

<http://code.google.com/p/rtm-ros-robotics/>

# エージェントシステム - RTM・ROS統合 -

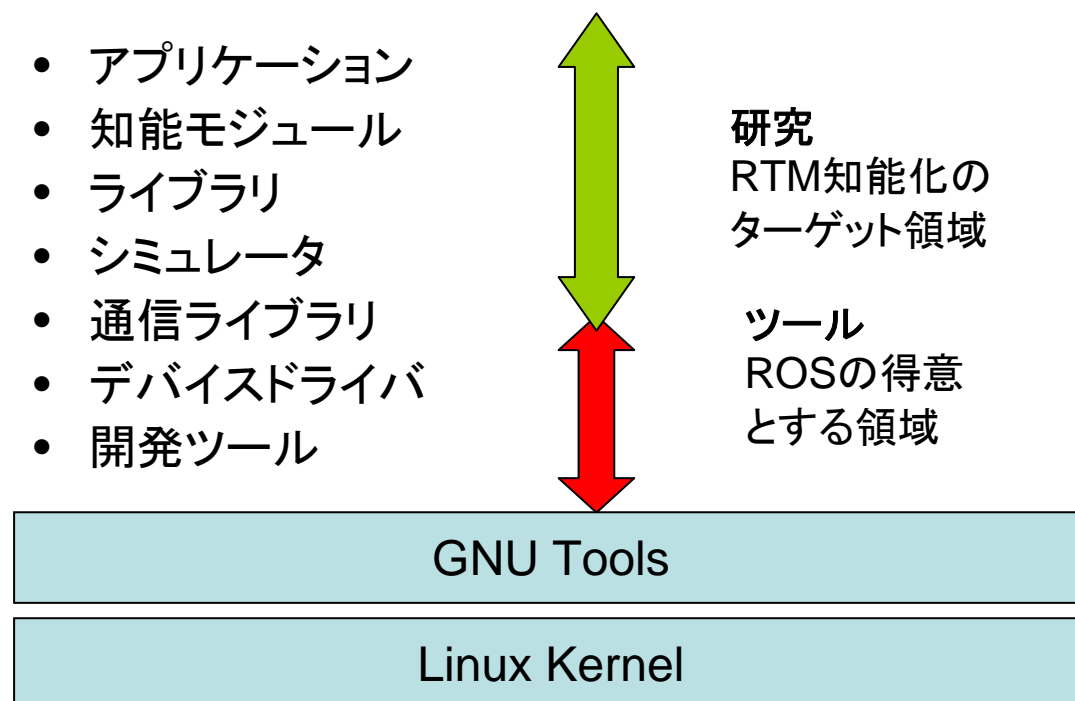
2011/06/08

岡田 慧

# RTMROS統合方式

## • 稼動実績のあるものから使う

- 何を書けばいいかわかってるのがよいプログラマ。なにを書き直せば(そして使い回せば)いいかわかってるのが、すごいプログラマ(エリックレイモンド『伽藍とバザール』)



WillowGarage社の幹部が好んで利用するスライド赤が研究に必要な雑多な作業(左のツールレベル). 緑が研究そのもの. 現状はほとんどの時間をツール作業に費やしている, ROSにより, 後者の割合を多くしたい. という説明がなされる. Steve Cousins speaking at Robo Development Tuesdayより

<http://code.google.com/p/rtm-ros-robotics/>

# RTM用依存関係解決コンパイルツール

```
$ rosmake mrobot_ros_bridge
```

```
/openrtm
/openrtm/manifest.xml
/openrtm/lib
/openrtm/lib/libRTC.so
/iis_idl
/iis_idl/manifest.xml
/iis_idl/idl
/iis_idl/idl/IIS.idl
/iis_idl/idl_gen
/iis_idl/idl_gen/cpp
/iis_idl/idl_gen/cpp/iis_idl/idl/IIS.hh.hh
/iis_idl/idl_gen/cpp/iis_idl/idl/IISK.cc
/iis_idl/idl_gen/cpp/iis_idl/idl/IISDynSK.cc
/iis_idl/idl_gen/cpp/iis_idl/idl/IISKskel.cpp
/iis_idl/idl_gen/cpp/iis_idl/idl/IISKskel.h
/iis_idl/idl_gen/cpp/iis_idl/idl/IISStub.cpp
/iis_idl/idl_gen/cpp/iis_idl/idl/IISStub.h
/iis_idl/idl_gen/lib
/iis_idl/idl_gen/lib/libIISKskel.so
/iis_idl/idl_gen/lib/libIISStub.so
/mrobot_ros_bridge
/mrobot_ros_bridge/manifest.xml
/mrobot_ros_bridge/bin
/mrobot_ros_bridge/bin/MobileRobotROSBridgeComp
/mrobot_ros_bridge/src
/mrobot_ros_bridge/src/MobileRobotROSBridge.cpp
/mrobot_ros_bridge/src/MobileRobotROSBridge.h
/mrobot_ros_bridge/src/MobileRobotROSBridgeComp.cpp
```

—————→ `<depend package="openrtm" />`

—————→ `<depend package="iis_idl" />`

(4) rtmパッケージに依存するプログラムは openrtm関連のインクルードファイルパス, ならびにライブラリをリンクする, という情報がかけられている.

(3) idlファイルがあるので, idlコンパイラでコンパイルしidl\_gen/cppフォルダ以下に生成ファイルを置く. さらに, これらのファイルをコンパイルし, idl\_gen/lib/ディレクトリ以下にライブラリを置く.

(2) mrobot\_ros\_bridge/manifest.xmlを参照し, パッケージの依存関係を調べ, openrtm, iis\_idlパッケージを見つける.

(1) mrobot\_ros\_bridgeディレクトリで, rosmakeとコマンドを打ち込む

(5) ソースコードをコンパイルし, iis\_idl以下のライブラリとリンクし実行ファイルを生成する.

<http://code.google.com/p/rtm-ros-robotics/>

# RTMROS統合コンポーネント起動ツール

- mrobot\_simulator.launch起動設定ファイル

```
<launch>
  <!-- BEGIN:openrtm setting -->
  <arg name="nameserver" default="localhost" />
  <env name="RTCTREE_NAMESERVERS" value="$(arg nameserver)" />
  <arg name="openrtm_args" value='-o "corba.nameservers:$(arg nameserver):2809" -o "naming.formats:%n.rtc" -o
"logger.file_name:/tmp/rtc%p.log"' />
  <arg name="openrtm_ext_args" value='$(arg openrtm_args) -o "exec_cxt.periodic.type: SynchExtTriggerEC"' />
  <!-- END:openrtm setting -->
  <node pkg="openhyp3" name="grxui" type="grxui.sh" args="$(find mrobot_ros_bridge)/launch/SimulationProject.xml"/>
  <node name = "IISMobileRobotControllerComp" pkg = "mrobot_ros_bridge" type = "IISMobileRobotControllerComp"
    args = "$(arg openrtm_ext_args)" output = "screen"/>
  <node name = "openhyp_controller_bridge" pkg = "openhyp3" type = "openhyp-controller-bridge"
    args = "--server-name IISMobileRobotControllerComp
      --out-port angle:JOINT_VALUE
      --out-port velocity:JOINT_VELOCITY
      --in-port torque:JOINT_TORQUE
      $(arg openrtm_ext_args)"
    output = "screen"/>
  <node name = "MobileRobotROSBridgeComp"
    pkg = "mrobot_ros_bridge" type = "MobileRobotROSBridgeComp" args = "$(arg openrtm_args)" output = "screen"/>
  <node pkg="pr2_teleop" type="teleop_pr2_keyboard" name="teleop_pr2_keyboard" output="screen" launch-prefix="xterm -e"/>

  <!-- BEGIN:openrtm connection -->
  <node name="rtmlaunch" pkg="openrtm" type="rtmlaunch.py" args="$(find mrobot_ros_bridge)/launch/mrobot_simulator.launch"
output = "screen"/>
  <rtconnect from="IISMobileRobotController0.rtc:torque" to="IISMobileRobotControllerComp(Robot)0.rtc:torque" />
  <rtconnect from="IISMobileRobotControllerComp(Robot)0.rtc:velocity" to="IISMobileRobotController0.rtc:velocity" />
  <rtconnect from="IISMobileRobotControllerComp(Robot)0.rtc:angle" to="IISMobileRobotController0.rtc:angle" />
  <rtconnect from="IISMobileRobotController0.rtc:out" to="MobileRobotROSBridge0.rtc:in" />
  <rtconnect from="MobileRobotROSBridge0.rtc:out" to="IISMobileRobotController0.rtc:in" />
  <rtactivate component="IISMobileRobotController0.rtc" />
  <rtactivate component="MobileRobotROSBridge0.rtc" />
  <!-- END:openrtm connection -->
</launch>
```

http://code.google.com/p/rtm-ros-robotics/

# 起動ツール埋め込みドキュメンテーション

```
$ roscd mrobot_ros_bridge; make  
$ rosdot rosdot mrobot_ros_bridge
```

The screenshot displays a Mozilla Firefox browser window titled "mrobot\_ros\_bridge ROS Launch Files — mrobot\_ros\_bridge v1.0 documentation - Mozilla Firefox". The address bar shows the file path: `file:///home/k-okada/ros/diamondback/rtm-ros-robotics/rtmros_common/`. The page content includes:

- Description:** mrobot\_ros\_bridge
- License:** BSD
- mrobot\_simulator.launch**
- `roslaunch mrobot_ros_bridge mrobot_simulator.launch`
- This script starts mobile robot simulator and keyboard telopep interface**

On the right side, there is a **Table Of Contents** section with links to `mrobot_ros_bridge.launch` and `mrobot_simulator.launch`, each with a **Contents** link. Below this is a **Quick search** bar with a **Go** button and a prompt to "Enter search terms or a module, class or function name."

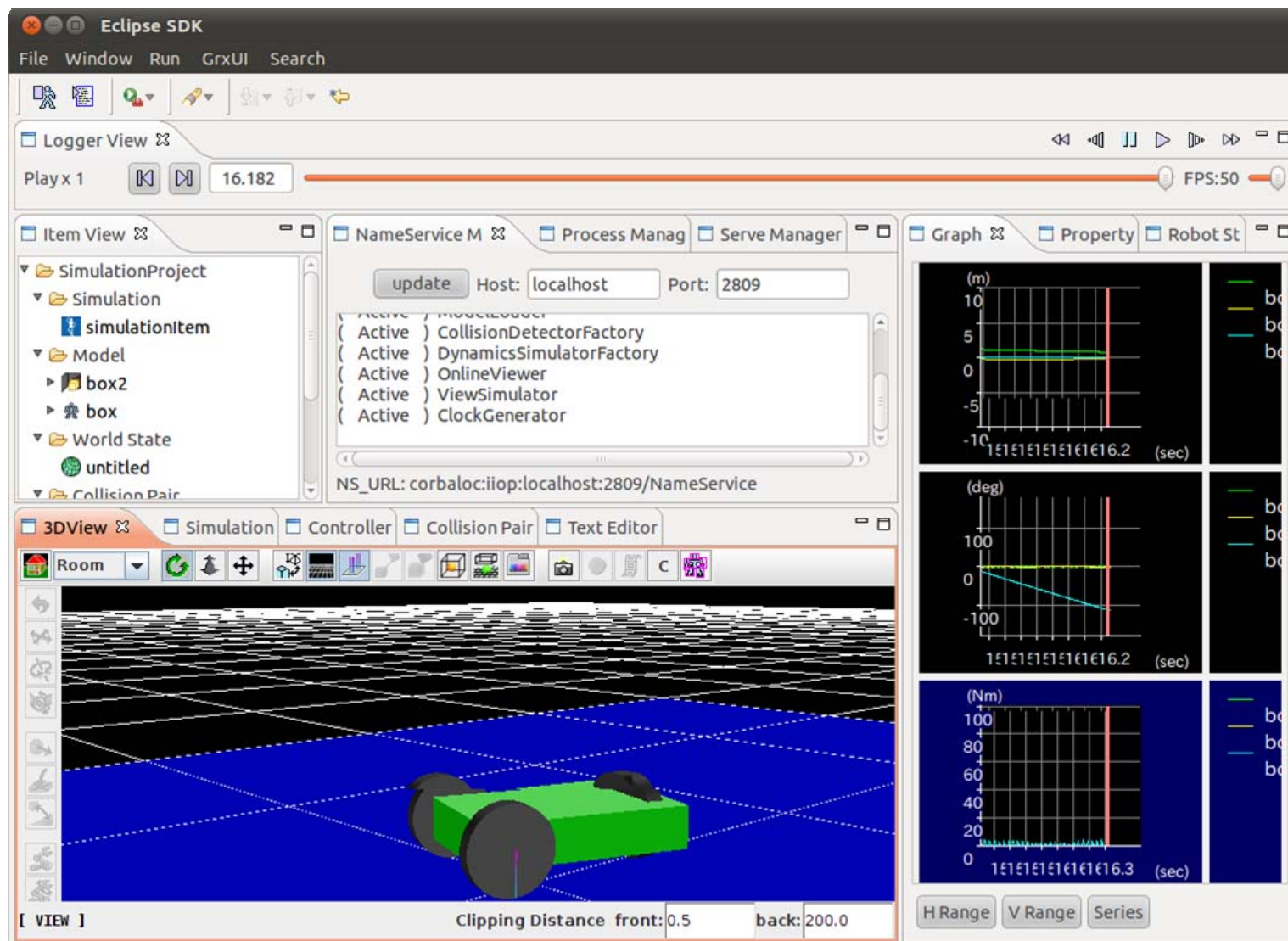
An inset image shows the Eclipse IDE running a simulation. The **3D View** displays a green mobile robot on a blue surface. The **Console** shows the output of the simulation, including the `roslaunch` command and the `roscd` command. The **Graph** window shows three plots: `linear`, `angular`, and `linear`, all showing a constant value of 0.0 over time.

完了

<http://code.google.com/p/rtm-ros-robotics/>

# RTMROS統合移動ロボットシミュレーション

```
$ roslaunch mrobot_ros_bridge mrobot_simulator.launch
```

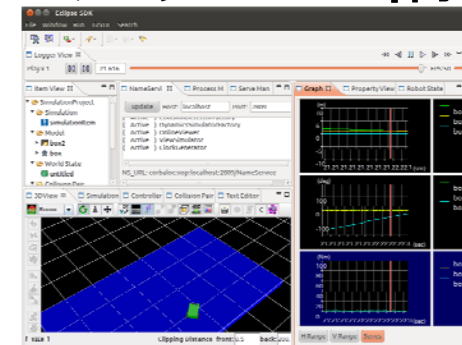
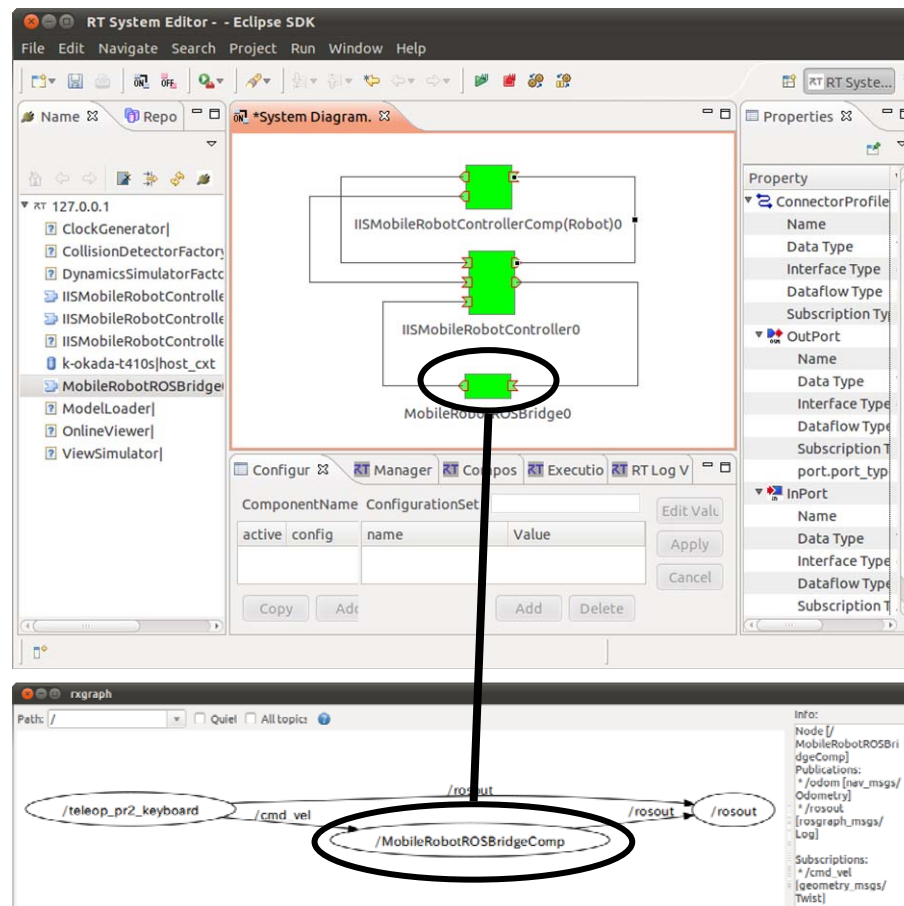




<http://code.google.com/p/rtm-ros-robotics/>

# RTMROS統合移動システム構成図

- 抽象化の壁の構築(計算機プログラムの構造と解釈. 2章)



TimedDoubleSeq

IISMobileRobotController

IIS:TimedVelocity  
(移動知能SWG)

MobleRobotROSBridge

/cmd\_vel

```
teleop_pr2_keyboard
Reading from keyboard
-----
Use 'WASD' to translate
Use 'QE' to yaw
Press 'Shift' to run
```

<http://code.google.com/p/rtm-ros-robotics/>

# 宿題

- 1: 移動台車のシミュレーションを実行せよ
- 2: 1を拡張し距離センサのシミュレーションを行ってみよ.  
([http://http://www.openrtp.jp/openhrp3/jp/range\\_sensor.html](http://http://www.openrtp.jp/openhrp3/jp/range_sensor.html))
- 3: 距離情報をROSのメッセージ(sensor\_msgs/LaserScan)としてpublishしてみよ.
- 4: これらのプログラムを実行するlaunchファイルを作成し, ドキュメントを埋め込め
- 宿題は1人でもチームでもかまわない. チームの場合はMLで議論を公開しながら進めること. MLでの紹介で提出とみなす.