ChoreonoidとOpenHRIを用いたシステム構築事例

産業技術総合研究所
原　　功

Choreonoid

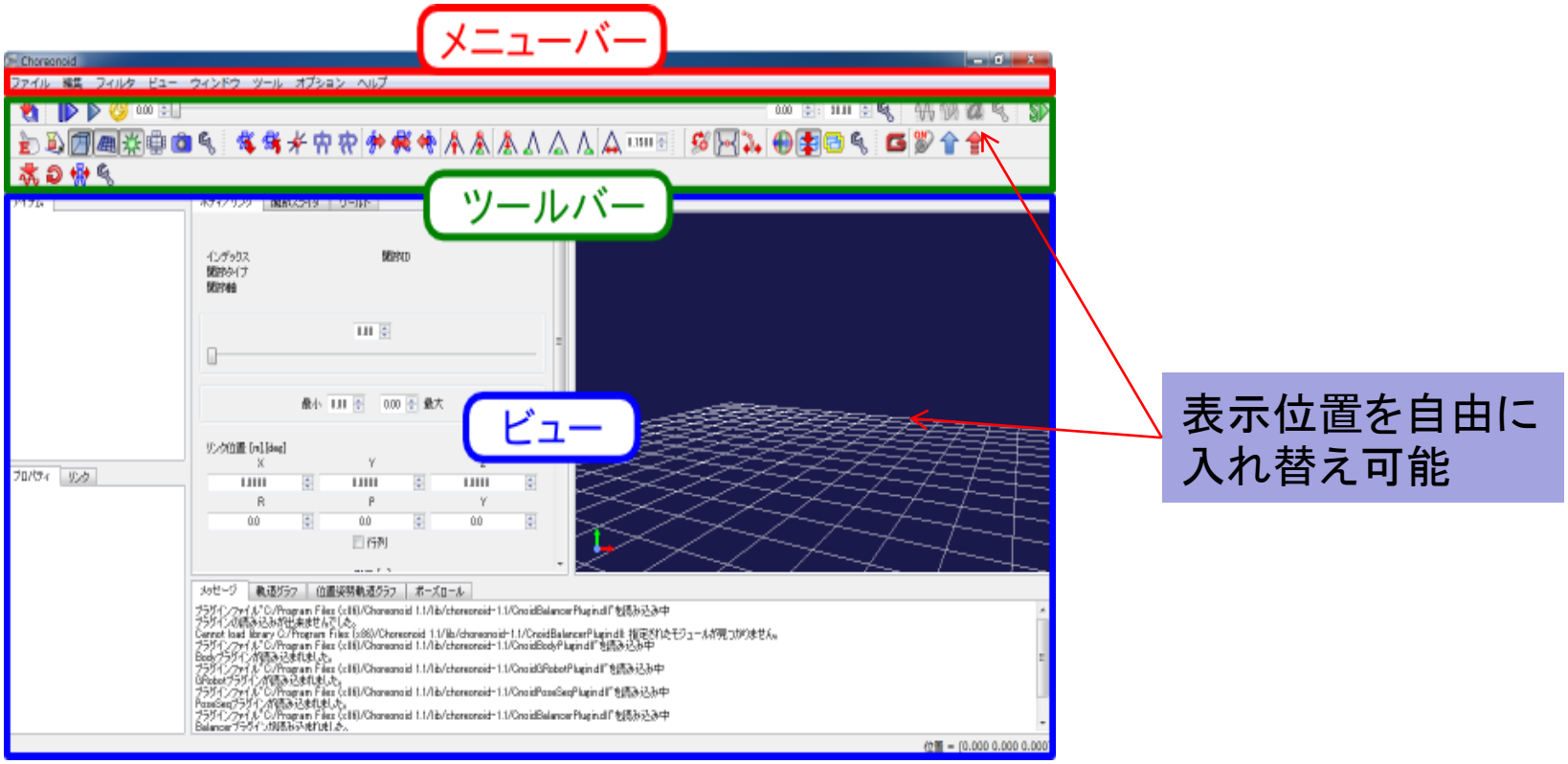
Choreonoid

- オープンソースのロボット用統合GUIソフトウェア
  - 動作振り付けソフトウェアとして
    - キーフレームベースの動作生成
    - CGの動作生成のようなUI
  - 動力学シミュレータとして
    - OpenHRP3, ODE,　Bullet Physics Engineなど利用可
  - 開発フレームワークとして
    - ロボットの幾何学、運動学、動力学や動作計算に関する各種モデル計算
    - ロボットの状態や動作データを可視化、入力、編集する各種GUI
    - 3DCGによるロボットモデルのレンダリング
    - 各種データの統合的な管理とファイル処理
    - 実機ロボットとの接続

Chreonoidを自分のロボットで利用する

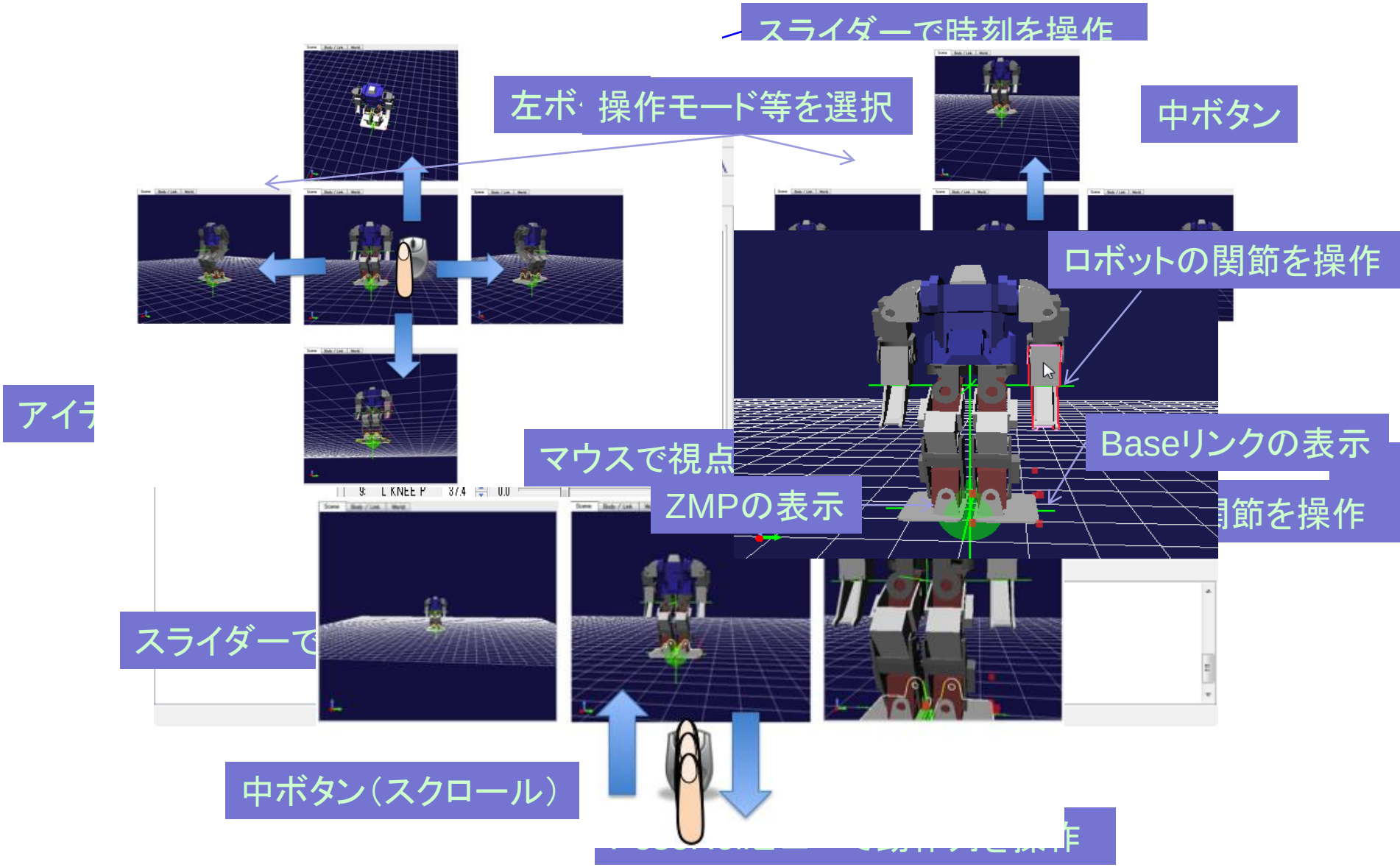
- Chreonoidを利用するに必要なもの
  - 対象となるロボットのモデル
    - ロボットのパーツのVRMLモデル
    - 各パーツの重心位置と慣性モーメントマトリックス
  - IK計算用プログラムモジュール
    - 独自開発
    - OpenRAVE提供のikfastを利用する
  - ロボット制御ソフトへ命令を与えるためのプラグイン

Choreonoidの基本的な操作

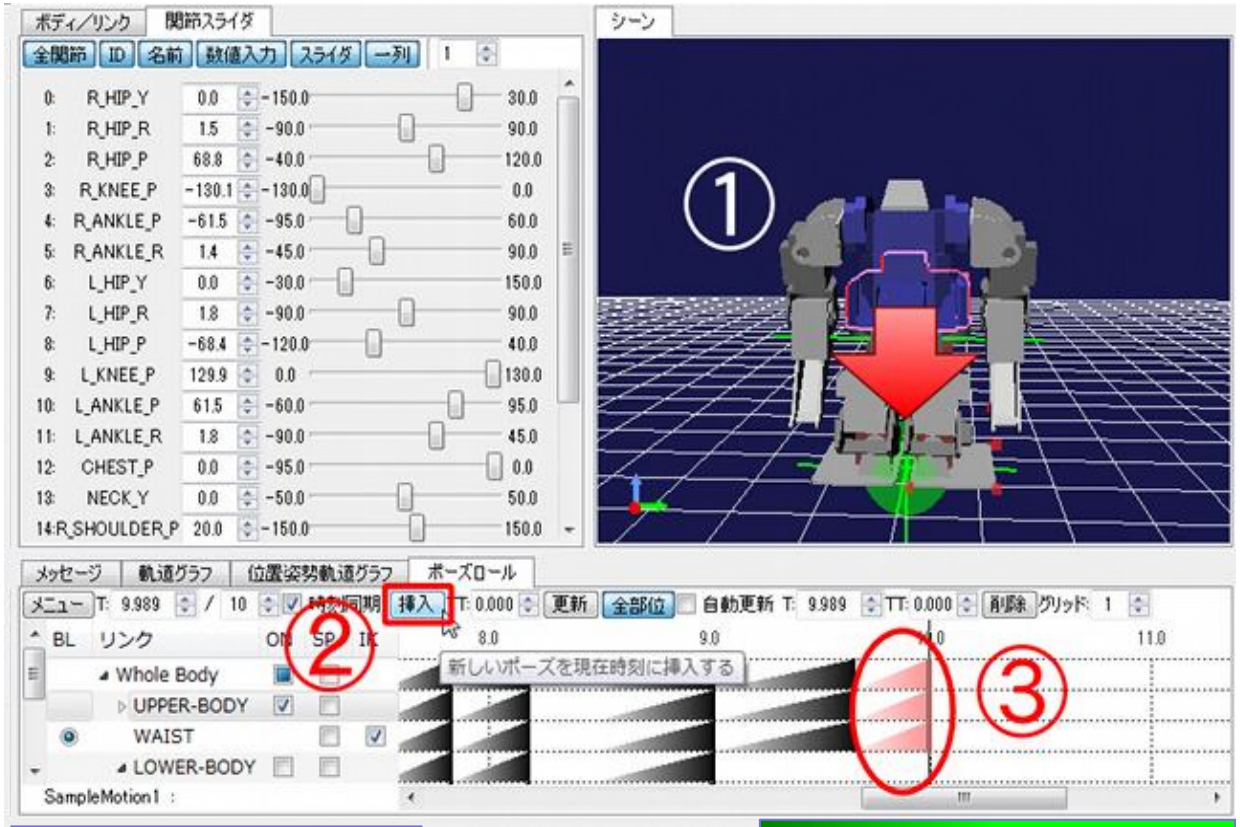


- メニューバー
  - 本メニューバーに格納されているメニューを用いることで、Choreonoidの各種操作を行うことが出来ます。メニュー項目はプラグインによって追加することも可能です。
- ツールバー
  - ツールバー領域には、ボタンやスライダ、数値入力ボックス等のGUI部品で構成されるツールバーが配置されます。ツールバーは機能ごとにグループ化されたものとなっており、各ツールバーの左端をマウスでドラッグすることで簡単に好みの場所に移動させることができます。
- ビュー
  - ビューは、Choreonoidにおいて各種情報を表示・編集するためのウィンドウ領域です。タブ内に格納される各矩形領域がひとつのビューに対応します。
  - Choreonoid本体に備わった基本的なビューとして、各種データ・モデル等（アイテム）を管理する「アイテムビュー」、各アイテムのプロパティを表示・編集するための「プロパティビュー」、3D表示にてモデルの確認や編集を行うための「シーンビュー」、テキストメッセージが出力される「メッセージビュー」などがあります。

Choreonoidの基本的な操作



キーポーズの変更と追加



キーポーズを更新

1. 動作の
2. キーポー

1. キーポーズを生成
2. キーポースを挿入
3. 動作時間を調整

- 動作パターンの編集は、キーポーズの追加と変更の繰り返し

Choreonoidにおけるロボットの動作表現

- Choreonoidでは、ロボットの動きをキーポーズの連続として表現する

- キーポーズは、動作中の状態が変化するポイントの姿勢

- 無理な姿勢のキーポーズは、身体バランス補正を行い、安定動作を生成する。

- ロボットへは、制御時間ごとの目標姿勢に変換して、命令列を与えて実行させる

OpenHRI

コミュニケーション知能モジュール群

OpenHRIとは

- OpenHRIは、音声認識・音声合成・対話制御など、ロボットのコミュニケーション機能の実現に必要な各要素を実現するコンポーネント群

- フリーで利用できる各オープンソースソフトウェアを使い易いコンポーネントとしてまとめたもの

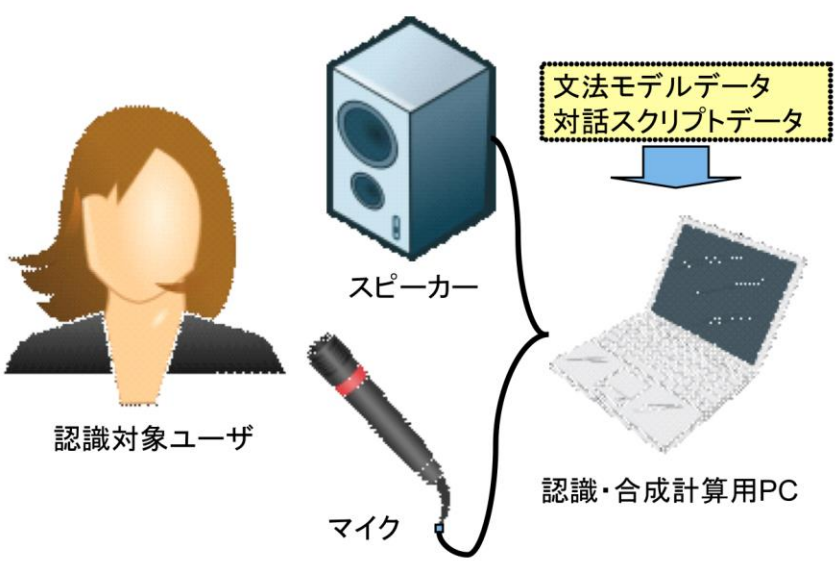
- 知能化PJにおいて開発され、EPL-1.0にて公開

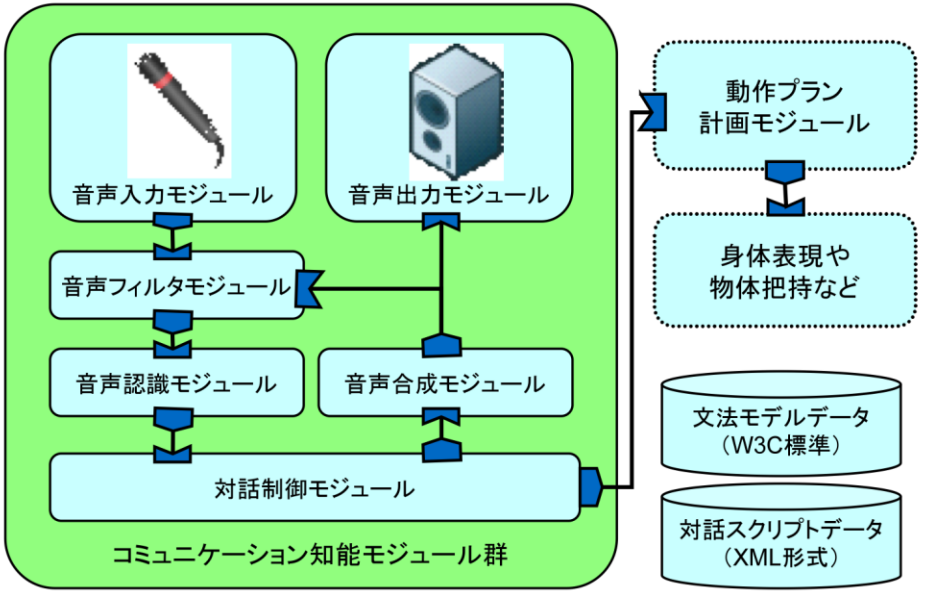
  http://openrtc.org/OpenHRI/index.html

  http://openhri.net

OpenHRIで提供する機能

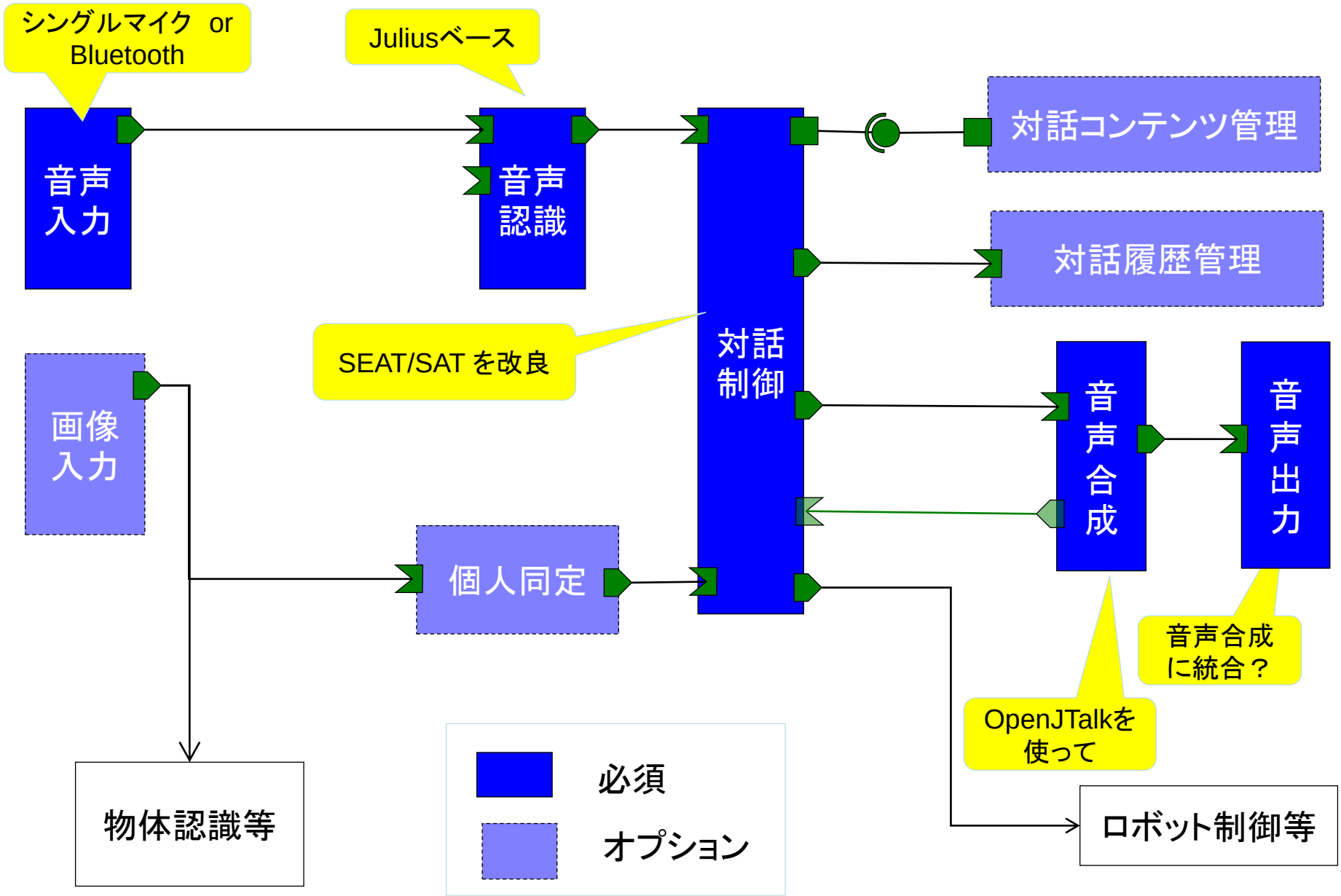
- ロボットのコミュニケーション機能の実現に必要な各要素
  （音声入出力・音声認識・音声合成・対話制御など）





※IFは全て各社共通形式（どのモジュールでも差し替え可能です）

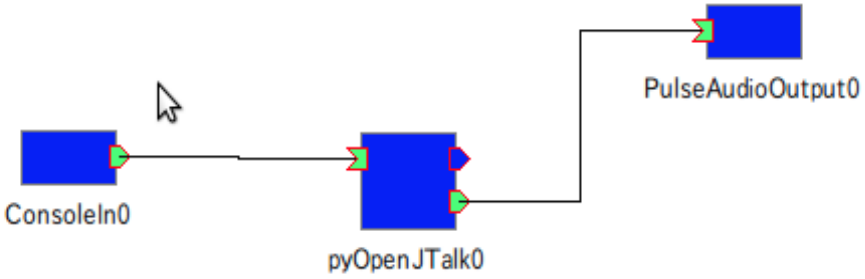
OpenHRIで提供する機能

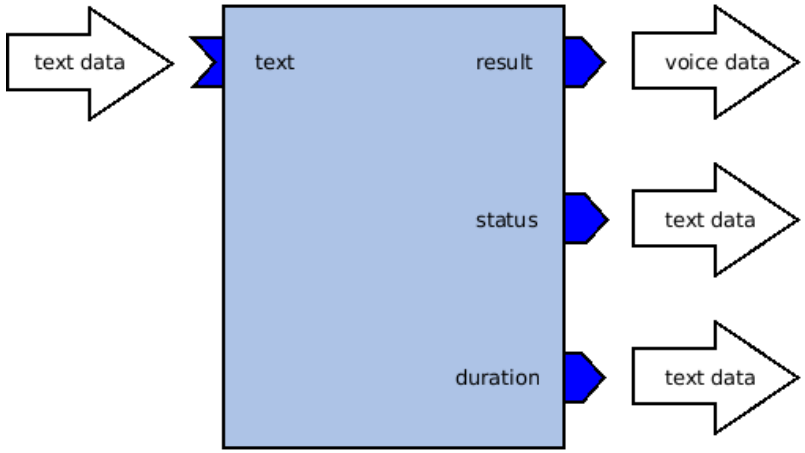


話す機能

- 音声合成コンポーネントは、フリーで利用できる音声合成エンジン(OpenJTalk、Festival) 、を利用
- テキストを入力とし、音声データ(Mono,16kHz, 16bit)を出力

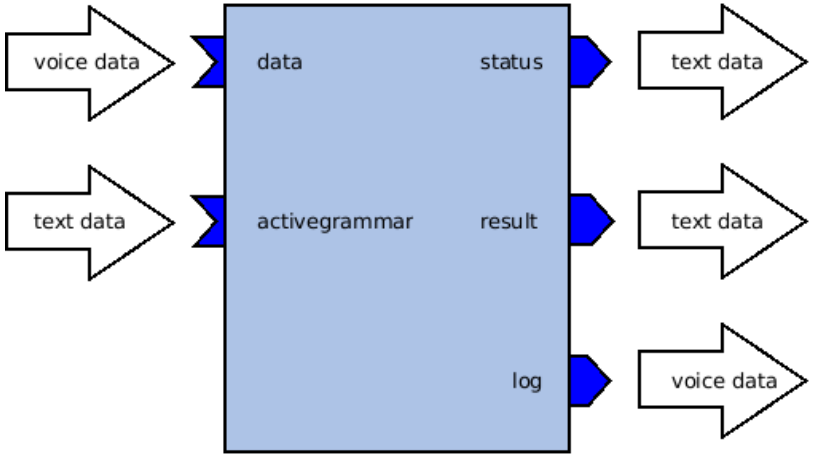
参考システム構成





聴く機能

- オープンソースの高性能な汎用大語彙連続音声認識エンジンJuliusを使った音声認識機能
- 受け取った音声データ(16bits, 16KHz)を音声認識して認識テキストに変換する
- 音声認識は、予め用意された音声認識文法ファイルに従って実行される

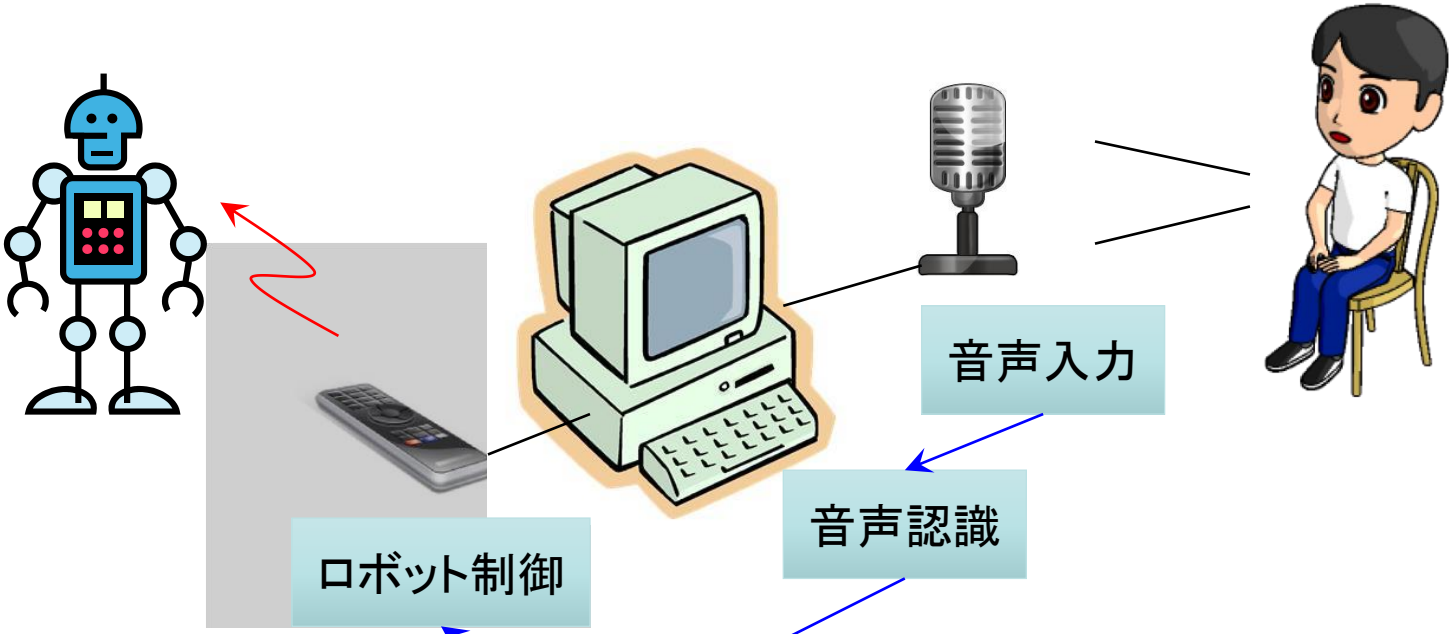


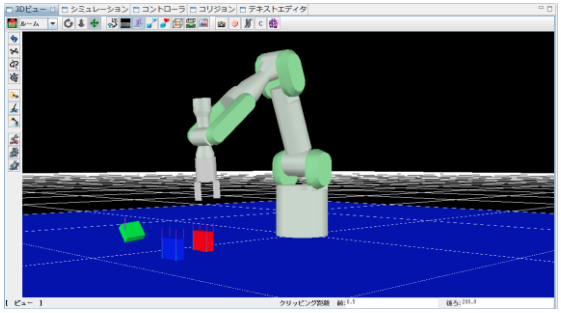
対話を制御する機能

- SEAT(Speech Event Action Transfer)
  - ロボットとの対話を実現するための小さく軽量な対話エンジン
  - ロボットの対話動作は、音声認識結果（条件）とシステムの動作（アクション）のペアを基本として対話動作を記述
  - 状態遷移モデルに基づく対話管理
  - XMLベースのSEATMLにより対話スクリプトを記述

OpenHRIを使ったシステム例

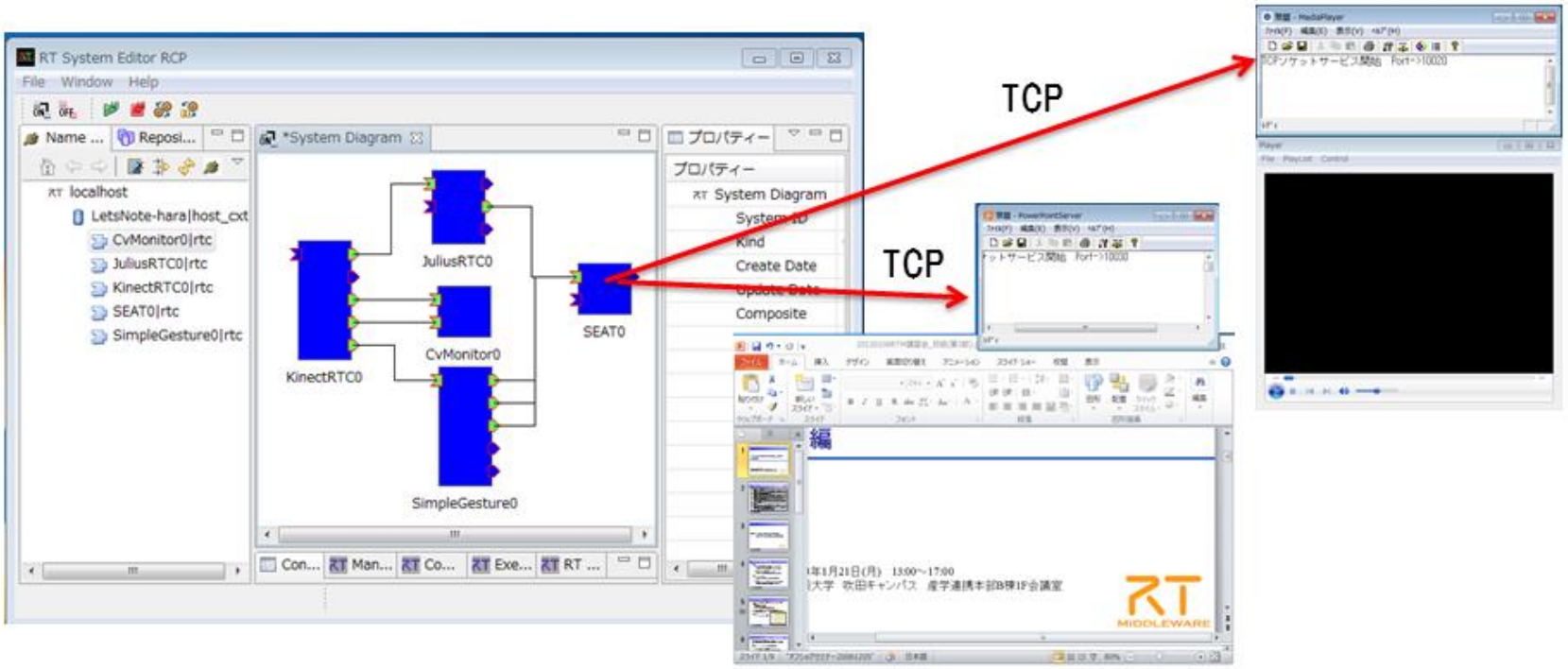
- ロボットに音声インタフェースをつける





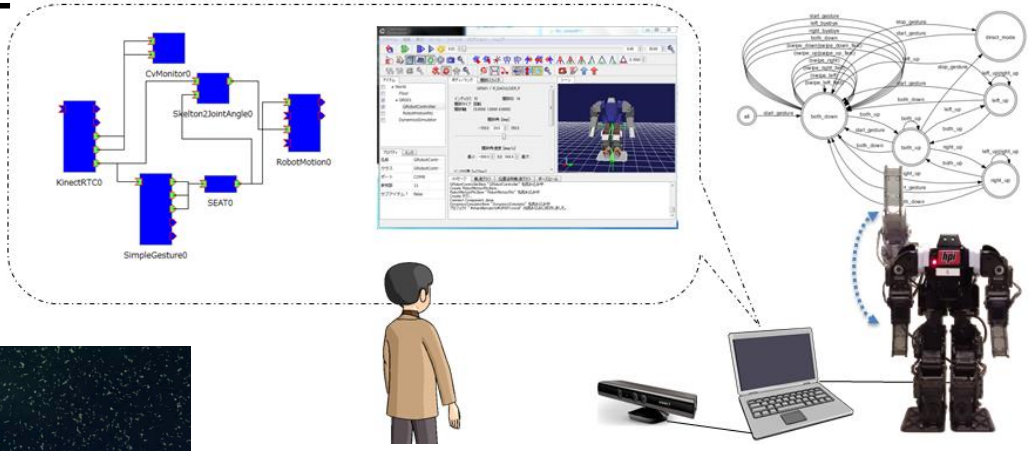
対話型システムの例

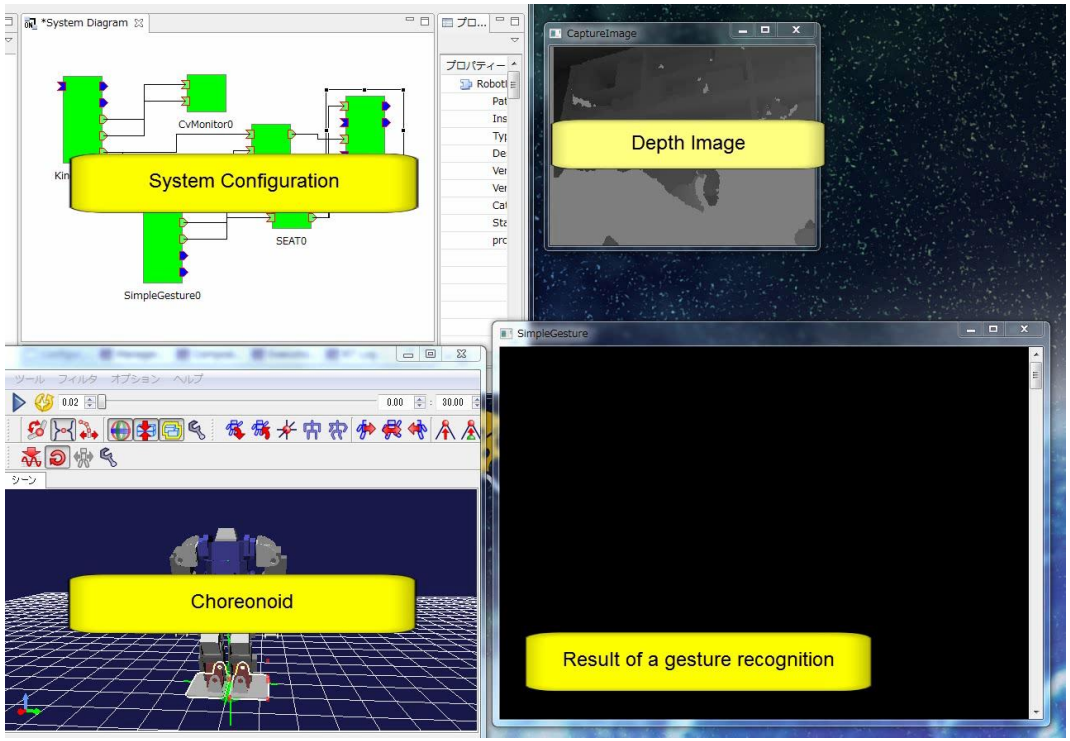
- 音声認識でパワポ、ムービーを制御
  - JuliusRTC
  - SEAT
  - パワポサーバー、ムービーサーバー



対話型システムの例

- 人型ロボットをジェスチャで制御
  - Kinectコンポーネント群
  - 簡易ジェスチャ認識





- 2つの操作モード
  - ダイレクト操作モード
  - ジェスチャコマンドモード

eSEAT

- 音声対話の対話制御から汎用の状態遷移コンポーネントへの拡張
  - OpenRTM-aistで取り扱う標準、拡張データ型への対応
  - 独自データ型への対応
  - 簡易GUIパネルの作成機能
  - Pythonスクリプトにより様々な制御機能
  - 外部Pythonスクリプトで機能拡張
  - 外部コマンド実行機能

SEATML

- eSEATの内部状態、イベント処理、GUI等の設定する
- XMLで記述
- 基本的にSEATで使っていたものも動作する

```
<?xml version="1.0" encoding="UTF-8" ?>
  <seatml>
    <general>
     - データポートの定義
     - preload scripts
    </general>
    <state name="st1">
     - ruleの定義
     - GUI部品の定義
    </state>
    <state name="st2">
            ...
            ...
    </state>
     ....
     ....
  </seatml>
```

データポート作成機能

- SEATMLの最初のgeneral要素内で設定
- adaptor要素でOpenRTM-aistのデータポートまたはTCP Socketを生成
- 独自データ型のデータポート生成時には、script要素を使ってクラスをインポート

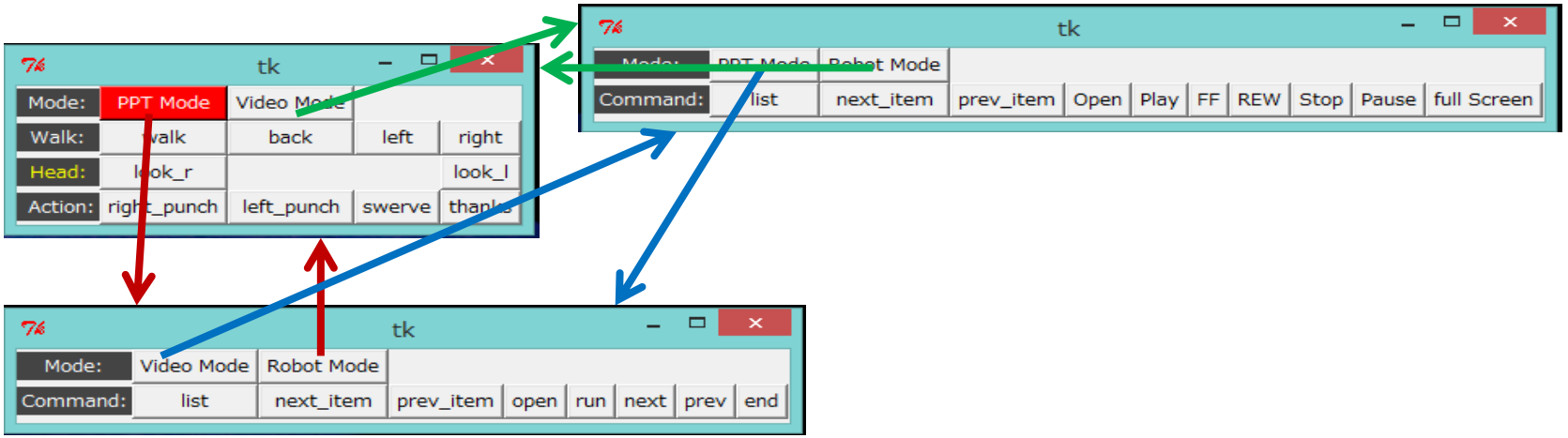
Script要素内は、Pythonスクリプトになるので、インデントに注意

```
<general name="LeapMotion">
 <script>import ssr</script>
  <adaptor name="gesture_out" type="rtcout" datatype="TimedString" />
  <adaptor name="hands_out" type="rtcout" datatype="TimedFloatSeq" />
 <adaptor name="frame" type="rtcin" datatype="ssr.Frame" />
</general>
```

独自データ型のときにはnamespaceを確認

簡易GUI作成機能

- システム開発中に使うテスト用GUIの作成
- 内部状態を持ちパネルレイアウトを自動変更
  - label要素： tk.Labelを生成
  - button要素: tk.Buttonを生成
  - input要素：tk.Entryを生成
  - text要素：tk.Textを生成



イベント処理

- eSEATでは、adaptor要素で設定したポートからデータを受信時、内部状態遷移時またはGUIアイテムを押下等を行った時にイベント処理が実行される
- イベント処理は、主としてrule要素で設定する
- GUIアイテムであるbutton要素、input要素、text要素に対するイベント処理は、コンポーネントが有効化されていなくても実行される
- 内部状態遷移時のイベント処理は、onentry要素、onexit要素で設定
- イベント処理用コマンド
  - message要素： sendto属性値のadaptoへtextデータを出力
  - script要素：textをPythonスクリプトとして実行、sendto属性値を設定の場合には、rtc_resultに代入されたデータを送信
  - shell要素：textをshellに渡してコマンドを実行。subprocess.Popen関数で実装されている
  - statetransition要素： eSETA内部状態を遷移させる

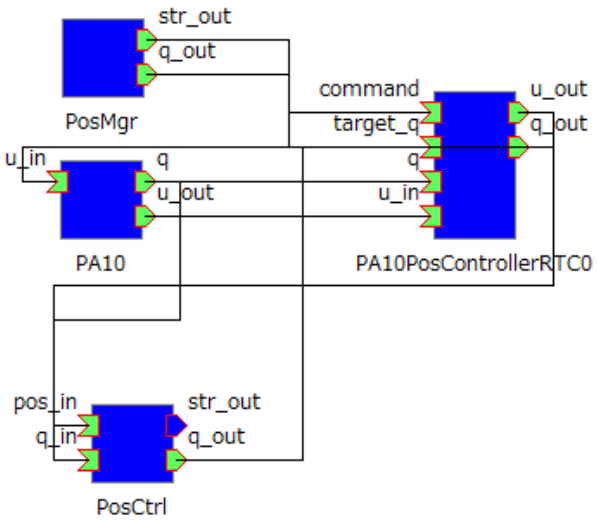
eSEATに関するその他の情報

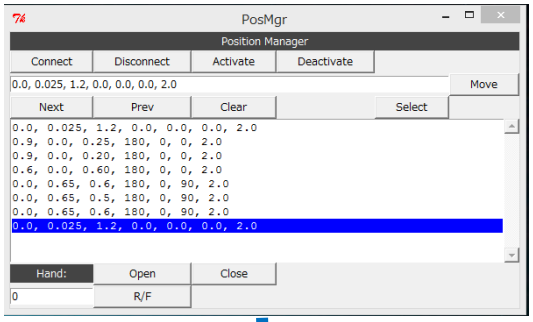
- eSEATを使ったサンプル
  - 単純な文字列入出力パネル
  - 電卓アプリ
  - rtc_handleを使ったRTC操作パネル
  - Choreonid-1.4でPA10の位置制御パネル
  - LeapRTCのデータ閲覧パネル
  - LeapRTCとChoreonoidでGR-001を動かす

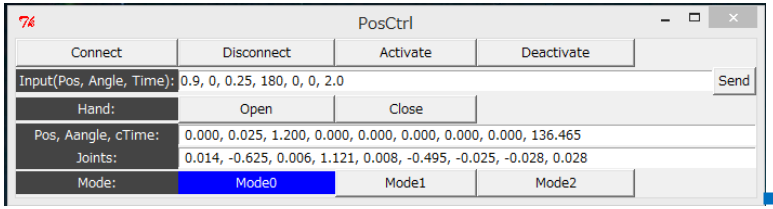
- eSEATの詳細については
  - http://hara.jpn.com/_default/ja/Software/eSEAT2.html

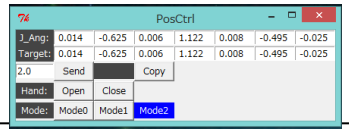
Choreonoid内のPA10の操作パネル

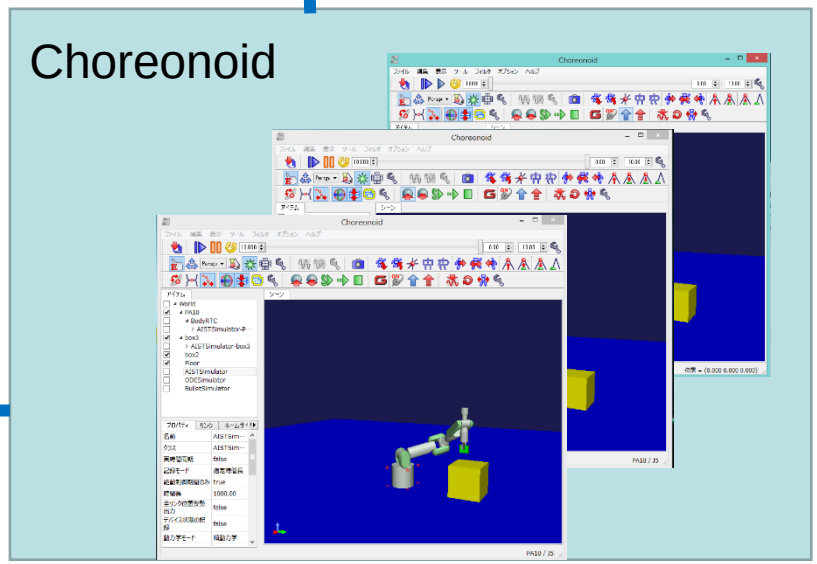
- ChoreonoidのサンプルPA10Pickupを参考に目標位置姿勢入力可能なプラグインを作成
- 上記プライグインに対応した操作パネルの作成



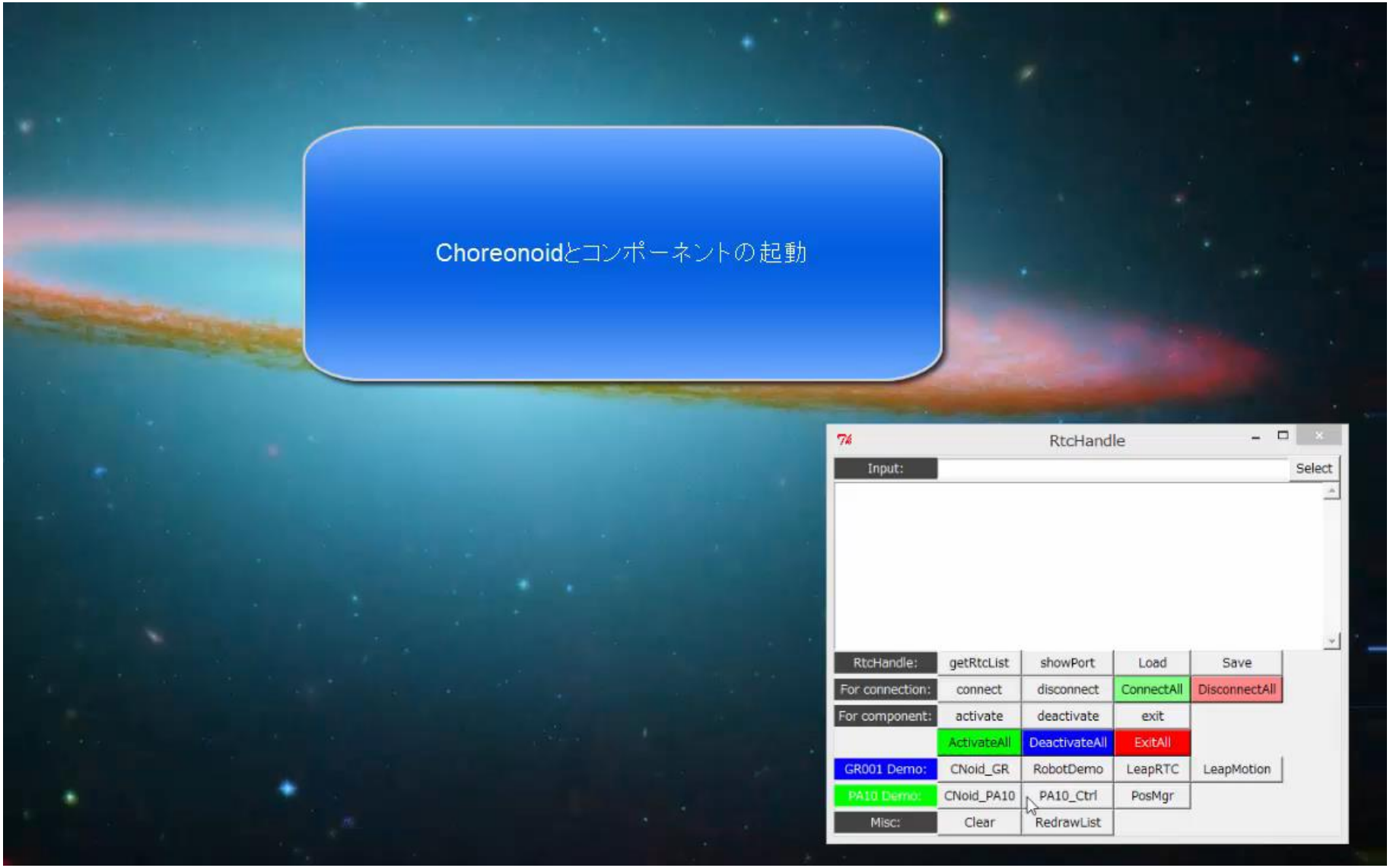






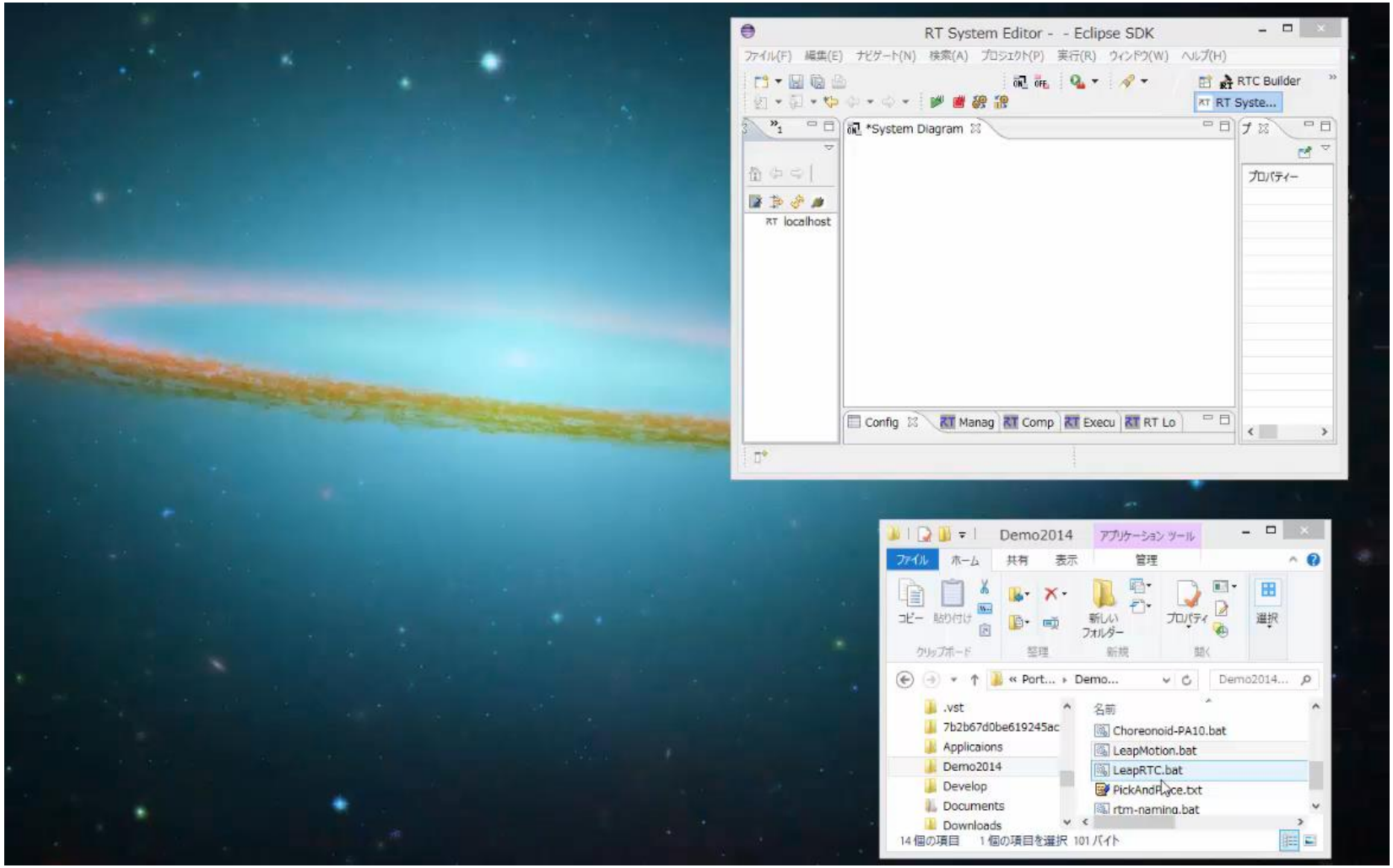


Choreonoid内のPA10の操作パネル



LeapRTCとの連携GUI

- SSR　菅さんの作成したLeapRTCのデータ表示パネル



Choreonoid+LeapRTC+GR001

- LeapRTCを使ってGR-001の操作する
- 以前開発したRobotMotionRtcPluhginと連携
- GR-001への動作は、Choreonoidを介して行う