CSCI 568

REQUIREMENTS ENGINEERING

GME MODELLING PROEJCT

BY:

AKSHAYA RAVICHANDRAN

1. **Meta Model:**

My metamodel is named RoboConMetamodel. It has 5 paradigm sheets.

1. RoboCon Paradigm Sheet
2. Route Paradigm Sheet
3. Modes Paradigm Sheet
4. Protocol OnFormation Paradigm Sheet
5. Protocol OnMovementParadigm Sheet
6. Protocol OnRecharge Paradigm Sheet
7. RoboCon Paradigm Sheet: -Convoy Model

**1.1 DESIGN:**

This consists of the metamodel for the top-level Convoy model. It is named

RoboConConvoy. The root folder contains one or more RCU Models and only 1 ACU Model

as per specifications.

The RCU further contains its parts/components.

It includes:

1) iRobot as model because it further contains Motor atom and Wheels Atom.

2) Front bumper sensor as atom (doesn’t further contain any component).

3) Video camera as atom (doesn’t further contain any component).

4) Lan adapter (wifi) as atom (doesn’t further contain any component).

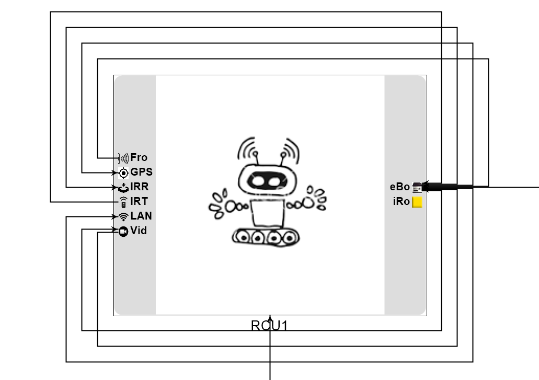
5) GPS Receiver as atom (doesn’t further contain any component).

6) IR Transmitter as atom (doesn’t further contain any component).

7) IR Receiver as atom (doesn’t further contain any component).

8) Ebox3854 Model with Software Controller as field attribute The Ebox contains atoms- FedoraLinux, Processor, Memory, Flash Storage. They are ports of the Ebox.

9) Wheels and motor are also ports of iRobot.

**FIG.1 CONVOY INSTANCE**



The ACU Model has:

1. The type of OS used as field attribute with string data type-runs on windows 7
2. Configuration type as enum attribute- for demonstration or for experiment. Ref fig. 2
3. Admin Software as field attribute with string data type.

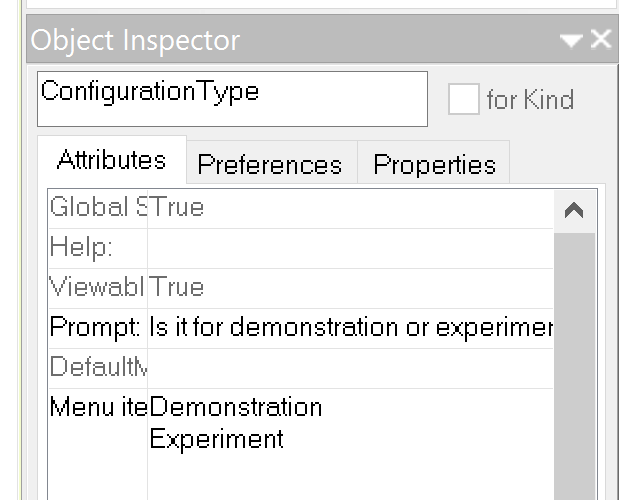
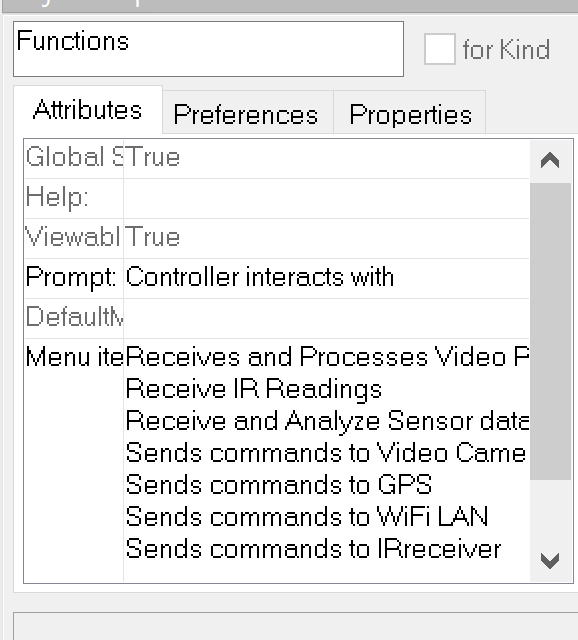


FIG 2.

**1.2. ASSUMPTIONS & DECISIONS:**

1. Although not mentioned in the SRS, I have used RCU Number as a field attribute of the RCU for identification of many different RCUs in the system.
2. I have taken the RCU Software Controller to be a field attribute. The document says the controller runs on ebox and does some functions listed hence it was easy to model it as an attribute of the ebox rather than a separate model or atom. I have used inheritance to connect the RCU Software Controller to sensor, gps, IR receiver and transmitter, LAN adapter and to ACU.
3. I have used enum attribute for the functions that are mentioned in the srs so that when a controller interacts with, say the sensor, then the corresponding function can be chosen from the dropdown. This is more visible in the RoboConConvoy model. This makes it easier to choose the functions/interactions between the controller/ebox and the different components individually for the designer/modeller. Additional functions can also be easily added in ccase need be. Fig3.



**FIG.3.**

**1.3. LIMITATIONS:**

I have not included the detailed internal working of the ACU. I have assumed the ACU to be a computer/admin. Details regarding the GUI and admin functionalities has not been spoken about in the document and couldn’t be implemented.

1. Route Paradigm Sheet:

**2.1 DESIGN:**

2.1.1) This sheet contains Route as top-level model. I have represented Waypoints, Obstacles and Charging stations as atoms. There must be 1 or more of each of them for it to be a route as shown by the cardinalities. Each of them have attributes to give them some further definition.

2.1.2) I have also included a model proxy for the RCU from the RoboCon paradigm sheet. This way an RCU is easier to represent and use in the route model.

**2.2. ASSUMPTIONS & DECISIONS:**

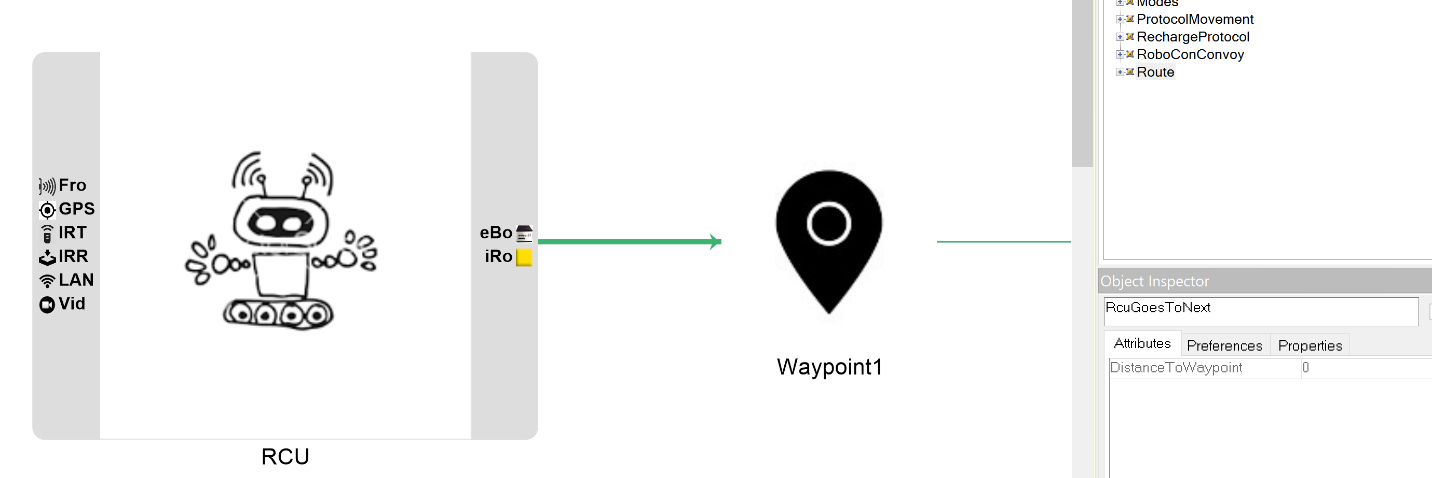
2.2.1) Waypoint includes as attributes x-coordinate and y-coordinate (for distance) and waypoint number to know which waypoint the rcu is at currently.

2.2.2) Obstacle also includes x and y coordinates similarly and obstacle number.

2.2.3) Charging station too has station number, and coordinate points to indicate its location to an rcu.

2.2.4) The avoidance algorithm is modelled as a proxy which is present in the Modes Paradigm Sheet. I have used attributes for connections to describe the property of the connection. Halt Movement, Continue Path and Stop To Recharge are Boolean attributes since they can be toggled between true and false depending on the state of the environment and RCU.

2.2.5) When RCU goes to next waypoint, it uses distance to do that. Hence the connection is named RCUGoesToNext and it prompts DistanceToWaypoint. Detailed explanation of the route is found in Route Model section 2 of the document. Fig4.



**FIG.4**

1. Modes Paradigm Sheet:
   1. **DESIGN:**

3.1.1) In this sheet, I have captured the different modes, states that trigger the change from one mode to another, some messages exchanged in the process, and the complete flow when RCUs move from one state to the next. Since I have tried to represent it like a sequence diagram, (all 3 modes combined) components similar to flowchart like the condition box –yes or no was used.

Yes and No are represented using atoms. Also the messages.

3.1.2) I have used Leader RCU and Follower Rcu as proxies separately to show that Leader leads and locates waypoints/obstacles but Follower only tracks.

* 1. **ASSUMPTIONS & DECISIONS:**

3.2.1) Modes model does have some messages involved in the process. Although I have separately shown 3 separate protocol models with the messages, I felt this way it would give users a better picture as the robot system progresses through the different modes. Since only after message has been received the corresponding RCUs move to another Mode I felt some necessary message needed to be included. Only few necessary messages that I felt had to be a part of the modes model has been used.

4. Protocol On Formation Paradigm Sheet:

There are 3 separate protocol sheets – one for each-Formation, Movement and Recharge.

**4.1. DESIGN:**

In this sheet, I have captured the messages, format and sequence of messages when the system is in formation. The ACU, Leader RCU, Follower RCU and Sensor are proxies to the ACU, RCU and FrontBumpSensor respectively of the first Convoy sheet. I have used atom to represent messages passed between the different components and included MessageType as field attribute of the message atom. The connection Assign Role on Startup consists of Role as enum attribute for better design.

**4.2. ASSUMPTIONS & DECISIONS:**

4.2.1) Connections are labelled to indicate the flow and sequence of the system in movement.

4.2.2) I have not mentioned any specific type of message. It can be modified to suit any kind of message. Hence the generic attribute type.

4.2.3) The connection Assign Role on Startup consists of Role as enum attribute for better design. This way I can choose the RCU to be a leader or as follower from the drop down option.

5. Protocol On Movement Paradigm Sheet:

**5.1. DESIGN:**

In this I have captured the messages, format and sequence of messages when the system is in movement. LeaderRCU and Follower RCU are proxies to RCU Model. Messages exchanged are represented as atoms with MessageType as attribute to indicate the type of message. I have not mentioned any specific type of message. It can be modified to suit any kind of message. Hence the generic attribute type.

**5.2. ASSUMPTIONS & DECISIONS:**

5.2.1) Connections are labelled to indicate the flow and sequence of the system in movement.

5.2.2) I have made use of 2 RCU proxies, one for Leader RCU and another for Follower RCU separately. This is because, I have assumed that when the RCU is in movement mode, the GPS sends Leader RCU signals to identify next waypoint. Along the way if obstacle is present, a message is sent to invoke the Avoidance Algorithm and then the Leader RCU continues to identify the next waypoint and move ahead. The above work is done only by the leader RCU.

5.2.3) At the same time the follower RCU initially uses its camera to track the leader. Since it simply tracks the leader I have shown it to only track. It doesn’t identify any waypoint or obstacle like the leader.

6. Protocol On Recharge Paradigm Sheet:

**5.1. DESIGN:**

In this I have captured the messages, format and sequence of messages when the system is in recharge. RCU is proxy to RCU Model. Messages exchanged like HaltMsg, and ChangeToMovement are represented as atoms with MessageType as attribute to indicate the type of message. I have not mentioned any specific type of message. It can be modified to suit any kind of message. Hence the generic attribute type.

**6.2. ASSUMPTIONS & DECISIONS:**

6.2.1) The wifi Broadcast connection has coordinate points as attribute to indicate that RCU identifies charging station by the wifi coordinates it receives. If distance is less than 10 meters then the RCU proceeds to check their battery levels.

1. **Model:**

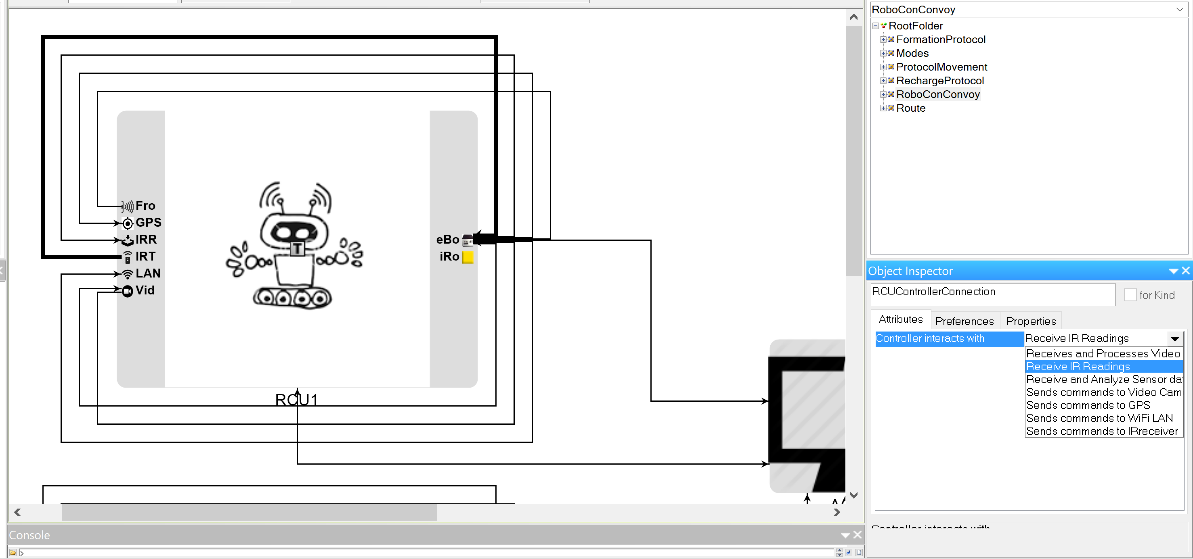
My model has the following:

1. RoboConConvoy
2. Route
3. Modes
4. Formation Protocol
5. Movement Protocol
6. Recharge Protocol
7. RoboConConvoy Model:
   1. **FLOW**:
      1. The RoboConConvoy model has 4 RCUs labelled numerically. There is only 1 ACU. Each of the RCU is an instance of the other. Each of the RCU interacts with the ACU also.
      2. The connections represent exchange of data and information between the ebox (RCU software Controller) and components such as sensor, IR, GPS, Wifi and ACU.

**1.2. ASSUMPTIONS AND DECISIONS:**

1.2.1) The data that can be exchanged between RCU and ACU is represented as a drop down list in the object inspector. Since many kinds of interactions happen between the RCU and the ACU I have shown it this way. Fig.5.

1.2.2) I have used 2-sided arrow to show 2-way interaction between ebox and ACU. The ACU primarily uploads the configuration (that is it assigns roles) to the RCUs through ebox.



**FIG.5**

1. Route Model:
   1. **FLOW**:

2.1.1) In my model, along the route, RCU goes to the next waypoint using the distance it has calculated. When it encounters an obstacle along the way, halt movement attribute of the connection is set to true and the Obstacle Avoidance Algorithm is invoked. Fig6.

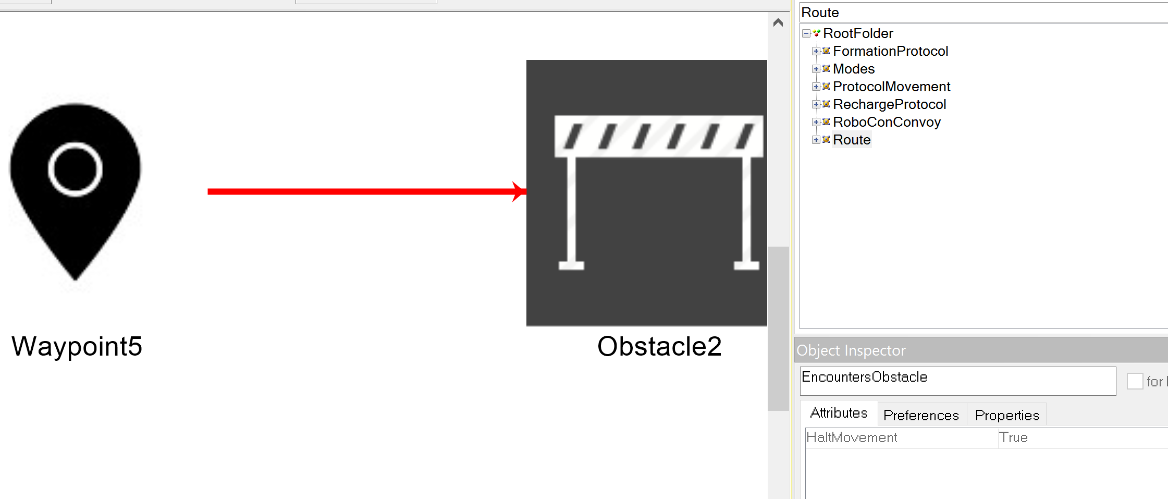


Fig. 6

The algorithm helps navigate around the obstacle and the RCU continues along the next waypoint. Again when it encounters charging station, stop to recharge attribute of the connection can be set to true or false based on the need of the RCU in the system. If true, it stops and recharges and then when done continues along the path to the next waypoint.

2.1.2) I have represented the flow using 7 waypoints (they are numbered sequentially), 2 obstacles and 1 charging station. Stations, waypoints and obstacles can be added along the way.

**2.2 ASSUMPTIONS & DECISIONS:**

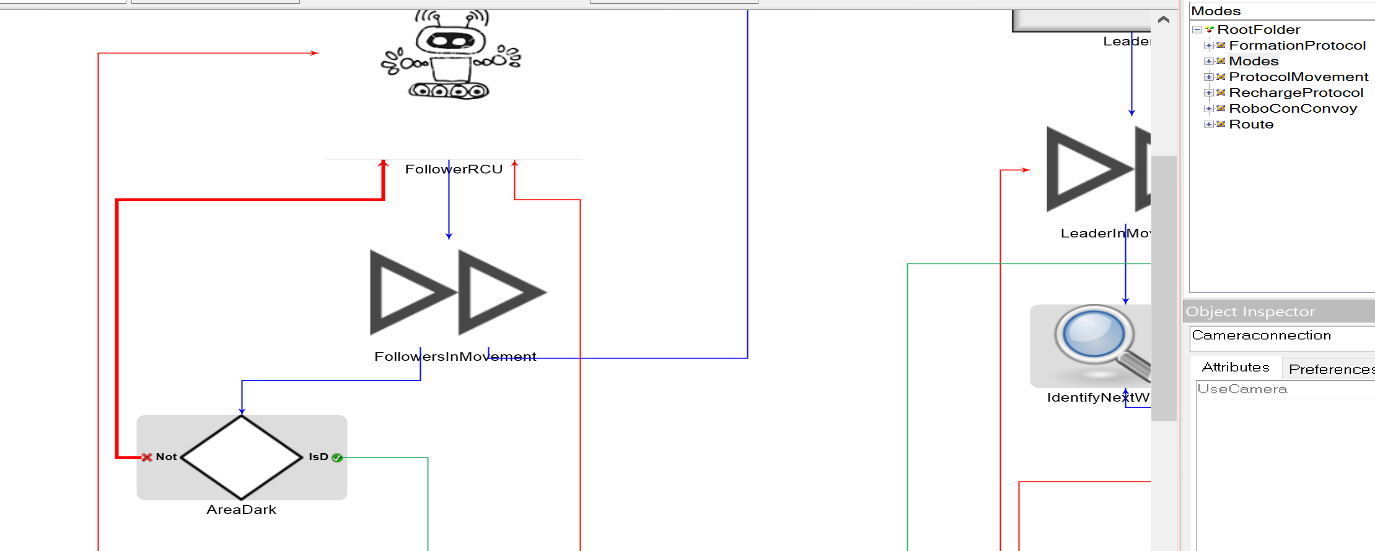
2.2.1) At the start, between RCU and waypoint and between one waypoint to the next, distance is used as attribute.

2.2.2) Green color represents smooth flow, Red color represents temporary halt due to the presence of obstacle or charging station

1. Modes Model:

* 1. **FLOW**:

1. When the system is switched ON, the ACU (represented as port for better design) uploads configurations and directs the RCUs to identify their roles. IdentifyConfigAndAssignRole state.
2. They then proceed to IdentifyGuide state.
3. Guide is found (yes) Or Guide not found.
4. So the Follower rcu will form a single file behind their corresponding guide (Formation\_BehindGuide state) and send ReadyTOSwitch Message to the leader.
5. If guide hasn’t been found (no), then that RCU must be the leader. (leader IS-A RCU) It then broadcasts SwitchTOMovement message to its followers.
6. Leader switches to movement and Followers Switch to Movement.
7. Follower tracks Leader. While In movement, if it encounters a dark area, it checks to see if it is indoors. If Area not dark, it continues to use camera for tracking. And if further area is dark and it is indoors, yes, it checks to see if the distance is small. If not, it uses GPS. If area is dark and it is indoors and distance is small, yes, it uses IR for tracking leader. If no, the rcu halts and waits for further instructions. Fig.7.



**Fig.7. Highlighted red line- CameraConnection and UseCamera**

1. Leader proceeds to IdentifyNextWaypoint state.
2. Along the path, it may encounter obstacle.
3. If yes, it invokes ObstacleAvoidanceAlgorithm.
4. After it navigates around the obstacle, the rcu moves on to ComputePathAndMoveAhead state. And then again continues to identify next waypoint.
5. While computing path, charging station may occur within a distance of less than 10m.
6. If yes, battery is checked for less than 20% or not. If not, rcu continues to be in movement.
7. If yes, it stops and recharges.
8. When done, Rcu resumes movement state.
   1. **ASSUMPTIONS & DECISIONS:**

3.2.1) If no guide is found, then the RCU assumes leader role.

3.2.2) In case area is dark and it is indoors and the distance between rcu is large, RCU halts and awaits further instructions or suitable environmental conditions.

3.2.3) Blue color –unconditional flow.

3.2.4) Red- in case of No condition.

3.2.5) Green- In case of yes conditions.

3.2.6) Yellow-Message send

1. Formation Protocol Model:

**4.1. FLOW**:

1. ACU assigns role on system startup to the RCUs.

2. Sensor sends sensor signal to the follower rcu to detect its guide.

3. After detecting guide, follower rcu sends ReadyToSwitch message to the Leader.

4. The leaded then sends back SwitchTOMovement message to the followers.

**4.2. ASSUMPTIONS & DECISIONS:**

4.2.1) 1 Leader and 1Follower RCU is modelled.

4.2.2) Green color-step1

Blue color-step 2

Orange color-step 3

Violet color-step 4

4.2.3) Message consists of no specific message format/type. It has a message type attribute which can be used to indicate any format as desired by the developer. NO restriction on the type of message sent for flexibility. (Whether tcp/udp/binary..etc.).

1. Movement Protocol Model:

**5.1.FLOW**:

1. System is in movement mode.

2. GPS sends GPS signals to IdentifyWaypoint.

Follower RCU uses camera to track leader.

2.1 IfOnTracking, it encounters AreaDArk\_and\_outdoors, it sends message SwitchToGps

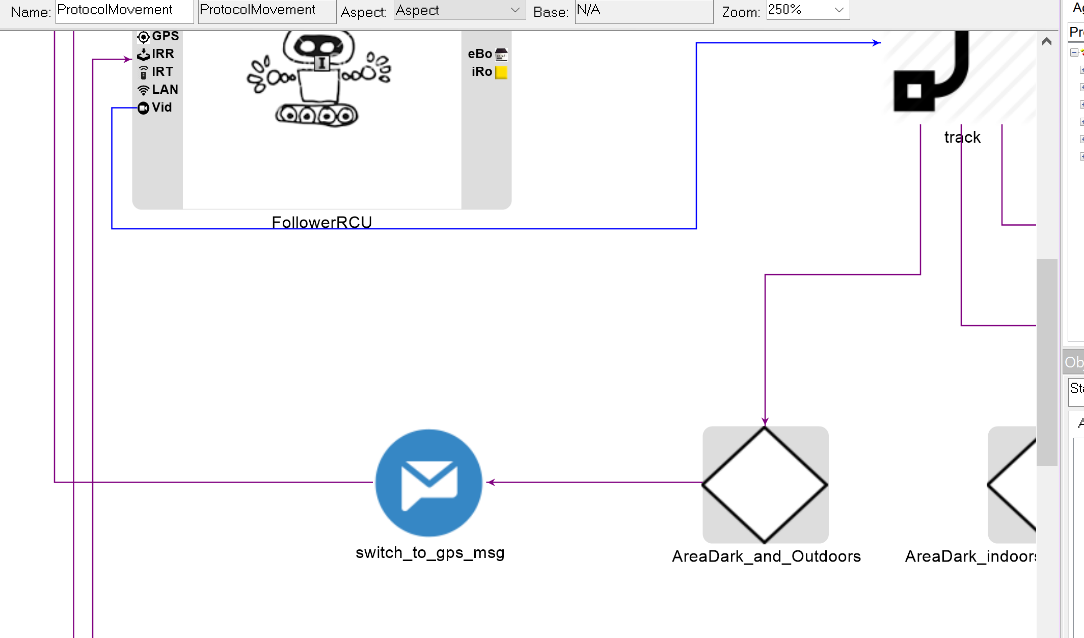
to switch to GPS. Fig.8

2.2 IfOnTracking, it encounters AreaDark\_and\_ indoors\_and\_distanceSmall, it sends

message SwiitchToIRto switch to IR.

2.3 IfOnTracking, it encounters BadEnvironemntal condition, it sends message Halt to

halt and await further instructions.



**FIG.8**

3. If obstacle is found, InvokeMessage is sent to invoke Avoidance Algorithm.

4. Algorithm navigates around obstacle.

5. Then RCU proceeds to IdentifyNextWaypoint.

**5.2. ASSUMPTIONS & DECISIONS:**

5.2.1) 1 Leader and 1Follower RCU is modelled.

5.2.2) Blue color-step1

Orange color-step 2

Green color-step 3

Violet color-step 4

5.2.3) Message consists of no specific message format/type. It has a message type attribute which can be used to indicate any format as desired by the developer. NO restriction on the type of message sent for flexibility. (Whether tcp/udp/binary..etc.).

1. Recharge Protocol Model:
   1. **FLOW:**
2. Over WifiBroadcast, RCU identifies charging station.
3. RCU checks if distance is less than 10m
4. And then ifBatteryLessThan20Percent. If yes,
5. Halt message is sent to other RCUs.
6. RCU that needs charging goes to recharge itself.
7. When done, RCU messages changeTOMovement message to all other RCUs and they change to movement.

* 1. **ASSUMPTIONS & DECISIONS:**

6.2.1) Green color-step1

Orange color-step 2 and 3

Blue color-step 4

Red color-step 5

Violet color- step 6

6.2.2) Message consists of no specific message format/type. It has a message type

attribute which can be used to indicate any format as desired by the developer. NO

restriction on the type of message sent for flexibility. (Whether tcp/udp/binary..etc.).

6.2.3) One RCU that needs charging is shown to send message to other RCUs.