ロボットミドルウエア

安藤慶昭
独立行政法人産業技術総合研究所
知能システム研究部門
主任研究員

概要

- ロボットミドルウエアとは？
- RTミドルウエア：OpenRTM-aist
- 様々なミドルウエア/プラットフォーム
- 標準化
- 終わりに

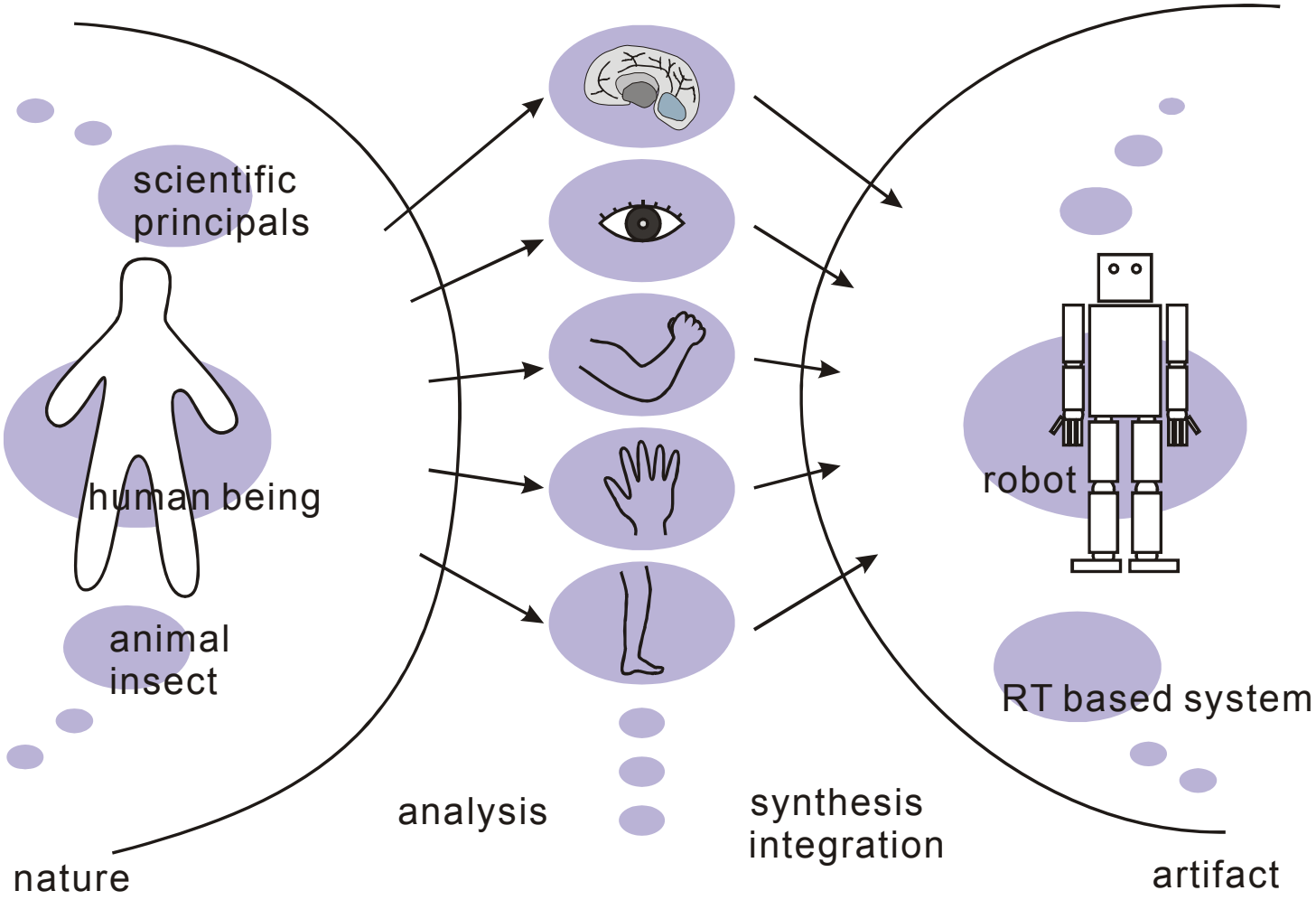
ロボットミドルウエアとは？

- 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ソフトウエア
  - 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
  - モジュール化されたソフトウエアやサーバ・クライアント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エアが開発・公開されている

ミドルウエア、コンポーネント、etc…

- ミドルウエア
  - OSとアプリケーション層の中間に位置し、特定の用途に対して利便性、抽象化向上のために種々の機能を提供するソフトウエア
  - 例：RDBMS、ORB等。定義は結構曖昧
- 分散オブジェクト（ミドルウエア）
  - 分散環境において、リモートのオブジェクトに対して透過的アクセスを提供する仕組み
  - 例：CORBA、Ice、Java RMI、DCOM等
- コンポーネント
  - 再利用可能なソフトウエアの断片（例えばモジュール）であり、内部の詳細機能にアクセスするための（シンタクス・セマンティクスともにきちんと定義された）インターフェースセットをもち、外部に対してはそのインターフェースを介してある種の機能を提供するモジュール。
- CBSD（Component Based Software Development）
  - ソフトウエア・システムを構築する際の基本構成要素をコンポーネントとして構成するソフトウエア開発手法

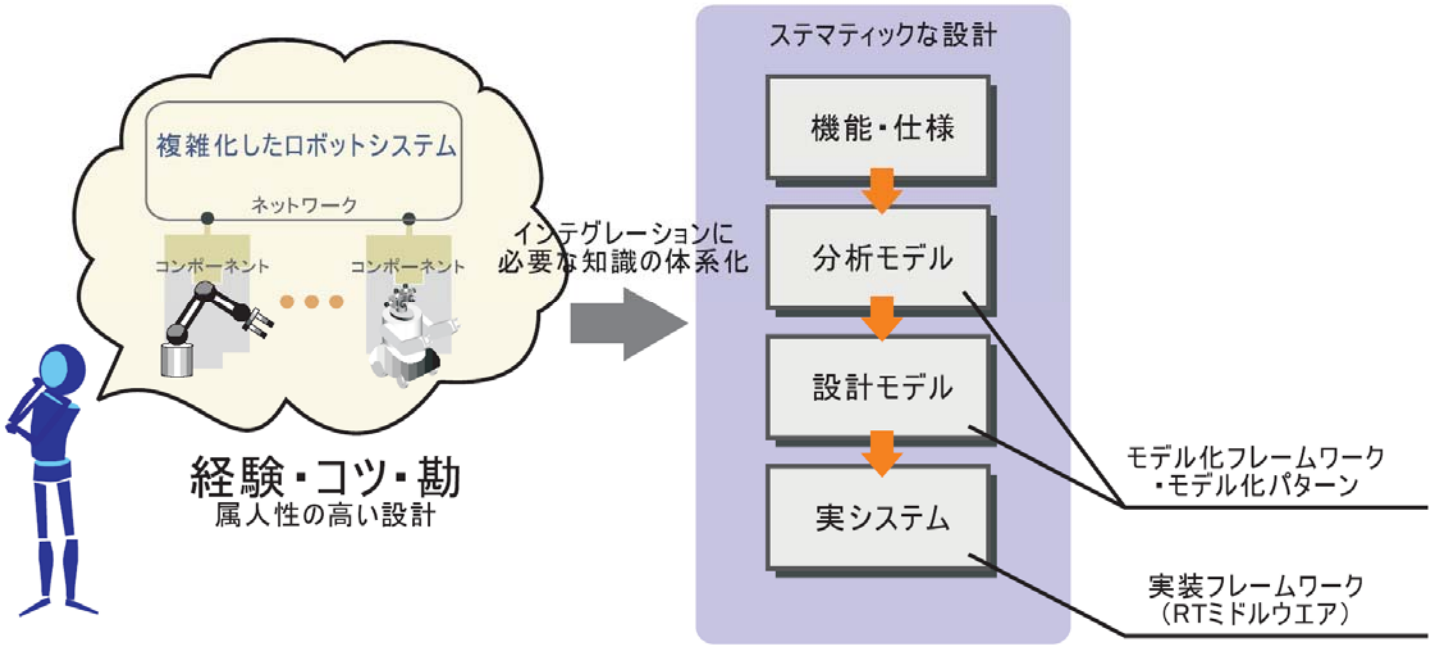
アナリシスからシンセシスへ



体系的システム開発

開発者の経験やノウハウに依存したロボットシステム開発

体系的システムデザイン: 分析、設計、実装の一連の流れ



様々なミドルウエア/プラットフォーム

- OpenRTM-aist
- ROS
- OROCOS
- OPRoS
- ORCA
- Microsoft Robotic Studio
- Player/Stage/Gazebo
- ORiN
- RSNP
- UNR Platform
- OPEN-R
- Open Robot Controller architecture

RT-Middleware
OpenRTM-aist

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ミドルウエア

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エア
- RT-Component（RTC)
  - RT-Middlewareにおけるソフトウエアの基本単位

RTミドルウエアプロジェクト

NEDO 21世紀ロボットチャレンジプログラム（2002-2004年度）
「ロボット機能発現のために必要な要素技術開発」

- RT分野のアプリケーション全体に広く共通的に使われる機能およびRT要素の部品化（モジュール化）の研究開発
- 分散オブジェクト指向システムのミドルウェアであるCORBAをベースとして行う。
- RT要素の分類を行い、モジュール化の形態、必要な機能、課題、インタフェース仕様などを明確にする。

１４年度成果報告書より

従来のシステムでは…

Joystick

Joystick
software

Robot Arm
Control software

Robot Arm

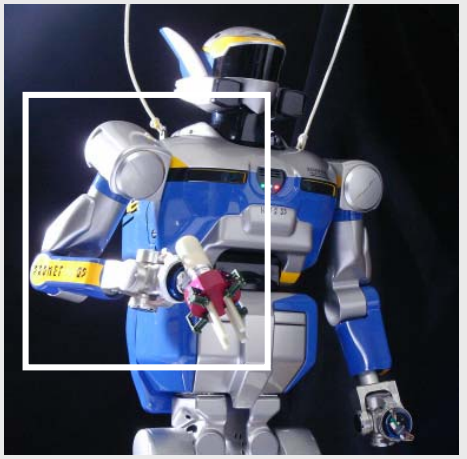
互換性のあるインターフェース同士は接続可能

従来のシステムでは…

Joystick

Joystick
software

Humanoid's Arm
Control software



Humanoid's Arm

Robot Arm
Control software

Robot Arm

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

RTミドルウエアでは…

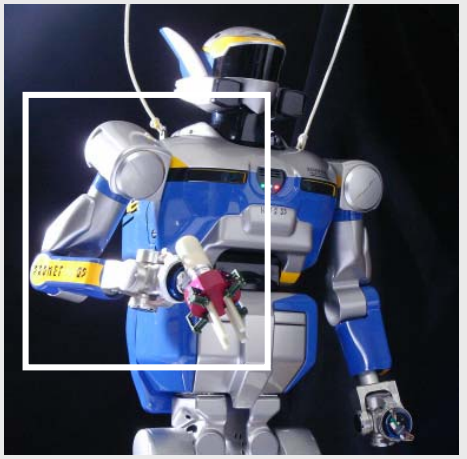
RTミドルウエアは別々に作られたソフトウエアモジュール同士を繋ぐための共通インターフェースを提供する

Joystick

Joystick software

compatible arm interfaces

Arm A Control software



Humanoid's Arm

Arm B Control software

Robot Arm

ソフトウエアの再利用性の向上
RTシステム構築が容易になる

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

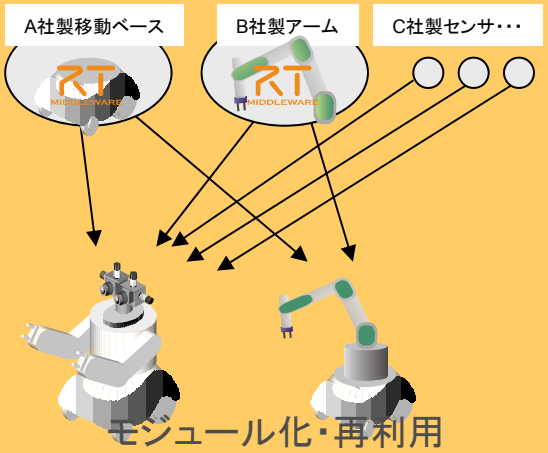
モジュール化のメリットに加えて

- ソフトウエアパターンを提供
  - ロボットに特有のソフトウエ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
- 分散ミドルウエア
  - 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ミドルウエアの目的

モジュール化による問題解決

コストの問題

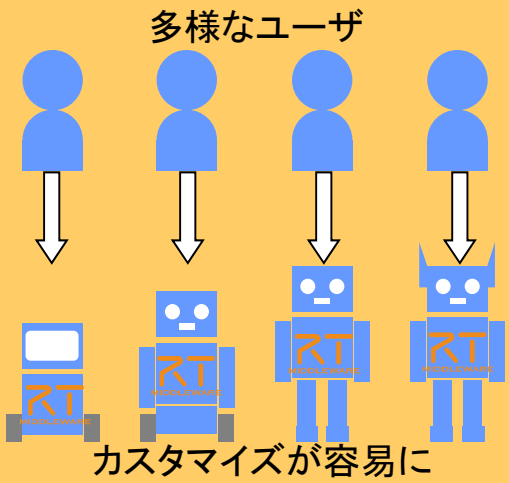


ロボットの低コスト化

技術の問題

最新技術を利用可能

ニーズの問題



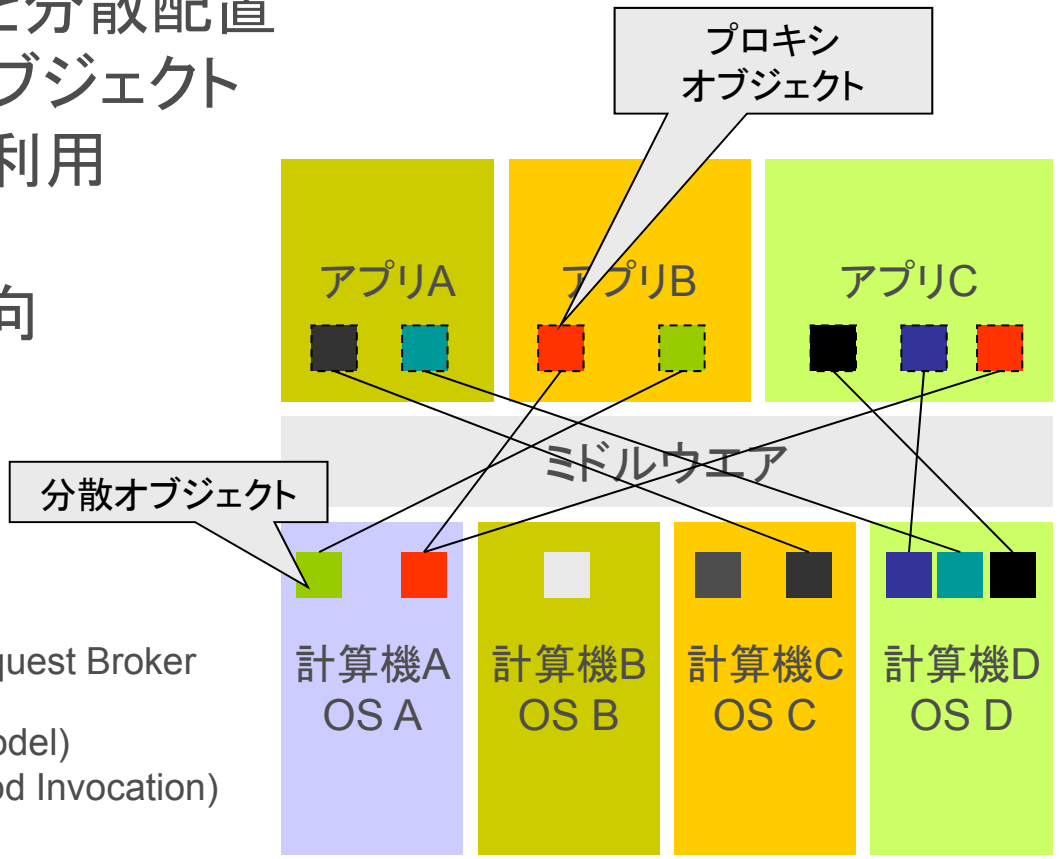
多様なニーズに対応

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

分散オブジェクトとは？

- システムの機能分割と分散配置
- ネットワーク透過なオブジェクト
- コンポーネント化と再利用

オブジェクト指向
＋
ネットワーク



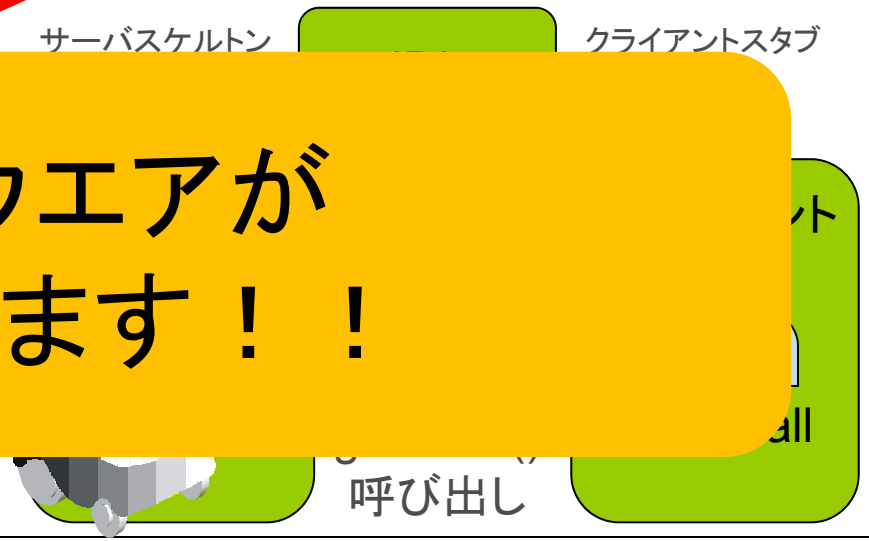
- 代表例
  - CORBA (Common Object Request Broker Architecture)
  - CCM (CORBA Component Model)
  - JavaRMI (Java Remote Method Invocation)
  - EJB (Enterprise Java Beans)
  - DCOM, HORB etc…

CORBAの例

本題にたどり着くまでが面倒

```
class MobileRobot_Impl
 : public virtual POA_MobileRobot,
   public virtual
     PortableServer::RefCountServant
```

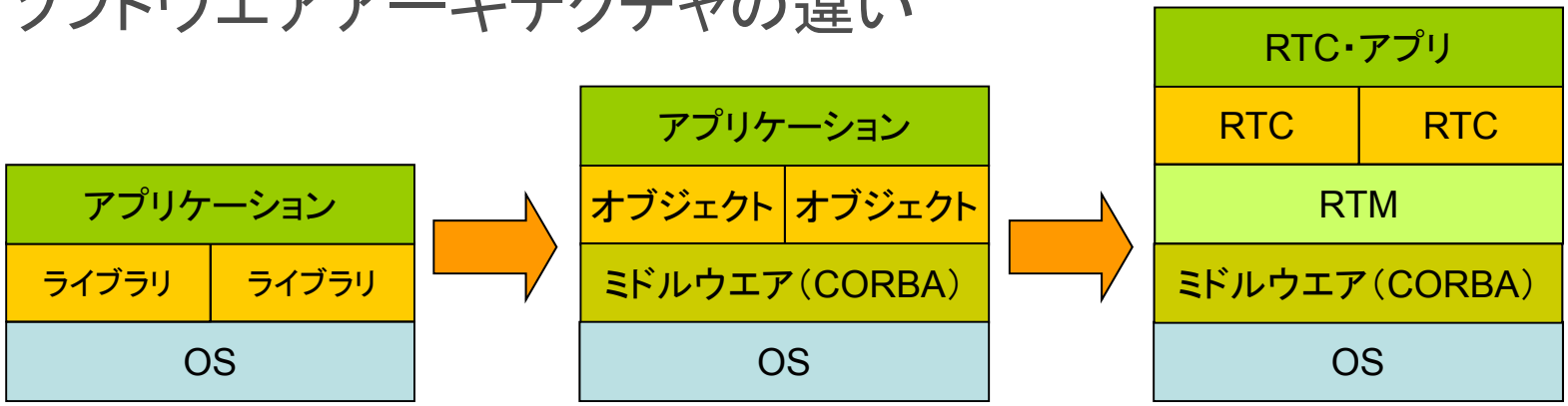
```
robot->gotoPos(pos);
```



RTミドルウエアが
全部面倒みます！！

RTM、RTCとは？

ソフトウエアアーキテクチャの違い



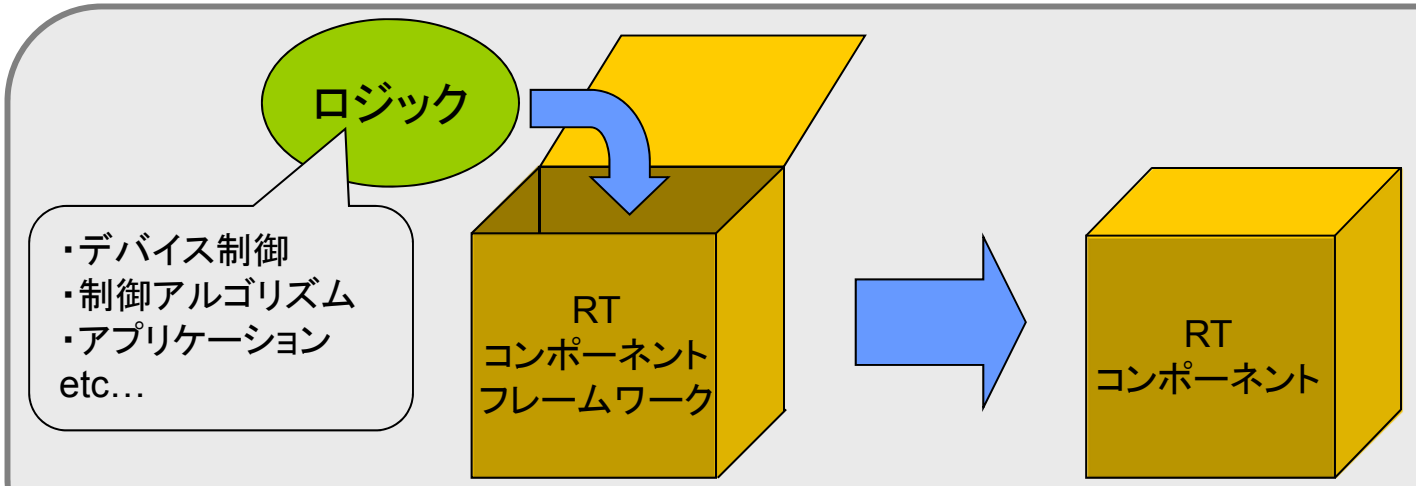
従来ソフトウエアから分散オブジェクトへ

- オブジェクト指向開発
- 言語・OSの壁を越えて利用できる
  - インターフェースをIDLで定義
  - 各言語へ自動変換
  - OS、アーキテクチャの違いを吸収
- ネットワーク透過に利用できる
  - 分散システムを容易に構築可能

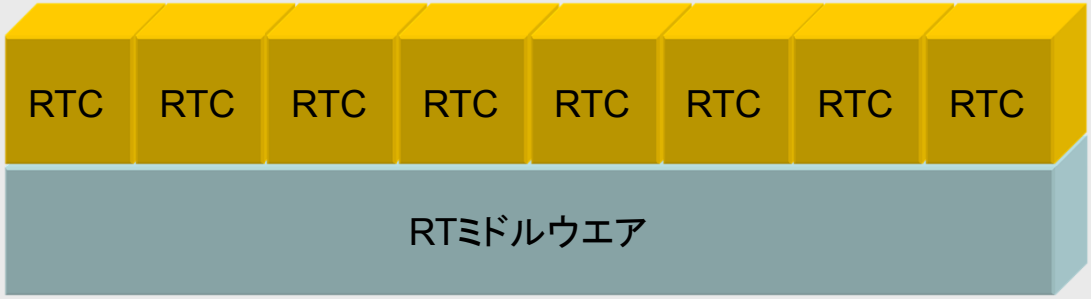
分散オブジェクトからRTCへ

- インターフェースがきちんと決まっている
  - IDLで定義された標準インターフェース
  - 呼び出しに対する振る舞いが決まっている(OMG RTC 標準仕様)
  - 同じ部品として扱える
- コンポーネントのメタ情報を取得することができる
  - 動的な接続や構成の変更ができる
- ロボットシステムに特有な機能を提供
  - 後述

RTミドルウエアとRTコンポーネント



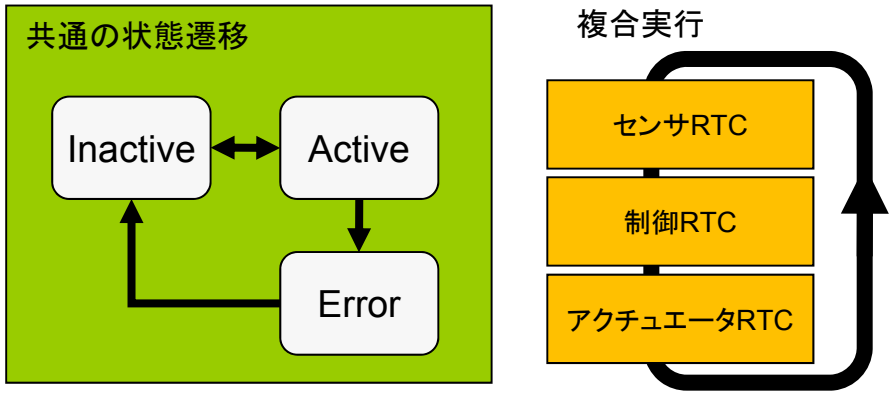
ロジックを箱（フレームワーク）に入れたもの＝RTコンポーネント（RTC）



RTCの実行環境（OSのようなもの）＝RTミドルウエア（RTM）
※RTCはネットワーク上に分散可能

RTコンポーネントの主な機能

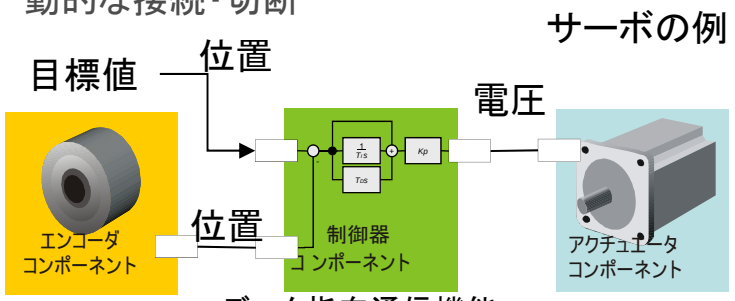
アクティビティ・実行コンテキスト



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

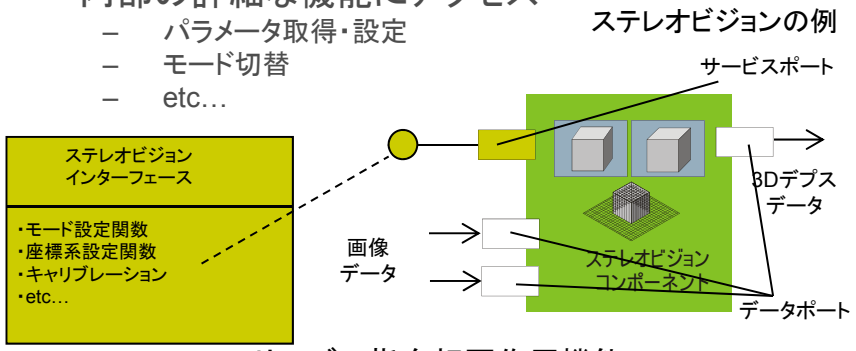
- データ指向ポート
- 連続的なデータの送受信
- 動的な接続・切断



データ指向通信機能

サービスポート

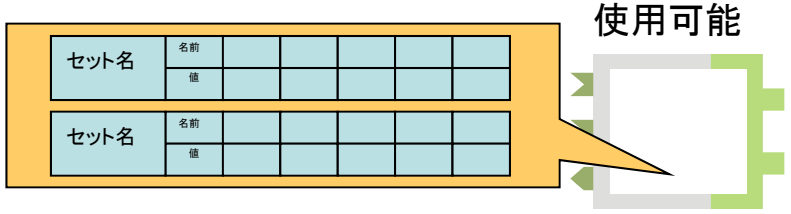
- 定義可能なインターフェースを持つ
- 内部の詳細な機能にアクセス
  - パラメータ取得・設定
  - モード切替
  - etc…



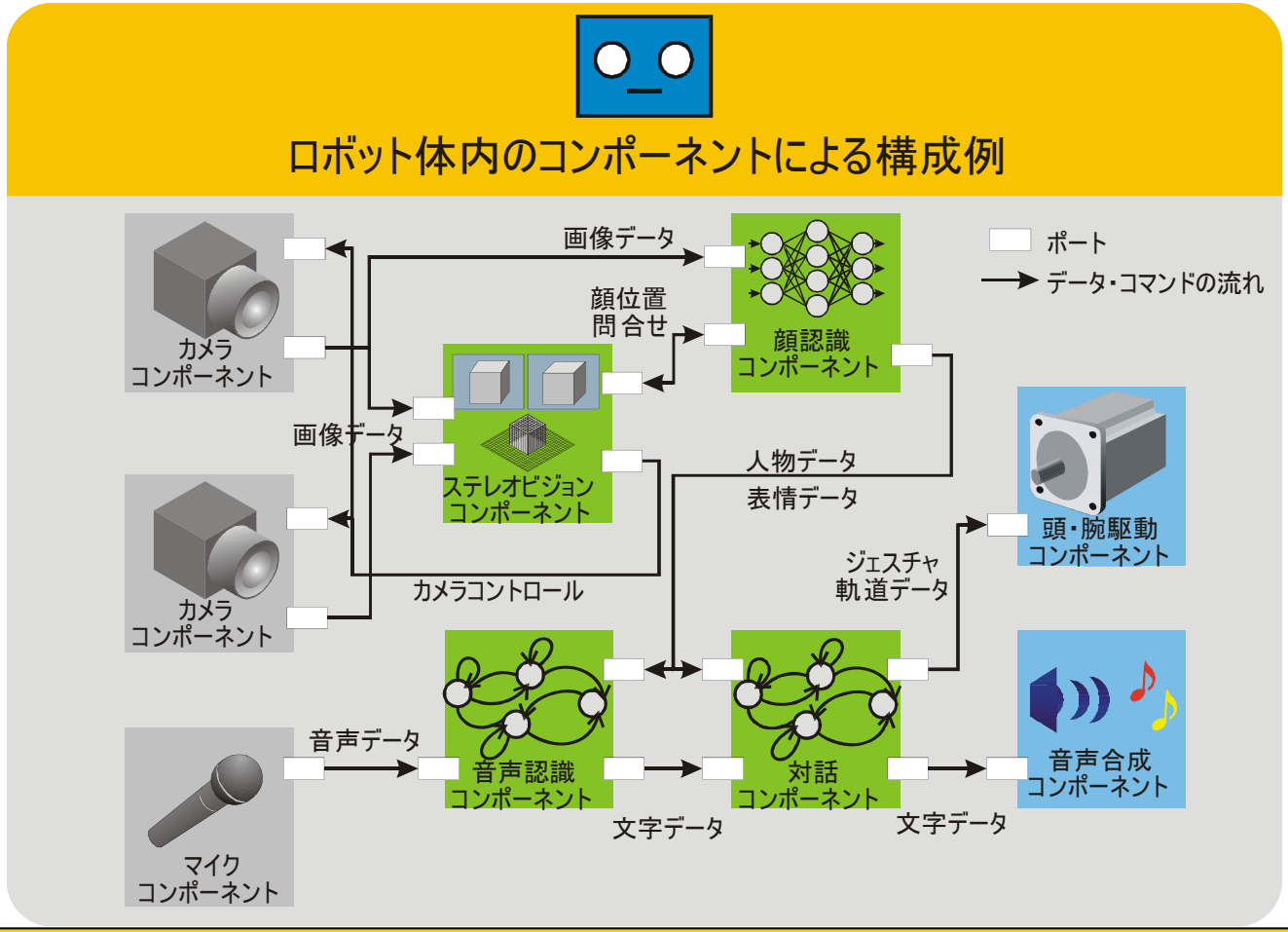
サービス指向相互作用機能

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



RTCの分割と連携



（モジュール）情報の隠蔽と公開のルールが重要

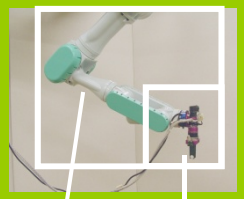
RTミドルウエアによる分散システム

RTMにより、ネットワーク上に分散するRTCをOS・言語の壁を越えて接続することができる。

ロボットA

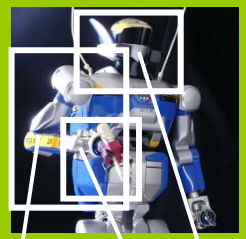
RTC | RTC
RTM
VxWorks

ロボットB



RTC | RTC
RTM
FreeBSD

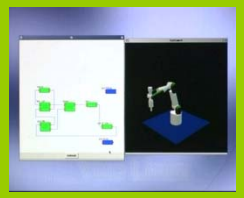
ロボットC



RTC | RTC | RTC
RTM
ARTLinux

ネットワーク

Linux
RTM
RTC | RTC



アプリケーション

Windows
RTM
RTC | RTC

操作デバイス

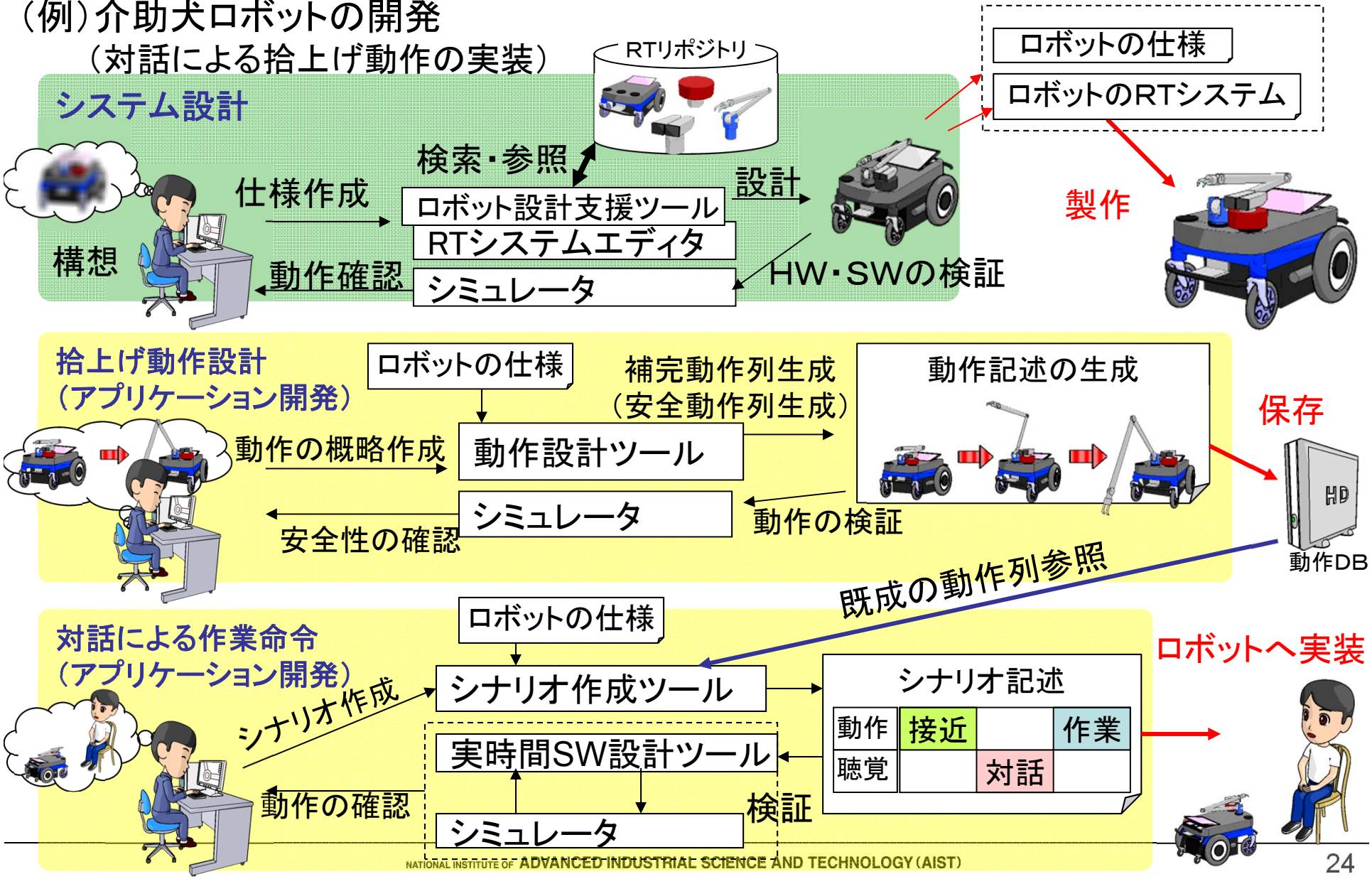
uITRON
RTM
RTC | RTC

センサ

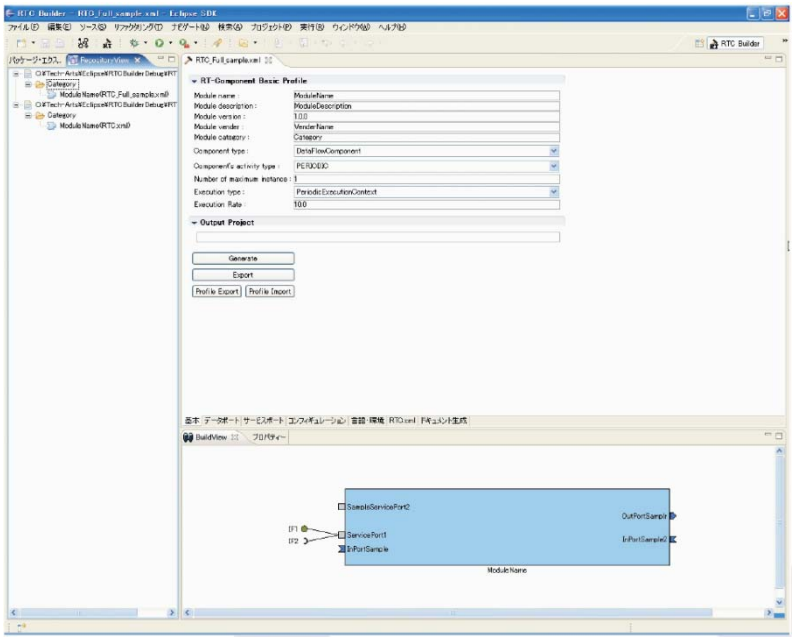
RTC同士の接続は、プログラム実行中に動的に行うことが出来る。

プラットフォーム概要

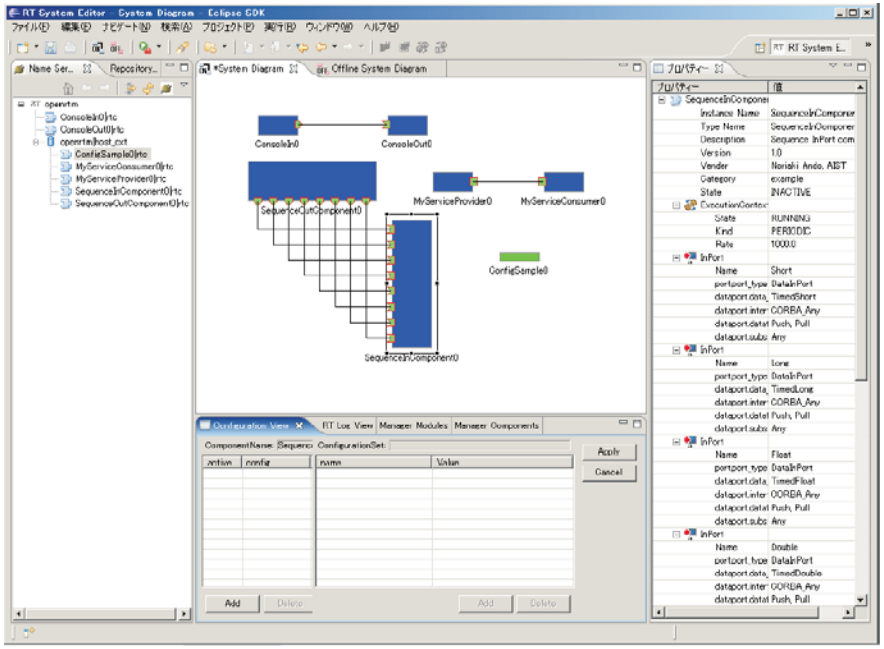
（例）介助犬ロボットの開発
（対話による拾上げ動作の実装）



ツールチェーン



RTCBuilder
RTコンポーネント設計・コード生成



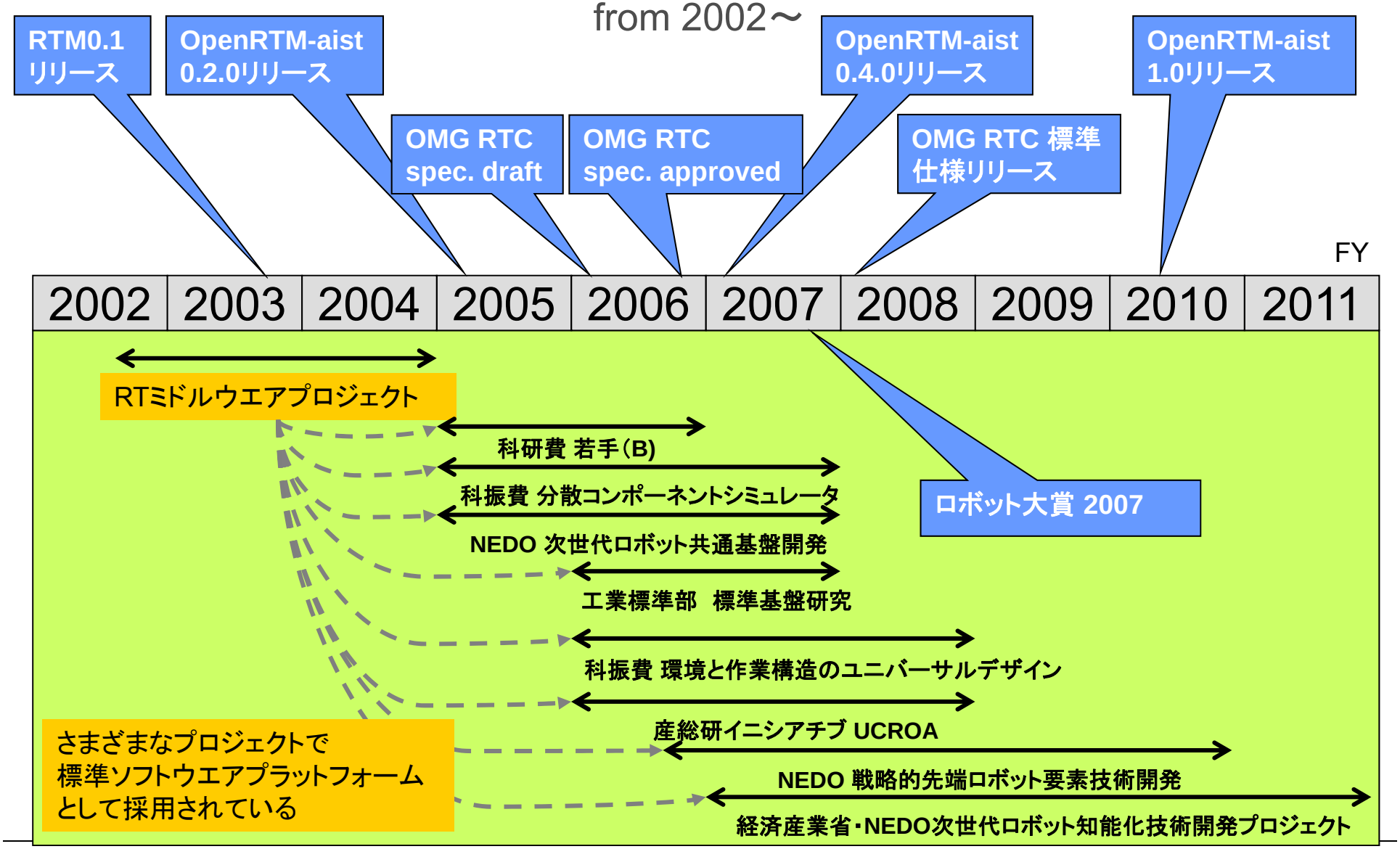
RTSystemEditor
RTCを組み合わせてシステムを設計

RTC・RTM統合開発環境の整備

RTC設計・実装・デバッグ、RTMによるインテグレーション・デバッグまでを
一貫して行うことができる統合開発環境をEclipse上に構築

RT-Middleware関連プロジェクト

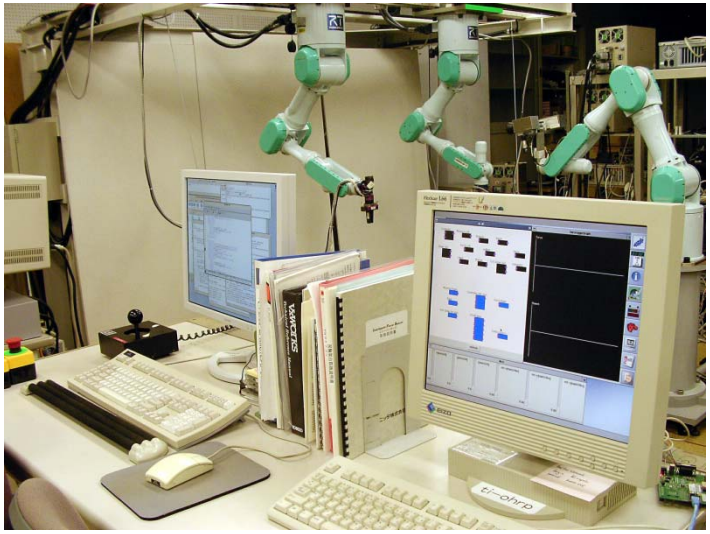
from 2002～



RTミドルウエアPJ

FY2002.12-FY2004

- 名称：NEDO 21世紀ロボットチャレンジプログラム
  - 「ロボット機能発現のために必要な要素技術開発」
- 目的：
  - RT要素の部品化（モジュール化）の研究開発
  - 分散オブジェクト指向開発
  - RT要素の分類・モジュール化に必要な機能・インタフェース仕様の明確化
- 予算規模：
  - 65百万円
  - 全体267.3百万円

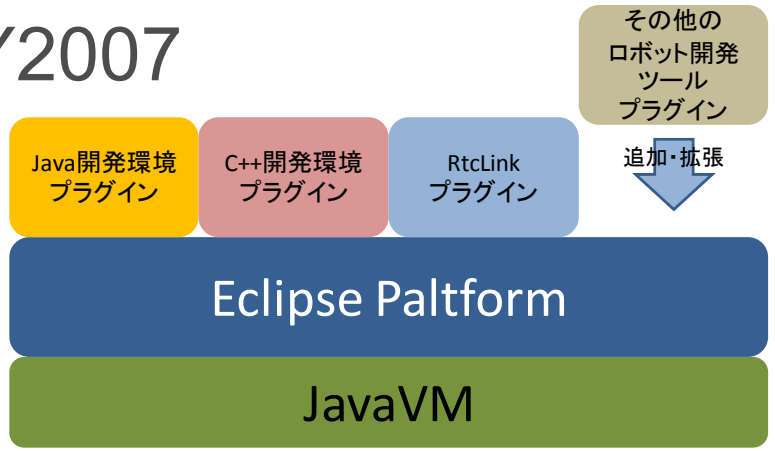


NEDO基盤PJ

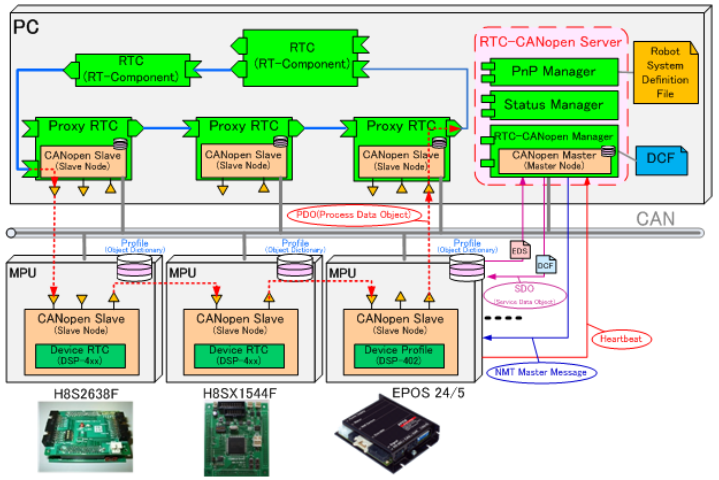
FY2003-FY2007

- 名称:「運動制御デバイスおよびモジュールの開発」
- 目的:
  - 運動制御デバイスの開発
  - デバイスに搭載するRTCの開発
  - その他モーションコントロールに資するRTM/RTCの開発
- 予算規模:
  - 15百万円/年
  - 371百万円、全体1,259百万円

dsPIC版RTC-Liteの開発



ツールのEclipseプラグイン化

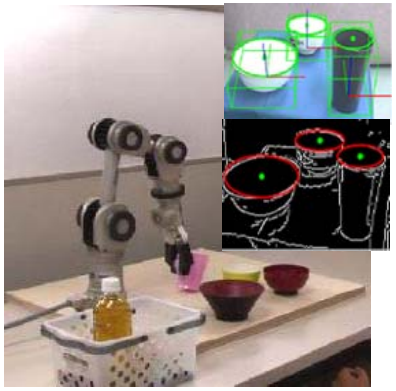


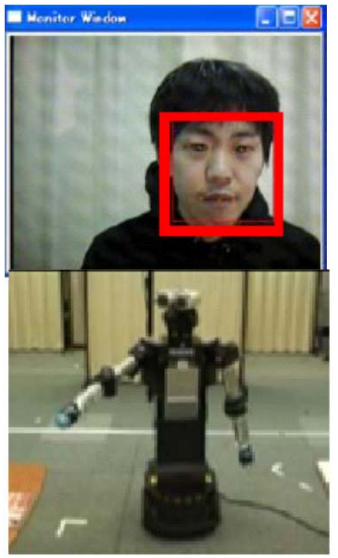
RTC-CANの開発

知能化PJ

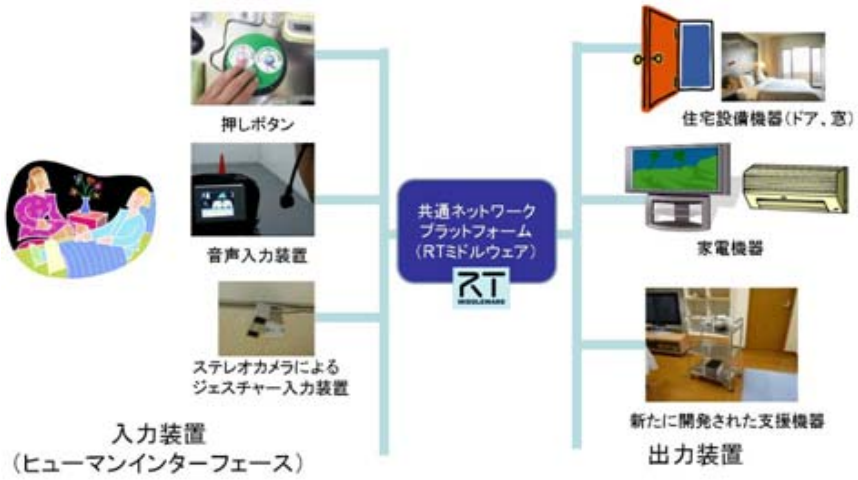
FY2007-FY2012

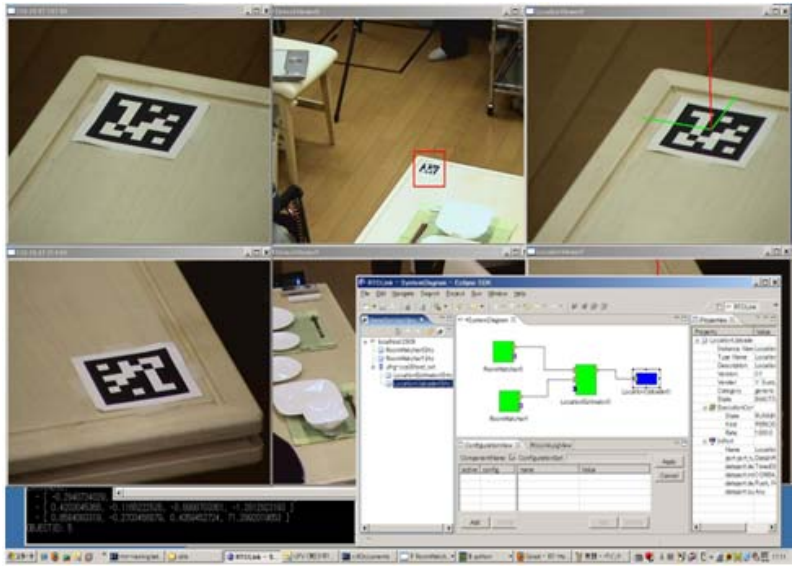
- 名称:「次世代ロボット知能化技術開発プロジェクト」
- 目的
  - ソフトウエアプラットフォームの開発
  - 作業知能、移動知能、コミュニケーション知能に関するモジュールの開発
- 予算:
  - 400百万円
  - 全体7,000百万円
- 研究グループ
  - 15グループ





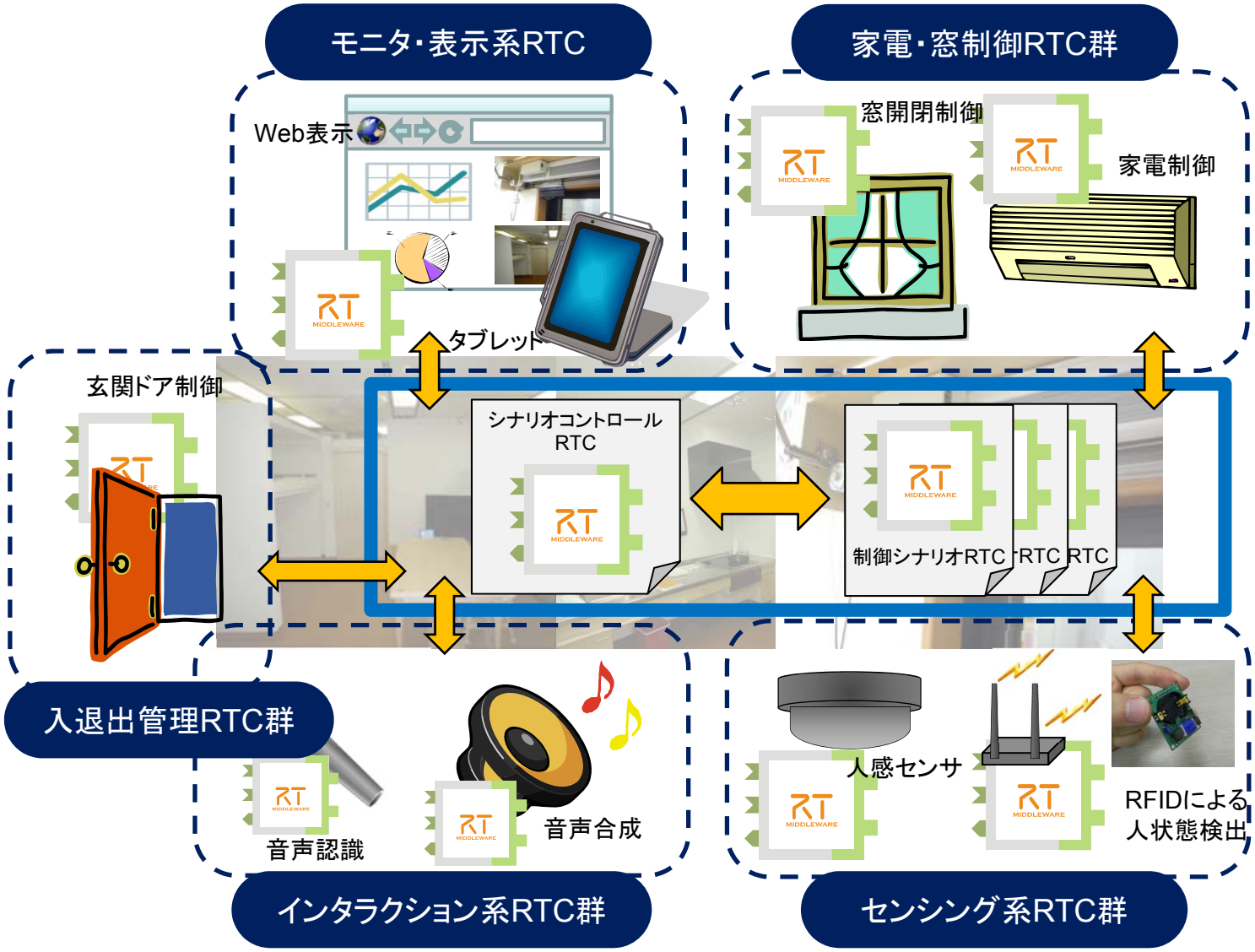
NEDOオープンイノベーションプロジェクト



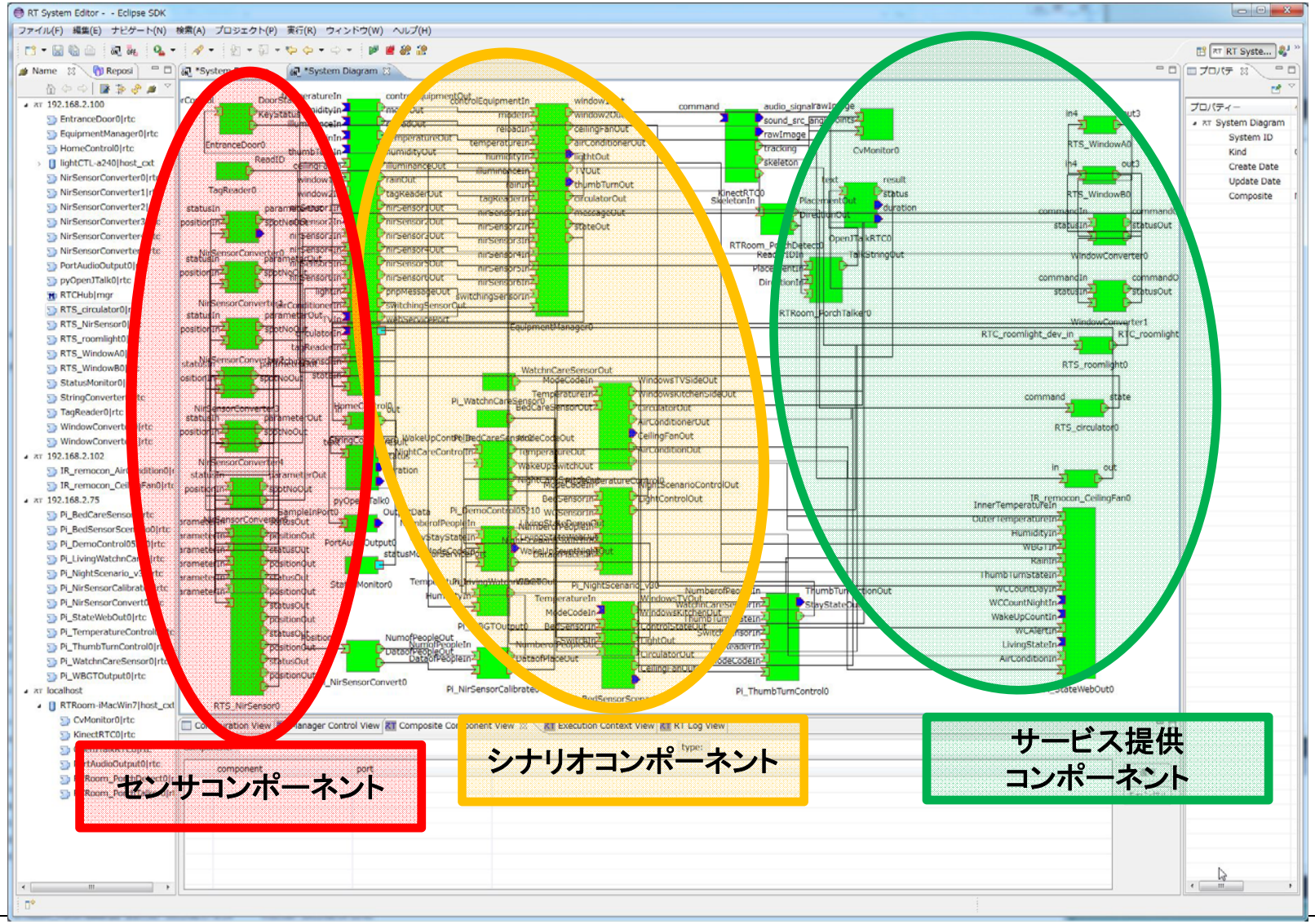


RTミドルウエアを基盤として、多数の
福祉機器同士を連携

RTルーム


モニタ・表示系RTC
Web表示
タブレット
家電・窓制御RTC群
窓開閉制御
家電制御
玄関ドア制御
シナリオコントロール
RTC
制御シナリオRTC
RTC
RTC
入退出管理RTC群
音声認識
音声合成
人感センサ
RFIDによる
人状態検出
インタラクション系RTC群
センシング系RTC群

RTルームのRTコンポーネント群



応用例


HRP-4/4C: Kawada/AIST

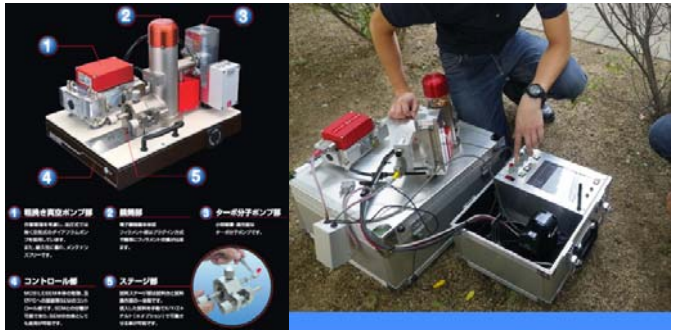
HIRO: Kawada/GRX

TAIZOU: GRX

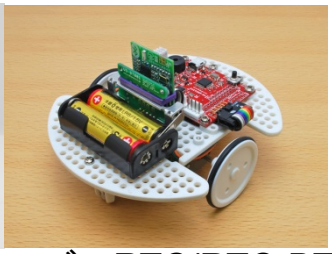
DAQ-Middleware: KEK/J-PARC
KEK: High Energy Accelerator Research Organization
J-PARC: Japan Proton Accelerator Research Complex

Life Robotics:RAPUDAアーム

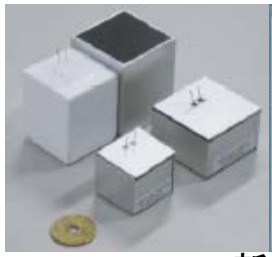
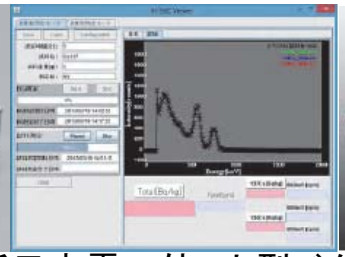
前川製作所･アールティ: OROCHI



新日本電工他: Mobile SEM

VSTONE: ビュートローバーRTC/RTC-BT

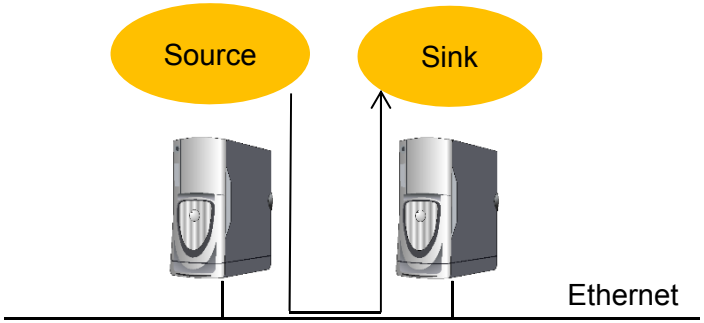



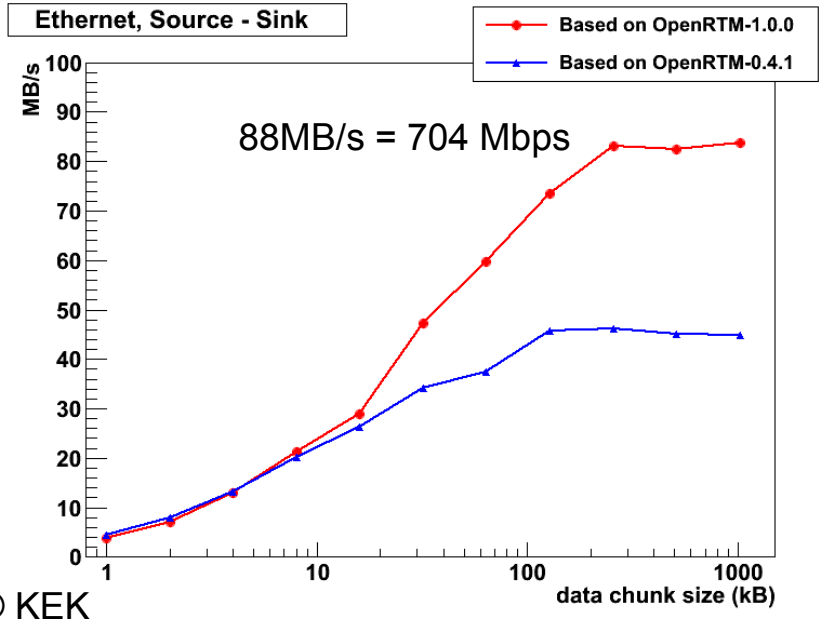

新日本電工他: 小型ベクレルカウンタ

Japan Proton Accelerator Research Complex

（J-PARC,大強度陽子加速器施設）

Model : Dell PowerEdge SC1430
CPU :Intel Xeon 5120 @ 1.86GHz 2 Cores ×2
Memory: 2GB
NIC: Intel Pro 1000 PCI/e (1GbE)
OS: Scientific Linux 5.4 (i386)

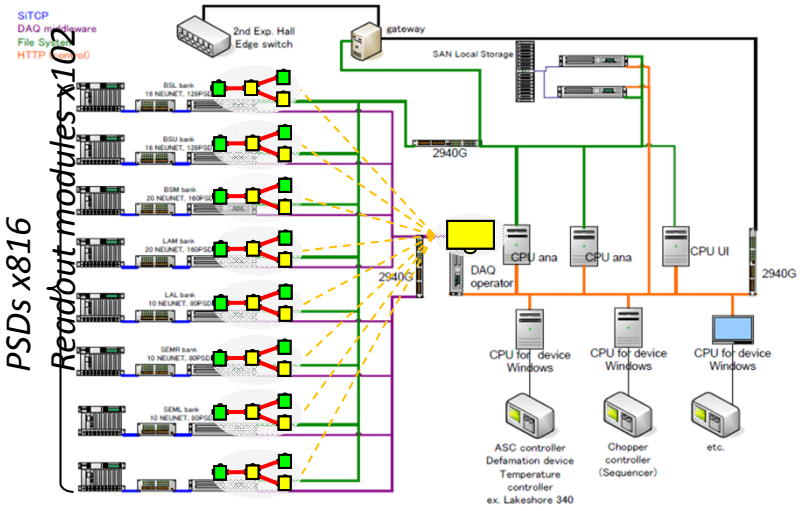




© KEK

© KEK

Joint Project between KEK and JAEA (former JAERI)



様々なミドルウエア/プラットフォーム

- OpenRTM-aist
- ROS
- OROCOS
- OPRoS
- ORCA
- Microsoft Robotic Studio
- Player/Stage/Gazebo
- ORiN
- RSNP
- UNR Platform
- OPEN-R
- Open Robot Controller architecture

ROS(Robot Operating System)

http://wiki.ros.org/

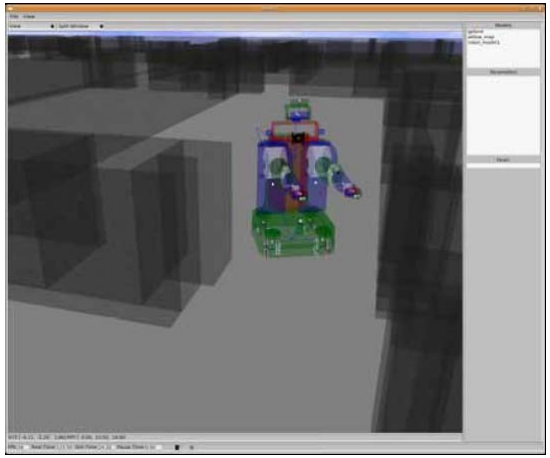
Willow Garage

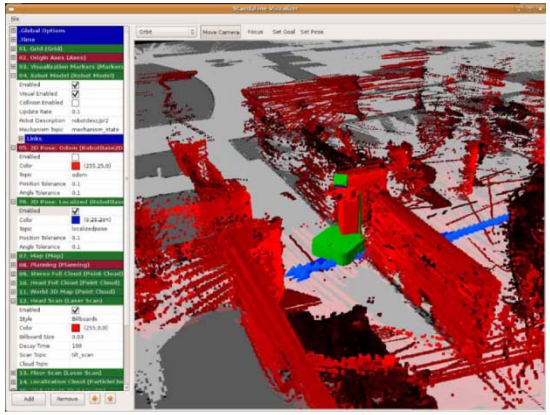
- Willow Garage
  - 2007年設立のロボットベンチャー（米、Menlo Park）
  - Scott Hassanが出資
    - googleの初期エンジンの作者の一人
  - Brian Gerkeyがソフトウエア部分のPL
    - Playerの作者の一人
  - ビジネスモデル
    - ソフト：ROS（無償）＋ハード：PR2 を販売
    - その他は不明、資金がたくさんあるので急いで利益を上げる必要はないとのこと
    - まずPR2を10台無償で大学などに配布予定

ROS（Robot Operating System）

- 目的：ロボット研究を効率的にする
- OSではない
  - Robot Operating System, Robot Open Source
- UNIX哲学
  - UNIX的なコンパクトなツール、ライブラリ群による効率的なインテグレーション環境を提供する。
  - モジュール性、柔軟性、再利用性、言語・OS非依存、オープンソース
- プラットフォームアーキテクチャ
  - 独自ミドルウエア（独自IDLからコード生成）
  - メッセージベース
  - pub/subモデル、RPCモデル

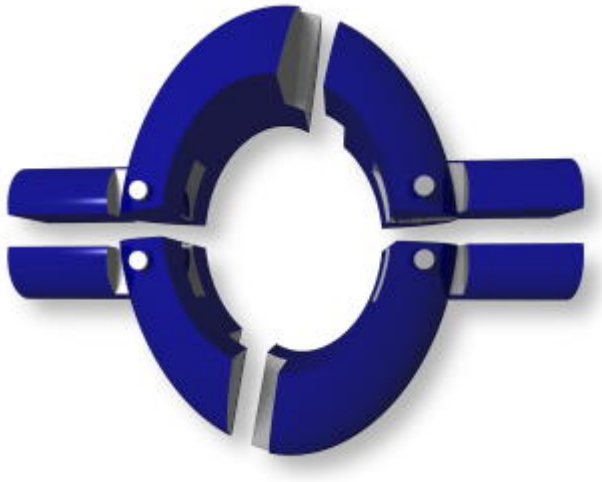
ROS（2）





- ライブラリ・ツール群が充実
  - シミュレータ：Gazeboベース
  - ナビゲーション：Player等
  - 座標変換・キャリブレーション
  - プランニング：TREX
  - マニピュレーション：OpenRAVE
  - ビジョン：OpenCV
  - 音声：ManyEars
- 外部のオープンソースライブラリを取り込みライブラリ群として整備

OROCOS



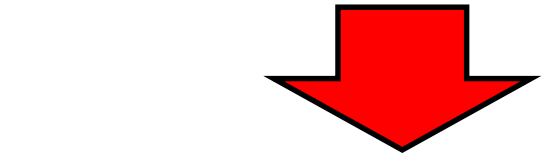
http://www.orocos.org/

OROCOS

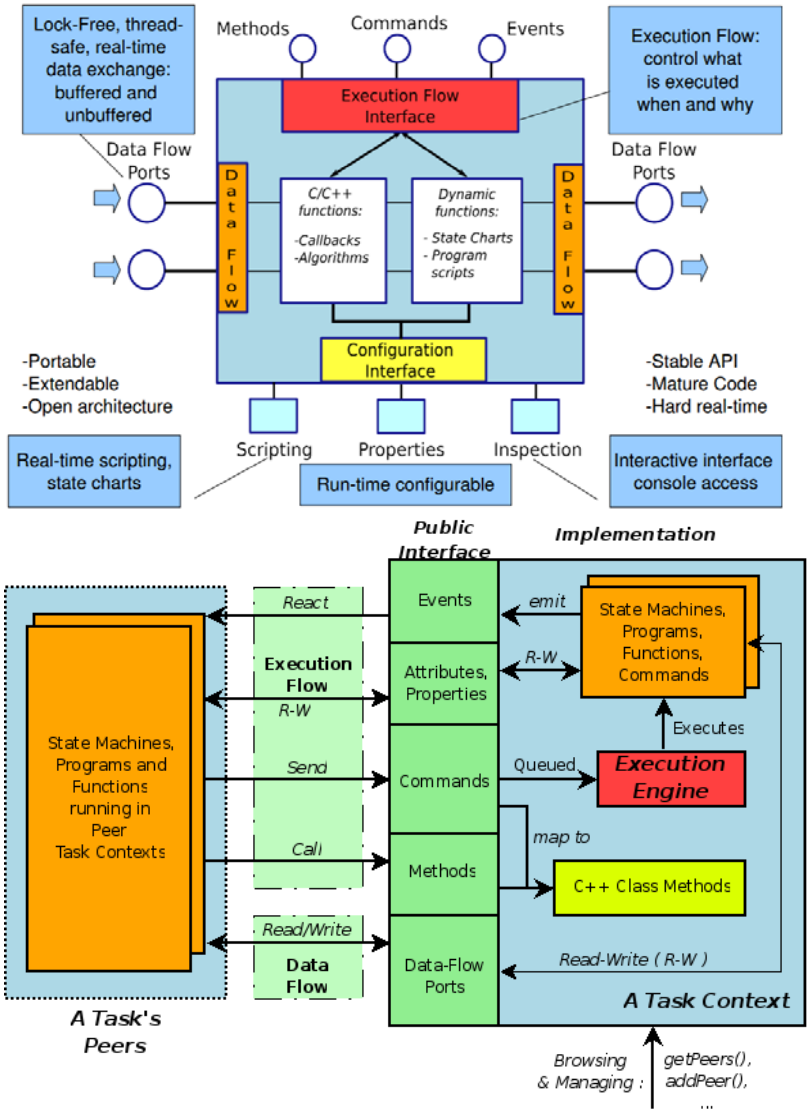
- EUプロジェクトで開発(2001-2003)
  - K.U.Leuven(ベルギー)
  - LAAS Toulouse(フランス)→ORCAへ
  - KTH Stockholm(スウェーデン)
- ハードリアルタイムのソフトウエアフレームワーク
- ロボットの制御に必要なライブラリ集（運動学、リアルタイム制御、etc…）
- 最近はコンポーネントベース開発のフレームワークも提供
- ツールによるモデルベース開発

OROCOS

- コンポーネントモデル
  - データフローポート
  - 種々のサービスインターフェース
  - コンフィギュレーション機能
  - コールバックベースのロジック実行フレームワーク



ほぼRTCを同じ（マネ？）



OPRoS

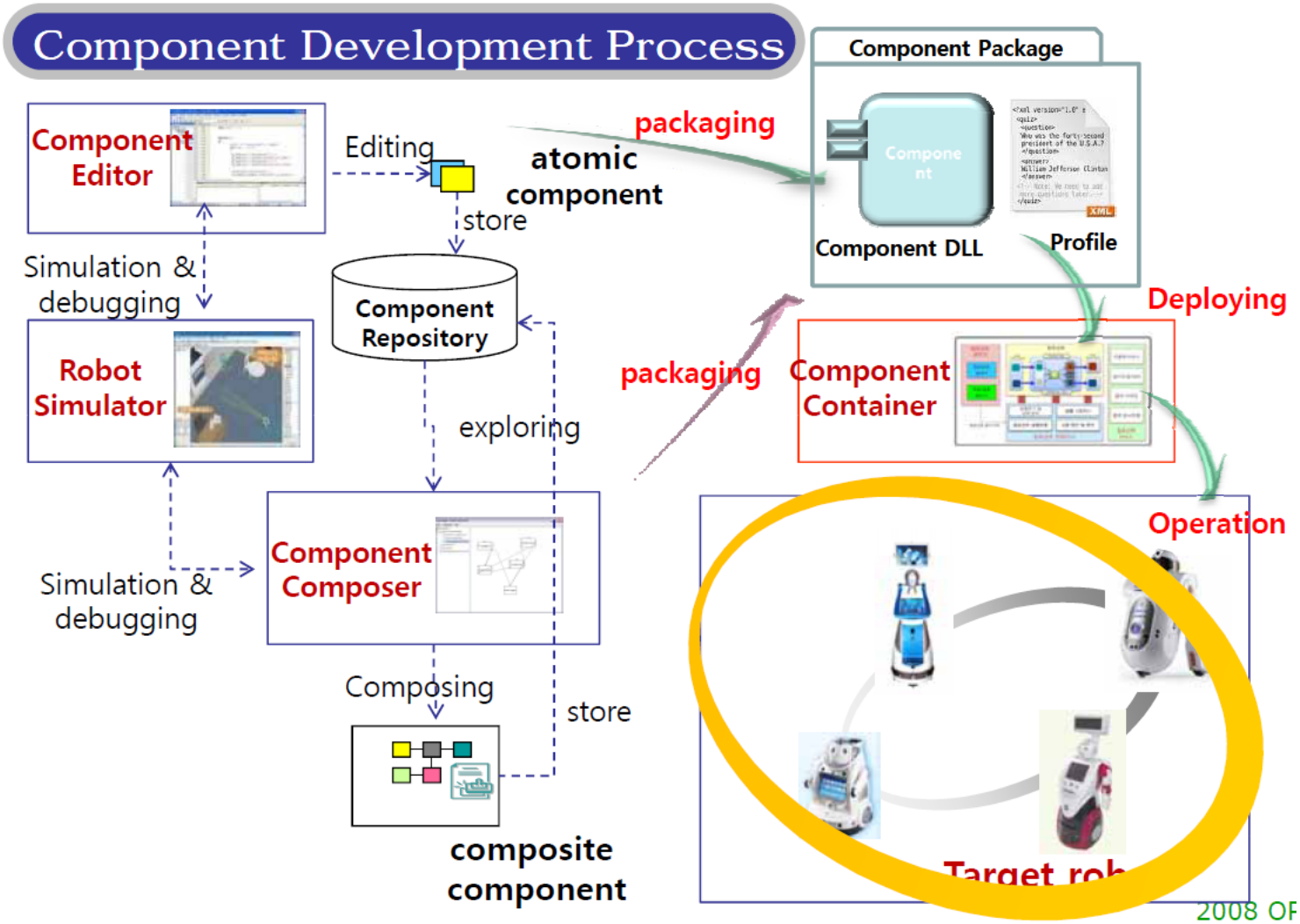
(The Open Platform for Robotic Services)

http://www.opros.or.kr/

OPRoS

- 韓国の国プロジェクトで開発されたロボット用プラットフォーム
  - ETRI(Electronics and Telecommunication Research Institute)
  - KIST, Kanwong Univ., etc…
- OMG RTC（ほぼ）準拠
- 通信ミドルウエアは独自（URC(Ubiquitous Robot Companion PJ)で開発したもの）
- ツールチェーンなども提供

OPRoS

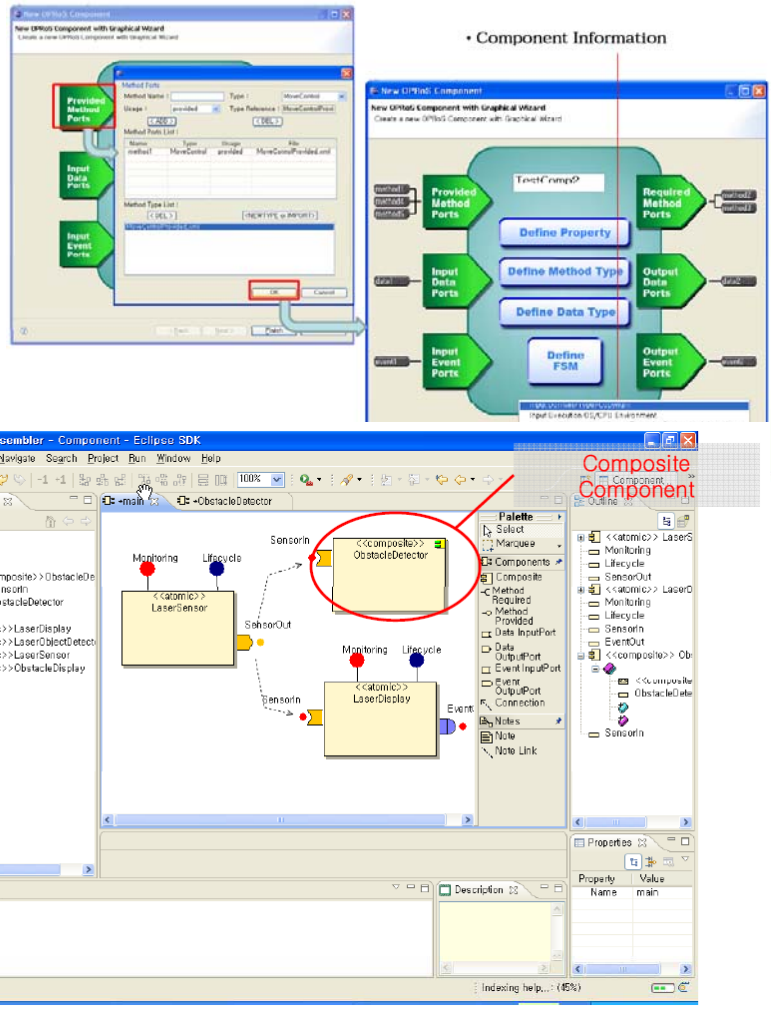


© ETRI, OMG Infra. WG, OPRoS Component Tools

OPRoS

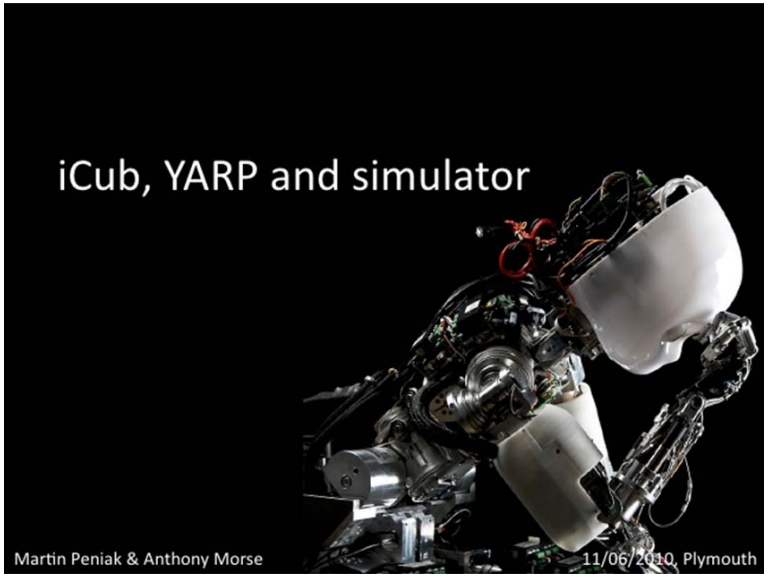
- コンポーネント開発
  - Component Editor
- システム構築
  - Component Composer

OpenRTM-aistに
よく似たツールチェーン



© ETRI, OMG Infra. WG, OPRoS Component Tools

YARP

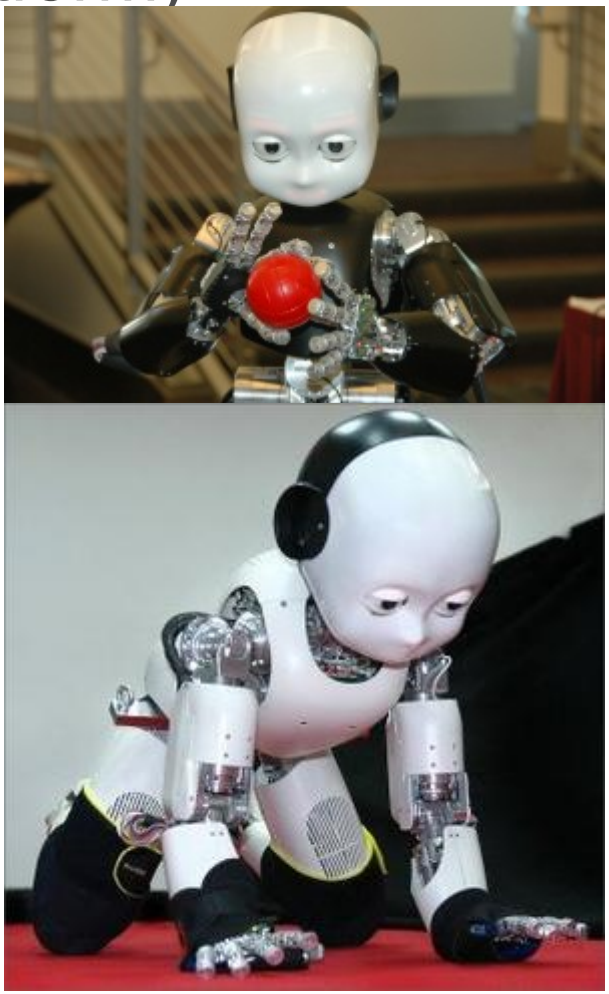


http://eris.liralab.it/yarp/

YARP

(Yet Another Robot Platform)

- IIT (Istituto Italiano di Tecnologia) で開発されたiCubのためのソフトウエアプラットフォーム
  - iCub: EUプロジェクト RobotCub,
  - 53DOFの赤ちゃんの様なヒューマノイド
- YARP

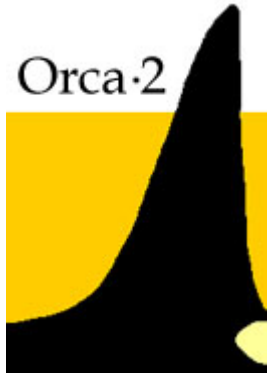


YARP

- コンポーネントフレームワークは無し
  - Mainから書き始める
  - 原則1プロセス1モジュール
- 多様な伝送方式のデータポートを提供
  - TCP, UDP, multicast
  - Carrier: 様々なマーシャリング、プロトコルを利用可能
- 簡単なRPCもある
- 独自のマーシャリング方式
- ノード間の利用にはネームサービスを利用
- CUIツール: yarp
  - 接続制御、モジュール制御

ORCA

http://orca-robotics.sourceforge.net/



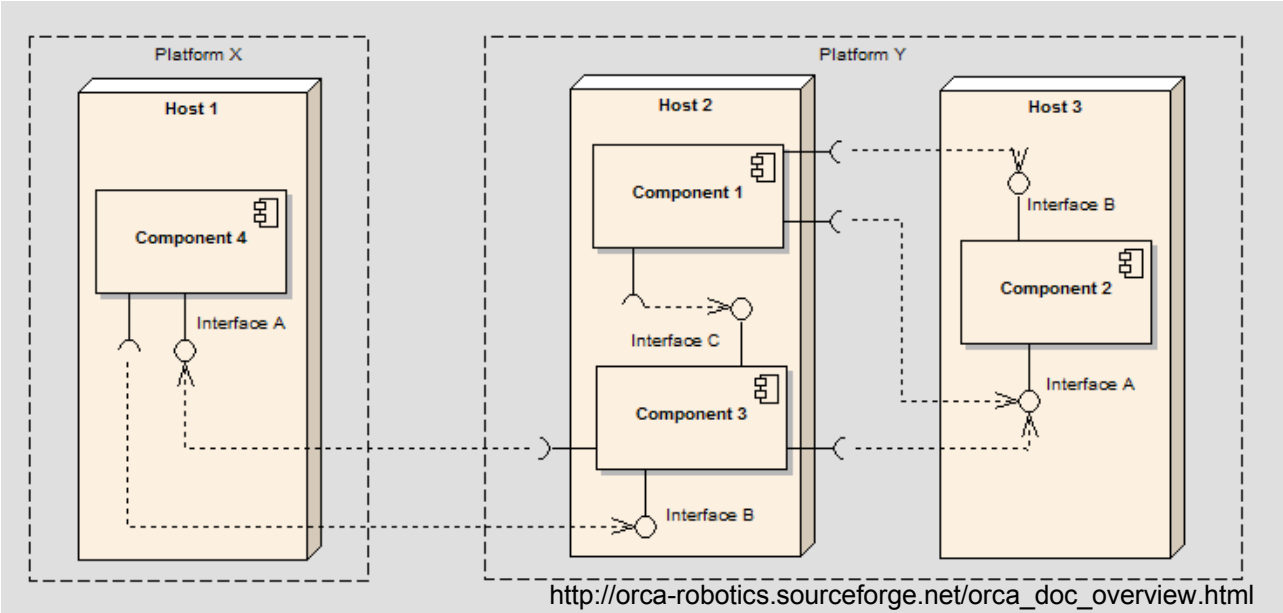
ORCA

- 開発元
  - KTH The Royal Institute of Technology (Stockholm, Sweden)
  - Australian Centre for Field Robotics (Sydney, Australia)
  - University of Technology Sydney (Sydney, Australia)
  - University of New South Wales (Sydney, Australia)
- コンポーネント志向ロボット開発をサポートするオープンソースのフレームワークを提供.
- プロジェクトのゴール:
  - 共通に利用可能なインターフェースを定義することによりソフトウエアの再利用を実現する
  - 効率的な上位APIを提供することにより、ソフトウエアの再利用性を向上させる
  - コンポーネントレポジトリを提供することによりソフトウエアの再利用を促進する
- http://orca-robotics.sourceforge.net/

- ORCAはOROCOSプロジェクトからの派生成果物
  - OrcaプロジェクトはEUのOROCOSプロジェクトから派生:
    - Open-Source RObotic COntrol System.
  - http://www.orocos.org/

ORCAの特徴

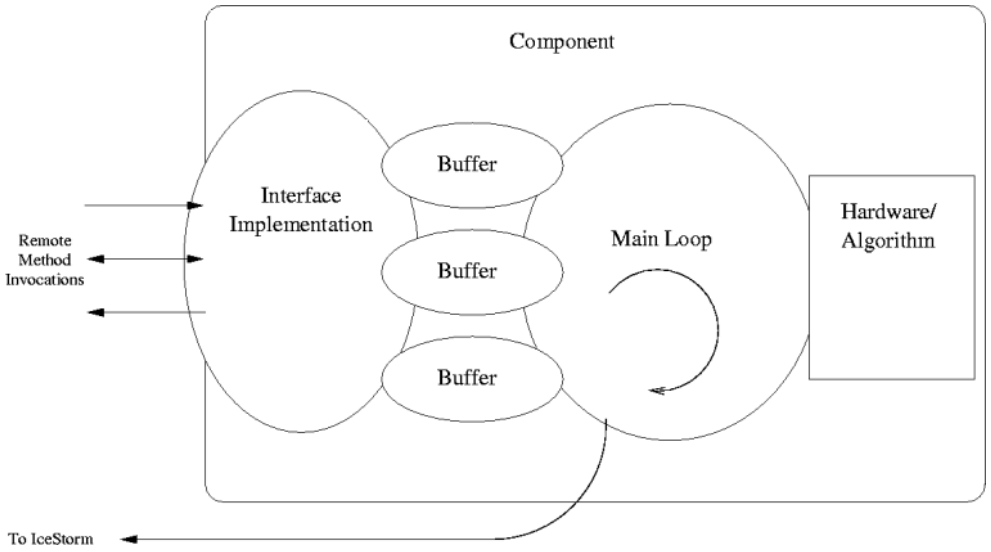
- CBSD (Component based software development)
- 分散オブジェクトミドルウエア
  - ORCA1: CORBA, CURDを利用
  - ORCA2: Ice (Internet Communications Engine)を利用
- GNU Lesser General Public License (LGPL)のもとで配布.



http://orca-robotics.sourceforge.net/orca_doc_overview.html

ORCAアーキテクチャ

- Component
  - orcaiceutil::Componentを継承
  - start(), stop() をコンポーネント開発者がオーバーライド.
- Interface Implementation
  - コンポーネント開発者がSlice(Iceのインターフェース定義言語：CORBAのIDLに相当)により定義し、実装する.
- Buffers
- Main Loop
- Hardware/Algorithm



http://orca-robotics.sourceforge.net/orca_doc_walkthrough.html

Microsoft Robotics Studio

http://www.microsoft.com/robotics/

Microsoft Robotics Studio

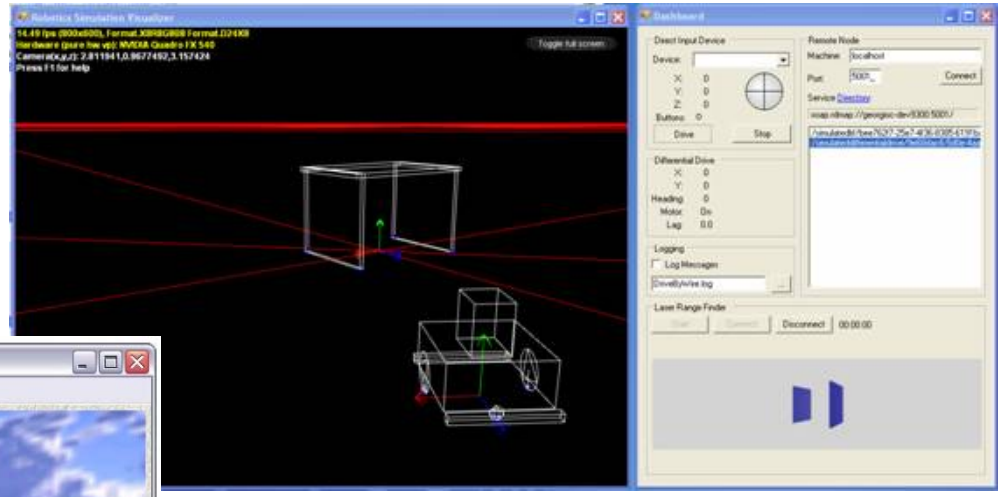
- 高レベルのビヘイビアやハードウエアのサービス志向の抽象化を提供
- 再利用性
- シミュレータエンジンを提供
  - AGEIA PhysXも使える
- ブラウザインターフェース
- Windowsのみで利用可能
  - ただし、ある種のハードウエアに対応
  - LEGO NXT, i-Robot, ActivMedia等

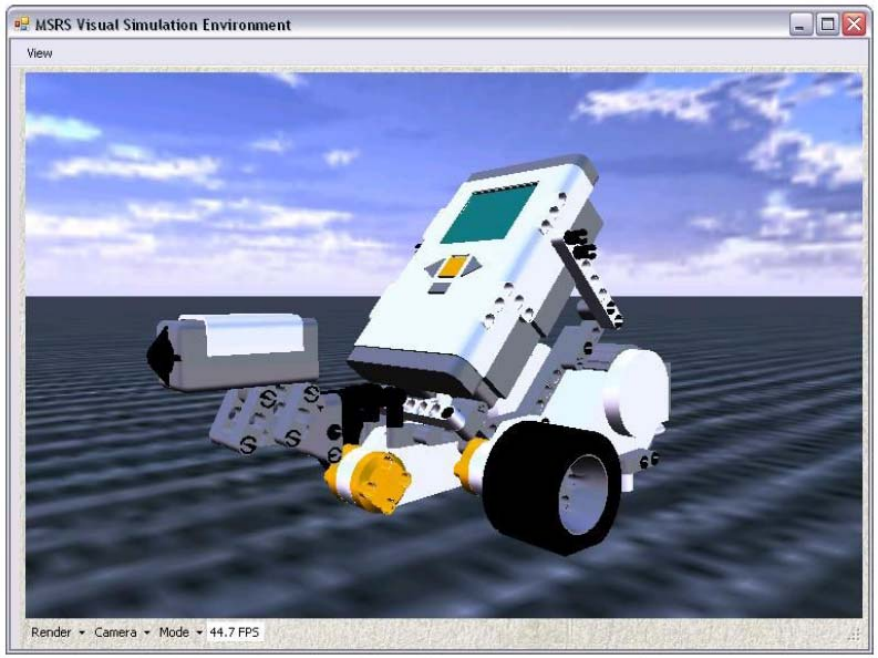
MSRSアーキテクチャ

DSSP：Decentralized Software Services Protocol

- DSSP: SOAPベースのプロトコルであり軽量なサービスモデルを提供する。
- DSSPはステート志向のメッセージオペレーションをサポート。
- アプリケーションはサービスの集合体として定義され、HTTPの拡張として提供される。
- サービスとは：アプリケーションのライフタイム中であれば、生成、操作、モニタリング、削除可能な軽量なエンティティ。
- サービス：ユニークなID、状態、振る舞い、コンテキストを持つ実体

MSRS シミュレータ




MSRS Visual Simulation Environment
View
Render
Camera
Mode
44.7 FPS

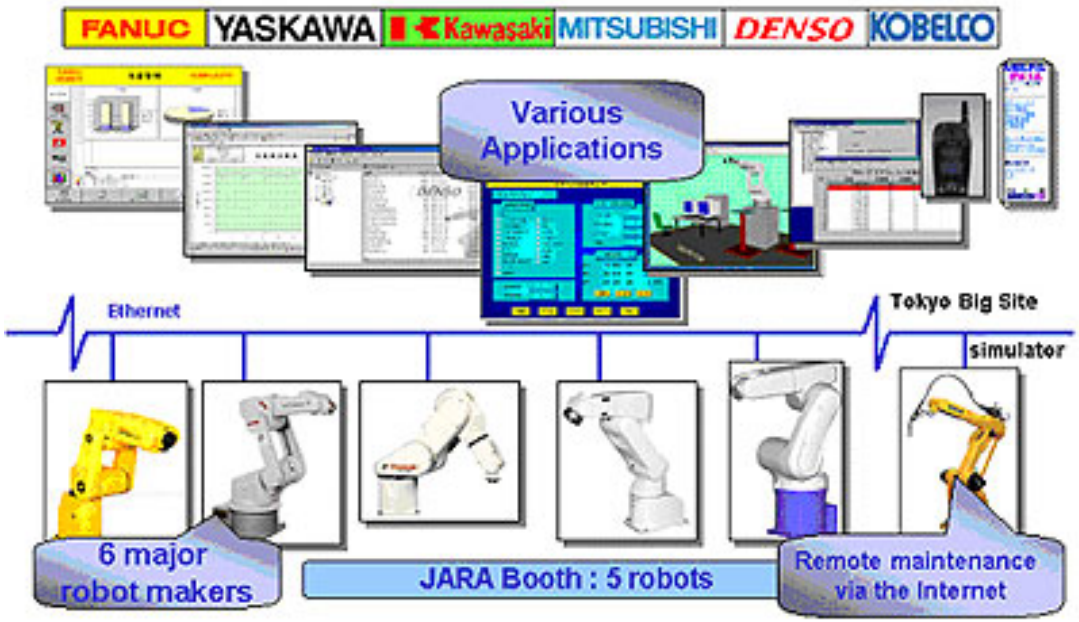
ORiN

http://www.orin.jp/

ORiN

- ORiN (Open Robot Interface for the Network)
  - 日本のFAロボット標準
  - FAロボットコントローラを抽象化
  - マルチベンダロボットシステムを容易に実現可能
  - メンバー:
    - ORiN consortium

    (FANUC, YASUKAWA, Kawasaki, MITSUBISHI, DENSO, KOBELCO)
    - http://www.orin.jp/

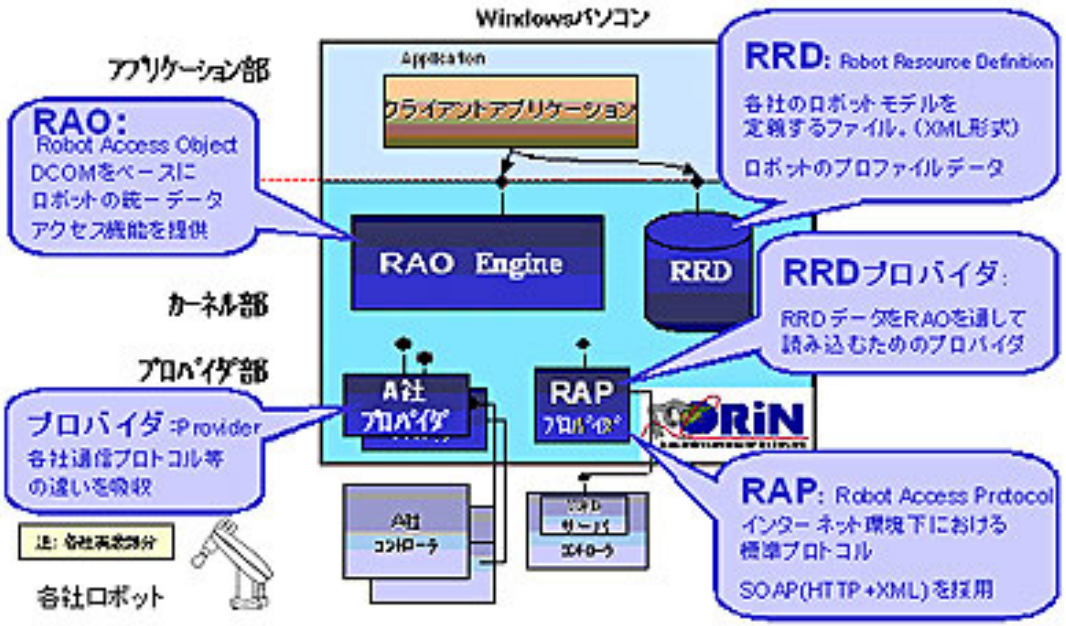


ORiN

- 設計ポリシ
  - 緩やかな標準化
    - 様々なタイプのロボット仕様を包含可能
  - 拡張性
    - ベンダ特有のオプションを定義可能
  - ネットワークプロトコルのモジュール化
    - 既存のロボットに適用可能
  - 実装と仕様の分離
    - OOP
- 実装ポリシ
  - デファクト標準
    - PC&Windows
  - Distributed object model (DCOM)
    - ネットワーク透過
    - 言語非依存
  - XML
    - ベンダ独自の仕様を記述するための標準フレームワーク
  - インターネット技術
    - HTTP, XML, SOAP

ORiNアーキテクチャ

- RAO (Robot Access Object)
  - ロボットコントローラに対する統一されたデータアクセシビリティを提供
- RRD (Robot Resource Definition)
  - ロボットプロファイルデータ
- RAP (Robot Access Protocol)
  - インターネットを介したアクセシビリティを提供



RSNP

（Robot Service Network Protocol）

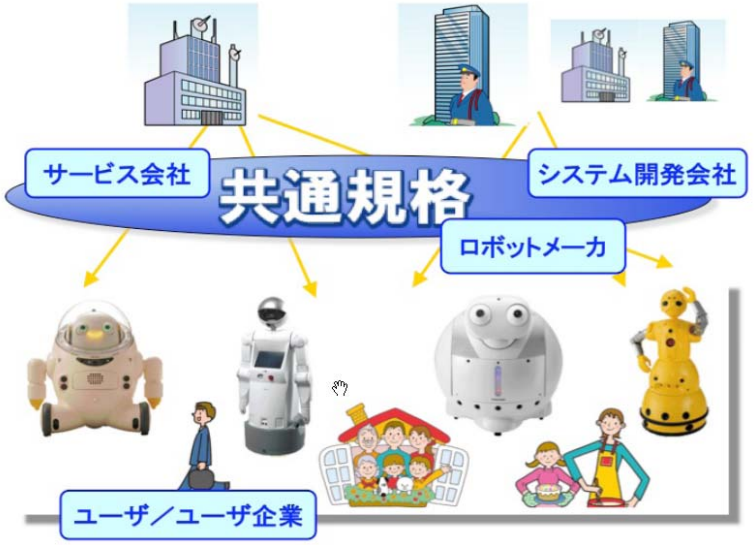
http://robotservices.org/

http://www.robotservices.org/wiki/jp/

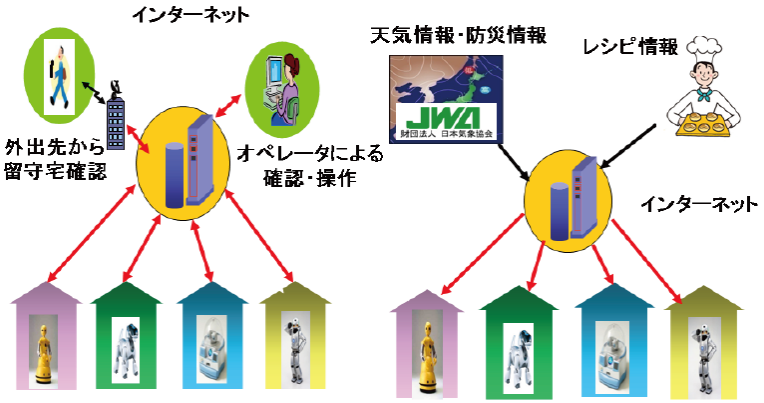
RSi
Robot Services initiative
ロボット サービス イニシアチブ

RSi (Robot Service Initiative)

- RSi (Robot Service Initiative)が主導となって定めるロボット用インターネットサービスのプロトコル
  - RSi:2004年発足
  - 富士通、三菱重工、東芝、安川電機、日本気象協会等が加盟
- ロボットをプラットフォームとしたインターネットと連携した新たなビジネス創出を目指す



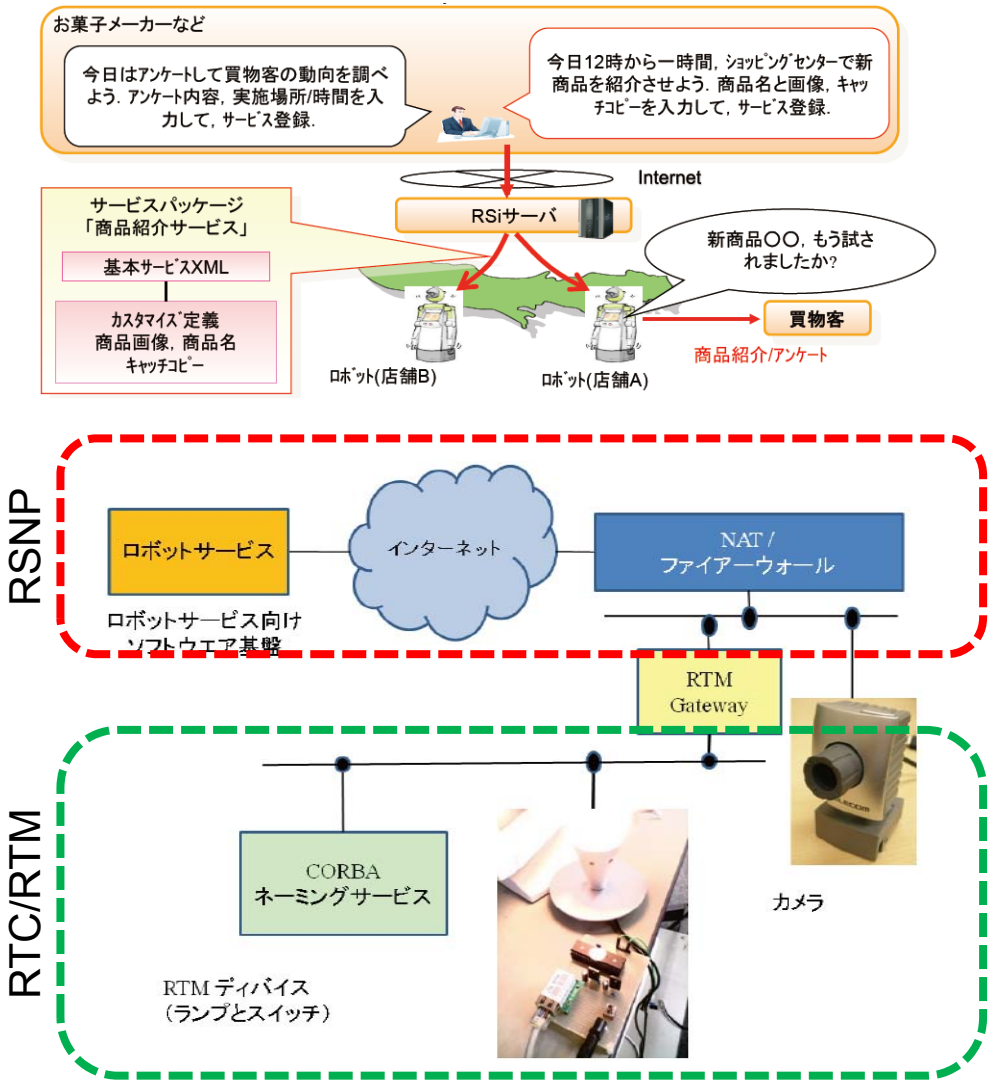
例1　遠隔操作によるサービス（見守りサービス等）　　例2　情報提供サービス



© 成田雅彦, 産業技術大学院大学
ROBOMEC2013 RTM講習会資料より

RSNP

- RSiサーバ等で提供するサービスを各地のロボットを介して提供
  - 天気予報、見守り、ロボットマップ、各種ロボット制御
- SOAPを利用
- 疑似Push機能を利用しFirewall越し通信を実現
- 主としてロボットとインターネット(クラウド)との連携に利用
  - cf. RTC



© 成田雅彦, 産業技術大学院大学 ROBOMEC2013 RTM講習会資料より

UNR Platform

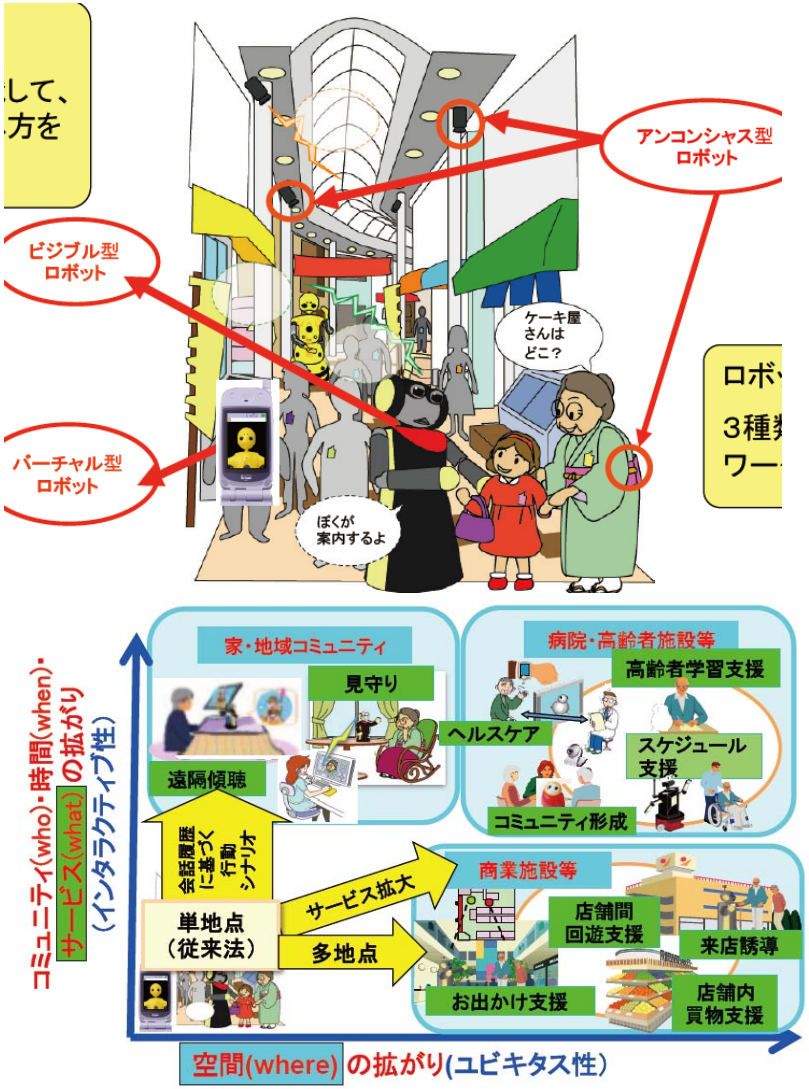
（Ubiquitous Network Robot Platform）

http://www.irc.atr.jp/std/UNR-Platform.html

ATR Advanced Telecommunications Research Institute International

UNR Platform

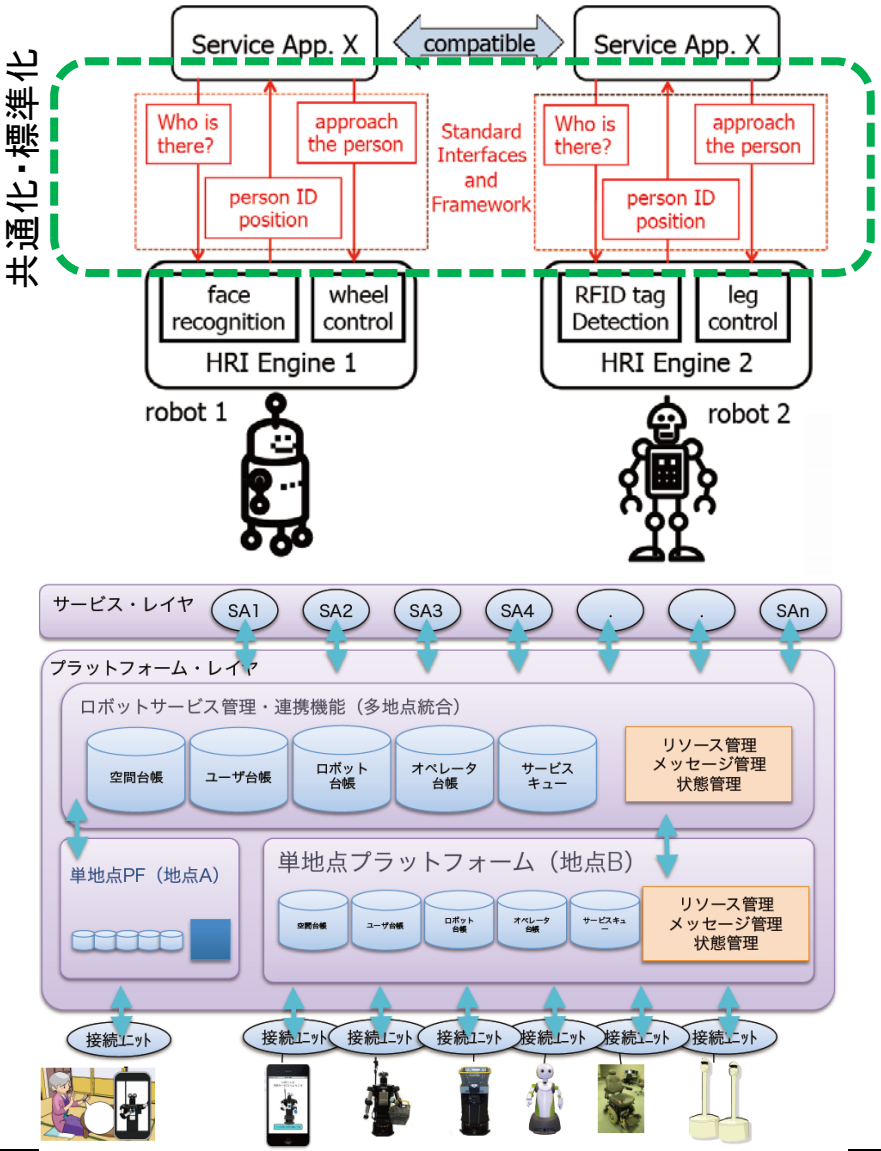
- 商業施設・病院・家などのさまざまな場所における 人々の活動を支援
- ロボット、スマートフォンアプリ、環境センサがネットワークを介して連携し、多地点で人々にサービスを提供することを目指す
- ARTにより開発、配布



© 亀井剛次 ATR, CNR研究会2013

UNR Platform

- 一部はOMG RoIS (Robot Interaction Service)標準に準拠
- 個々のロボット仕様を気にせずアプリケーションを記述可能
- アプリと下位コンポーネントのデータのやり取りを仲介

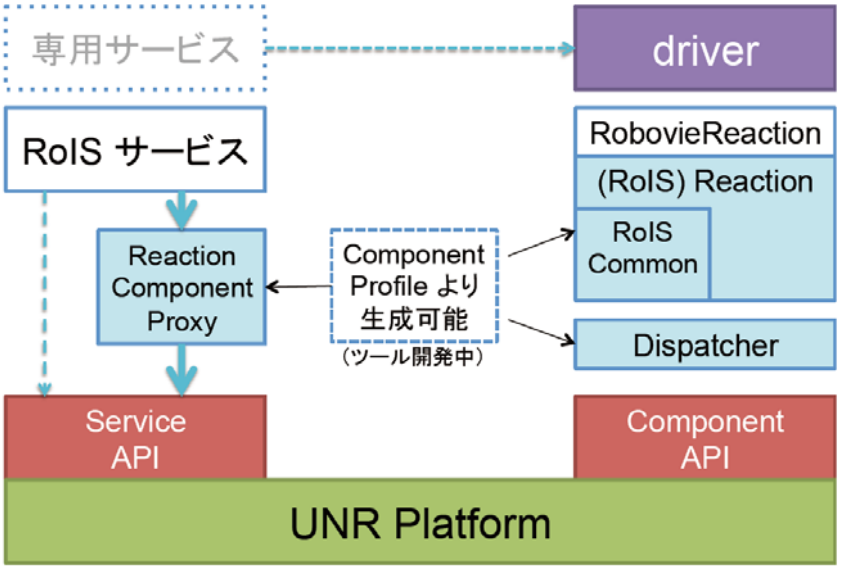


UNR Platform

- RoIS: ロボット対話サービスに必要な機能を標準化
- UNRプラットフォームは各種RoISサービスをクライアントの要求に応じて仲介
- ロボット側はRTMやROSなど何を利用してもよい

RoIS機能コンポーネント群

1）システム情報	9）音声認識
2）人検出	10）ジェスチャ認識
3）人位置検出	11）音声合成
4）個人同定	12）応答動作
5）顔検出	13）ナビゲーション
6）顔位置検出	14）追従
7）音検出	15）移動
8）音源位置検出	



Comparison of open frameworks

	Target	Open spec/source	Real-time	Language	OS	modularity	communication
OpenRTM-aist	Universal	○/○ OMG RTC	○	C++, C, Python, Java, .NET, Android	UNIX, Mac OS X, Windows, uITRON, QNX, VxWorks	○ CBSD	CORBA
ROS	Universal	△/○	○	C++, Python, Java, LISP, Matlab	Linux, (OS X, Windows)	× Free style	original protocol
OROCOS	Universal	△/○	○	C++, (scripting: Lua)	Linux, Windows, Etc..	○ CBSD	Ice, CORBA
OPRoS	Universal	○/○ OMG RTC	○	C++	Windows, Linux	○ CBSD	Original protocol
YARP	Humanoid/ Universal	△/○	×	C++, Java, Python, Lua, Matlab	Linux, Windows, Mac OS X	△ Free style	Original protocol
ORCA/ORCA2	Universal	△/○	○	C++, Python	RTLinux, Other	○ CBSD	CORBA, CURD, Ice
MSRS	Universal	△/△	×	.NET（C++,C#,VB,etc.）	Windows	△ SOA	DSS・SOAP

Comparison of open frameworks

	Target	Open spec/source	Real-time	Language	OS	modularity	communication
OpenRTM-aist	Universal	○/○ OMG RTC	○	C++, C, Python, Java, .NET, Android	UNIX, Mac OS X, Windows, uITRON, QNX, VxWorks	○ CBSD	CORBA
ORiN	FA robots	○/× ISO ORiN	×	C++	Windows	△ OOP	DCOM, CORBA
RSNP	Internet Service	○/× RSi RSNP	×	Java	Java VM	△ OOP	SOAP
UNR Platform	Internet Service	○/○ OMG RoIS/RLS	×	Java	Java VM	△ OOP	SOAP
PlayerStage	Mobile robot	△/○	○	C, C++, Tcl, LISP, Java, and Python	Linux, Etc..	△ PO	original protocol
OPEN-R	AIBO, SDR3X	○/×	×	C++	Linux	△ OOP	original protocol
Open Robot Controller Architecture	Universal	△/×	×	Java, Python	Linux, Etc..	△ OOP	HORB

標準化

標準の役割

- 単純化
- 互換性
- 伝達手段
- 記号の統一
- 経済効果
- 安全・生命・健康の確保
- 消費者利益保護
- 消費社会の利益保護
- 貿易障壁の除去

新宅純二郎 他, ``コンセンサス標準戦略—事業活用のすべて'', p.3, 日本経済新聞出版社, 2008, ISBN978-4-532-31409-5

標準の類型

- 合意標準
  - デジュール標準
    - 国際機関標準：ISO、IEC等
    - 国家標準：JIS、JAS等
  - コンセンサス標準（もともとは国際標準を指す用語）
    - 業界標準：業界団体主導
    - コンソーシアム標準：興味のある企業連合、複数分野
    - フォーラム標準：現在ではコンソーシアム標準と区別が難しい
- 単独標準
  - デファクト標準
    - デファクト標準のオープン化
    - 狭義のデファクト：1社独占

OMG標準

複雑化した現代的システムではデファクトによる市場独占は困難である

このほか、公開・非公開による分類も可能であり、かつ類型の境界があいまいな標準もあるため、単純には分類できない。

なぜ標準が必要？

RTミドルウエアの目的

- 人々の間で共有される共通ソフトウエアモデル
- オープンな仕様を提供
  - 誰でも実装可能→実装の多様性
  - 仕様を策定することが主たる目的
  - 実装(OpenRTM-aist)：仕様の妥当性検証
- 実装技術に非依存なソフトウエアモデル
  - 特定の言語、OS、分散オブジェクトミドルウエアに依存しないモデル（PIM: Platform Independent Model）
- 標準化された仕様
  - OMG (Object Management Group) における標準化
  - オープンな標準化プロセス

インターフェース標準の特徴

- 標準化する範囲が狭い＝差別化領域を確保
- 製品標準化による差別化困難というデメリットが起こりにくい
- インターフェース両側ともにプロダクトイノベーションを継続可能
- 互換性の確保によるユーザの安心感
- 市場拡大効果はそれほど大きくないが、市場と市場を接続する効果がある

新宅純二郎 他, ``コンセンサス標準戦略—事業活用のすべて'', p.29、日本経済新聞出版社, 2008, ISBN978-4-532-31409-5

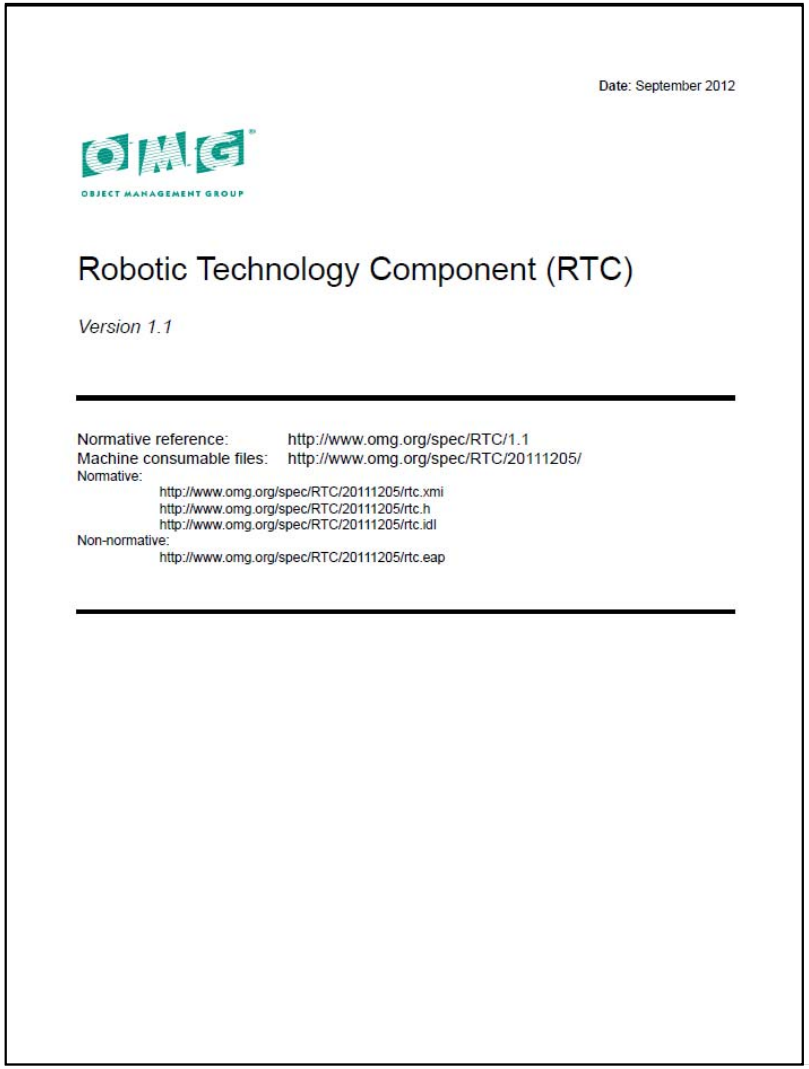
OMGにおける標準化

- OMG（Object Management Group）
  - ソフトウエア標準化団体
  - UML、CORBAなどの仕様策定で実績

- MDA
  - システムをPIM化することにより、抽象化されたモデルの寿命が延び実行可能性変数（品質、コスト、寿命の積）が向上する。（MDA: Model Driven Architecture の考え方。）
- PIM (Platform Independent Model)
  - プラットフォーム（ここでは、CORBA, JavaRMI, SOAP, HORB等分散オブジェクトプラットフォームを指す。)に依存しないモデル。
- PSM (Platform Specific Model)
  - プラットフォーム毎にPSMから変換されたモデル。
  - CORBA PSM, SOAP PSM etc…

OMG RTC 標準化

- 2005年9月
  RFP： Robot Technology Components (RTCs) 公開。
- 2006年2月
  Initial Response：PIM and PSM for RTComponent を執筆し提出
  提案者：AIST(日)、RTI(米)
- 2006年4月
  両者の提案を統合した仕様を提案
- 2006年9月
  ABにて承認、事実上の国際標準獲得
  FTFが組織され最終文書化開始
- 2007年8月
  FTFの最後の投票が終了
- 2007年9月
  ABにてFTFの結果を報告、承認
- 2008年4月
  OMG RTC標準仕様 ver.1.0公式リリース
- 2010年1月
  OpenRTM-aist-1.0リリース
- 2012年9月
  ver. 1.1改定



Date: September 2012

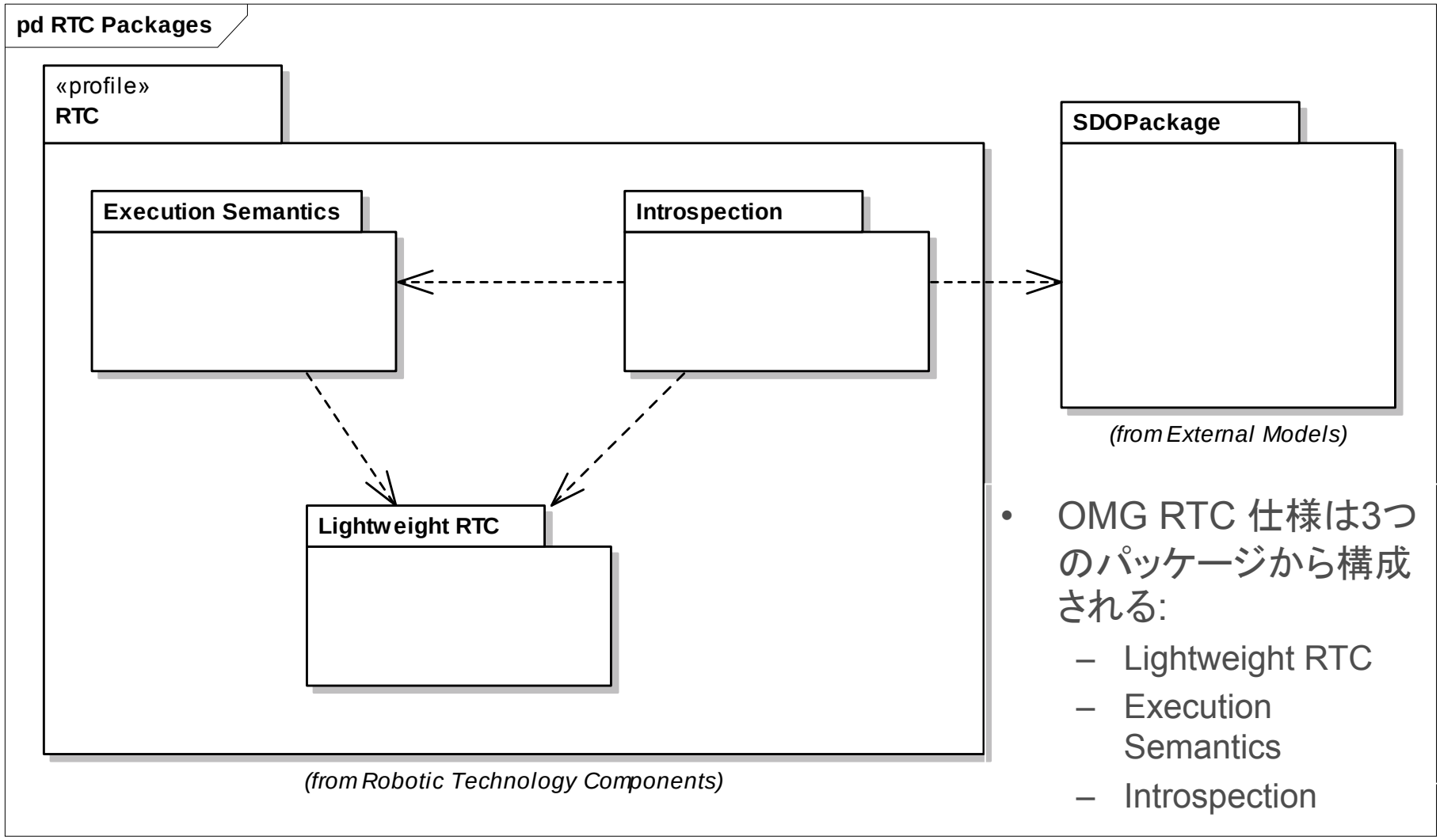
OMG
OBJECT MANAGEMENT GROUP

Robotic Technology Component (RTC)

Version 1.1

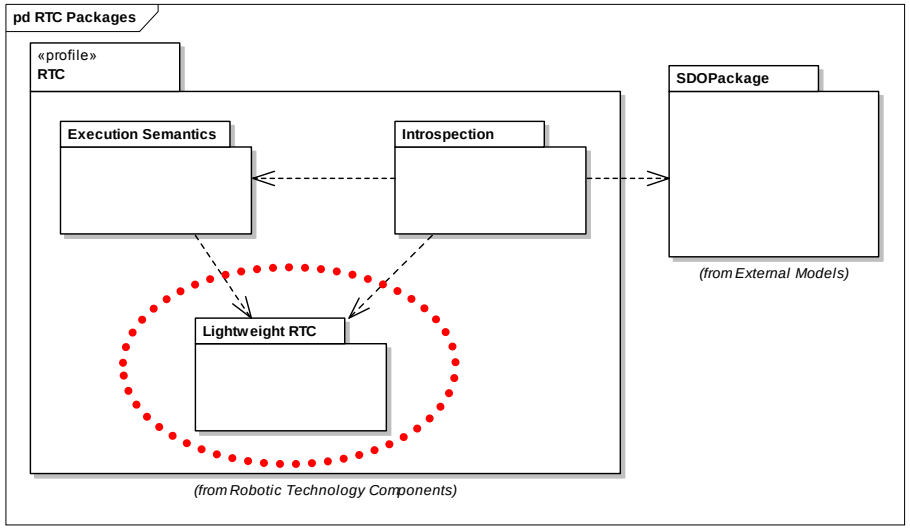
Normative reference: http://www.omg.org/spec/RTC/1.1
Machine consumable files: http://www.omg.org/spec/RTC/20111205/
Normative:
http://www.omg.org/spec/RTC/20111205/rtc.xmi
http://www.omg.org/spec/RTC/20111205/rtc.h
http://www.omg.org/spec/RTC/20111205/rtc.idl
Non-normative:
http://www.omg.org/spec/RTC/20111205/rtc.eap

PIM (Platform Independent Model)

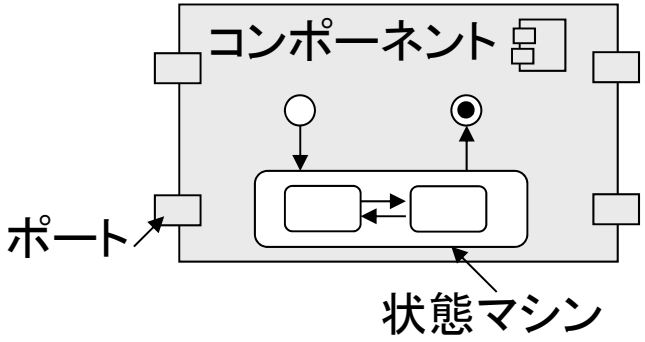


- OMG RTC 仕様は3つのパッケージから構成される:
  - Lightweight RTC
  - Execution Semantics
  - Introspection

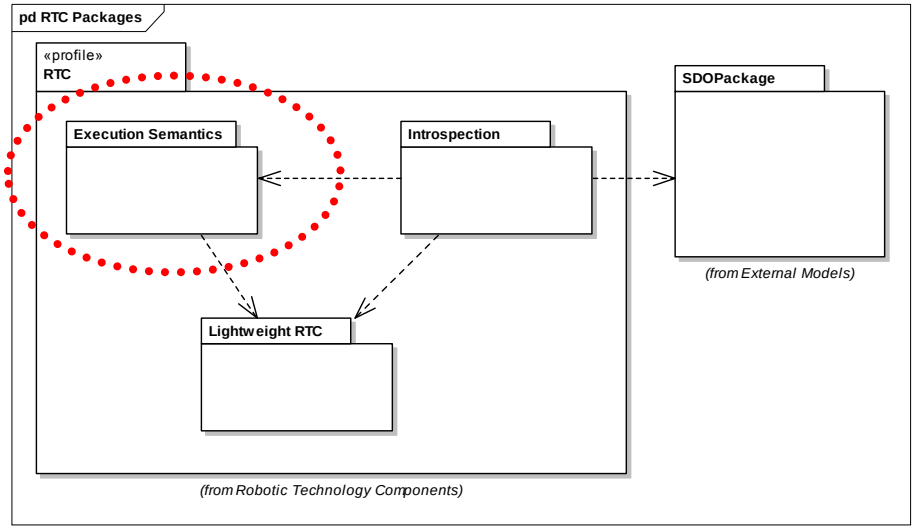
パッケージ1: Lightweight RTC

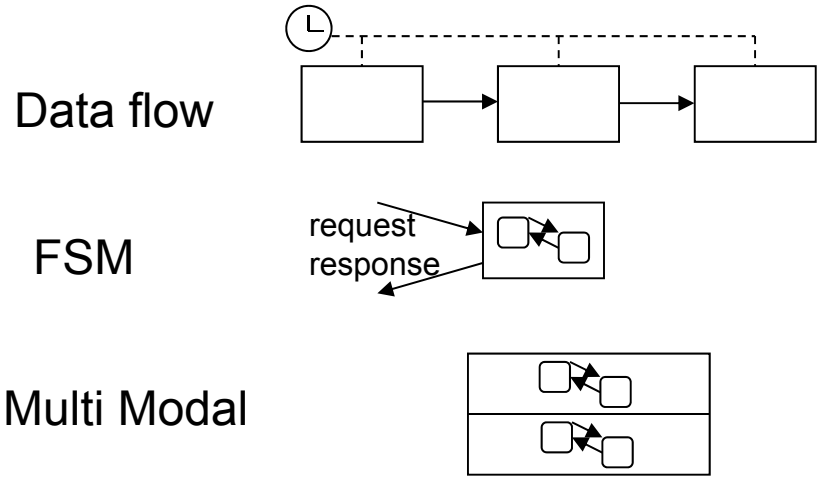


- Lightweight RTC
  - コンポーネント、ポート、コネクタ等のスレレオタイプ
  - コンポーネントライフサイクル
  - 実行コンテキスト
  - コンポーネントのメタ情報を取得するイントロスペクション機能は含まれない
  - 静的に構成されるコンポーネント



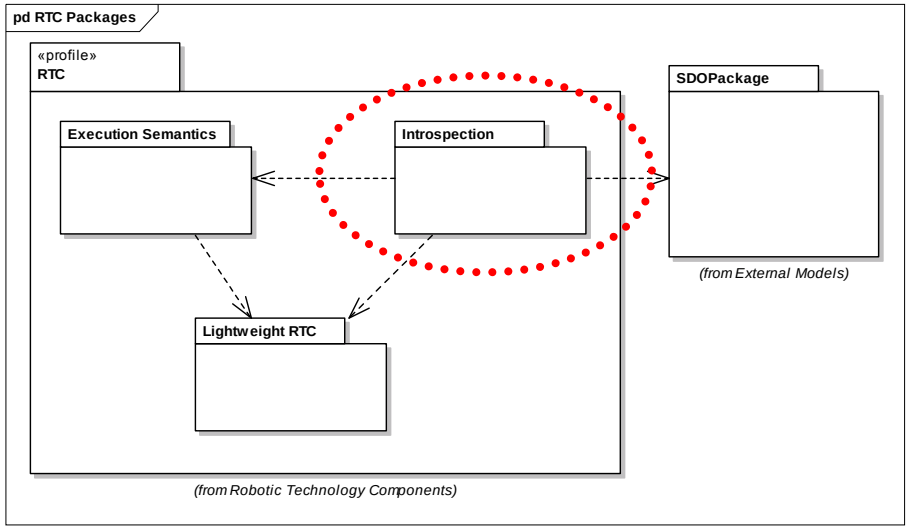
パッケージ2: Execution Semantics

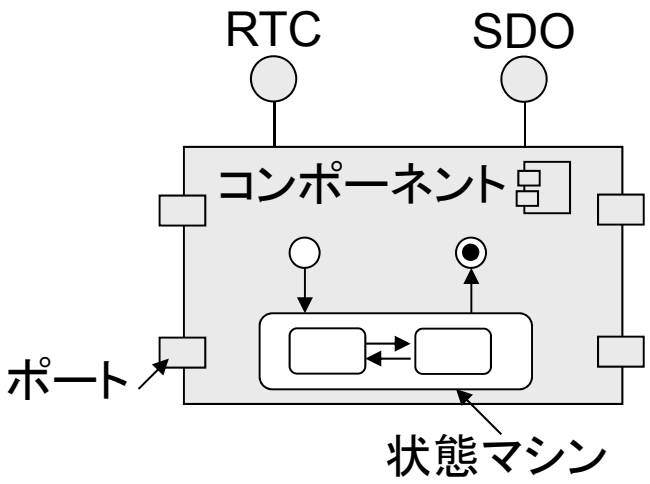




- Execution Semantics
  - ロボットシステム一般によく用いられる振舞いのパターンを提供する
  1. 時刻に同期して、データの流れにより駆動されるタイプ（データフロー型）
  2. Stimulus-response型あるいはイベントドリブン型の実行タイプ (FSM)
  3. 幾つかのモードを内包するモード型

パッケージ2: Introspection





- Introspection
  - コンポーネントのメタ情報取得のためのインターフェースを提供
  - 別のOMG標準であるSDO（Super Distributed Object）に準拠
  - 動的な構成（コンポーネント間の接続等）が可能なコンポーネント

OMG RTC ファミリ

名称	ベンダ	特徴	互換性
OpenRTM-aist	AIST	C++, Python, Java	---
OpenRTM.NET	SEC	.NET(C#,VB,C++/CLI, F#, etc..)	◎
RTM on Android	SEC	Android版RTミドルウエア	◎
H-RTM (仮称)	本田R&D	OpenRTM-aist互換、FSM型コンポーネントをサポート	◎
RTC-Lite	AIST	PIC, dsPIC上の実装	○(ブリッジ)
miniRTC, microRTC	SEC	CAN・ZigBee等を利用した組込用RTC実装	○(ブリッジ)
RTMSafety	SEC/AIST	機能安全認証 (IEC61508) capableなRTM実装，商用	○(ブリッジ)
RTC CANOpen	SIT, CiA	CANOpen-RTCマッピングを定めたCiA 標準	○(ブリッジ)
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	×

同一標準仕様に基づく多様な実装により
- 実装(製品)の継続性を保証
- 実装間での相互利用がより容易に

終わりに

- RTMの目的、アーキテクチャ、応用
  - 共通プラットフォーム導入によるあらたなロボットビジネス市場の創生
- 現在のロボットソフトウエアの動向
- 標準化

詳しくは…

検索 openrtm

本日の資料はopenrtm.orgに掲載します。
http://openrtm.org/openrtm/node/5451

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:10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ミドルウエアコンテスト

- 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
  - エントリー〆切：9月ごろ
  - ソフトウエア登録：10月ごろ
  - 講演原稿〆切：10月ごろ
  - 発表・授賞式：12月ごろ
- 2012年度実績
  - 応募数：17件
  - 計測自動制御学会学会RTミドルウエア賞（副賞10万円）
  - 奨励賞（賞品協賛）：3件
  - 奨励賞（団体協賛）：10件
  - 奨励賞（個人協賛）：5件
- 詳細はWebページ： openrtm.org
  - コミュニティー→イベント をご覧ください

レポート課題(1)

1. ミドルウエアを利用したサンプルプログラムを示せ
   a. ロボットミドルウエアを一つ選び、データの送信を行う手順・方法を調べ説明せよ。結果として、コメントを付したソースコード(完全である必要はないが、データ送信に必要な最低限の部分を示すこと)を添付せよ。
   b. 同様に、データの受信を行う手順・方法を調べ説明せよ。結果として、コメントを付したソースコード(完全である必要はないが、データ送信に必要な最低限の部分を示すこと)を添付せよ。

コードに付記されたコメントを重視します。

レポート課題(2)

2. サービスロボットを設計せよ
   a. どのようなサービス・機能を提供するロボットか仕様を定めよ。
   b. そのロボットが有するハードウエア(センサ、アクチュエータ等)を仮定せよ。
   c. そのロボットの持つ機能のうち少なくとも一つを実現するために必要なソフトウエアコンポーネントを考え、データ・コマンドの流れとともに図示せよ。
3. 授業の感想

仕様とコンポーネントの整合性に注意。

Player/Stage and Gazebo

http://playerstage.sourceforge.net/

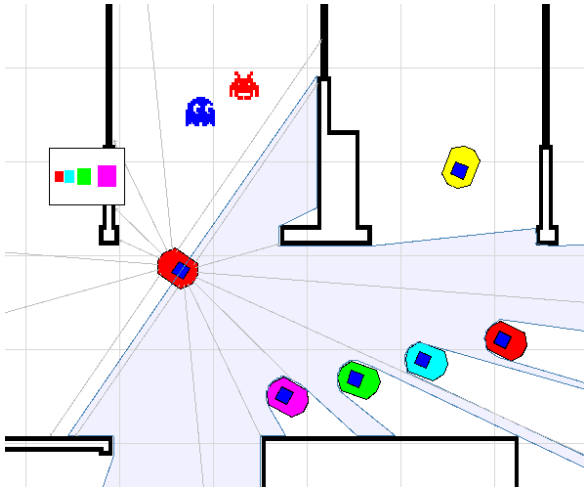
Player/Stage and Gazebo

- USC Robotics Lab
- Player
  - ロボット用ネットワークサーバ
  - ネットワーク経由でセンサ・アクチュエータにアクセスするためのインターフェースを提供
  - Playerは多くのロボットハードウエアをサポート
    - ex. ActivMedia Pioneer 2 family

“Player” は移動ロボット制御のための一連の共通インターフェースを提供するプラットフォーム

Player/Stage and Gazebo

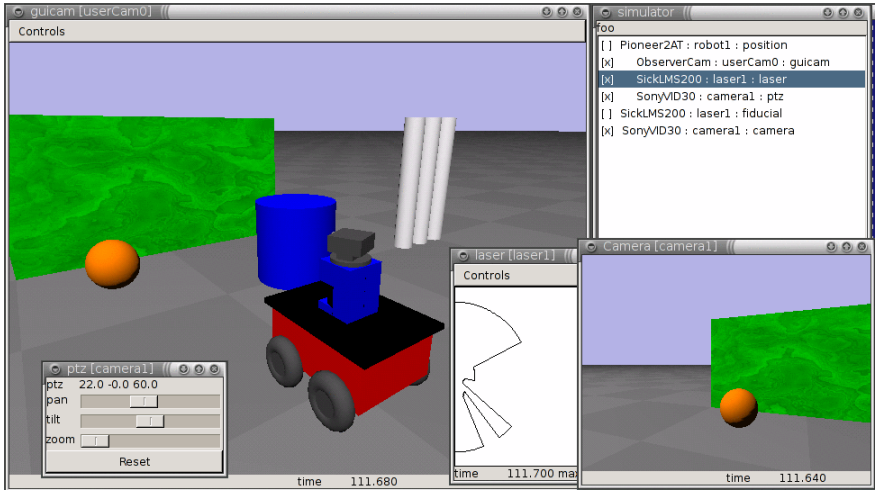
- Stage
  - 2次元のビットマップベースの環境において、移動ロボット、センサ、オブジェクト群をシミュレートする.
  - マルチエージェントシステムの研究のために設計されている.



“Stage” は2次元の移動ロボットシミュレーション環境を提供する.

Player/Stage and Gazebo

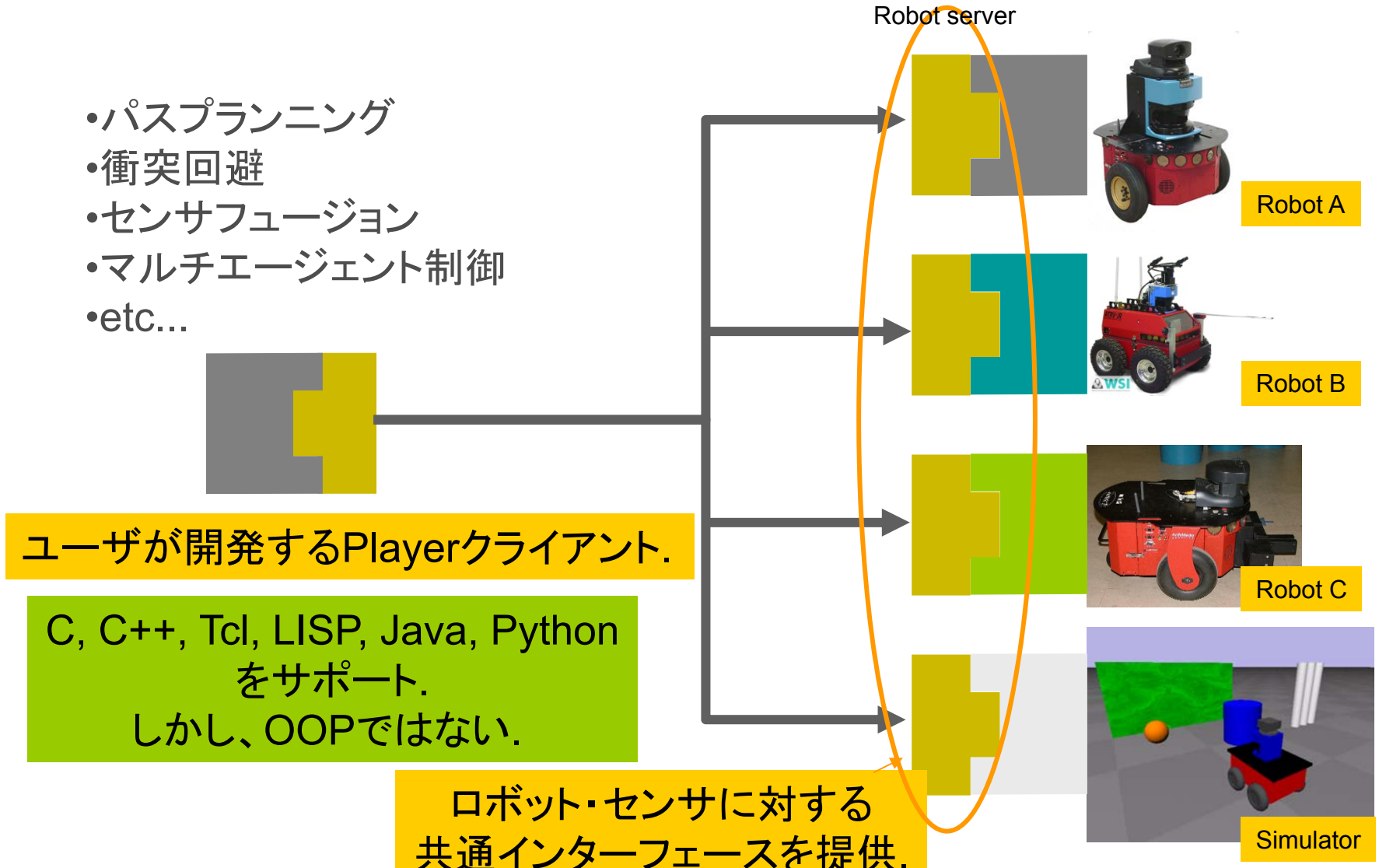
- Gazebo
  - 屋外環境のマルチロボットシミュレーション環境を提供.
  - 3次元環境のロボット、センサ、オブジェクト群のシミュレーション環境を提供する.
  - 現在はDARPAの支援を受けてOSRF†において開発を継続中



†OSRF: OpenSource Robotics Foundation

3次元移動ロボットシミュレーション環境を提供.

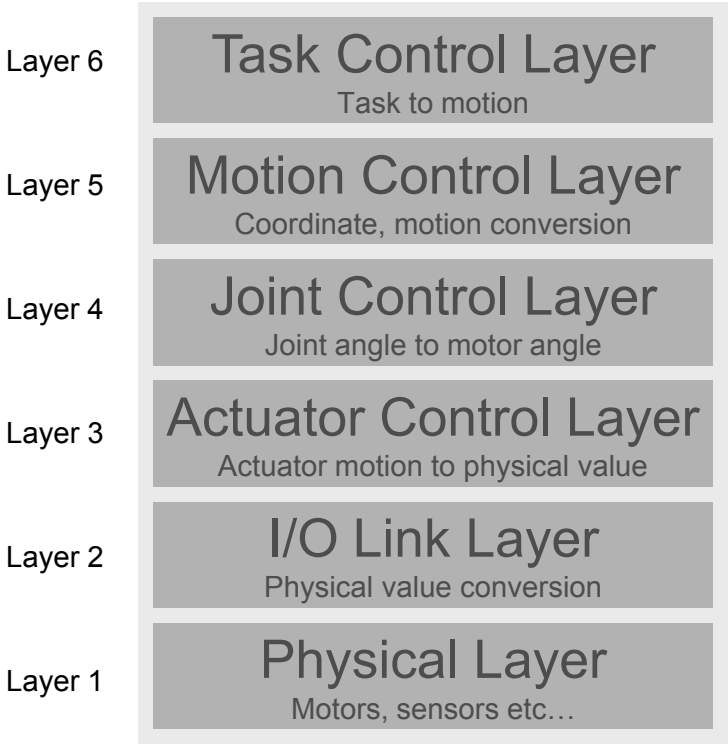
Player/Stage and Gazebo



Open Robot Controller Architecture

ORCA (Open Robot Controller Architecture)

- 開発元:東芝
- 6層 RT-RM
  (Robot Technology Reference Model)
- 分散オブジェクトミドルウエア:HORB上に構築
- Pythonベースのロボット制御言語を提供



HORB

- Java用の分散オブジェクトミドルウエア (ORB)
- 産総研の平野氏が開発
- 100% Java
- オープンソース
- 高速・軽量リモートメソッド起動
  - (JavaのRMI, CORBA: Visibrokerよりも速い)

ApliAlpha

(Advanced Personal Robotic Interface Type α)

- 東芝が開発したホームユースロボット
- アーキテクチャはORCAを採用
- ホームアプライアンスに対するインターフェースロボット
- 多くのコンポーネントが利用可能であり、交換可能。
- プラグアンドプレイ

ORCAの特徴

- オープンソースプロダクトのみ使用
  - Java, HORB, Python
- Pythonを利用したロボット制御言語
  - 多くのライブラリが利用可能
- OSやハードに非依存
  - Java, Python

OPEN-R

OPEN-R

OPEN-R

- 開発元：SONY
- AIBO, SDR-[3,4]X, QURIO等の制御アーキテクチャ
- 目的
  - スケーラビリティ
  - ポータビリティ
  - インターオペラビリティ
    - 異なるタイプのロボットに対して同じインターフェースを提供
  - スタイル・フレキシビリティ
    - 車輪型、4足歩行型、2足歩行型…

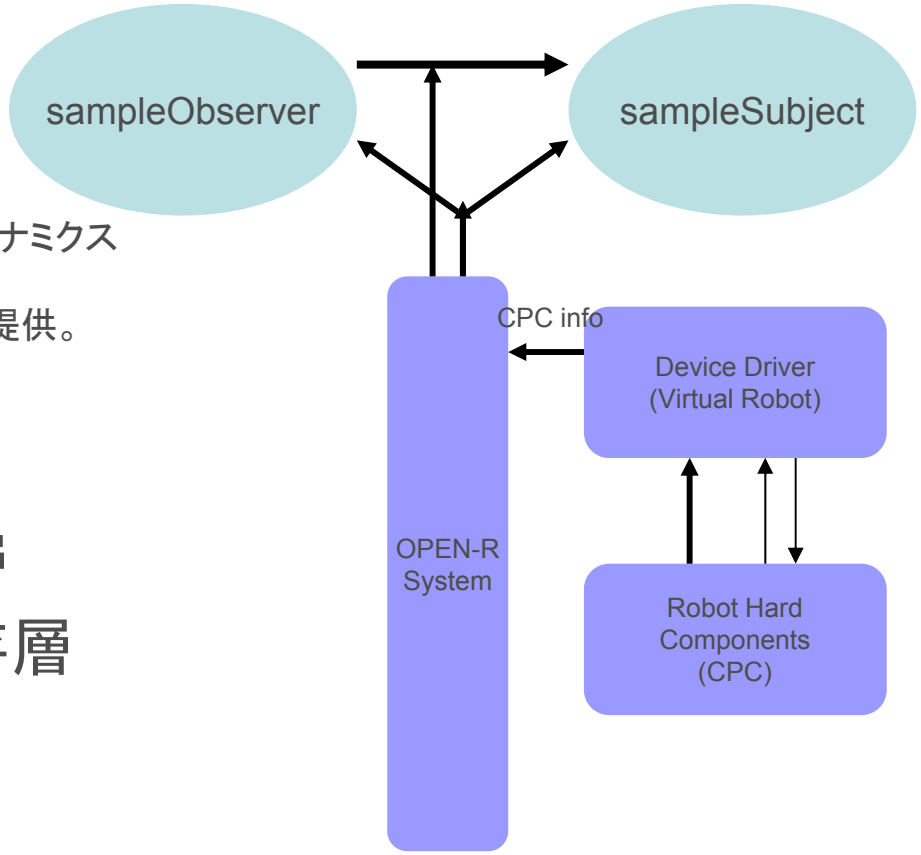
OPEN-Rアーキテクチャ

ハードウエアコンポーネント

- CPC (Configurable Physical Component)
  - 各コンポーネントは自身の機能、位置、ダイナミクスパラメータなどの情報を内蔵している。
  - ハードウエア自体がリフレクティブな機能を提供。

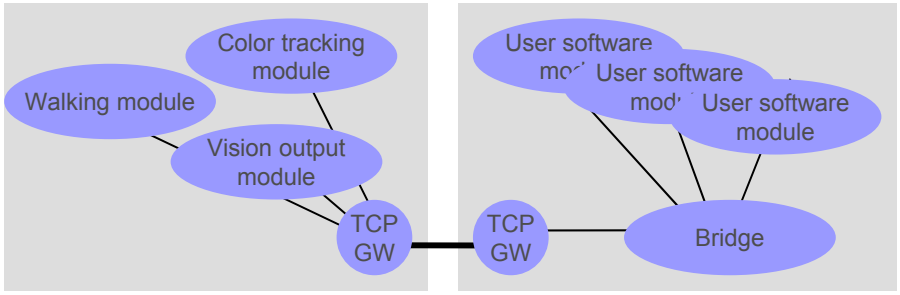
ソフトウエアコンポーネント

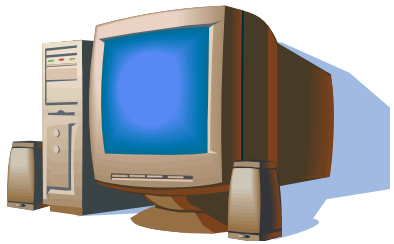
- コンフィギュレーション依存層
- コンフィギュレーション非依存層
- コンポーネント指向



OPEN-Rの特徴

- リモート処理
  - コードのポータビリティ
    - 組込システムとPC間
  - ネットワークを介したメッセージパッシング
  - ソフトウエアの再利用





- コンカレント開発環境
  - PCと組込み
- スケーラビリティ
- モジュール非依存性の確保
  - メッセージパッシング
- モジュラリティ
  - ハード＆ソフト

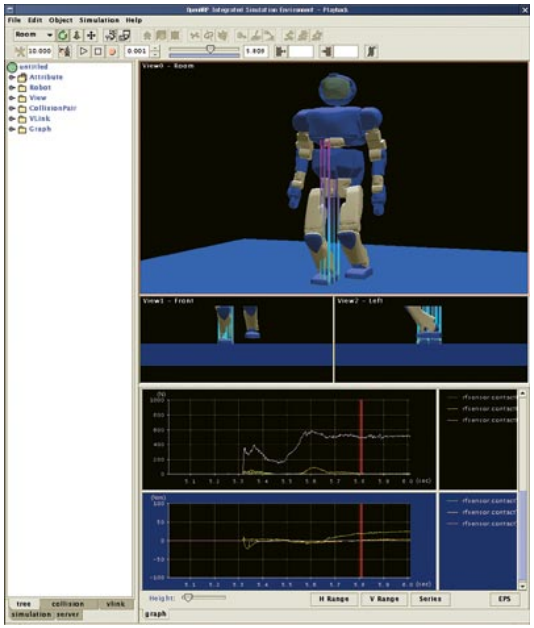
OpenHRP

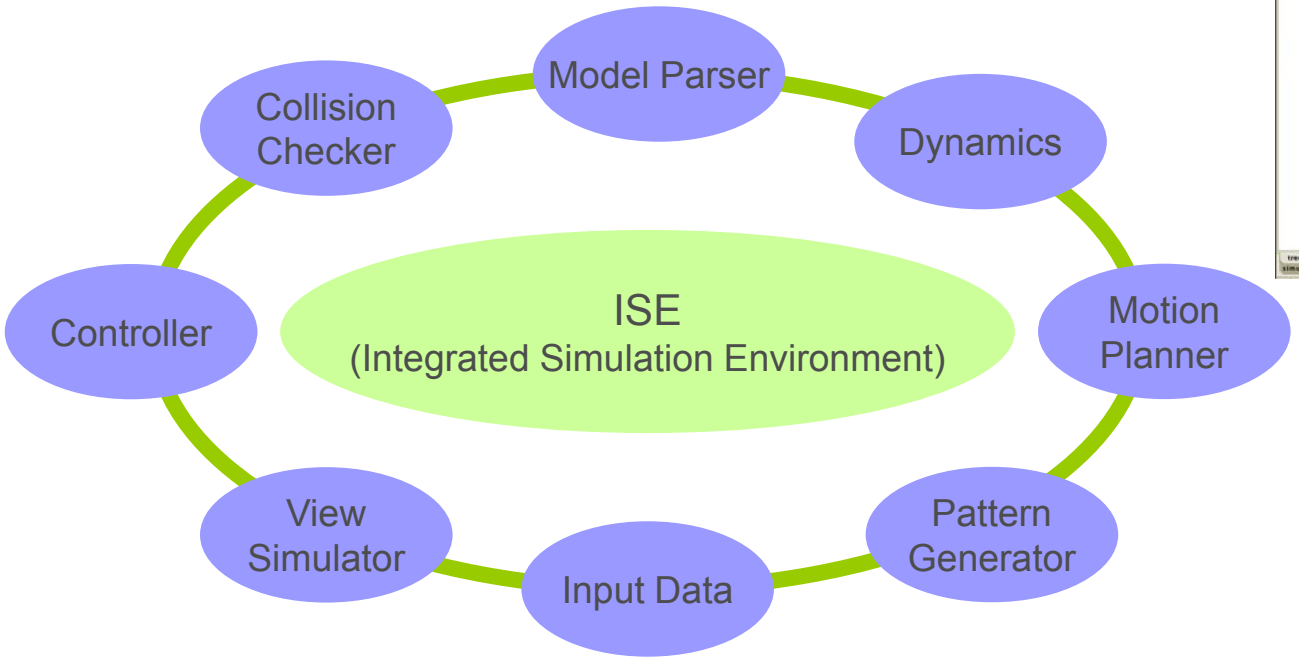
OpenHRP (Open architecture Humanoid Robotics Platform)

- 産総研, 東大, 川田工業.
- OpenHRPシミュレータ
  - ダイナミクスシミュレータ
  - 視覚情報シミュレータ
  - モデルエディタ (VRML)
- OpenHRPコントローラ
  - プラグイン
  - プラグインはシミュレータと実機で共通利用可能
  - コンカレント開発可能

ISE (Integrated Simulation Environment)

- いくつかのCORBAオブジェクトから構成
- 言語: Java & C++

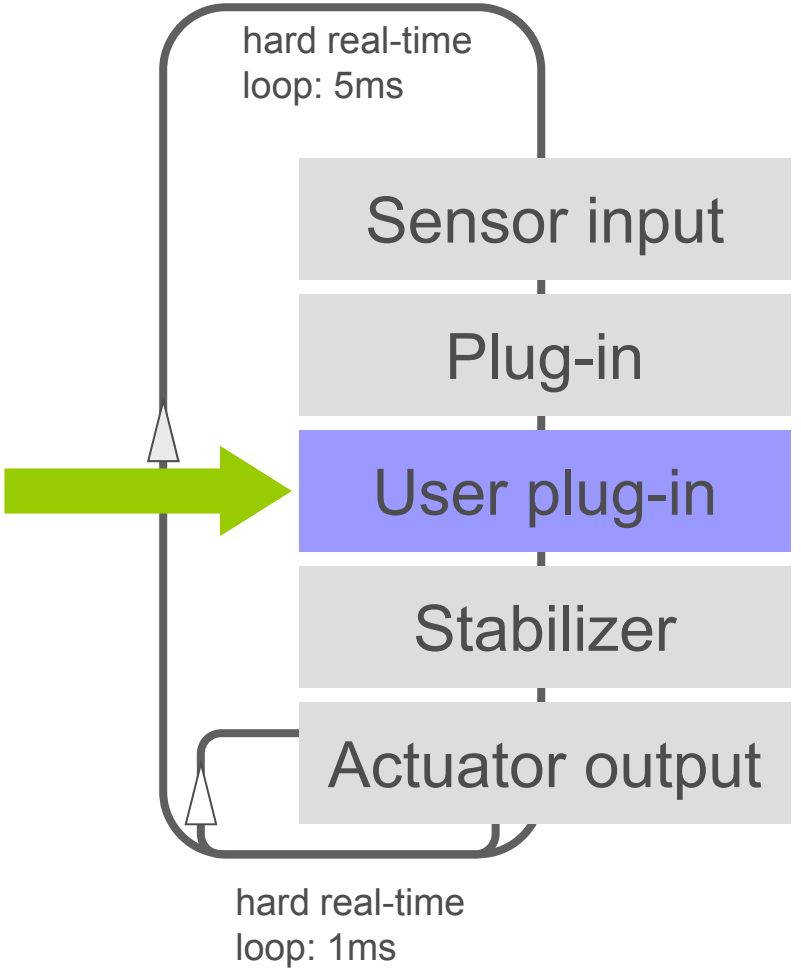




OpenHRPコントローラアーキテクチャ

- ARTLinux
  - ハードリアルタイムLinux拡張
  - 産総研（旧電総研）の石綿氏が開発
- 開発者はリアルタイム制御ループ内に開発したプラグインを挿入可能
- プラグインのダイナミックローディング
- サンプリングタイム: 5ms

```
class MyPlugin : public plugin
{
        MyPlugin();
        ~MyPlugin();
        bool setup(state *, command *);
        void control(state *, command *);
        bool cleanup(state *, command *);
}
```



OpenHRPのメリット

- ヒューマノイドロボットの統合開発環境
  - ダイナミクスシミュレータ、コリジョンディテクタ、視覚シミュレータ等
- CORBAによる分散システム
  - 負荷分散・フレキシビリティ
- リアルタイム制御アーキテクチャ
- プラグインアーキテクチャによるモジュラリティの向上