IoT Devices and Robot Communication Platform by ROS2

Rawin Chaisittiporn

Faculty of Science
Chandrakasem Rajabhat University
Bangkok, Thailand
rawin.ch@chandra.ac.th

Abstract—This paper presents how to use ROS2, open source middleware for robot to be a platform for IoT devices and robots communication. ROS2 important features are selected to be a model for these jobs. There are QoS communication, life cycle management, behavior tree coding style. By using these features the platform can be applied effectively, connected seamlessly and cheap. Case study are set up by raspberry pi 4 with sensors, ROS2 foxy and python code. The robot is TIAgo gazebo simulation with python code. The results show that ROS2 can support message passing between IoT devices sensors and robot. Moreover, it can adapt to network traffic quality by its QoS policies. Behavior tree help robot design to dynamically adapt to uncertain environment. Finally, life cycle manageable node makes robust functions for IoT device nodes. Summarily, ROS2 is an interesting alternative choice for co-cooperative IoT devices and robot ecosystem, on open source based software.

Keywords—ROS2, QoS, IoT, Behavior Tree, Robotics

I. INTRODUCTION

Robot Operating System 2, ROS2 is a latest version of open source robot middleware. Recently, ROS has been a popular version of robot middleware. Developers all around the world use it to build robot freely. It has been adopted by many organizations and industries around the world, because of its plenty of libraries and tools. Especially, it has no license fee and it is open source software.

Now, ROS2 project has been developed to become separated software packages, not an extended version from ROS. Actually, ROS2 is a middleware mainly for robot development. It has several libraries and tools for robot software development and debugging. Moreover, it has many new features to standardize robot software engineering, such as, DDS middleware, life cycle management, behavior tree.

Recently, there are so many IoT platforms and still has no standard. Most of IoT devices relate to sensors, micro controller, network module. They are various IoT devices configurations, such as, brands, prices, performances.

IoT devices can communicate with robots in some projects. Unfortunately, they can not communicate seamlessly with each other. Network protocol and message passing are the main problems. There are less IoT platforms that can communicate with robot, and some platforms have high cost.

This paper presents ROS2 platform for robot and IoT devices integration. It presents advantages in using ROS2 with message passing, quality of service (QoS) of message communication, life cycle management and robot behavior tree. We design new solution to set up ecosystem for robots working with IoT jobs. So, it can vastly extends abilities of IoT and robot projects. Especially, this solution is cost effective for modern industrial and educational purpose.

II. RELATED WORKS

There are much research about IoT field and it covers many areas of interesting, but there are not much about IoT and robots.

Pilavan Kongtongnok has developed object detection system for the visually impaired person by raspberry pi connect with API cloud and use IoT to alert visually impaired person [1]. Savitree Peongam has developed smart farm rental system by using Internet of Things technology. It help reduce production cost and increase farm efficiency [2].

L. Dauphin, E. Baccelli and C. Adjih explore RIOT-ROS2, ROS2 software running on IoT operating system, RIOT. It shows that it can use in cheap, off-the-shelf hardware element perfectly [3]. T. Farnham et al. have developed UMBRELLA project, IoT testbed currently deployed in UK. Now the testbed is related to warehouse robotics. They use ROS2 and docker container [4].

Y. Park, J. Choi and J. Choi have designed system architecture to control a robot based on the acquisition of sensory data from IoT environment. This architecture includes the sensor data conversion process to provide appropriate control information for robot tasks [5]. M. Kumari, A. Kumar and R. Singhal have studied surveillance robot that can be used in domestic areas and many other places and can communicate with home IoT devices [6].

We have found that there are not much about research or platform about IoT and robots even it is very important in many fields. Robot can do mission that IoT devices or anything can not do, such as, surveying, surveillance, pictures taking, monitoring, etc. Robots can take action responsively with IoT devices.

So this paper presents a simple but work architecture for robot and IoT communication by ROS2 software. That obviously appears that it can manage the IoT and robots system efficiently.

III. EXPERIMENT

A. Architecture design

First, select the features from ROS2 that suitable for this architecture, that can seamlessly co-operate IoT devices with robots. ROS2 important features are DDS, QoS, message passing with publish-subscribe mechanism, C++ compatible, python compatible, service node, action node, life-cycle management, behavior tree style coding, many libraries and tools for robot development.

We design conceptual framework for working of IoT devices and robot that can utilize features from ROS2 suitably. We present hardware architecture in figure 1 and software architecture in figure 2.

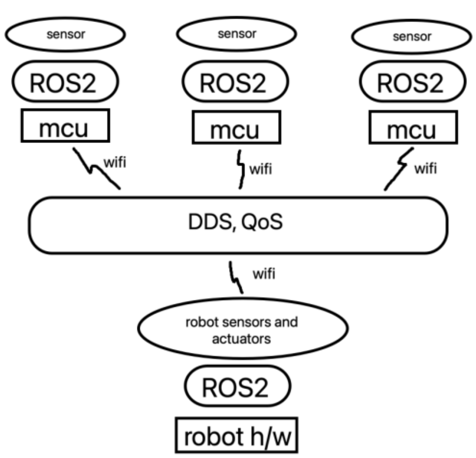

Figure 1. hardware architecture

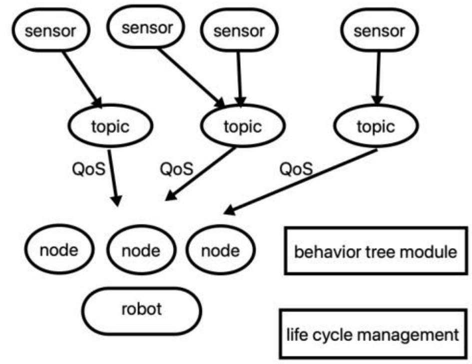

Figure 2. software architecture

From figure 1 and figure 2 show concepts of how IoT devices and robot can communicate in ROS2 platform both hardware and software. They utilize features from ROS2, such as, DDS, QoS, behavior tree and life cycle management and can set up system for IoT devices and robot co-operation. QoS help robots to set policies in their jobs. Robots can operate even in lossy network, or in some jobs they will stop operation if data is not reliable. Figure 3 shows criteria of QoS policies in ROS2.

So developers can set QoS policies for robot depend on the priorities of IoT sensors. If sensors have high priorities for robot, they can set robot topic subscription to reliable. However, if sensors have low priorities, they can set robot topic subscription to best effort.

publisher	subscriber	compatible
best effort	best effort	yes
best effort	reliable	no
reliable	best effort	yes
reliable	reliable	yes

Figure 3. criteria of QoS policies

Behavior tree can be used to design robot tasks, because it is easy for robot management, such as, job designation, job adaptation, maintenance. For life cycle management we can use it with node implementation, so it can make nodes more efficient and resource safety.

B. Case study testing

Next method is to set up case study experiment for this architecture testing. In this experiment we use raspberry pi 4 with temperature and humidity sensors to be IoT devices. ROS2 foxy is installed in raspberry pi 4, installed with ubuntu server 20.04 and coding nodes with python. For robot this test use TIAgo mobile robot, gazebo simulation software, running on Ubuntu 20.04 notebook. The nodes of sensors we set QoS policies to reliable for temperature and humidity and set publish frequency to 1 Hz both of them. Then, set WI-FI for wireless local area network and set DDS domain id to 1, so IoT devices and robot can communicate each other.

For robot nodes that subscribe to temperature and humidity sensors we set QoS policies to reliable and best effort, respectively. So, they are both compatible in communication.

The behavior tree is designed to set goal points of robot navigation. It is determined by value of sensors and battery level simulation of robot to set goal of its charge station when it is in low battery state. Figure 4 describe this paradigm.

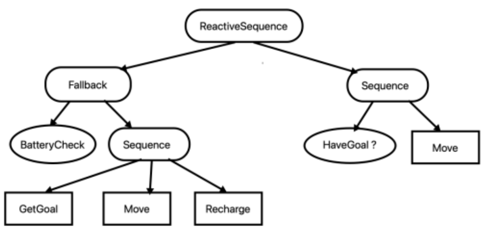

Figure 4. behavior tree of robot

Behavior tree in figure 4 is an efficient method to design how robot work with IoT devices. The robot continually checks for the battery level if it gets low then it will get the goal of recharge station and move to charge the battery. Then it checks the goal to move that sent from sensors if sensors have some problems. If there are goals then the robot can move there. Reactive sequence means it will do these jobs iteratively.

For node design we use life cycle model to make node more manageable. Figure 5 and 6 show how life cycle design can help nodes do their jobs efficiently. They describe how life cycle can be applied with sensors nodes and robot nodes, respectively.

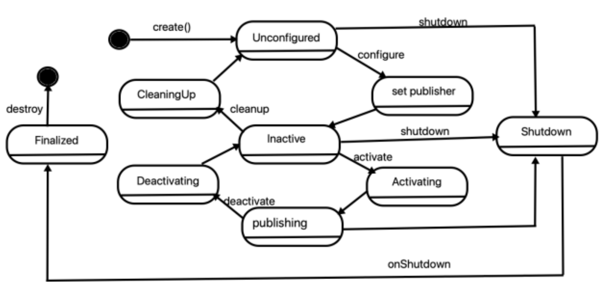

Figure 5. life cycle in sensors nodes

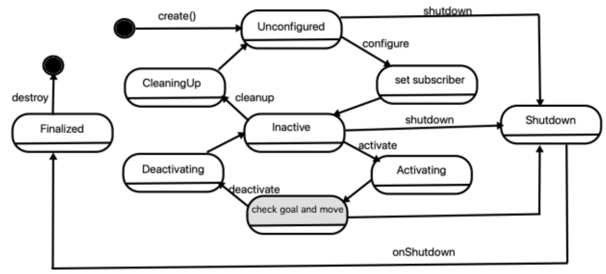

Figure 6. life cycle in robot nodes

For temperature and humidity sensor nodes are programmed to set the goals randomly to make robot move. Additionally, the battery simulation node is set to slowly decrease when robot move. Figure 7 shows these working.

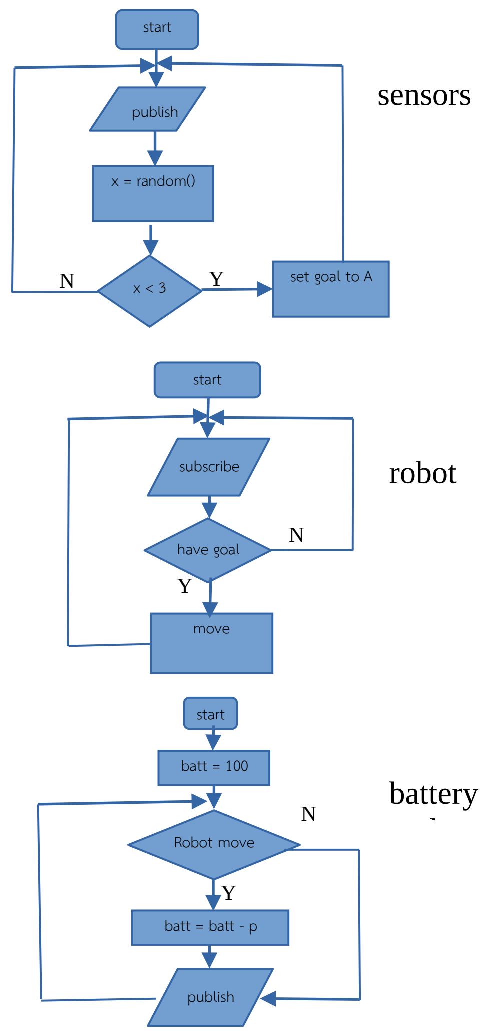

Figure 7. flowchart of sensors, and robot nodes

C. Testing

After design of case study testing, then we implement them and testing its functions. First we start IoT sensors devices and run publisher nodes by their QoS policies. Then we start robot nodes in gazebo simulation, start behavior tree nodes. So it start listening to messages from IoT sensors.

We use ros2topic list to test messages from IoT sensors devices. The messages that we check are /temp, /humidity, /battery_level and /goal. Then we check how robot move to the goal.

IV. RESULT

After setting up all of hardware and software and start all functions. Consequently, IoT sensors devices have sent message by the /temp, /humidity topic continuously. In the /goal topic it shows periodically goal of the sensors and the charge station. Table I show quality of messages in WI-FI ranges testing.

quality of message passing		
topic	***WI-FI ranges***	***quality***
/temp	1 m.	reliable
	5 m.	reliable
	15 m	best effort
	20 m.	can not receive
/humidity	1 m.	reliable
	5 m.	reliable
	15 m	best effort
	20 m.	best effort

Table I quality of message passing in WI-FI ranges

robot coding methods and maintenance		
system	***coding style***	***maintenance***
C++	hard code	hard
python	hard code	hard
ROS node	C++, python node	moderate
ROS2 node	behavior tree	easy

Table II robot coding methods and maintenance

When robot is moved out of WI-FI range then it can not receive /temp message reliably and can receive some message from /humidity, because of its QoS policy.

The robot in the gazebo simulation can move to any goal by ROS2 Navigation library and can move to charging station when /battery is low.

Behavior tree is the new style of robot coding in ROS2. It may be complicate to study but it has more flexible in adaptation of robot jobs. Table II shows this advantage compared with other systems and styles of coding.

V. CONCLUSIONS

It clearly shows that ROS2 can be used as a platform for IoT devices and robots communication seamlessly, efficiently and cheap. It can reduce development time of projects that have both IoT devices and robots working together, because ROS2 can smoothly connect them together. So, developers just install ROS2 both in IoT devices and robots, then they can program data communication by using the same method.

ROS2 has new standards for robot software engineering. Absolutely, it can be used in real industry as a middleware both for robot and IoT devices or even other software categories.

QoS of ROS2 communication create flexible message passing by the publisher-subscriber model. Developers can set topic's QoS policy depend on its importance. So, they can prioritize their robot's jobs based on importance of the sensors. Actually, DDS protocol of ROS2 is the main feature of this function. It can be configured to set data communication by UDP/IP in best effort policy, and set to TCP/IP when use reliable policy.

Life cycle management model of node coding in ROS2 will make robustness for node working. The resources can be managed efficiently and safely. Developers can design their flow of node functioning and make it run cleanly.

Behavior tree is a new coding style that suitable for robot jobs design. Because robot workspace is very flexible, such as, factory, assembly line, warehouse, etc. Developers can adapt their robot jobs by edit XML file of behavior tree. Moreover, this can be done in run-time, not necessary to re-edit code or re-compile. It can vastly help developers in real situation.

Summarily, ROS2, open source robot middleware clearly is a good choice for IoT devices and robots ecosystem. ROS2 provides many features and standards for IoT or robot software development. It can be used in any real industrial inexpensively.

Especially, it is open source software, so there are big community to share problems, solutions, update software, experiences. The most important point is its free from license fee and market monopoly. However, ROS2 has some limitation caused by open source style software. So, it has no technical support and it is not naturally compatible with MS Windows or MacOS.

References

[1] Pilawan Kongtongnok, “Object detection system for the visually impaired person”, 2017, King Mongkut’s University of Technology North Bangkok. Thailand.

[2] Savitree Peongam, “Smart farm rental system using Internet of Things technology”, 2017, King Mongkut’s University of Technology North Bangkok. Thailand.

[3] L. Dauphin, E. Baccelli and C. Adjih, "RIOT-ROS2: Low-Cost Robots in IoT Controlled via Information-Centric Networking," 2018 IFIP/IEEE International Conference on Performance Evaluation and Modeling in Wired and Wireless Networks (PEMWN), 2018, pp. 1-6, doi: 10.23919/PEMWN.2018.8548798.

[4] T. Farnham et al., "UMBRELLA Collaborative Robotics Testbed and IoT Platform," 2021 IEEE 18th Annual Consumer Communications & Networking Conference (CCNC), 2021, pp. 1-7, doi: 10.1109/CCNC49032.2021.9369615.

[5] Y. Park, J. Choi and J. Choi, "A system architecture to control robot through the acquisition of sensory data in IoT environments," 2016 13th International Conference on Ubiquitous Robots and Ambient Intelligence (URAI), 2016, pp. 749-752, doi: 10.1109/URAI.2016.7734106.

[6] M. Kumari, A. Kumar and R. Singhal, "Design and Analysis of IoT-Based Intelligent Robot for Real-Time Monitoring and Control," 2020 International Conference on Power Electronics & IoT Applications in Renewable Energy and its Control (PARC), 2020, pp. 549-552, doi: 10.1109/PARC49193.2020.236673.

[7] F. Martin Rico, A Concise Introduction to Robot Programming with ROS2, Chapman and Hall/CRC, 2022.

[8] ROS2 design, 2022. ROS on DDS. [online] Available at: https://design.ros2.org/articles/ros_on_dds.html.

[9] ROS2 design, 2022. Managed nodes. [online] Available at: https://design.ros2.org/articles/node_lifecycle.html.

[10] ROS.org, 2022. TIAGo Tutorials. [online] Available at: https://wiki.ros.org/Robots/TIAGo/Tutorials.

[11] BehaviorTree.CPP, 2022. About this library. [online] Available at: https://www.behaviortree.dev.