

Real-time documentation for ROS 2

Real-Time Working Group
09.03.2023

Table of Contents

[Links](#)

[Real-time documentation](#)

[Current status](#)

[Proposal](#)

[Brainstorming, Ideas](#)

[Meeting minutes](#)

[2023-02-08 Meeting notes](#)

[2023-03-09 Break-out session: draft tutorials](#)

Links

Related issues:

- [ros2_documentation/issue:3320 Update ROS 2 real-time documentation](#)
- [ros-realtime/issue:41 Consolidate documentation](#)

[Fork of ros2_documentation \(for the proposal\)](#)

Real-time documentation

Current status

Documentation at docs.ros.org

Ressource	What-to-do?	Assignee	Status
Tutorial/Building-Realtime-rt_preempt-kernel-for-ROS-2.html	outdated link - same as below	Andrei	
Tutorials/Miscellaneous/Building-Realtime-rt_preempt-kernel-for-ROS-2.html	=> update, keep as tutorial	Andrei	
Tutorials/Demos/Real-Time-Programming.html	=> integrate in new RT Tutorial	Carlos	
How-To-Guides/Building-ROS-2-with-Tracing-Instrumentation.html	keep as-is	-	OK
How-To-Guides/Installing-on-Raspberry-Pi.html	keep as-is	-	OK
How-To-Guides/Using-callback-groups.html	keep as-is	-	OK
Concepts/About-Executors.html	keep as-is	-	OK

Documentation at ros-realtime.github.io

Ressource	What-to-do?	Assignee	Status
ros-realtime.github.io/Guides	Move ROS2 Tracing to docs.ros.org	Oren	
ros-realtime.github.io/Benchmarks			
ros-realtime.github.io/Subprojects			

ros-realtime.github.io/Related-Projects			
ros-realtime.github.io/Resources			

Other documentation:

Ressource	What-to-do?	Assignee	Status
Executor Workshop	add link to some page		
Introduction to Real-time Systems	review and add relevant information to new page in Concepts	Jan	
https://shuhaowu.com/blog/2022/03-linux-rt-appdev-part3.html	integrate (copy/paste) into Concepts page	Shuhao	

Documentation Work-In-Progress

Ressource	What-to-do?	Assignee	Status
ros.docs.org/How-To-Guides/Developing-a-real-time-application	convert into Tutorial/Real-time Application Configuration	Jan	

Proposal

13.02.2023

Summary:

1. move one tutorial about “ROS 2 Tracing” from ros-realtime to mainline
2. create a subgroup “Real-Time” under “Tutorial/Advanced” with two new Tutorials about Real-Time Development
3. add one page under Concepts with some Background information about Real-Time in ROS 2.

Proposed new structure (changes are shown in **bold** font):

Tutorials

- Advanced
 - **ROS 2 Tracing**
https://ros-realtime.github.io/Guides/ros2_tracing_trace_and_analyze.html)
 - Change original article to redirect stub
 - **Real-Time**
 - **[Real-Time Application Configuration \(Carlos, Jan\)](#)**
 - describe callback groups and thread priorities (one solution only - not going into different options (possibly advanced tutorial on ros-realtime later on)
 - demonstrate with [Pendulum Demo](#)
 - Update the demo proposal:
<https://github.com/ros-realtime/community/issues/37>
 - <https://github.com/ros-realtime/ros2-realtime-examples>
 - **[Operating System Setup with RT Linux Kernel \(moved from ros-realtime\)](#) (Andrei)**

Concepts

- **About-Real-Time-In-ROS2 (Jan, Shuhao)**
 - background about real-time and ROS 2 Middleware/Executors
 - integrate relevant information from Design document [Introduction to Real-time Systems](#)
- [Concepts/About-Executors.html](#) (keep as-is)

How-To-Guides (keep all as-is)

- [Building-Realtime-rt_preempt-kernel-for-ROS-2](#)
- [Building-ROS-2-with-Tracing-Instrumentation](#)
- [Installing-on-Raspberry-Pi](#)

- [Using-callback-groups](#)

Brainstorming, Ideas

Tutorials/Advanced:

- How to develop a real-time application in ROS2?
 - multiple single-threaded Executors, which have just one node, running in different threads (with real-time priorities)
 - callback groups (grouping callbacks from multiple nodes and assign it to one executor, that runs in a real-time thread)
 - WIP:
 - https://github.com/boschresearch/ros2_documentation/blob/feature/how-to-develop-real-time-application/source/How-To-Guides/Developing-a-real-time-application.rst
-
-
- Advanced tutorial with the reference_system? Comparison of different executors on RT Linux Rasp Pie?
 - single_threaded
 - static_single_threaded
 - multi-threaded
 - thread_priorities
 - callback groups and RT priorities
- Demos
 - Real-Time Demo
 - <https://docs.ros.org/en/rolling/Tutorials/Demos/Real-Time-Programming.html>
 - (needs to be updated to be a complete tutorial / or would this better be a How-To?)
- Miscellaneous
 - xx

How-To-Guides

- Setup RT Linux for ROS 2
 - [Tutorials/Miscellaneous/Building-Realtime-rt_preempt-kernel-for-ROS-2.html](https://docs.ros.org/en/rolling/Tutorials/Miscellaneous/Building-Realtime-rt_preempt-kernel-for-ROS-2.html)
- [How-To-Guides/Building-ROS-2-with-Tracing-Instrumentation.html](https://docs.ros.org/en/rolling/How-To-Guides/Building-ROS-2-with-Tracing-Instrumentation.html)
 - could be combined with:
 - https://ros-realtime.github.io/Guides/ros2_tracing_trace_and_analyze.html
 - ?
- [How-To-Guides/Installing-on-Raspberry-Pi.html](https://docs.ros.org/en/rolling/How-To-Guides/Installing-on-Raspberry-Pi.html)

- [How-To-Guides/Using-callback-groups.html](https://ros-realtime.github.io/Guides/Using-callback-groups.html)
- How To Configure RMW middleware (QoS)
 - https://ros-realtime.github.io/Guides/Configure-RMW-Implementation/configure_rmw.html
 - no information available
- How to setup a real-time operating system (or would this be a Tutorial?)
 - https://ros-realtime.github.io/Guides/Real-Time-Operating-System-Setup/rtos_setup.html

Concepts

- [Concepts/About-Executors.html](https://ros-realtime.github.io/Concepts/About-Executors.html)
- About-Real-Time-In-ROS2
 - Middleware (Executors, Multi-threaded-Executors(but no priorities)
 - sequential processing in single threaded executor
 - static executor (improved performance, no shared pointers)
 - events executor (event queue instead of wait-set)
 - dynamic memory allocation for messages
 - Real-time operating system (thread priorities)
 - Combination of Executors/callback groups/multiple Linux threads
 -

Meeting minutes

2023-02-08 Meeting notes

- do packages need to be re-compiled for real-time?
- have only a few tutorials (in advanced section)

2023-03-09 Break-out session: draft tutorials

Real-time processing using multiple threads

- use-case:
 - application with a hard real-time control loop (e.g. 1kHz control loop) and a number of best effort subscription callbacks
- what you will learn:
 - configure a real-time thread (aka a thread with a real-time priority)
 - map the processing of a callback to a real-time thread
- details:
 - two threads t1, t2
 - thread t1: executor e1
 - thread t2: executor e2
 - t1 with real-time priority
 - hard real-time control callback cb_1 => added to e1
 - multiple soft real-time callbacks cb_i => added to e2
- Related:
 - https://ros-realtime.github.io/ros2-realtime-examples/minimal_scheduling/
 - https://github.com/ros-realtime/ros2-realtime-examples/blob/rolling/minimal_scheduling/minimal_scheduling_callback_group.cpp
 - https://github.com/ros-realtime/ros2-realtime-examples/blob/rolling/minimal_scheduling/minimal_scheduling_spin_thread.cpp

Minimize end-to-end latency of multiple-node system

- use-case:
 - larger application with multiple nodes. Each node has multiple callbacks. There is one critical path to minimize the end-to-end latency
- what you will learn:
 - executor with callback groups
 - distribution of callbacks to callback groups and different prioritized threads
- details

Memory locking

- avoid page-falls
- Related:
 - https://github.com/ros-realtime/ros2-realtime-examples/blob/rolling/minimal_memory_lock/minimal_memory_lock.cpp
 - https://ros-realtime.github.io/ros2-realtime-examples/minimal_memory_lock/

Thread-configuration of ROS middleware provider

- What happens with other threads (RMW, DDS threads)?
- Example to configure DDS-threads
 - main thread configuration before DDS is configured (rclcpp::init)
 - tuning of DDS configuration
 -
- Related
 - https://github.com/ros-realtime/ros2-realtime-examples/blob/rolling/minimal_scheduling/minimal_scheduling_middleware_threads.cpp
- Related:
 - <https://fast-dds.docs.eprosima.com/en/v2.0.0/realtime.html>
 - https://github.com/ros-realtime/ros2-realtime-examples/tree/rolling-experimental/minimal_dds_tuning

Best practices

- using MultiThreadedExecutor; but callbacks are not processed in parallel
 - one node, multiple subscriptions added to the node => implicitly they are all added to the same default callback_group::MutuallyExclusive
 - node added to MultiThreadedExecutor => all callbacks are processed sequentially
 - Solution1: Each subscription derived from Node, add all Nodes to an Executor (simple)
 - Solution2: use callback_groups to configure which callbacks can run in parallel (more complex)
 - reference to “Using callback groups” page
- large jitter of subscription latency even though it is called in real-time thread
 - setup:
 - e.g. 2 subscription callbacks A, B
 - one callback C with real-time requirements
 - what happened: if subscription callbacks A and B received new data: their callbacks might be processed before callback C (Executor processes incoming events sequentially, similarly with MultiThreadedExecutor modulo number of threads)
 - Solution: callback_groups and real-time threads
 - two threads T1, T2; T2 with real-time priority.
 - two executors E1, E2
 - put A, B in cbg_1, add cbg_1 to E1
 - put C in cbg_2, add cbg_2 to E2
 - E1 processed in T1, E2 processed in T2

Improving performance with Static Executors

- better performance
- static configuration of entities at configuration time

Backlog:

Deterministic execution order with WaitSets

- use-case:
 - sensor fusion: synchronize multiple sensor input with different rates
 -
- what you will learn:
 - specify deterministic execution order using `rclcpp::WaitSet`
 - be aware of potential busy-loop: performance degradation!
 -
- details
 - example: synchronization:
https://github.com/ros2/examples/blob/b327f8cd19a8278631116f01b96bdfc7b55205ec/rclcpp/wait_set/src/wait_set_random_order.cpp#L55
 -
 - example: multiple rate publishers:
https://github.com/ros2/examples/blob/rolling/rclcpp/wait_set/src/wait_set_topics_with_different_rates.cpp

similar to message filters