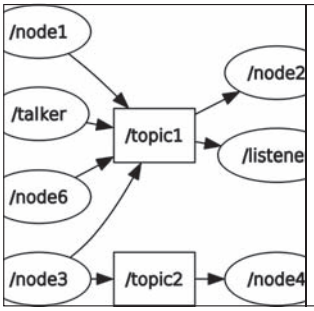
本誌のご購入はこちら



第3章 多数のアプリケーションが非同期に動くための通信をシンプルなプログラムで試す

作ってわかる！ ROSの通信機能のしくみ

高瀬 英希 Hideki Takase

ROSを使うメリットは，充実したツールやライブラリの他に，複数のソフトウェアが相互に通信して複雑な処理をこなす構造を作れることです．

ROSの通信のしくみがどのようなものなのかを解説します．ごく簡単なアプリケーションを自分で作って動かしてみると理解が進みます． 〈編集部〉

本稿の目的

● ROSの構成単位

ROSにおけるアプリケーションの実行単位は「ノード」と呼ばれます．ノードの実装は基本的には実行可能ファイルに対応しています．ROSは通信ミドルウェアとして，ノード同士がさまざまな方法で相互に通信する機能を提供しています．そして複数のノードで構成される機能の集合のことを「パッケージ」と呼びます．

● ROSの通信機能

ROSで最も一般的な通信機能は，非同期方式の出版購読型通信である「トピック」です．

図1のように，任意のノードが自身のタイミングでデータを出版できます．購読側のノードではトピックにデータが到着したら，ノードに対して登録しておいたコールバック関数を呼び出して処理します．

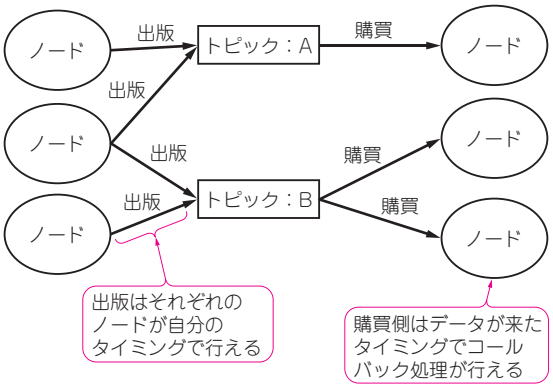


図1 ROSのトピック通信は，任意のノードが任意のトピックについて非同期で出版および購読できるしくみ

● 手を動かすと理解が早い

ノードとパッケージの関係およびノード間の通信機能は，ROSの初学者が最初に理解すべき重要な基本要素です．それらを理解するために，本稿ではROSのパッケージとノードを自分で作ってみます．ROS通信機能としてはトピックを介した出版購読型通信を取り上げ，開発言語にはC++を利用します．

事前の準備と前提条件

● Ubuntu 18.04とROS Melodicをインストール済み

本章では，前章で解説しているUbuntu 18.04とROS Melodic Moreniaのインストールが済んでいて，さらに下記のコマンドを実行済みであることとします．

つまり作業用のROSワークスペースを ~/catkin_ws として作成，設定しています．

```
$ source /opt/ros/melodic/setup.bash
$ mkdir -p ~/catkin_ws/src
$ cd ~/catkin_ws/src
$ catkin_init_workspace
$ cd ~/catkin_ws
$ rosdep install -y -r
      --from-paths src --ignore-src
$ catkin_make
$ source devel/setup.bash
```

シェルはbashを使用します．以降のターミナルでは，下記のコマンドが実行されていることを前提とします．~/.bashrc に記載しておいてもよいでしょう．

```
$ source ~/catkin_ws/devel/setup.
  bash
```

ターミナルから設定を確認できます．

```
$ env | grep ROS
ROS_ETC_DIR=/opt/ros/melodic/etc/ros
CHOOSE_ROS_DISTRO=melodic
ROS_ROOT=/opt/ros/melodic/share/ros
ROS_MASTER_URI=http://localhost:11311
ROS_VERSION=1
ROS_PYTHON_VERSION=2
ROS_PACKAGE_PATH=/opt/ros/melodic/
```