Real-Time ROS for Embedded Systems

Yigit Günay, Dejan Pangercic, Arne Hamann, Rainer Leupers, Gerd Ascheid, Róbert Lajos Bücs
Robert Bosch Deepfield Robotics, Robert Bosch Corporate Research, RWTH Aachen University

Embedded systems are used to interact with sensors and actuators of a robot in hard real-time, and high-level software is used for vision, perception, reasoning, and planning [1] tasks. Hence robots form a distributed system consisting of low-level and high-level components. The ROS middleware allows for abstracting those components in distributed nodes, which simplifies software development process for robotics applications. However, there is currently a gap in robotics software development since ROS is not completely supported for low-cost microcontrollers that are used for interfacing sensors and realizing control tasks.

We have researched several approaches [1,2] for bridging the gap between microcontrollers and general-purpose computers. Some approaches have an embedded ROS client library. In some approaches, there is a proxy node to map serial messages transmitted from an embedded device to ROS messages. A seamless transfer of ROS messages may be preferred or serial communication may be a bottleneck in many applications. Some approaches allow only one node per device, which can make software development more complex. Currently, no existing approach has a programming API that allows running multiple ROS nodes concurrently on an embedded system that can directly communicate with ROS nodes on other devices.

In order to simplify software development for future robotics applications, we have developed an embedded system supporting multiple ROS nodes that are able to communicate with one another using inter-process communication and with other ROS-compatible devices over a network, as shown in the below figure. This system consists of an STM32F4Discovery board, the FreeRTOS operating system, an embedded ROS middleware, and an embedded ROS client library.

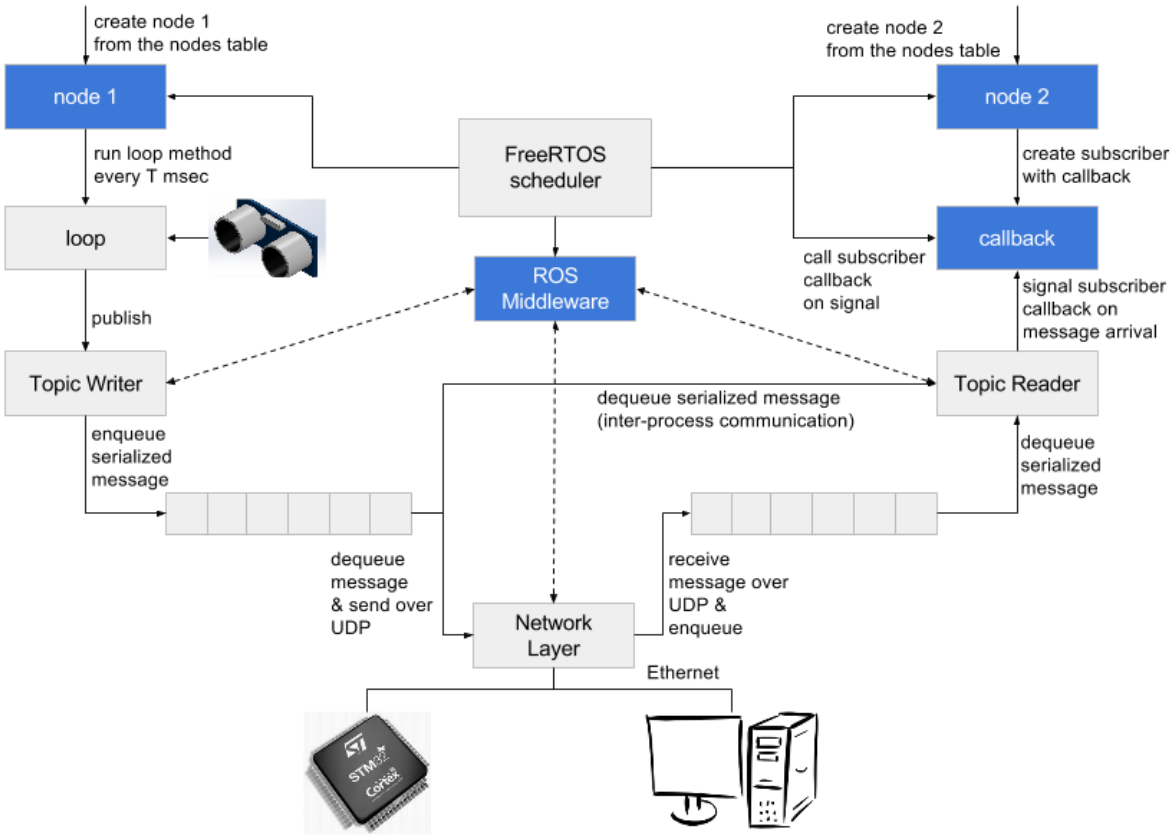

Figure showing real-time ROS software concept. Blue blocks represent FreeRTOS tasks.

The embedded ROS middleware abstracts the underlying transport mechanism of our system so that any ROS communication protocol can be supported. Currently, the middleware can make negotiations with a master node on the network using XMLRPC and send ROS messages using the UDPROS protocol. Our lightweight implementation, however, does not include services, parameter server, or TCPROS at the moment. In the future, we plan to support ROS 2.0 by replacing XMLRPC and UDPROS with a royalty-free, portable DDS implementation.

The embedded ROS client library transparently maps concurrent activities in ROS such as nodes and subscriber callbacks to the underlying task system of FreeRTOS. Thus, the RTOS mechanisms are constructively used to cope with run-time determinism and real-time properties while the complexity of FreeRTOS is hidden from the ROS developer. Furthermore, our embedded system introduces optional API extensions that allow the ROS programmer to easily control the real-time behavior of the ROS system using deadline scheduling mechanisms.

Our real-time ROS bridges the gap between embedded and general-purpose software. We have provided a tutorial on how to compile and flash the code to an STM32F4Discovery for ROS developers, along with two example applications: publishing ultrasonic and IMU sensor measurements.

Link to our software: https://github.com/bosch-ros-pkg/stm32

References

[1] Paul Bouchier, “Embedded ROS [ROS Topics],” Robotics & Automation Magazine, IEEE , vol.20, no.2, pp.17,19, June 2013
[2] Morgan Quigley, Open Source Robotics Foundation, “Bridging ROS to Embedded Systems” http://www.osrfoundation.org/wordpress2/wp-content/uploads/2015/04/ros_and_embedded_systems.pdf