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

エージェントシステム

- RTM･ROS統合 -

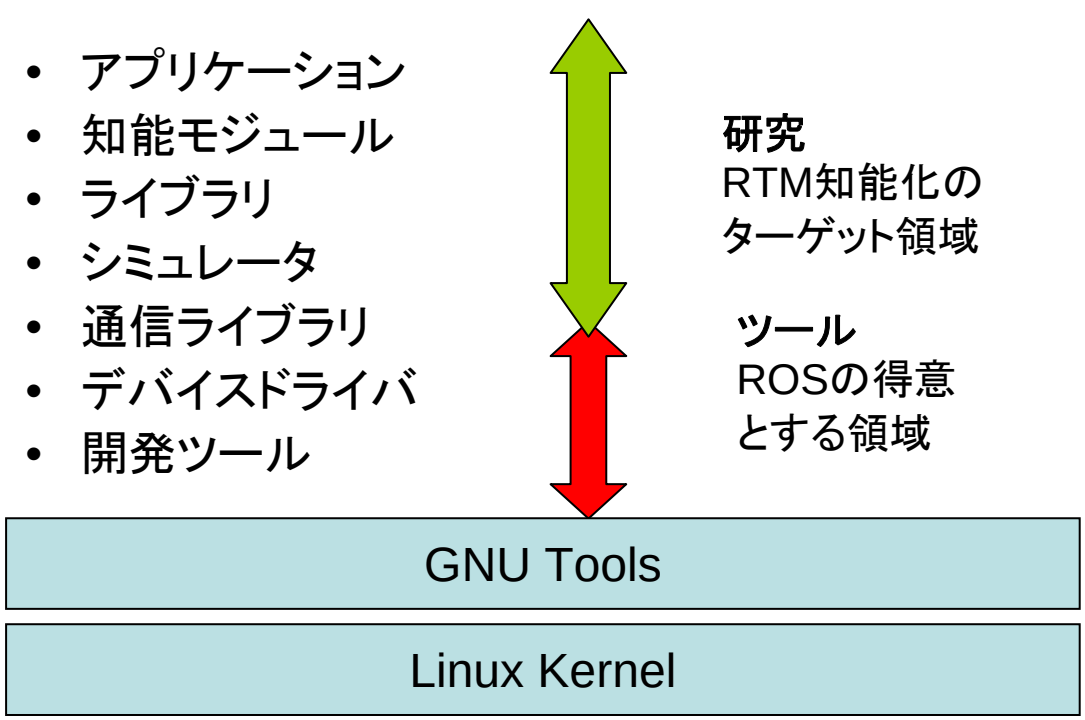
2011/06/08

岡田　慧

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

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/IISSK.cc
/iis_idl/idl_gen/cpp/iis_idl/idl/IISDynSK.cc
/iis_idl/idl_gen/cpp/iis_idl/idl/IISSkel.cpp
/iis_idl/idl_gen/cpp/iis_idl/idl/IISSkel.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/libIISSkel.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，という情報がかかれている．

（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RTM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

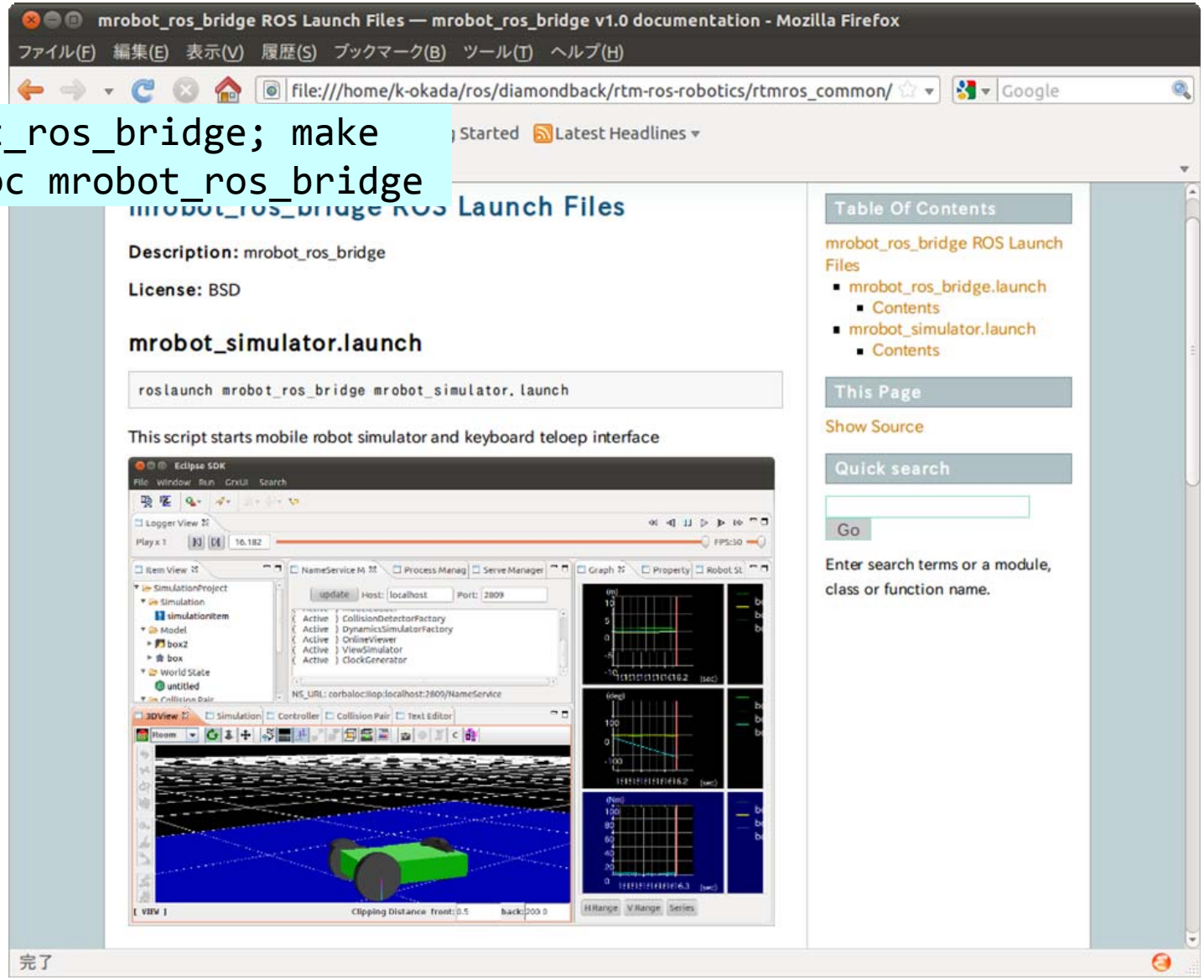
```
<launch>
  <!- BEGIN:openrtm setting →
  <arg name="nameserver" default="localhost" />
  <env name="RTCTREE_NAMESERVERS" value="$(arg nameserver)" />
  <arg name="openrtm_args" value='-o "corba.nameservers:$(arg nameserver):2809" -o "naming.formats:%n.rtc" -o
"logger.file_name:/tmp/rtc%p.log"' />
  <arg name="openrtm_ext_args" value='$(arg openrtm_args) -o "exec_cxt.periodic.type: SynchExtTriggerEC"' />
  <!-- END:openrtm setting -->
  <node pkg="openhrp3" name="grxui" type="grxui.sh" args="$(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 = "openhrp_controller_bridge" pkg  = "openhrp3" type = "openhrp-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

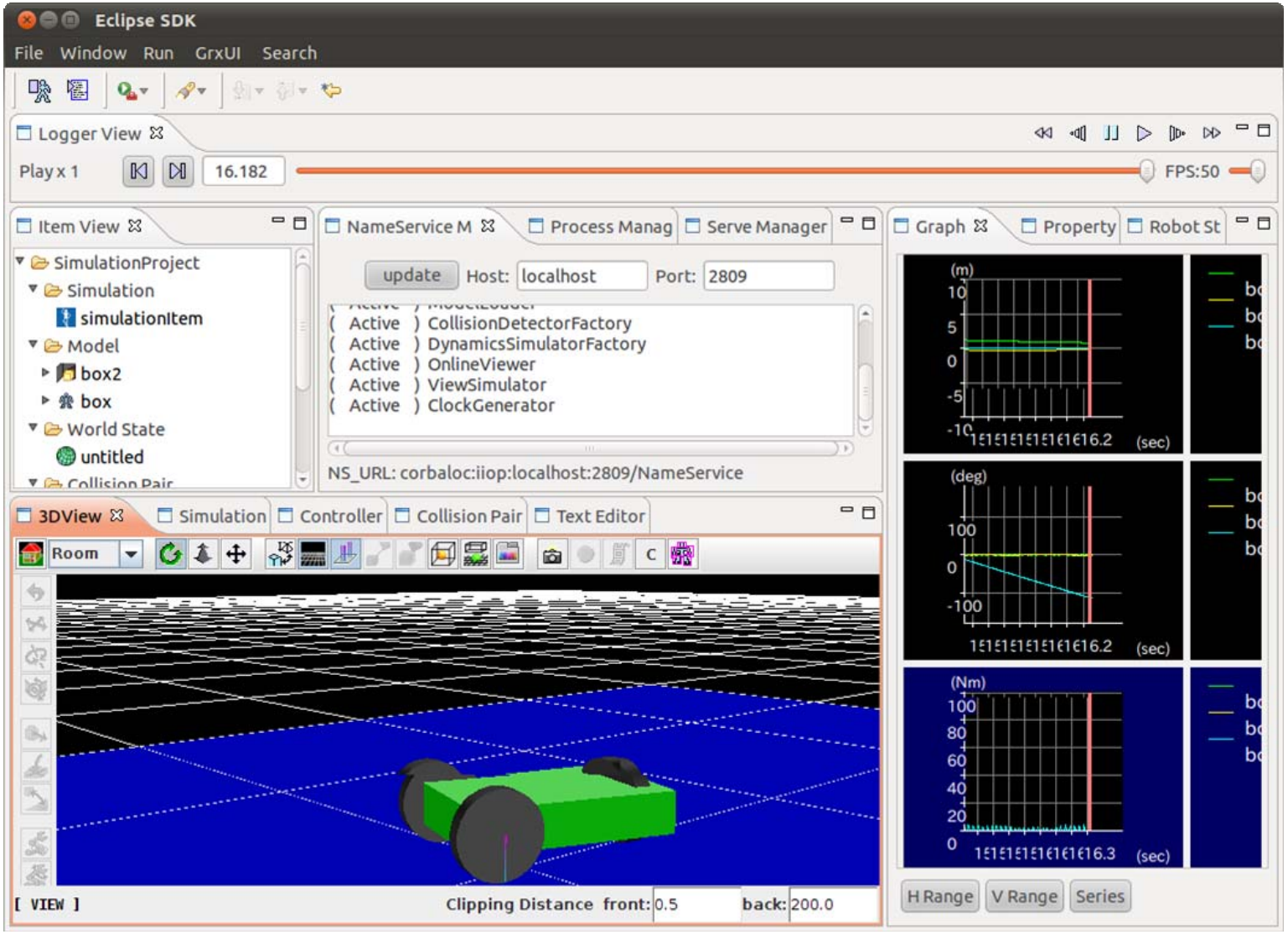
```
$ roscd mrobot_ros_bridge; make
$ rosdoc rosdoc mrobot_ros_bridge
```



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

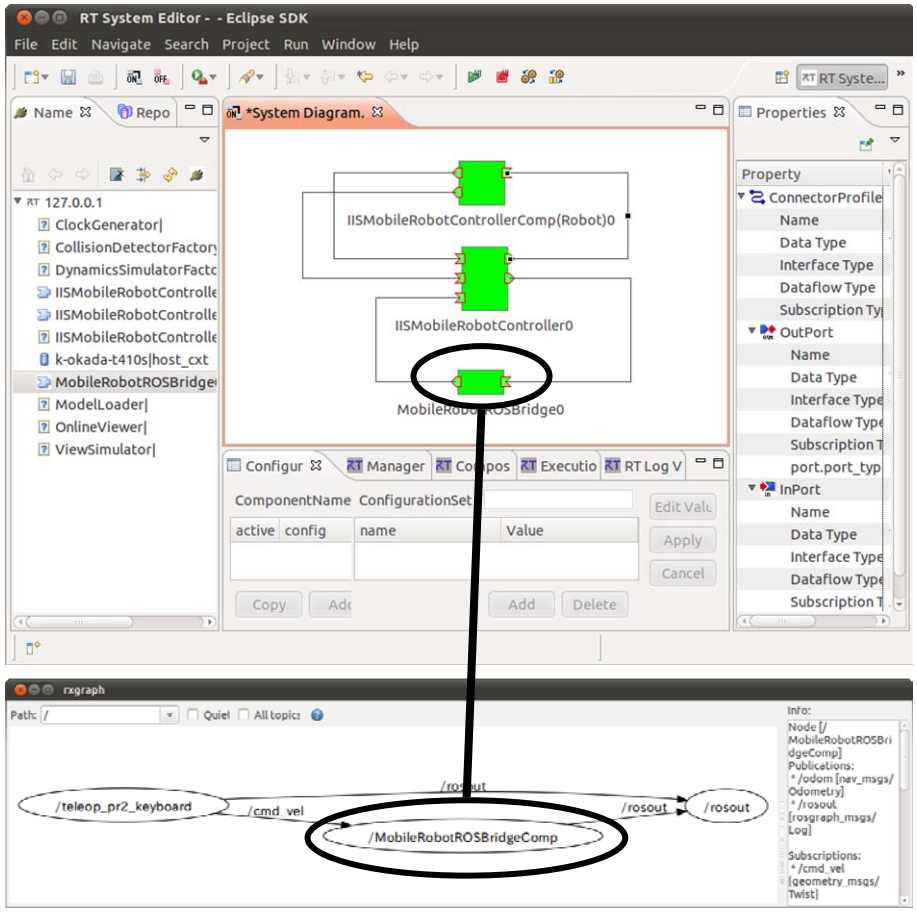
```
$ roslaunch mrobot_ros_bridge mrobot_simulator.launch
```

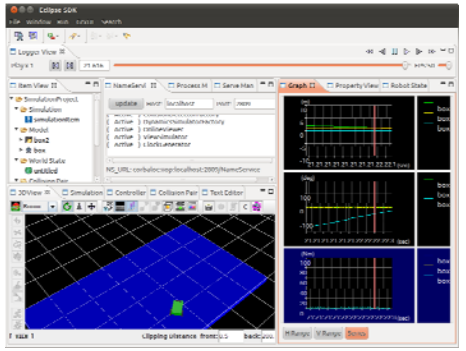


http://code.google.com/p/rtm-ros-robotics/

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

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）
- 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．