GraspPlugin Manual

GripperManipulation をRTコンポーネントに改造する

水, 07/13/2011 - 18:38 — asahi

Choreonoid/extPlugin/graspPlugin/GripperManipulation というプラグインに、RTコンポーネントの「口」を取り付ける。

オリジナルの GripperManipulation について

双腕ロボットの把持計画を行う Grasplan プラグインであり、ファイル構成は以下のようになっている。

- CMakeLists.txt cmake 定義ファイル
- GripperManipulationMain.cpp Choreonoidプラグイン ManipPlugin クラスのソース
- ManipBar.cpp Choreonoid ツールバー ManipBar クラスのソース
- ManipBar.h Choreonoid ツールバー ManipBar クラスのヘッダ
- ManipController.cpp 把持計画コントローラ ManipController クラスのソース
- ManipController.h 把持計画コントローラ ManipController クラスのヘッダ
- PRM 把持計画データファイルのディレクトリ

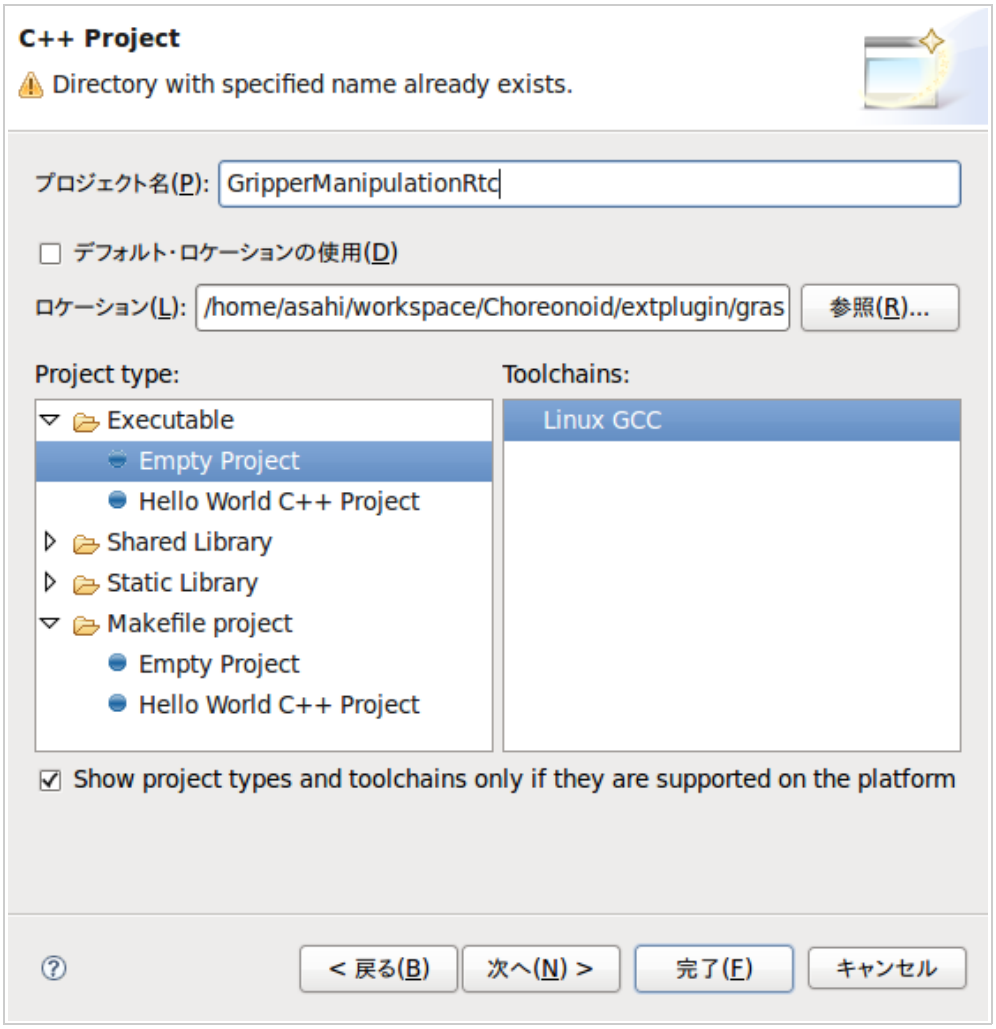
ここで、プログラムの構造がわかりやすいよう、GripperManipulationMain.cpp を、ManipPlugin.cpp にリネームする。
合わせて CMakeLists.txt も修正して、コンパイルできることを確認する。

RTC BuilderでRTコンポーネントのスケルトンを作成する

下準備として、まず GripperManipulation の下に rtcディレクトリを作成し、ここに GraspController.idl を置く。

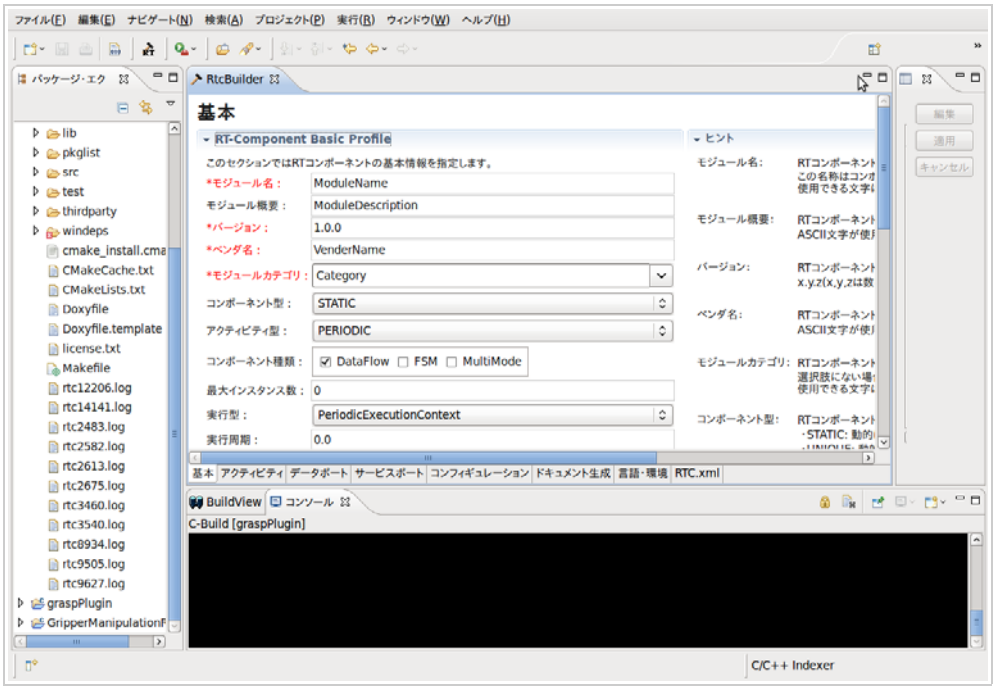
さらに、Eclipse を起動し、このrtc ディレクトリにダミーの新規 Eclipse プロジェクトを設定する。
ウィザード選択では「C++ Project」を選び、各項目を以下のように設定する。

- プロジェクト名 GraspControllerRTC
- ロケーション (ユーザーの作業ディレクトリ)/Choreonoid/extplugin/graspPlugin/GripperManipulation/rtc
- Project type: Executable - Empty Project



完了ボタンを押すと、C++ パースペクティブを開きますか? というダイアログが出るが、そのまま閉じればよい。

これで下準備は完了したので、Eclipse で RTC Builder パースペクティブを開く。
RTのアイコンをクリックすると、RtcBuilder ビューが開くので、タブごとに必要な情報を設定していく。



「基本」タブ

モジュール名 GripperManipulation

モジュール概要 Gripper Manipulation

ベンダ名 AIST

モジュールカテゴリ TestInterface

Output Project GripperManipulationRTC（先に作成したプロジェクトを選択する）

その他の項目はデフォルトのままでよい。

「言語・環境」タブ

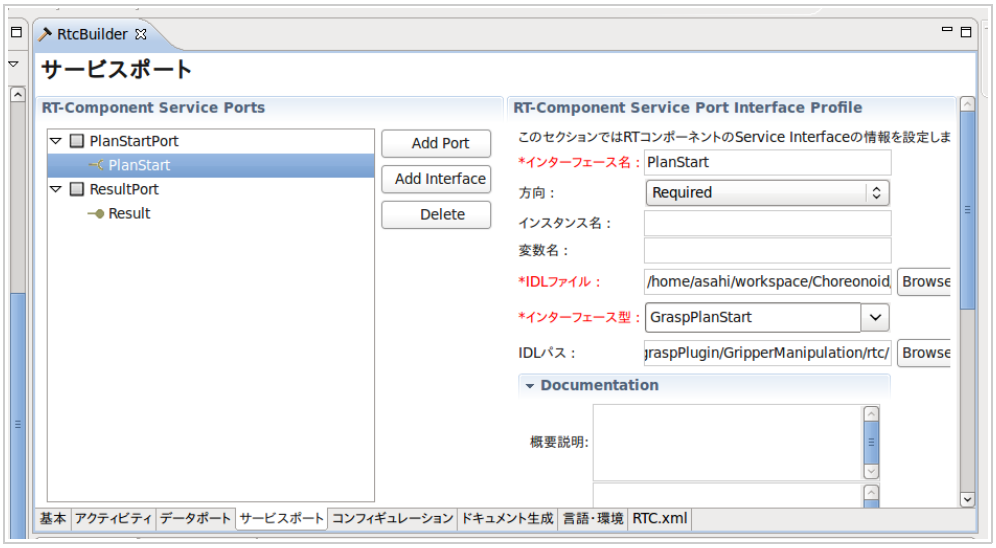
C++を選択する。

「サービスポート」タブ

Add Port ボタンを押す。

sv_nameというポートができるので、マウスでクリックして右側にプロファイル設定画面を表示する。

ポート名をPlanStartPortに設定して、今度はAdd Interfaceボタンを押し、新しくできたインターフェースに以下の情報を設定する。



インターフェース名 PlanStart

方向 Required

IDLファイル /home/asahi/workspace/Choreonoid/extplugin/graspPlugin/GripperManipulation/rtc/GraspController.idl

インターフェース型 GraspPlanStart(選択項目)

IDLパス /home/asahi/workspace/Choreonoid/extplugin/graspPlugin/GripperManipulation/rtc/

同じように、ポート ResultPort とインターフェース Result を作成する。

インターフェース名 Result

方向 Provided

IDLファイル /home/asahi/workspace/Choreonoid/extplugin/graspPlugin/GripperManipulation/rtc/GraspController.idl

インターフェース型 GraspPlanResult

IDLパス /home/asahi/workspace/Choreonoid/extplugin/graspPlugin/GripperManipulation/rtc/

コード生成

以上で RTC Builder の設定は完了である。
再び基本タブを開き、コード生成ボタンを押すと、Choreonoid/extPlugin/graspPlugin/GripperManipulation/rtc に、以下のファイルが作られる。

GraspControllerSVC_impl.cpp

GraspControllerSVC_impl.h

GripperManipulation.conf

GripperManipulation.cpp

GripperManipulation.h

GripperManipulationComp.cpp

GripperManipulationComp_vc8.vcproj

GripperManipulationComp_vc9.vcproj

GripperManipulation_vc8.sln

GripperManipulation_vc8.vcproj

GripperManipulation_vc9.sln

GripperManipulation_vc9.vcproj

Makefile.GripperManipulation

RTC.xml

copyprops.bat

user_config.vsprops

画面の指示にしたがって C++パースペクティブに切り替える。

このとき 一緒に保存されている RTC.xml に、一連の設定が保存されている。

RTコンポーネントの設定を変更する時は、RTC Builderの基本タブの中にある、プロファイルのインポート機能で、この設定を呼び戻すことができる。

rtc をコンパイルする

端末を開いて GripperManipulation/rtc ディレクトリに移って、以下のコマンドを実行する。

```
% omniidl -bcxx GraspController.idl
```

GraspController.hh と GraspControllerSK.cc が生成される。
続いて、make を実行する。Makefile を指定する必要がある。

```
% make -f Makefile.GripperManipulation
```

これによって、GraspControllerSkel.cpp, GraspControllerStub.cpp, GraspControllerSkel.h, GraspControllerStub.h が生成される。
（コンパイルは通らないが、そのまま進める）

Choreonoid プラグインとRTコンポーネントの橋渡しクラスを作成する

graspPlugin/GraspConsumer から、
GraspRtcController.h と GraspRtcController.cpp を GripperManipulation にコピーして、編集する。

ManipController.h(cpp) とまぎらわしいので、まずファイル名を ManipRtc.h(cpp) に変更する。

サンプルパッチ

サンプルとして、一連のファイル編集を一度に行うパッチファイルを作成した。

```
%  svn co -r131 http://subaru.ait.kyushu-u.ac.jp/svn/grasp-plugin/trunk/graspPlugin/GripperManipulation
```

として GripperManipulation のリビジョン131をチェックアウトしたあと、RTC Builderの設定を行い、ManipRtc.h(cpp) を作成したところで、
パッチファイル GripperManipulation_patch.txt をダウンロードし、

```
% patch -p0 < GripperManipulation_patch.txt
```

上のコマンドを実行すると、変更点が一度に適用されるので、ただちにビルドを行える。

以下は、パッチファイルの変更点の概要である。

ManipRtc.h(cpp)

クラス名を ManipRtc に変更する。(Eclipseのリファクタリング機能が便利だが、誤って元のGraspConsumer まで変更しないよう、注意！)

GraspConsumer.h をインクルードしている部分を、rtc下のGraspController.hh と GripperManipulation.hに変更する。

comp の型を、GraspConsumer *から、::GripperManipulation * に変更する。

graspPlanResult の引数で、const GraspPlanResult となっている部分をすべて::GraspPlanResult に変更する。

ついでに、インクルードガードの識別子をヘッダ名と合わせて MANIPRTC_H とする。

ManipPlugin.cpp

ManipPlugin::initialize メソッドに、RTコンポーネントをスタートさせるコードを追加する。

```
  virtual bool initialize() {
    //Robotics::SceneBodyManager* manager = Robotics::SceneBodyManager::instance();
      grasp::GraspController::instance(ManipController::instance());
      ManipRtc::instance()->RtcStart();
      //manage(manager->addSceneBodyFactory(create));
```

rtc/GripperManipulation.h

grasp::ManipRtc クラスを、GripperManipulation のフレンドクラスに設定する。

```
using namespace RTC;

namespace  grasp {
   class ManipRtc;
}

...

class GripperManipulation
 : public RTC::DataFlowComponentBase
{
public:
 friend class grasp::ManipRtc;
```

rtc/GraspControlerSVC_impl.cpp/h

RTCBuilderはIDLデータ型の引数をCORBAとして出力するが、omniidlは同じものを::CORBAとして出力するため、型が異なるとコンパイラがエラーを出してしまう。
そこで、omniidlの出力に合わせて、CORBAネームスペースの型の引数の宣言を、::CORBA に書き換える。
また、GraspPlanResultSVC_impl クラスのメソッドの引数について、GraspPlanStart ネームスペースの型の引数宣言を GraspPlanResult に書き換える。(これはRTCBuilder のバグか)

CMakeLists.txt

set(rtc-dir ./rtc) を追加

include_directories(${rtc-dir}) を追加

sources に ManipRtc.cpp, ${rtc-dir}/GraspControllerSVC_impl.cpp, ${rtc-dir}/GripperManipulation.cpp, ${rtc-dir}/GraspControllerSkel.cpp を追加。

headers に ManipRtc.h, ${rtc-dir}/GraspController.hh, ${rtc-dir}/GraspControllerSVC_impl.h, ${rtc-dir}/GripperManipulation.h, ${rtc-dir}/GraspControllerSkel.h, ${rtc-dir}/GraspControllerStub.h を追加。

add_definitions に -fPIC -O2 -I/usr/include -I/usr/include/rtm/idl -I. を追加。

target_link_libraries に uuid dl pthread omniORB4 omnithread omniDynamic4 RTC coil を追加。

if(UNIX)と、対になるelseif(MSVC)~endif(UNIX)までの行を削除。

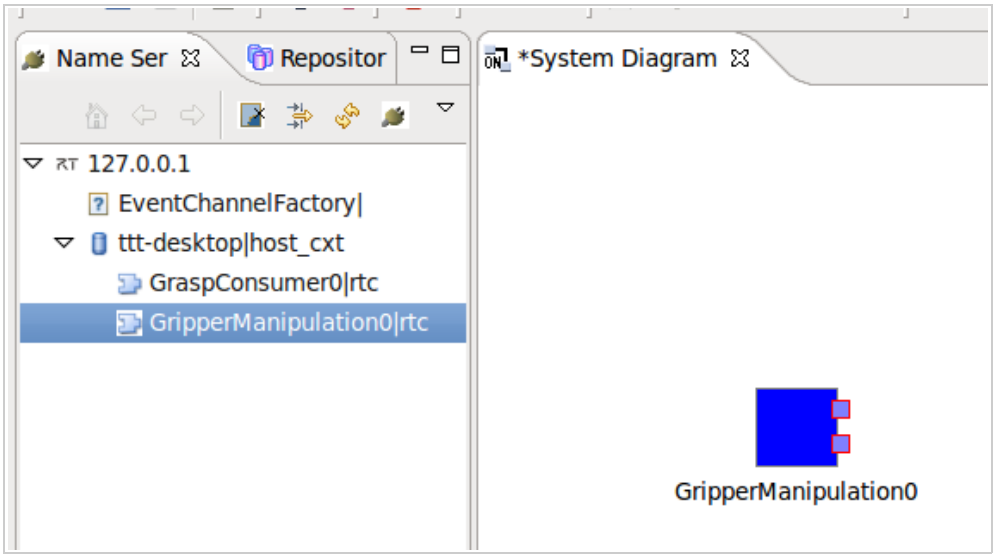
ビルドと動作確認

初めの一回は、Choreonoid ディレクトリで ccmake を実行する。

```
% ccmake .
```

GRASP_PLUGINS に GripperManipulation を追加して、configure と generate を実行すると、Makefile が作成される。
make して、lib/choreonoid-0.9/libCnoidGripperManipulationPlugin.so が無事にビルドされたら、動作確認を行う。

Choreonoid と RT System Editor を起動して、NameServer の127.0.0.1 の(ホスト名) | host_cxtの下に、GripperManipulation0|rtc があることを確認する。
これを System Diagram 上にドラッグ &ドロップできれば成功である。



（ただし、これを他のRTコンポーネントと結びつけても、まだ何も起こらない。GripperManipulation をRTコンポーネントに改造する・実装編につづく）

‹ /etc/hosts の設定　　↑ 上位　　GraspRTC プラグイン解説 ›

印刷用ページ　ログイン（登録）してコメントを投稿