

第2部

RTコンポーネント作成入門



第2部の目的

- RT System Editorを用いたRTCベースのシステム構築方法の習得 (RTC運用時に必要な知識)
- RTC Builderを用いたRTコンポーネントのひな形作成方法の習得 (RTC開発時に必要な知識)

OpenRTM-aistの開発支援ツール

- ロボット知能ソフトウェアプラットフォーム
- <http://www.openrtp.jp/wiki/>
- システム設計, シミュレーション, 動作生成, シナリオ生成などをサポート
- OpenRT Platformツール群
- コンポーネント開発, システム開発における各開発フェーズの作業支援
- 開発プラットフォームにEclipseを採用
- 構成
- RTCビルダ
- RTCデバッグ
- RTシステムエディタ
- ロボット設計支援ツール
- シミュレータ
- 動作設計ツール
- シナリオ作成ツール



The screenshot shows the OpenRT Platform Official Site with the following content:

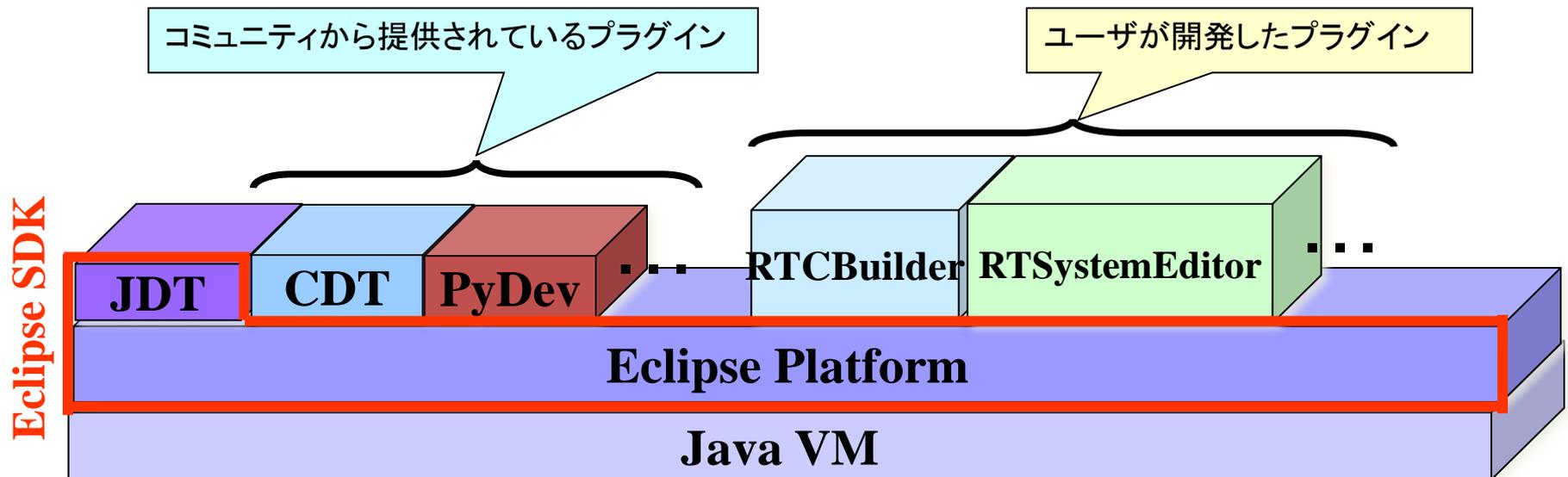
- OpenRT Platform オフィシャルサイト**
- Navigation: Home, Software, RTC
- Left sidebar menu:
 - ダウンロード (Downloads Open Source Software)
 - 知識化PJ (Project members only)
 - 産総研コンソ (Consortium members only)
 - 再利用WG (WG members only)
 - 問い合わせ先
 - Log in Sivika
- Main content:
 - ロボット知能ソフトウェアプラットフォーム**
次世代ロボットシステムの効率的かつ効果的な研究開発環境を構築するために、ロボット知能化技術をRTコンポーネントとしてモジュール化し、これらを統合して次世代ロボットシステムのシステム設計、シミュレーション、動作生成、シナリオ生成を行うことができるロボット知能ソフトウェアプラットフォームの研究開発を行っています。
 - 新着情報** (12 Next)
 - 23 Mar 2012 JDKのバージョンライセンス変更**
OracleによりJDKのライセンスが変更になっています。そのため、Ubuntuなどのディストリビューションでsun-java6などのパッケージ配布が中止になりました。
この変更に伴い、OpenHRP3.1では、そのままではパッケージインストールが不可能になっております。
もし、OpenHRP3.1をスクラッチから導入されたい場合には、開発チーム(openrtp@m.aist.go.jp)までご連絡をお願いします。
[Ubuntu10.04 LTSに対する暫定的な対応](#)
 - 23 Mar 2012 OpenRTM-aist-1.0.0 C言語実装**
OpenRTM-aist-1.0をC言語で実装中です。また、α版ですが、ドキュメントとソースコードを公開します。
<http://openrtp.jp/OpenRTM-C/index.html>
 - 25 Oct 2011 国際ロボット展 セミナー詳細のつか**
2011年11月11日に国際ロボット展で開催されるセミナーの時刻表と各セミナーの概要をアップしました。
また、Choreonoidに関するセミナーの配布資料もアップ致しましたので、参照して下さい。

セミナー概要:iRex2011
セミナー2:iRex2011Seminar2



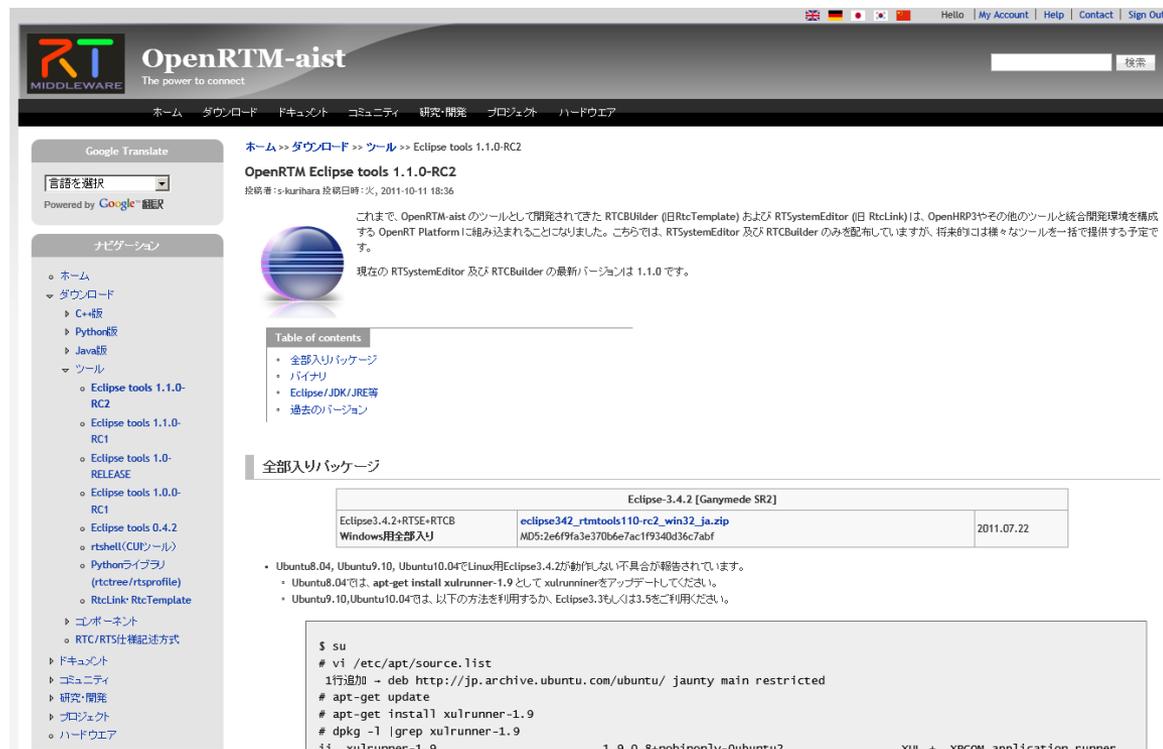
RTC BuilderとRT System Editor

- オープンソース・コミュニティで開発されている統合開発環境
- マルチプラットフォーム対応. WindowsやLinuxなど複数OS上で利用可能
- 「Plugin」形式を採用しており, 新たなツールの追加, 機能のカスタマイズが可能
- RCP(Rich Client Platform)を利用することで, 簡単に単独アプリ化が可能



RTC BuilderとRT System Editorの利用方法

- OpenRTM-aist-1.1.2のインストール時に、インストールされるので、そちらを利用。
「スタート」→「すべてのプログラム」→「OpenRTM-aist 1.1.2」→「Tools」にある、OpenRTPを選択。
もしくは、検索でOpenRTPと打ち込んで検索
- 単体でダウンロードする場合、Linuxの場合は、下記のサイトからダウンロードし、解凍。
(別途、JAVAの環境が必要になるので、注意。)



The screenshot shows the OpenRTM-aist website with the following content:

OpenRTM-aist
The power to connect

Home | My Account | Help | Contact | Sign Out

ホーム ダウンロード ドキュメント コミュニティ 研究・開発 プロジェクト ハードウェア

Google Translate
言語を選択
Powered by Google 翻訳

ナビゲーション

- ホーム
- ダウンロード
 - C++版
 - Python版
 - Java版
 - ツール
 - Eclipse tools 1.1.0-RC2
 - Eclipse tools 1.1.0-RC1
 - Eclipse tools 1.0-RELEASE
 - Eclipse tools 1.0.0-RC1
 - Eclipse tools 0.4.2
 - rtshell(CUPツール)
 - Pythonライブラリ (rtctree/rtprofile)
 - RtcLink- RtcTemplate
 - コンポーネント
 - RTC/RTS仕様記述方式
 - ドキュメント
 - コミュニティ
 - 研究・開発
 - プロジェクト
 - ハードウェア

ホーム >> ダウンロード >> ツール >> Eclipse tools 1.1.0-RC2

OpenRTM Eclipse tools 1.1.0-RC2
投稿者: s-kurihara 投稿日時: 火, 2011-10-11 18:36

これまで、OpenRTM-aistのツールとして開発されてきた RtcBuilder (旧RtcTemplate) および RTSystemEditor (旧 RtcLink)は、OpenHRP3やその他のツールと統合開発環境を構成する OpenRT Platform に組み込まれることになりました。こちらでは、RTSystemEditor 及び RtcBuilder のみを配布していますが、将来的には様々なツールを一括で提供する予定です。

現在の RTSystemEditor 及び RtcBuilder の最新バージョンは 1.1.0 です。

Table of contents

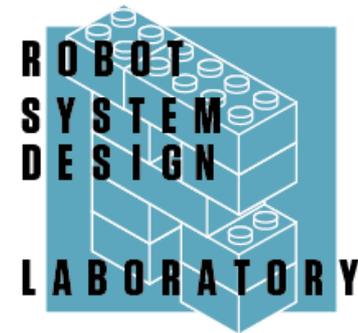
- 全部入りパッケージ
- バイナリ
- Eclipse/JDK/JRE等
- 過去のバージョン

全部入りパッケージ

Eclipse-3.4.2 [Ganymede SR2]		
Eclipse3.4.2+RTSE+RTCB Windows用全部入り	eclipse342_rtmtools110-rc2_win32_ja.zip MD5:2e6f9fa3e370b6e7ac1f9340d36c7abf	2011.07.22

- Ubuntu8.04, Ubuntu9.10, Ubuntu10.04でLinux用Eclipse3.4.2が動作しない不具合が報告されています。
- Ubuntu8.04では、`apt-get install xulrunner-1.9`としてxulrunnerをアップデートしてください。
- Ubuntu9.10, Ubuntu10.04では、以下の方法を利用するか、Eclipse3.3もしくは3.5をご利用ください。

```
$ su
# vi /etc/apt/source.list
1行追加 → deb http://jp.archive.ubuntu.com/ubuntu/ jaunty main restricted
# apt-get update
# apt-get install xulrunner-1.9
# dpkg -l |grep xulrunner-1.9
ii xulrunner-1.9
```



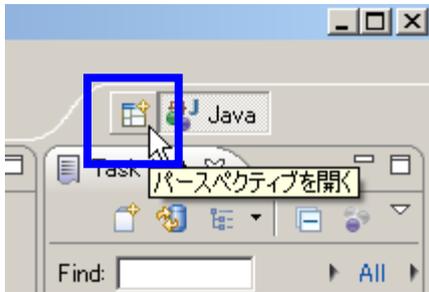
システム構築支援ツール RTSystemEditor



初期設定

■ パースペクティブの切り替え

- ① 画面右上の「パースペクティブを開く」を選択し、一覧から「その他」を選択



- ② 一覧画面から対象ツールを選択



※ パースペクティブ

Eclipse上でツールの構成を管理する単位
メニュー、ツールバー、エディタ、ビューなど
使用目的に応じて組み合わせる
独自の構成を登録することも可能

RT System Editorの概要

RTコンポーネントを組み合わせて、RTシステムを構築するためのツール。(Simlinkのようなイメージ。)

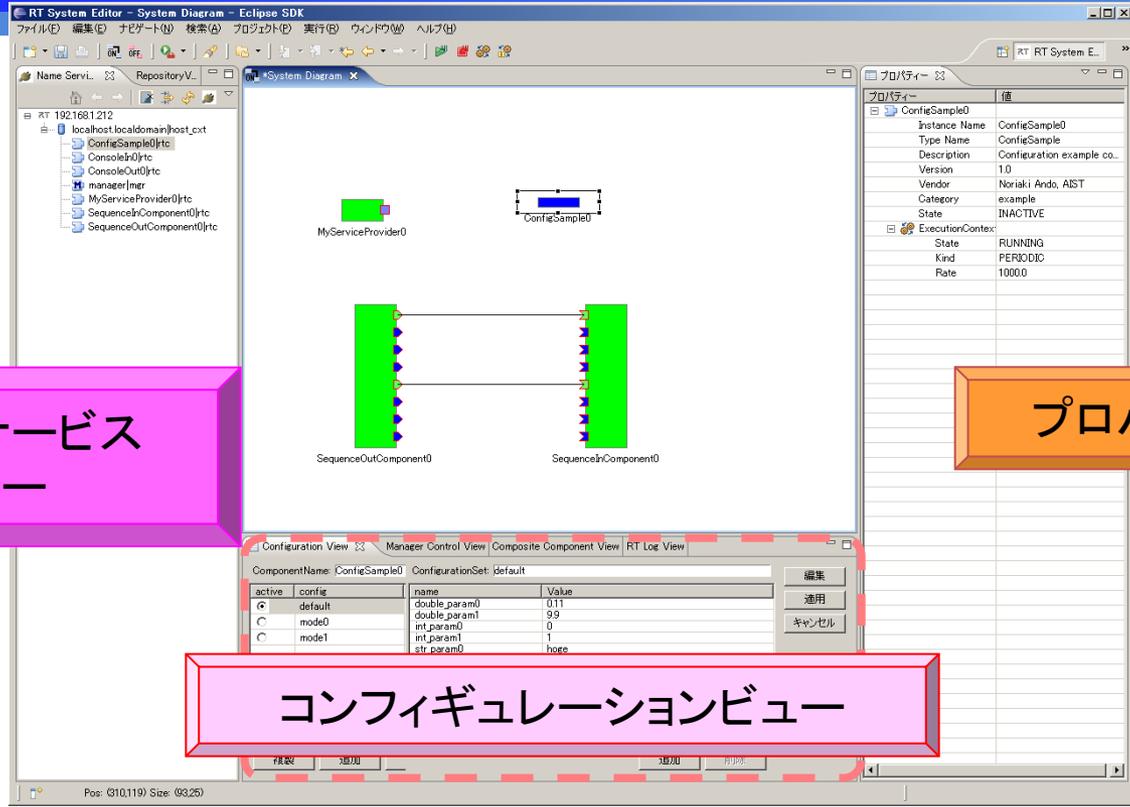
The screenshot displays the RT System Editor interface with the following components:

- System Diagram:** A central workspace showing components: `MyServiceProvider0` (green rectangle), `ConfigSample0` (blue rectangle), `SequenceOutComponent0` (green vertical bar with purple triangles), and `SequenceInComponent0` (green vertical bar with purple triangles). Lines connect `SequenceOutComponent0` to `SequenceInComponent0`.
- Repository View (Left):** A tree view showing the project structure under `localhost.localdomain.host_ext`, including `ConfigSample0|rtc`, `ConsoleIn0|rtc`, `ConsoleOut0|rtc`, `manager|lmer`, `MyServiceProvider0|rtc`, `SequenceInComponent0|rtc`, and `SequenceOutComponent0|rtc`.
- Properties View (Right):** A table showing properties for `ConfigSample0`.

プロパティ	値
Instance Name	ConfigSample0
Type Name	ConfigSample
Description	Configuration example co...
Version	1.0
Vendor	Noriaki Ando, AIST
Category	example
State	INACTIVE
ExecutionContext:	
State	RUNNING
Kind	PERIODIC
Rate	1000.0
- Configuration View (Bottom):** A table showing configuration sets for `ConfigSample0`.

active	config	name	Value
<input checked="" type="radio"/>	default	double_param0	0.11
<input type="radio"/>	mode0	double_param1	9.9
<input type="radio"/>	mode1	int_param0	0
		int_param1	1
		str_param0	hoee
		str_param1	dara
		vector_param0	00,1,02,03,04,0

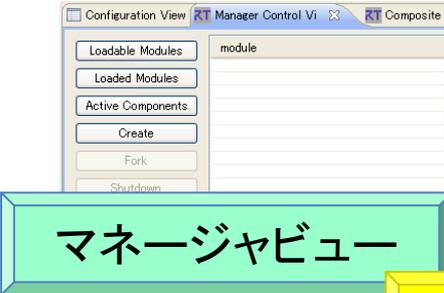
画面説明



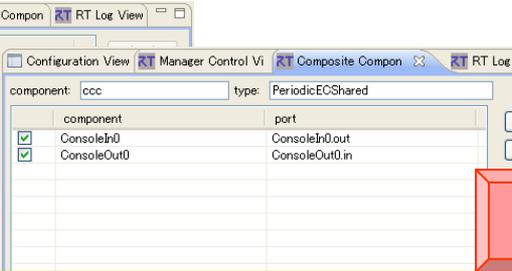
ネームサービス
ビュー

プロパティビュー

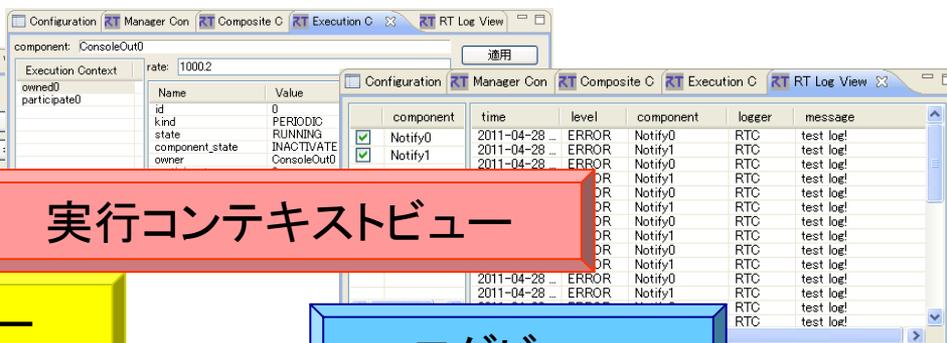
コンフィギュレーションビュー



マネージャビュー



複合コンポーネントビュー



実行コンテキストビュー

ログビュー

RT System構築演習

■ Naming Serviceの起動

■ [スタート]メニューから

[プログラム]→[OpenRTM-aist 1.1.2]→[tools]

→[Start C++ Naming Service]

■ ConsoleInCompの起動

■ [スタート]メニューから起動

[プログラム]→[OpenRTM-aist 1.1.2]→[C++]→[components]
→[Example]→[ConsoleInComp.exe]

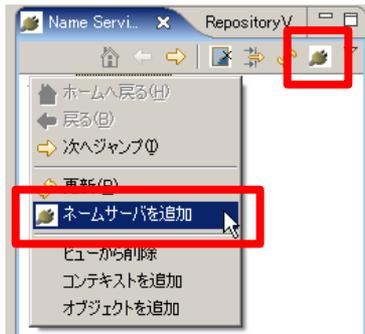
■ ConsoleOutCompの起動

■ [スタート]メニューから起動

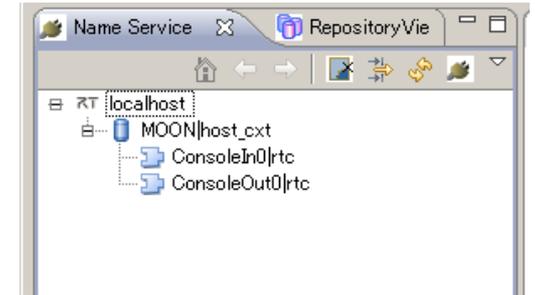
[プログラム]→[OpenRTM-aist 1.1.2]→[C++]→[components]
→ [Example]→[ComsoleOutComp.exe]

ネームサービスへの接続

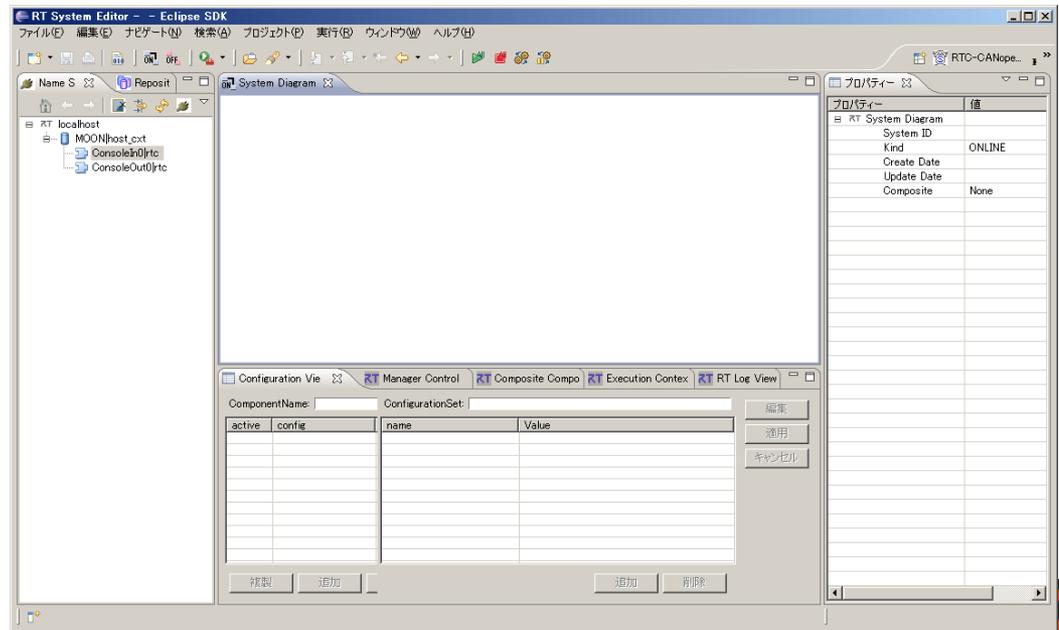
■ ネームサービスへ接続



※対象ネームサーバのアドレス, ポートを指定
→ポート省略時のポート番号は
設定画面にて設定可能

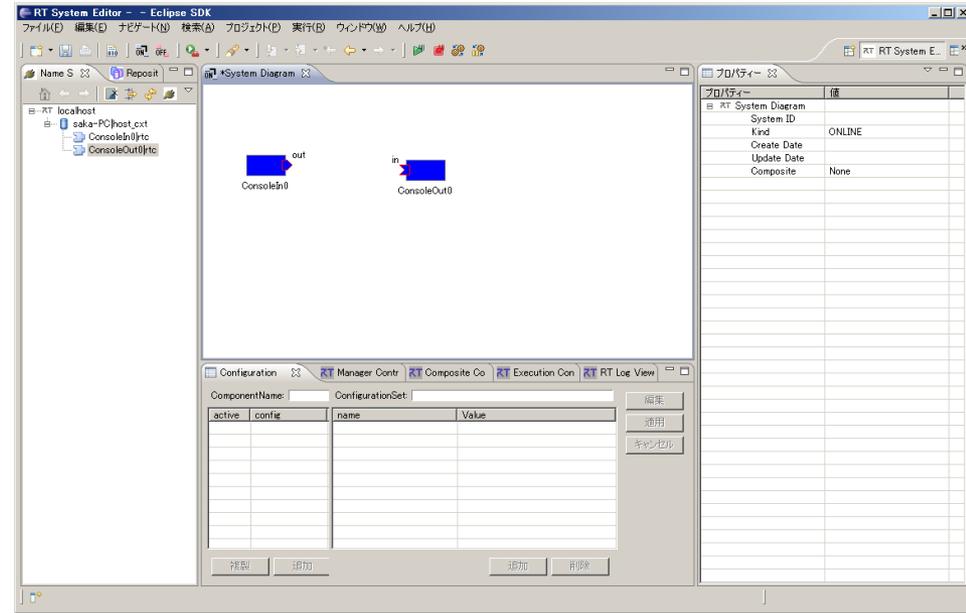
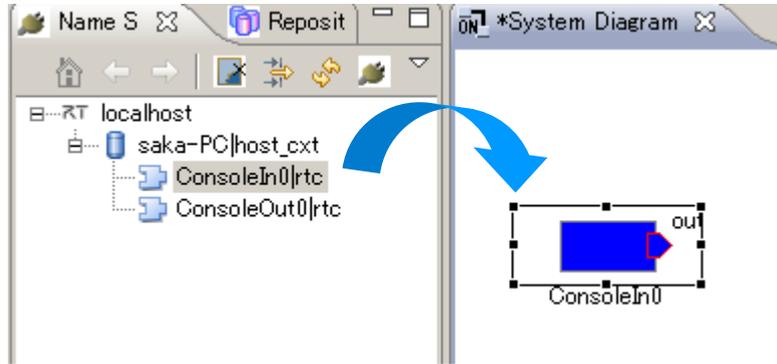


■ システムエディタの起動



RTコンポーネントの配置方法

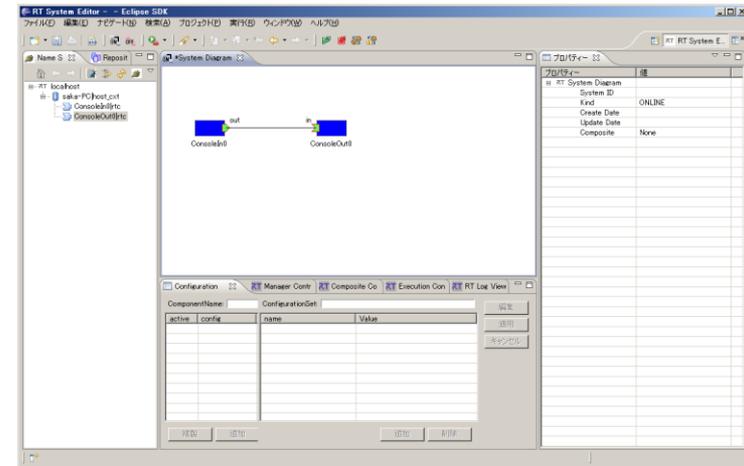
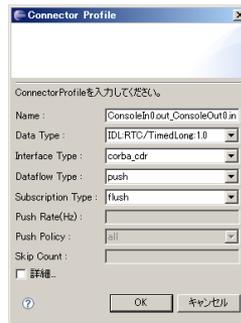
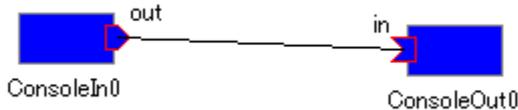
RTコンポーネントの配置



※ネームサービスビューから対象コンポーネントをドラッグアンドドロップ

ポートの接続

- ①接続元のポートから接続先のポートまでドラッグ
- ②接続プロファイルを入力



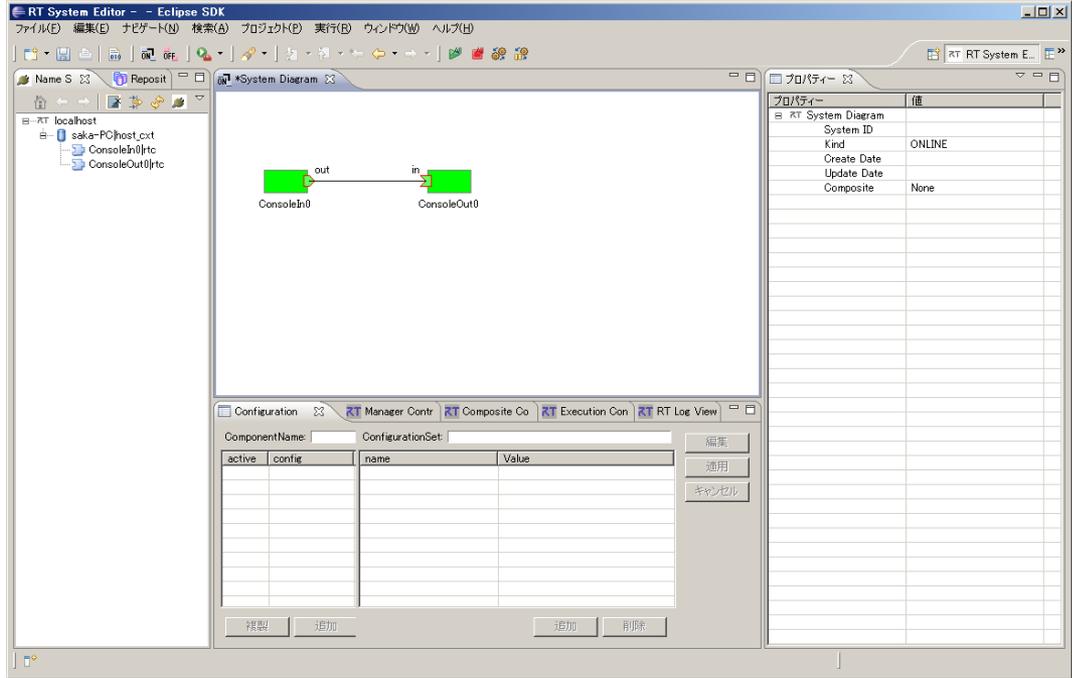
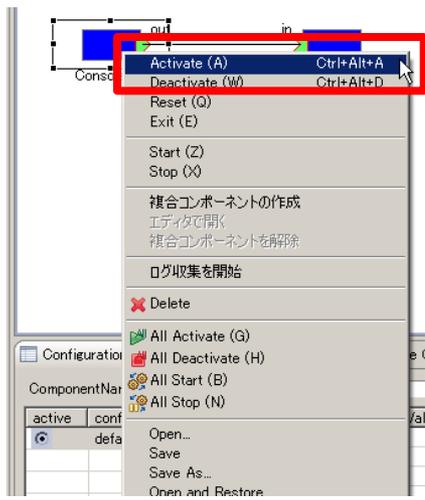
※ポートのプロパティが異なる場合など、接続不可能なポートの場合にはアイコンが変化



RTコンポーネントの起動

■ コンポーネントの起動

※各RTC単位で起動する場合



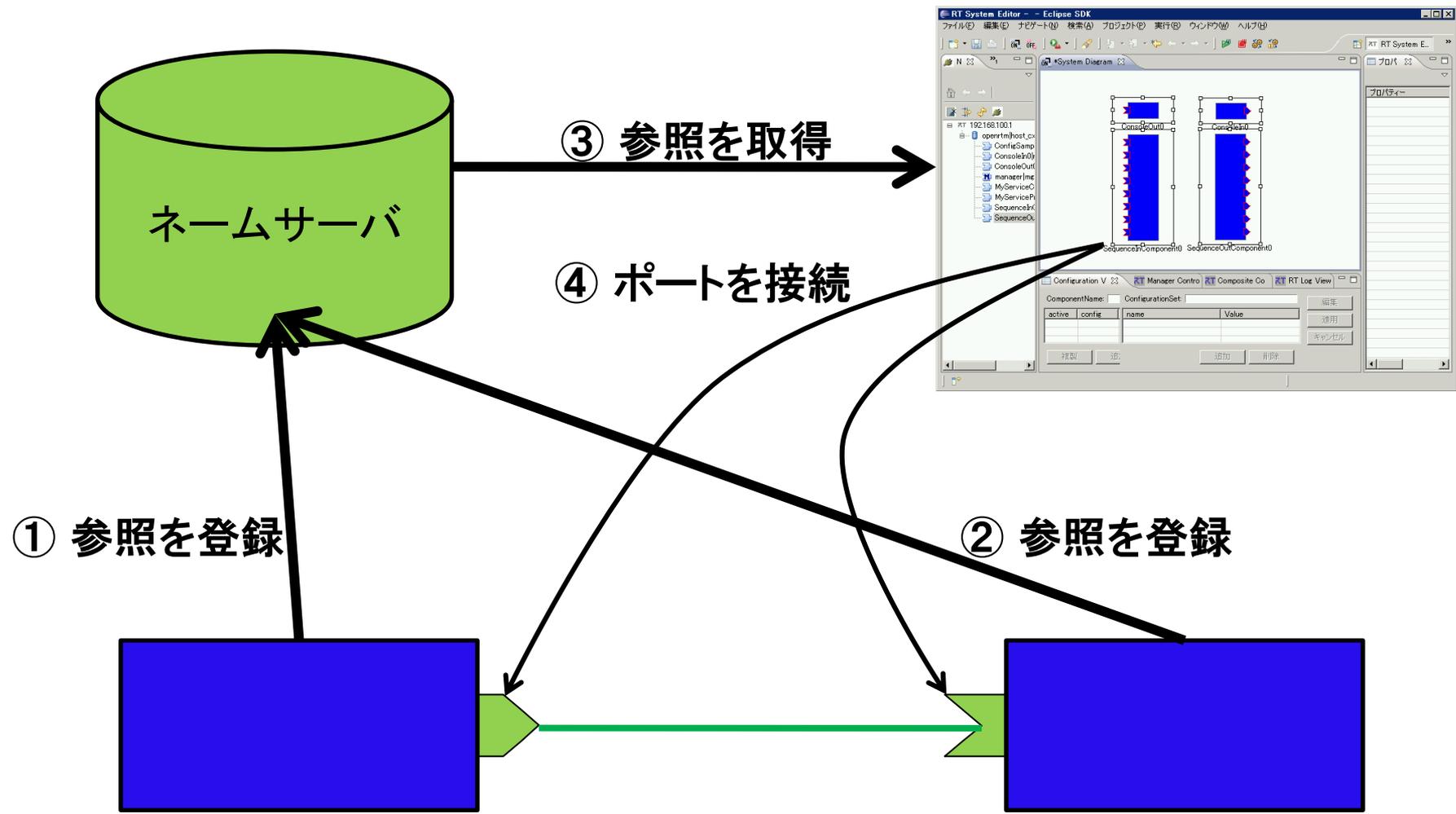
※全てのRTCを一括で起動する場合



※停止はDeactivateを実行

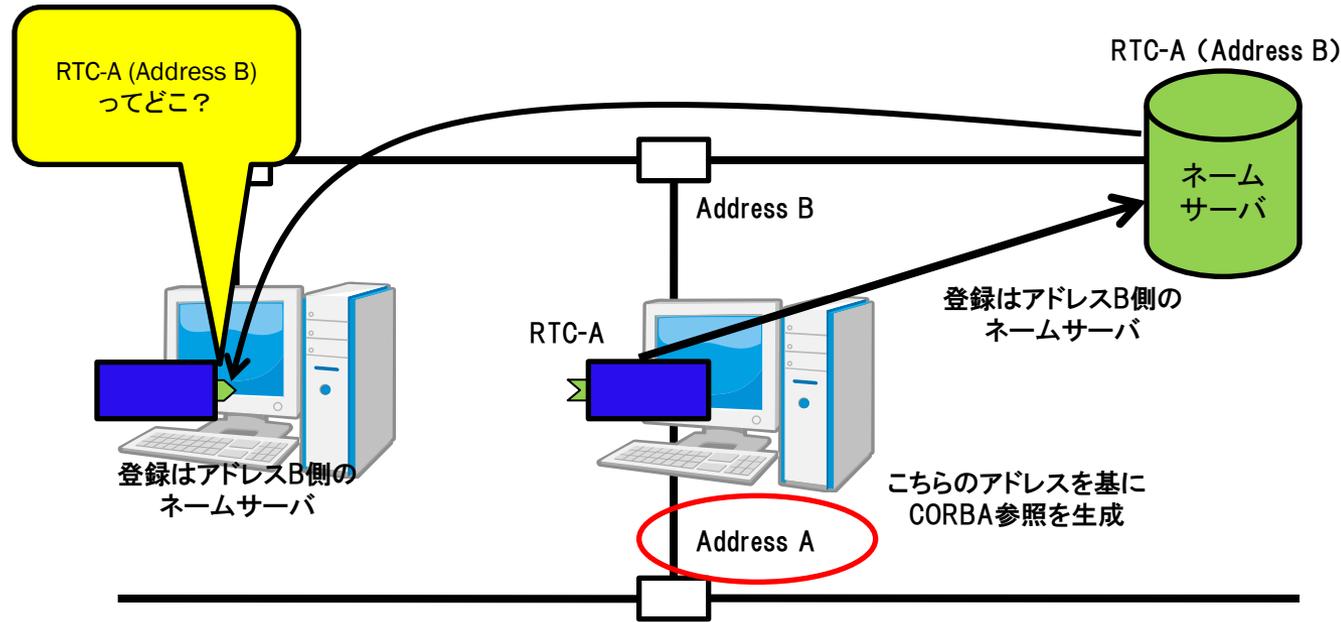
※RTC間の接続を切る場合には接続線をDeleteもしくは、右クリックメニューから「Delete」を選択

RTコンポーネントとネームサーバ



RTシステム構築時の注意点

■ ネットワークインターフェースが2つある場合



■ rtc.confについて

■ RTC起動時の登録先NamingServiceや、登録情報などについて記述

■ 記述例:

■ **corba.nameservers**: localhost:9876

■ **naming.formats**: SimpleComponent/%n.rtc

■ **corba.endpoints**:192.168.0.12:

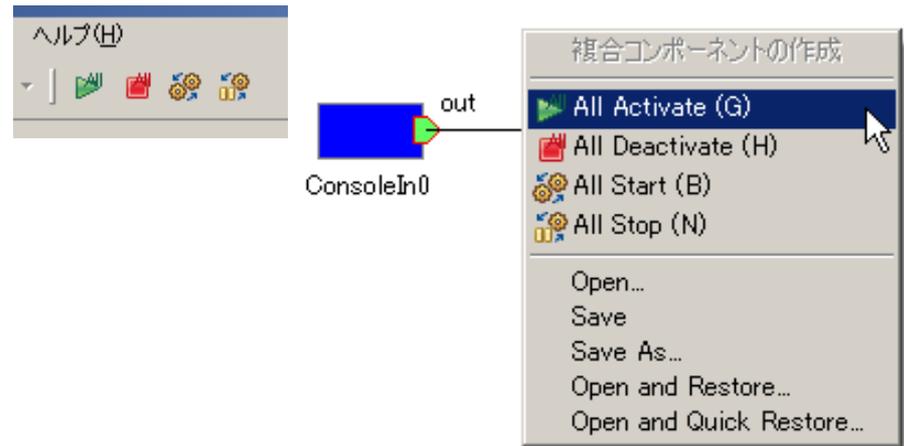
RTコンポーネントの動作

アクション名	説明
Activate	対象RTCを活性化する
Deactivate	対象RTCを非活性化する
Reset	対象RTCをエラー状態からリセットする
Exit	対象RTCの実行主体(ExecutionContext)を停止し, 終了する
Start	実行主体(ExecutionContext)の動作を開始する
Stop	実行主体(ExecutionContext)の動作を停止する

■ 各コンポーネント単位での動作変更



■ 全コンポーネントの動作を一括変更



※ポップアップメニュー中でのキーバインドを追加

※単独RTCのActivate/Deactivateについては, グローバルはショートカットキー定義を追加

接続プロファイル(DataPort)について

項目	設定内容
Name	接続の名称
Data Type	ポート間で送受信するデータの型. ex)TimedOctet, TimedShortなど
Interface Type	データを送受信するポートの型. ex)corba_cdrなど
Data Flow Type	データの送信方法. ex)push, pullなど
Subscription Type	データ送信タイミング. 送信方法がPushの場合有効 . New, Periodic, Flushから選択
Push Rate	データ送信周期(単位:Hz). SubscriptionTypeがPeriodicの場合のみ有効
Push Policy	データ送信ポリシー. SubscriptionTypeがNew, Periodicの場合のみ有効 . all, fifo, skip, newから選択
Skip Count	送信データスキップ数. Push PolicyがSkipの場合のみ有効

■ SubscriptionType

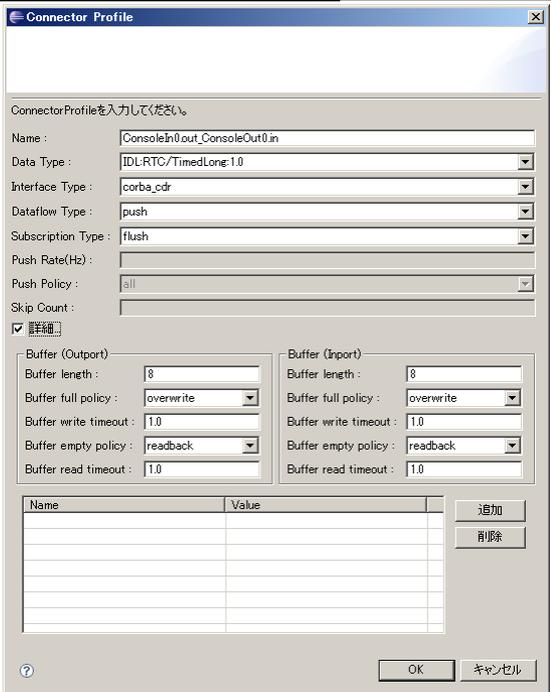
- New : バッファ内に新規データが格納されたタイミングで送信
- Periodic : 一定周期で定期的にデータを送信
- Flush : バッファを介さず即座に同期的に送信

■ Push Policy

- all : バッファ内のデータを一括送信
- fifo : バッファ内のデータをFIFOで1個ずつ送信
- skip : バッファ内のデータを間引いて送信
- new : バッファ内のデータの最新値を送信(古い値は捨てられる)

接続プロファイル(DataPort)について

項目	設定内容
Buffer length	バッファの大きさ
Buffer full policy	データ書き込み時に, バッファフルだった場合の処理. overwrite, do_nothing, blockから選択
Buffer write timeout	データ書き込み時に, タイムアウトイベントを発生させるまでの時間(単位:秒)
Buffer empty policy	データ読み出し時に, バッファが空だった場合の処理. readback, do_nothing, blockから選択
Buffer read timeout	データ読み出し時に, タイムアウトイベントを発生させるまでの時間(単位:秒)



- ※OutPort側のバッファ, InPort側のバッファそれぞれに設定可能
- ※timeoutとして「0.0」を設定した場合は, タイムアウトしない

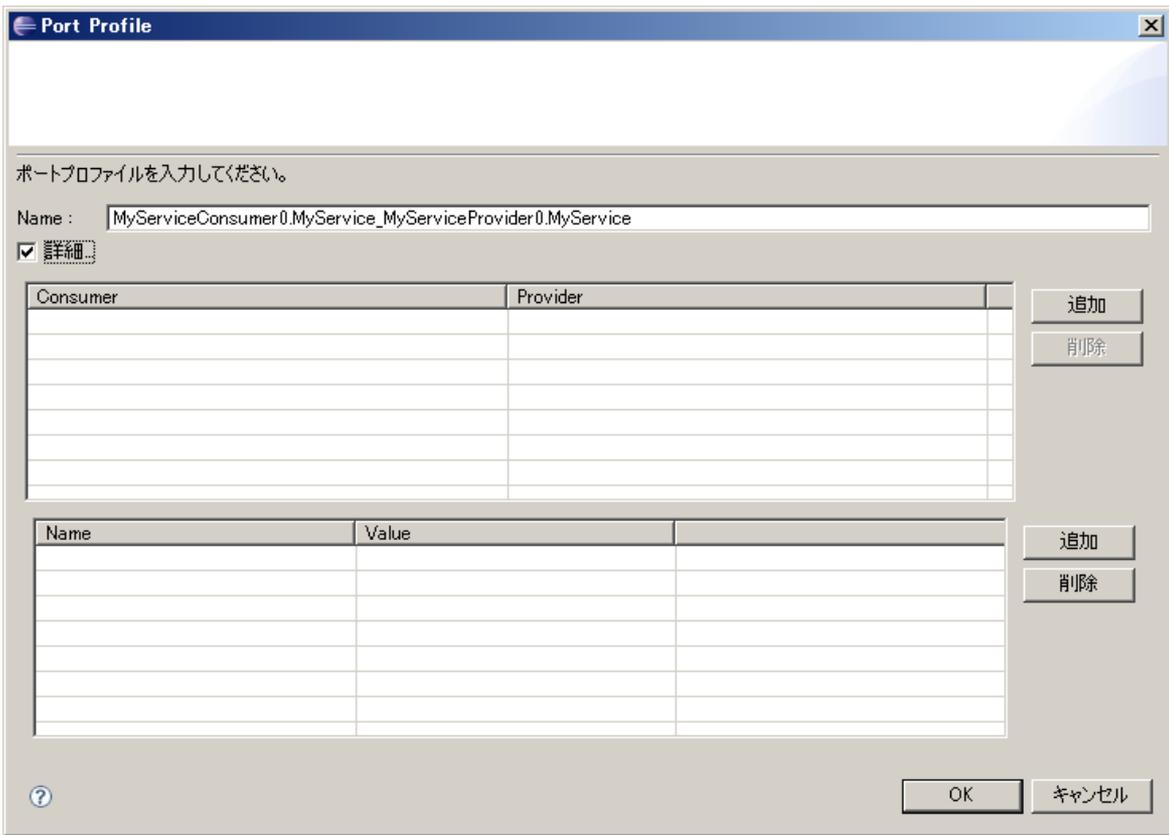
■ Buffer Policy

- overwrite : 上書き
- readback : 最後の要素を再読み出し
- block : ブロック
- do_nothing : なんもしない

※Buffer Policy = Block+timeout時間の指定で, 一定時間後読み出し/書き込み不可能な場合にタイムアウトを発生させる処理となる

接続プロファイル(ServicePort)について

項目	設定内容
Name	接続の名称
インターフェース情報	接続するインターフェースを設定. 接続対象のServicePortに複数のServiceInterfaceが定義されていた場合, どのインターフェースを実際に接続するかを指定



Port Profile

ポートプロファイルを入力してください。

Name : MyServiceConsumer0.MyService_MyServiceProvider0.MyService

詳細

Consumer	Provider

追加
削除

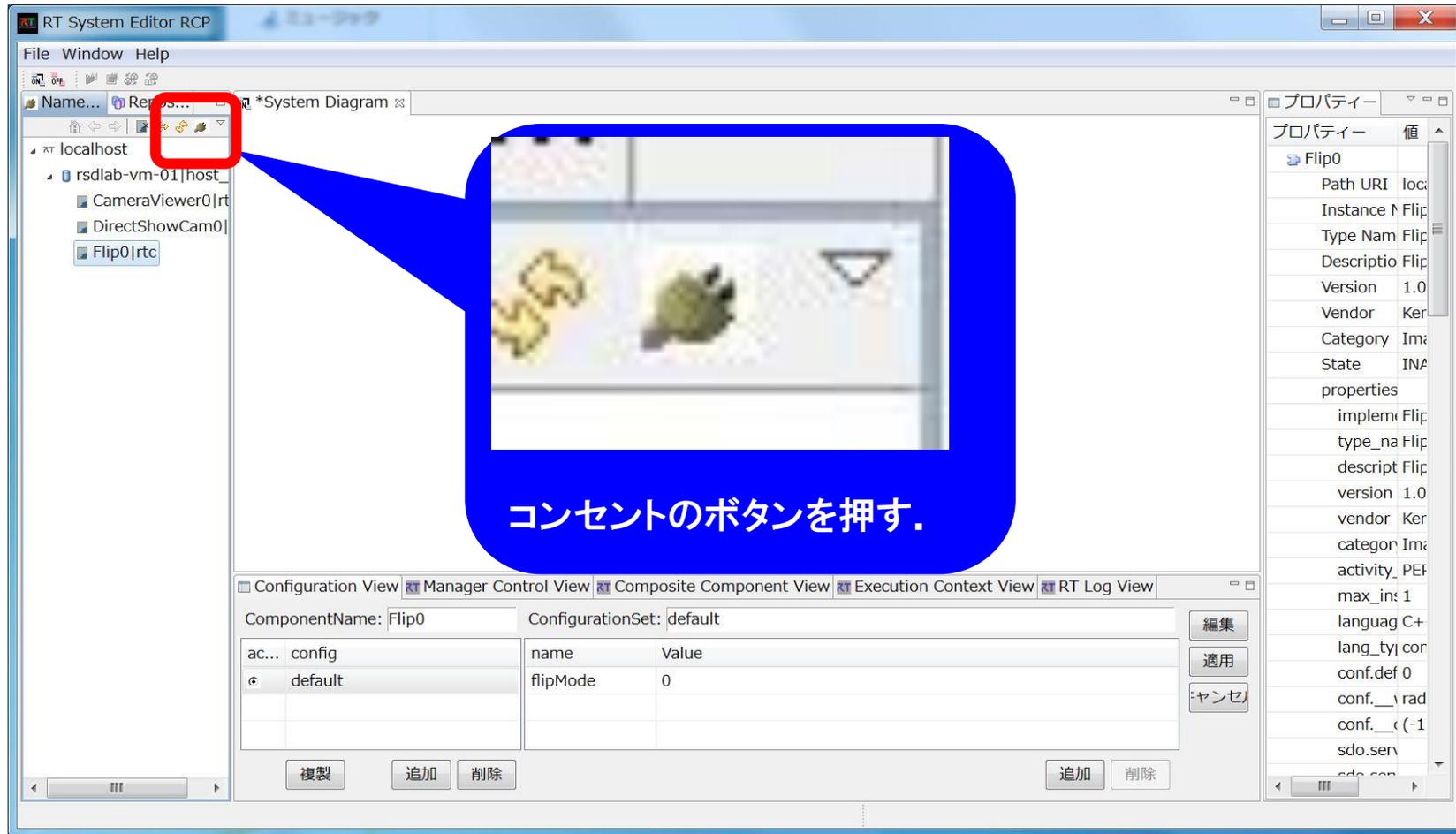
Name	Value

追加
削除

? OK キャンセル

ネットワーク上のPC間でのシステム構築

アクセス可能なネットワーク上に存在する別のPCで動作するネーミングサービスにアクセス



The screenshot shows the RT System Editor RCP interface. On the left, a tree view shows a project structure under 'localhost' with a sub-project 'rsdlab-vm-01|host' containing components like 'CameraViewer0|rtc', 'DirectShowCam0|rtc', and 'Flip0|rtc'. The main workspace displays a system diagram with a 'Flip0' component highlighted. A red box highlights a button in the diagram area, and a blue callout box points to it with the text 'コンセントのボタンを押す.' (Press the consent button.)

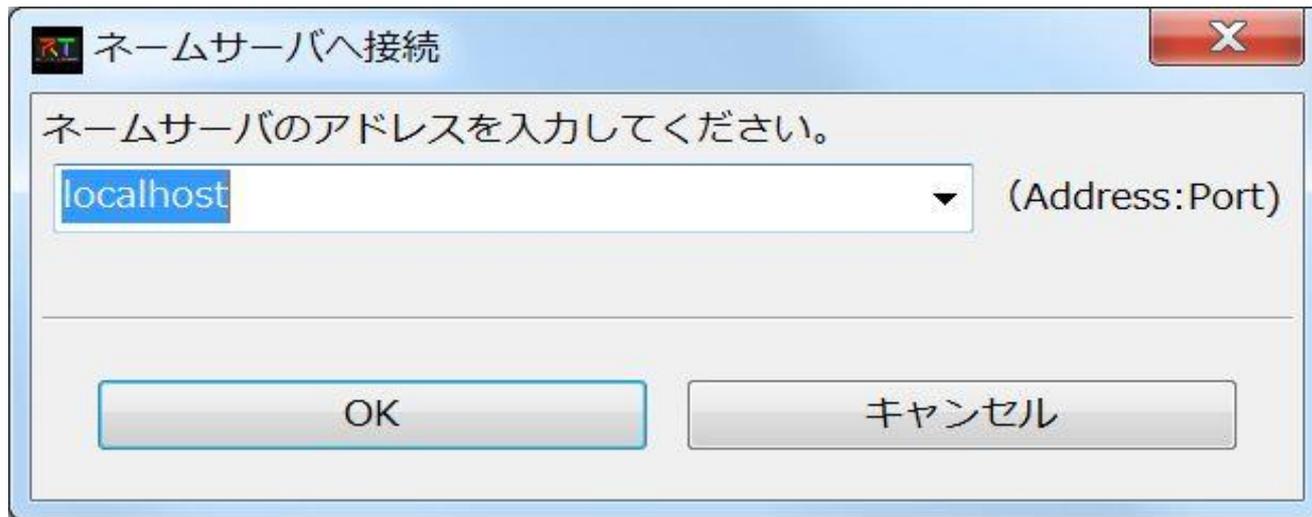
At the bottom, the Configuration View for 'Flip0' is shown with the following configuration:

ComponentName:	Flip0	ConfigurationSet:	default
ac...	config	name	Value
☺	default	flipMode	0

On the right, the Properties View shows various properties for 'Flip0', including Path URI, Instance Name, Type Name, Description, Version (1.0), Vendor (Kernel), Category (Image), and State (Inactive).

ネットワーク上のPC間でのシステム構築

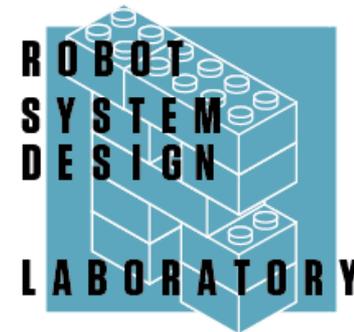
参照したいネーミングサービスが起動しているPCのIPアドレスとポートを入力する。



IPアドレスの確認方法

コマンドプロンプトにおいて、「ipconfig」と入力する。

他のPCで起動しているコンポーネントの閲覧およびRTCの遠隔利用ができる！
(ファイヤーウォールがある場合は見えません(利用できない))

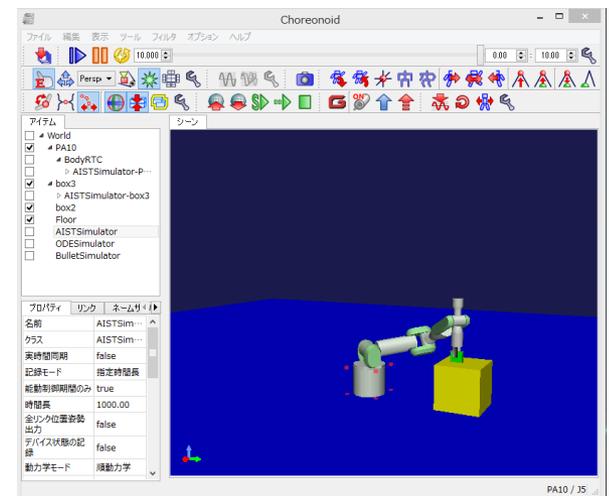
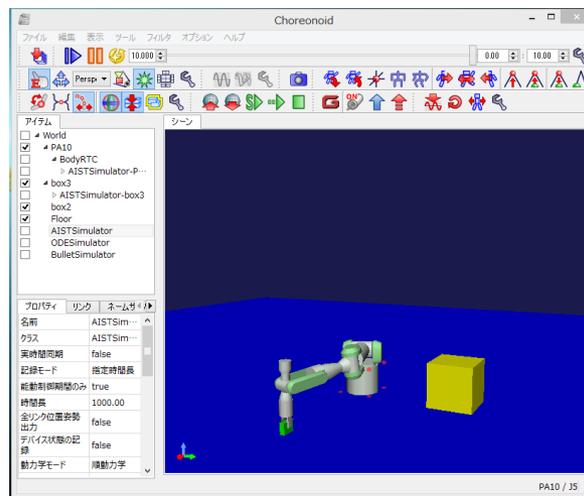
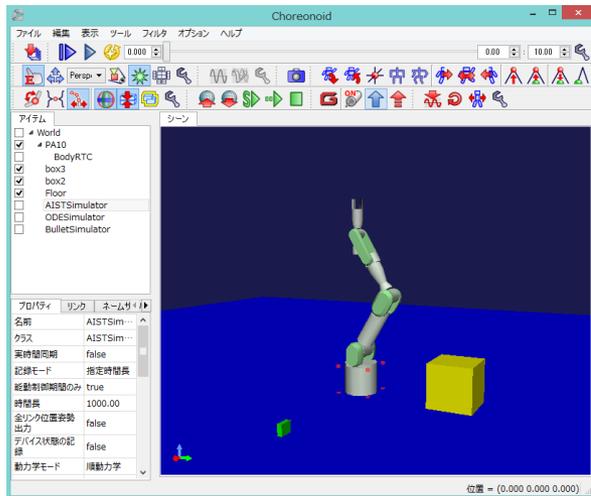


コンポーネント開発ツール RTC Builderについて



想定するコンポーネントのモデル

- ロボットアームの制御用RTコンポーネント
 - 手先位置を任意の位置に移動するためのRTC
 - アーム先端に取り付けられたグリッパの開閉が可能なRTコンポーネント



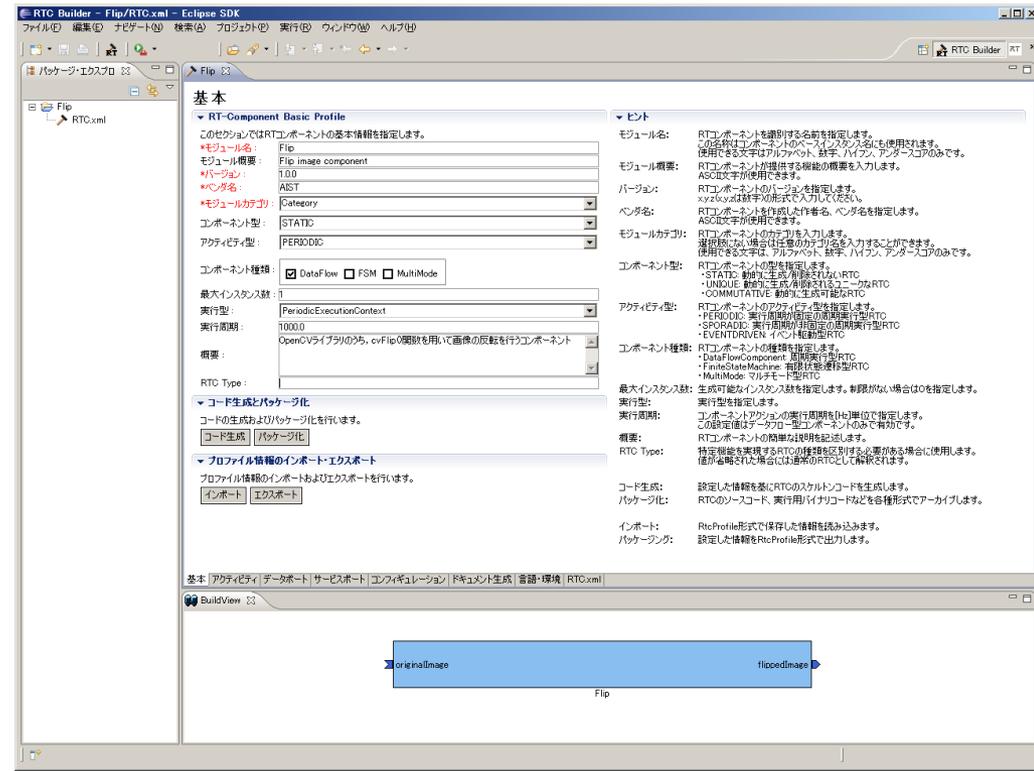
RTCBuilderの概要

■ RTCBuilderとは？

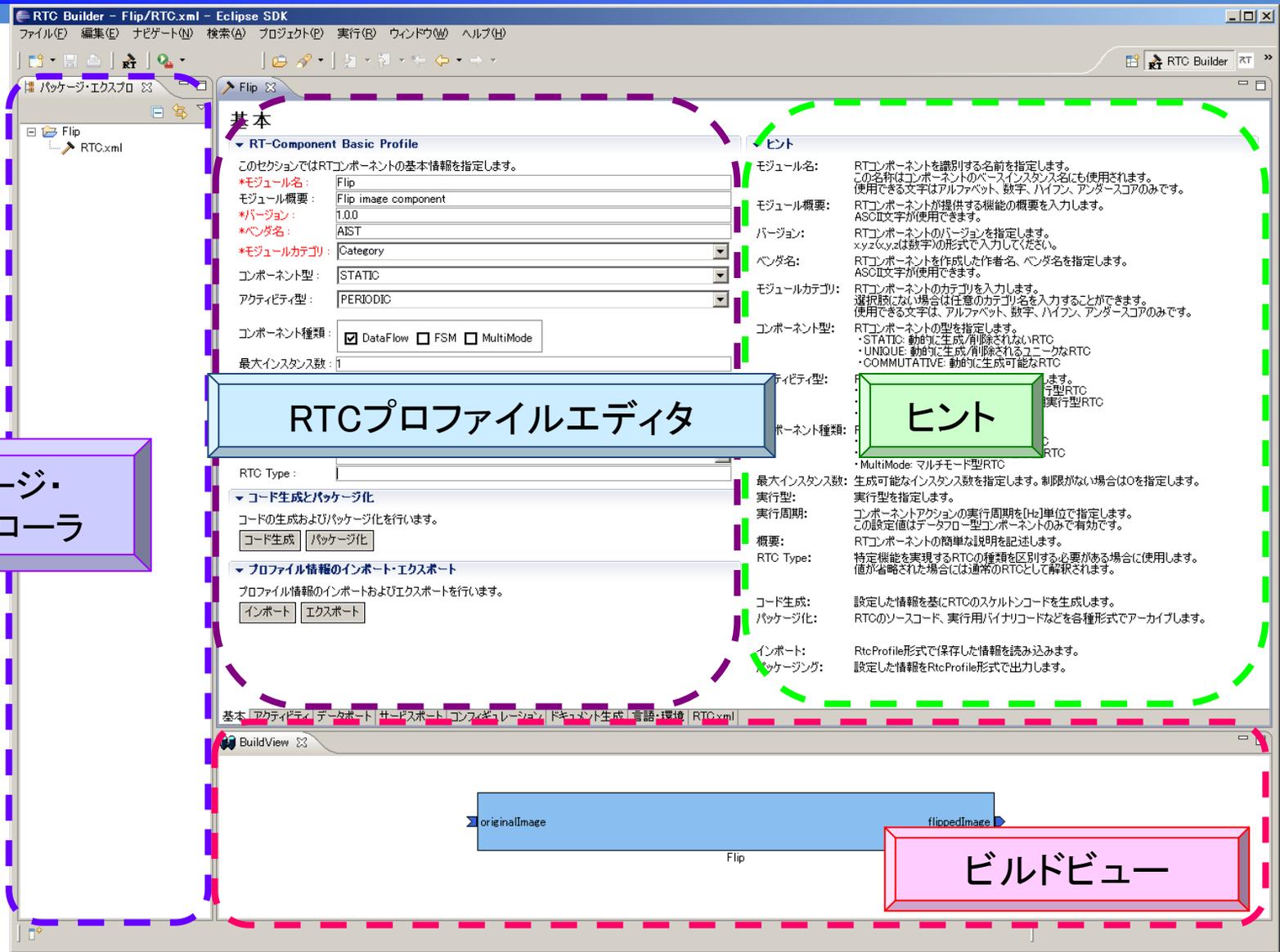
- コンポーネントのプロファイル情報を入力し、ソースコード等の雛形を生成するツール
- 開発言語用プラグインを追加することにより、各言語向けRTCの雛形を生成することが可能

- C++
- Java
- Python

- ※ C++用コード生成機能はRTCBuilder本体に含まれています。
- ※ その他の言語用コード生成機能は追加プラグインとして提供されています



RTCBuilderの外観



The screenshot shows the RTC Builder interface within Eclipse SDK. The main window is titled "RTC Builder - Flip/RTC.xml - Eclipse SDK".

基本 (Basic Profile):

- モジュール名: Flip
- モジュール概要: Flip image component
- バージョン: 1.0.0
- ベンダ名: AIST
- モジュールカテゴリ: Category
- コンポーネント型: STATIC
- アクティビティ型: PERIODIC
- コンポーネント種類: DataFlow FSM MultiMode
- 最大インスタンス数: 1

ヒント (Hints):

- モジュール名: RTコンポーネントを識別する名前を指定します。この名称はコンポーネントのベースインスタンス名にも使用されます。使用できる文字はアルファベット、数字、ハイフン、アンダースコアのみです。
- モジュール概要: RTコンポーネントが提供する機能の概要を入力します。ASCII文字が使用できます。
- バージョン: RTコンポーネントのバージョンを指定します。xyz.y.zは数字の形式で入力してください。
- ベンダ名: RTコンポーネントを作成した作者名、ベンダ名を指定します。ASCII文字が使用できます。
- モジュールカテゴリ: RTコンポーネントのカテゴリを入力します。選択されない場合は任意のカテゴリ名を入力することができます。使用できる文字は、アルファベット、数字、ハイフン、アンダースコアのみです。
- コンポーネント型: RTコンポーネントの型を指定します。
 - STATIC: 動的に生成/削除されないRTC
 - UNIQUE: 動的に生成/削除されるユニークなRTC
 - COMMUTATIVE: 動的に生成可能なRTC
- コンポーネント種類:
 - MultiMode: マルチモード型RTC
- 最大インスタンス数: 生成可能なインスタンス数を指定します。制限がない場合は0を指定します。実行型を指定します。
- 実行周期: コンポーネントアクションの実行周期を[Hz]単位で指定します。この設定値はデータフロー型コンポーネントのみで有効です。
- 概要: RTコンポーネントの簡単な説明を記述します。
- RTC Type: 特定機能を実現するRTCの種類を区別する必要がある場合に使用します。値が省略された場合には通常のRTCとして解釈されます。
- コード生成: 設定した情報を基にRTCのスケルトンコードを生成します。
- パッケージ化: RTCのソースコード、実行用バイナリコードなどを各種形式でアーカイブします。
- インポート: RtcProfile形式で保存した情報を読み込みます。
- パッケージング: 設定した情報をRtcProfile形式で出力します。

パッケージ・エクスプローラ (Package Explorer): Shows the project structure with "Flip" and "RTC.xml".

ビルドビュー (Build View): Shows the build process with "originalImage" and "flippedImage" components.

パッケージ・エクスプローラ

RTCプロファイルエディタ

ヒント

ビルドビュー

RTCBuilderの起動

- Windowsの場合
 - Eclipse.exeをダブルクリック
- Unix系の場合
 - ターミナルを利用してコマンドラインから起動
 - Ex) \$ /usr/local/Eclipse/eclipse

■ ワークスペースの選択(初回起動時)



■ ワークスペースの切替(通常時)



※ワークスペース
 Eclipseで開発を行う際の作業領域
 Eclipse上でプロジェクトやファイルを作成するとワークスペースとして指定したディレクトリ以下に実際のディレクトリ、ファイルを作成する

初回起動の場合

- 初期画面のクローズ
 - 初回起動時のみ

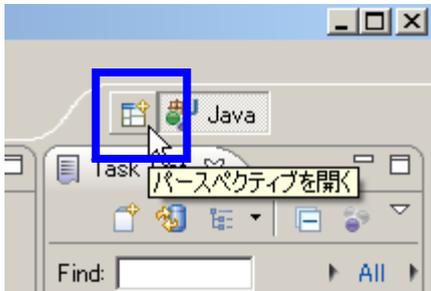


※パースペクティブ

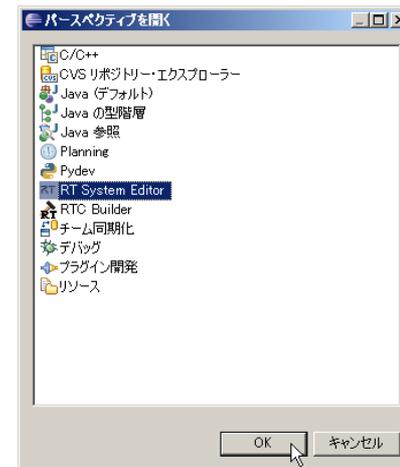
Eclipse上でツールの構成を管理する単位メニュー、ツールバー、エディタ、ビューなど使用目的に応じて組み合わせる独自の構成を登録することも可能

■ パースペクティブの切り替え

①画面右上の「パースペクティブを開く」を選択し、一覧から「その他」を選択



②一覧画面から対象ツールを選択



プロジェクト作成/エディタ起動

① ツールバー内のアイコンをクリック



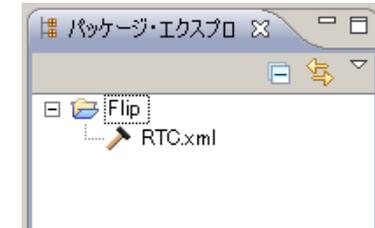
② 「プロジェクト名」欄に入力し, 「終了」



※メニューから「ファイル」-「新規」-「プロジェクト」を選択

【新規プロジェクト】画面にて「その他」-「RtcBuilder」を選択し, 「次へ」

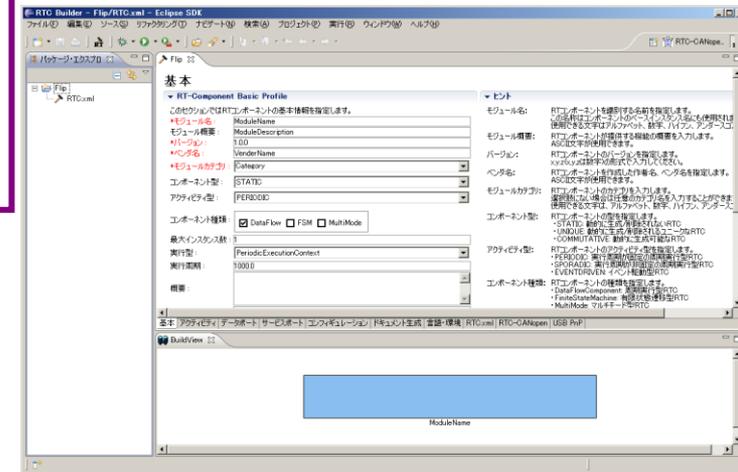
※メニューから「ファイル」-「Open New Builder Editor」を選択



※任意の場所にプロジェクトを作成したい場合

②にて「デフォルト・ロケーションの使用」チェックボックスを外す
「参照」ボタンにて対象ディレクトリを選択

→物理的にはワークスペース以外の場所に作成される 論理的にはワークスペース配下に紐付けされる



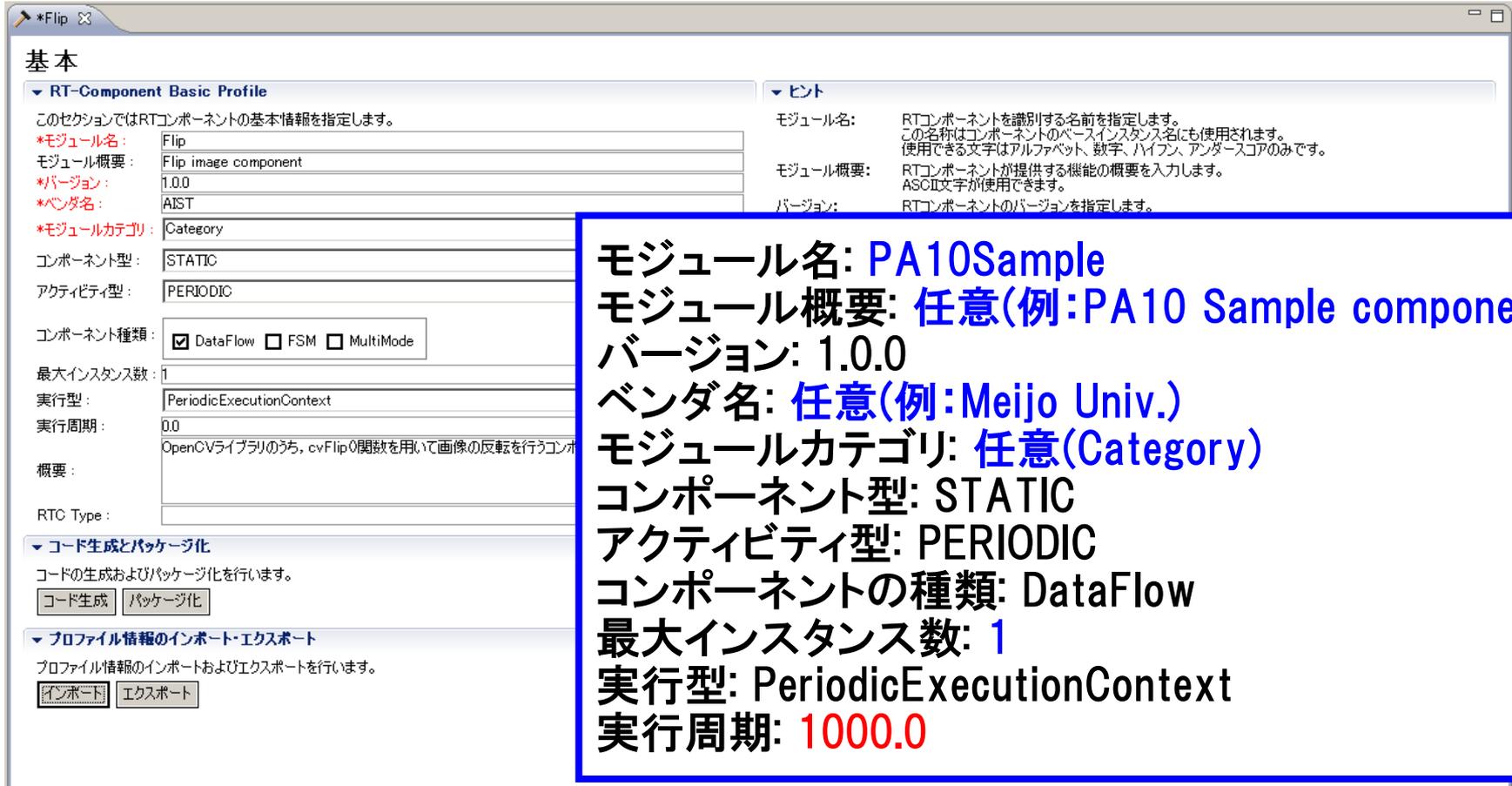
プロジェクト名: PA10Sample

RTCプロフィールエディタ

画面要素名	説明
基本プロフィール	RTコンポーネントのプロファイル情報など、コンポーネントの基本情報を設定. コード生成, インポート/エクスポート, パッケージング処理を実行
アクティビティ・プロファイル	RTコンポーネントがサポートしているアクティビティ情報を設定
データポート・プロファイル	RTコンポーネントに付属するデータポートに関する情報を設定
サービスポート・プロファイル	RTコンポーネントに付属するサービスポートおよび各サービスポートに付属するサービスインターフェースに関する情報を設定
コンフィギュレーション	RTコンポーネントに設定するユーザ定義のコンフィギュレーション・パラメータセット情報およびシステムのコンフィギュレーション情報を設定
ドキュメント生成	生成したコードに追加する各種ドキュメント情報を設定
言語・環境	生成対象コードの選択やOSなどの実行環境に関する情報を設定
RTC.xml	設定した情報を基に生成したRTC仕様(RtcProfile)を表示

基本プロフィール

■ RTコンポーネントの名称など，基本的な情報を設定



基本

▼ RT-Component Basic Profile

このセクションではRTコンポーネントの基本情報を指定します。

***モジュール名:** Flip
 モジュール概要: Flip image component
 ***バージョン:** 1.0.0
 ***ベンダ名:** AIST
 ***モジュールカテゴリ:** Category
 コンポーネント型: STATIC
 アクティビティ型: PERIODIC

コンポーネント種類: DataFlow FSM MultiMode

最大インスタンス数: 1
 実行型: PeriodicExecutionContext
 実行周期: 0.0
 概要: OpenCVライブラリのうち、cvFlip関数を用いて画像の反転を行うコンポーネント

RTC Type:

▼ コード生成とパッケージ化

コードの生成およびパッケージ化を行います。

▼ プロファイル情報のインポート・エクスポート

プロファイル情報のインポートおよびエクスポートを行います。

▼ ヒント

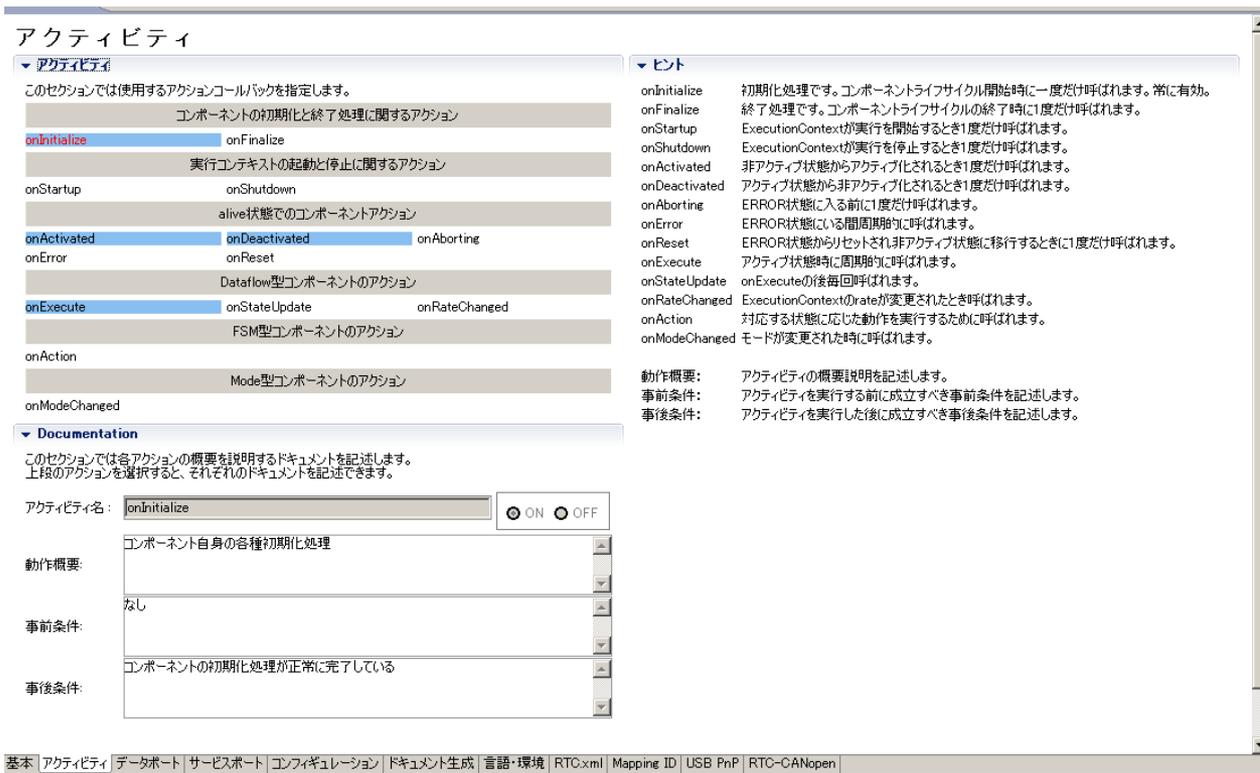
モジュール名: RTコンポーネントを識別する名前を指定します。この名称はコンポーネントのベースインスタンス名にも使用されます。使用できる文字はアルファベット、数字、ハイフン、アンダースコアのみです。
 モジュール概要: RTコンポーネントが提供する機能の概要を入力します。ASCII文字が使用できます。
 バージョン: RTコンポーネントのバージョンを指定します。

モジュール名: PA10Sample
モジュール概要: 任意(例:PA10 Sample component)
バージョン: 1.0.0
ベンダ名: 任意(例:Meijo Univ.)
モジュールカテゴリ: 任意(Category)
コンポーネント型: STATIC
アクティビティ型: PERIODIC
コンポーネントの種類: DataFlow
最大インスタンス数: 1
実行型: PeriodicExecutionContext
実行周期: 1000.0

- ※エディタ内の項目名が赤字の要素は必須入力項目
- ※画面右側は各入力項目に関する説明

アクティビティ

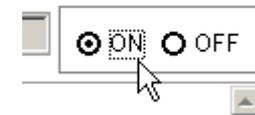
■ 生成対象RTCで実装予定のアクティビティを設定



① 設定対象のアクティビティを選択



② 使用/未使用を設定



以下をチェック：
 onActivated
 onDeactivated
 onExecute

※現在選択中のアクティビティは、一覧画面にて赤字で表示

※使用(ON)が選択されているアクティビティは、一覧画面にて背景を水色で表示

※各アクティビティには、「動作概要」「事前条件」「事後条件」を記述可能
 →記述した各種コメントは、生成コード内にDoxygen形式で追加される

データポート

■ 生成対象RTCに付加するDataPortの情報を設定

データポート

DataPortプロフィール

このセクションではRTCコンポーネントのDataPort(データポート)の情報を設定します。

*ポート名 (InPort)	*ポート名 (OutPort)
originalImage	flippedImage

Buttons: Add, Delete

Detail

このセクションではデータポート毎の概要を説明するドキュメントを記述します。上のデータポートを選択すると、それぞれのドキュメントが記述できます。

ポート名: originalImage (InPort)

*データ型: RTC:CameraImage

変数名: originalImage

表示位置: LEFT

Documentation: キャプチャされた画像データ

概要説明:

データ型: CameraImage型OpenRTM-aistのInterfaceDataTypes.idlにて定義されているデータ型

データ数: 任意

意味: 反転処理の対象となる画像データ

単位: なし

① 該当種類の欄の「Add」ボタンをクリックし、ポートを追加後、直接入力で名称設定

ポート)の情報を設定します。

*ポート名 (OutPort)
dp_name

Buttons: Add, Delete

② 設定する型情報を一覧から選択

Detail

このセクションではデータポート毎の概要を説明するドキュメントを記述します。上のデータポートを選択すると、それぞれのドキュメントが記述できます。

ポート名: originalImage (InPort)

*データ型: RTC:CameraImage

変数名: RTC:BumperArrayGeometry
RTC:BumperGeometry
RTC:CameraImage

表示位置: RTC:CameraInfo
RTC:Carlike

Documentation:

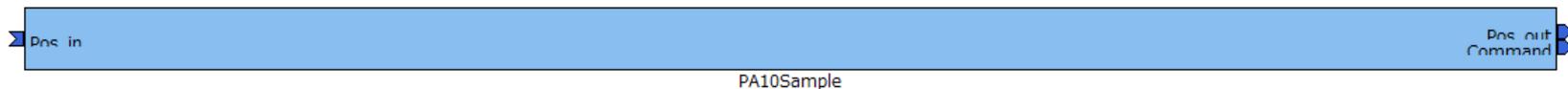
※データ型は、型定義が記載されたIDLファイルを設定画面にて追加することで追加可能

※OpenRTM-aistにて事前定義されている型については、デフォルトで使用可能
→[RTM_Root]rtm/idl 以下に存在するIDLファイルで定義された型

※各ポートに対する説明記述を設定可能

→記述した各種コメントは、生成コード内にDoxygen形式で追加される

※Portの設定内容に応じて，下部のBuildViewの表示が変化



- InPort

ポート名: Pos_in

データ型: RTC::TimedDoubleSeq

変数名: Pos_in

表示位置: left

- OutPort1

ポート名: Command

データ型: RTC::TimedString

変数名: Command

表示位置: right

- OutPort2

ポート名: Pos_out

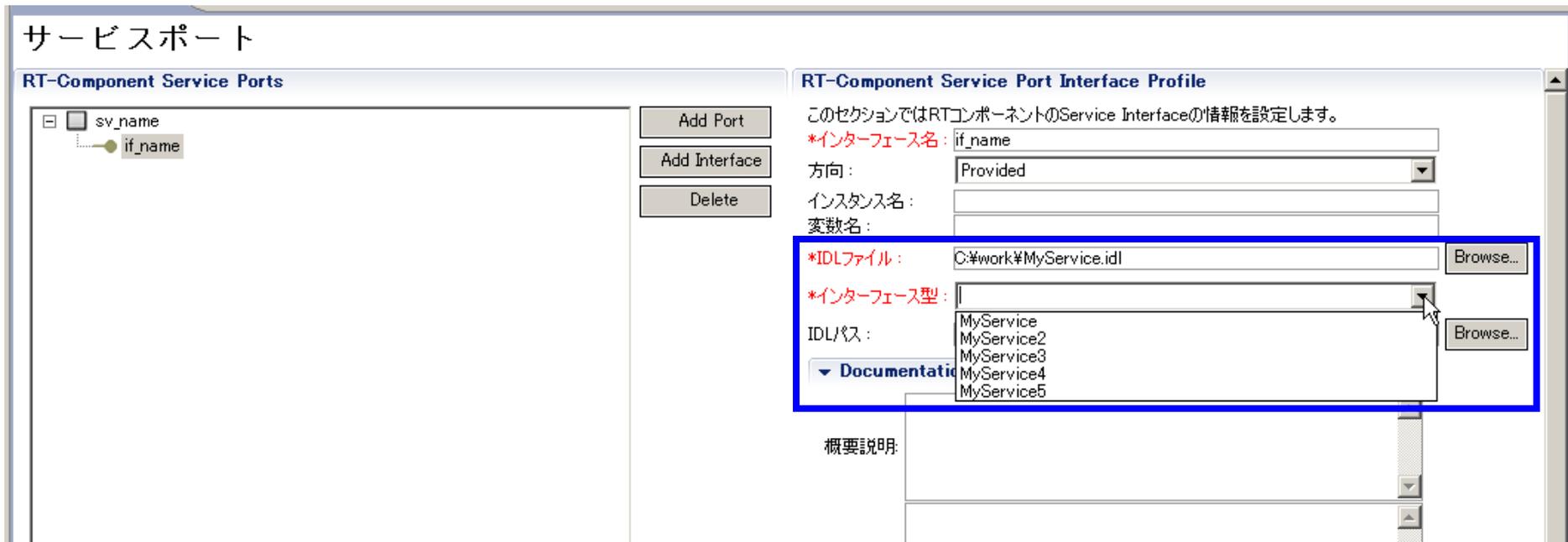
データ型: RTC::TimedDoubleSeq

変数名: Pos_out

表示位置: right

サービスポートの設定

■ 生成対象RTCに付加するServicePortの情報を設定



■ サービスインターフェースの指定

- IDLファイルを指定すると、定義されたインターフェース情報を表示

今回のサンプルでは未使用

コンフィギュレーションの設定

■ 生成対象RTCで使用する設定情報を設定

コンフィギュレーション・パラメータ

▼ RT-Component Configuration Parameter Definitions

このセクションではRTコンポーネントのコンフィギュレーション・パラメータを指定します。

*名称	
flipMode	

Add Delete

▼ Detail

このセクションでは各コンフィギュレーション・パラメータの詳細情報を指定します。

パラメータ名: flipMode

*データ型: int

*デフォルト値: 0

変数名: flipMode

単位:

制約条件: (-1,0,1)

Widget: radio

Step:

Documentation

データ名: flipMode

デフォルト値: 0

概要説明: 画像の反転方法を指定するパラメータ

単位: なし

データ範囲: -1,0,1

制約条件: 0: 上下反転したい場合
1: 左右反転したい場合
-1: 上下左右反転したい場合

▼ ヒント

Config. Param: RTコン
*名称
*データ型
*デフォルト値
*変数名
*単位
*制約条件
*Widget
*Step

パラメータ名: コン
*名称
*データ型
*デフォルト値
*変数名
*単位
*制約条件
*Widget
*Step

データ型: コン
*名称
*データ型
*デフォルト値
*変数名
*単位
*制約条件
*Widget
*Step

デフォルト値: コン
*名称
*データ型
*デフォルト値
*変数名
*単位
*制約条件
*Widget
*Step

変数名: コン
*名称
*データ型
*デフォルト値
*変数名
*単位
*制約条件
*Widget
*Step

単位: コン
*名称
*データ型
*デフォルト値
*変数名
*単位
*制約条件
*Widget
*Step

制約条件: コン
*名称
*データ型
*デフォルト値
*変数名
*単位
*制約条件
*Widget
*Step

Widget: コン
*名称
*データ型
*デフォルト値
*変数名
*単位
*制約条件
*Widget
*Step

Step: コン
*名称
*データ型
*デフォルト値
*変数名
*単位
*制約条件
*Widget
*Step

①「Add」ボタンをクリックし、追加後、直接入力で名称設定

▼ RT-Component Configuration Parameter Definitions

このセクションではRTコンポーネントのコンフィギュレーション・パラメータを指定します。

*名称	
conf_name0	

Add Delete

②詳細画面にて、型情報、変数名などを設定

今回のサンプルでは未使用

※データ型は、short,int,long,float,double,stringから選択可能(直接入力も可能)

※制約情報とWidget情報を入力することで、RTSystemEditorのコンフィギュレーションビューの表示を設定することが可能

コンフィグレーションにおける制約条件の設定方法

■ 制約条件について

- データポートとコンフィギュレーションに設定可能
- チェックはあくまでも**コンポーネント開発者側の責務**
 - ミドルウェア側で検証を行っているわけではない

■ 制約の記述書式

- 指定なし: 空白
- 即値: 値そのもの
 - 例) 100
- 範囲: $<$, $>$, $<=$, $>=$
 - 例) $0 \leq x \leq 100$
- 列挙型: (値1, 値2, ...)
 - 例) (val0, val1, val2)
- 配列型: 値1, 値2, ...
 - 例) val0, val1, val2
- ハッシュ型: { key0: 値0, key1: 値1, ... }
 - 例) { key0: val0, key1: val1 }

■ Widget

- text(テキストボックス)
 - デフォルト
- slider(スライダ)
 - **数値型**に対して**範囲指定**の場合
 - 刻み幅をstepにて指定可能
- spin(スピナ)
 - **数値型**に対して**範囲指定**の場合
 - 刻み幅をstepにて指定可能
- radio(ラジオボタン)
 - 制約が**列挙型**の場合に指定可能

※指定したWidgetと制約条件がマッチしない場合は、テキストボックスを使用

開発環境・動作環境の設定

■ 生成対象RTCを実装する言語，動作環境に関する情報を設定

言語・環境

▼ 言語

このセクションでは使用する言語を指定します

C++
 Python
 Java
 Ruby

Use old build environment.

▼ ヒント

言語: RTコンポーネントを作成する言語を選択します。リスト中の言語から選択可能です。
 環境: 言語ごとのライブラリの依存関係や、使用するOSなどの環境を選択します。
 詳細情報で設定した内容(OS情報、ライブラリ情報など)は、プロフィール内にもみ保存されます。

▼ 環境

このセクションでは依存するライブラリや使用するOSなどを指定します

Version	OS

Add
Delete

このチェックボックスをONにすると、旧バージョンと同様なコード(Cmakeを利用しない形式)を生成

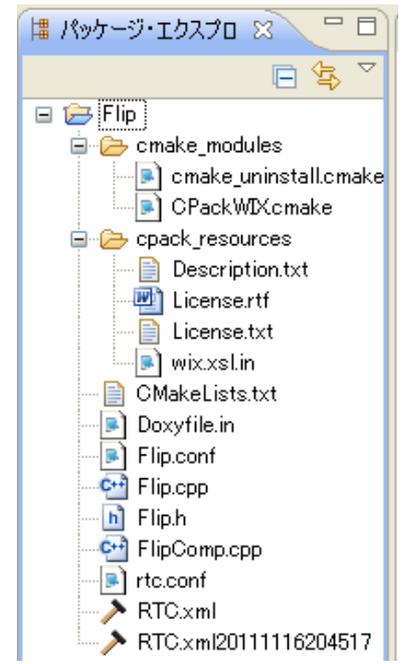
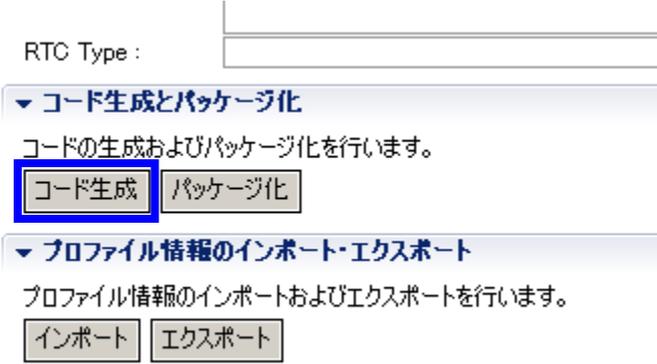
詳細情報

OS Version	Add	CPU	Add
	Delete		Delete

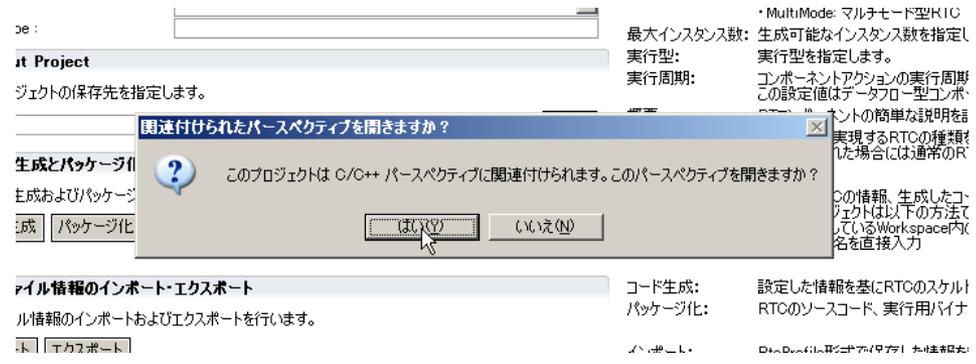
「C++」を選択

コード生成

■ コード生成



■ コード生成実行後，パースペクティブを自動切替

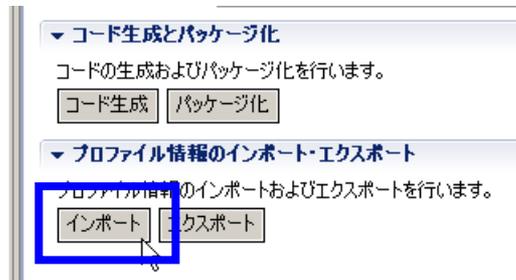


※生成コードが表示されない場合には、「リフレッシュ」を実行

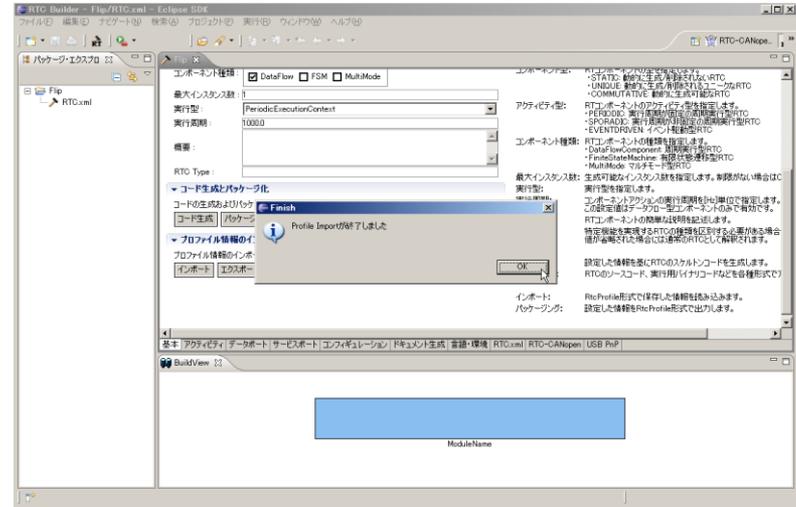
C++版RTC → CDT
 Java版RTC → JDT
 (デフォルトインストール済み)
 Python版 → PyDev

既存のRTCの設定を利用する場合

①「基本」タブ下部の「インポート」ボタンをクリック



②【インポート】画面にて対象ファイルを選択



- 作成済みのRTコンポーネント情報を再利用
 - 「エクスポート」機能を利用して出力したファイルの読み込みが可能
 - コード生成時に作成されるRtcProfileの情報を読み込み可能
 - XML形式, YAML形式での入出力が可能

CMakeの利用

- RTC Builderで出力したファイル群そのものでは, RTCの実行ファイルの生成はできない.
- Cmakeを利用し, ソースファイルのコンパイルに必要な設定が含まれたVisual Studio用のソリューションファイルを生成する.
 - Linuxの場合はソースファイルのコンパイルに必要な設定が含まれたMakefileを生成する.
- CMakeの起動(Windows 7の場合)
 - 「スタート」->「すべてのプログラム」->「CmakeX.X」->「CMake(CMake-gui)」
- Ubuntuの場合
 - Dushホームから, CMakeと入力するとCMake-guiがでてくるので, それを利用.

Cmakeの起動画面・説明

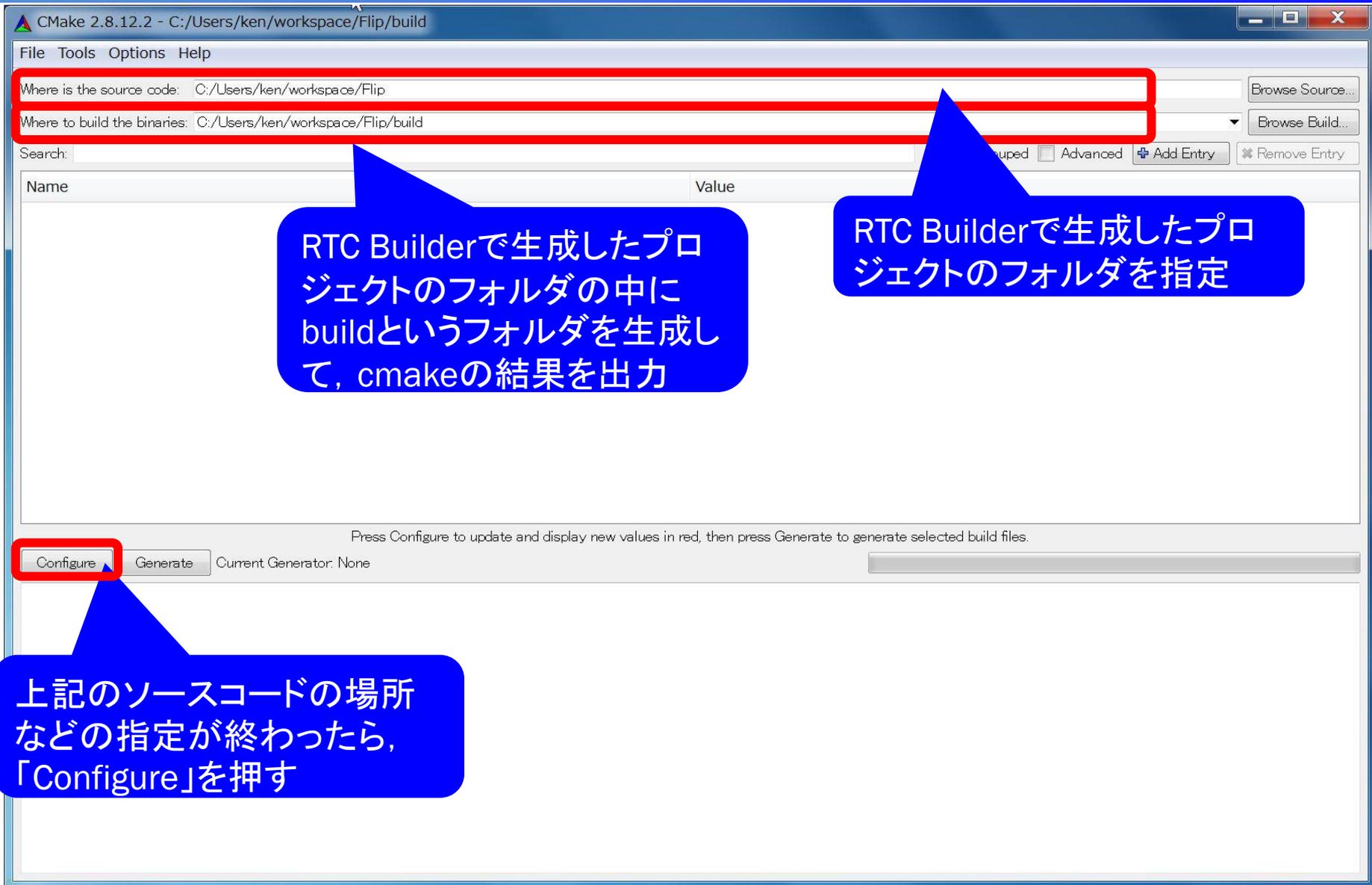
1. ソースファイルの場所を入力

2. ソリューションファイルなどを出力する場所を入力。区別しやすいように「build」というフォルダを指定することが多い。

3. 「Configure」のボタンを押すと、指定されたソースコードをコンパイルするのに必要な情報を収集する。

4. 「Generate」を押すと、ソリューションファイルなどが生成される。

生成したソースファイルをCMake



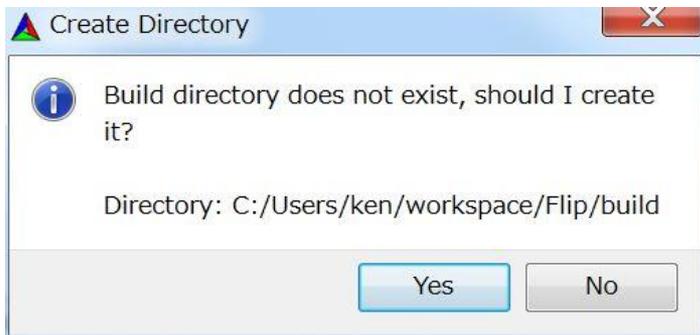
The screenshot shows the CMake 2.8.12.2 GUI. The title bar reads "CMake 2.8.12.2 - C:/Users/ken/workspace/Flip/build". The menu bar includes "File", "Tools", "Options", and "Help". Two input fields are highlighted with a red rectangle: "Where is the source code: C:/Users/ken/workspace/Flip" and "Where to build the binaries: C:/Users/ken/workspace/Flip/build". Below these is a search bar and a table with columns "Name" and "Value". At the bottom, the "Configure" button is highlighted with a red rectangle. Three blue callout boxes provide instructions in Japanese.

RTC Builderで生成したプロジェクトのフォルダの中に build というフォルダを生成して, cmakeの結果を出力

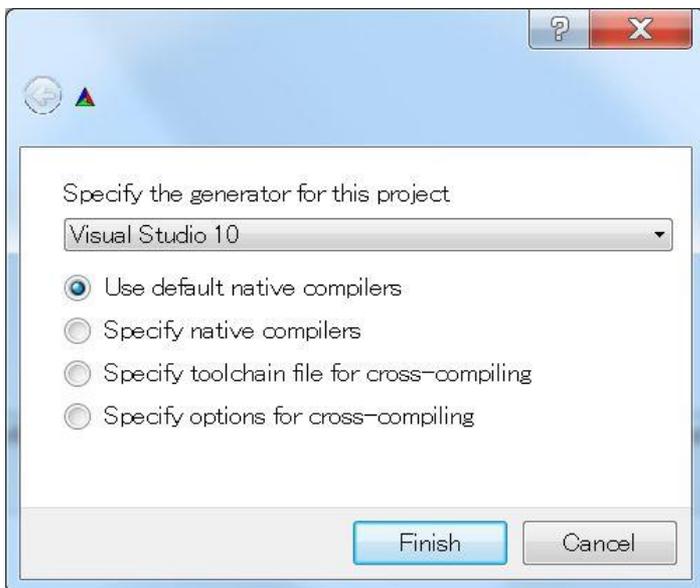
RTC Builderで生成したプロジェクトのフォルダを指定

上記のソースコードの場所などの指定が終わったら, 「Configure」を押す

生成したソースをCMake



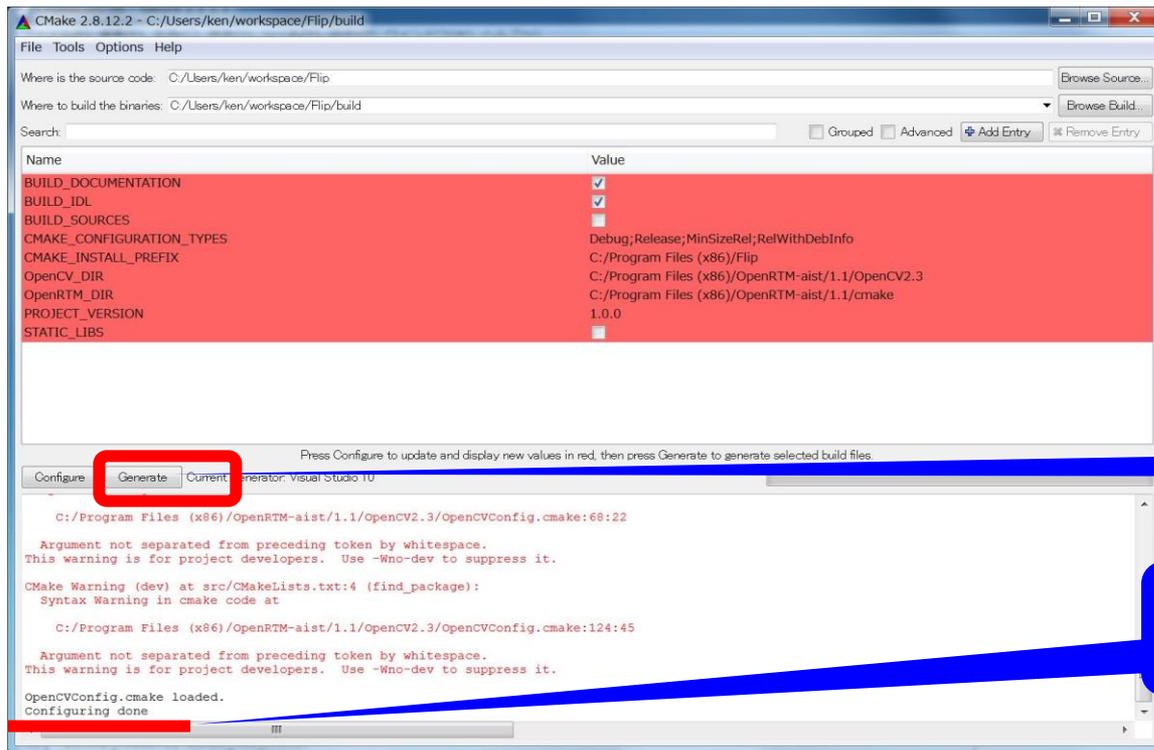
出力先に指定したbuildのフォルダがない場合、生成する旨が表示される



使用するビルド環境を指定する。
Visual Studioであればそのバージョンを指定。
(Visual Studioのバージョンとの表記の違いに注意)

Visual Studio 2010 -> Visual Studio 10
Visual Studio 2012 -> Visual Studio 11
Visual Studio 2013 -> Visual Studio 12
Visual Studio 2015 -> Visual Studio 13
Linuxの場合 Unix Makefiles を指定

生成したソースをCMake



2. 「Generate」をクリック

1. 「Configuration Done」と出てい
ればOK



3. 「Generation Done」と出ればOK

まとめ

- 第2部では, RTコンポーネント開発とRTコンポーネントを用いたシステム構築に必要なツールであるRT System Editorの使い方を体験した.
- RTC BuilderやRT System Editorについては, 産総研原氏によりブラウザ上で動作するバージョンが開発が進められている.

<http://openrtp.org/r tcbow/index.html>

http://hara.jpn.com/siwiki/_hara/ja/Software/RTSEoW.html

- RT System Editorを用いたシステム構築は初期段階での運用には適しているが, 実運用段階では, rtshellなどのRTシステムの自動構築を可能にするツールの利用が好ましい. (1.1.2からはインストーラに同梱)

<http://openrtm.org/openrtm/ja/node/869>