Poster: How to Send Large Data in ROS 2

Sanghoon Lee, Taehun Kim, Jiyeong Chae, and Kyung-Joon Park
Department of Electrical Engineering and Computer Science, DGIST, Daegu, Republic of Korea
Email: {leesh2913, taehun, cowldud3, kjp}@dgist.ac.kr

***Abstract*—High-resolution data streams-such as images, LiDAR point-clouds-are common in robotic communication, yet Robot Operating System 2 (ROS 2) struggles to transmit these streams due to increased average latency. On lossy wireless links, the default DDS communication stack in ROS 2 suffers significant performance degradation. This paper presents the first comprehensive network-layer analysis of ROS 2's DDS stack operating over wireless links with large payloads. We analyze three network bottlenecks that emerge during large-payload data transfers and presents DDS-level optimizations for each one. The proposed solutions are exposed through an XML-based QoS configuration interface, allowing them to be easily tuned. Experiments demonstrate that our approach transmits large-payload data successfully while maintaining lower latency than existing methods.**

***Index Terms*—Robot Operating System 2 (ROS 2), Data Distribution Service (DDS), Robotic Communication)**

I. Introduction

Modern robotic systems transmit and receive data from a variety of high-resolution sensors-such as LiDARs and RGB cameras. These high-performance sensors increase the volume of data, and exchanging such large-payload data between robots has become essential. In particular, multiple mobile robots and collaborative robots are widely deployed in real-world settings, and in such environments delay, packet loss, and jitter are critical performance criteria.

Robot Operating System 2 (ROS 2) has emerged as the de facto standard for robotic communication [1]. It relies on the Data Distribution Service (DDS) middleware to transmit and receive messages. DDS is built on the Real-Time Publish-Subscribe(RTPS) protocol and can exchange messages through a variety of Qualify of Service (QoS) policies.

Although its widespread adoption, DDS is not well suited for processing large-payload data [2]. DDS provides User Datagram Protocol (UDP)-based communication, which is intended to enable multicast transmission and minimize latency and protocol overhead. Fast DDS-the default implementation in ROS 2-offers a LARGE_DATA mode [3] to facilitate large-payload transmission, but this feature operates over Transmission Control Protocol (TCP) and therefore does not fundamentally address the UDP-based nature of DDS. We identify three primary causes that prevent DDS from transmitting large-payload data smoothly. Each cause is analyzed and a corresponding solution is proposed, culminating in a comprehensive optimization mode that facilitates large-payload data transmission. The proposed optimization mode is compared with existing methods and validated through experiments.

979-8-3315-0376-5/25/$31.00 ©2025 IEEE

II. DDS Optimization

A. IP Fragmentation

The dominant cause of message loss when transmitting large-payload data is packet-level fragmentation. In DDS, message transmission is fragmented twice: one by the maximum RTPS payload size and again by the IP Maximum Transmission Unit (MTU). By default, DDS sets the maximum RTPS message size to 64 KB, while the typical IP MTU is 1500 B. Consequently, the number of IP fragments for a single message, $N_{\mathrm{IP}} = \lceil 64\,\mathrm{KB}/1500 \rceil = 44$. Adjusting the RTPS message size is therefore a far more effective means of reducing traffic overhead.

The most effective and simple way to mitigate this problem is to prevent IP fragmentation by applying a one-to-one mapping between each RTPS message and a single IP packet (i.e., $N_{\mathrm{IP}} = 1$). Most DDS vendors provide an XML profile interface that allows users to configure the maximum RTPS message size. Since the RTPS header size is 28 B and the standard MTU is 1500 B, setting the maximum RTPS message size to 1472 B is recommended to ensure fragmentation-free transmission at the network layer.

B. Retransmission Timing

DDS ensures reliability by providing retransmission mechanism using heartbeat and Acknack. Within the mechanism, asynchronous interactions between publishing and retransmission can cause buffer overflow or link congestion. DDS can automatically reduce average latency and jitter by increasing the retransmission rate. Using probabilistic analysis, Park et al. [4] showed that shortening the retransmission interval markedly improves ROS 2 DDS latency, reducing both average delay and jitter.

However, an excessively high retransmission rate can incur overhead due to increased processing by the heartbeat mechanism and a surge in RTPS control packets. Lee et al. [5] demonstrated that configuring the retransmission rate to twice the publish rate yields a local optimum. Based on this study, we propose setting the retransmission rate to twice the publish rate to suppress retransmission-induced traffic bursts and minimize latency and jitter. The retransmission rate can be easily adjusted through a QoS profile in the DDS XML configuration.

C. Buffer Burst

Unlike wired networks, a prominent challenge in wireless communication is link outage. Because wireless links lack physical connectivity, link loss frequently arises from obstacles

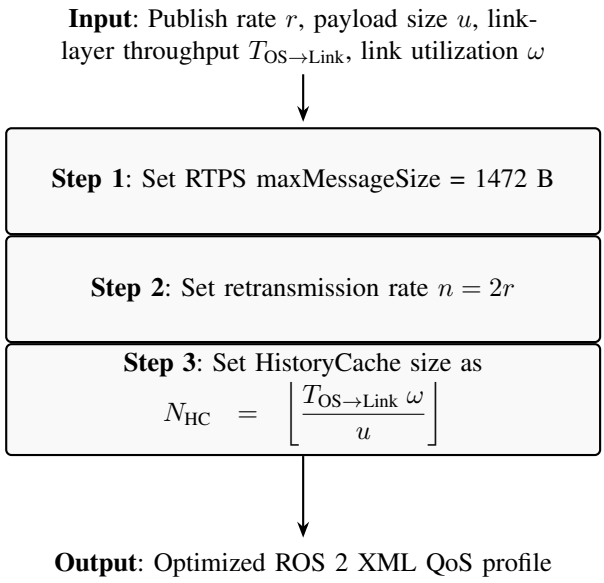


Fig. 1: DDS Optimization for Wireless Large Payload Transfer

or movement beyond the coverage area. Even when a link outage occurs, ROS applications and the underlying DDS middleware remain unaware of the disconnection. Consequently, ROS applications continue to publish data, and the DDS middleware continues to store samples in the HistoryCache. These samples accumulate until the HistoryCache reaches its capacity. DDS retains each sample until acknowledgments are received from all expected subscribers, eventually saturating the buffer with unacknowledged samples.

When the link is restored after a link outage, DDS triggers a buffer burst. During this process, the DataWriter sequentially retransmits all samples stored in the HistoryCache. If the HistoryCache is too large and contains a large number of samples, the buffer bursts, saturating the wireless channel. In wireless networks, link saturation acts as a positive feedback loop that significantly increases latency by continuously causing per-second delays, leading to serious data transmission errors.

To prevent this problem, we recommend limiting the HistoryCache size via a DDS QoS parameter that can be tuned in XML settings. We have developed a formula for setting an appropriate N_{HC} size, and setting the calculated value prevents buffer bursts.

D. DDS Optimization Framework

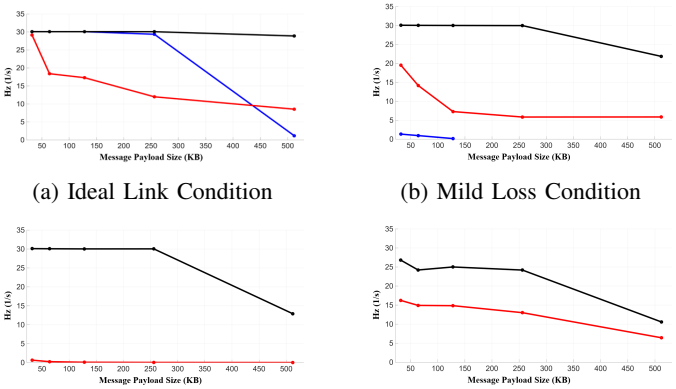
We propose the DDS Optimization Framework shown in Fig. 1, which addresses these three challenges. This framework can be easily implemented using the ROS 2 XML-based QoS configuration interface without requiring any additional, unnecessary operations. It is suitable as a simple solution because it can be applied to any ROS 2 application with minimal XML-based configuration.

III. Experimental Evaluation

We conducted experiments comparing the default DDS configuration, Fast DDS's LARGE_DATA mode, and our proposed optimized configuration. The experiments are conducted using two laptops connected via an IEEE 802.11ac 5 GHz wireless router. The theoretical link throughput $T_{\text{OS}\rightarrow\text{Link}}$ is 433 Mb/s, and the system runs Ubuntu 22.04 with ROS 2 Humble. The default DDS implementation is FastDDS v2.6.9.

To simulate real-world communication, we considered four wireless conditions: Ideal link-clear channel with no external interference; Mild loss-1 % packet-error rate (PER); Severe loss-20 % PER; and Link outage-a complete disconnection lasting 20 s. For each condition, we use ROS 2's rclpy interface to send messages of four types (standard ROS 2 message types, images point cloud, string, and UInt8Array) with a total of five payloads (32, 64, 128, 256, and 512 KB) at 30Hz. All transmissions are configured with RELIABLE and KEEP_ALL histories to ensure strict end-to-end reliability.

The results, shown in Fig. 2, demonstrate that the proposed DDS Optimization is suitable for a variety of network conditions in real-world environments. The y-axis shows the reception rate (Hz), and it is evident that the optimized configuration (black line) maintains 30 Hz more consistently than both the default (blue line) and the LARGE_DATA modes (red line). Importantly, with the proposed optimization applied, efficient communication can be achieved without relying on Fast DDS's TCP-based LARGE_DATA mode.



(a) Ideal Link Condition (b) Mild Loss Condition

(c) Severe Loss Condition (d) Link Outage Condition

Fig. 2: Reception rate (Hz) across DDS modes and payload sizes (Black: optimized profile, Red: LARGE_DATA mode, Blue: default profile)

IV. Conclusion

In this paper, we proposed a lightweight DDS optimization framework that improves transmission robustness by utilizing XML-based QoS configuration without requiring any changes to the protocol or application logic. We demonstrate that, through DDS tuning, our optimization technique paves the way for efficient data transmission in edge-based robotic systems operating over wireless networks. In order to completely solve the challenges of wireless robotics, future research should explore DDS optimization framework in multi-node environments that interact over shared physical links.

References

[1] S. Macenski, T. Foote, B. Gerkey, C. Lalancette, and W. Woodall, "Robot operating system 2: Design, architecture, and uses in the wild," *Science robotics*, vol. 7, no. 66, p. eabm6074, 2022.

[2] Y. Maruyama, S. Kato, and T. Azumi, "Exploring the performance of ROS 2," in *Proceedings of the 13th international conference on embedded software*, pp. 1–10, 2016.

[3] eProsima, "Fast DDS performance tests." eProsima Fast DDS Benchmark, 2023. Accessed: July 29, 2025.

[4] H.-S. Park, S. Lee, D. Um, H. Ryu, and K.-J. Park, "An analytical latency model of the data distribution service in ROS 2," in *IEEE INFOCOM 2025-IEEE Conference on Computer Communications*, pp. 1–10, IEEE, 2025.

[5] S. Lee, H.-S. Park, J. Chae, and K.-J. Park, "Probabilistic latency analysis of the data distribution service in ROS 2," *arXiv preprint arXiv:2508.10413*, Aug. 2025.