

2019年12月14日(土)
SI2019 RTミドルウェア講習会



第1部：OpenRTM-aistおよび RTコンポーネントプログラミングの概要

国立研究開発法人産業技術総合研究所
ロボットイノベーション研究センター
ロボットソフトウェアプラットフォーム研究チーム長
安藤 慶昭

- RTミドルウェアの概要
 - 基本概念
- OpenRTM-aist-1.2の新機能と開発ロードマップ
- ROSとの比較、動向
- プラットフォームロボットプロジェクト
- RTMコミュニティ活動
- まとめ

RTミドルウェアとは？

RTとは?

- RT = Robot Technology cf. IT
 - ≠Real-time
 - 単体のロボットだけでなく、さまざまなロボット技術に基づく機能要素をも含む (センサ、アクチュエータ, 制御スキーム、アルゴリズム、etc….)

産総研版RTミドルウェア

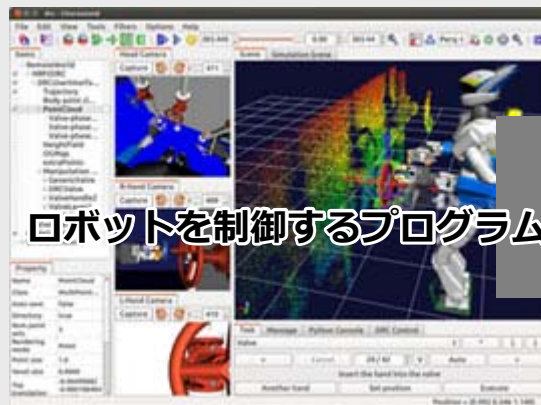
OpenRTM-aist

- RT-Middleware (RTM)
 - RT要素のインテグレーションのためのミドルウェア
- RT-Component (RTC)
 - RT-Middlewareにおけるソフトウェアの基本単位

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

従来のシステムでは…



ロボットを制御するプログラム

Controller software

Controller



Robot Arm Control software

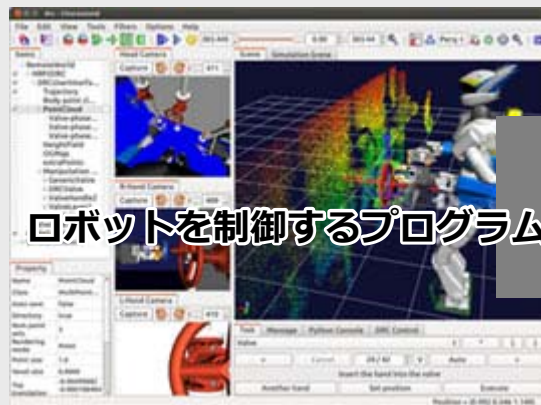


制御対象のロボット

Robot Arm1

互換性のあるインターフェース同士は接続可能

従来のシステムでは…



ロボットを制御するプログラム

Controller

Controller software

Humanoid's Arm Control software



制御対象のロボット

Robot Arm2



制御対象のロボット

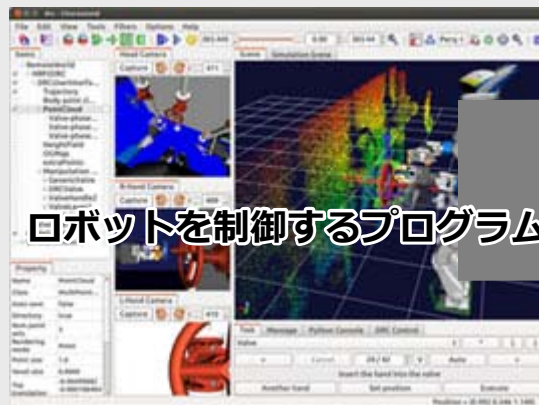
Robot Arm2

ロボットによって、インターフェースは色々
互換性が無ければつながらない

Robot Arm Control software

RTミドルウェアでは…

RTミドルウェアは別々に作られたソフトウェアモジュール同士を繋ぐための共通インターフェースを提供



ロボットを制御するプログラム

Controller software

Controller

ソフトウェアの再利用性の向上
RTシステム構築が容易になる

Arm A
Control software



compatible
arm interfaces

Arm B
Control software



制御対象のロボット

Robot Arm2



制御対象のロボット

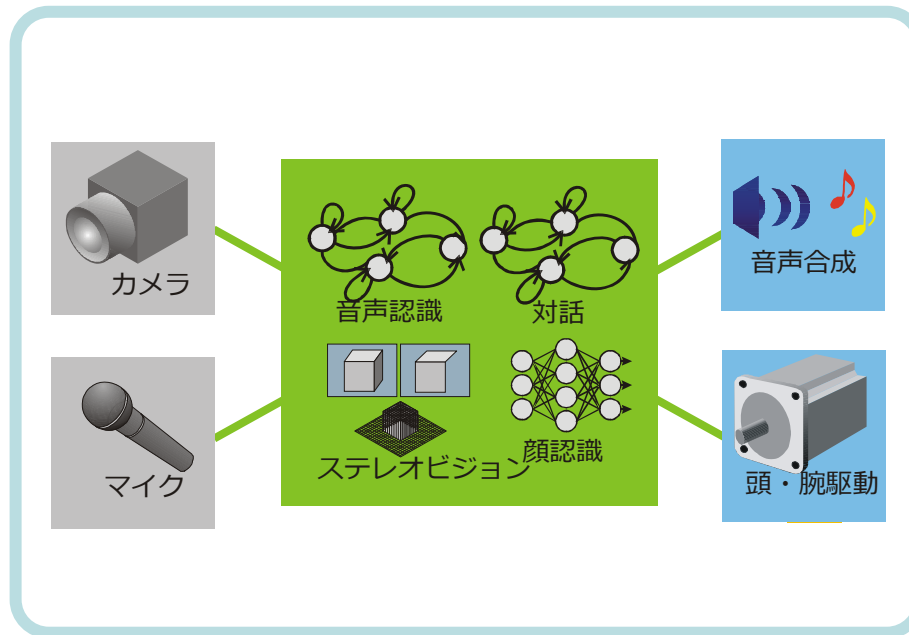
Robot Arm1

ロボットソフトウェア開発の方向

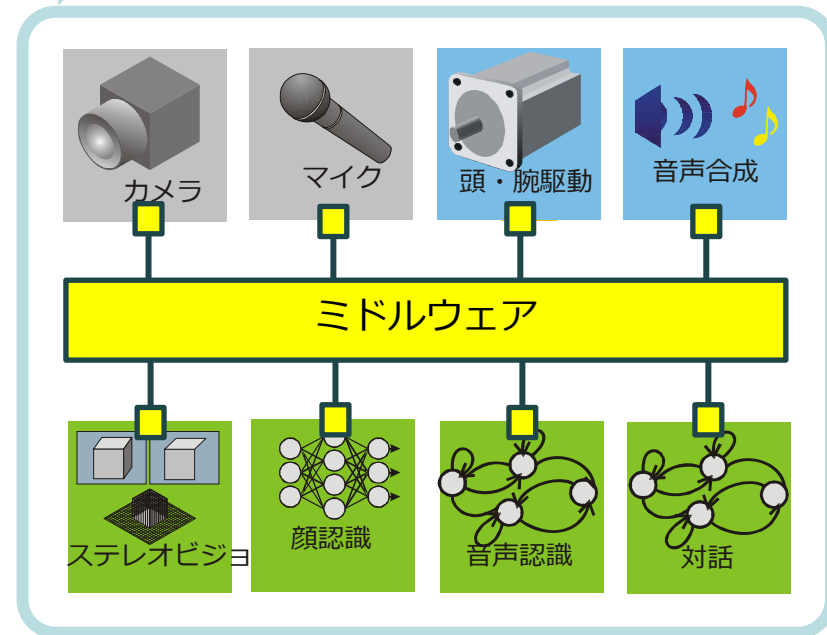
従来型開発



コンポーネント指向開発



- ✓ 様々な機能を融合的に設計
- ✓ 実行時の効率は高いが、柔軟性に欠ける
- ✓ システムが複雑化してくると開発が困難に



- ✓ 大規模複雑な機能の分割・統合
- ✓ 開発・保守効率化（機能の再利用等）
- ✓ システムの柔軟性向上

モジュール化のメリット

- 再利用性の向上
 - 同じコンポーネントをいろいろなシステムに使いまわせる
- 選択肢の多様化
 - 同じ機能を持つ複数のモジュールを試すことができる
- 柔軟性の向上
 - モジュール接続構成かえるだけで様々なシステムを構築できる
- 信頼性の向上
 - モジュール単位でテスト可能なため信頼性が向上する
- 堅牢性の向上
 - システムがモジュールで分割されているので、一つの問題が全体に波及しにくい

RTコンポーネント化のメリット

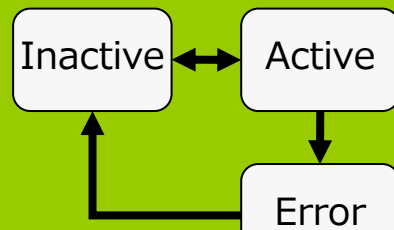
モジュール化のメリットに加えて

- ソフトウェアパターンを提供
 - ロボットに特有のソフトウェアパターンを提供することで、体系的なシステム構築が可能
- フレームワークの提供
 - フレームワークが提供されているので、コアのロジックに集中できる
- 分散ミドルウェア
 - ロボット体内LANやネットワークロボットなど、分散システムを容易に構築可能

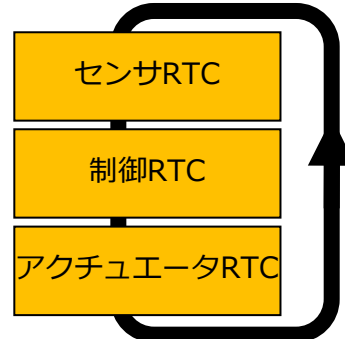
RTコンポーネントの主な機能

アクティビティ・実行コンテキスト

共通の状態遷移



複合実行

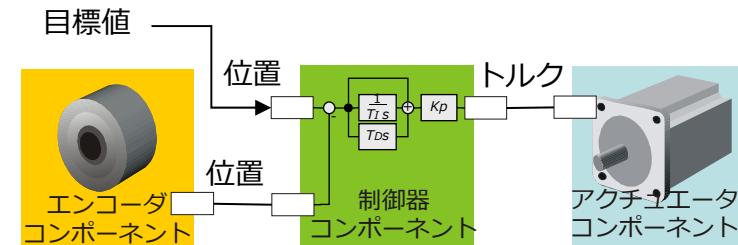


ライフサイクルの管理・コアロジックの実行

データポート

- データ指向ポート
- 連続的なデータの送受信
- 動的な接続・切断

サーボの例



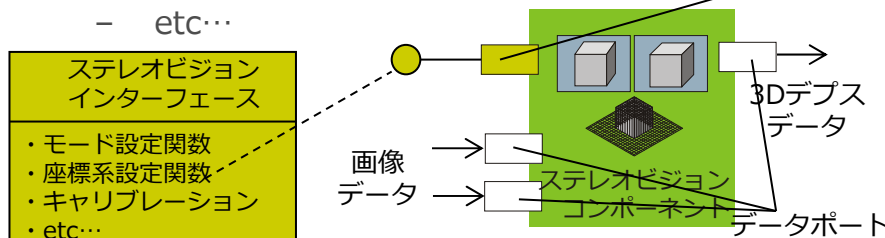
データ指向通信機能

サービスポート

- 定義可能なインターフェースを持つ
- 内部の詳細な機能にアクセス
 - パラメータ取得・設定
 - モード切替
 - etc...

ステレオビジョンの例

サービスポート

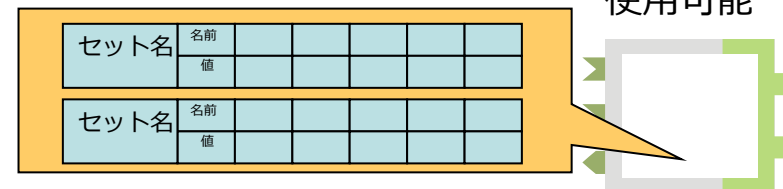


サービス指向相互作用機能

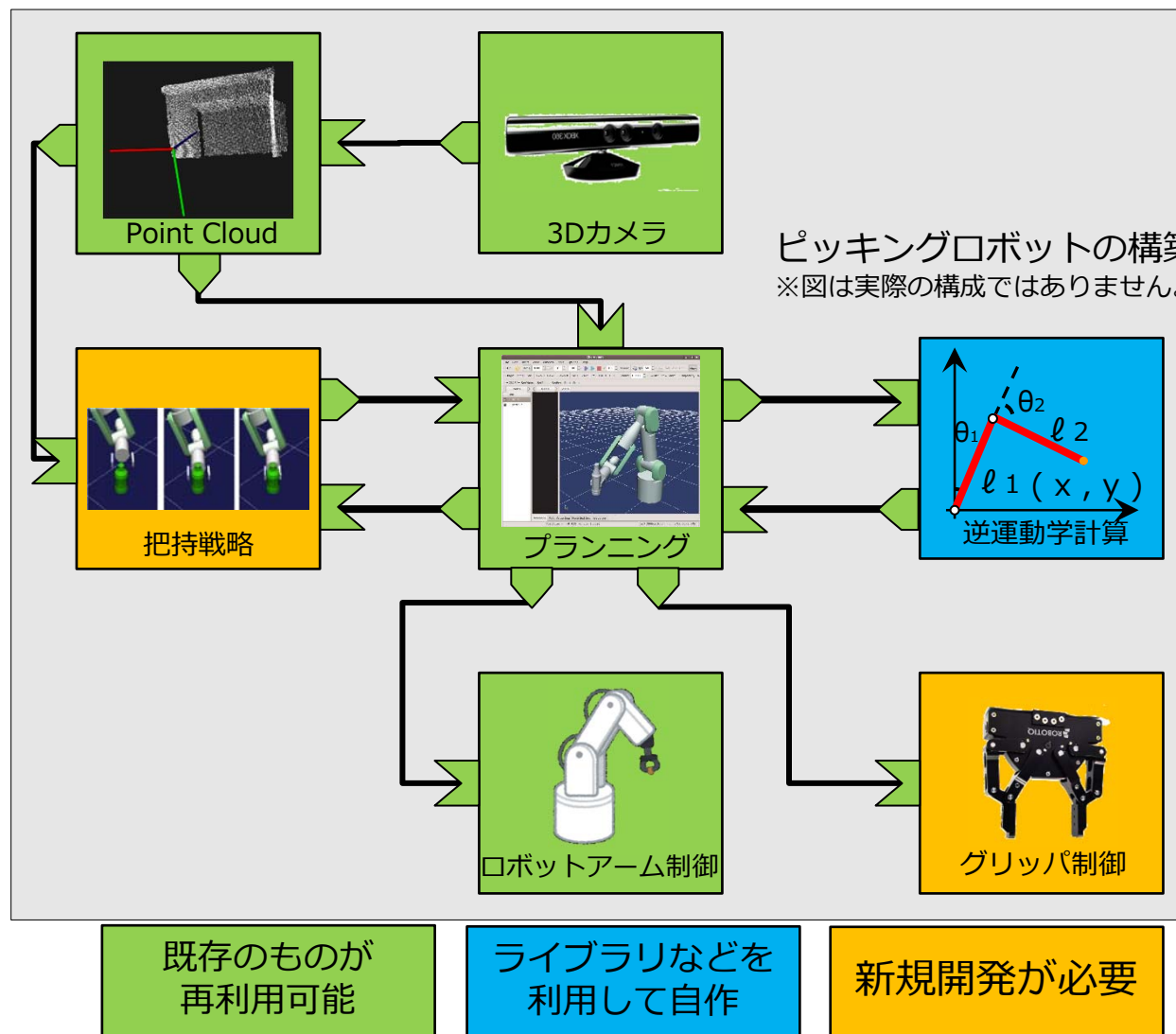
コンフィギュレーション

- パラメータを保持する仕組み
- いくつかのセットを保持可能
- 実行時に動的に変更可能

複数のセットを
動作時に
切り替えて
使用可能



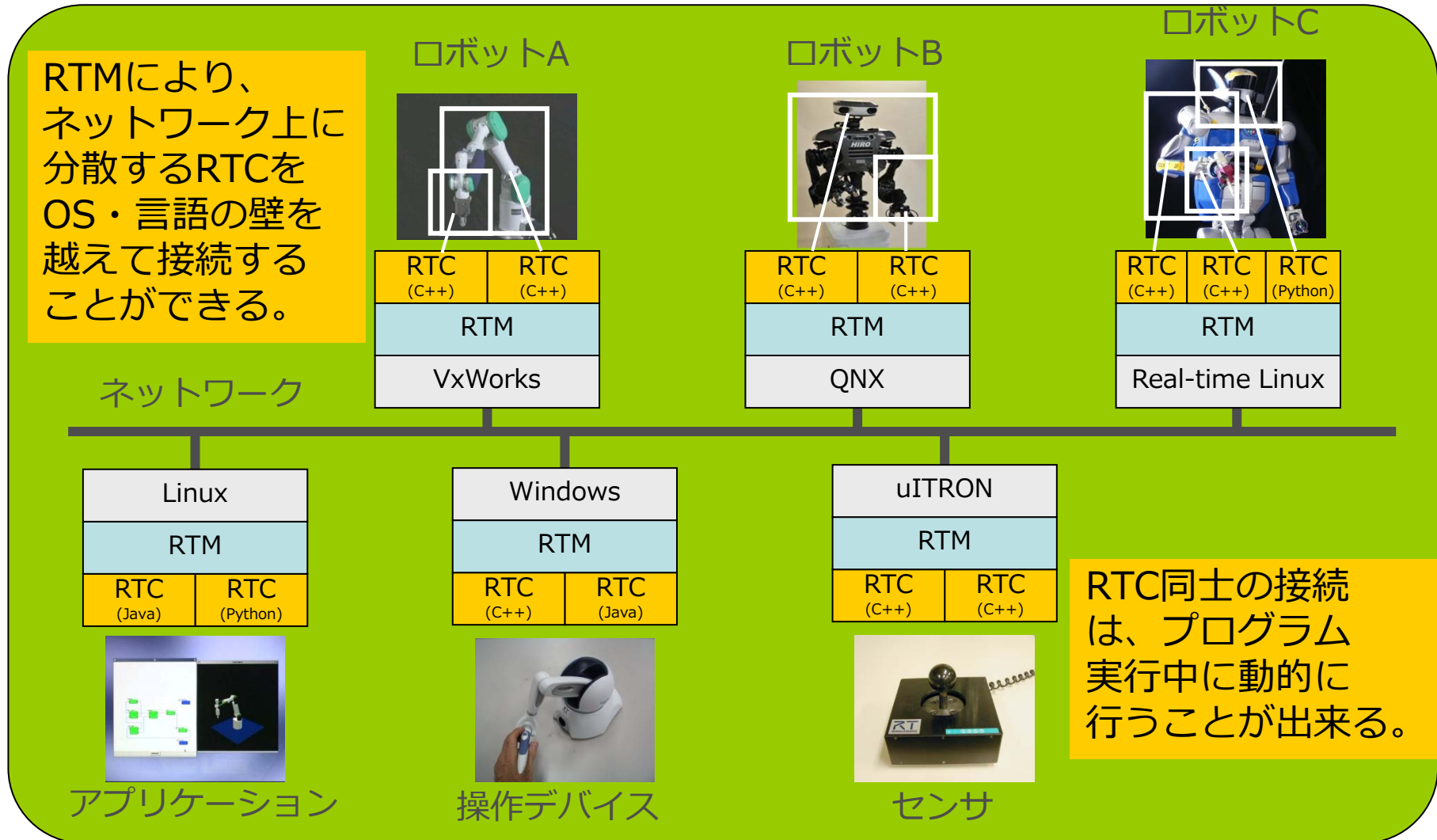
ミドルウェアを利用した開発の利点



ミドルウェアを利用すると、**既存のモジュール**が利用できる

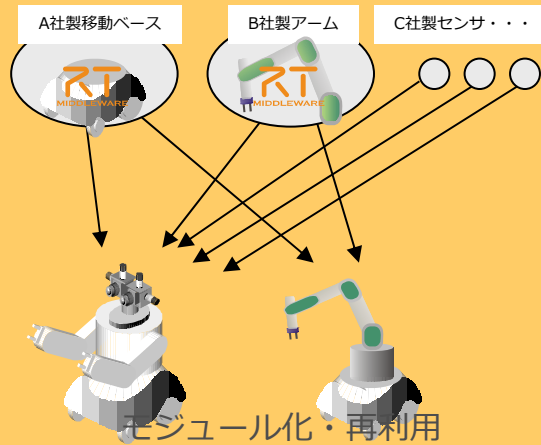
開発するときに**新規に作らなければならない部分**は少なくて済む

RTミドルウェアによる分散システム



RTミドルウェアの目的 モジュール化による問題解決

コストの問題



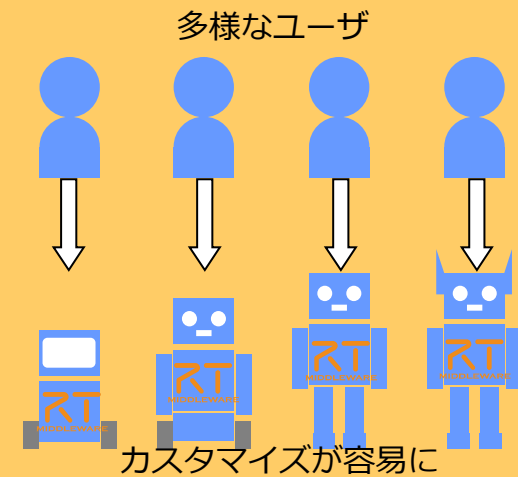
ロボットの低コスト化

技術の問題



最新技術を利用可能

ニーズの問題



多様なニーズに対応

ロボットシステムインテグレーションによるイノベーション

実用例・製品化例



HRPシリーズ: 川田工業、AIST

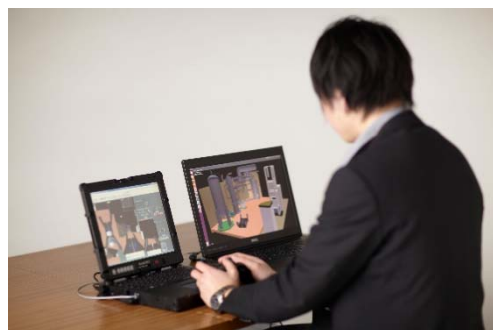


S-ONE : SCHAFT



DAQ-Middleware: KEK/J-PARC

KEK: High Energy Accelerator Research Organization
J-PARC: Japan Proton Accelerator Research Complex



災害対応ロボット操縦シミュレータ:
NEDO/千葉工大



HIRO, NEXTAGE open: Kawada Robotics



RAPUDA : Life Robotics



ビュートロバーRTC/RTC-BT(VSTONE)



OROCHI (アールティ)



新日本電工他: Mobile SEM

RTミドルウェアは国際標準

Date: September 2012



Robotic Technology Component (RTC)

Version 1.1

Normative reference: <http://www.omg.org/spec/RTC/1.1>
 Machine consumable files: <http://www.omg.org/spec/RTC/20111205/>
 Normative:
<http://www.omg.org/spec/RTC/20111205/rtc.xml>
<http://www.omg.org/spec/RTC/20111205/rtc.h>
<http://www.omg.org/spec/RTC/20111205/rtc.idl>
 Non-normative:
<http://www.omg.org/spec/RTC/20111205/rtc.eap>

標準化履歴

- 2005年9月
Request for Proposal 発行(標準化開始)
- 2006年9月
OMGで承認、事実上の国際標準獲得
- 2008年4月
OMG RTC標準仕様 ver.1.0公式リリース
- 2012年9月
ver. 1.1改定
- 2015年9月
FSM4RTC(FSM型RTCとデータポート標準) 採択

OMG国際標準

標準化組織で手続きに沿って策定

- 1組織では勝手に改変できない安心感
- 多くの互換実装ができつつある
- 競争と相互運用性が促進される

RTミドルウェア互換実装は10種類以上

名称	ベンダ	特徴	互換性
OpenRTM-aist	産総研	NEDO PJで開発。参照実装。	---
HRTM	ホンダ	アシモはHRTMへ移行中	◎
OpenRTM.NET	セック	.NET(C#, VB, C++/CLI, F#, etc..)	◎
RTM on Android	セック	Android版RTミドルウェア	◎
RTC-Lite	産総研	PIC, dsPIC上の実装	○
Mini/MicorRTC	SEC	NEDOオープンイノベーションPJで開発	○
RTMSafety	SEC/AIST	NEDO知能化PJで開発・機能安全認証取得	○
RTC CANOpen	SIT, CiA	CAN業界RTM標準	○
PALRO	富士ソフト	小型ヒューマノイドのためのC++ PSM 実装	×
OPRoS	ETRI	韓国国家プロジェクトでの実装	×
GostaiRTC	GOSTAI, THALES	ロボット言語上で動作するC++ PSM 実装	×

特定のベンダが撤退しても
ユーザは使い続けることが可能

ROSとの比較、動向

ロボットOS：現状と課題

ロボットOS勢力図



日本発ロボットOS (NEDO・産総研)
2002年からNEDO PJで開発
OMG国際標準を獲得し10以上の
実装が様々なベンダからリリース



欧州発ミドルウェア



OROCOS



韓国



課題



HONDA



- ROSはデファクト標準であり、OSRFにより仕様を変えられてしまう恐れあり。
 - ⇒ バージョンが上がるごとに、ユーザが振り回される
 - ⇒ ROSは技術的に古く、次期バージョンアップで大幅に変わる可能性あり。→ROS2へ
- 国内ユーザと海外ユーザの利用OSの差 (国内：Windows, ITRON, 海外：Linux)
 - ⇒ Linuxのみ対応するROSと、様々なOSに対応するRTMの利用比率は半々程度
- 国内でのソフトウェアの再利用が進んでいない。
 - ⇒ オープンソース文化の欠如。
 - ⇒ NEDO知能化PJで蓄積したソフトウェア資産も再利用が進んでいない。

ROSとRTM

ROSの特長

- UNIX文化に根差した設計思想
 - Windows等Linux以外は原則サポートしない
 - ROS2ではLinux以外もサポート
- 独自のパッケージ管理システムを持つ
- ノード（コンポーネント）の質、量ともに豊富
 - OSRFが直接品質を管理しているノードが多数
- ユーザ数が多い
- メーリングリストなどの議論がオープンで活発
 - コアライブラリの仕様が固定しにくい
- 英語のドキュメントが豊富

OpenRTMの特長

- 対応OS・言語の種類が多い
 - Windows、UNIX、uITRON、T-Kernel、VxWorks、QNX
 - Windowsでネイティブ動作する
- 日本国内がメイン
 - 日本語ドキュメント・ML・講習会
 - ユーザ数が少ない
- GUIツールがあるので初心者向き
 - コマンドラインツールの種類は少ない
 - rtshell
- 仕様が標準化されている
 - OMGに参加すればだれでも変更可
 - サードパーティー実装が作りやすい
 - すでに10程度の実装あり
- コンポーネントモデルが明確
 - オブジェクト指向、UML・SysMLとの相性が良い
 - モデルベース開発
- IEC61508機能安全認証取得
 - RTMSafety

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。

ROS2

- 最新版：2016年12月 Beta版リリース
- 標準ミドルウェアの利用
 - 自前主義からの脱却（NASAや商用システムでは何らかの標準標準が求められるため）
 - 通信部分はOMG標準のDDS（Data Distribution Service、OMG規格）を採用
- コンポーネントモデルを導入することにした
 - RTMのように組み込み、性能を意識したアーキテクチャへ
- 対応OSの拡大
 - これまでは、ある特定のLinux（Ubuntu Linux）のみ
 - ROS2では、Windows、Macにも対応
 - ただし、リアルタイムOS（VxWorks、QNX等）への対応はなし

OMGによる通信規格DDSを採用



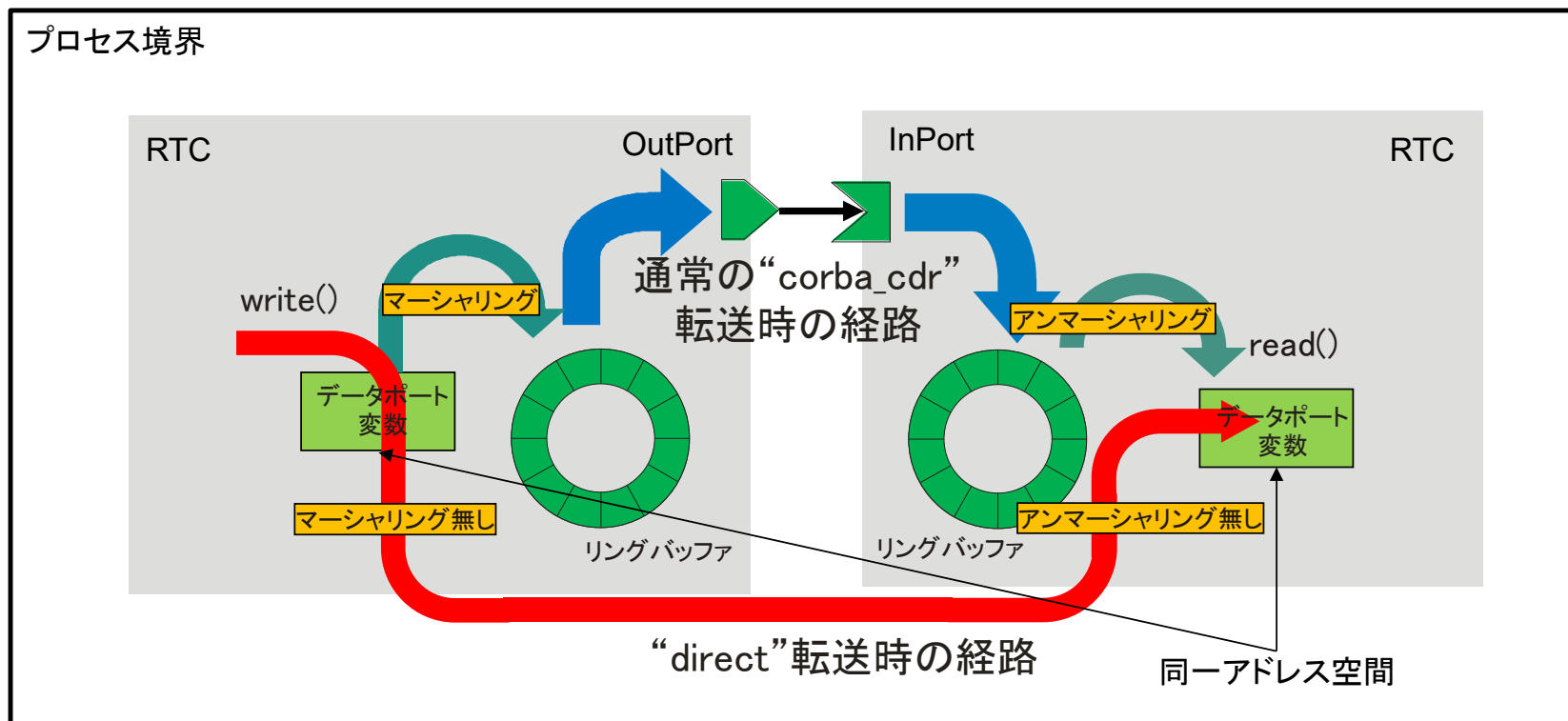
航空・軍事・医療・鉄道などで実績のある通信ミドルウェア標準

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

OpenRTM-aist-1.2の新機能 と今後の開発ロードマップ

- 2019年3月15日リリース
 - OpenRTM-aist (C++, Python, Java)
 - ミドルウェアライブラリ
 - OpenRTP-aist (RTSystemEditor, RTCBuilder)
 - RTC開発ツール、RTシステム開発ツール
- 新機能等
 - マネージャ機能の充実
 - データポートのパフォーマンス向上
 - CORBA呼び出し時のパフォーマンス向上 (omniidl shortcut)
 - RTCの命名・指定方法の拡張
 - ログ収集機構の拡張
 - コード品質の向上 (MISRA C++相当)
 - 雑多なバグフィックス

Direct接続

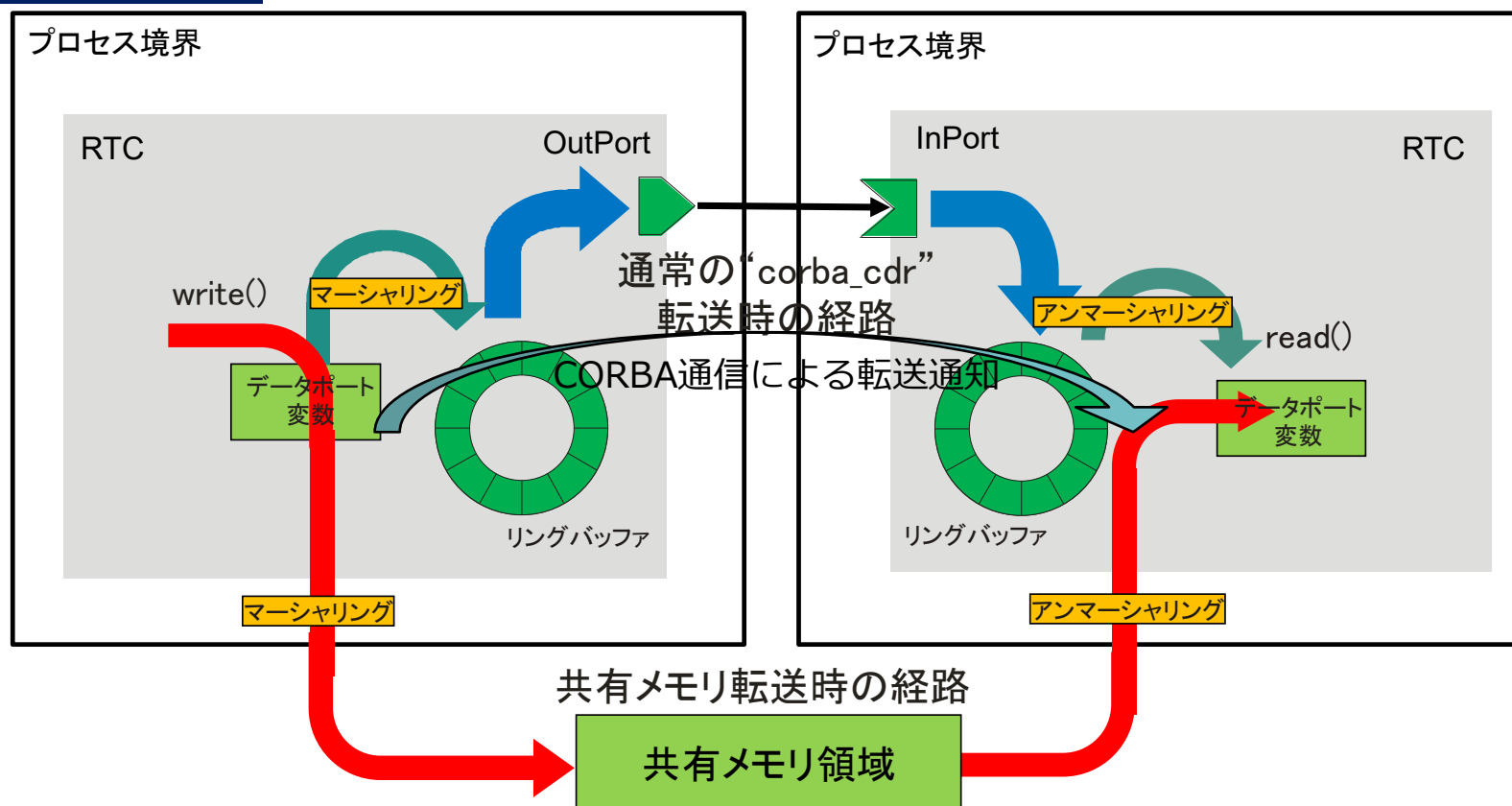


バッファ、マーシャリングをバイパスして直接変数領域でデータを受け渡し（変数間コピーなので高速）



パフォーマンスを気にせずモジュール分割可能

共有メモリ接続

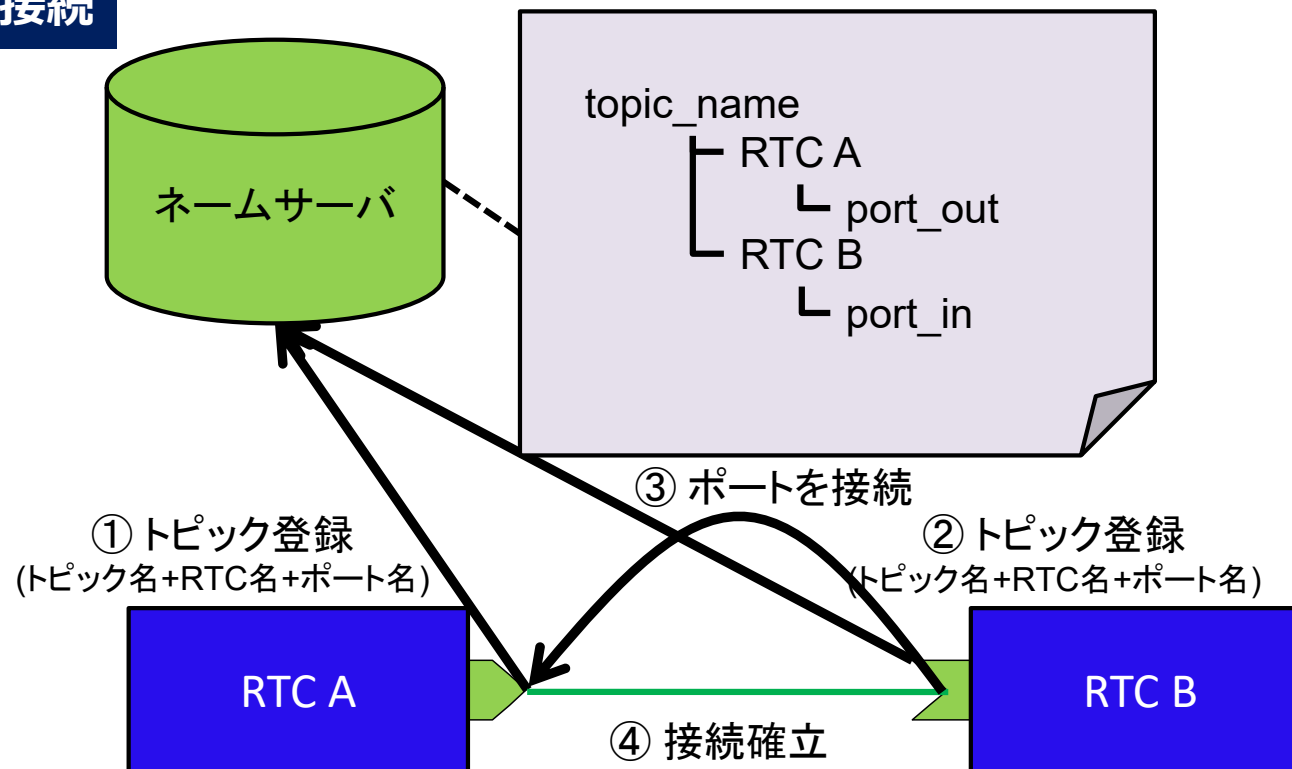


プロセス境界をまたいで共有メモリでデータを転送



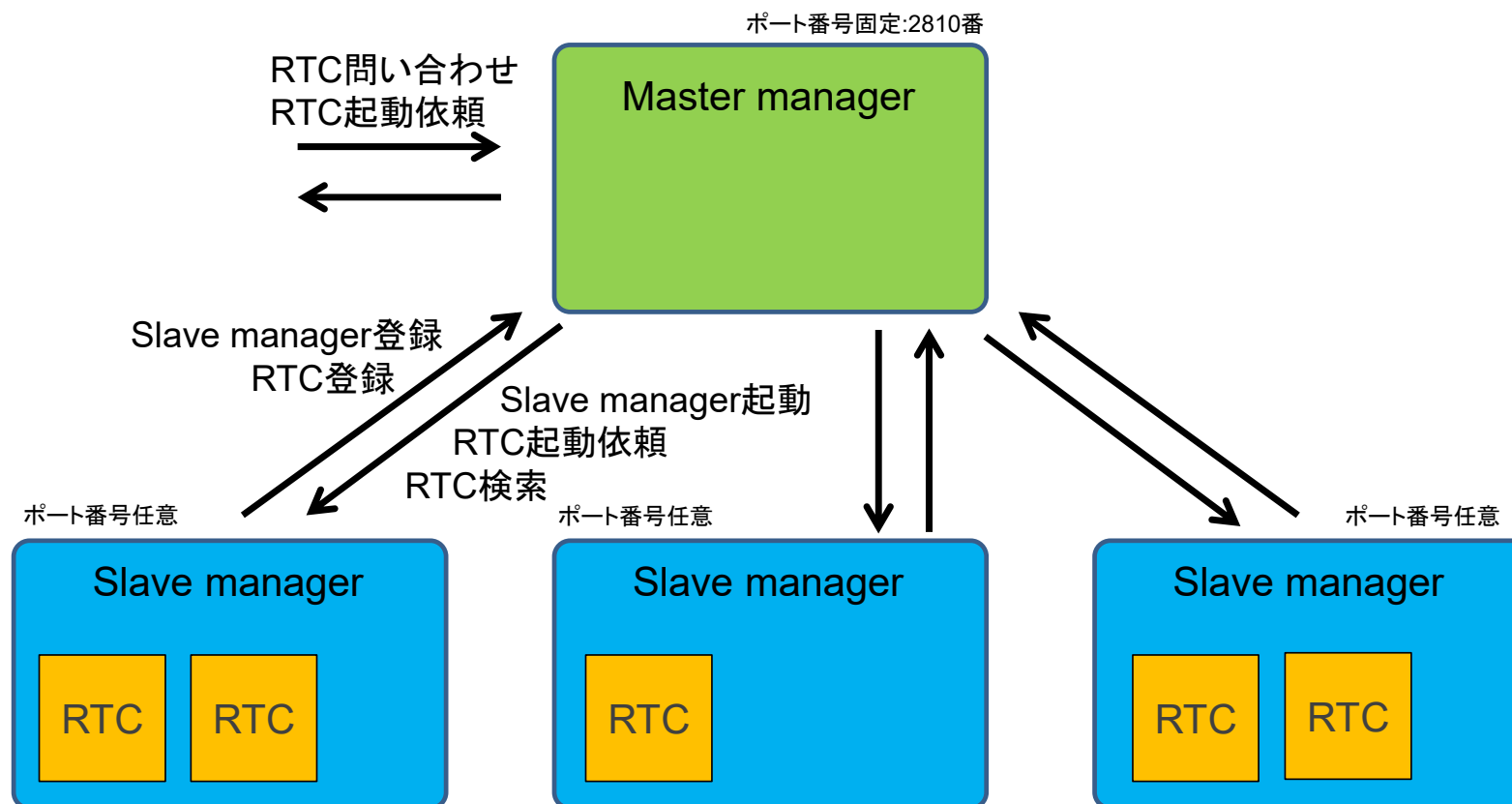
言語が異なるRTC同士でも高速通信可能

トピック接続



相手のポートを特定せずに、同一のトピック（ラベル）を持つポート間で自動接続を行う機能。

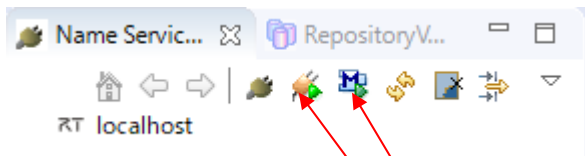
※ROS等のpub/sub通信と同等の方式。



コンポーネントの起動～終了（ライフサイクル）をリモートからすべて制御可能に。多数のノードを統一的に管理可能となり、運用時の効率向上が図れる。

別のノード（PC）上のRTCをリモート起動し、システム全体を一括立ち上げ可能。

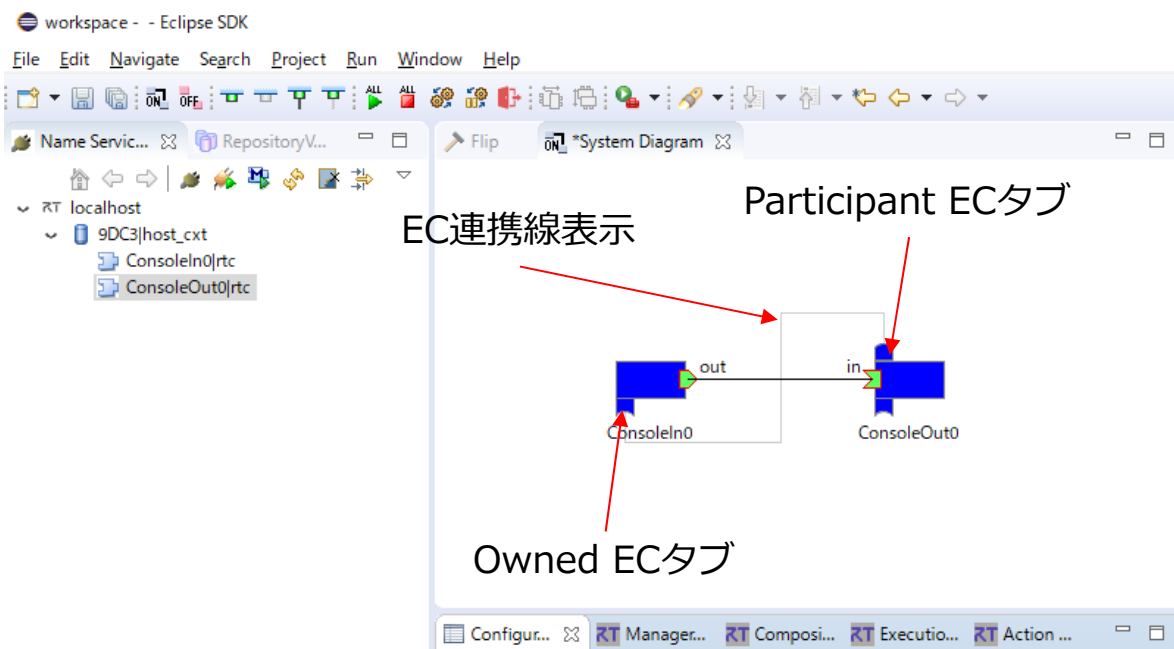
ネームサービスビュー



- マネージャ起動ボタン
- ネームサーバ起動ボタン

これまでの、
「ネームサービス起動→システム構築」
の手順をRTSystemEditor内で完結可能

実行コンテキスト連携タブ



- RTC間の実行コンテキストの連携が可能に
- 複合コンポーネント化しなくても同期実行可能

これまで：実行コンテキスト(EC)は起動時に設定するのみ

今回の拡張：起動後もECの操作（他のRTCをアタッチ、EC毎にActivate/Deactivate等）設定が可能に



RTCロジックの実行方法を柔軟に選択可能に

- Windows
 - VS2017対応 (1.2.1ではVS2019対応)
 - VCバージョンチェンジャーツール提供
 - Rtshell 2.4.0を同梱
 - 32bit、64bit同時インストール可能
 - 同梱OpenCV 3.4へ変更
 - 同梱JRE OpenJDK 1.8へ変更
 - CMakeによるmsiパッケージ作成可能
- Linux
 - OpenRTM-aist-Java, OpenRTPパッケージ提供
 - インストールスクリプト1本化 (C++, Python, Java, ツール)
 - CMakeによる deb/rpmパッケージ作成が可能
 - Ubuntu (14.04, 16.04, 18.04, 18.10)
 - Debian (近々配布予定)
 - Fedora (近々配布予定)
 - Raspbian/ARM64(JETSON) (近々配布予定)

2.0に向けて

- 新標準FSM4RTCの正式サポート
 - 状態遷移型RTCの枠組み導入
- ROSやその他のミドルウェアとの連携機能
 - 要望多数
 - データポート相互接続
 - ROS1/ROS2実装済み→github
 - 運用時互換性等
 - コンポーネント起動方向等
- 開発効率向上のための機能
 - パッケージ化
 - 自動ビルド・CI
- 運用効率向上のための機能
 - ロギング機能、構成管理機能
 - RTC合成、組み込み機能
- サードパーティーRTC・ツール取り込み
- ROSノードの取り込み



Version 2.0
2019年度末

Version 2.1
2020年度中

RTミドルウェアと コミュニティ

プロジェクトページ

- ユーザが自分の作品を登録
- 他のユーザの作ったRTCを探ることができる

タイプ	登録数
RTコンポーネント群	287
RTミドルウェア	14
ツール	19
仕様・文書	4
ハードウェア	28

The screenshot displays the OpenRTM-aist project page, which serves as a repository for various robot arm components. The page is organized into sections for different robot models and their associated RTCs (Real-Time Control Components).

Robot Arm Components:

- CRANE-X7:** A section for the CRANE-X7 robot arm, showing its RTC components and a list of users who have created components for it.
- Universal Robots UR5:** A section for the Universal Robots UR5, showing its RTC components and a list of users who have created components for it.
- ToroboArm:** A section for the ToroboArm robot, showing its RTC components and a list of users who have created components for it.
- ots UR5:** A section for the ots UR5 robot, showing its RTC components and a list of users who have created components for it.

Ports and Services:

The page also includes sections for **Ports** and **Services**, which provide information about the communication interfaces and services available for these robot arms.

Input Ports:

ポート名	型	詳細
mode	TimedOctet	標準軌道制御モード/自由軌道モードの切り替えが行える
in_joint	TimedFloatSeq [6]	指定した位置にジョイント変間で線形に移動する(move)
in_pose	TimedPose3d	指定した位置にツール変間で線形に移動する
grip	TimedOctet	Gripを操作できる。ただし2-Fingerタイプのみ制御可能

Output Ports:

ポート名	型	詳細
out_joint	TimedFloatSeq [6]	現在のジョイント位置を出力する
out_pose	TimedPose3d	現在のツール位置を出力する
is_moving	TimedBoolean	ロボットが動作中かどうかを出力する(動作中:True)
force	TimedFloatSeq [1]	TCPの力矩を出力する

Services:

The services section provides information about the communication interfaces and services available for these robot arms.

Tools:

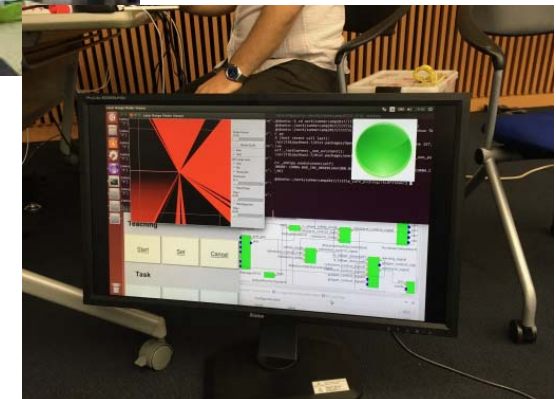
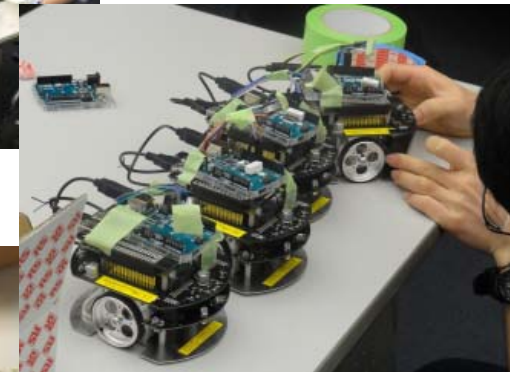
The tools section provides information about the tools available for these robot arms.

Hardware:

The hardware section provides information about the hardware available for these robot arms.

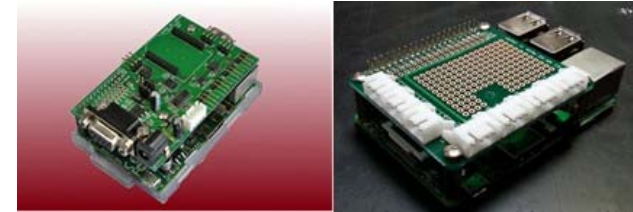
サマーキャンプ

- 毎年夏に1週間開催
- 今年：7月29日～8月2日
- 募集人数：20名
- 場所：産総研つくばセンター
- 座学と実習を1週間行い、最後にそれぞれが成果を発表
- 産総研内のさくら館に宿泊しながら夜通し？コーディングを行う！



RTミドルウェアコンテスト

- SICE SI（計測自動制御学会 システムインテグレーション部門講演会）のセッションとして開催
 - 各種奨励賞・審査基準開示:6月頃
 - エントリー〆切：8月9日(SI2019締切)
 - 講演原稿〆切：9月24日(SI予稿締切)
 - ソフトウェア登録：10月中旬
 - オンライン審査：11月下旬～
 - 発表・授賞式：12月ごろ
- 2018年度実績
 - 応募数：10件
 - 計測自動制御学会学会RTミドルウェア賞（副賞10万円）
 - 奨励賞（賞品協賛）：2件
 - 奨励賞（団体協賛）：9件
 - 奨励賞（個人協賛）：8件
- 詳細はWebページ：openrtm.org
 - コミュニティ→イベント をご覧ください



- 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！

まとめ

- RTミドルウェアの概要
 - 基本概念
- ROSとの比較、動向
- OpenRTM-aist-1.2の新機能
- 2.0以降の開発ロードマップ
- RTMコミュニティ活動