Vol.2022-EMB-60 No.3
2022/7/8

ロボットソフトウェア軽量実行環境mROS 2のPOSIX対応の検討

細合 晋太郎[1]　田中 晴亮[1]　高瀬 英希[1]

概要：mROS 2は，ロボットソフトウェアの開発を加速するプラットフォームであるROSの軽量実行環境である．本研究では，mROS 2のソフトウェア階層内における通信プロトコルスタックのPOSIXへの対応を検討する．通信層において利用されているリアルタイムOSの機能を整理し，これらと同等の機能を提供するPOSIX APIの互換層を設計する．まずはUbuntu環境に対して検討結果を実装し，本環境上で動作するmROS 2ノードがRTPSプロトコルに基づく自律的な通信ができることを確認した．

1. はじめに

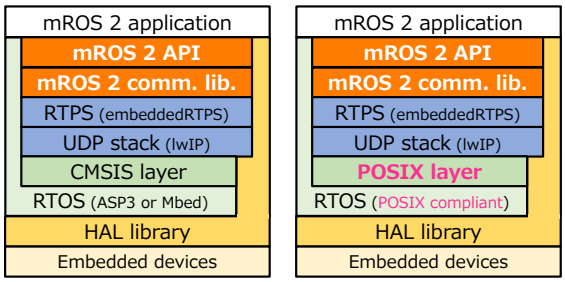
ロボットシステムの開発を加速するプラットフォームであるROS（Robot Operating System）[1]の普及が進んでいる．ROSの本質は通信であり，基本機能として出版購読型の通信方式に基づく．ROS 2[2]の通信ミドルウェアであるDDS（Data Distribution Service）およびRTPS（Real-Time Publish Subscribe Protocol）は，通信相手の探索および通信経路の確立を自律的に行う機能を備える．

我々は，組込みデバイス向けのROS 2ノードの軽量実行環境であるmROS 2[3]の研究開発を進めている．本環境では，中規模の組込みデバイス上で実行されるプログラムについて，汎用デバイス上のROS 2ノードと自律的に通信する機能を提供する．分散型のロボットシステムへの組込み技術の導入の促進が期待され，通信処理にかかる応答性やリアルタイム性の向上，および，エッジデバイスにおける消費電力の削減に貢献できる．なお，我々は，本環境によって，ROS 2およびDDS/RTPSの通信技術を広域分散型のIoTシステムの構築へ展開することも狙っている[4]．

本研究では，mROS 2の通信プロトコルスタックにおけるPOSIX（Portable Operating System Interface）[5]への対応を目指す．POSIXは主にOSの標準的なインタフェースを定義するIEEE規格であり，UNIX系の汎用OSに限らず，組込み向けのリアルタイムOSでも概ね準拠または互換性のあるラッパ層を提供するものがある．すなわち本研究の目的を達成することで，mROS 2の通信機能を様々なOS環境で動作できるようになることが期待される．

2. mROS 2

本章では，組込みデバイス向けの高効率なROS 2通信方式およびメモリ軽量な実行環境を提供するmROS 2[3]について，ソフトウェア構成を中心に概説する．



(a) 既存研究での設計　(b) 本研究での設計

図 1　mROS 2のソフトウェア構成

文献[3]において提案したmROS 2のソフトウェア構成を**図 1(a)**に示す．上階層から順に，まずmROS 2アプリケーション層は，ユーザが実装するROS 2ノードに相当する．mROS 2 API層および通信ライブラリ層は，ROS 2の`topic`に相当するAPIおよび通信機能を提供する階層であり，rclcppと互換性を保つように設計している．

RTPSプロトコルスタックには，C++で実装されたembeddedRTPS[6]を採用している．メモリ領域の静的確保など組込みデバイスでの稼働を想定して設計されており，通信の自律性が確保されている．UDPについては組込み向けのC実装であるlwIPが用いられている．これらの階層はCMSIS-RTOS APIに実装依存している．

mROS 2では，実行環境としてリアルタイムOSを採用し，組込みデバイス上での効率的な出版購読の通信処理を実現している．現時点では，TOPPERS/ASP3カーネル[*1]およびMbed OS 6[*2]を用いた実装をそれぞれ公開しており，任意型のメッセージの通信処理方式[7]にも対応している．最下層にはハードウェア抽象化ライブラリがある．

ROSコミュニティにおけるデファクトの地位にある組込み向けの実行環境にmicro-ROS[8]がある．文献[9]では，mROS 2およびmicro-ROSについてそれぞれの優位性を議論した．STM32 NUCLEO-F767ZIを対象として通信性能およびメモリサイズを定量的に評価したところ，mROS 2

[1] 東京大学
The University of Tokyo

[*1] `https://github.com/mROS-base/mros2-asp3-f767zi`
[*2] `https://github.com/mROS-base/mros2-mbed`

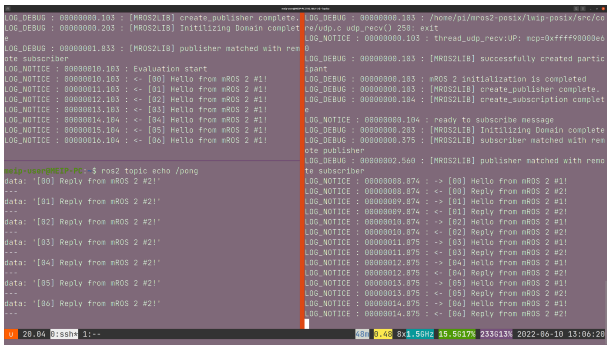


図 2 mros2-posix の動作例

がともに遥かに優れる結果を示すことを立証している．

3. POSIX への対応

POSIX 準拠として公式認証を受けているものは，UNIX 系の汎用 OS が中心となる．組込み向けのリアルタイム OS では，概ね準拠したものは NuttX や RTMES など，互換性のあるラッパ層を提供するものは FreeRTOS などがある．

2 章で示した通り，mROS 2 の通信プロトコルスタックである embeddedRTPS およびその下層の lwIP は，CMSIS-RTOS API に依存している．Mbed OS 6 は本レイヤを標準的に備えている．TOPPERS/ASP3 カーネル向けの実装については，それぞれの API 差分を吸収するラッパ層である cmsis-asp3[*3]を用意して対応した．

以上の議論を踏まえ，POSIX 互換のラッパ層を備える軽量実行環境である mros2-posix を提案する．本実行環境のソフトウェア構成を**図 1(b)** に示す．mROS 2 内部で RTPS 通信を実現するための機能は，スレッドの生成や開始などの管理機能，ミューテックスによる排他制御機能，RTPS パケット処理のためのメッセージキューによる同期・通信機能とメモリの管理機能，および，OS 時刻や時間経過待ちに関する時間管理機能に大別される．これらは現代の OS では一般的な機能であり，POSIX でも標準的に備える．そこで本研究では，cmsis-asp3 において担っていたこれらのラッパ機能を起点とし，CMSIS-RTOS API と POSIX のラッパ層である cmsis-posix を設計した．加えて lwIP については，UDP パケットの処理における OS 資源の操作に関する lwip-posix を設計した．

本研究では，Ubuntu 20.04 LTS（Focal）を実装対象とした．ただし，本 OS で準拠している POSIX API[10] のうち，今後の POSIX 準拠のリアルタイム OS 向けの移植容易性を考慮して，NuttX のユーザ向け API[11] として公開されているものを選択することとした．

図 2 に，本研究において実装した mros2-posix の動作例を示す．左上のターミナルでは，String 型のメッセージを出版している．右側のターミナルでも mros2-posix 上のノードを実行し，本メッセージを購読したのちに別のトピックにそのまま出版する処理を行っている．左下のターミナルでは，ROS 2 のネイティブ環境が返送されたメッセージを購読している．各ターミナルは異なるデバイス上で動作しており，提案する実行環境上で動作する mROS 2 ノードが互いに自律的に通信できていることがわかる．

[*3] https://github.com/mROS-base/cmsis-asp3

4. おわりに

本研究では，mROS 2 の POSIX 対応版である mros2-posix の検討を進めた．まずは Ubuntu 環境に対して検討結果を実装し，本環境上で動作する mROS 2 ノードが RTPS プロトコルに基づく自律的な通信ができることを確認した．

mROS 2 の通信ライブラリは，GitHub（https://github.com/mROS-base/mros2）にて OSS として公開している．mros2-posix の現状の実装は，文献 [7] の任意型メッセージの通信処理方式に未対応であり，これを完了したのちに同じく公開予定である．また，実際のリアルタイム OS 上で動作する mros2-posix の実装例も公開できればと考えている．ご興味のある方は，ぜひ試用いただき，研究開発への採用や参画を検討いただけたら幸甚である．

謝辞 本研究の一部は，JST CREST JPMJCR21D2 ならびに国立研究開発法人情報通信研究機構の委託研究（04001）により得られたものである．本研究の実装に協力いただいた永和システムマネジメントの森崇氏に深くいたします．

参考文献

[1] Quigley, M., et al.: ROS: an open-source Robot Operating System, *Proc. of ICRA workshop on open source software*, No. 3.2, pp. 1–6 (2009).
[2] Macenski, S., et al.: Robot Operating System 2: Design, architecture, and uses in the wild, *Science Robotics*, Vol. 7, No. 66 (2022).
[3] 高瀬 英希，祐源 英俊：mROS 2：組込みデバイス向けの ROS 2 ノード軽量実行環境，情報処理学会研究報告，Vol. 2022-EMB-59, No. 37, pp. 1–8 (2022).
[4] 高瀬英希：ROS (Robot Operating System) の紹介と IoT/IOT 分野への展開，RICC-PIoT workshop 2022.
[5] IEEE SA: P1003.1 (online), https://standards.ieee.org/ieee/1003.1/7700/ (2022.06.12).
[6] Kampmann, A., et al.: A Portable Implementation of the Real-Time Publish-Subscribe Protocol for Microcontrollers in Distributed Robotic Applications, *Proc. of ITSC*, pp. 443–448, (2019).
[7] 檜原 陽一郎，中村 宏，高瀬 英希：ROS 2 ノード軽量実行環境 mROS 2 における任意型メッセージの通信処理方式，情報処理学会研究報告，Vol. 2022-EMB-59, No. 38, pp. 1–8 (2022).
[8] micro-ROS | ROS 2 for microcontrollers (online), https://micro.ros.org/ (2022.06.12).
[9] 高瀬 英希，檜原 陽一郎：組込み向け ROS 2 ノード実行環境の定量的評価，ロボティクス・メカトロニクス講演会 2022 講演論文集 Vol. 22-2, No. 1P1-Q09, pp. 1–4 (2022).
[10] Ubuntu Manpage: Directory Listing (online), https://manpages.ubuntu.com/manpages/focal/en/man3/ (2022.06.12).
[11] Userspace API – NuttX latest documentation (online), https://nuttx.apache.org/docs/latest/reference/user/index.html (2022.06.12).