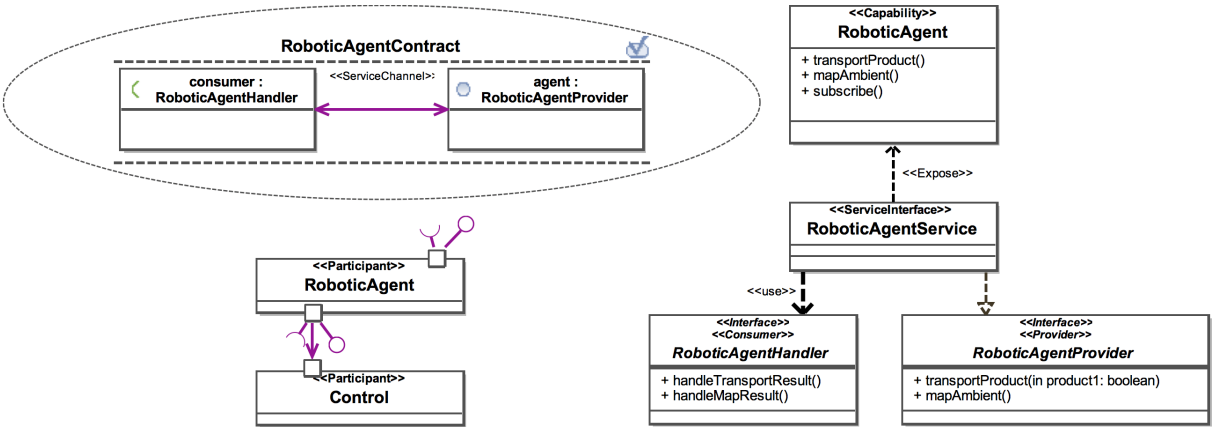

Figure 24: Service interface, contract, and protocol of Robotic Agent service

3.2.17 Robotic Application Service

Figure 25 describes robotic application service interface, contract and protocol. Service Interface offers only one functionality: (i) stockTransport, an instance of executeApplication, which controls both robots cooperatively, chooses the best one for the transport order and monitors transport operations. Consumer must provide a functionality of handleApplicationResult to be notified of its progression.

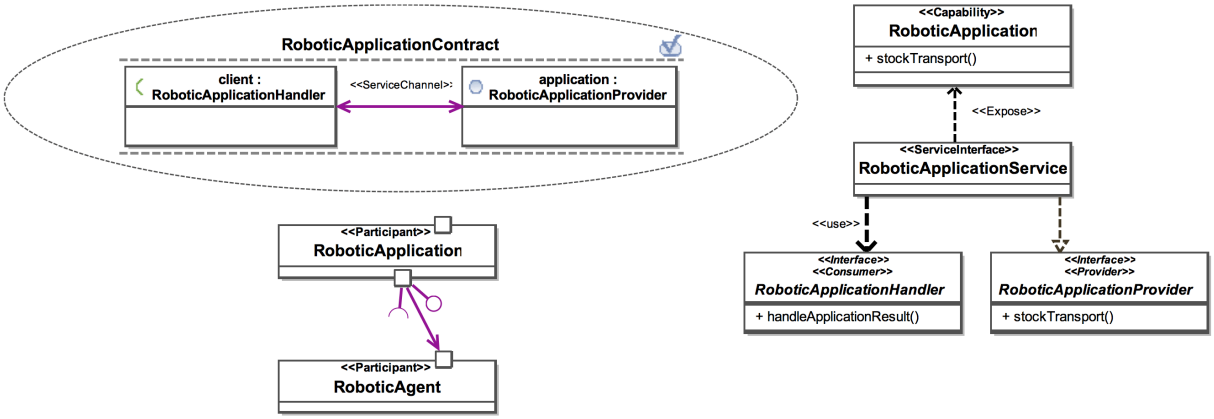

Figure 25: Service interface, contract, and protocol of Robotic Application service

3.3 Define Robotic Services Composition

On section 3.2 contracts were defined for each service, however it was not defined which services will be using these contracts in the context of this project. Figure 26 presents this project consumers and providers and contracts that bound them together represented in a SoaML Services Architecture diagram.

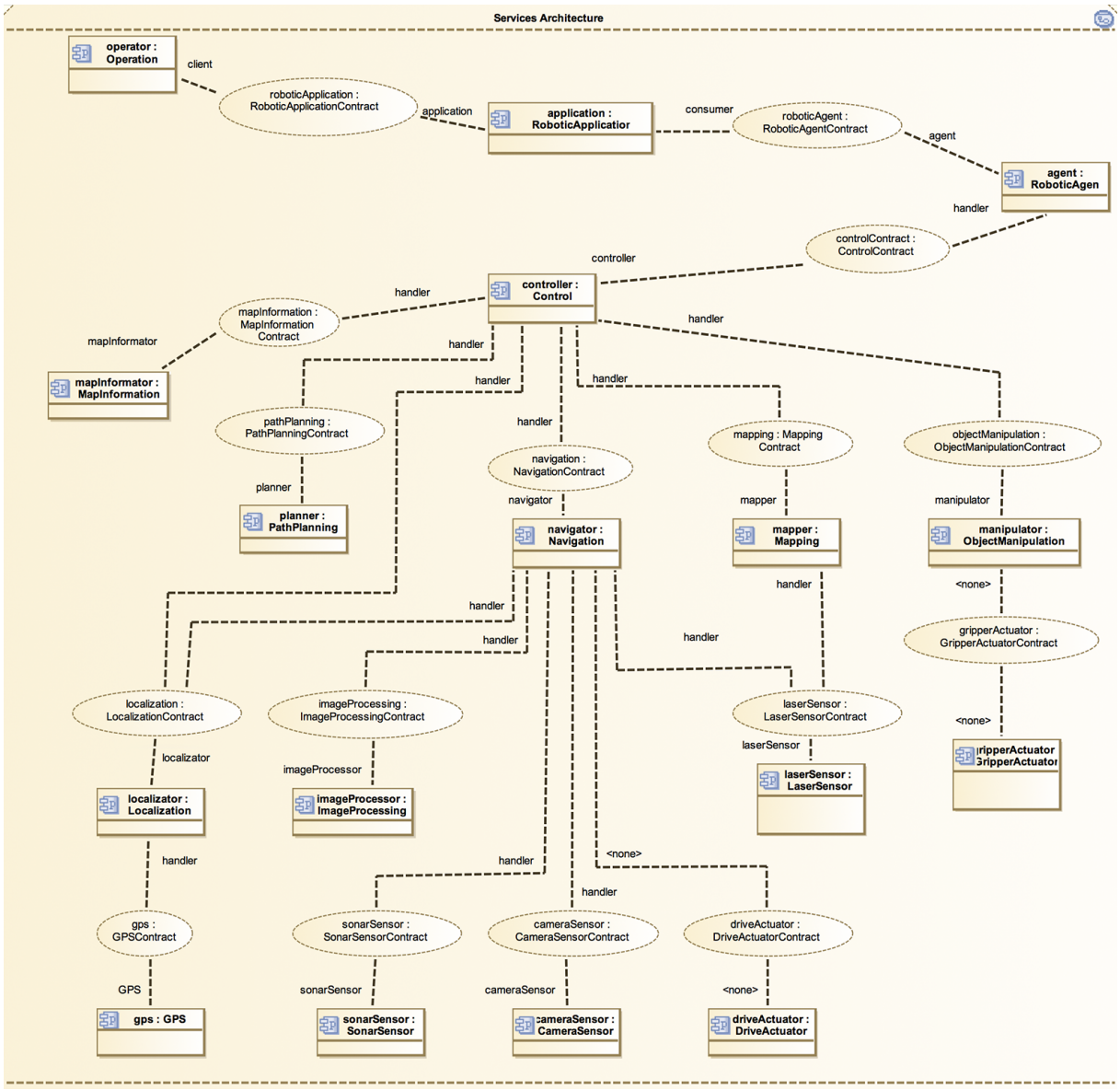

Figure 26: Services Architecture Diagram

4 Step RA-4 Robotic Architecture Detailing

This step will describe implementation strategies during concretization of services for the context of this project. Also deployment decisions on which services will be deployed locally in the robot, on a back-end server in a notebook or on the client.

4.1 Implementation Strategies

It is yet to be defined whether the identified services are going to come from online repositories or developed for the project. In case they are developed for the project these are the implementation

strategies that will be used. Hardware infrastructure will be as described in the RobAFIS documentation, for the context of this project. Notice that a simulated versions of the hardware will be used.

Gripper Actuator Service

It will be controlling a 2 axis, 2 degree-of-freedom robotic gripper. This gripper arms opens and closes horizontally to grasp an object, and raises and lowers when carrying the grasped object or putting it back on the floor, respectively.

Drive Actuator Service

It will be controlling a 2-wheel differential drive, with rear balancing caster. The two front wheels can be controlled independently to set different speeds when it needs to turn.

Laser Sensor Service

It will be an abstraction of a laser rangefinder sensor. Laser rangefinder is a type of rangefinder that uses laser beam to determine the distance to an object. This specific rangefinder will provide a range of 180 measues of distance in a single plan and reachability of 8.000mm.

Camera Sensor Service

It controls a indoor pan-tilt-zoom (PTZ) camera. This camera features autofocus and automatic brightness/gain control, and an image resolution of up to 704x576. This makes possible the processing of images needed to identify colored marked areas in the floor.

Sonar Sensor Service

It is an abstraction of eight sonar sensors located at the front of the robot. Sonar sensors are another way to measure distances from objects, this has a reachability of 3.000mm within 8 diferent angles.

GPS Sensor Service

It will be controlling a GPS device that provides absolute position and orientation of the robot.

Image Processing Service

Image Processing will be used for two purposes, identify colored marked areas on the floor and position of a product. To identify marked areas there are three different colors to represent them: (i) black, which is a resting area; (ii) blue, which is a product unit; and (iii) red, which is a storage unit. For identifying the product it needs to look for a small green cylinder in the middle of a blue marked area. Upon receiving a request from

navigation with an image as input, it will provide with the found object or area and the distance an angle between the robot, when the picture was taken, and the identified object or area.

Localization Service

It will translate position information from GPS Sensor service to a position relative to the ambient. Since it will be subscribed to the GPS service, as soon as new information about the robot position arrives it calculates the new relative position based on the top left corner of the rectangular ambient.

Object Manipulation Service

It is an abstraction of the gripper service functionalities applied to this project context. The products its going to interact with are cylinders 150mm high, their top and bottom are 10mm high and 50mm diameter, and their center is 130mm high and 20mm diameter. Picking and releasing the product will use this measures to correctly utilize the gripper to carry a product.

Operation Service

Operation service will provide a web user interface from where a operator are able to monitor the application or each individual robot and when needed drive a robot. Monitor the application will collect data from it and each of its robots to display battery life, which task they are doing in real-time, which robot is available, what is the perceived map of the ambient, current paths they are following, their position and orientation and the products queue to be transported. When a robot fail to continue autonomously transporting products the operator will be warned and simple interface to control the robot will be available.

Mapping Service

The environment specified in the projects document is a rectangle area with a size of 10.000mm x 10.000mm and walls 50mm thick. Because of that specification the representation of the map will be a 2D grid with 100mm x 100mm cells. When creating the map, information from laser sensor service will be used to fill cells ocupied by walls. Since both robots will be mapping at the same time each will cover half the map, from resting area 1 to resting area 3 and contrariwise. Figure 27 presents a sequence diagrams of activities performed to create the map.

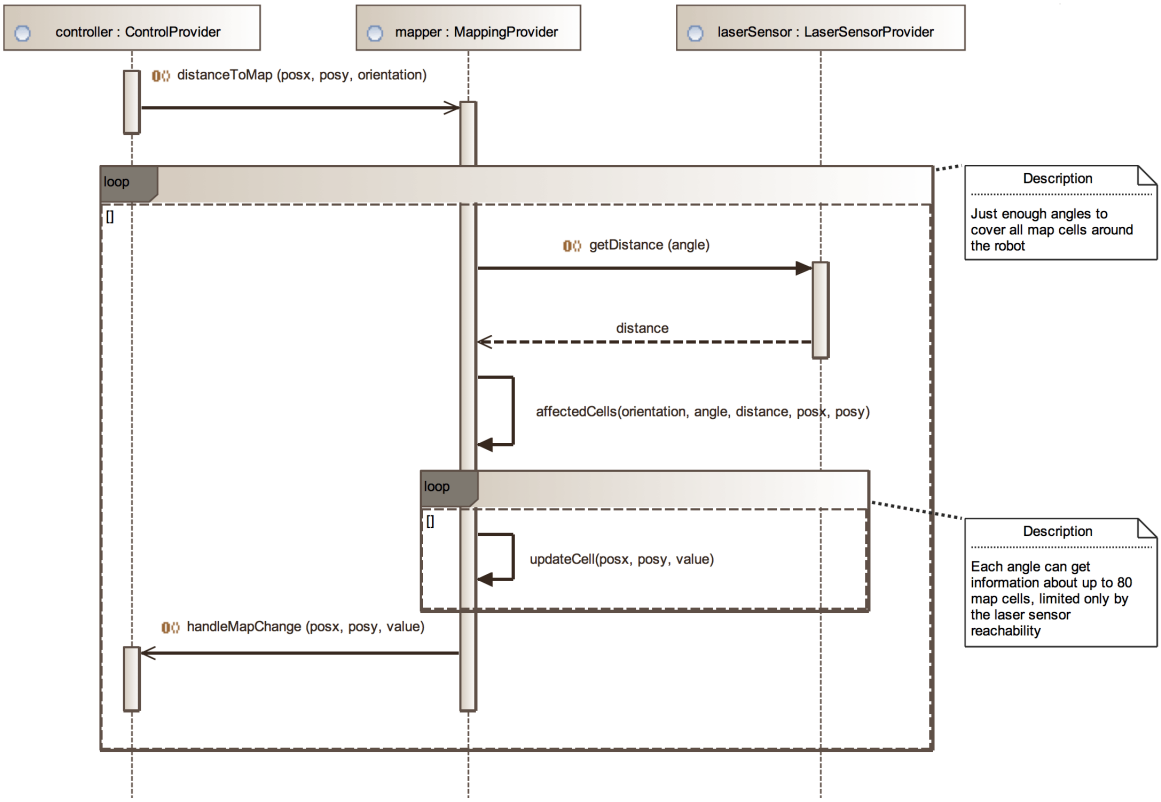

Figure 27: Mapping Sequence Diagram

Path Planning Service

Path Planning provides global and local paths for navigation. Global paths will be implemented using A* algorithm, because of its accuracy and great performance with grid maps. Local paths will be implemented using potential fields a very efficient way to avoid collision with walls and other robots.

Navigation Service

Navigation service will be driving the robot by following directions from the control service or following a wall. When following a wall it will be listening to distance updates from the laser sensor service, so whenever it gets too close or too far it should turn a little to maintain its distance. When following directions from control service, it will receive waypoints to where to drive next. To avoid collision, it will always checking distances from some angles of the laser sensor service and distances from sonar sensor. Image processing will be used when docking to pick up the product or going directly to a marked area. Figure 28 a sequence diagram of the functionality followWall.

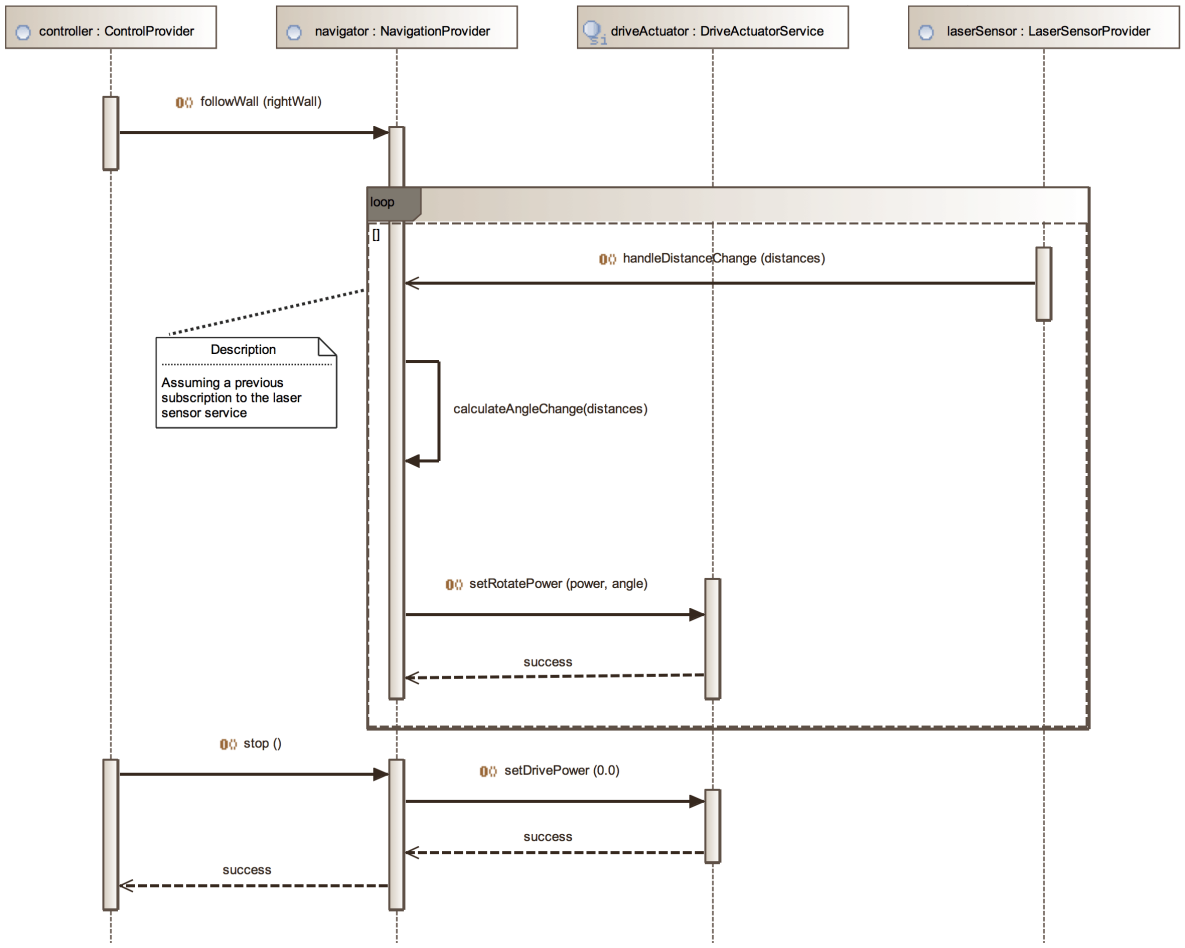

Figure 28: Follow Wall Sequence Diagram

Map Information Service

It will store and give access to a shared map, since the representation of the map will be a grid, it will be storing the map in a simple array with ones and zeros marking occupied and free space respectively. This service will be running in the back-end server to allow both robots to access or update its information.

Control Service

Control service is responsible to coordinate other services to complete tasks given by the robotic agent service. To map the ambient it will use navigation service to drive the robot following a wall, while using mapping service to transform distances from the laser sensor service into useful map information. While mapping it will update and access the map in map information service to get and update new map information from the mapping service. To pick or deliver a product it will get the robot position from localization service, use the position of the robot and the product to get a position of both on the map, also a map representation from the mapping service, then use both to request the shortest path

between them from the path planning service, finally execute this path giving waypoints to navigation until it reaches its destination. Returning to a resting area is not so different, with the exception that if occupied the robot must go to another resting area. Figure 29 illustrates control executing its functionality mapAmbient by a sequence diagram.

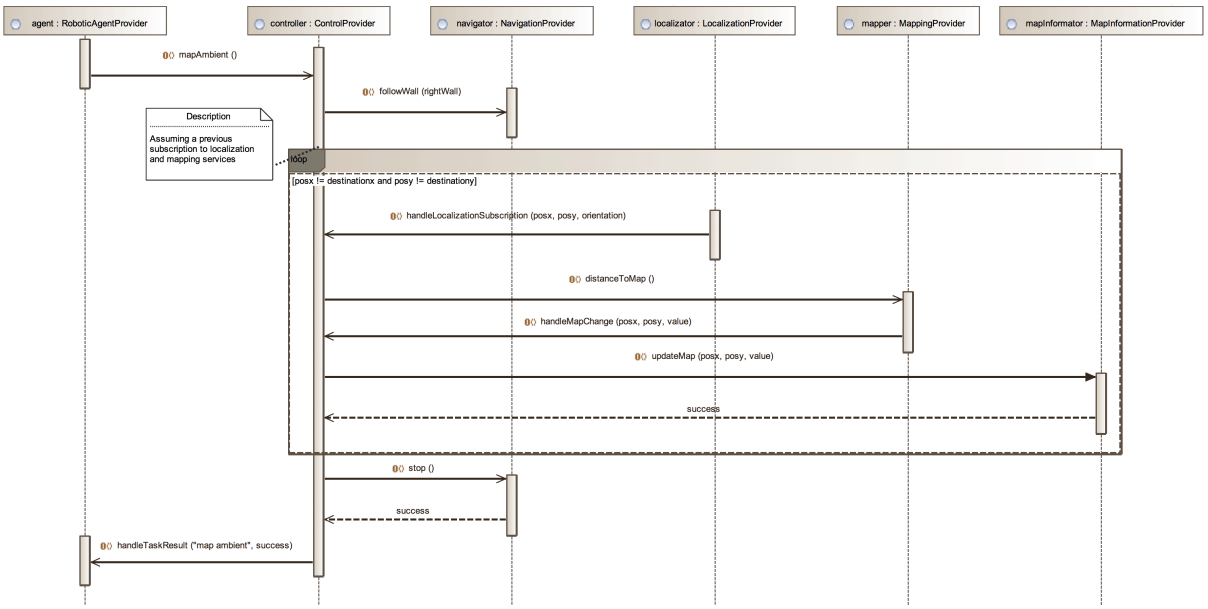

Figure 29: Map Ambient Sequence Diagram

Robotic Agent Service

It will represent a Pioneer P3-DX robot with the main function of performing actions to help reach the system goal. Each robot is 450mm length, 490mm width, and 500mm height, it can reach speeds of 1.6 meters per second and carry up to 23kg, while powered by three hot-swappable 9Ah sealed batteries. When requested by the application it will use control to retrieve a path to the product and calculate its distance to it. For each task it requests control it will receive a response of completion when finished. When a task is completed it moves on to the next one, until the main activity requested by the application is completed. Figure 30 is a sequence diagram that represents robotic agent functionalities.

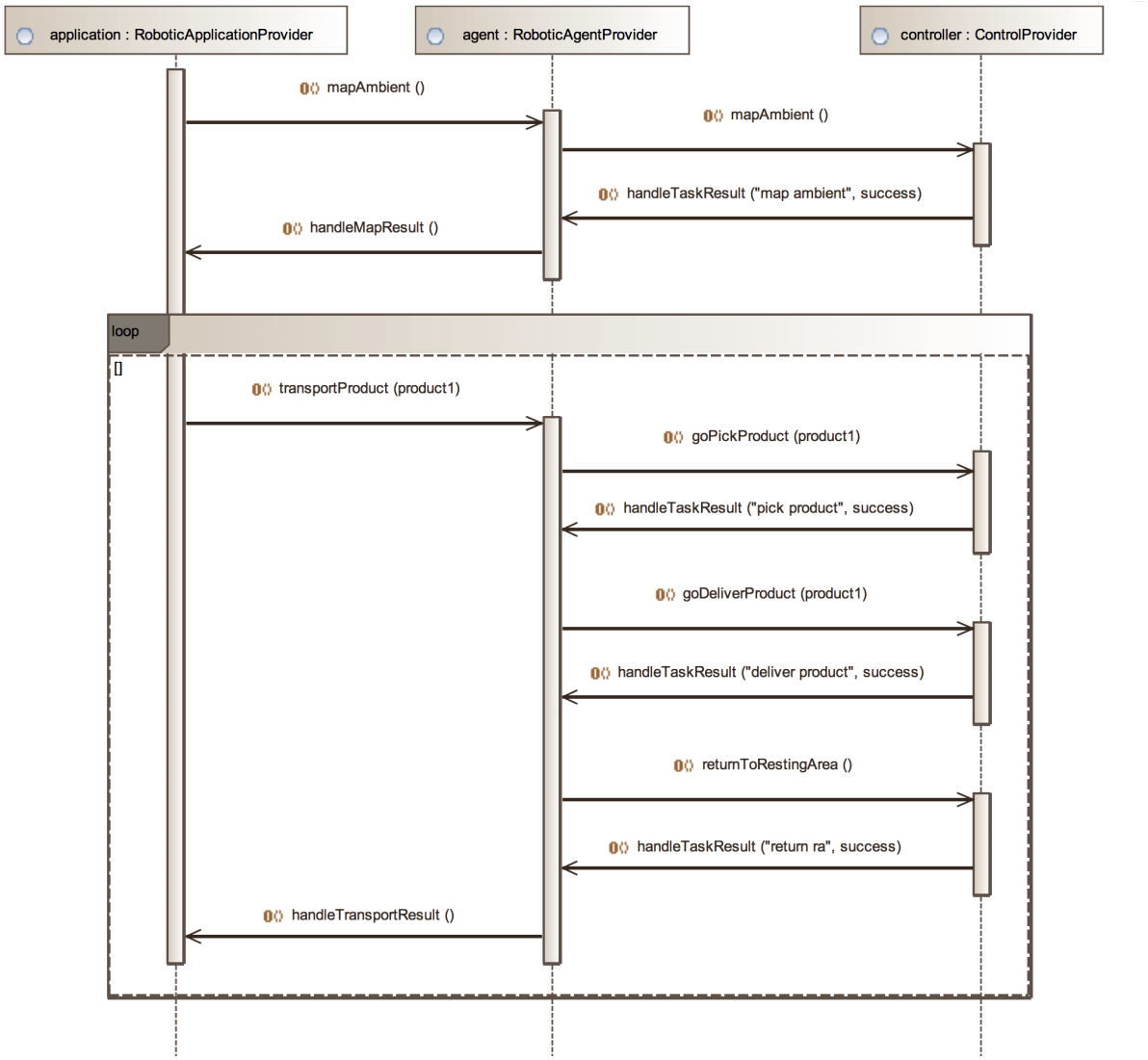

Figure 30: Robotic Agent Sequence Diagram

Robotic Application Service

It will always be aware of where are each robot and their availability to go transport a product. Upon receiving a transport order it will compare each robot path distance to the product and order it to transport that product. When both robots are occupied the transport order will be queue to be later designated to a robot. In case a robot gets disconnected or fails it will continue transporting with only one robot.

4.2 Deployment

Deployment describes where the services are going to physically be operating. Complex or services shared by more than one robot will be operating on a local back-end server. Simple or device drive types of service, that control devices attached to a robot, will be operating from

the robot. Figure 31 presents the Deployment Diagram, explanations for where each service will be implemented are described bellow:

Back-end Server

- Operation service provides a user interface for a operator, since the operator will be using it from the back-end server and lost of connection does not prevent the application from completing its goal it should be there.
- Services like Path Planning and Image Processing uses complex algorithm, some latency its tolerable for what they provide and results will be quicker even with a slow connection.
- Map Information will be shared between robots, its information will be mostly used while composing the map at the beginning of each day. Because it will be used rarely, without the need of periodic update or too much access, it can easily be in the back-end.
- Robotic Application will be controlling both robots, if implemented in one of them communication between robots would also have to be implemented. Since it doesn't have any critical functionality regarding robot control it will be better to implement on the back-end server.

Robot

- Because all Device Driver services must have real-time access to hardware measures and control, it is critically important that they are implemented inside the robot.
- Localization and Object Manipulation services both use one or more Device Driver service directly. They don't require a lot o processing and need to give real-time feedback to other services.
- Robotic Agent, Navigation and Control services all coordinate other services that are inside the robot. They are all critically needed by the robot, if implemented in the back-end server and lost or had a bad connection with a robot, navigation would make the robot keep driving without a stop or with huge delays between actions and control and robotic agent would make the robot stop or delayed when doing tasks and sending responses to the application. In order to avoid these risks they should be implemented in the robot.

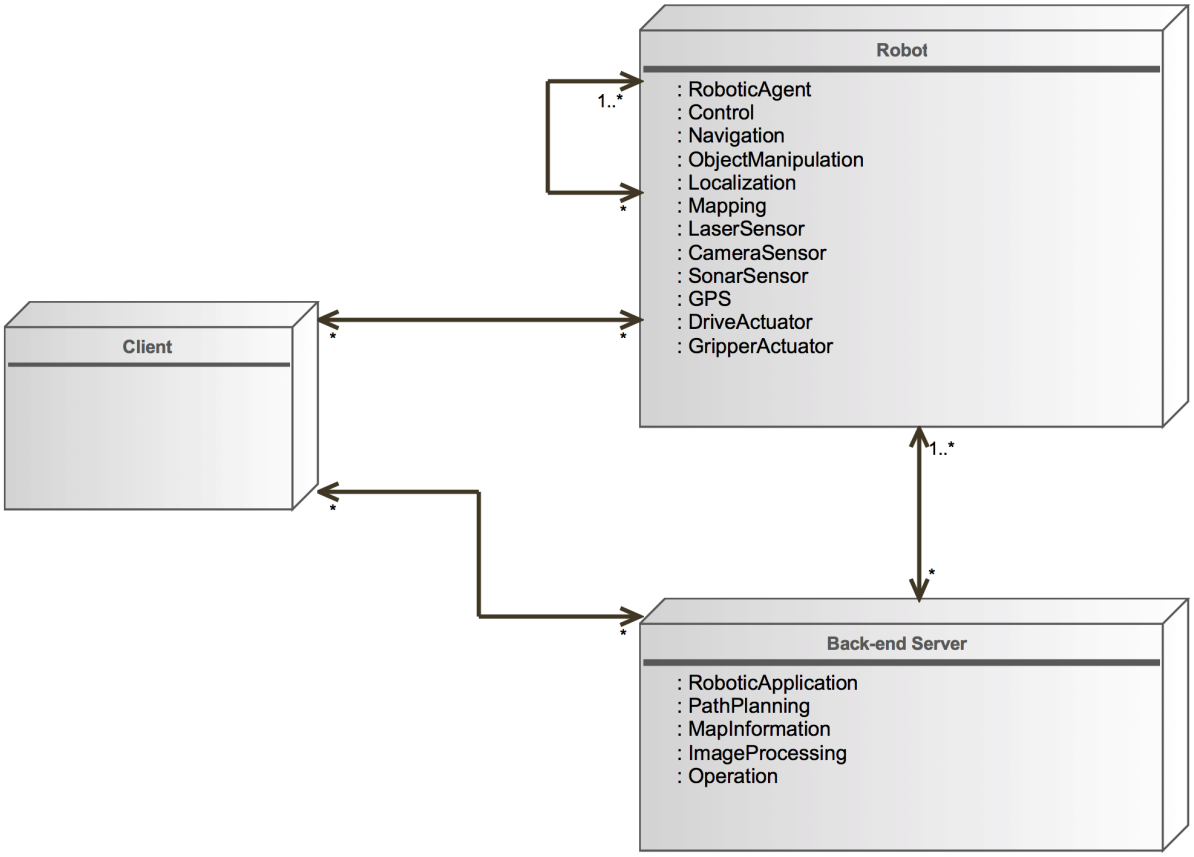

Figure 31: Deployment Diagram