RTミドルウエアサマーキャンプ2016

RTM-ROS相互運用とJSKでの取り組み

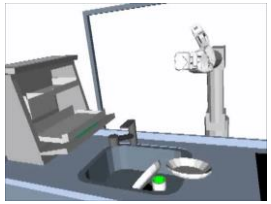
東京大学　情報システム工学研究室
岡田　慧

11:00－11:30

k-okada@jsk.t.u-tokyo.ac.jp

情報システム工学研究室（JSK）

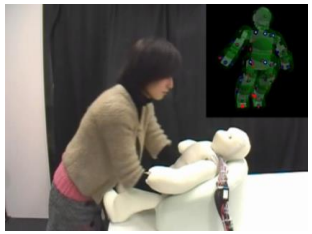
教授　稲葉雅幸　　講師：垣内洋平，矢口裕明
准教授　岡田慧　　助教：野沢峻一，菅井人仁



MUJIN
出杏光

SCHAFT
中西，浦田，西脇

H6 & H7 人型ロボット
1999 加賀美，西脇

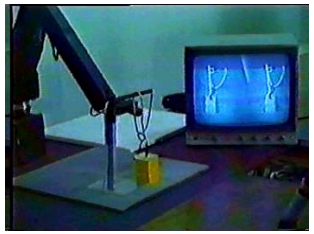


センサ埋め込み肉質外装
2006 吉海

腱駆動ヒューマノイド
2000 水内

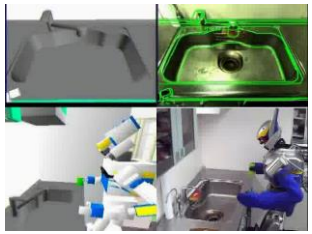
文科省先端融合領域イノベーション創出拠点の形成IRT
2006-
花井，山崎

PR2ナビゲーション
2011 岡田

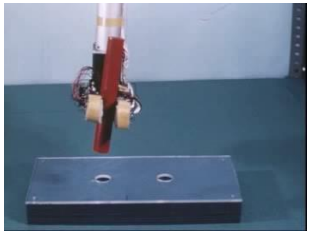


COSMOS:
1981 井上，小笠原，松井，溝口，稲葉

リモートブレインロボット
1993, 稲葉



生活支援ヒューマノイド
2002 岡田



人口の手の研究
1969 井上博允

VC

OSS

Home Assistance

HRP2 Task Integration

Musculoskeletal Humanoid

Sensor Suit　Sensor Flesh

Remote-Brained Robotics: about 60 robots

HARP: Humanoid Autonomous Robot Project H1-H7

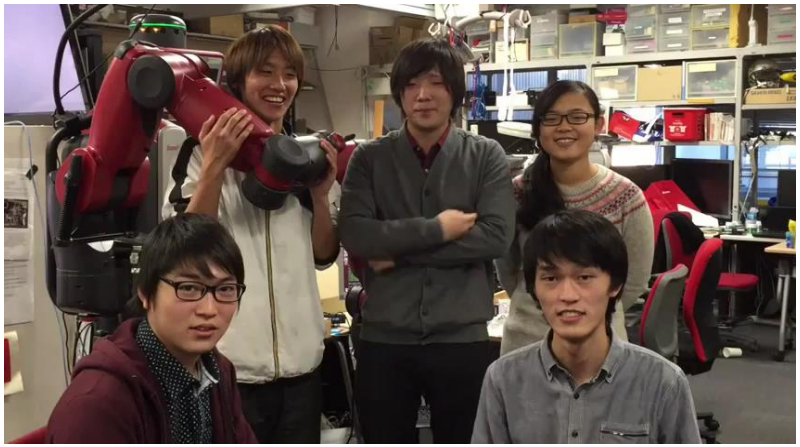
Vision-Based Robotics: Manipulation, Interaction, Navigation

COSMOS: Lisp-based Robot System Integration Environment

1980　1990　2000　2010

Amazon Picking Challenge

- Amazon Robotics が主催
- 2014/10 APC2015ルール発表
- 2015/1　棚/物品の申込み
- 2015/3　旅費サポート申込み
- 2015/5　APC2015 @ シアトル
- 2016/2　申込み
- 2016/6　APC2016 @ ライプチッヒ

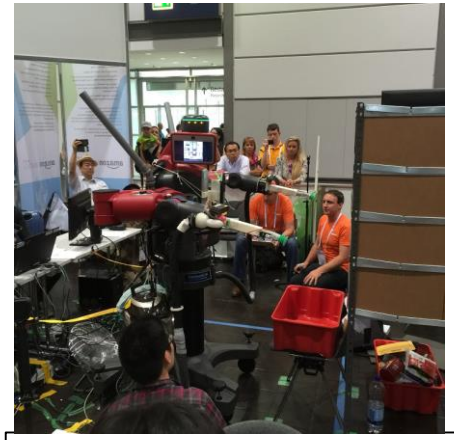


2015 物品申し込みビデオ

2015 旅費ビデオ

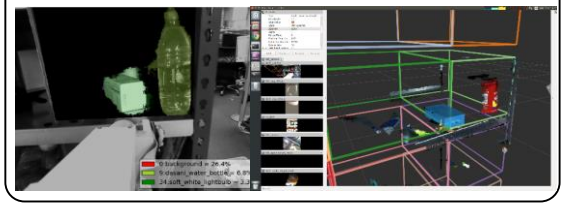
2015 大会の様子

APC 2016 Pick&Stow System



Grab Object for Pick

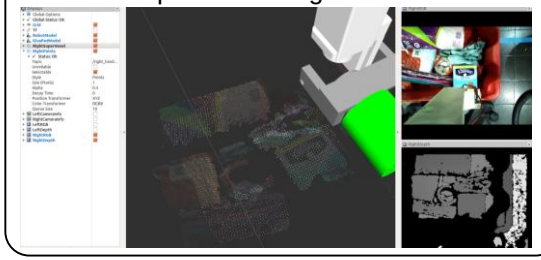
Segmentation In Bin
FCN(fully convolution network) +
Bounding Box Extraction



Manually annotated 200 Training Data

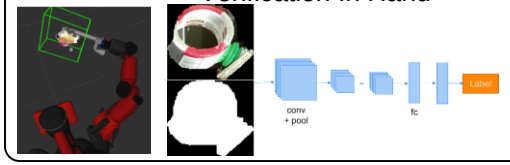
Grab Object for Stow

Segmentation in Tote
Super Voxel Segmentation

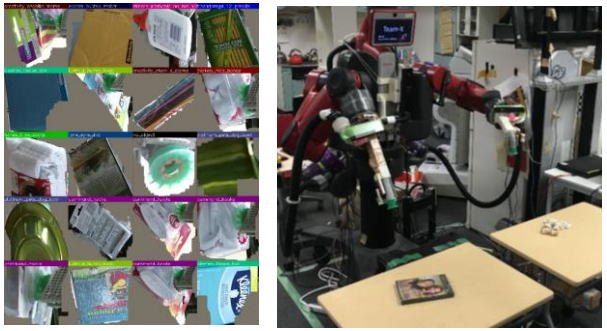


Grab Object for Stow

Verification In Hand



Autonomously obtained ~1000 Training Data



Put Object in Bin/Tote

Delft	産x1	蘭、Delft Robotics・デルフト工科大学	105
PFN	産x2	日、PFNベンチャー企業（東大）	105
NimbRo Picking	URx1	独、ボン大学	97
MIT	産x1	米，MIT大学	67
Team K	Baxter	日、東大	49
ACRV	Baxter	豪、オーストラリアベンチャー企業	42
HARP	URx1	米、CMU大学	33
C^2M	産x2	日、中京大学・中央大学・三菱電機	21
Dataspeed -Grizzly	Baxter	米、企業・オークランド大学	21
AA-team	Aero	日、東大・THK	16
IITK-TCS	WAMx1	伊、IIT	16
Applied Robotics	URx1	米、Applied Robotics	10
Duke	Baxter	米、デューク大学	0
KTH	Baxter	スウェーデン、KTH大学	0
microRecycler	Baxter	豪、ベンチャー企業、UNSW大、キャンベラ大	0
Rutgers ARM	産双x1	米、ラトガー大学	N/A

Delft	産x1	蘭、Delft Robotics・デルフト工科大学	214
NimbRo Picking	URx1	独、ボン大学	186
MIT	産x1	米，MIT大学	164
PFN	産x2	日、PFNベンチャー企業（東大）	161
IITK-TCS	WAMx1	伊、IIT	109
HARP	URx1	米、CMU大学	88
Duke	Baxter	米、デューク大学	66
Team-K	Baxter	日、東大	52
KTH	Baxter	スウェーデン、KTH大学	40
microRecycler	Baxter	豪、オーストラリアベンチャー企業	20
C^2M	産x2	日、中京大学・中央大学・三菱電機	16
Applied			

HRP2を使ったAPCタスクの実現

3D Object Segmentation for Shelf Bin Picking by Humanoid with Deep Learning and Occupancy Voxel Grid Map

Kentaro Wada Masaki Murooka Kei Okada Masayuki Inaba
(The University of Tokyo)

IEEE-RAS International Conference on Humanoid Robots 2016

hrpsysシステムの発展経緯

HRP2 (2002)
- Sequencer
- Balancer
- PatternGenerator

hrpsys

JSK
- Collision detector
- Soft Joint Limiter
- Temperature estimator
- Autobalancer
- Impedance controller

hrpsys@JSK

Open-source

Closed-source

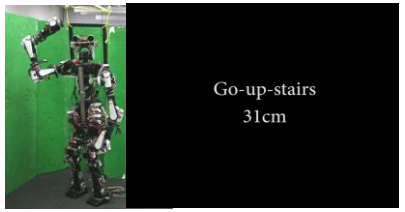
HiroNX (2009)

hrpsys/OpenRTM

hrpsys (2011)

HironNX / NextageOpen

HRP-4R



STARO

Prototype-1

hrpsys@AIST
hrpsys@Osaka
hrpsys@Tohoku
hrpsys@NAIST

- Stack of Tasks
- Pattern Generataor

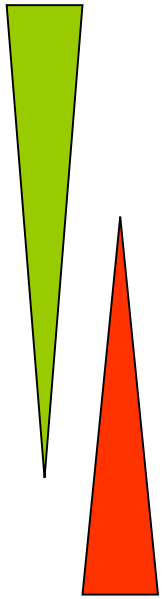
hrpsys@CNRS

hrpsys/OpenRTM

HRP-4R (2010)

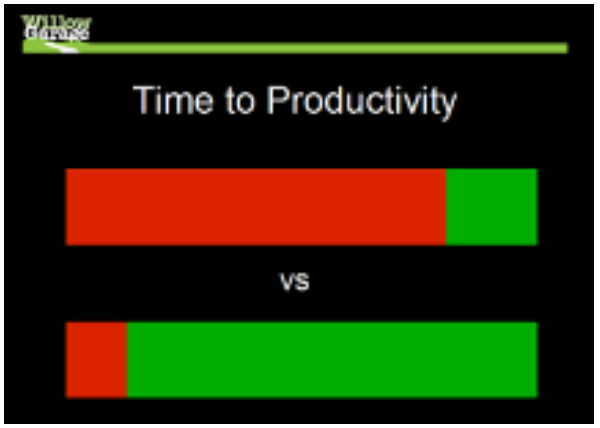
NEDO知能化プロジェクト(2011)における RTM-ROS相互運用方式

- アプリケーション
- 知能モジュール
- ライブラリ
- シミュレータ
- 通信ライブラリ
- デバイスドライバ
- 開発ツール



研究・事業化
RTM知能化の
ターゲット領域

ツール
ROSの得意とする領域



WillowGarage社のスライドより．赤が研究に必要なツール作成等の雑作業．緑が研究そのもの．現状は上．多くの時間をツール作業に費やす．ROSは研究サポートを行うツール
（ Steve Cousins speaking at Robo Development
http://www.willowgarage.com/blog/2008/11/17/steve-cousins-speaking-robo-development-tuesday
より）

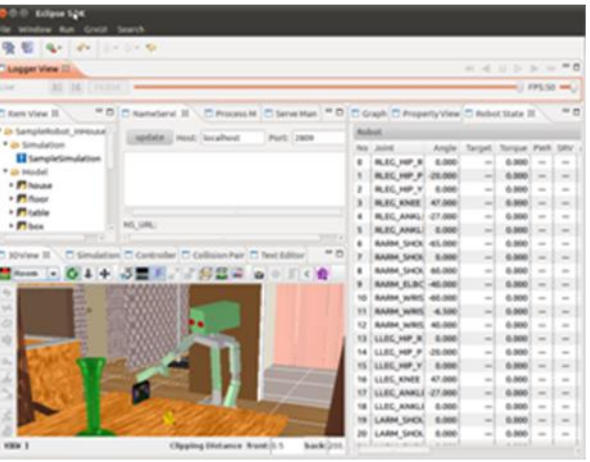
→ オープンソースツール上にRTM-ROS統合環境を構築

- ねらい1：世界中の研究成果をOpenRTMロボットに取り込み統合できるように
- ねらい2：RTMモジュールの効率的な開発・保守環境により更なる発展を可能に

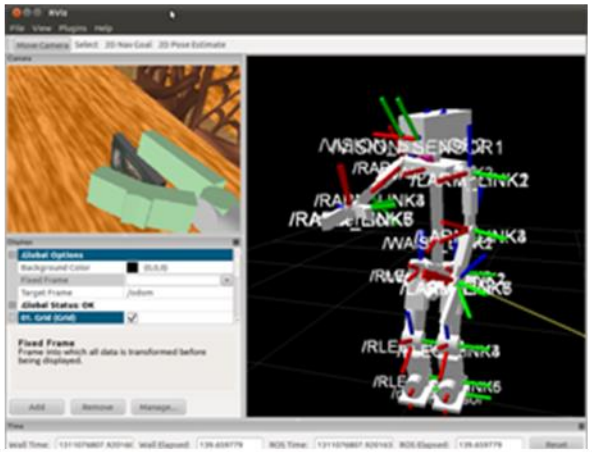
相互運用環境構築

http://wiki.ros.org/rtmros_common

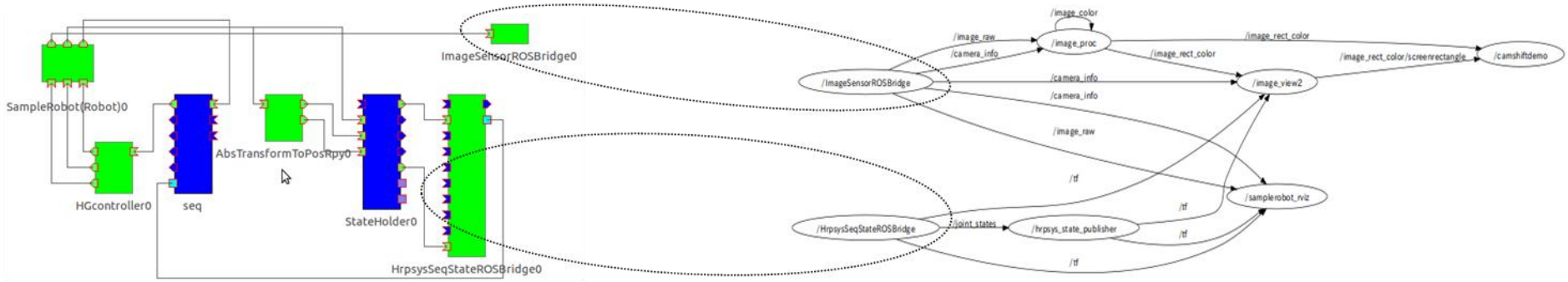
```
# Install RTMROS environment
$ sudo sh -c 'echo "deb http://packages.ros.org/ros/ubuntu `lsb_release -cs` main" > /etc/apt/sources.list.d/ros-latest.list‘
$ wget http://packages.ros.org/ros.key -O - | sudo apt-key add –
$ sudo apt-get update
$ sudo apt-get ros-indigo-hrpsys-ros-bridge
```



RTM
- Dynamics simulation
- Robot Model
- Controller
- Sequence
- Sensor Holder



ROS
- Sensor Viewer
- Image Processing



ロボット知能化コンポーネント統合開発支援環境

- RTM統合支援開発ツール
  - パッケージソースコード開発管理ツール (rosmake/catkin)
    - 依存関係のあるパッケージのコンパイル．インクルードファイルへのパス，ライブラリへのリンク，IDLファイルのC++/Java/Pythonへのコンパイル，リンクを自動で行う
  - プログラムコンポーネント実行管理ツール (roslaunch/rtmlaunch)
    - 複数のプログラムコンポーネントの実行
    - オンラインでのパラメータ設定，引数設定
    - コンポーネント間のでのデータポート，サービスポートの接続
  - ドキュメンテーションツール・テスティングサーバ
- ROSコンポーネントの取り込み
  - ナビゲーション，プランニング，三次元認識

ROSツールを用いた コンポーネントの起動と接続

```
$ rtmlaunch openrtm_ros_bridge myservice_example.launch
$ roslaunch openrtm_ros_bridge myservice_rosbridge.launch
```

1つのコマンドでに必要な全てのROS, RTMコンポーネントの起動，設定と相互接続を行う.

RTMコンポーネントのサービスをROSのサービスから利用する

```
<launch>
  <arg name="HAVE_DISPLAY" default="true" />

  <!-- BEGIN:openrtm setting -->
  <env name="RTCTREE_NAMESERVERS" value="localhost:15005" />
  <arg name="openrtm_args" value='-o "corba.nameservers:localhost:15005" -o "naming.formats:%n.rtc" -o "logger.file_name:/tmp/rtc%p.log"
  <!-- END:openrtm setting -->

  <!-- RTC Service Provider sample -->
  <node name="provider" pkg="openrtm_tools" type="MyServiceProviderComp"
              args='$(arg openrtm_args)' />

  <!-- RTC Service COnsumet sample -->
  <node name="consumer" pkg="openrtm_tools" type="MyServiceConsumerComp"
        args='$(arg openrtm_args)' />

  <!-- BEGIN:openrtm connection -->
  <node name="rtmlaunch_example" pkg="openrtm_tools" type="rtmlaunch.py"
        args="$(find openrtm_ros_bridge)/samples/myservice_example.launch"/>
  <rtactivate component="MyServiceProvider0.rtc" />
  <rtactivate component="MyServiceConsumer0.rtc" />
  <rtconnect from="MyServiceProvider0.rtc:MyService"
             to="MyServiceConsumer0.rtc:MyService"/>
  <!-- END:openrtm connection -->

</launch>
```

RTCコンポーネント自動相互変換ノード

RTCをROSから利用する

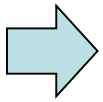
- RTCコンポーネントから自動的にROSノードを生成するプログラム
- rtmbuild.cmake に以下のcmake macrosを記述

```
rtmbuild_init()
rtmbuild_genidl
rtmbuild_genbridge()
```

- idl ファイルをコンパイルし，ROS のサービスメッセージファイルと，OpenRTM/ROS相互変換ノードのソースプログラムを自動生成する

```
module SimpleService {
 typedef sequence<string> EchoList;
 typedef sequence<float> ValueList;
 interface MyService
 {
   string echo(in string msg);
   EchoList get_echo_history();
   void set_value(in float value);
   float get_value();
   ValueList get_value_history();
 };
 };
```

SimpleService IDL File

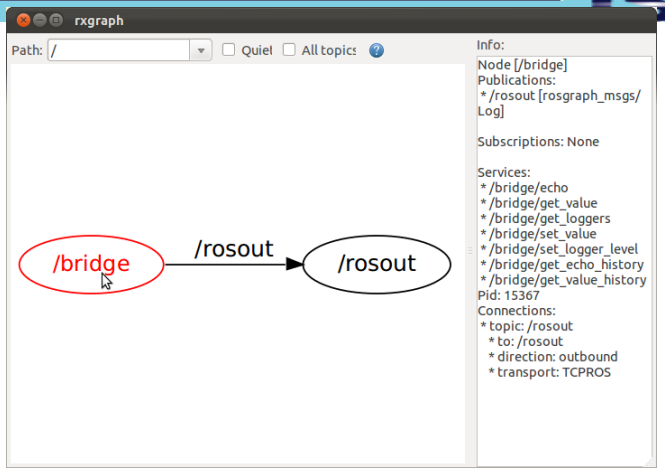


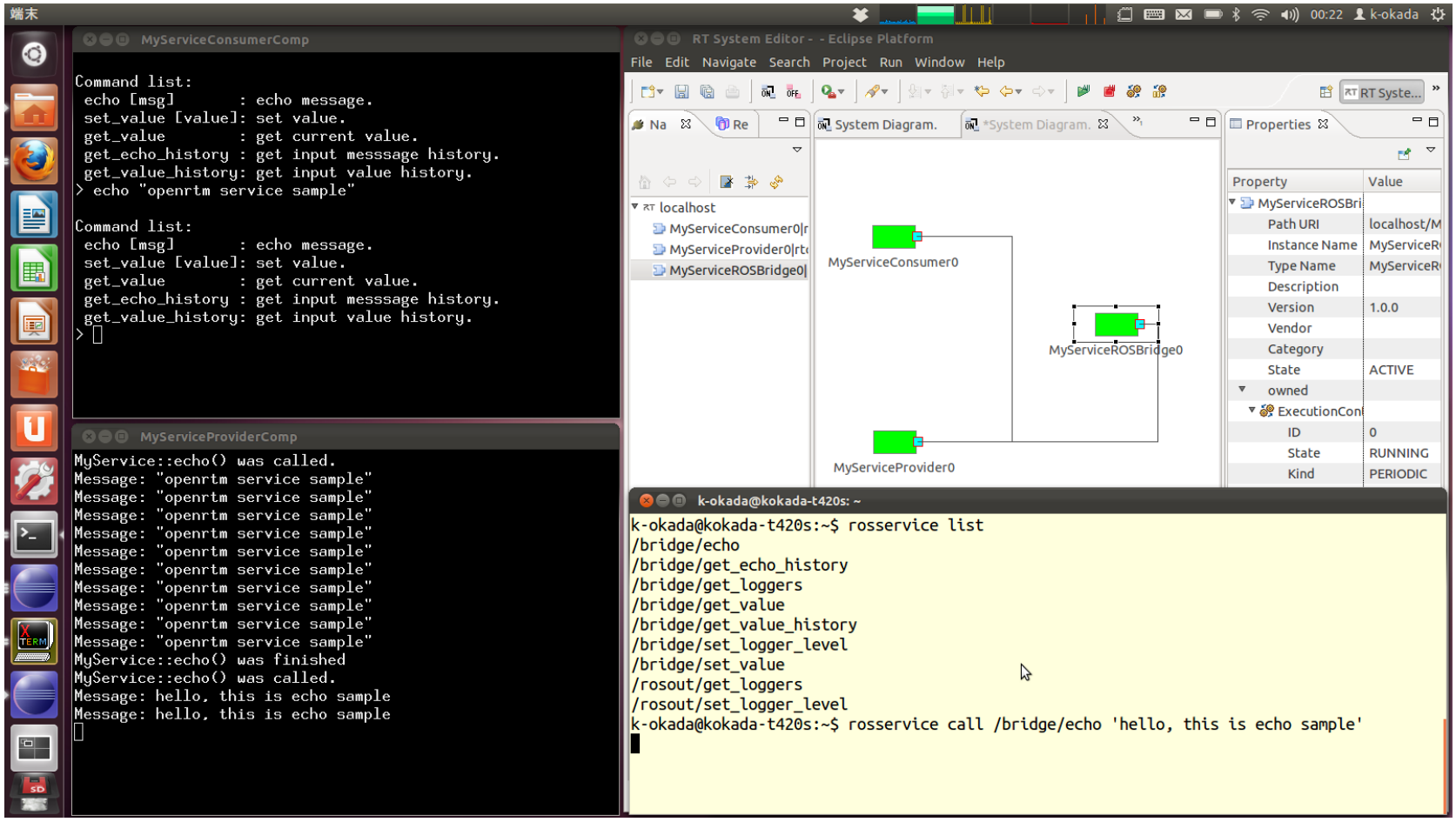
```
$ ls rc_gen/
MyServiceROSBridge.cpp
MyServiceROSBridge.h
MyServiceROSBridgeComp.cpp

$ ls srv
SimpleService_MyService_echo.srv
SimpleService_MyService_get_echo_history.srv
SimpleService_MyService_get_value.srv
SimpleService_MyService_get_value_hisotory.srv
SimpleService_MyService_set_value.srv
```

Generated Service messages

例1：SimpleService

- OpenRTM のSimple Service example を rosservice から呼ぶ





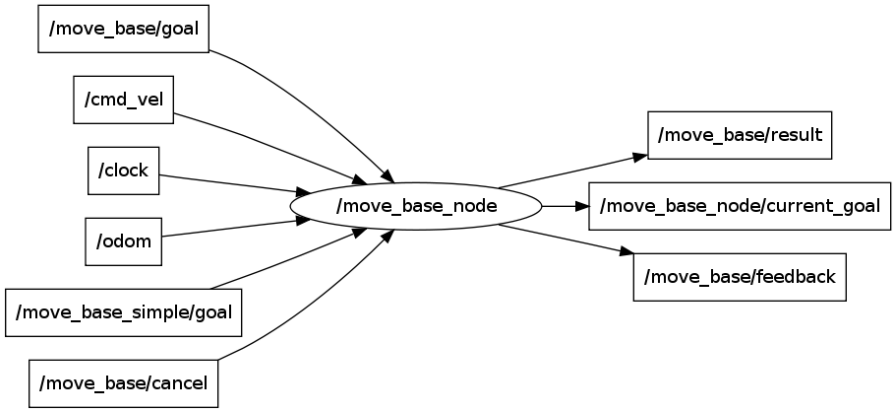
ROSモジュール自動相互運用RTC

ROSモジュールをRTMから利用する

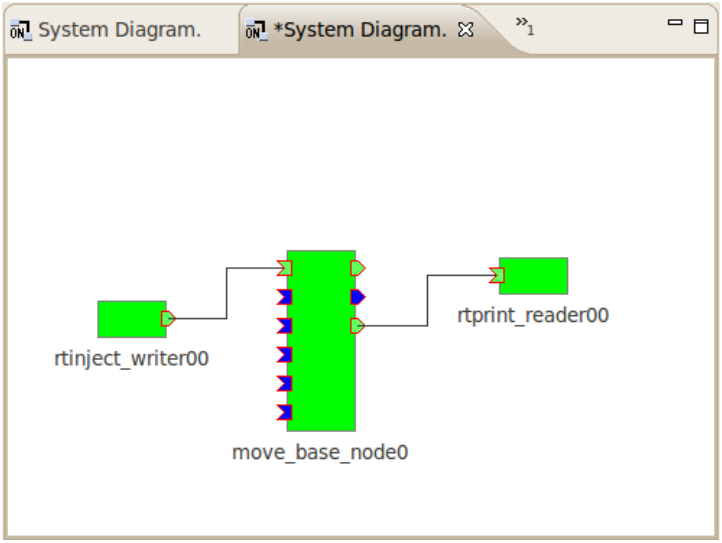
- 稼動しているROSモジュールから自動的にRTMコンポーネントを作成するプログラム

```
$ roslaunch navigation_stage move_base_amcl_2.5cm.launch
$ roslaunch rosnode_rtc stage_sample.launch
```

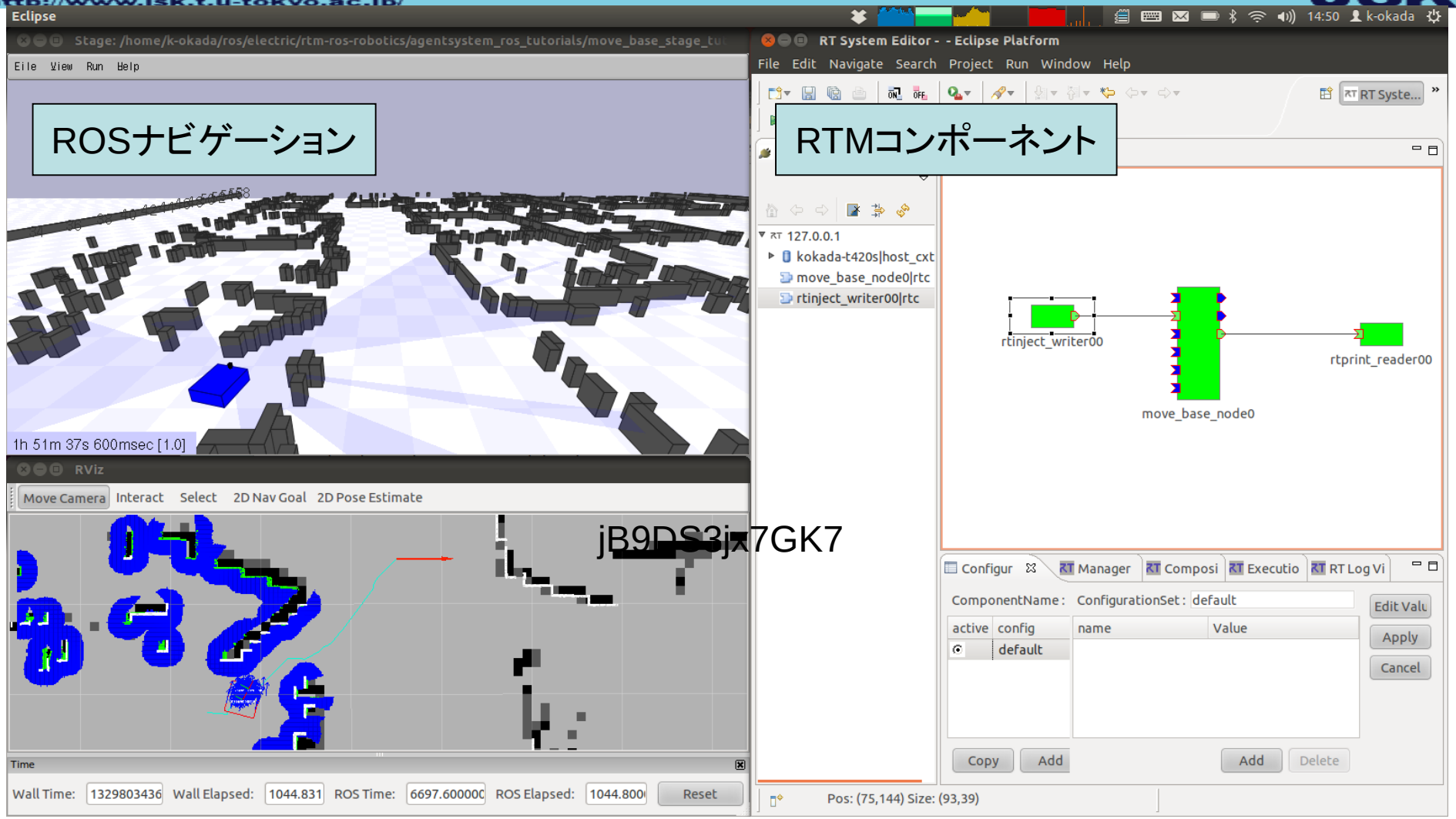
  - 稼動しているROSモジュールの入出力メッセージを解析
  - 対応するOpenRTM用のIDLファイルを出力
  - 生成されたIDLファイルを利用してRTMコンポーネントを生成．ROSメッセージとRTMデータポートの相互変換を提供
- ROSで記述された3000のパッケージを全てRTMから利用するための基盤を構築



ROSのナビゲーションノード



自動生成されたROSナビゲーション機能を提供するRTMコンポーネント



ROSナビゲーションチュートリアルの実行

```
$ roslaunch navigation_stage move_base_amcl_2.5cm.launch
```

ROSノードからRTMコンポーネントの自動生成

```
$ roslaunch rosnode_rtc stage_sample.launch
```

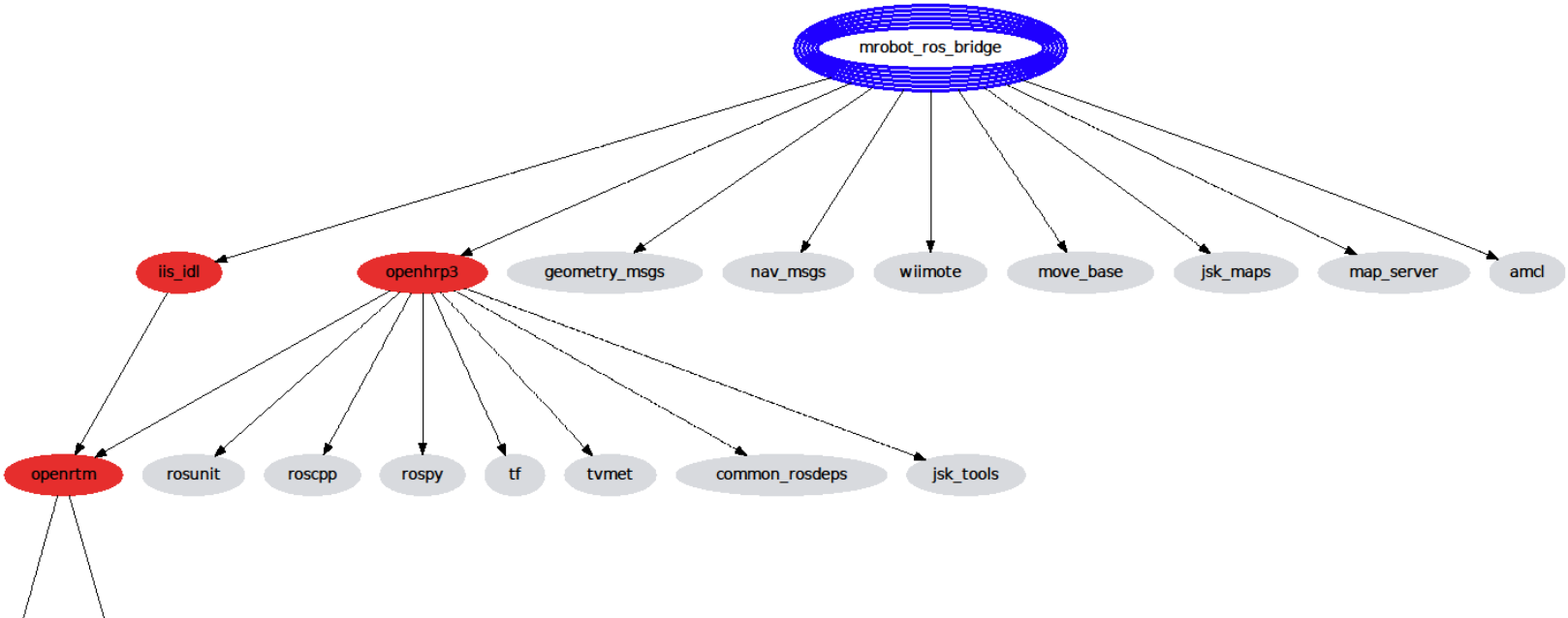
RTMのデータポートからROSのナビゲーションを制御する例

```
$ rosrun rosnode_rtc stage_sample_send_goal.sh
```

複数パッケージの依存関係解消とコンパイル

```
$ catkin build openrtm_ros_bridge
```

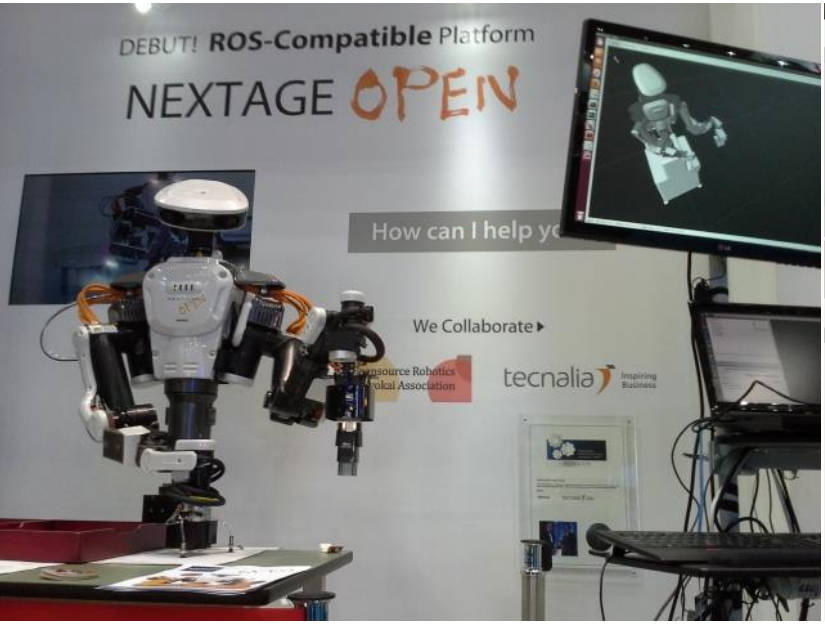
　1コマンドでROS, RTMが混在する複雑な依存関係を解析し，必要な全てのパッケージのコンパイルやリンク，IDL生成を含めてターゲットコンポーネントの生成処理を行う．

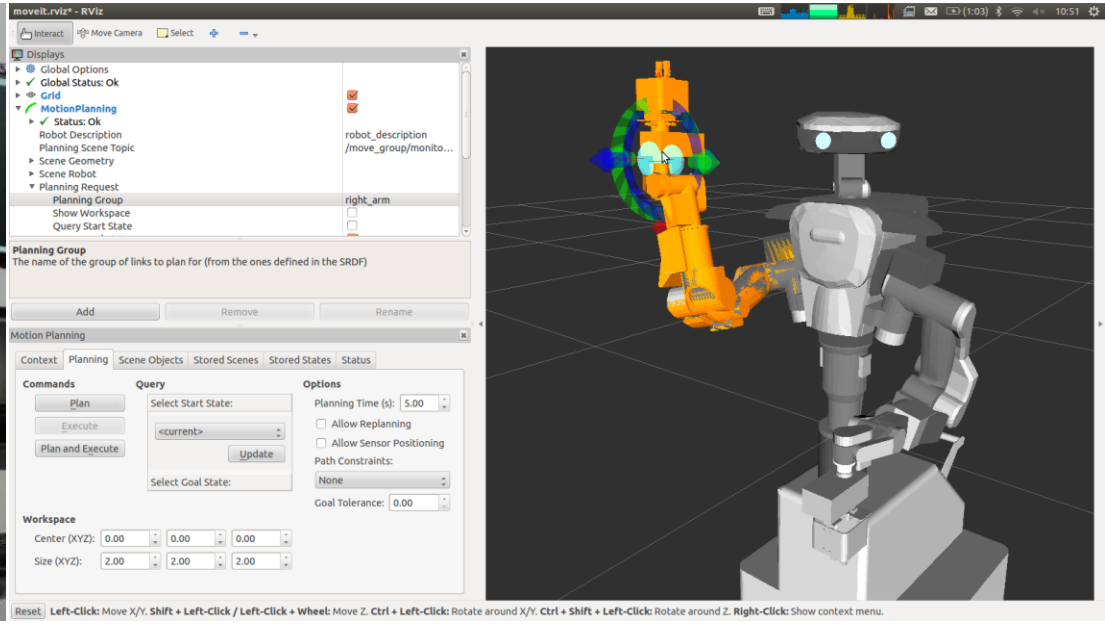


相互運用環境デモ

http://wiki.ros.org/rtmros_nextage

```
$ sudo apt-get ros-indigo-rtmros-nextage
# Launch RTMROS environment
$ source /opt/ros/indigo/setup.bash
$ rtmlaunch nextage_ros_bridge nextage_ros_bridge_simulation.launch
$ hrpsyspy
… OpenRTMで経由でロボットコマンドを実行
$ roslaunch nextage_moveit_config moveit_planning_execution.launch
… ROS経由でコマンドを実行
```





What is ROS exactly?

ROS = 通信ライブラリ + ツール + 基盤 + エコシステム

通信ライブラリ：ROSは分散型計算システムの迅速，簡易な構築のために設計された出版・購読型のメッセージ通信基盤を提供するものである．

ツール：ROSは分散型計算システムの設定，起動，監視，デバッグ，視覚化，ログ取り，停止を行う広範囲なツールを提供するものである．

基盤：ROSは移動，操作，認識を中心に大量の有用なロボットライブラリ群を提供するものである．

エコシステム：ROSはインテグレーションとドキュメンテーションを中心として大規模なコミュニティによって支えられ，発展している．ros.orgは世界中の開発者から提供された大量のROSパッケージを見つけて，習得するためのワンストップサービスである．

Dec 06 '11
Brian Gerkey

http://answers.ros.org/question/12230/what-is-ros-exactly-middleware-framework-operating

Complain!!!

1週間のご滞在、本当にご苦労様でした。

初日に、Eric Bergerが、"COMPLAIN IT"と申しておりましたように、ROS、ソフトウェア、ハード、滞在中のプログラムの組み方に関して、よかったGood
改善すべき点がある。このようにしたらよいのでは。
Imorovement & Recommendation
と思われたことがあれば、細かいことも含め、お気づきの点をすべてお書きください。

コミュニティ型開発 伽藍とバザール19の教訓より

1. はやめのリリース、ひんぱんなリリース。そして顧客の話をきくこと。
2. ユーザを共同開発者として扱うのは、コードの高速改良と効率よいデバッグのいちばん楽ちんな方法。
3. ベータテスタと共同開発者の基盤さえ十分大きければ、ほとんどすべての問題はすぐに見つけだされて、その直し方もだれかにはすぐわかるはず。
    1. 目玉の数さえ十分あれば、どんなバグも深刻ではない
    2. だれかが問題を見つける。そしてそれを理解するのはだれか別の人だよ。そして問題を見つけることのほうがむずかしいとぼくが述べたことは記録しておいてね
4. ベータテスタをすごく大事な資源であるかのように扱えば、向こうも実際に大事な資源となることで報いてくれる。
5. 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。
6. 捨てることをあらかじめ予定しておけ。どうせいやでも捨てることになるんだから（フレッド・ブルックス『人月の神話』第11章）
7. 「完成」（デザイン上の）とは、付け加えるものが何もなくなったときではなく、むしろなにも取り去るものがなくなったとき。
8. あるソフトに興味をなくしたら、最後の仕事としてそれを有能な後継者に引き渡すこと。

The Cathedral and the Bazaar Eric S. Raymond
伽藍とバザール　山形浩生訳 http://cruel.org/freeware/cathedral.html より引用

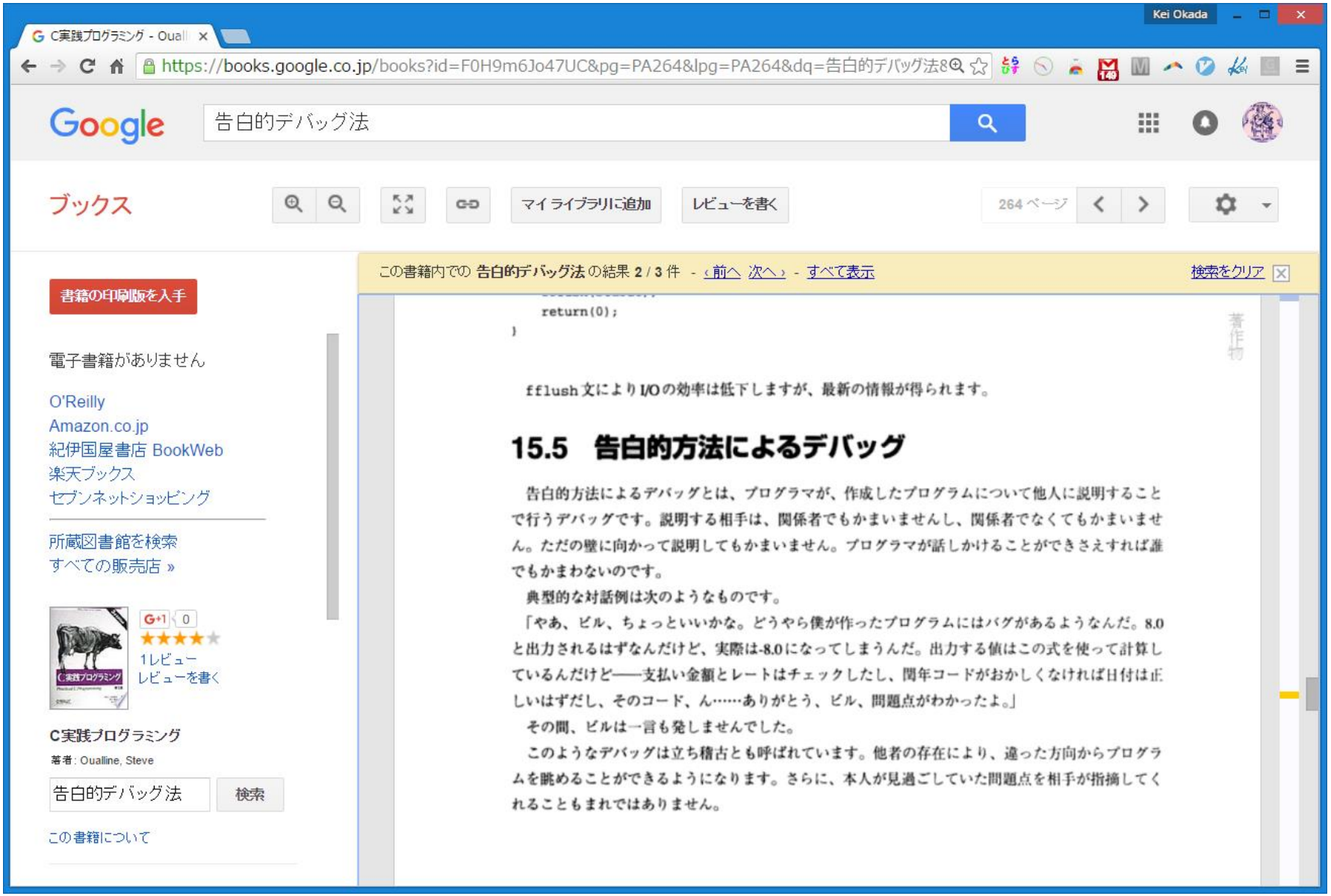
質問しよう！

- #openrtm
- openrtm-users@openrtm.org

- ロボット分野は日本語のコミュニティがあるのが特徴（分野／地域によっては英語しかない場合も）

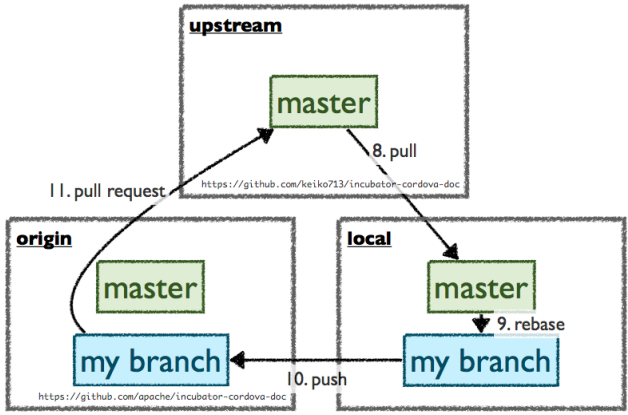
- 英語の世界に出っていても怖くない.
- 世の中の人は積極的に質問している　→

告白的デバッグ法


Kei Okada
C実践プログラミング - Oual
https://books.google.co.jp/books?id=F0H9m6Jo47UC&pg=PA264&lpg=PA264&dq=告白的デバッグ法8
Google
告白的デバッグ法
ブックス
マイ ライブラリに追加
レビューを書く
264 ページ
この書籍内での 告白的デバッグ法 の結果 2 / 3 件 - ‹ 前へ 次へ › - すべて表示
検索をクリア
書籍の印刷版を入手
電子書籍がありません
O'Reilly
Amazon.co.jp
紀伊国屋書店 BookWeb
楽天ブックス
セブンネットショッピング
所蔵図書館を検索
すべての販売店 »
1レビュー
レビューを書く
C実践プログラミング
著者: Oualline, Steve
告白的デバッグ法
検索
この書籍について
return(0);
}
fflush 文により I/O の効率は低下しますが、最新の情報が得られます。
15.5 告白的方法によるデバッグ
告白的方法によるデバッグとは、プログラマが、作成したプログラムについて他人に説明することで行うデバッグです。説明する相手は、関係者でもかまいませんし、関係者でなくてもかまいません。ただの壁に向かって説明してもかまいません。プログラマが話しかけることができさえすれば誰でもかまわないのです。
典型的な対話例は次のようなものです。
「やあ、ビル、ちょっといいかな。どうやら僕が作ったプログラムにはバグがあるようなんだ。8.0と出力されるはずなんだけど、実際は-8.0になってしまうんだ。出力する値はこの式を使って計算しているんだけど——支払い金額とレートはチェックしたし、閏年コードがおかしくなければ日付は正しいはずだし、そのコード、ん……ありがとう、ビル、問題点がわかったよ。」
その間、ビルは一言も発しませんでした。
このようなデバッグは立ち稽古とも呼ばれています。他者の存在により、違った方向からプログラムを眺めることができるようになります。さらに、本人が見過ごしていた問題点を相手が指摘してくれることもまれではありません。

Github：プログラマのSNS

- オリジナルのコードに変更部分を報告する機能（Pull Request)が素晴らしい
- 開発者仲間の活動状況がわかるのが素晴らしい
- プライベートリポジトリも利用可能

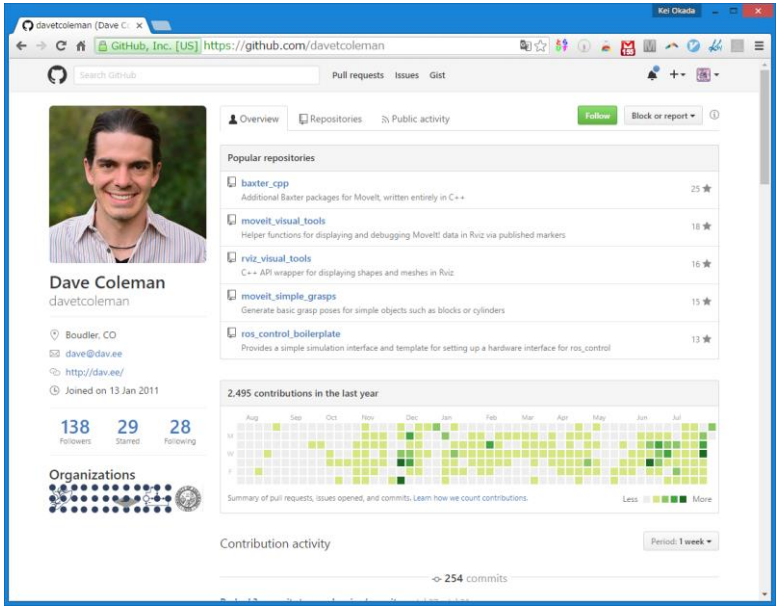


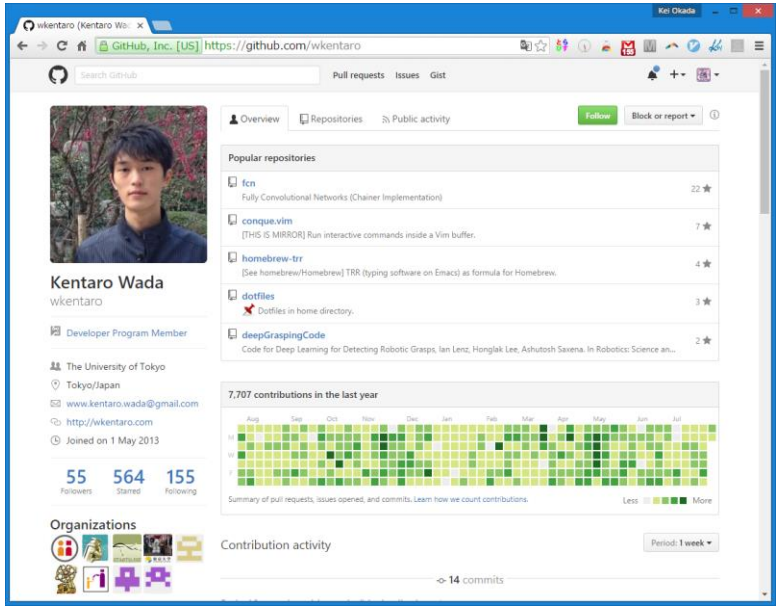
http://kik.xii.jp/archives/179

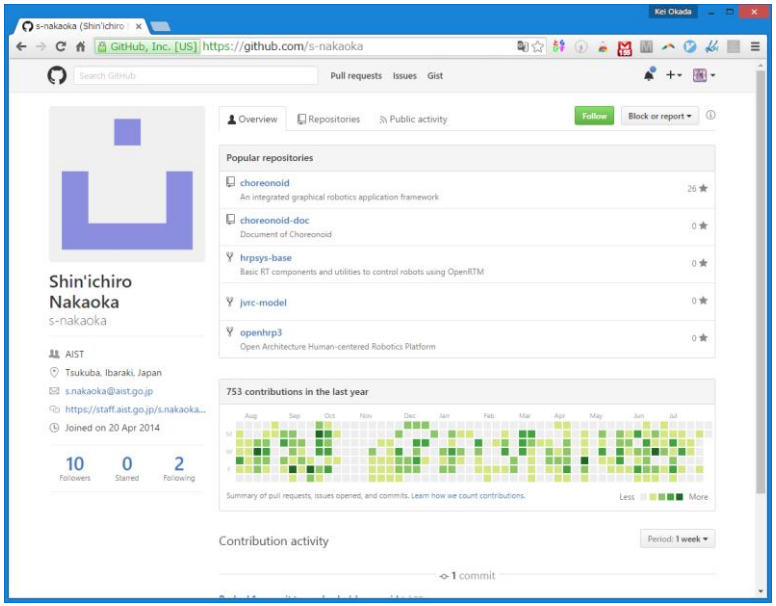
#	User	Contribs	Location	Picture
#1	fkanehiro (Fumio KANEHIRO)	1627		
#2	s-nakaoka (Shin'ichiro Nakaoka)	753	Tsukuba, Ibaraki, Japan	
#3	durka (Alex Burka)	727		
#4	ysuga (Yuki Suga)	227	Tokyo, Japan	
#5	hattorishizuko	152		
#6	jun0	99		
#7	gbiggs (Geoffrey Biggs)	67		
#8	sugarsweetrobotics (Sugar Sweet Robotis)	41	Tokyo, Japan	
#9	maniram477 (Manikandan R)	17	India	
#10	futail2a (asato)	7		

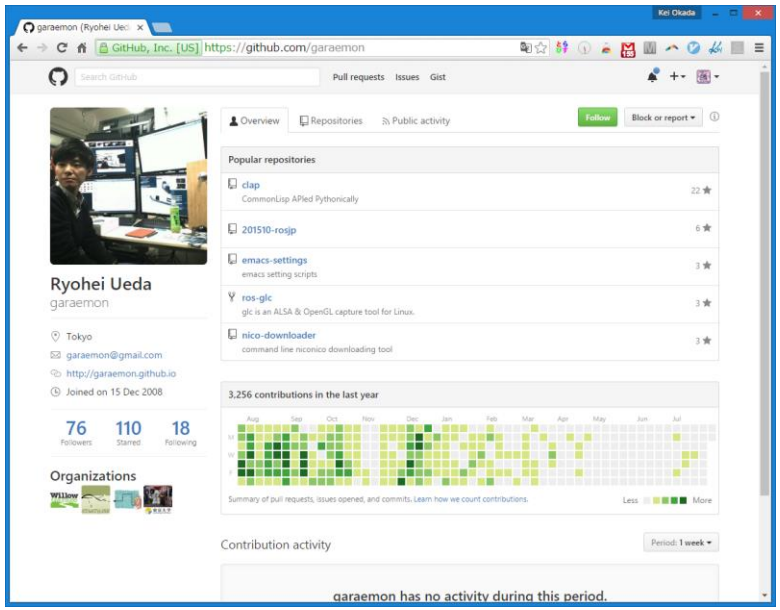
#	User	Contribs	Location	Picture
#1	wkentaro (Kentaro Wada)	7698	Tokyo/Japan	
#2	dirk-thomas (Dirk Thomas)	5592	Mountain View, CA	
#3	k-okada (Kei Okada)	5306	Tokyo, Japan	
#4	mkoval (Michael Koval)	2953	Pittsburgh, PA	
#5	DLu (David V. Lu!!)	2576	Pittsburgh, PA	
#6	davetcoleman (Dave Coleman)	2495	Boudler, CO	
#7	DaikiMaekawa (Daiki Maekawa)	2269	Japan/Switzerland	
#8	felixduvallet (Felix Duvallet)	2176		
#9	130s (Isaac I.Y. Saito)	2127	Sacramento, CA / Tochigi, Japan	
#10	stonier (Daniel Stonier)	2064	Seoul, Korea	
#11	asmodehn (AlexV)	1899		
#12	vrabaud (Vincent Rabaud)	1852	Paris, France	
#13	wjwwood (William Woodall)	1778	Mountain View, CA	

Github のステータスページは履歴書代わり









まとめ

- 国内のロボットコミュニティを盛り上げていきましょう！

- 私からの提案
  - COMPLAIN
  - サマーキャンプ中は1日1質問
    - （#openrtm, openrtm-users）！
  - サマーキャンプの恥はかき捨て！
  - 講師の先生方に教わった内容もポスト
  - 自分が分からないことは，他の人もわからない