

2021年12月18日(土)
NEDO特別講座 講習会

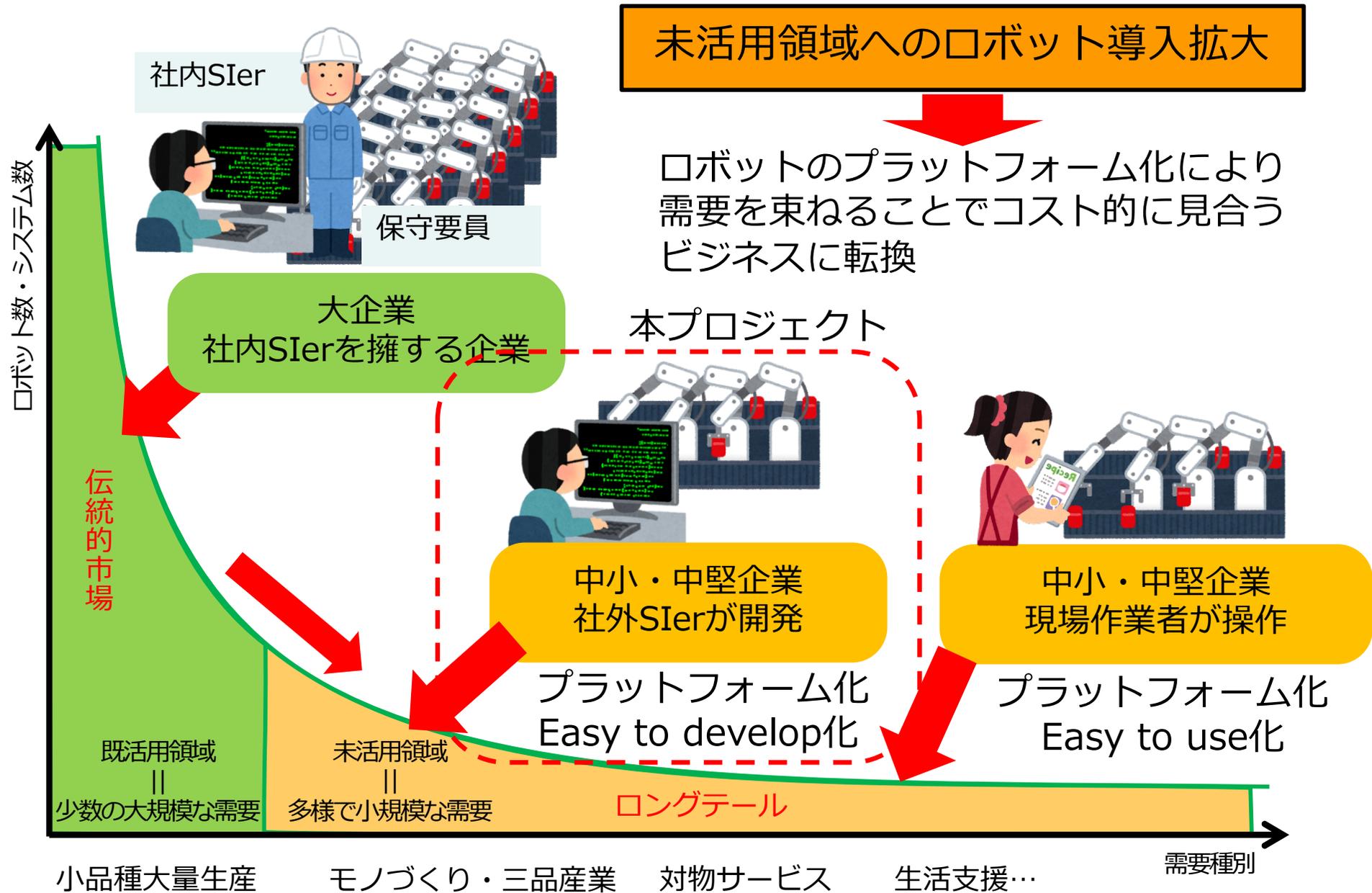
 ROS

第1部：NEDO特別講座 およびROSの概要

国立研究開発法人産業技術総合研究所
インダストリアルCPS研究センター
ソフトウェアプラットフォーム研究チーム長
安藤 慶昭

- NEDO特別講座について
- ROS概要
- ROSの基礎

NEDO特別講座について



RRI（ロボット革命イニシアティブ協議会）WG3における議論を経て2017年度から開始

事業期間

- 2017～2019年度

2017年度公募・採択

ハード開発事業者（7事業者）

- カワダロボティクス
- 富士ソフト・日本電産
- 東芝
- セック・THK・名城大
- 川崎重工
- YOODS
- パナソニック

ソフト開発事業者（5事業者）

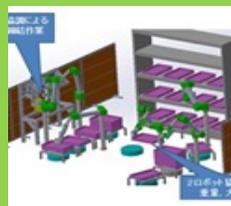
- 産総研
- 東大
- TORC
- イーソル
- JQA

ハードウェアコンソーシアム

カワダ
ロボティクス



富士ソフト
日本電産



東芝



セック・THK
名城大



川崎重工



YOODS



パナソニック



ライフ
ロボティクス

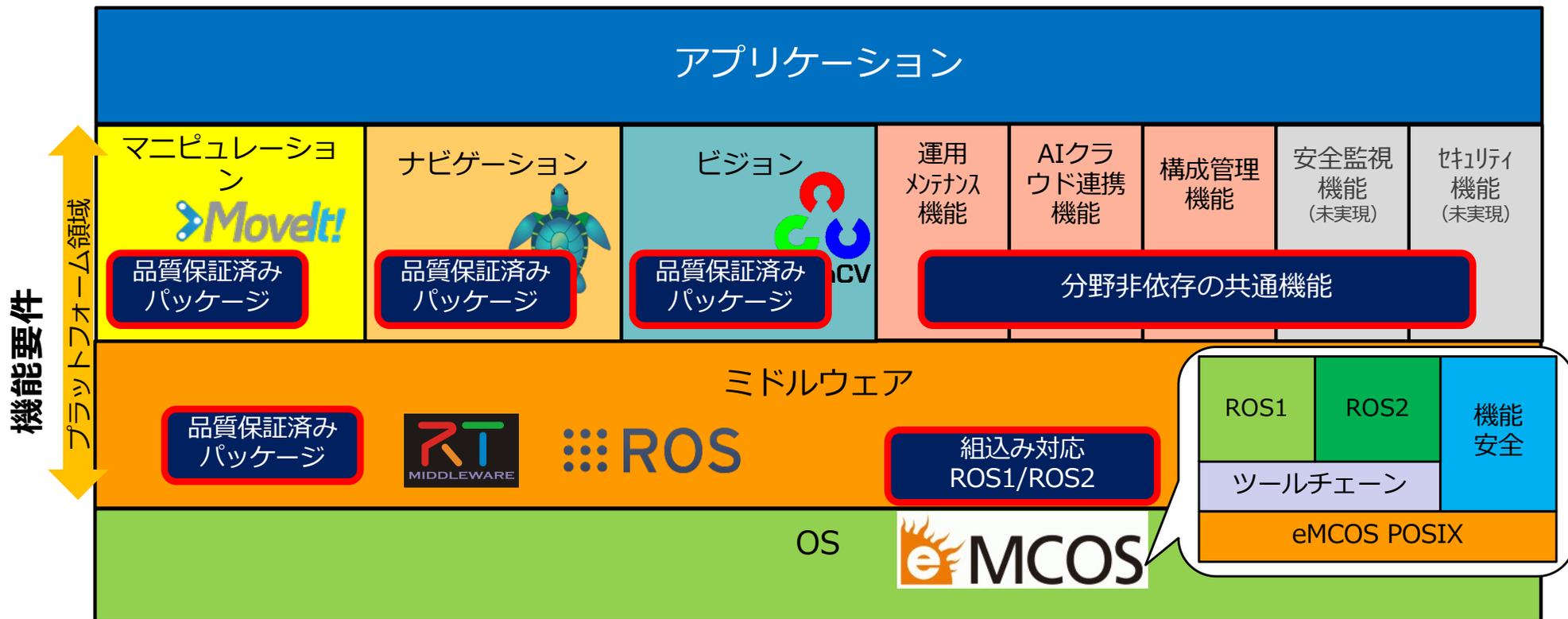


ソフトウェアコンソーシアム





ソフト・ハードともに
「ベース機能」 + 「オプション機能」 + 「個別開発」
 でアプリケーション開発可能にすることで、SIコスト削減、分業促進



非機能要件

- テスト・検証に関するエビデンス
- モジュールの性能指標
- ドキュメンテーション
- 共通インターフェース仕様
- 教育・コミュニティー
- 安全規格とのマッチングに関するガイドライン
- ライセンス・特許に関するガイドライン
- 長期的な保守・運用体制

ROSコード解析・品質改善

保守・維持管理体制検討

一般的な製品における指摘密度

指摘密度[件/KLOC]	エラー	計
最大	0.51	0.41
平均		0.9

- ロボット安全設計開発
- ロボットソフトウェア特許・ライセンス
- ロボットソフトウェアアーキテクチャ
- ロボットシステム開発プロセス・品質管理
- 次世代ロボット実装手法
- 移動ロボット評価指標

調査検討6委員会

物流台車搬送 - 安全対策の進め方

JIS B 9700 (ISO 12100), JIS B 8455 (ISO 13482)

安全対策支援

6月までに

3ステップによるリスク低減

ステップ1: 現状のリスク評価

ステップ2: 安全対策の検討

ステップ3: 安全設計の実現

ポイント

- 既存のカゴ台車系に含まれるロボット安全の整理 / 運用ルールと商品性のバランス
- プラットフォームロボットに求められるユースケースおよびリスクアセスメントの汎用度合い

MoveIt! 詳細ドキュメント

https://robo-marc.github.io/moveit_documents/

Navigation Stack 詳細ドキュメント

https://robo-marc.github.io/navigation_documents/

システム・ノードの仕様レベルまで解説することで、詳細を知らないソフトウェアエンジニアでも利用可能に。

ROSコード改善プロジェクト

https://github.com/robo-marc/ros_comm

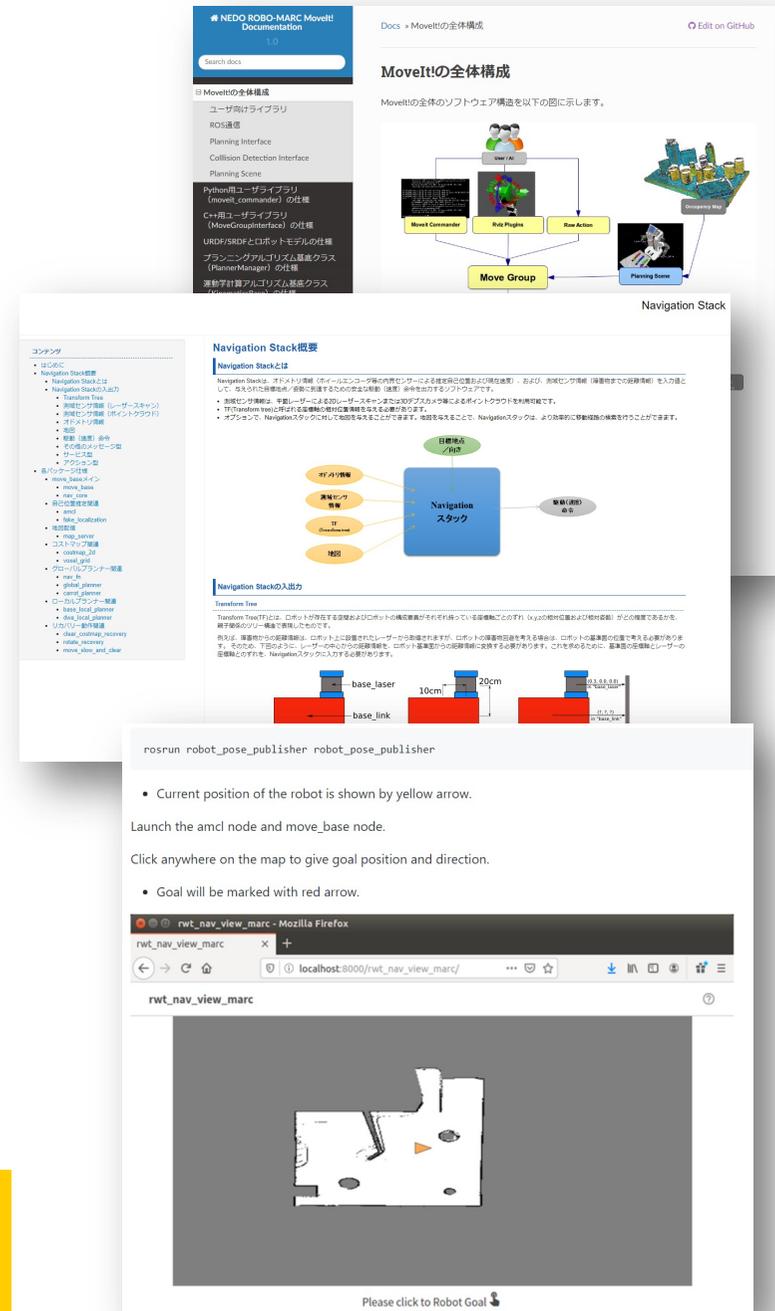
ROS、MoveIt!, NavigationStack等のコードを静的解析ツールで解析、バグ、記述上の問題点を調査し、改善、本家へフィードバック

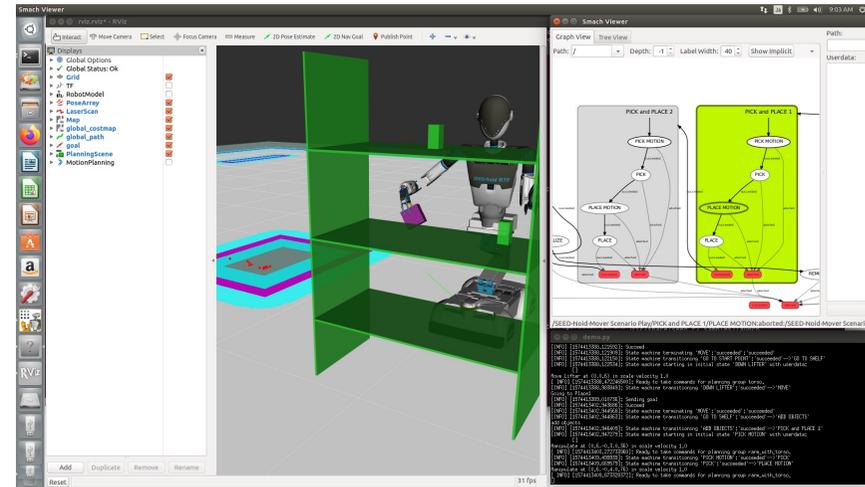
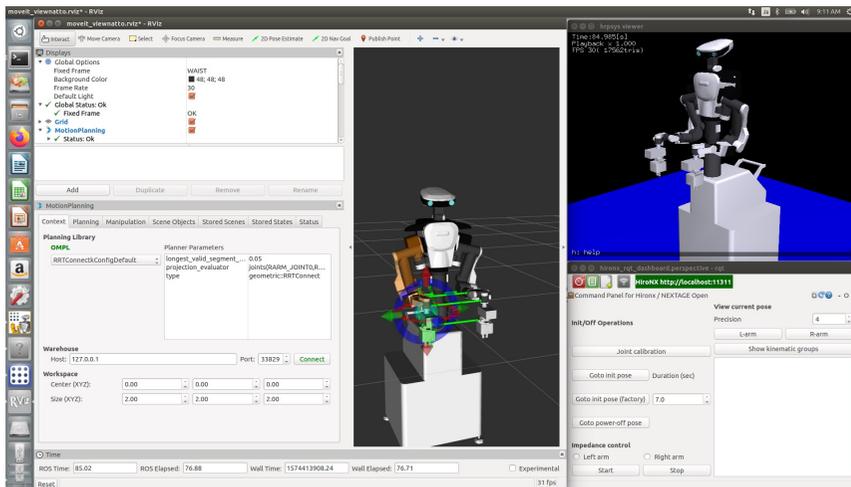
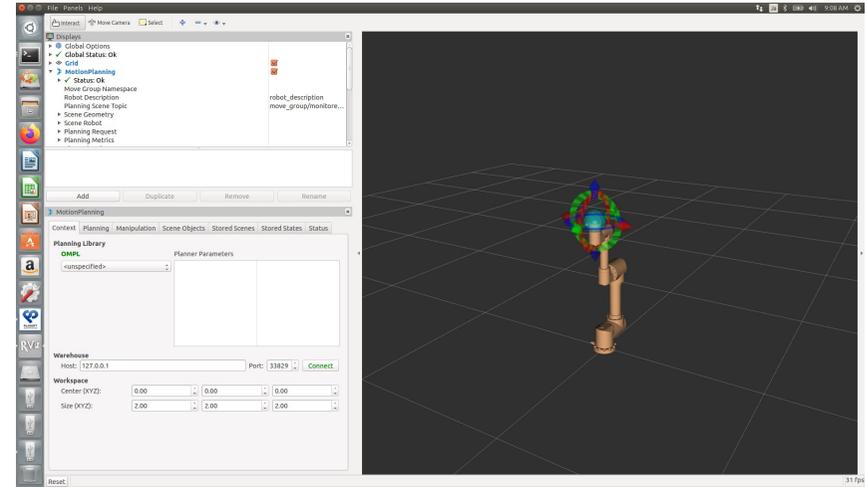
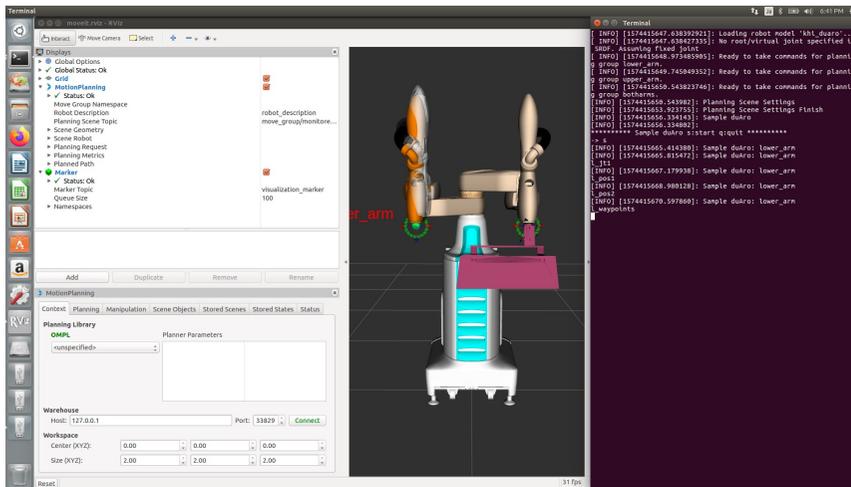
RobotWebTool拡張

https://github.com/robo-marc/visualization_rwt

ブラウザからROSシステムの操作・モニタなどを行うRobotWebToolsを拡張、GUI部品等を追加。Webベースでのシステム構築効率を向上させた。

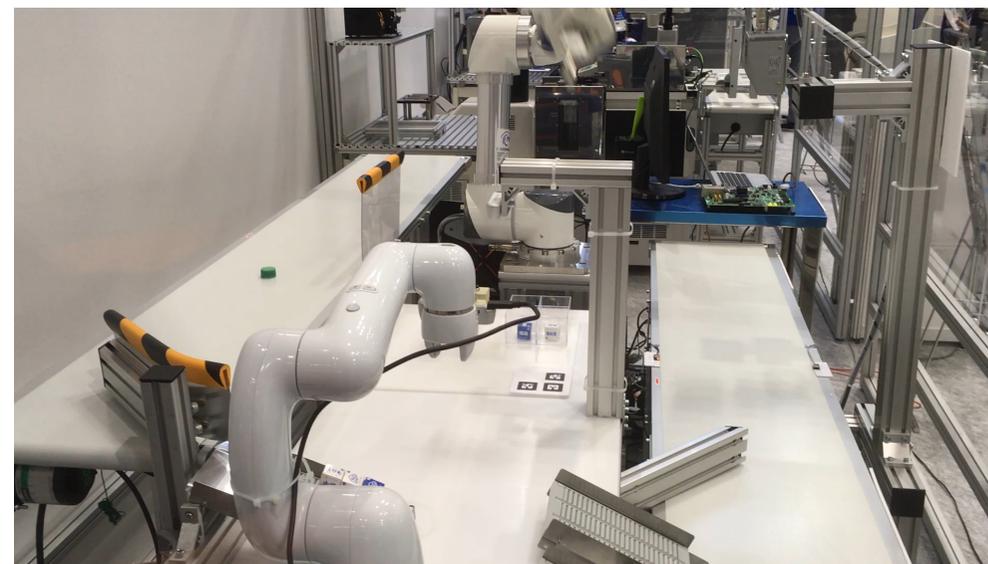
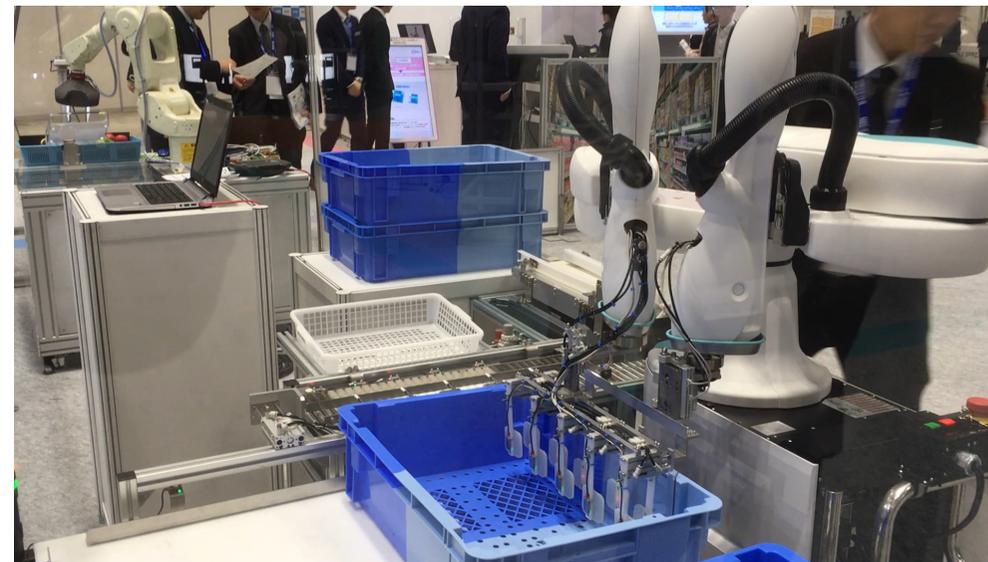
<https://github.com/robo-marc>
をぜひご覧ください。





USBメモリ

ROS Kinetic + 川崎重工、富士ソフト、カワダロボティクス、THKのロボットモデルを動かします！！
 Melodicへアップデート



多数のロボットをROSおよび共通IFで制御

ROS概要

ロボットミドルウェアについて

- ロボットシステム構築を効率化するための共通機能を提供する**基盤ソフトウェア**
 - 「ロボットOS」と呼ばれることもある
 - インターフェース・プロトコルの共通化、標準化
 - 例として
 - モジュール化・コンポーネント化フレームワークを提供
 - モジュール間の通信をサポート
 - パラメータの設定、配置、起動、モジュールの複合化（結合）機能を提供
 - 抽象化により、OSや言語間連携・相互運用を実現
- 2000年ごろから開発が活発化
 - 世界各国で様々なミドルウェアが開発・公開されている

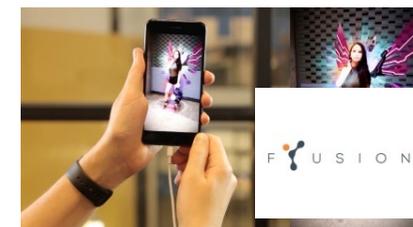
ROS(Robot Operating System)



- Willow Garage
 - 2007年設立のロボットベンチャー (米、Menlo Park)
 - 2014年事業停止
 - Scott Hassanが出資
 - googleの初期エンジンの作者の一人
 - ビジネスモデル
 - ソフト：ROS (無償) +ハード：PR2 を販売
 - PR2を10台無償で大学などに配布
 - スピナウト創出を狙う



Industrial Perception, Inc.



Open Source Robotics Foundation

概要：

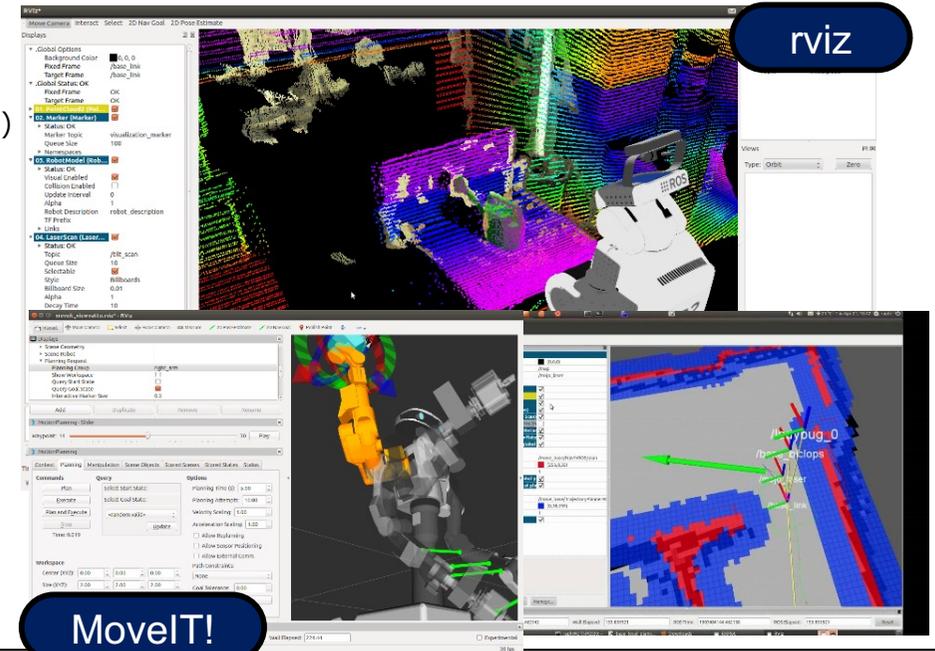
- 米国ベンチャー Willow Garageが開発したロボット用OS。
- 2007年から開発開始。
- オープンソースとして広くソースコードを公開。
- 現在はOpen Roboticsが管理。
- ロボット機能要素（センサ、モータ、アーム、移動機能）をモジュール化。個別に動くプログラムを連携する通信部分を提供。（RTMとコンセプトは同じだが、リアルタイム・密結合機能はない）
- Linuxのソフトウェア管理機構を活用し（Linuxに慣れた人なら）インストールが容易。

ユーザ：

- Saviok、Fetchロボティクス、Clearpath Robotics、トヨタなどがロボット用OSとして採用。
 - 学術分野ではデファクトスタンダードとなっている

特徴：

- Ubuntu Linux†上で動作（コマンド入力による操作が基本）
 - 他のOSは公式にはサポートしていない
- キラーアプリケーションにより人気獲得
 - rviz:ロボットの様々な状態を3Dで表示
 - MoveIT!:アームの軌道計画
 - Navigation Stack: 地図作成・経路計画
- 現在次バージョン：ROS2に移行中
 - RTM同様の密結合・リアルタイム機能含む
 - 通信はOMGのDDS標準を利用（RTMと互換性有）



- aibo
 - 新AiboではOpen-RではなくROSを利用している模様
 - ROS kinetic (バージョン) を利用
 - 約500以上のオープンソースソフトウェア (ROS含む) を利用している
 - ライセンスをWebサイト上で明記



- ROSに対して手を加えている
 - 通信効率化の追加モジュール開発
- SDK (ソフトウェア開発キット)

- 将来的にユーザに提供
- ROSベースSDKの予定

```

=====
ros-gazebo
/*****
* Software License Agreement (BSD License)
*
* Copyright (c) 2008, Willow Garage, Inc.
* All rights reserved.
*
* Redistribution and use in source and binary forms, with or without
* modification, are permitted provided that the following conditions
* are met:
*
* * Redistributions of source code must retain the above copyright
*   notice, this list of conditions and the following disclaimer.
* * Redistributions in binary form must reproduce the above
*   copyright notice, this list of conditions and the following
*   disclaimer, in the documentation and/or other materials provided
*   with the distribution.
* * Neither the name of the Willow Garage nor the names of its
*   contributors may be used to endorse or promote products derived
*   from this software without specific prior written permission.
*
* THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
* "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
* LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS
* FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE
* COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT,
* INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING,
* BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES;
* LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER
* CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT
* LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN
* ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE
* POSSIBILITY OF SUCH DAMAGE.
=====
ros-container-network-setter
/*****
* Software License Agreement (BSD License)
*
* Copyright (c) 2008, Willow Garage, Inc.
* All rights reserved.
*
* Redistribution and use in source and binary forms, with or without
* modification, are permitted provided that the following conditions
* are met:
*
* * Redistributions of source code must retain the above copyright
*   notice, this list of conditions and the following disclaimer.
* * Redistributions in binary form must reproduce the above
*   copyright notice, this list of conditions and the following
*   disclaimer, in the documentation and/or other materials provided
*   with the distribution.
* * Neither the name of the Willow Garage nor the names of its
*   contributors may be used to endorse or promote products derived
*   from this software without specific prior written permission.
*
* THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
* "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
  
```

<http://oss.sony.net/License/ERS-1000/>

ROSに関連する情報は以下の場所にまとまっています。

- ROS Wiki: <http://wiki.ros.org/ja>
 - 様々な情報がwiki上に集約されている
- ROS Discourse: <https://discourse.ros.org/>
 - 掲示板、メールでも読むことができる
- Qiita: <https://qiita.com/search?q=ROS>
 - 様々な人が解説、導入記録等をあげている。体系的ではないが、自分と同じトラブルの場合にはたいへん役に立つ。
- Slack: ROS Community, ROS Japan UG などがあります。
- その他、検索すると様々な情報がでてきます。特に、エラーが起こった場合は、まず **エラーメッセージ** をそのまま検索窓に入れて検索してみましょう。

- DISTRO (=Distribution)
 - 配布パッケージのこと
 - バージョンごとに名前がついている。(亀に関係がある)
 - DISTRO nameは覚えておいた方がよい。

現行Distro: <http://wiki.ros.org/Distributions>

- Kinetic (Kinetic-Kame)
 - Ubuntu 16.04対応版
 - 現行で一番古いが、まだユーザが多い
 - 2021年4月まで
- (Lunar Loggerhead)
 - 2019年5月サポート終了
- Melodic (Melodic Morenia (ビルマメダマガメ))
 - Ubuntu 18.04対応版
 - 次第にこちらに移りつつある
 - 2023年5月まで
- Noetic (Noetic Ninjemys (メイオラン科のカメ))
 - 推奨版
 - Ubuntu 20.04対応版
 - 2025年3月まで

Distro	Release date	Poster	Turtle, turtle in tutorial	EOL date
ROS Noetic Ninjemys (Recommended)	May 23rd, 2020			May, 2025 (Focal EOL)
ROS Melodic Morenia	May 23rd, 2018			May, 2023 (Bionic EOL)
ROS Lunar Loggerhead	May 23rd, 2017			May, 2019
ROS Kinetic Kame	May 23rd, 2016			April, 2021 (Xenial EOL)
ROS Jade Turtle	May 23rd, 2015			May, 2017
ROS Indigo Igloo	July 22nd, 2014			April, 2019 (Trusty EOL)
ROS Hydro Medusa	September 4th, 2013			May, 2015
ROS Groovy Galapagos	December 31, 2012			July, 2014
ROS Fuerte Turtle	April 23, 2012			-
ROS Electric Emys	August 30, 2011			-
ROS Diamondback	March 2, 2011			-
ROS C Turtle	August 2, 2010			-
ROS Box Turtle	March 2, 2010			-

カメにまつわるコードネーム
頭文字はアルファベット順

【ユースケースの変化】

- 複数のロボット
- 組込みCPU
- リアルタイム
- 理想的でない通信環境
- 製品向け使用
- あらかじめ規定されたパターンにのっとった構造化したシステム構成



【ROS1では】

- 単体のロボット
- 強力なCPU
- 非リアルタイム
- 途切れない通信環境
- 研究向け
- 自由な枠組み（main関数）

【新たな技術】

- Zeroconf (avahi, bonjour, UPnP等)
- Protocol Buffers
- ZeroMQ (and the other MQs)
- Redis(次世代高速key-valueデータストア)
- WebSockets
- **DDS (Data Distribution Service).**



ROSを大幅に改良したROS2への移行を発表
(ROSとの互換性はない)

http://design.ros2.org/articles/why_ros2.html

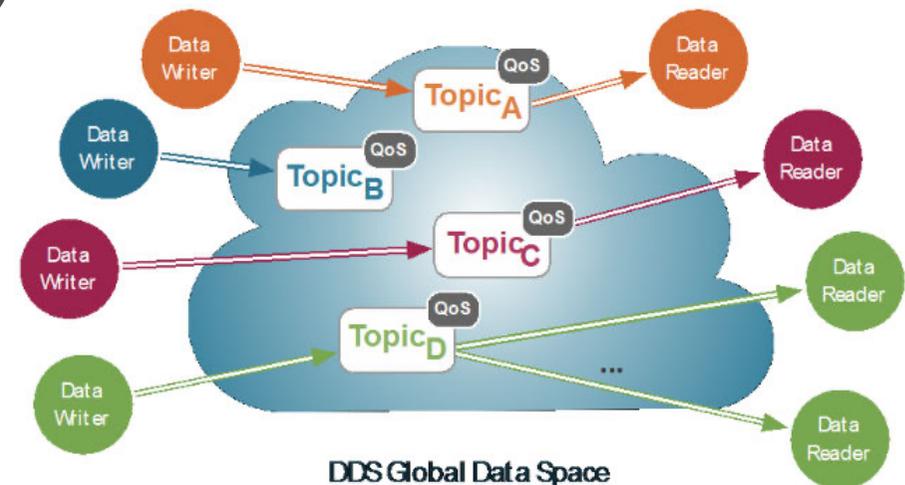
- 通信：DDS(OMG標準のpub/sub通信ミドルウェア)
 - ミドルウェア層を作製し複数のDDS実装（製品も含む）を使うように
 - 単一障害点がない（ROS1ではmasterが落ちると×）
 - QoS制御が可能に(History, Depth, Reliability, Durability)
- コンポーネントモデルを導入
- 複数のOSに対応（Windows、MacOS）
- ROS1とはブリッジで通信
 - 直接は通信できない、互換性なし
- セキュリティ対応（DDSセキュリティを利用）
- リアルタイム実行可能（Linuxのみ）
- 組み込み対応

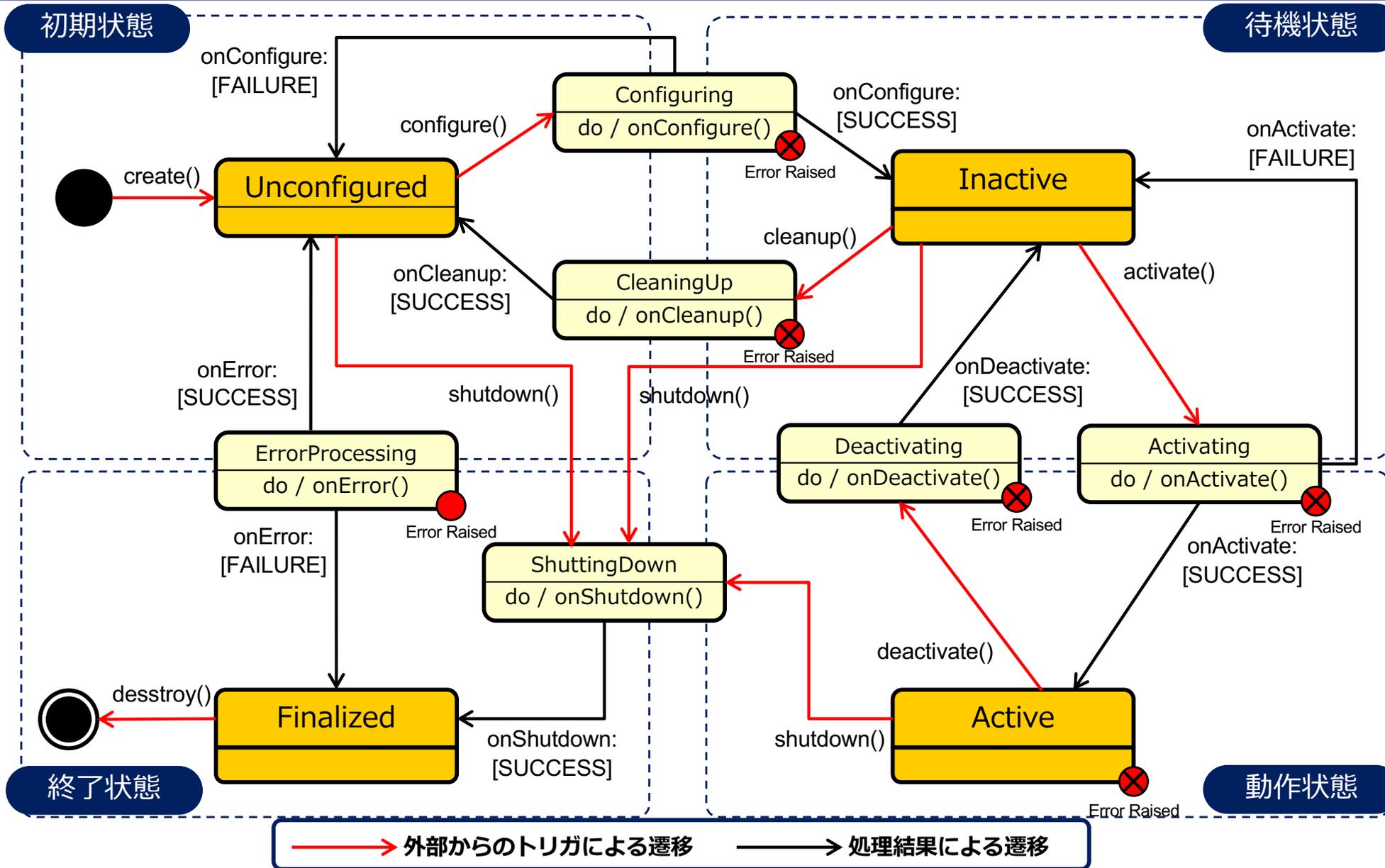
http://design.ros2.org/articles/ros_middleware_interface.html

ROS1→ROS2へ

- NASAの仕事を請け負った時に、独自形式のROS messageはNASAでは使えないから、プロトタイプをROSで実装後にすべて作り直した
- NASAでは何らかの標準に準拠したものでないと使えない。その時は結局DDSを使用した。
- それ以外にもROS1では、1ノード1プロセス、コンポーネントモデルがないので、モデルベース開発にならない、ROS masterがSPOFになっているなど不都合な点が多々ある
- それ故、ROS2ではこれらの問題点を克服するため全く新しい実装にする予定。

- OMG(Object Management Group)で標準化されたpub-sub型データバスミドルウェア
- Pub/sub型通信
 - トピックが同じ送受信者間でデータが配信される仕組み
- SPOF (単一障害点) がない discoveryメカニズム
- OMGのミドルウェアTF(MARS)で最もアクティブに活動しているTFで策定
 - 関連標準仕様は10程度



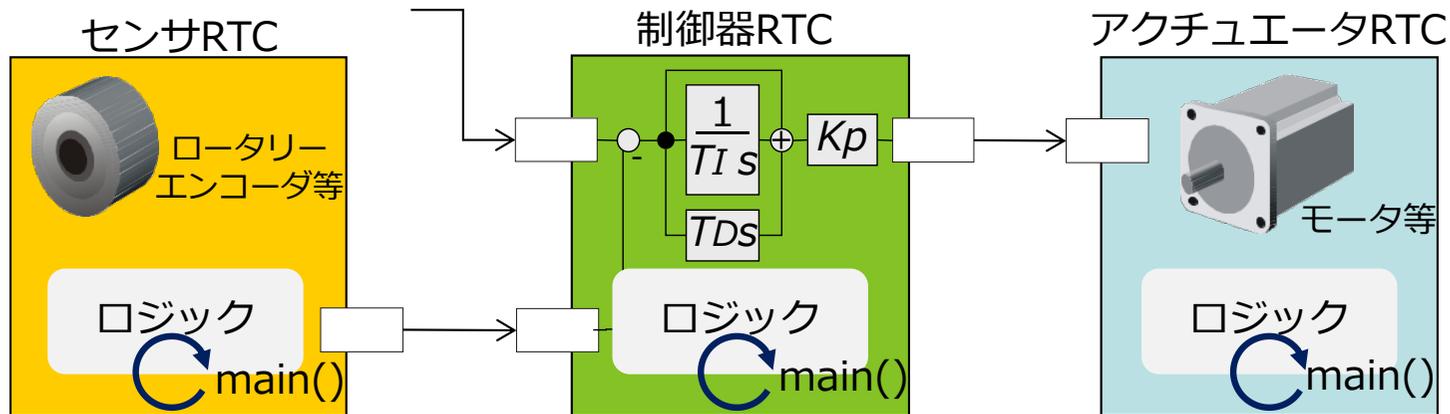


過渡状態を状態とみなしている以外はRTCをほぼ同じ
 エラー処理が明確に、終了処理が一体になっている

ROS1

モジュール
(ノード)
の粒度

大

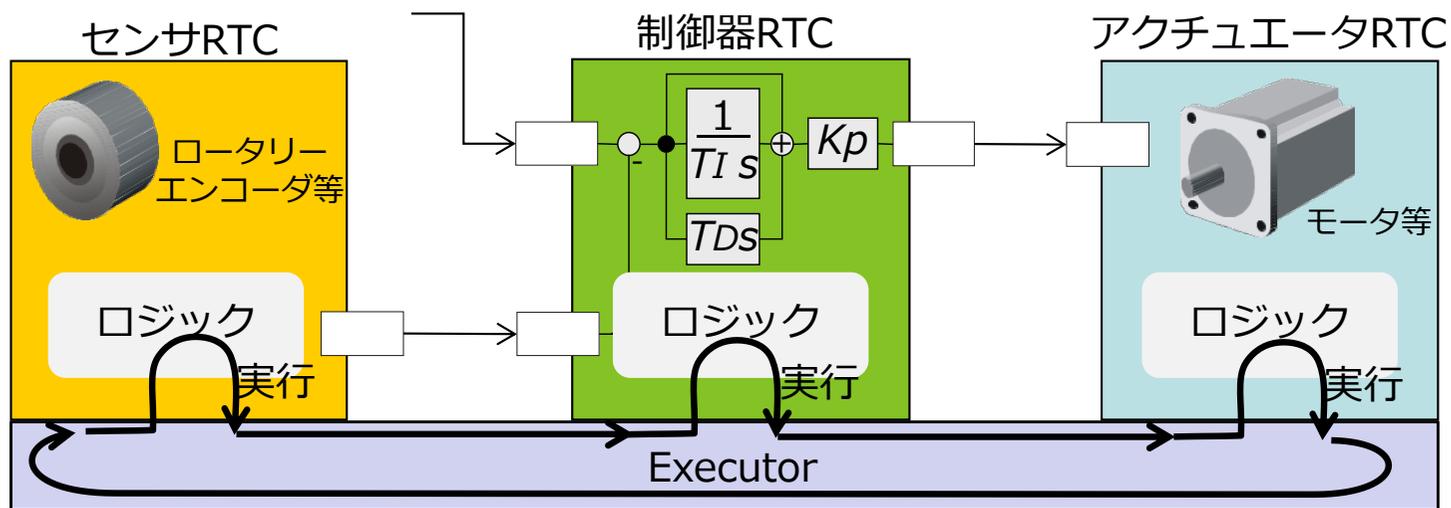


ノードの実行タイミング・順序は制御できない
複数のノードを密結合してリアルタイムシステムを構成できない

ROS2

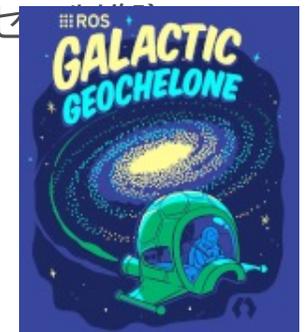
モジュール
(ノード)
の粒度

小



ノードの実行タイミング・順序をExecutorで決定できる
複数のノードを密結合してリアルタイムシステム化可能

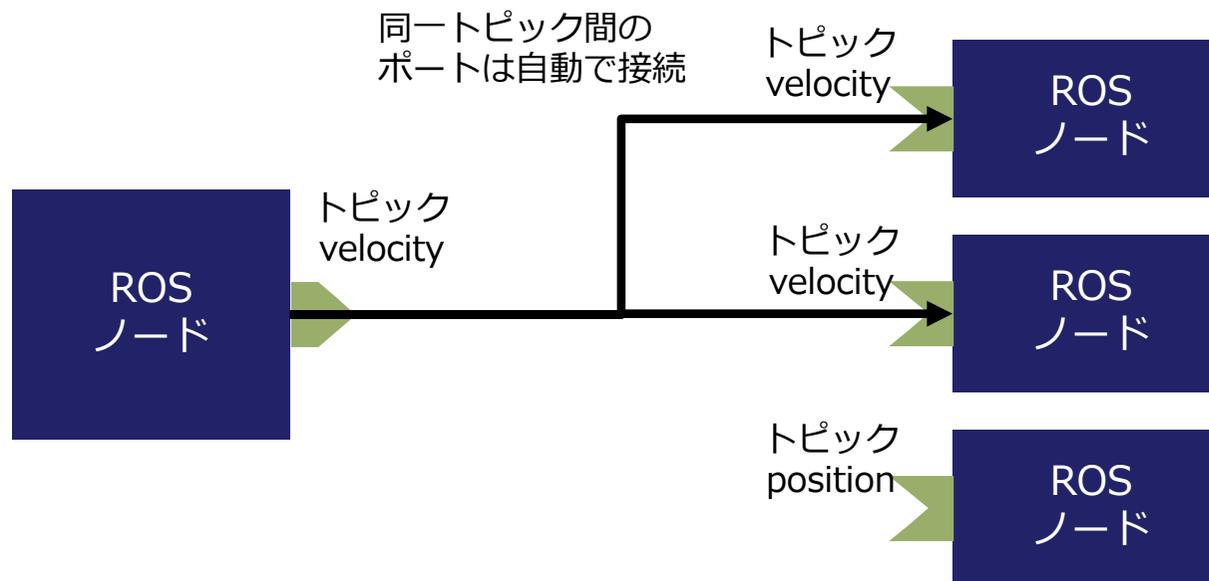
- Ardent Apalone (2017.12)
 - コードネーム : Ardent(熱烈) Apalone (アパロンカメ、北米原産のスッポン科のカメ)
- Bouncy Bolson (2018.6)
 - コードネーム : Bouncy (弾む) Bolson (メキシコゴファーガメ、Bolson tortoise)
- Crystal Clemmys (2018.12)
 - コードネーム : Crystal(透明な) Clemmys (キボシイシガメ Clemmys guttata)
- Dashing Diamemata (2019.5→2021.5)
 - Dashing(走る) Diademata (カムリヤマガメ (Rhinoclemmys diademata))
 - Componentノードの推奨、OMG IDL 4.2利用開始, MoveIt!2 alpha
- Eloquent Elusor
- Rolling Ridley: スナップショット開発版
- Foxy Fitzroy (2020.6→2023.5)
 - Foxy (狐のような) Fitzroy (ハヤセガメ (Fitzroy River turtle))
 - バグ修正、雑多なAPI改善, Security Enclave機能 (同一IDのノードのみのアクセス)
- Galactic Geochelone (2021.5→2022.11)
 - Galactic (銀河の) Geochelone (ゾウガメ)
 - デフォルトのDDSがGalactic Geochelone (ADLINK社製) へ変更
 - Nav2 (navigation stack for ROS2)の強化
- Humble Hawksbill (2022.5?)



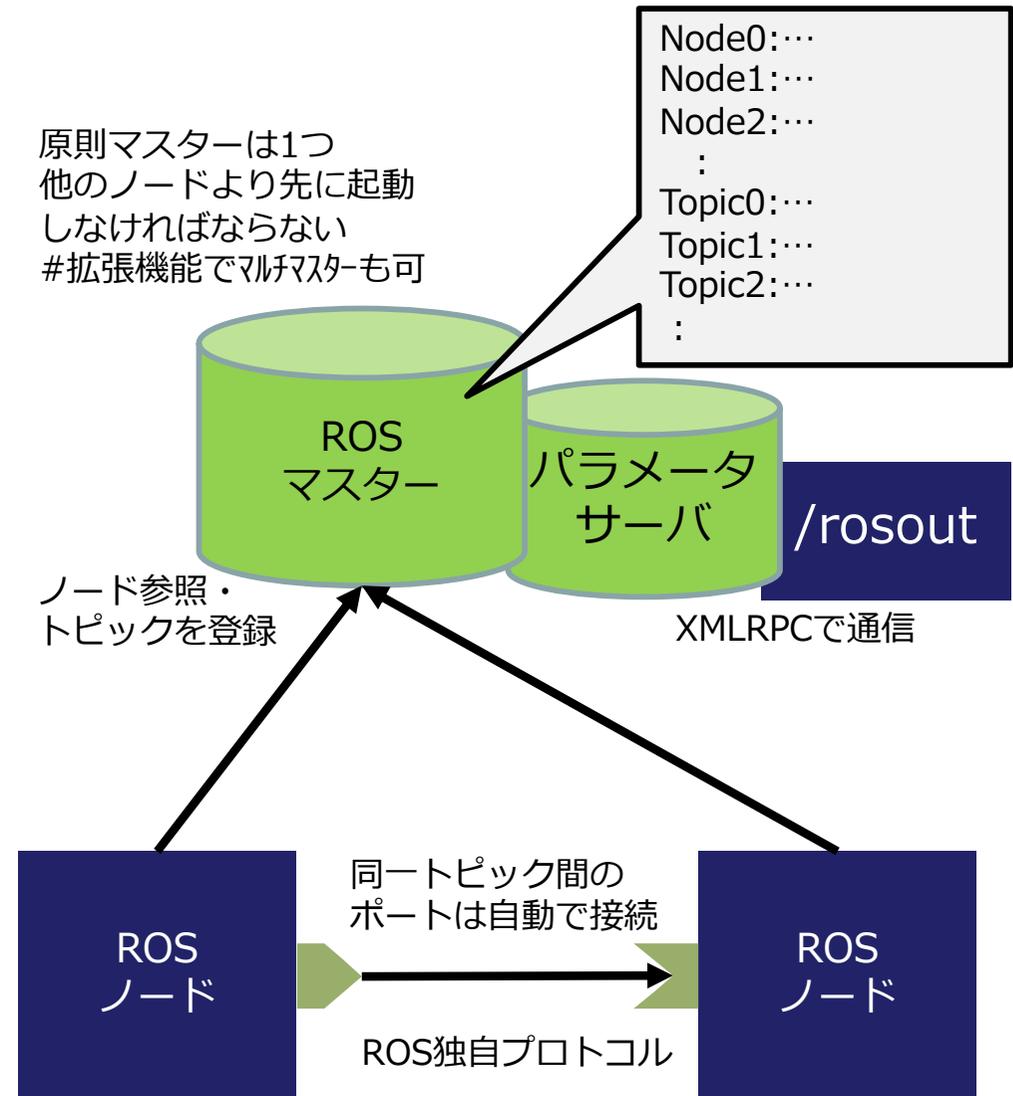
ROS1同様にカメにまつわるコードネーム

ROSの基礎

- ノード
 - モジュールプログラム、通常は実行可能な一つのプログラムとして、ソースおよびバイナリが提供される
- パッケージ
 - ノードや設定ファイル、コンパイル方法などをまとめたもの
- メッセージ
 - ノード間でやりとりするデータ
- トピック
 - ノード間でやりとりするデータのラベル、同じトピック名を持つ出力と入力データがやり取りされる



- マスター**：ノードの参照やトピックを保持するネームサービス
 - システム全体で原則一つ→SPOF (Single Point of Failure)
 - 他のノードより先に起動しなければならない
- パラメータサーバ**：ノードのパラメータを保持するデータベース
 - マスター内で動作、ノードからはXMLRPCでアクセス
- rosout**：ノードに対してstdout, stderrのような役割を果たす

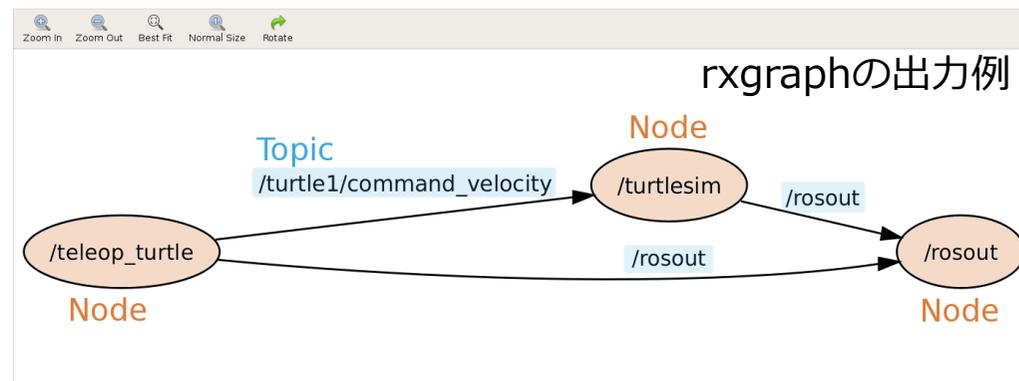


- **ワークスペース作成 :**
 - 自作パッケージを置く場所
 - 通常ホームdir直下の “~/catkin_ws” とすることが多い
 - 自分で作る必要あり
 - mkdir catkin_ws
 - cd catkin_ws ; catkin build ← WS初期化
- **パッケージ作成 :**
 - ワークスペースいかにパッケージを作成する
 - catkin create pkg my_pkg <opt>
 - パッケージ名は小文字と“_”のみ
- **ビルド:**
 - catkin build でビルド
 - cd ~/catkin_ws
 - catkin build <package name>

Catkinのワークスペース構成

```
~/catkin_ws/ ←このディレクトリでビルド
+ .catkin_workspace
+ build/
+ devel/
+ src/
  + CMakeLists.txt ←最上位のcmake file
  + PackageA/
    + CMakeLists.txt
    + package.xml ←マニフェスト
    + include
      + パッケージ名
      + インクルードファイル
  + src/ ← ソースファイルディレクトリ
    + ソースファイル
```

- グラフィソース名
 - ノード名、トピック名、メッセージタイプ名、サービス名などに付与するユニークな名前
 - “/” から始まり、“/”で区切られて階層化される
 - /<name0>/<name1>/<name2>/…
- 相対名
 - デフォルトの名前空間が指定されている場合 <名前空間名>/<リソース名> で指定可能
 - 先頭に“/” がつかないことで区別される
 - 環境変数 ROS_NAMESPACE で指定したり、launchファイルで <node ns=“名前空間” /> で指定することができる
- プライベート名
 - “~” から始まる名前、ノード内でプライベートな名前空間
- アノニマス名
 - 多数のノード名を自動でつけたい時などに使用
 - Ros::init()の引数にinit_options::AnonymousName を指定することで、自動的にノード type名+時刻で一意的な名前を自動でつけてくれる



- ROSノードをビルドするツール
 - ROSは独自のビルド・パッケージツールを提供している。ROSの特徴の一つ。様々なソフトウェアを効率よく管理・ビルド・配布することができる。
- rosbuidl
 - 廃止された最初のROS用ビルドツール
- **catkin** (ヤナギなどの尾状花序)
 - **現行のROS用ビルドツール**

ROS2以降

- **ament** (ヤナギなどの尾状花序)
 - 最初のROS2用ビルドツール
 - 現在は表向きは次のcolconが利用されているが、内部的にはamentが使われている模様
- **colcon** (collective construction)
 - ROS2用ビルドツール
 - メタビルドツール、内部的に用途ごとに適切なビルドツールを呼び出す
 - <https://colcon.readthedocs.io/en/released/>

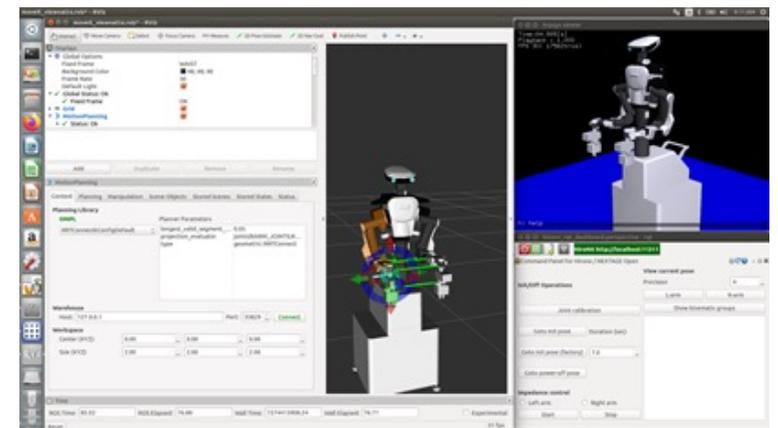
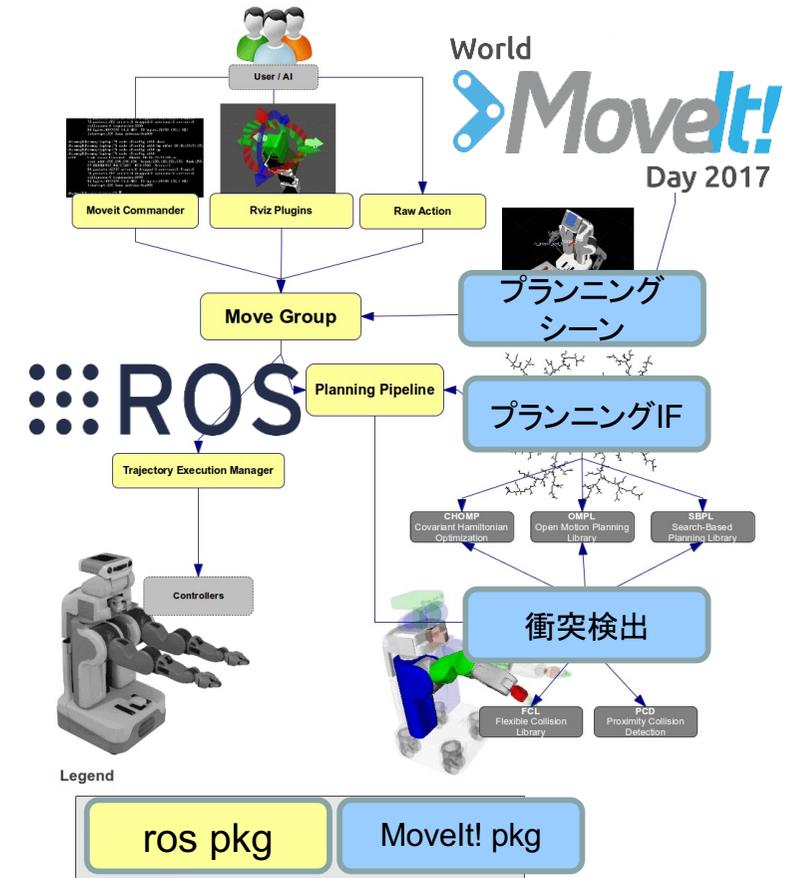
- ROSノード（群）を起動するツール
 - ROSは独自のビルド・パッケージツールを提供している。ROSの特徴の一つ。様々なソフトウェアを効率よく管理・ビルド・配布することができる。
- rosruln
 - ROSノード（1個）を起動するコマンド
- roslaunch
 - ROSノード群（複数）を起動するコマンド
 - Roslaunch <package> <launch_file>

ROS launchファイル例

```
<launch>
  <arg name="mode" default="true">
  <group if="$(arg mode)">
    <node name="インスタンス1" pkg="パッケージ名" type="ノード名" output="出力先"/>
  </group>
  <node name="インスタンス名2" pkg="パッケージ名" type="ノード名" output="出力先">
    <remap from="改名前トピック名" to="改名後トピック名"/>
    <param name="content" value="hogehoge"/>
  </node>
</launch>
```

- XMLファイルとして記述
 - <launch>タグで囲む
- <node>タグ：起動するノードを指定
 - Name: インスタンス名、同一ノード名でも別名をつけて複数起動させることができる
 - Pkg: パッケージ名
 - Type: ノード名
 - Output: 出力先、screen で標準出力に出力される、複数のノードの出力を標準出力に出力可能
- <remap>タグ：トピック名を変更 from, toで改名前後のトピック名指定
- <arg>タグ：roslaunch起動時にとる引数を宣言可能
 - nameで引数名、defaultでデフォルト値を指定
- <group>タグ：グルーピング
 - If文でargの引数をとることができる
 - 上の例では、mode=falseを指定すると“インスタンス名1”のノードは起動しない
 - Ex. roslaunch <パッケージ名> <launchファイル>.launch mode:=false ← インスタンス名1を起動させない
- <param >タグ：パラメータ指定
 - コード内であらかじめ定義されているパラメータを起動時に指定する
 - Ex. contentという名前のパラメータ(string型) に “hogehoge” を代入

- MoveIt!
 - マニピュレータ（ロボットアーム）のためのモーションプランニングのフレームワーク・ライブラリ
 - 目的の手先位置・姿勢を与えると、マニピュレータの手先の現在位置・姿勢から目的位置・姿勢までの経路を自動的に計算する
 - 計算結果に基づき、実際にマニピュレータを制御し移動させる
 - 障害物の情報があらかじめ与えられている場合、それらにぶつからない経路を計算することができる。
- ROS、rviz
 - MoveIt!には、ROSでMoveIt!利用するのに便利なパッケージ・ノード群が提供されている。
 - ROSの可視化ツールrvizと連携し、ロボットの実機がなくても、プランニング結果をアニメーションで表示することが可能
 - rviz上からGUIでMoveIt!でロボットの操作が可能
- MoveIt! Commander
 - MoveIt!をコマンドラインから使用するインターフェース



- 自前主義はやめよう！！
 - 書きたてのコードより、いろいろな人に何万回も実行されたコードのほうが動くコードである！！
 - 自分にとって本質的でない部分は任せて、本当にやりたい部分・やるべき部分のコードを書こう！！
 - 誰かがリリースしたプログラムは一度は動いたことがあるプログラムである！！
 - 人のコードを読むのが面倒だからと捨ててしまうのはもったいない！！
- オープンソースにコミットしよう！！
 - 臆せずMLやフォーラムで質問しよう！！
 - どんなに初歩的な質問でも他の人にとっては価値ある情報である。
 - 要望を積極的にあげよう！！
 - できればデバッグしてパッチを送ろう！