

VC-RAN

1 Introduction

The VC-RAN (Visual Components Robot Agent Node) is the OPIL IoT Nodes layer module devoted to the control of the virtual robots. VC-RAN provides two main functionalities:

- it manages virtual robot navigation inside Visual Components simulation software
- it works as an interface between the virtual robots and the FIWARE Orion Context Broker (FOCB) of the OPIL Cyber Physical Middleware layer.

2 Components and structure

Figure 1 presents the structure of VC-RAN. The VC-RAN is composed of four simulation components:

- **AGV** Automated Guided Vehicle used in the simulation and controlled by OPIL. This component handles the movement tasks of the AGV. Note that this AGV is required if OPIL manages the AGV, meaning that simulated AGVs cannot be controlled by OPIL.
- **RanLogic** This component handles the messages between the RAN and the AGV and the AgvAction components. Each AGV and AgvAction in the simulation requires one RanLogic component.
- **AgvAction** This component handles the action tasks of the AGV. Currently, three actions are supported (0,1,2). Action "1" is used for loading a part and "2" is for unloading it. Action "0" is a dummy action meant for tasks that do not include actions.
- **RAN** handles the communication between the FIWARE Orion Context Broker (FOCB) and RanLogic component. While for each AGV in the simulation layout one *AgvAction* and one *RanLogic* component are required, only one *RAN* is required in the simulation to handle different AGVs.

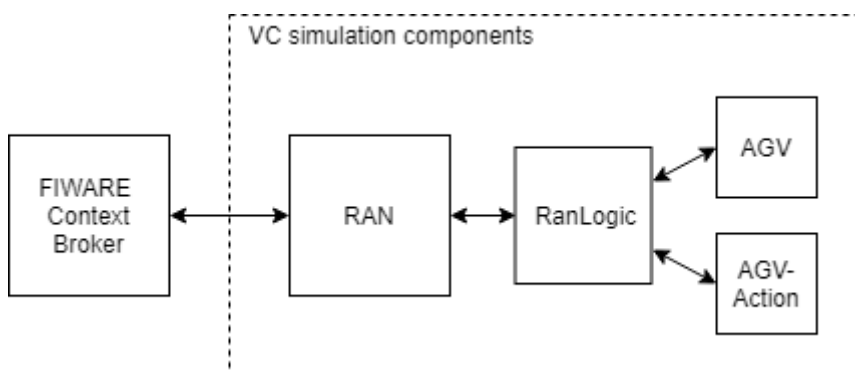


Figure 1. Structure of VC-RAN.

3 Description of the Components in the Simulation

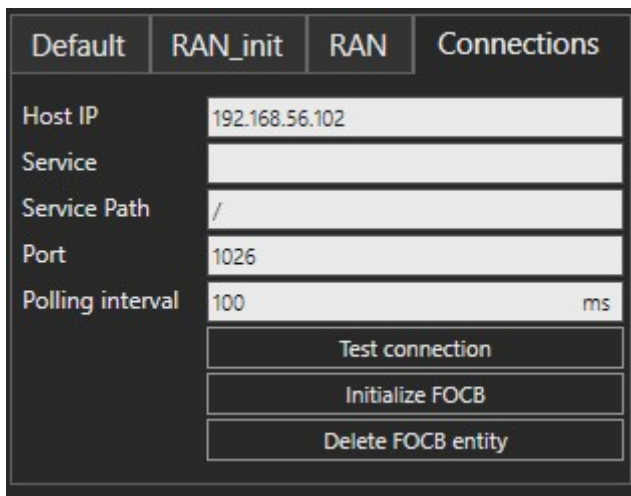
As mentioned previously, for each *AGV*, there is one *RanLogic* and one *AgvAction* connected through the *RanLogic* to the *RAN*. One *RAN* can support several *AGVs*.

This organization of the VC-RAN has been created to maintain and update the components in the simulation faster, but it is expected that in the future versions the design will be simplified.

3.1 RAN

The *FOCB* messages are received and delivered by the *RAN* into the simulation. The *RAN* component handles messages, which are transmitted using signals and interfaces in the simulation layout, to the *AGVs* connected to the *FOCB* through the *RAN*.

Figure 2 shows the connections tab of the *RAN*. *Host IP* is the IP address of the *FOCB*. Once the IP has been set up, test the connection and the create the entity by pressing "*Initialize FOCB*". When the component is removed, the entity can be deleted by pressing "*Delete FOCB entity*"



Default	RAN_init	RAN	Connections
Host IP	192.168.56.102		
Service			
Service Path	/		
Port	1026		
Polling interval	100		ms
Test connection			
Initialize FOCB			
Delete FOCB entity			

Figure 2. Screenshot of the RAN configuration properties tab.

3.2 RanLogic

The *RanLogic* component filters the messages according to the robot id of the *AGV*. *RanLogic* stores the data (motion and action assignments) to data structures based on the task id and receive time and handles the data delivery to the *AGV* (motion assignments) and *AgvAction* (action assignments).

Robot id for the *RanLogic* can be set in the *RanLogic* properties tab (see Figure 3)

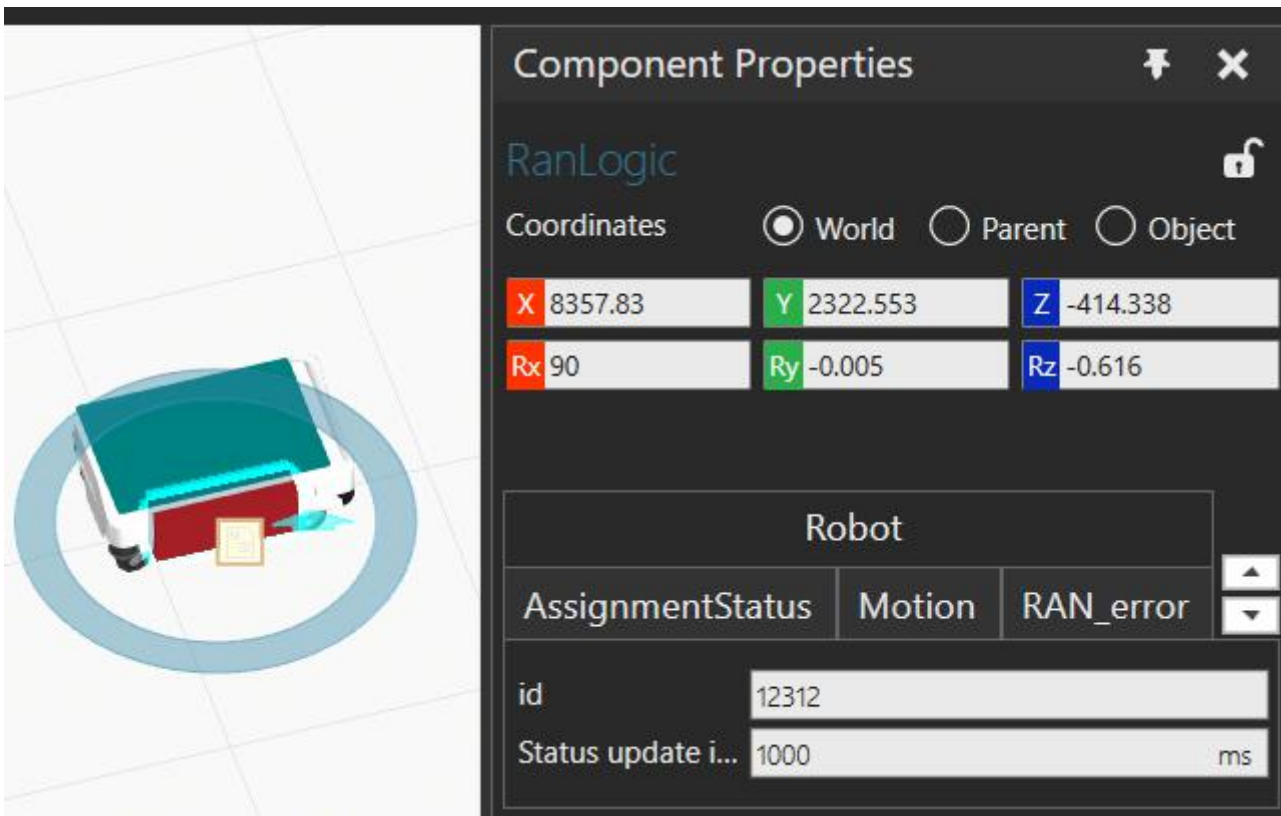


Figure 3. Screenshot of the RanLogic configuration properties tab, AGV ID.

RanLogic also maintains data structures for completed and cancelled tasks as well as for task variables such as current and last motion and action.

RanLogic sends the following messages:

- "Description message" every minute to *RAN*, which delivers the message as is to FOCB.
- "RobotState message" is sent by the *AGV* every second to *RanLogic* that then sends the "RAN-state message" to *RAN*.
- "Assignment status message" is sent to the *RAN* through the *RanLogic* every second.

RAN delivers both RAN-state and assignment status messages to FCB. *AGV* and *AgvAction* components send a message after every completed assignment to *RanLogic*, which the *RanLogic* uses to update its data structures, task variables and state.

3.2.1 Task handling (AgvAction and AGV)

Tasks are executed in the order they are received. When a motion or action assignment with a particular task id is first received, it is inserted into a data structure together with timestamp. Collectively, they form the task queue. Timestamps for the last motion assignment and action assignment are also stored in the same data structure.

Received tasks are executed in the order of ascending timestamp value. Robot motion planning and execution begins once all the motion assignments with the same current task's id are received. This ensures that all the required messages have been received. All the motion assignments belonging to the same current task are sent at the same time to the AGV. After all the motion assignments are completed, action assignments belonging to the current task are sent to the VC-AgvAction component one at a time. The task is completed when all the action assignments have been carried out.

If all the assignments (motion or action) belonging to the same task are not received within specified timeout limit, the task is cancelled and removed from task queue. Likewise, if cancel task command is received, task is cancelled in the same manner. If a task is cancelled mid-execution, the current motion or action assignment execution is continued, but the following assignments are cancelled.

3.2.2 State machine

RanLogic is always in one of the finite states specified in Figure 4. Starting state is state-waiting, which changes to state-task after new task is received. When all the motion assignments belonging to the current task have been received, the state changes to state-motion and VC-RanLogic will send the motion assignment sequence to the AGV. After all the motion assignments have been completed and the action assignments belonging to current task received, state changes to state-action. In this state, VC-RanLogic sends action assignments one by one to the VC-AgvAction component until all the assignments are completed and VC-RanLogic returns to state-waiting. If the current task is cancelled, the state changes to state-waiting.

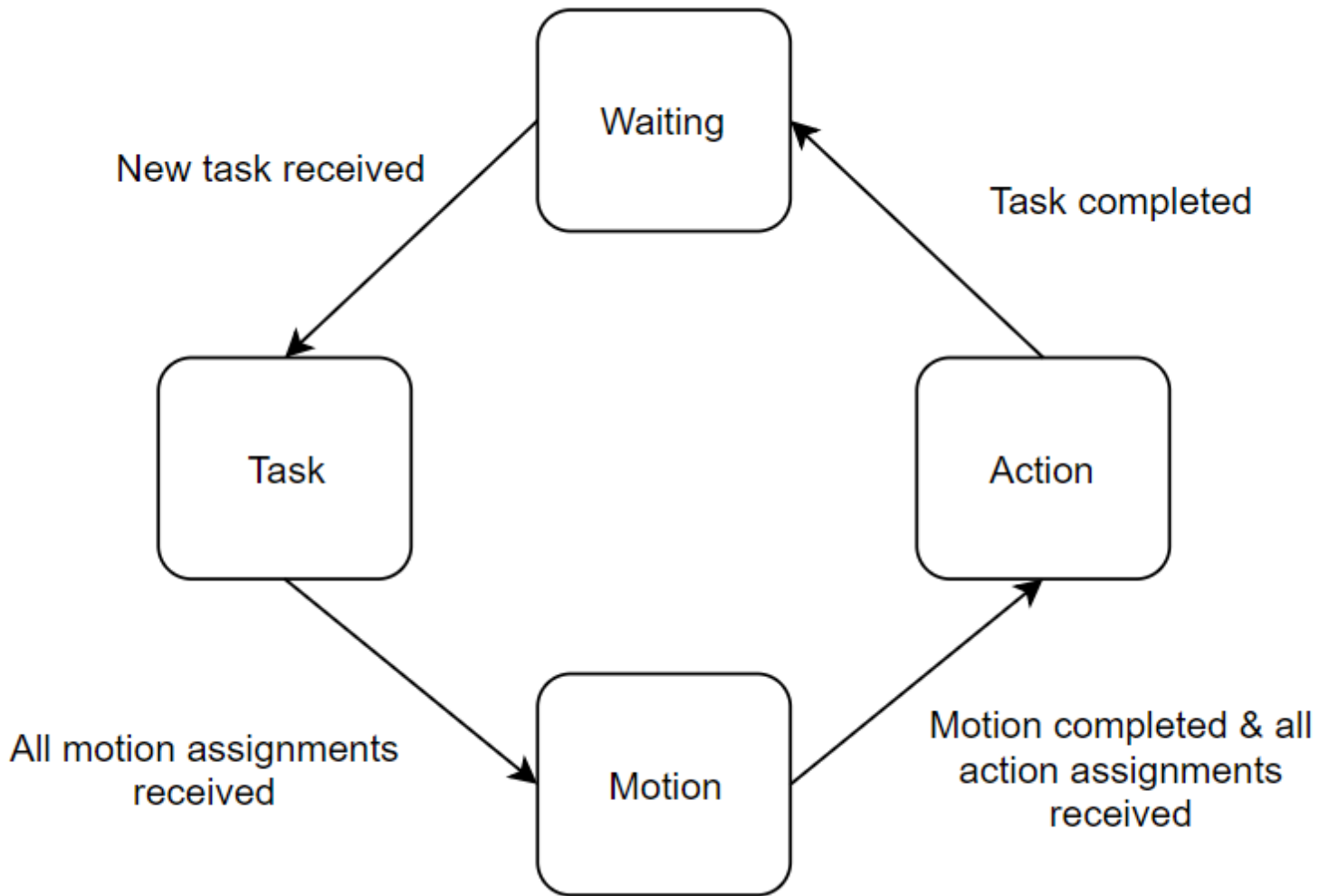


Figure 4. RanLogic state machine.

3.3 AgvAction

This component handles the action tasks of the AGV. Currently, the following actions are supported:

- 0 Dummy action used for task that does not include actions
- 1 Loading parts action
- 2 Unloading parts action

3.4 AGV

The identification of the AGV is unique and is determined by an ID. This ID can be set in the AGV properties tab (See Figure 5). The ID of the AGV should be the same as the *RanLogic* to ensure that the movement messages are delivered to the AGV. (Figure 5)

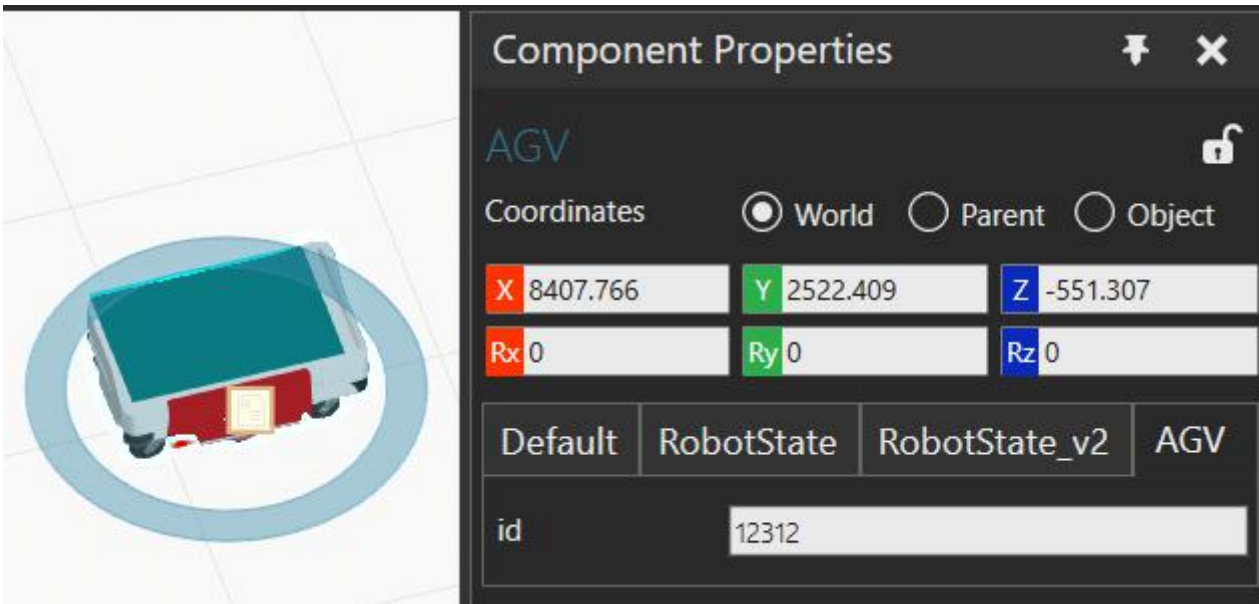


Figure 5. Screenshot of the AGV configuration properties tab, AGV ID.

Currently, in the VC-RAN demo an AGV component called "*AGV interpolator*" is used. This AGV interpolates positions between waypoints transmitted through *FOCB*.

3.5 VC-RAN components deployment

In order to facilitate the deployment of the simulation of VC-RAN related components, two components are provided: *RAN* and *AGV Interpolator*, which contain the *AGV*, the *AgvAction* and the *RanLogic*. When using the *AGV Interpolator* it is required to check during the configuration that the *AGV* and the *RanLogic* have the same robot id.

4 Setup

Adding VC-RAN components to the simulation follows the same logic as creating a simulation in Visual Components. Components available in the *eCat* can be picked and placed in the simulation.

As previously indicated, in order to ease the creation of the simulation, *AGV*, *AgvAction* and *RanLogic* have been deployed as a single component, *AGV Interpolator*.

Pick and place the *RAN* component in the simulation and place the required "AGV Intepolator" components.

Under the connections tab in the *RAN*, as shown in Figure 2, set the Host IP to point to the IP of the *FOCB*. Connect the *RAN* component to every *RanLogic* component interface by joining *RanLogic* and *RAN* interfaces as shown in Figure 6.

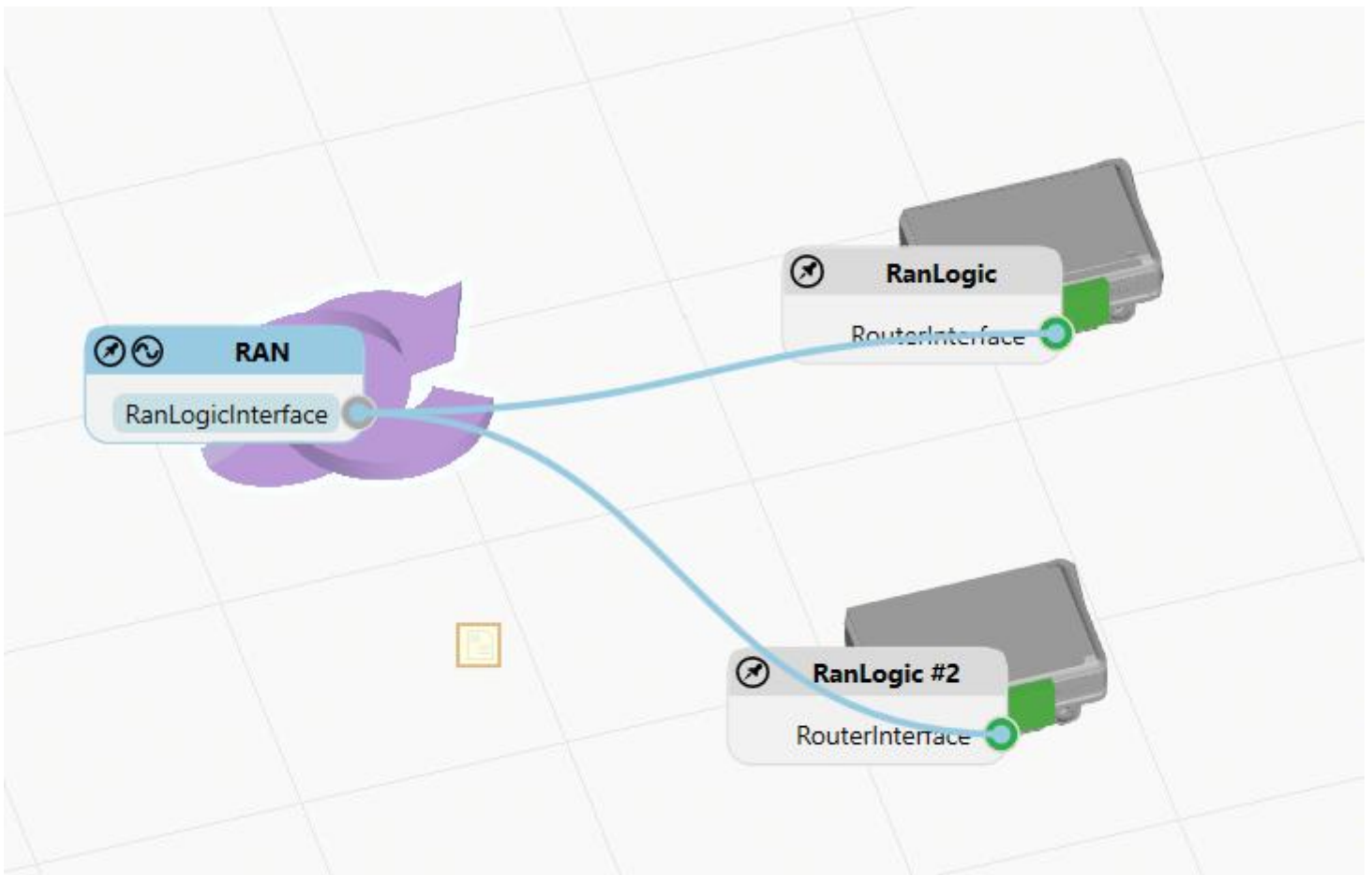


Figure 6. Screenshot of the connection of the *RAN* with several AGVs through the *RanLogic*.

Don't forget to check that the robot id in the *RanLogic* is the same as that in the AGV and the new configuration is ready to be tested.

5 VC-RAN Demo Layout

A demo Layout is provided for testing. The demo layout (RobotAgentNode_Demo) contains a basic production layout. In the simulation, one part is produced in one side of the layout. An AGV controlled through OPIL is called when the part produced is ready and waypoints are provided to move the part from one side to another where the simulation continues.

In the demo layout, there are three tasks which are executed in sequence. First task includes two motion assignments and one action assignment: the AGV moves next to a conveyor through two points and loads a part. Then the AGV moves through two points next to the other conveyor and unloads the part on the conveyor. Finally, the AGV returns to its initial location through three points. This is followed by an empty dummy action to signal the end of the task. The collection of task assignments for the demo is provided as part of a JMeter file.

6 Potential future improvements

The work in the VC-RAN continues as the entities are updated in the documentation. Features currently missing include:

- Cancelling individual assignments
- Subscribing to FOCB entity updates (at this moment is polling)
- Assignment queue from AssignmentStatus message
- Error messages not implemented (not defined)