- Github Pages:
- https://onl.bz/BcUMtdA

May 1, 2024
Ubiquitous Robotics

MIDDLEWARE

About OpenRTM-aist
and
Outline of RT-Component Programming

National Institute of Advanced Industrial Science and Technology (AIST), Japan
Industrial CPS Research Center, Researcher
Nobuhiko Miyamoto, Ph.D.

Outline

- Basic concept and overview of RT-Middleware
- Activities of RT-Middleware community
- RTC development overview
- Conclusion

What is RT-Middleware?

What is RT?

- RT = Robot Technology cf. IT
  - not only standalone robots, but also robotic elements (sensors, actuators, etc.…)

RT-Middleware developed by AIST

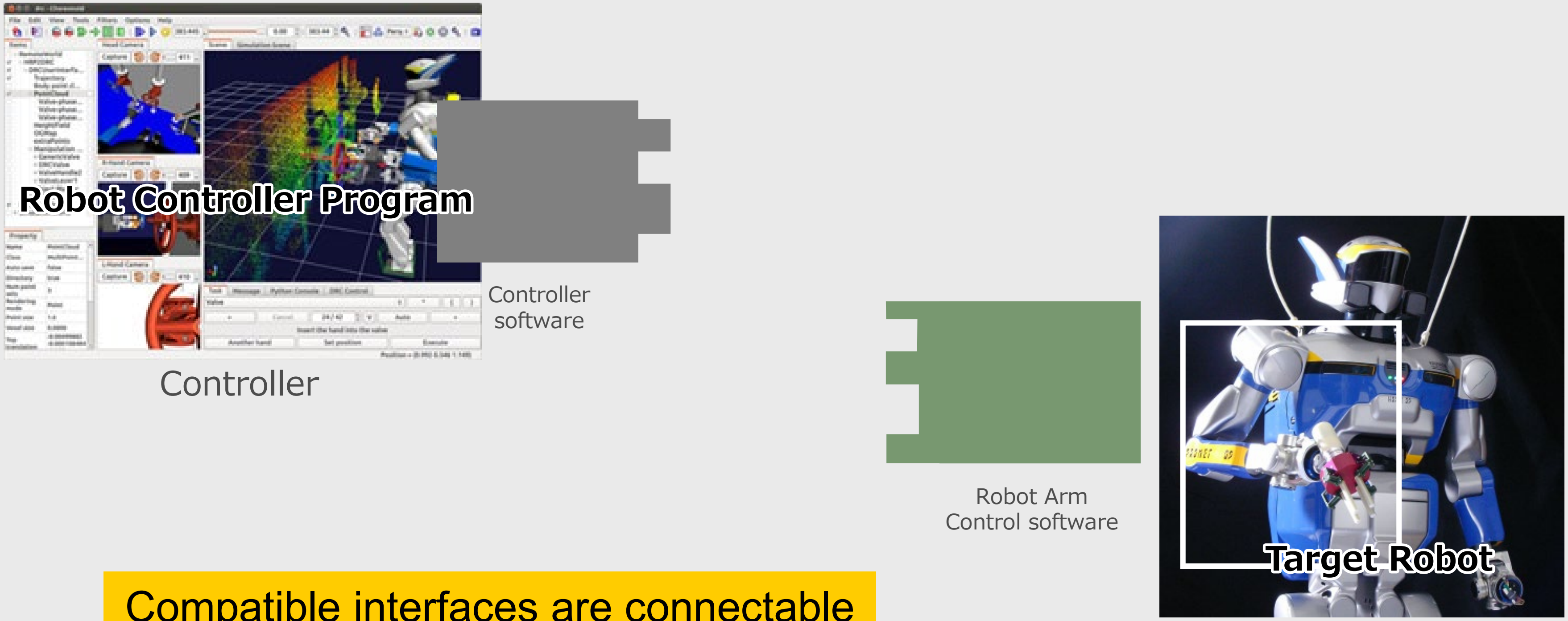
OpenRTM-aist

- RT-Middleware (RTM)
  - middleware and platform for RT-element integration
- RT-Component (RTC)
  - basic software unit in RT-Middleware

About Robot Middleware

- **Platform software** that provides common functions to streamline robot system construction
  - Sometimes called "robot OS"
  - Commonization and standardization of interface and protocols
  - Examples
    - Providing modular or componentized frameworks
    - Supports communication among modules
    - Provides parameter setting, deployment, startup, and module composition functions
    - Realize inter-OS and inter-language cooperation / interoperability by abstraction
- Development became active from around 2000
  - Various middleware is being developed and released all over the world

Conventional systems

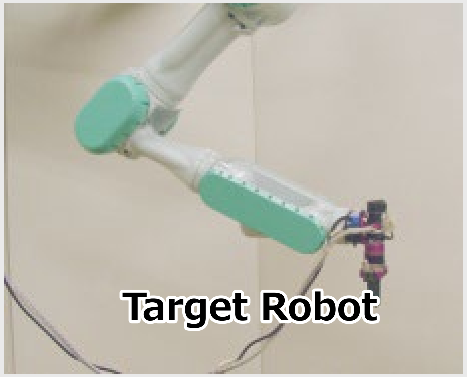


Robot Arm1

Compatible interfaces are connectable

Conventional systems

Humanoid's Arm
Control software



Robot Arm2

Controller
software

Controller

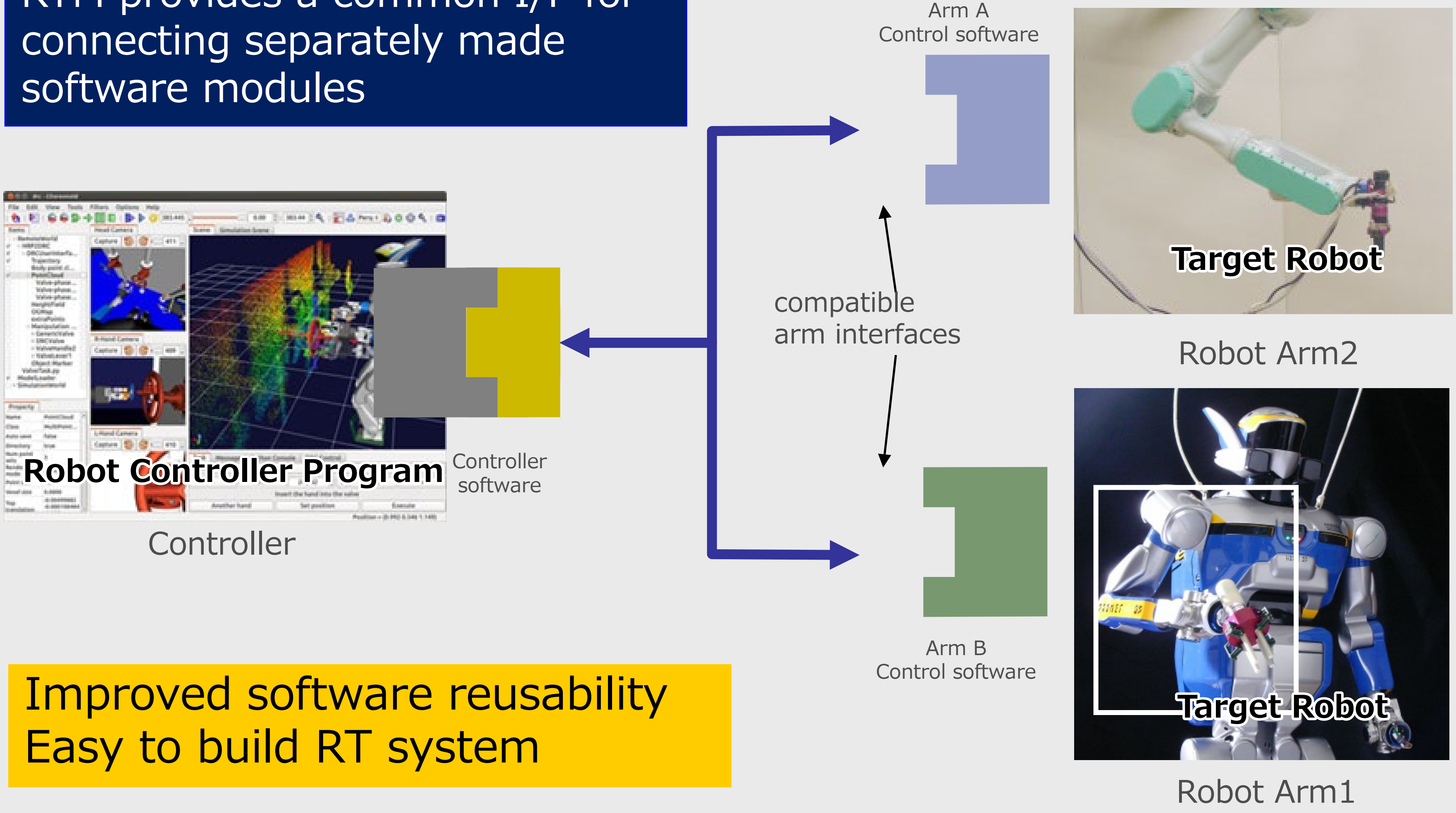
Robot Arm
Control software

Robot Arm2

Each robot has each control interfaces.
If no compatibility, cannot connect each other.

By using RT-Middleware

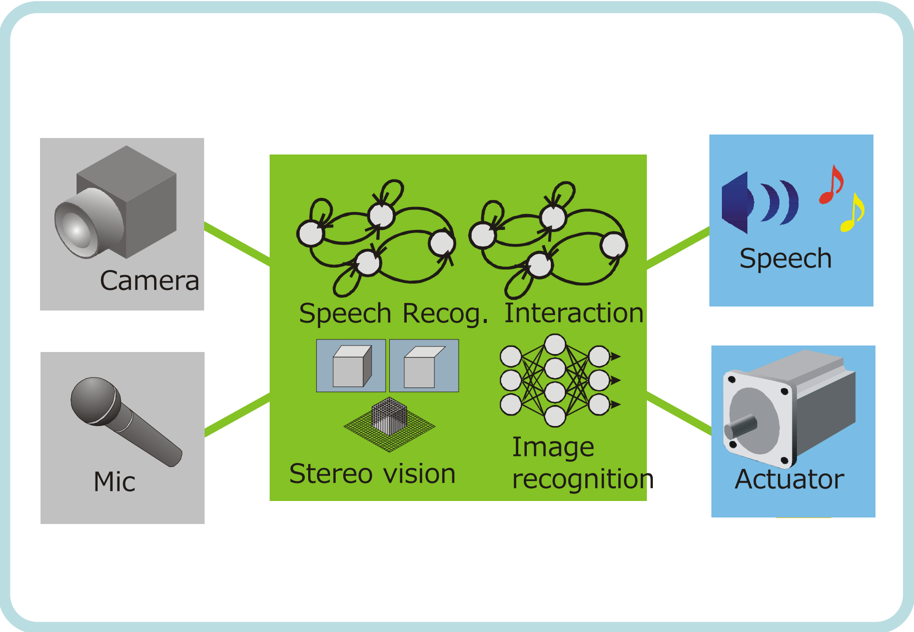
RTM provides a common I/F for connecting separately made software modules

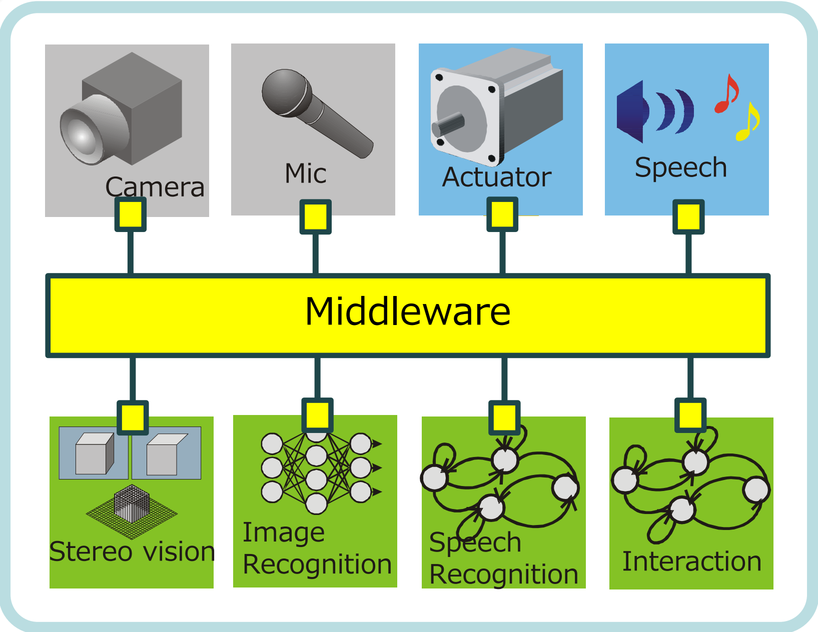


Improved software reusability
Easy to build RT system

Trend of Robot Software Development

Conventional Style → Component Oriented Development





- ✓ Integrated design of various functions
- ✓ High run-time efficiency, but inflexible
- ✓ Development becomes difficult as the system becomes more complex

- ✓ Division/integration of large-scale complex functions
- ✓ Improvement of development and maintenance efficiency (reuse of functions, etc.)
- ✓ Increased system flexibility

The benefits of modularization

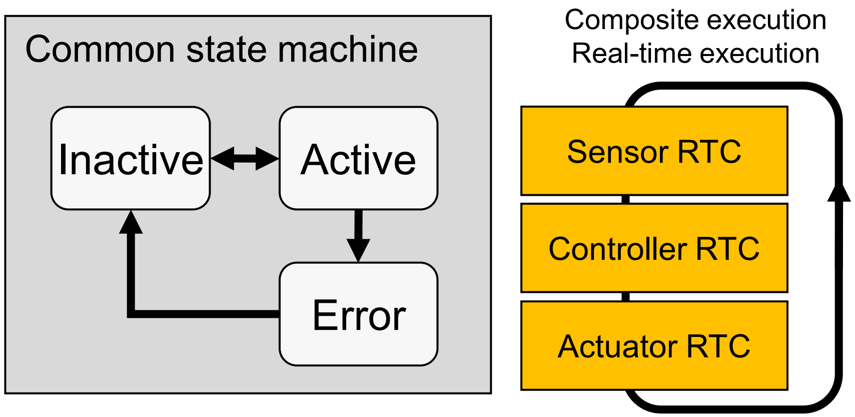
- Reusability
  - A component can be reused in various systems.
- Diversification
  - Various type of same functional modules can be tried in systems.
- Flexibility
  - System structure can be changed easily.
- Reliability
  - Easy to test a module and well tested modules are reliable.
- Durability
  - Well divided and independent module error does not affect too much to whole systems.

The benefits of RT-Component model

- Provides rich component lifecycle to enforce state coherency among components
- Defines data structures for describing components and other elements
- Supports fundamental design patterns
  - Collaboration of fine-grained components tightly coupled in time (*e.g*. Simulink)
  - Stimulus response with finite state machines
  - Dynamic composition of components collaborating synchronously or asynchronously

Main features of RT-Component

Activity, Execution context

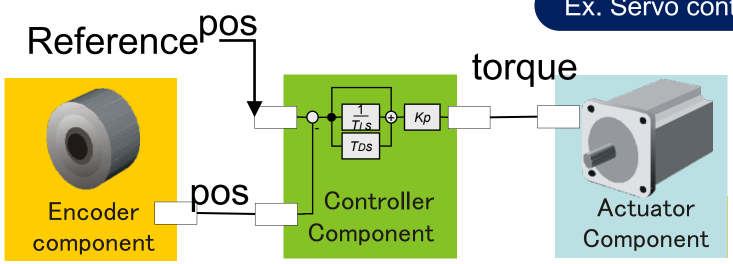


Life cycle management, core logic execution

Data Port

- Data centric communication
- Continuous data transfer
- Dynamic connection/disconnection

Ex. Servo control

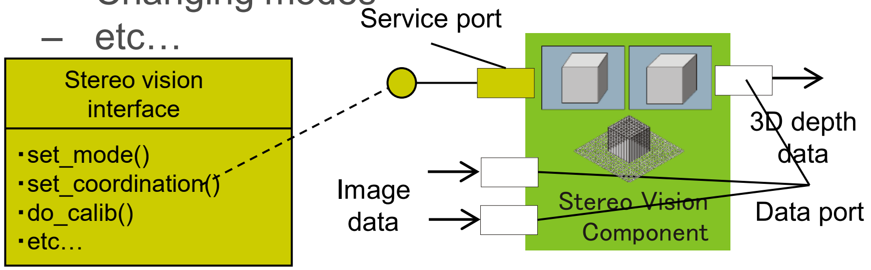


Data-centric communication

Service Port

- User defined interface
- Access to detailed functionality of RTC
  - Getting/setting parameters
  - Changing modes
  - etc…

Ex. Stereo vision

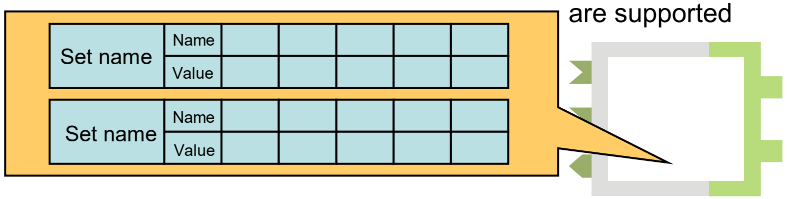


Service oriented interaction

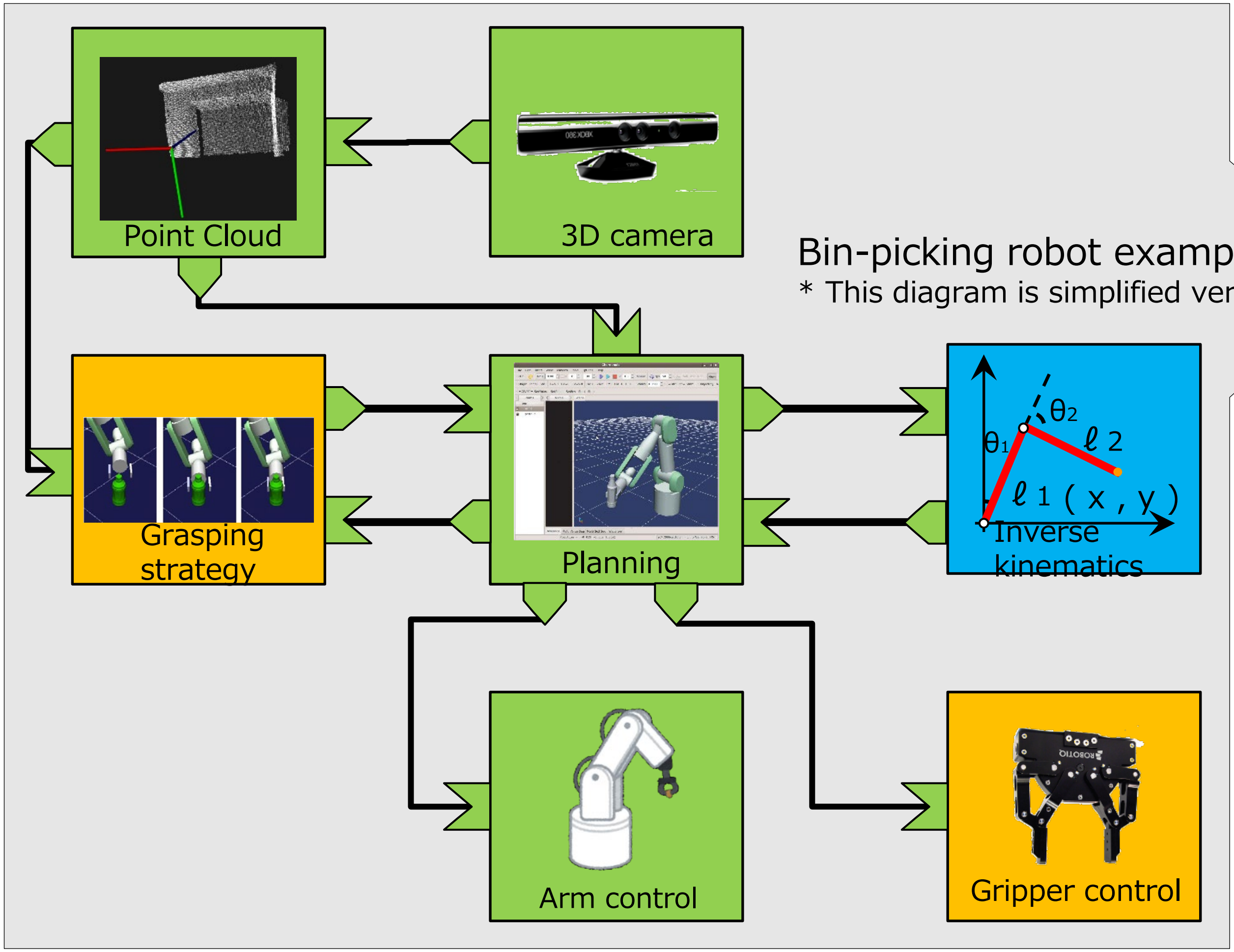
Configuration

- Function for internal parameter
- Multiple parameter sets
- They can be changed from remote in run-time

RTC can have several configuration sets. Runtime reconfiguration and dynamic switching are supported

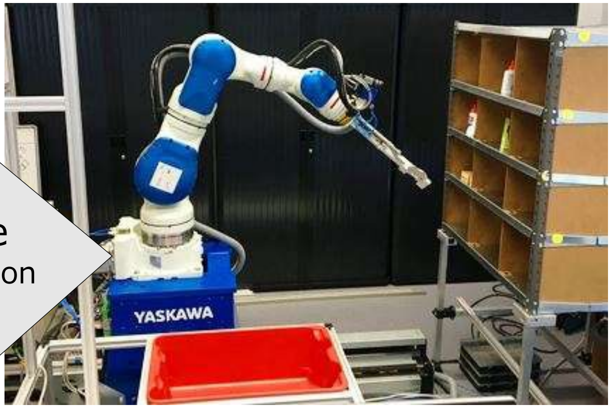


Advantages of RTM based development



Bin-picking robot example

＊ This diagram is simplified version



By using RTM, existing module can be reused

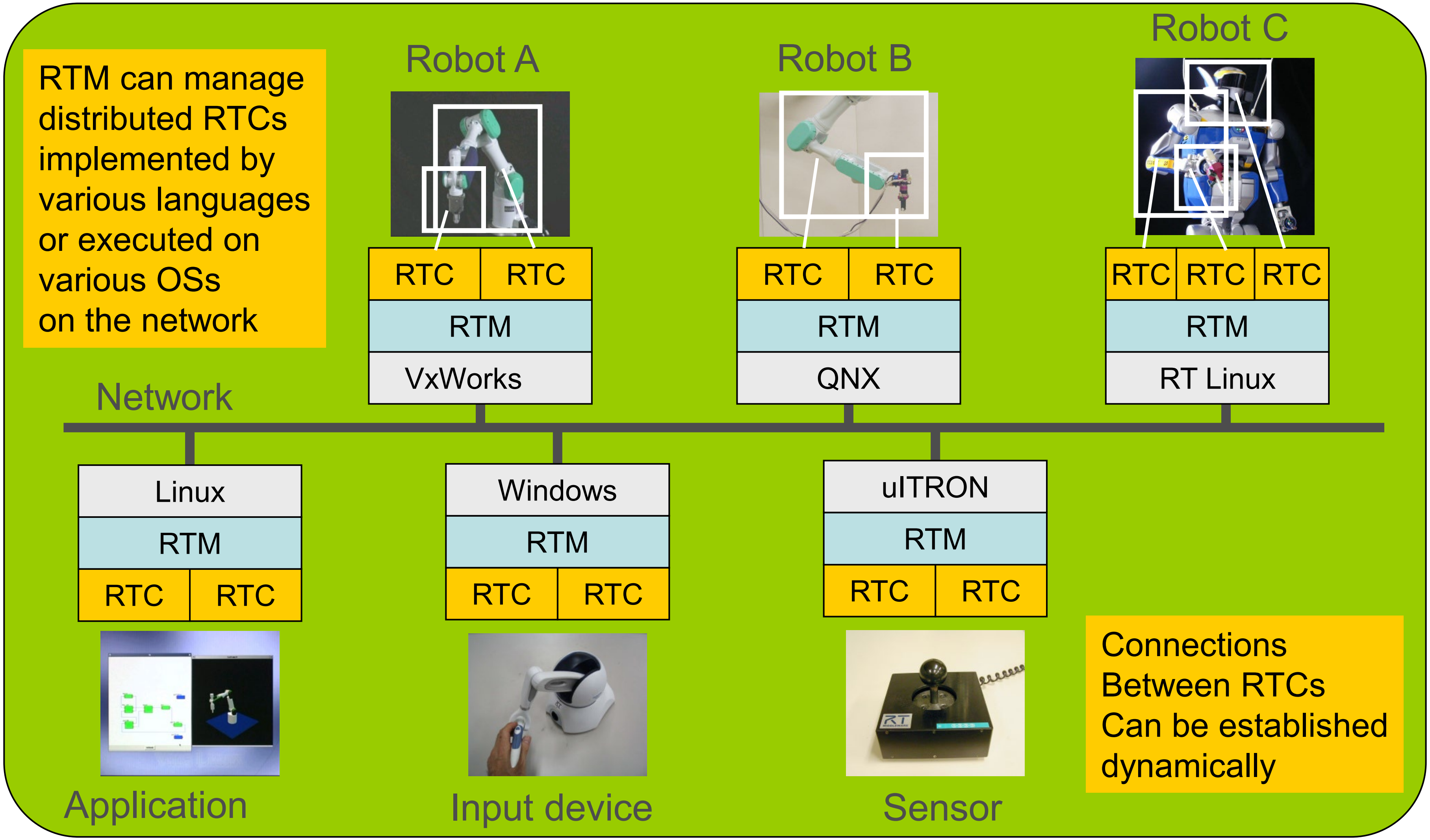
Reduce development from scratch

Reusable RTCs

Self development using existing libraries

Develop from scratch

RTM based Distributed Systems



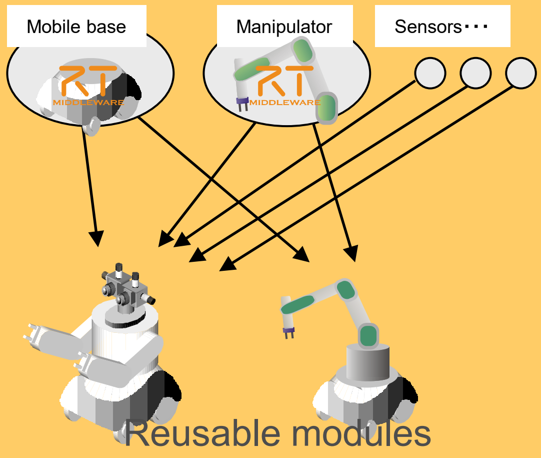
RTM can manage distributed RTCs implemented by various languages or executed on various OSs on the network

Connections Between RTCs Can be established dynamically

The aim of RT-Middleware

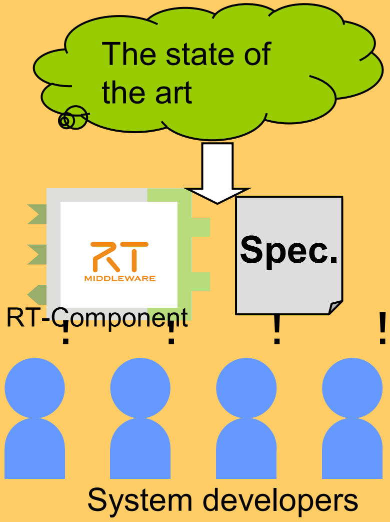
Problem Solving by Modularization

Cost



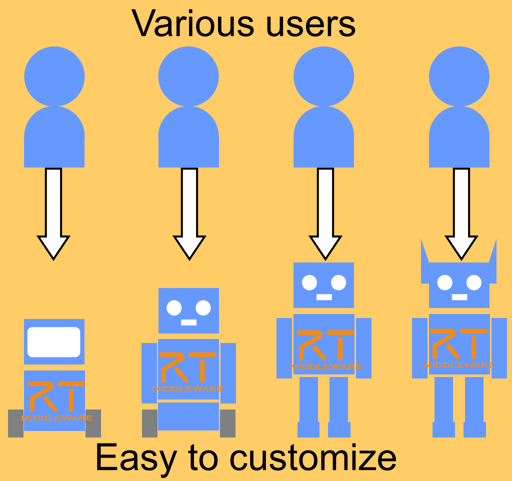
Realize
low-cost robots

Technical Issue



Utilize
the state of the art

Needs



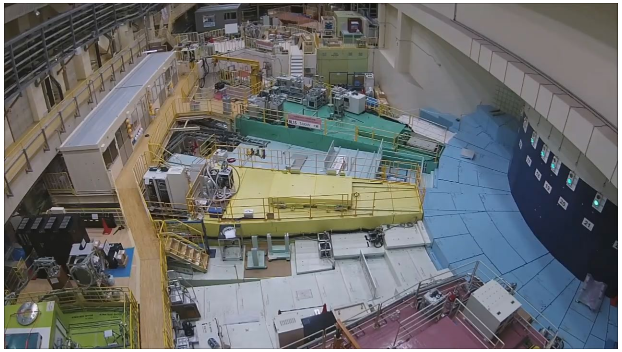
Satisfy
various needs

Robot System Integration Innovation

Practical/commercialization examples

HRP series: KAWADA and AIST

S-ONE : SCHAFT



DAQ-Middleware: KEK/J-PARC

KEK: High Energy Accelerator Research Organization

J-PARC: Japan Proton Accelerator Research Complex

HIRO, NEXTAGE open: Kawada Robotics

THK

The Mark of Linear Motion

THK: SIGNAS system

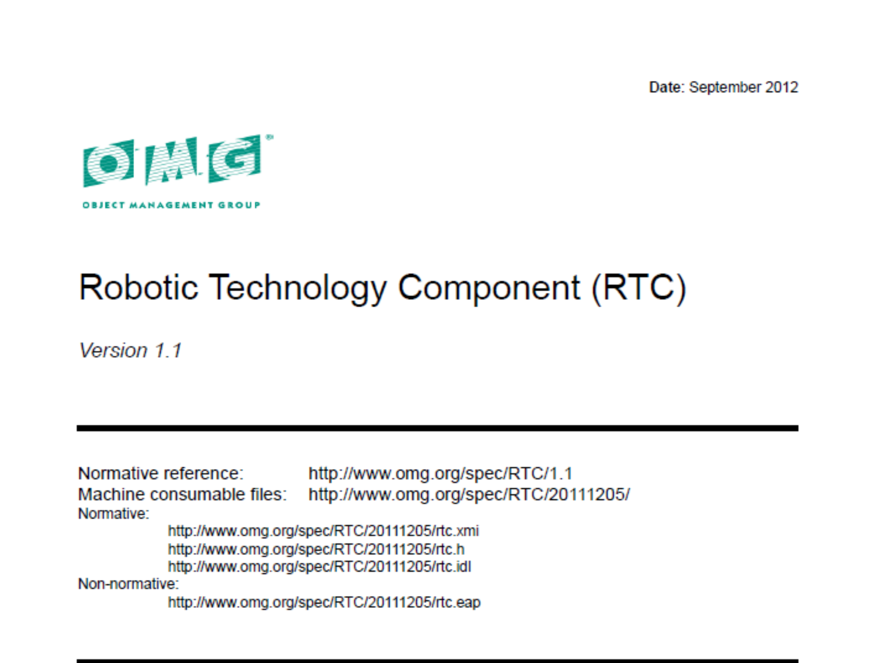
TOYOTA L&F : Air-T

VSTONE's education robots

OROCHI（RT corp.）

Robot operation simulator: NEDO

RTM as a International Standard



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

History

- September, 2005
  Request for Proposal issued (starting standardization)
- September, 2006
  Specification approved by OMG
- April, 2008
  OMG RTC ver.1.0 released
- September, 2012
  Updated to ver. 1.1
- September, 2015
  FSM4RTC (FSM based RTC standard) adopted

OMG Standard

Standardized by OMG process
→ It can not be modified by just one company
→ Various compatible implementation
→ It promoted competition and interoperability

Ten or more RT-Middleware implementation exist

Implementation	Vendor	Features	Compatibility
OpenRTM-aist	AIST	Reference implementation by AIST	---
HRTM	Honda R&D	ASIMO is now moving to HRTM	◎
OpenRTM.NET	SEC	.NET(C#,VB,C++/CLI, F#, etc..)	◎
RTM on Android	SEC	RTM implementation for Android	◎
RTC-Lite	AIST	Tiny implementation on PIC and dsPIC	○
Mini/MicorRTC	SEC	RTM/RTC for CAN and Zigbee	○
RTMSafety	SEC/AIST	Functional safety standard capable RTM implementation	○
RTC CANOpen	SIT, CiA	RTM for CANOpen standard	○
PALRO	Fujisoft	Yet another C++ PSM implementation	×
OPRoS	ETRI	Implementation of Korean national project	×
GostaiRTC	GOSTAI, THALES	C++ PSM implementation on a robot language	×

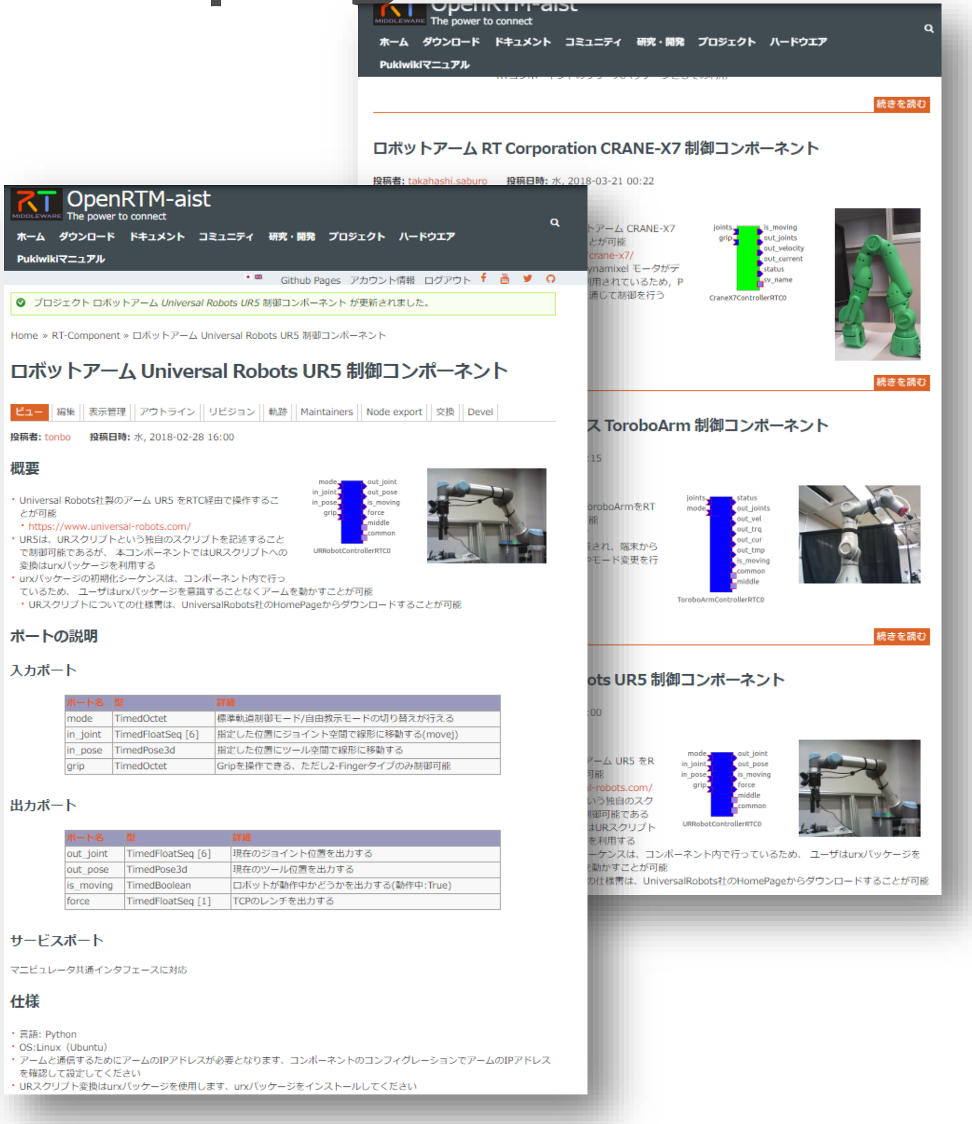
Users can chose and continue to use
on of the RTM implementations

RT-Middleware community

Project web pages

- Users can upload their own RTCs on the openrtm.org
- Users can search and download other users RTCs

Project type	Number
RT-Components	429
RT-Middleware	42
Tools	27
Documents	13
Hardware related RTCs	28



OpenRTM-aist
The power to connect

ホーム ダウンロード ドキュメント コミュニティ 研究・開発 プロジェクト ハードウエア
Pukiwikiマニュアル

続きを読む

ロボットアーム RT Corporation CRANE-X7 制御コンポーネント

投稿者: takahashi.saburo **投稿日時:** 水, 2018-03-21 00:22

トアーム CRANE-X7
とが可能
/crane-x7/
ynamixel モータがデ
用されているため，P
通じて制御を行う

続きを読む

ス ToroboArm 制御コンポーネント

:15

oroboArmをRT
能
され、端末から
モード変更を行

続きを読む

ots UR5 制御コンポーネント

:00

アーム UR5 をR
可能
l-robots.com/
いう独自のスク
制御可能である
はURスクリプト
を利用する
ーケンスは、コンポーネント内で行っているため、ユーザはurxパッケージを
を動かすことが可能
の仕様書は、UniversalRobots社のHomePageからダウンロードすることが可能

OpenRTM-aist
The power to connect

ホーム ダウンロード ドキュメント コミュニティ 研究・開発 プロジェクト ハードウエア
Pukiwikiマニュアル

Github Pages アカウント情報 ログアウト

プロジェクト ロボットアーム *Universal Robots UR5* 制御コンポーネント が更新されました。

Home » RT-Component » ロボットアーム Universal Robots UR5 制御コンポーネント

ロボットアーム Universal Robots UR5 制御コンポーネント

ビュー | 編集 | 表示管理 | アウトライン | リビジョン | 軌跡 | Maintainers | Node export | 交換 | Devel

投稿者: tonbo **投稿日時:** 水, 2018-02-28 16:00

概要

- Universal Robots社製のアーム UR5 をRTC経由で操作することが可能
  - https://www.universal-robots.com/
- UR5は、URスクリプトという独自のスクリプトを記述することで制御可能であるが、 本コンポーネントではURスクリプトへの変換はurxパッケージを利用する
- urxパッケージの初期化シーケンスは、コンポーネント内で行っているため、 ユーザはurxパッケージを意識することなくアームを動かすことが可能
  - URスクリプトについての仕様書は、UniversalRobots社のHomePageからダウンロードすることが可能

ポートの説明

入力ポート

ポート名	型	詳細
mode	TimedOctet	標準軌道制御モード/自由教示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空間で線形に移動する(movej)
in_pose	TimedPose3d	指定した位置にツール空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

出力ポート

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のレンチを出力する

サービスポート

マニピュレータ共通インタフェースに対応

仕様

- 言語: Python
- OS:Linux（Ubuntu）
- アームと通信するためにアームのIPアドレスが必要となります、コンポーネントのコンフィグレーションでアームのIPアドレスを確認して設定してください
- URスクリプト変換はurxパッケージを使用します、urxパッケージをインストールしてください

RT-Middleware Summer Camp

- 1 week camp every summer
- This year: August 19-23
- Venue: AIST Tsukuba center (Tsukuba city, Ibaraki pref.)
- Lectures, practical work and presentation by five teams.
- Staying in the AIST’s accommodation and coding endlessly every night.

RT-Middleware Contest

- Held as an organized session in SICE SI conference
  - Various prizes
  - Entry deadline: Sep. 23rd
  - Software registration：Oct.
  - Paper submission due: Oct. 26th
  - Online examination：from end of Nov.
  - Presentation and award ceremony: Dec.
- Record of year 2023
  - Number of applications ：11
  - SICE RT-Middleware award x1
  - Product supporting award x2
  - Company supporting award x9
  - Personal supporting award x10
- See more details: openrtm.org
  - Menu: community -> events

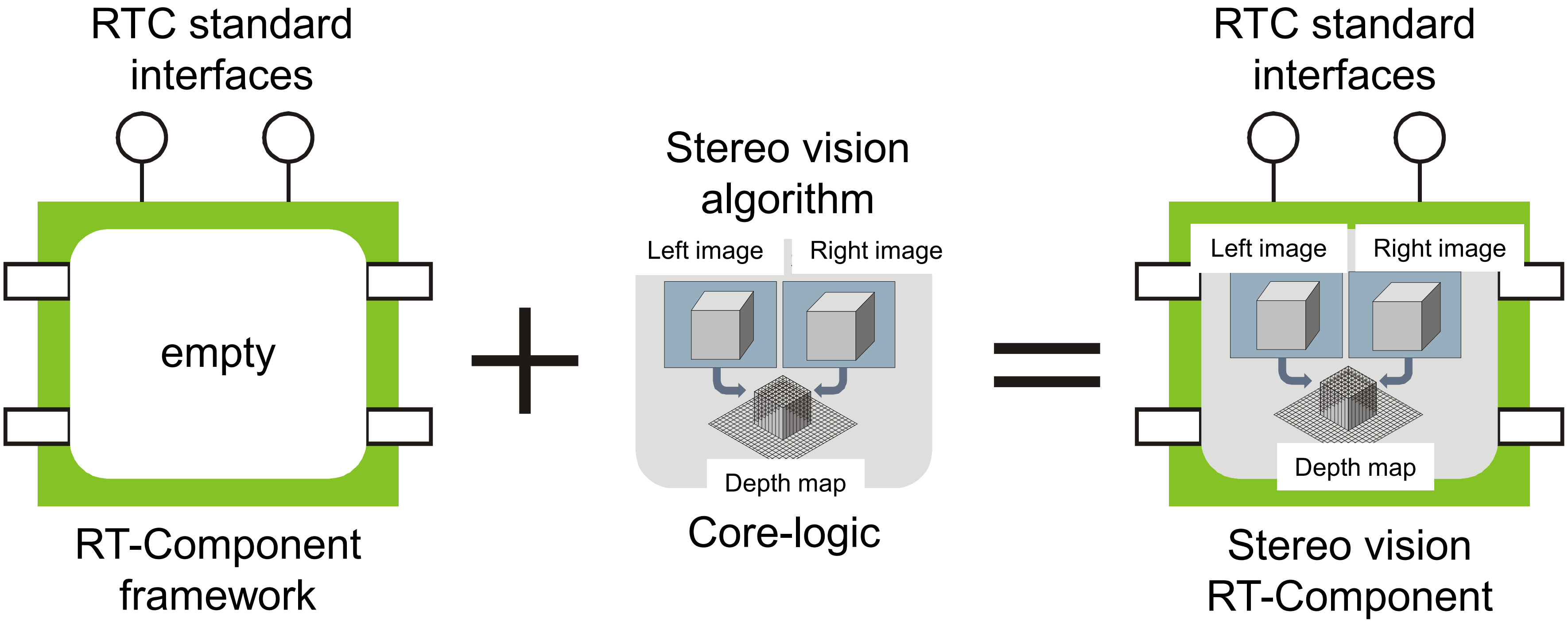
最優秀賞（副賞10万円）
計測自動制御学会RTミドルウエア賞
総合評価として一番優秀な開発成果に対して最優秀賞として「計測自動制御学会RTミドルウエア賞」を表彰します。

Recommendation

- Let’s stop reinventing the wheel!!
  - Code that has been executed thousands of times by different people works better than your code from scratch!!
  - Let's write the code you really need to write and borrow the other non-essential part for you.
  - A program released by someone is a program that has worked once!!
  - Other persons code is hard to read, but you shouldn't throw it away for that reason!!
- Commit to open source projects!!
  - Don't hesitate to ask questions on ML and forums!!
  - No matter how rudimentary a question is, it is valuable information for others.
  - Let’s complain to the project!! (good feedback grow the project)
  - Debug and send patches if you can!!

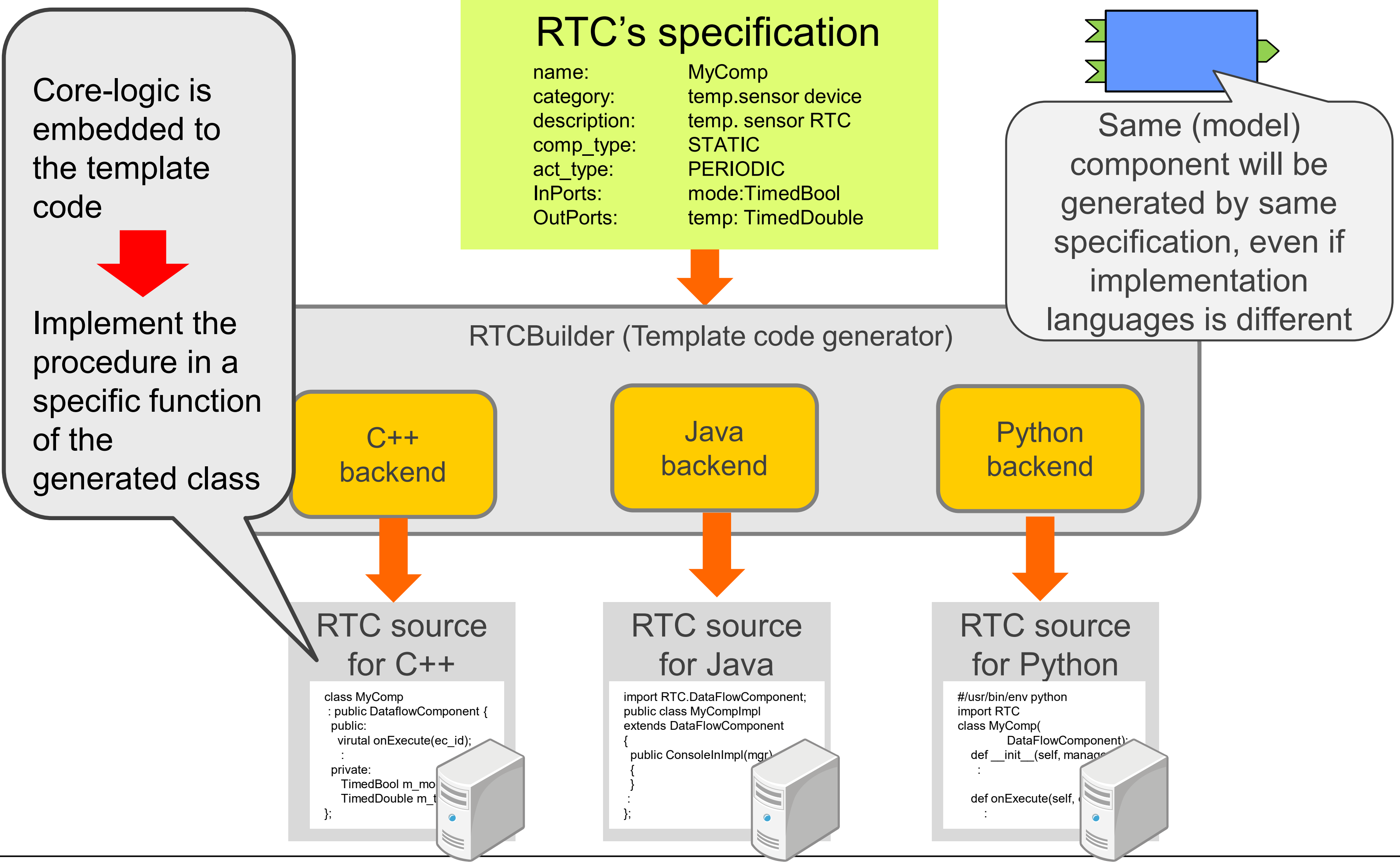
RTC development overview

Framework and core-logic



RTC framework ＋ Core logic ＝ RT-Component

Code generation by model



RTC's specification

name:	MyComp
category:	temp.sensor device
description:	temp. sensor RTC
comp_type:	STATIC
act_type:	PERIODIC
InPorts:	mode:TimedBool
OutPorts:	temp: TimedDouble

Core-logic is embedded to the template code

Implement the procedure in a specific function of the generated class

Same (model) component will be generated by same specification, even if implementation languages is different

RTC source for C++

```
class MyComp
 : public DataflowComponent {
  public:
    virutal onExecute(ec_id);
      :
  private:
     TimedBool m_mo
     TimedDouble m_t
};
```

RTC source for Java

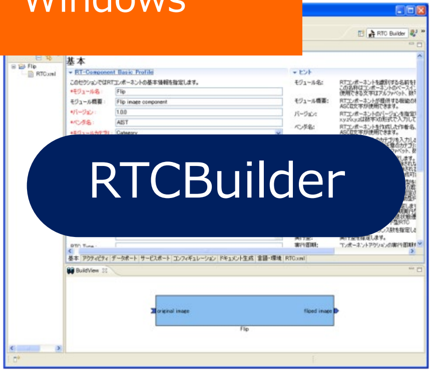
```
import RTC.DataFlowComponent;
public class MyCompImpl
extends DataFlowComponent
{
  public ConsoleInImpl(mgr)
  {
  }
 :
};
```

RTC source for Python

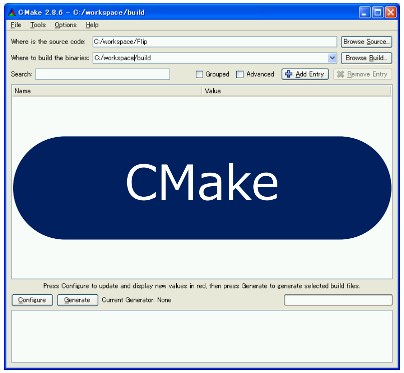
```
#/usr/bin/env python
import RTC
class MyComp(
              DataFlowComponent):
   def __init__(self, manag
     :

   def onExecute(self,
      :
```

RTC development flow

Windows


RTCBuilder

→


CMake

CMake 2.8.6 - C:/workspace/build

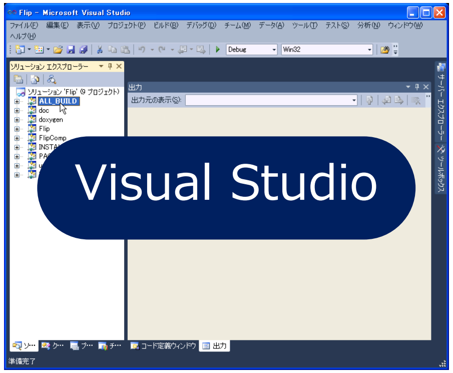
Where is the source code: C:/workspace/Flip

Where to build the binaries: C:/workspace/build

Press Configure to update and display new values in red, then press Generate to generate selected build files.

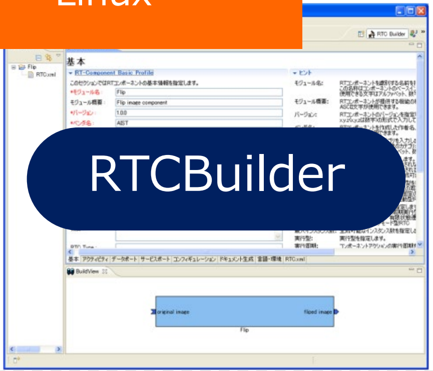
Configure | Generate | Current Generator: None

→

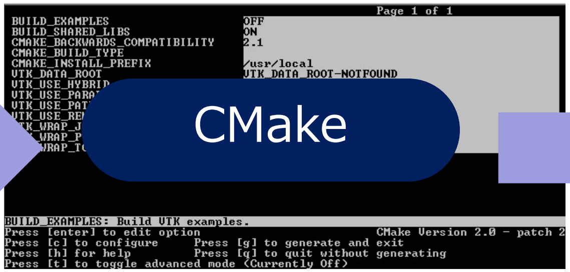

Visual Studio

Flip - Microsoft Visual Studio

Linux


RTCBuilder

→


CMake

```
                                            Page 1 of 1
BUILD_EXAMPLES                  OFF
BUILD_SHARED_LIBS               ON
CMAKE_BACKWARDS_COMPATIBILITY   2.1
CMAKE_BUILD_TYPE
CMAKE_INSTALL_PREFIX            /usr/local
VTK_DATA_ROOT                   VTK_DATA_ROOT-NOTFOUND
VTK_USE_HYBRID
VTK_USE_PARA
VTK_USE_PAT
VTK_USE_REN
VTK_WRAP_J
VTK_WRAP_P
VTK_WRAP_TC

BUILD_EXAMPLES: Build VTK examples.
Press [enter] to edit option                  CMake Version 2.0 - patch 2
Press [c] to configure          Press [g] to generate and exit
Press [h] for help              Press [q] to quit without generating
Press [t] to toggle advanced mode (Currently Off)
```

→

make
+
gcc (g++)

Input spec of RTC

Generating VC project or Makefile

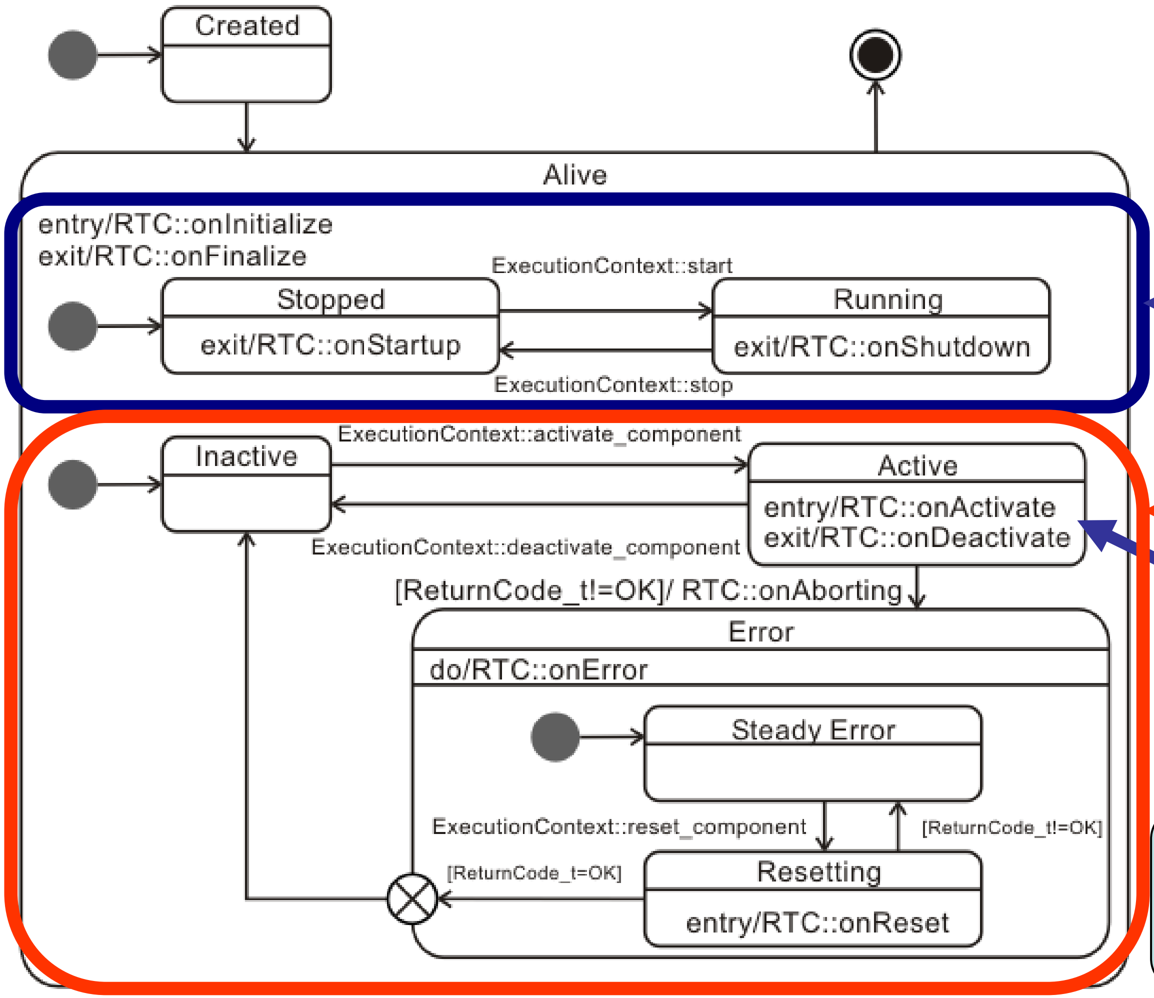
Implement logic and compile

Almost all steps are the same, except compiler

CMake

- Open source software for compiler-independent build automation
- Can generate build files for different development environments on different operating systems
- Generate Makefile on Linux
- Generate VC (Visual C++) project file on Windows
- Most of the recent open source software is built with CMake.

State machine and lifecycle of RTC



You don’t need to aware here

Be aware state here and its callback functions.

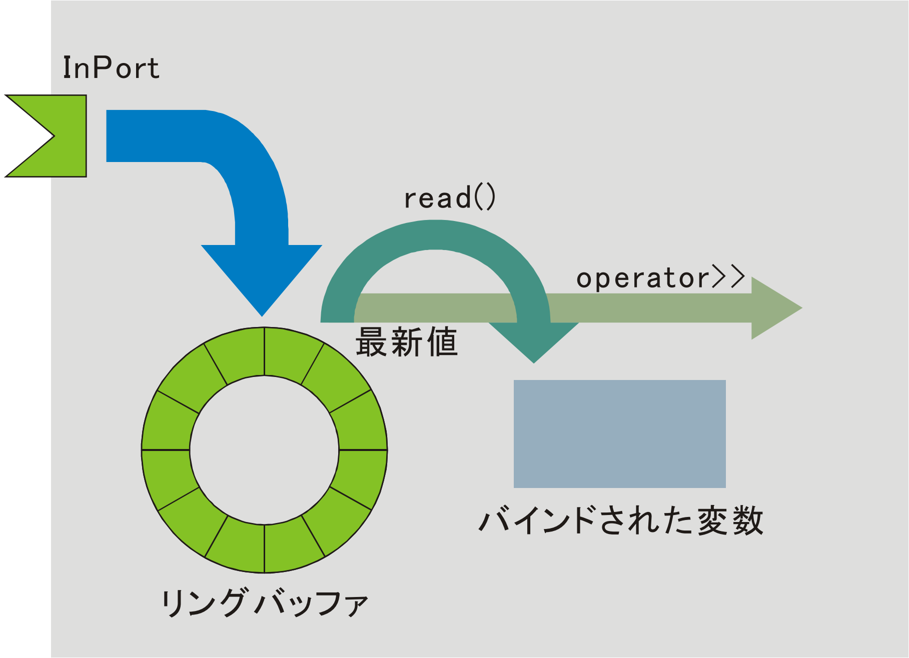
ActiveDo/RTC::onExecute() callback function is here

Activity (Callback functions)

Callback functions	Meanings
onInitialize	Initialization
onActivated	Called once when RTC is activated
onExecute	Called periodically when RTC is in the active state
onDeactivated	Called once when RTC is deactivated
onAborting	Called once when entering ERROR state
onReset	Called once when resetting
onError	Called periodically when RTC is in the error state
onFinalize	Called once when finalizing RTC
onStateUpdate	Called after onExecute everytime
onRateChanged	Called when ExecutionContext’s rate is changed
onStartup	Called once when ExecutionContext starting
onShutdown	Called once when ExecutionContext stopping

InPort

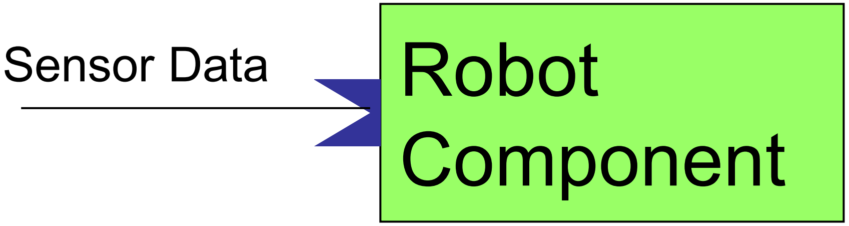
- InPort
  - Input port for data flow type communication
- Methods of InPort class
  - isNew(): check if new data arriving
  - read(): retrieve data from InPort buffer to the variable bound to the InPort
  - >> : same as above



Basically paired with OutPort

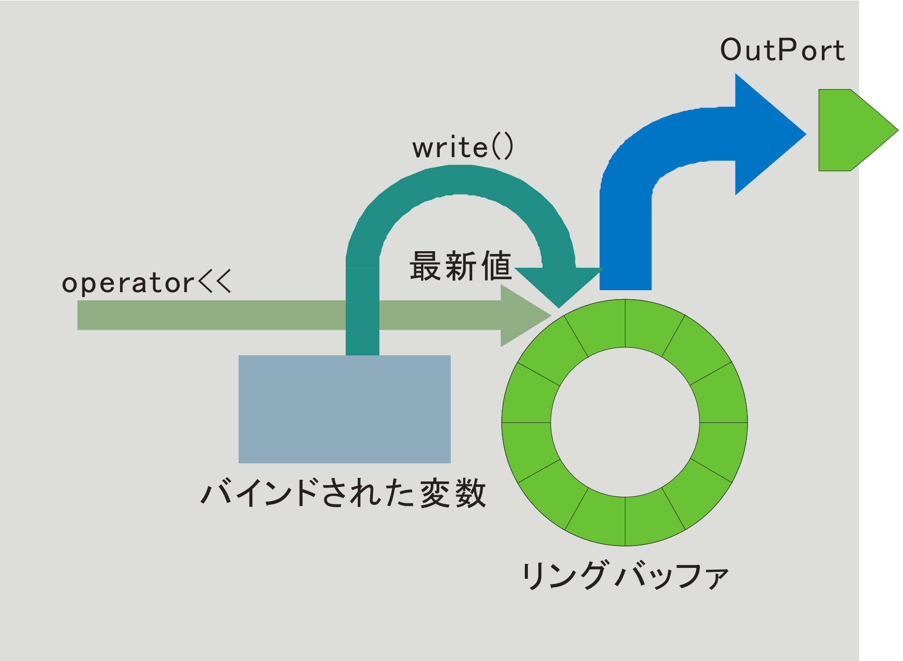
Data ports (InPort/OutPort) must have the same type

Example



OutPort

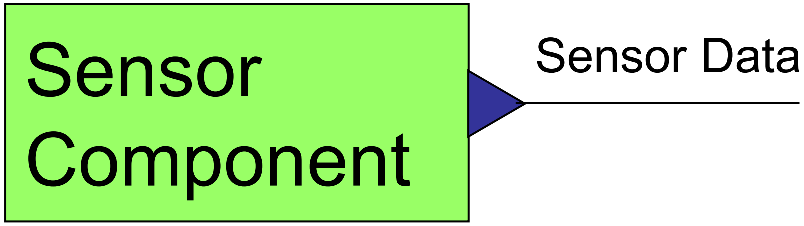
- OutPort
  - Output port for data flow type communication
- Methods of OutPort class
  - write(): push data from OutPort’s variable into OutPort’s buffer to be published to the remote InPort
  - << : same as above



Basically paired with InPort

Data ports (InPort/OutPort) must have the same type

例



Conclusion

- Basic concept and overview of RT-Middleware
- Activities of RT-Middleware community
- RTC development overview