

## 第3部の資料

### ■ 設定ファイルの差し替え

- [OpenRTM\_Root]/1.0/bin 配下にある以下の設定ファイルを差し替え

`rtc.conf`, `configsample.conf`

※デフォルト設定でインストールした場合

`C:\Program Files\OpenRTM-aist\1.0\bin`

# RTミドルウェア講習会

日時: 2011年1月21日(金) 11:00~17:00  
場所: 山形大学 次世代ロボットデザインセンター



# RTミドルウェア講習会

11:00- 12:00	<b>第1部: RTミドルウェアの現状と今後の展開について</b>
	担当: 神徳徹雄 (産総研)
	概要: RTミドルウェアの現状および、今後の展望について。
13:00- 13:45	<b>第2部: OpenRTM-aist サンプルコンポーネントの紹介とその利用法</b>
	担当: 栗原真二 (産総研)
	概要: 開発実習にて作成するRTCに関するサンプルコンポーネントについて紹介し、RTCの便利さ、面白さを体感して頂きます。
14:00- 15:00	<b>第3部: OpenRTM-aist開発支援ツールの紹介とその利用法</b>
	担当: Geoffrey Biggs (産総研)
	概要: RTコンポーネントを作成するツールRTCBuilder、およびRTシステムを設計するツールRTSystemEditor(GUI版)、rtshell(CUI版)の使い方について解説します。
15:15- 17:00	<b>第4部: コンポーネント開発実習</b>
	担当: 栗原真二 (産総研)
	概要: OpenRTMのインストール方法やテスト方法を解説します。OpenRTM-aistでのコンポーネント作成方法を実際に体験していただきます。RTCBuilderを使用したRTコンポーネントの設計とRTSystemEditorでのRTシステム作成を行います。



# OpenRTM-aist開発支援ツールの紹介と その利用法

産業技術総合研究所  
知能システム研究部門

Geoffrey Biggs

栗原 眞二

# OpenRTM-aistの開発支援ツールについて



# OpenRT Platform

## ■ 次世代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

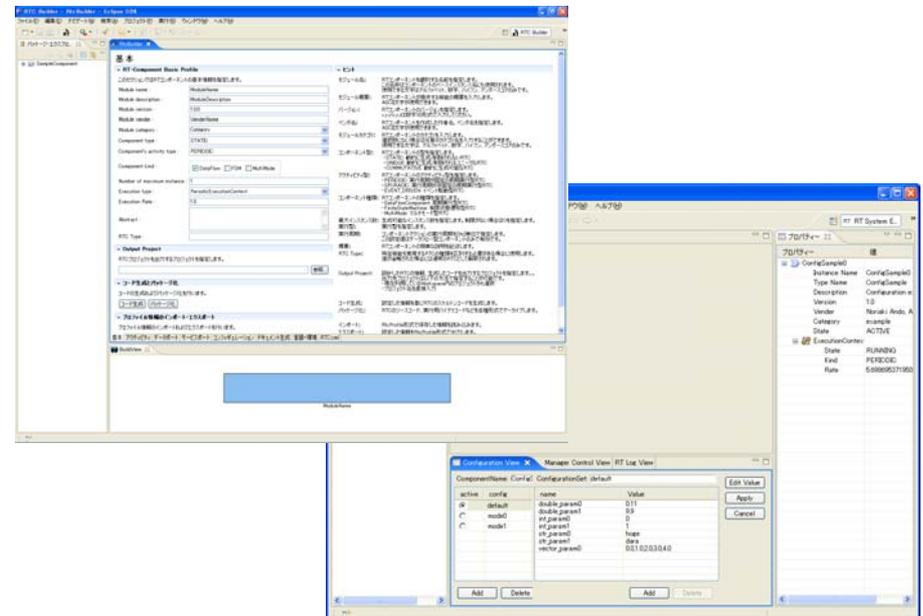
## ■ OpenRT Platformツール群

- 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

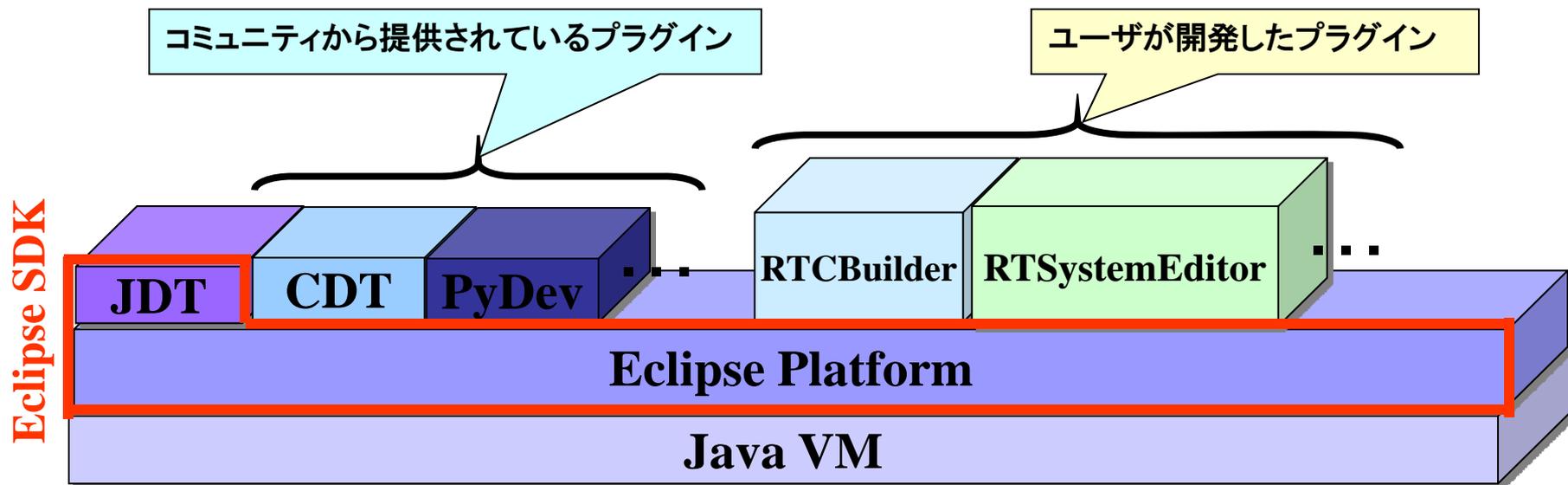
- RTCビルダ
- RTCデバッガ
- RTシステムエディタ
- ロボット設計支援ツール
- シミュレータ
- 動作設計ツール
- シナリオ作成ツール

など



# 統合開発環境Eclipse

- 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
  - 「Plug-in」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



# RTCBuilder, RTSystemEditorのインストール

## ■ ダウンロードし、解凍するだけ

- Javaの実行環境については、別途インストールが必要

The screenshot shows the OpenRTM-aist website interface. The main content area is titled "OpenRTM Eclipse tools 1.0-RELEASE". It includes a "Table of contents" section with links to "全部入りパッケージ" (Full Package), "バイナリ" (Binaries), "RTSystemEditor/RTCBuilderデベロップメント" (Development), "Eclipse/JDK/JRE等" (Eclipse/JDK/JRE etc.), and "過去のバージョン" (Previous versions). Below this is a table of download links for Eclipse 3.4.2 [Ganymede SR2] on various operating systems.

Eclipse-3.4.2 [Ganymede SR2]		
Eclipse3.4.2-RTSE+RTCB Windows用全部入り	<a href="#">eclipse342_rtmttools100release_win32_ja.zip</a> MD5:A52450B24F0A1C59402D5340D9FA8D56	2010.06.01
Eclipse3.4.2-RTSE+RTCB (英語版) Windows用全部入り	<a href="#">eclipse342_rtmttools100release_win32_en.zip</a> MD5:2A1895F0E01D874E35CDC29EFDCE1DE7	2010.06.01
Eclipse3.4.2-RTSE+RTCB Linux用全部入り	<a href="#">eclipse342_rtmttools100release_linux_ja.tar.gz</a> MD5:F5486388B72A351D92ACD22CD4099C7	2010.06.01
Eclipse3.4.2-RTSE+RTCB (英語版) Linux用全部入り	<a href="#">eclipse342_rtmttools100release_linux_en.tar.gz</a> MD5:4B1F4ACEE7F8E99B9C36D080680BE5E1	2010.06.01
Eclipse3.4.2-RTSE+RTCB MacOSX用全部入り	<a href="#">eclipse342_rtmttools100release_macosx_ja.tar.gz</a> MD5:19277C8E1E672688347C6767B57D9D1F	2010.06.10
Eclipse3.4.2-RTSE+RTCB (英語版) MacOSX用全部入り	<a href="#">eclipse342_rtmttools100release_macosx_en.tar.gz</a> MD5:72C63AEE628CFA9F3919A2550AF4604E	2010.06.10

Additional notes at the bottom of the page:

- Ubuntu8.04, Ubuntu9.10, Ubuntu10.04でLinux用Eclipse3.4.2が動か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。

# ツールの起動

- 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

# 準備

## ■ Naming Serviceの起動

### ■ [スタート]メニューから

[プログラム] → [OpenRTM-aist] → [C++] → [tools] → [Start Naming Service]

## ■ ConsoleIn コンポーネント/ConsoleOut コンポーネントの起動

### ■ [スタート]メニューから

[プログラム] → [OpenRTM-aist] → [C++]

→ [components] → [examples] → [ConsoleInComp.exe]

[プログラム] → [OpenRTM-aist] → [C++]

→ [components] → [examples] → [ConsoleOutComp.exe]

```

Start Naming Service
Starting omniORB omniNames: khatib:2809

Mon Jun 07 19:03:12 2010:

Starting omniNames for the first time.
Wrote initial log file.
Read log file successfully.
Root context is IDR:010000002e00000049444e3a6f6d672e6f72672f436f734e616d696e672f4
4e616d696e67436f6e746578744578743a312e3000000100000000000000000000010102000d00
00003139322e313632e34362e310000f90a0e0000004e616d655365727669636550002000000000
0000080000000100000000054544101000001e0000001000000010001000100000010001050301
01000100000009010100
Checkpointing Phase 1: Prepare.
Checkpointing Phase 2: Commit.
Checkpointing completed.
    
```

```

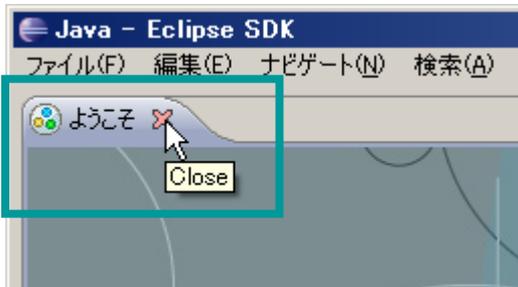
ConsoleInComp.exe
Creating a component: "ConsoleIn"...succeed.
-----
Component Profile
-----
InstanceID: ConsoleIn0
Implementation: ConsoleIn
Description: Console input component
Version: 1.0
Maker: Noriaki Ando, AIST
Category: example
Other properties
-----
Port0 (name): ConsoleIn0.out
-----
- properties -
port.port_type: DataOutPort
dataport.data_type: TimedLong
dataport.subscription_type: flush,new,periodic
dataport.dataflow_type: push,pull
dataport.interface_type: corba_cdr
    
```

```

ConsoleOutComp.exe
succeed.
-----
Component Profile
-----
InstanceID: ConsoleOut0
Implementation: ConsoleOut
Description: Console output component
Version: 1.0
Maker: Noriaki Ando, AIST
Category: example
Other properties
-----
Port0 (name): ConsoleOut0.in
-----
- properties -
port.port_type: DataInPort
dataport.data_type: TimedLong
dataport.subscription_type: Any
dataport.dataflow_type: push,pull
dataport.interface_type: corba_cdr
    
```

# 準備

- 初期画面のクローズ
  - 初回起動時のみ

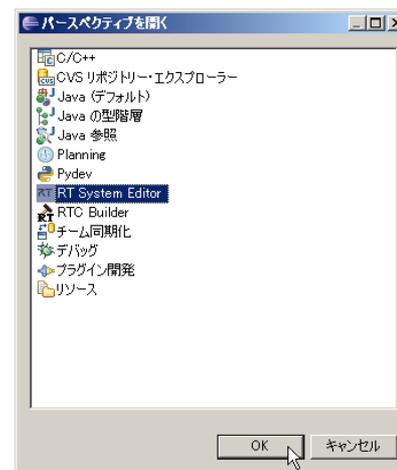
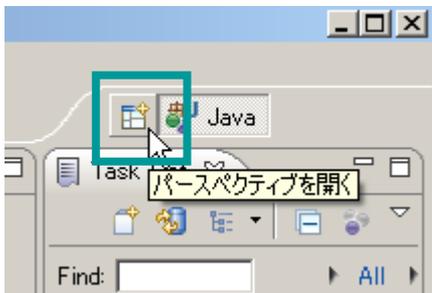


※パースペクティブ  
Eclipse上でツールの構成を管理する単位  
メニュー, ツールバー, エディタ, ビューなど  
使用目的に応じて組み合わせる  
独自の構成を登録することも可能

## ■ 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



# RTSystemEditorについて



# RTSystemEditor概要

## ■ RTSystemEditorとは？

- 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

The screenshot displays the RTSystemEditor interface within the Eclipse SDK. The main window shows a system diagram with components: MyServiceProvider0, ConfigSample0, SequenceOutComponent0, and SequenceInComponent0. The left sidebar shows a project tree for 'localhost.localdomain/host\_ext'. The bottom panel shows the configuration 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	hoge
		str_param1	dara
		vector_param0	0.0,1.0,2.0,3.0,4.0

# 画面構成

システムエディタ

MyServiceProvider0

ConfigSample0

SequenceOutComponent0

SequenceInComponent0

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

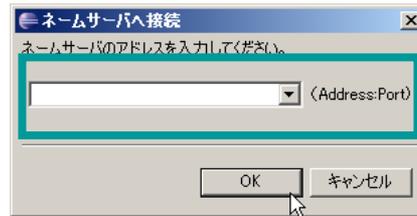
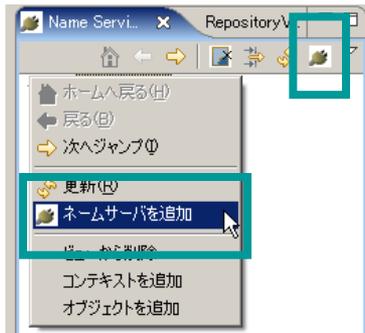
マネージャビュー

コンポジットコンポーネントビュー

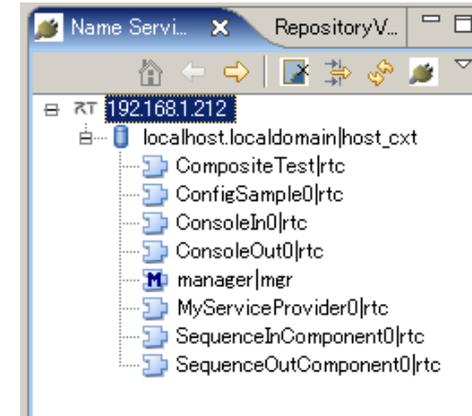
component	port
SequenceInComponent0	Short
SequenceInComponent0	Long
SequenceInComponent0	Float
SequenceInComponent0	Double
SequenceInComponent0	ShortSeq
SequenceInComponent0	LongSeq
SequenceInComponent0	FloatSeq
SequenceOutComponent0	
SequenceOutComponent0	
SequenceOutComponent0	

# RTシステム構築の基本操作

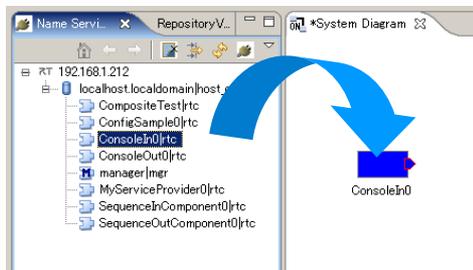
## ■ ネームサービスへ接続



※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



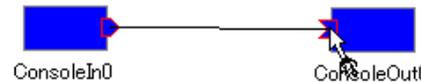
## ■ RTコンポーネントの配置



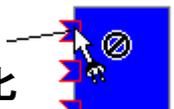
※ネームサービスビューから対象コンポーネントをドラッグアンドドロップ

## ■ 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



# 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
DataType	ポート間で送受信するデータの型. ex)TimedOctet, TimedShortなど
InterfaceType	データを送受信するポートの型. ex)corba_cdrなど
DataFlowType	データの送信方法. ex)push, pullなど
SubscriptionType	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

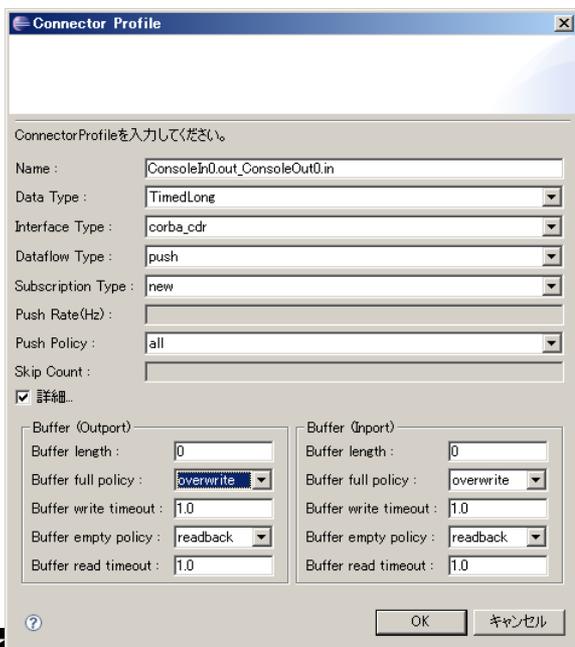
- New : バッファ内に新規データが格納されたタイミングで送信
- Periodic : 一定周期で定期的にデータを送信
- Flush : バッファを介さず即座に同期的に送信

## ■ 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

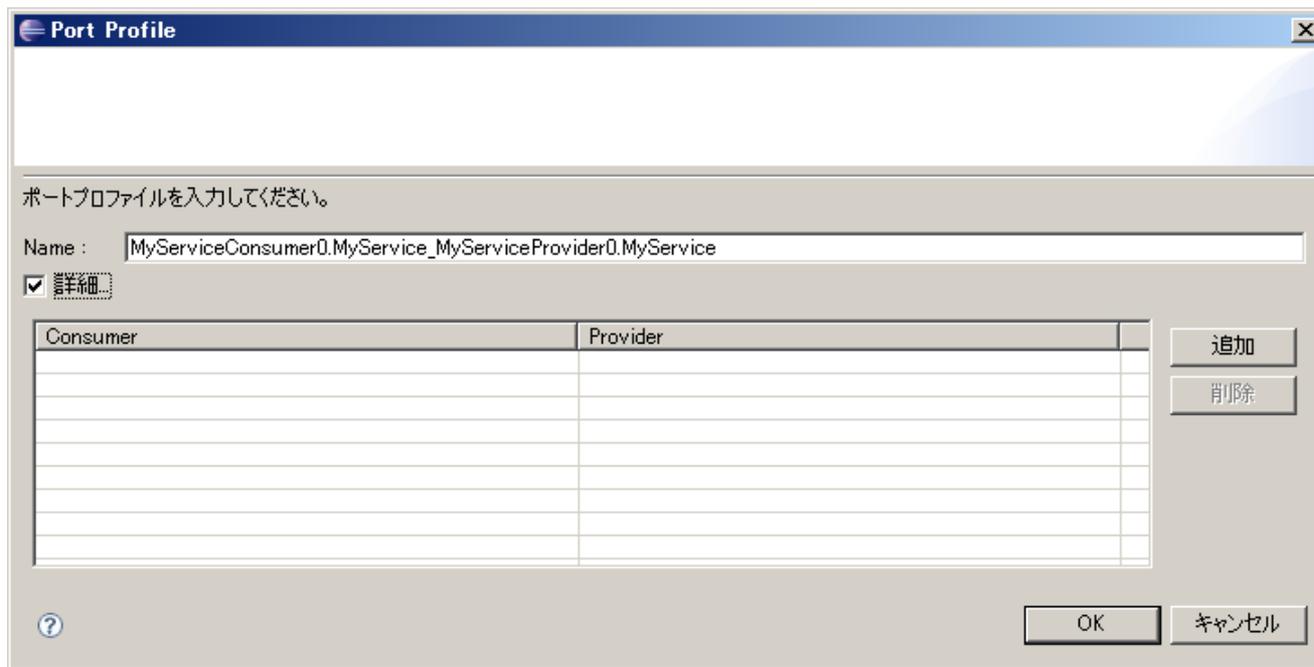
## ■ Buffer Policy

- overwrite : 上書き
- readback : 最後の要素を再読み出し
- block : ブロック
- do\_nothing : なにもしない

- ※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



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

? OK キャンセル

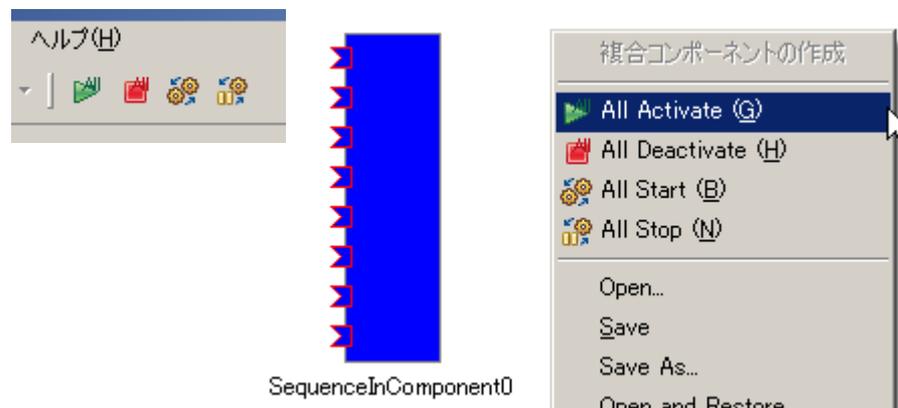
# RTコンポーネントの動作

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

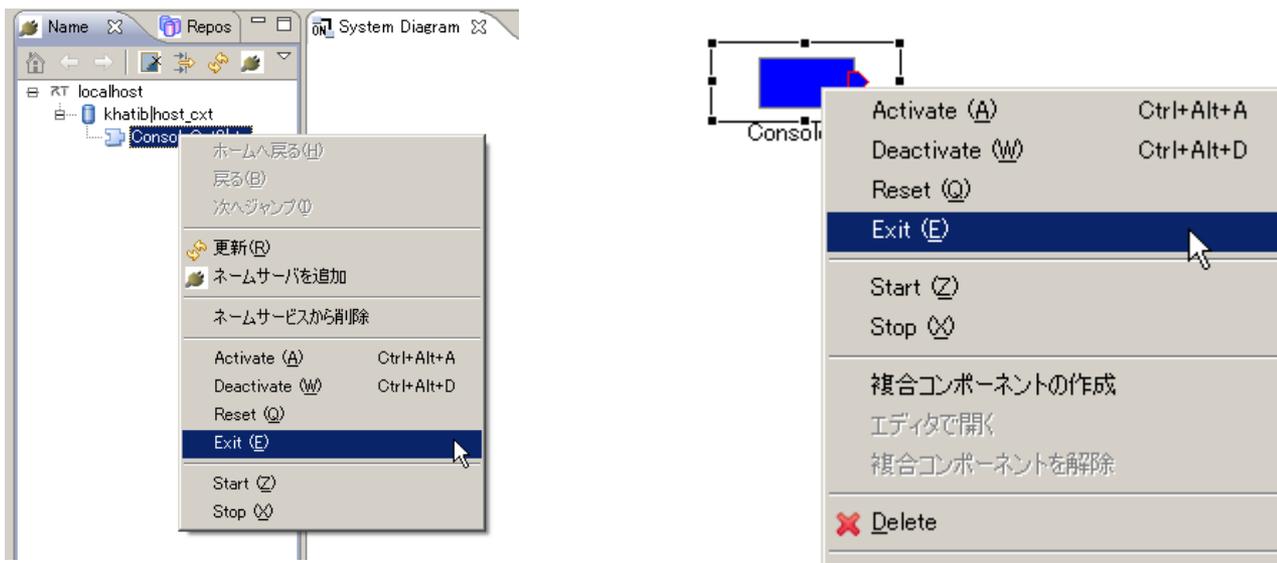


※ポップアップメニュー中でのキーバインドを追加

※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

# サンプルの切り替え

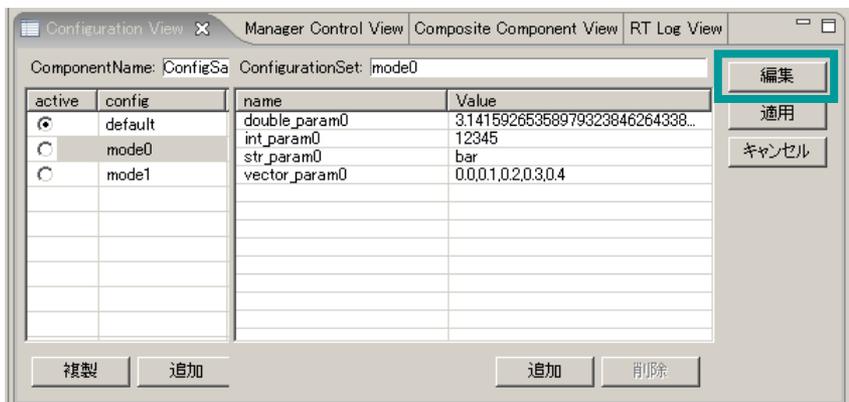
- ConsoleIn コンポーネント/ConsoleOut コンポーネントの終了
  - 入力コンソールを終了
  - ネームサービスビュー内で対象RTCを選択, 右クリックし「Exit」
  - システムエディタ上で対象RTCを選択, 右クリックし「Exit」



- RTC daemonの起動
    - [スタート]メニューから起動
- [プログラム] → [OpenRTM-aist] → [C++] → [tools] → [Start RTC daemon]

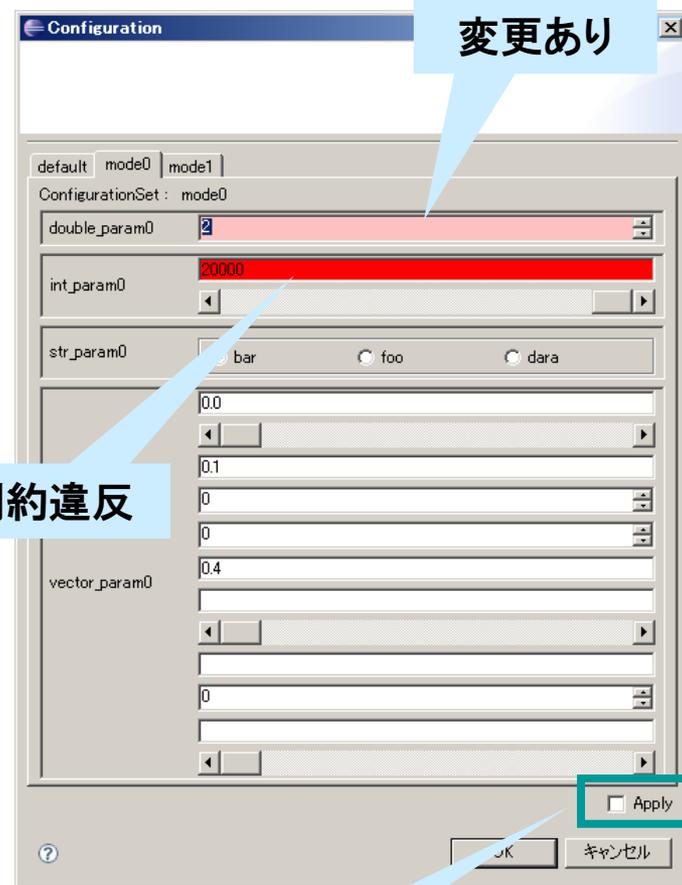
# コンフィギュレーションビュー

## RTコンポーネントのコンフィギュレーション情報の確認/編集



※「編集」ボタンにより、各種コントロールを用いた一括編集が可能

※「Apply」チェックボックスがONの場合、設定値を変更すると即座にコンポーネントに反映  
 →テキストボックスからフォーカス外れる、ラジオボタンを選択する、スライダーを操作する、スピナを変更する、などのタイミング



制約違反

変更あり

即時反映

# コンフィギュレーション情報の設定方法

- rtc.conf内

[カテゴリ名]. [コンポーネント名]. config\_file: [コンフィギュレーションファイル名]

※例) example.ConfigSample.config\_file: configsample.conf

- コンフィギュレーションファイル内

- コンフィギュレーション情報

conf. [コンフィグセット名]. [コンフィグパラメータ名] : [デフォルト値]

※例) conf.mode0.int\_param0: 123

- Widget情報

conf. \_\_widget\_\_. [コンフィグパラメータ名] : [Widget名]

※例) conf.\_\_widget\_\_.str\_param0: radio

- 制約情報

conf. \_\_constraints\_\_. [コンフィグパラメータ名] : [制約情報]

※例) conf.\_\_constraints\_\_.str\_param0: (bar,foo,foo,dara)

conf. \_\_[コンフィグセット名]. [コンフィグパラメータ名] : [制約情報]

※例) conf.\_\_mode1.str\_param0: (bar2,foo2,dara2)

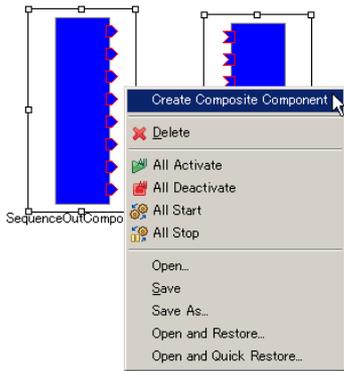
RTCの利用者が設定するのではなく、RTC開発者、RTC管理者が設定することを想定。

RTCBuilderを使用することで設定可能

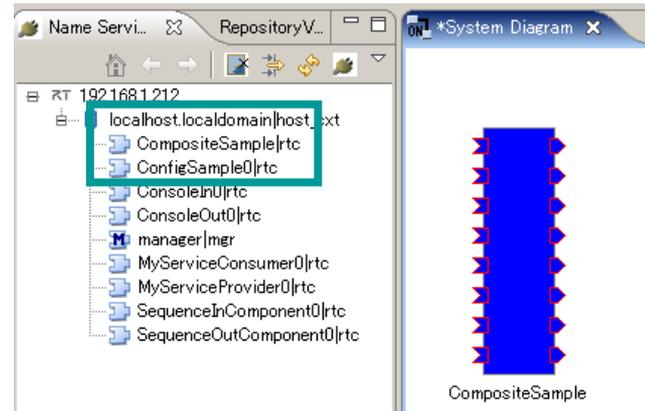
# 複合コンポーネント

- 複数のRTCをまとめて、1つのRTCとして扱うための仕組み
- 複合コンポーネントの作成方法

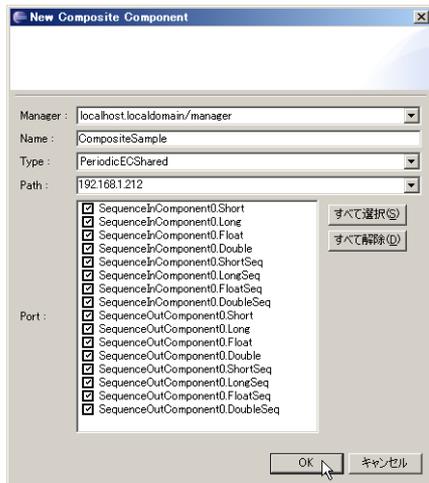
① 複数RTCを選択している状態で右クリック



③ 複合コンポーネントを生成



② 複合コンポーネントのプロパティを設定



項目	設定内容
Manager	複合コンポーネントを制御するマネージャを選択
Name	複合コンポーネントのインスタンス名を入力
Type	複合コンポーネントの型を選択
Path	複合コンポーネントのパスを入力
Port	外部に公開するポートを選択

※生成対象複合コンポーネント外部と接続されているPortは強制的に公開されます

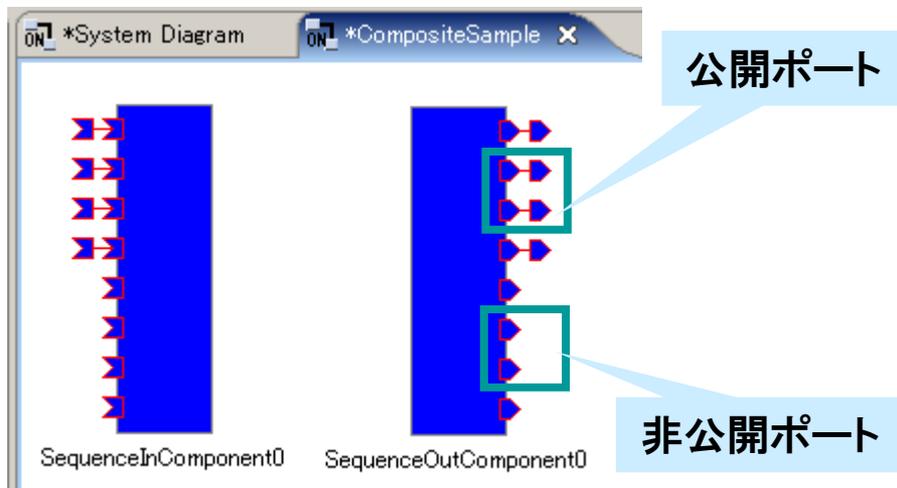
# 複合コンポーネント

## ■ 複合コンポーネントのタイプについて

タイプ名	説明
PeriodicECShared	実行主体であるExecutionContextのみを共有. 各子コンポーネントはそれぞれの状態を持つ
PeriodicStateShared	実行主体であるExecutionContextと状態を共有
Grouping	便宜的にツール上のみでグループ化

## ■ 複合コンポーネントエディタ

- 複合コンポーネントをダブルクリックすることで表示



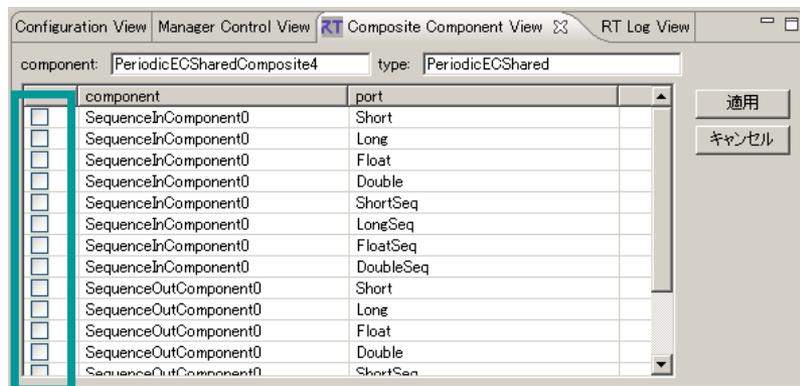
- ※エディタ内に別RTCをDnDすることで、子コンポーネントの追加が可能  
→追加したRTCのポートは全て非公開に設定
- ※エディタ内のRTCを削除することで、子コンポーネントの削除が可能  
→削除されたRTCは、親エディタに表示

# 複合コンポーネント

## ■ 公開ポートの設定

### ● 複合コンポーネントビュー

ポート公開情報

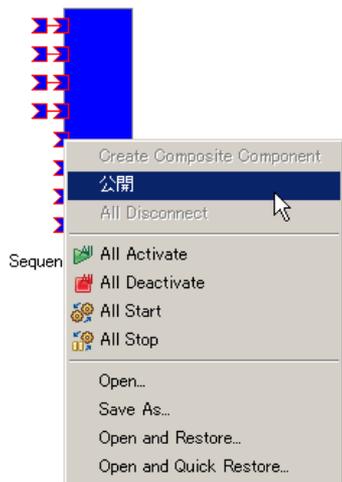


※ポート公開情報を変更し、「適用」をクリック

### ● 複合コンポーネントエディタ

※非公開ポートを「公開」

※公開ポートを「非公開」

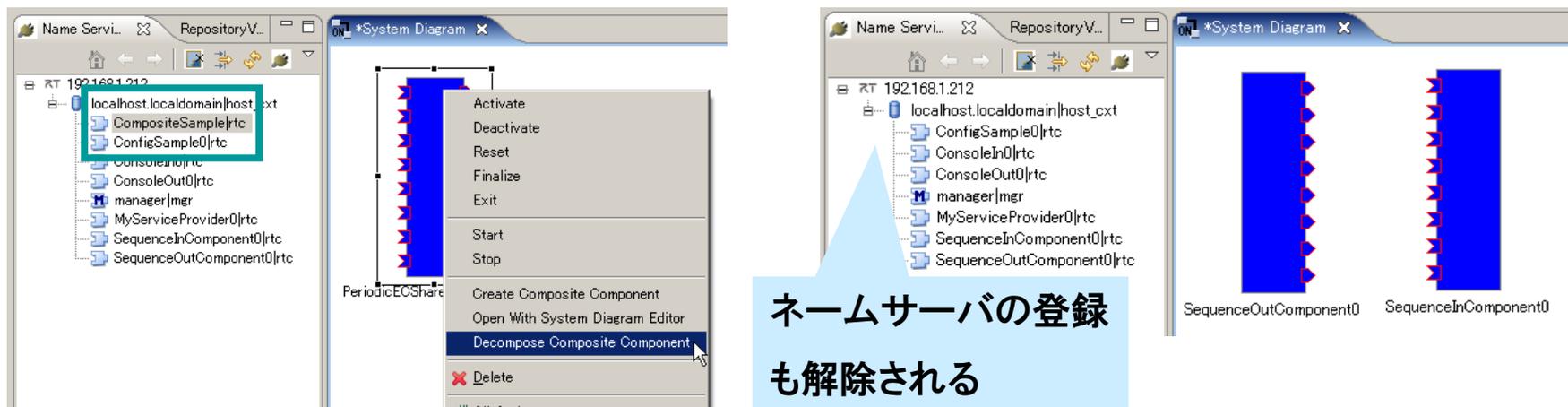


外部コンポーネントと接続されているポートを「非公開」に設定することはできません

# 複合コンポーネント

## ■ 複合コンポーネントの解除

- ① 複合RTCを右クリックし、複合コンポーネントの解除を選択
- ② 複合コンポーネントが分解され、内部のRTCが表示



※エディタ上で、(Deleteキーなどで)単純に削除した場合は、エディタから表示が消えるのみ複合コンポーネントは解除されない

# RTCBuilderについて



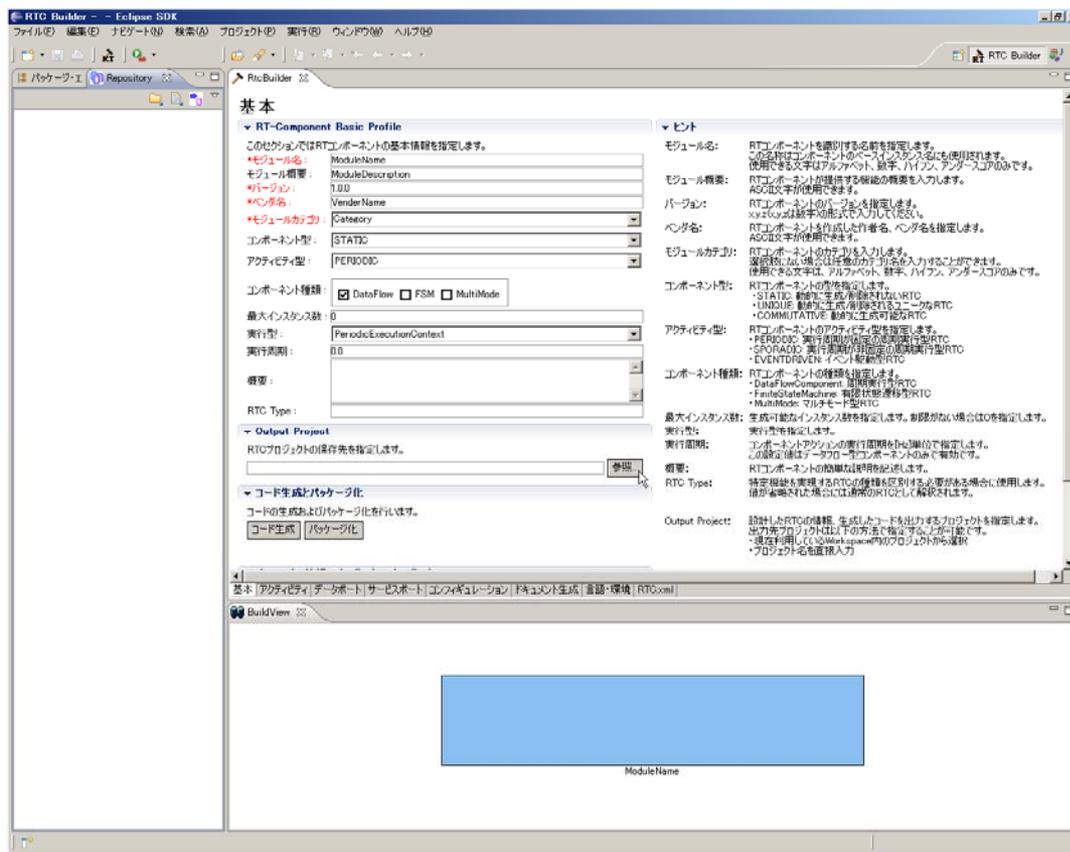
# RTCBuilder概要

## ■ 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



# 画面構成

The screenshot shows the Eclipse IDE with the RTC Builder plugin. The main editor displays the 'RTC-Component Basic Profile' configuration page. The interface is annotated with four callout boxes:

- リポジトリレビュー (Repository Review):** A purple box on the left side of the interface, pointing to the 'Repository' view.
- RTCプロファイルエディタ (RTC Profile Editor):** A blue box in the center, pointing to the 'Basic' configuration tab.
- ヒント (Hint):** A green box on the right, pointing to the 'ヒント' (Hint) section of the configuration page.
- ビルドビュー (Build View):** A pink box at the bottom right, pointing to the 'BuildView' view.

The configuration page includes sections for '基本' (Basic), 'ヒント' (Hint), and 'ビルドビュー' (Build View). The '基本' section contains fields for Module Name, Description, Version, Vendor Name, Category, Component Type, Activity Type, Component Kind, and Execution Type. The 'ヒント' section provides detailed instructions for each field. The 'ビルドビュー' section contains the 'Output Project' field.

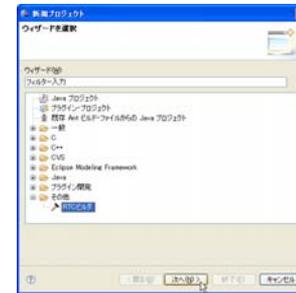
# プロジェクト作成/エディタ起動

※パースペクティブを「RtcBuilder」に切り替え

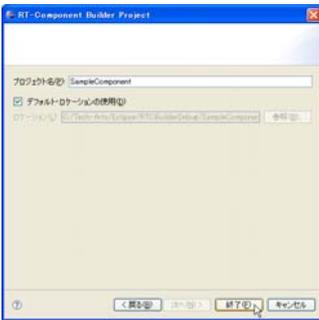
①メニューから「ファイル」-「新規」-「プロジェクト」



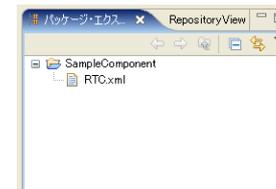
②「その他」-「RtcBuilder」を選択し、「次へ」



③「プロジェクト名」欄に入力し、「終了」



④指定した名称のプロジェクトを生成



※任意の場所にプロジェクトを作成したい場合

③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

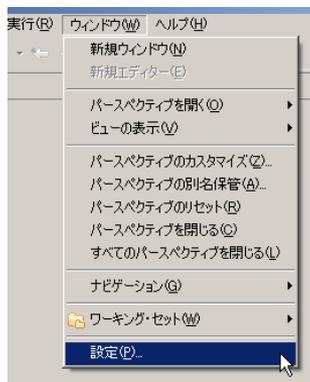
プロジェクト名: Flip

# 各種設定・起動

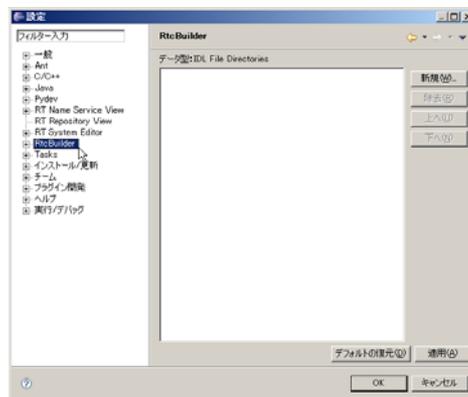
## ■ DataPortにて利用するデータ型の指定

→データ型を定義したIDLファイルが格納されているディレクトリを指定

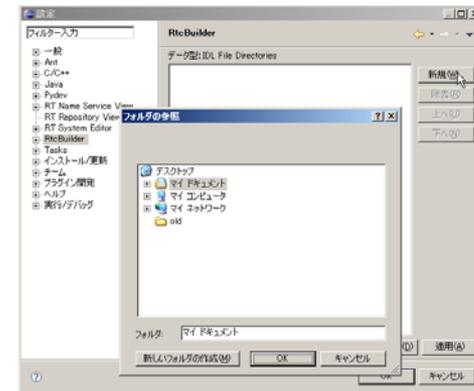
①メニューから  
「ウィンドウ」-「設定」



②「RtcBuilder」を選択

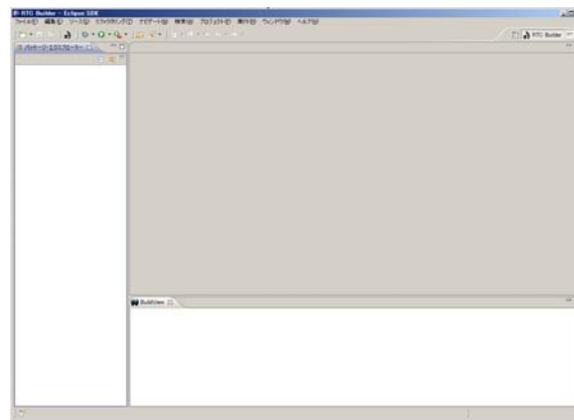
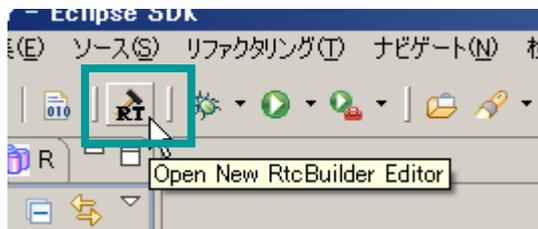


③「新規」ボタンにて表示される  
ディレクトリ選択ダイアログ  
にて場所を指定



## ■ RtcBuilderの起動

画面上部のアイコンをクリック



# 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

# 基本プロファイル

## ■ 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: 1.00  
 \*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:

RTC Type:

▼ Output Project

RTCプロジェクトの保存先を指定します。

Flip

▼ 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。

コード生成 パッケージ化

▼ 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。

インポート エクスポート

▼ ヒント

モジュール名: RTコンポーネントを識別する名称はコンポーネント名として使用できる文字はASCII文字が使用

モジュール概要: RTコンポーネントがASCII文字が使用

バージョン: RTコンポーネントのxyz.cy.dは数字0

ベンダ名: RTコンポーネントをASCII文字が使用

モジュールカテゴリ: RTコンポーネントの選択されない場合使用できる文字は、

コンポーネント型: RTコンポーネントの  
 ・STATIC: 動的に  
 ・UNIQUE: 動的に  
 ・COMMUTATIVE

アクティビティ型: RTコンポーネントの  
 ・PERIODIC: 実行  
 ・SPORADIC: 実行  
 ・EVENTDRIVEN:

コンポーネント種類: RTコンポーネントの  
 ・DataFlowCompo  
 ・FiniteStateMach  
 ・MultiMode: マルチ

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

実行型: 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設定値はデフォルト値

概要: RTコンポーネントの

RTC Type: 特定機能を実現する値が省略された場合

Output Project: 設計したRTCの出力先プロジェクト  
 ・現在利用しているプロジェクト名を直

コード生成: 設定した情報を基に

パッケージ化: RTCのソースコードを

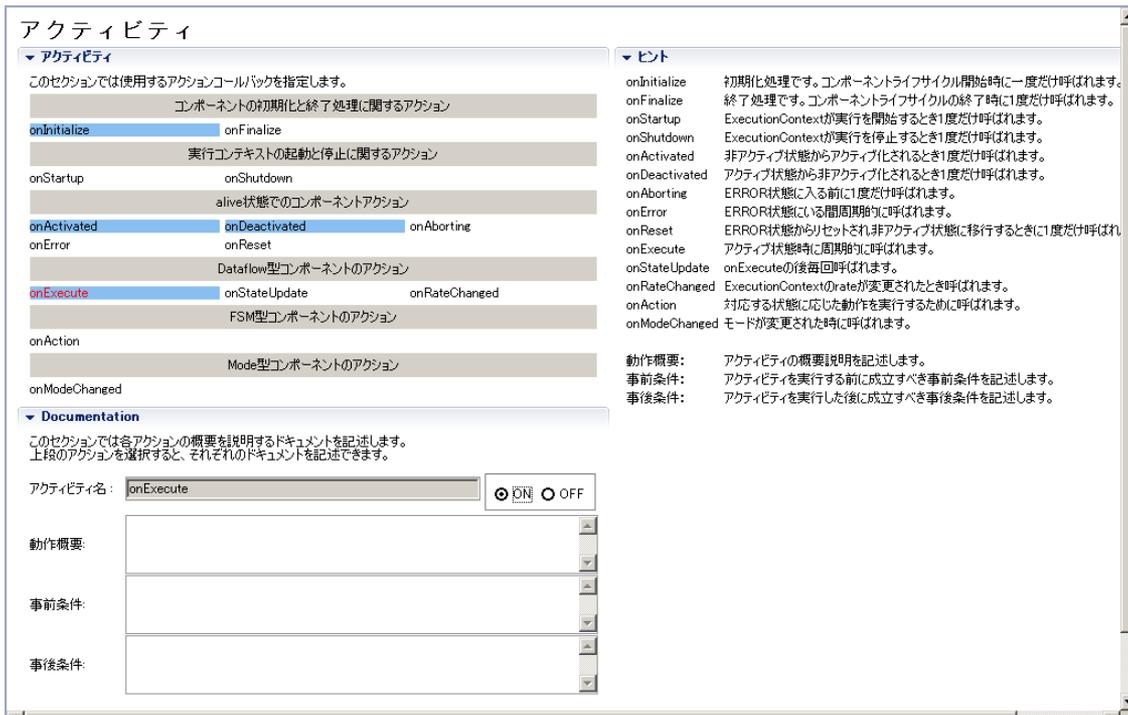
インポート: RTコンポーネントの

モジュール名: Flip  
 モジュール概要: 任意(Flip image component)  
 バージョン: 1.0.0  
 ベンダ名: 任意(AIST)  
 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: 0.0  
 Output Project: Flip

- ※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・プロファイル

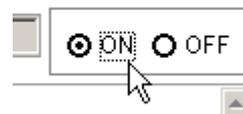
## ■ 生成対象RTCで実装予定のアクティビティを設定



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:  
 on\_activated  
 on\_deactivated  
 on\_execute

- ※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・プロファイル

## ■ 生成対象RTCに付加するDataPortの情報を設定

データポート

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)	Add	*ポート名 (OutPort)	Add
original_image		fliped_image	
	Delete		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: `fliped_image (OutPort)`

\*データ型: `RTC:TimedOctetSeq`

変数名: `image_flip`

表示位置: `RIGHT`

Documentation

概要説明:

データ型:

データ数:

意味:

**▼ ヒント**

データポート: RTコンポ  
データを  
InPortとC

InPort: RTコンポ  
他のRTC

OutPort: RTコンポ  
他のRTC

ポート名: データポ  
ポート名  
ポート名  
ASCII文

データ型: データポ  
InPortとC  
データ型  
使用する

変数名: データポ  
変数の名

ポートの場所: RTSystem  
このプロ

ドキュメント: データポ  
全てを記  
レベルの

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。

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: `original_image (InPort)`

\*データ型: `RTC:Time`

変数名: `RTC:TimedCharSeq`

表示位置: `RTC:TimedBooleanSeq`

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

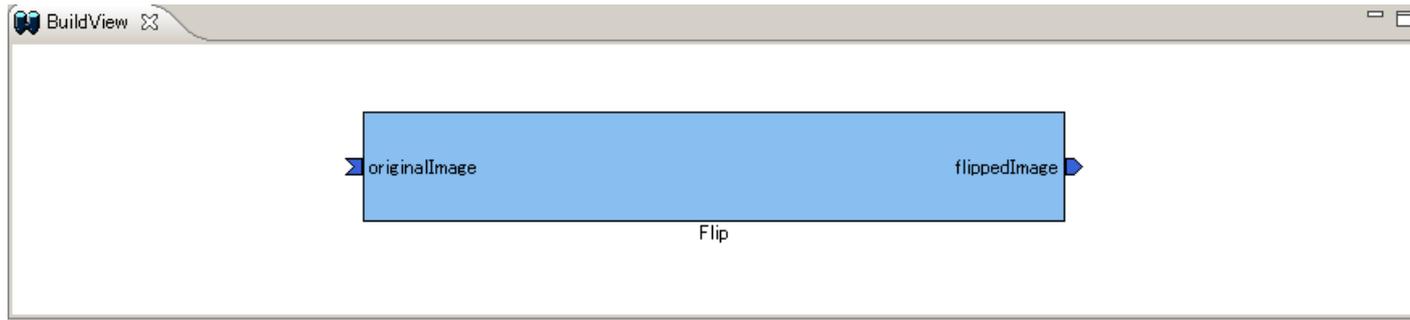
※ OpenRTM-aistにて事前定義されている型については、デフォルトで使用可能  
→ [OpenRTM\_Root]/1.0/rtm/idl 以下に存在するIDLファイルで定義された型

※ 各ポートに対する説明記述を設定可能

RT → 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・プロファイル

※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: **originalImage**

データ型: **RTC::CameraImage**

変数名: **originalImage**

表示位置: **left**

## ● OutPort

ポート名: **flippedImage**

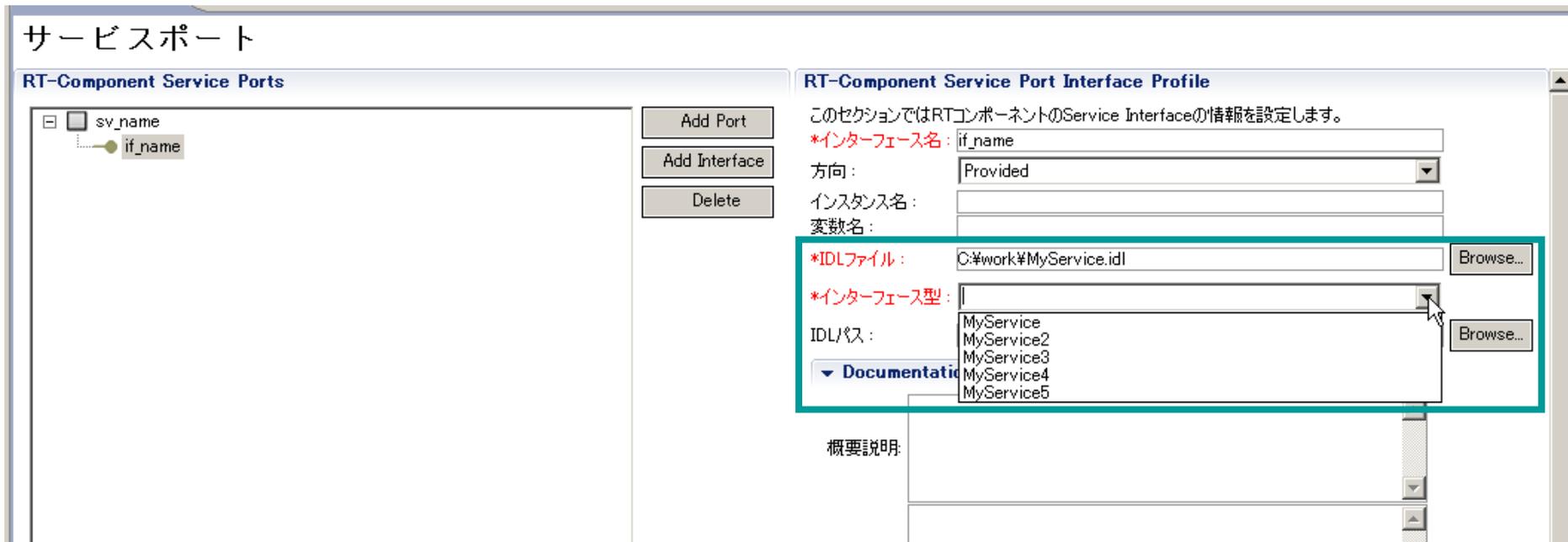
データ型: **RTC::CameraImage**

変数名: **flippedImage**

表示位置: **right**

# サービスポート・プロファイル

## ■ 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



# 制約条件, Widget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

# 言語・環境・プロファイル

- 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++
- Java
- C#
- Python
- VB.NET
- Ruby

**▼ 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、プロファイル内のものに

**▼ 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

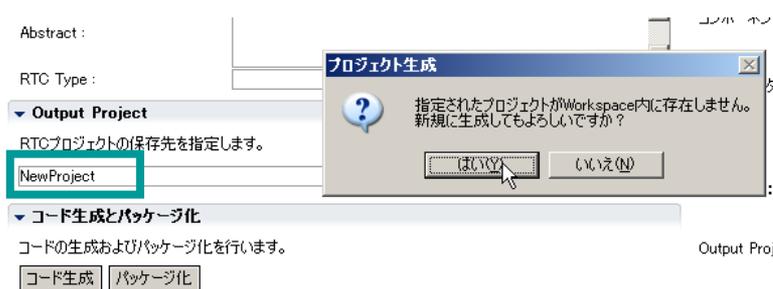
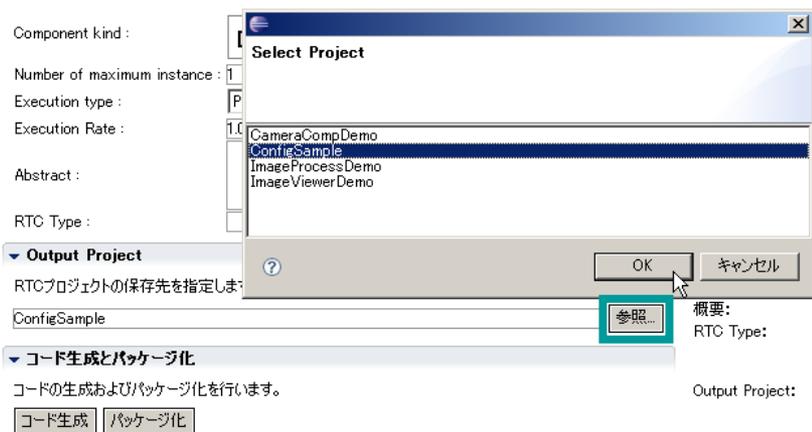
**詳細情報**

<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 80%;">OS Version</th> <th style="width: 20%;"></th> </tr> </thead> <tbody> <tr><td> </td><td style="text-align: center;">Add</td></tr> <tr><td> </td><td style="text-align: center;">Delete</td></tr> </tbody> </table>	OS Version			Add		Delete	<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 80%;">CPU</th> <th style="width: 20%;"></th> </tr> </thead> <tbody> <tr><td> </td><td style="text-align: center;">Add</td></tr> <tr><td> </td><td style="text-align: center;">Delete</td></tr> </tbody> </table>	CPU			Add		Delete
OS Version													
	Add												
	Delete												
CPU													
	Add												
	Delete												

「C++」を選択

# コード生成

## ■ 出力先プロジェクト選択

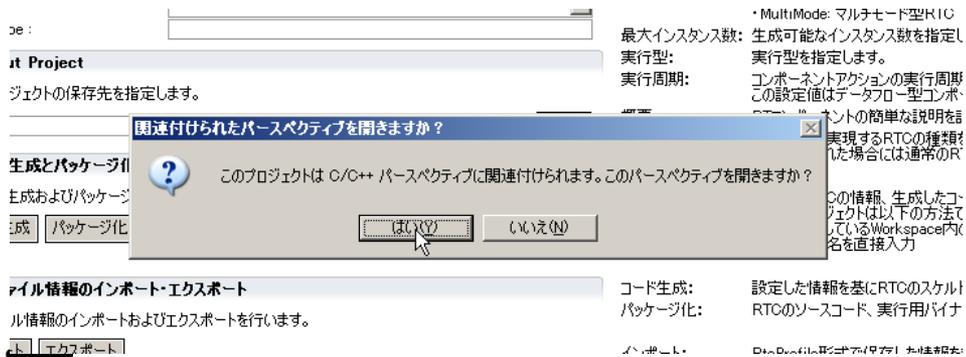


※ワークスペース内のプロジェクトから選択

※プロジェクト名を直接入力

→該当プロジェクトがワークスペース内に存在しない場合、新規作成することも可能

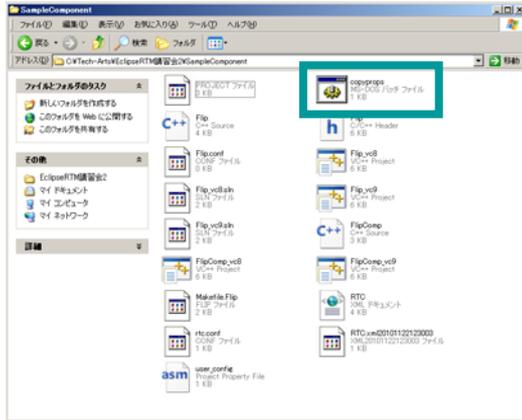
## ■ コード生成実行後、パースペクティブを自動切替



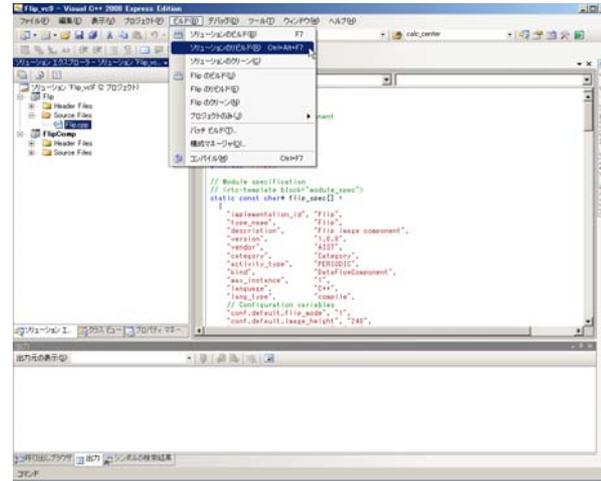
C++版RTC → CDT  
 Java版RTC → JDТ  
 (デフォルトインストール済み)  
 Python版 → PyDev

# コンパイル・実行

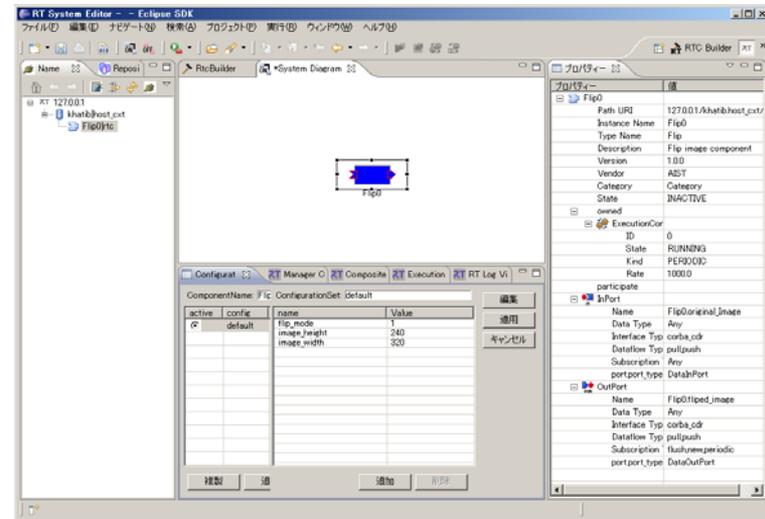
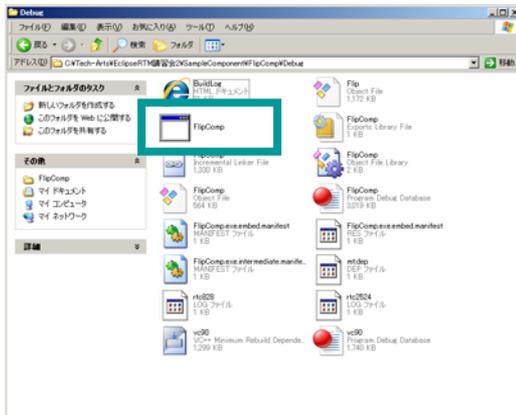
①コード生成先ディレクトリ内の「copyprops.bat」をダブルクリックして、設定ファイルをコピー



②VisualStudioを用いたビルド

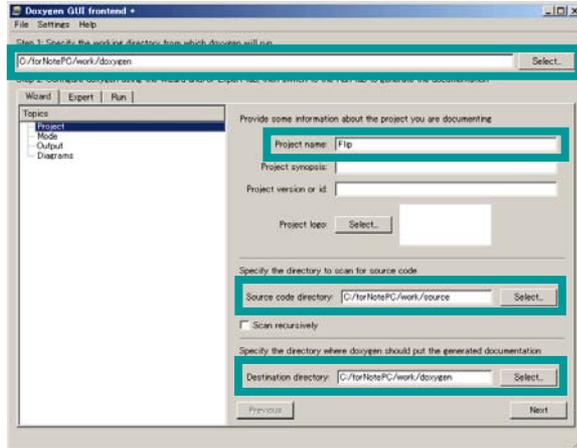


③FlipComp¥¥Debug内のFlipComp.exeを起動

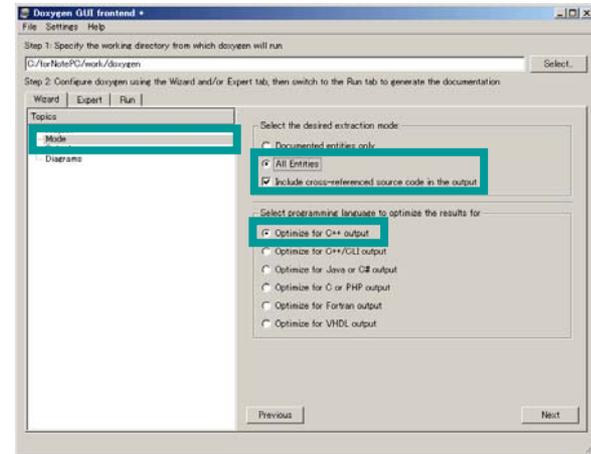


# ドキュメント作成

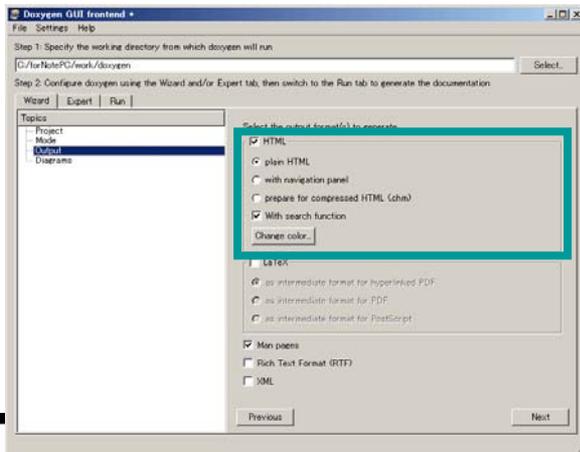
- ① Doxygen用GUIツールを起動  
作業用ディレクトリ,ソース格納場所,  
生成ファイル出力先,プロジェクト名を指定



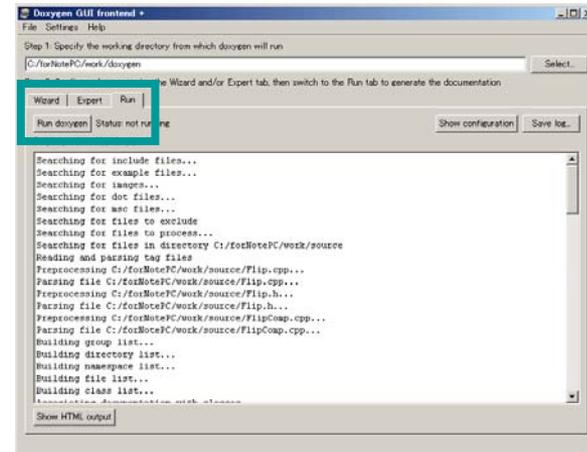
- ② 「Mode」セクションにて,  
出力内容,使用言語を指定



- ③ 「Output」セクションにて, html出力を指定



- ③ 「Run」タブにて, 「Run doxygen」を実行



# ドキュメント作成

## 生成されたドキュメントの例

### Flip

Main Page	<b>Classes</b>	Files	Search
Class List	Class Index	Class Members	

Public Member Functions | Protected Attributes

### Flip Class Reference

Flip image component. More...

```
#include <Flip.h>
```

List of all members.

#### Public Member Functions

	Flip (RTC::Manager *manager)
	constructor
	~Flip ()
	destructor
virtual RTC::ReturnCode_t	onInitialize ()
virtual RTC::ReturnCode_t	onActivated (RTC::UniqueId ec_id)
virtual RTC::ReturnCode_t	onDeactivated (RTC::UniqueId ec_id)
virtual RTC::ReturnCode_t	onExecute (RTC::UniqueId ec_id)

#### Protected Attributes

	int	m_flipMode
CameraImage		m_originalImage
InPort< CameraImage >		m_originalImageIn
CameraImage		m_flippedImage
OutPort< CameraImage >		m_flippedImageOut

#### Detailed Description

Flip image component.

InPortからの入力画像を反転しOutPortから出力するコンポーネント。  
 反転の対象軸はRTCのコンフィギュレーション機能を使用してflipModeという名前のパラメータで指定。  
 flipModeは、反転したい方向に応じて下記のように指定してください。  
 •上下反転したい場合、0  
 •左右反転したい場合、1  
 •上下左右反転したい場合、-1

作成するRTCの入出力仕揃は以下のとおりです。  
 •InPort: キャンパスされた画像データ(CameraImage)  
 •OutPort: キャンパスされた反転画像(CameraImage)

#### Member Function Documentation

RTC::ReturnCode\_t Flip::onActivated ( RTC::UniqueId ec\_id ) [virtual]

データ領域の確保  
 ・イメージ用メモリの初期化  
 ・outPortの画面サイズの初期化

Definition at line 129 of file Flip.cpp.

RTC::ReturnCode\_t Flip::onDeactivated ( RTC::UniqueId ec\_id ) [virtual]

データ領域の解放  
 ・イメージ用メモリの解放

Definition at line 139 of file Flip.cpp.

RTC::ReturnCode\_t Flip::onExecute ( RTC::UniqueId ec\_id ) [virtual]

Flip処理  
 ・新規データのチェック  
 ・InPortの画像データを内部バッファにコピー  
 ・内部バッファの画像データを反転  
 ・反転した画像データをOutPortにコピー

Definition at line 152 of file Flip.cpp.

RTC::ReturnCode\_t Flip::onInitialize ( ) [virtual]

コンポーネント自身の各種初期化処理

Definition at line 76 of file Flip.cpp.

#### Member Data Documentation

int Flip::m\_flipMode [protected]

画像の反転方法を指定するパラメータ

- Name: flipMode flipMode
- DefaultValue: 0
- Unit: なし
- Range: -1,0,1
- Constraint: 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

Definition at line 297 of file Flip.h.

### Flip

Main Page	Classes	<b>Files</b>
File List	File Members	

### C:/forNotePC/work/source/Flip.cpp

Go to the documentation of this file.

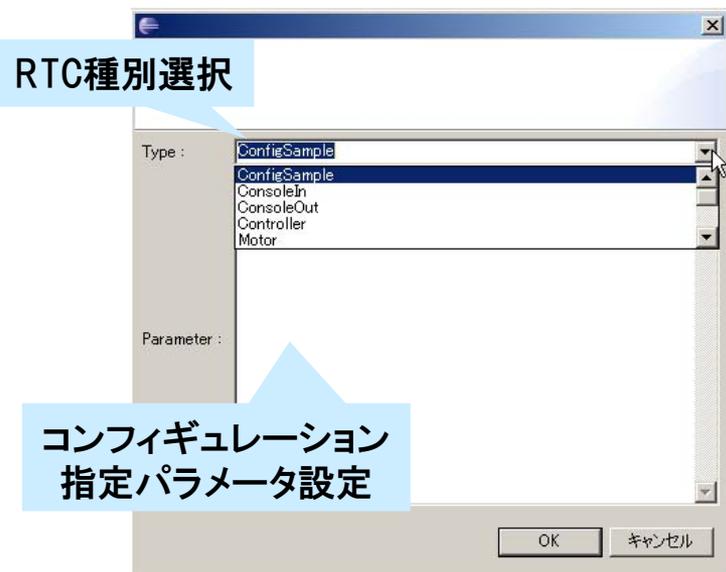
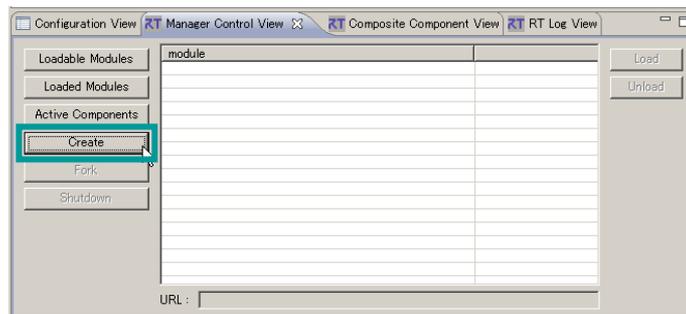
```
00001 // -*- C++ -*-
00002 #include "Flip.h"
00003
00004 // Module specification
00005 // <rtc-template block="module_spec">
00006 static const char* flip_spec[] =
00007 {
00008     "implementation_id", "Flip",
00009     "type_name", "Flip",
00010     "description", "Flip image component",
00011     "version", "1.0.0",
00012     "vendor", "AIST",
00013     "category", "Category",
00014     "activity_type", "PERIODIC",
00015     "kind", "DataFlowComponent",
00016     "max_instance", "1",
00017     "language", "C++",
00018     "lang_type", "compile",
00019     // Configuration variables
00020     "conf.default.flipMode", "0",
00021     // Widget
00022     "conf._widget_.flipMode", "radio",
00023     // Constraints
00024     "conf._constraints_.flipMode", "(-1,0,1)",
00025     ""
00026 };
00027 // </rtc-template>
00028
00029 Flip::Flip(RTC::Manager* manager)
00030 : RTC::DataFlowComponentBase(manager),
00031   m_originalImageIn("originalImage", m_originalImage),
00032   m_flippedImageOut("flippedImage", m_flippedImage)
00033 {}
00034
00035 // </rtc-template>
00036 {
00037 }
00038
00039 // <rtc-template>
00040
00041
00042
00043
00044
00045
00046
00047
00048
00049
00050
00051
00052
00053
00054
00055
00056
00057
00058
00059
00060
00061
00062
00063
00064
00065
00066
00067
00068
00069
00070
00071
00072
00073
00074
00075
00076
00077
00078
00079
00080
00081
00082
00083
00084
00085
00086
00087
00088
00089
00090
00091
00092
00093
00094
00095
00096
00097
00098
00099
00100
00101
00102
00103
00104
00105
00106
00107
00108
00109
00110
00111
00112
00113
00114
00115
00116
00117
00118
00119
00120
00121
00122
00123
00124
00125
00126
00127
00128
00129
00130
00131
00132
00133
00134
00135
00136
00137
00138
00139
00140
00141
00142
00143
00144
00145
00146
00147
00148
00149
00150
00151
00152
00153
00154
00155
00156
00157
00158
00159
00160
00161
00162
00163
00164
00165
00166
00167
00168
00169
00170
00171
00172
00173
00174
00175
00176
00177
00178
00179
00180
00181
00182
00183
00184
00185
00186
00187
00188
00189
00190
00191
00192
00193
00194
00195
00196
00197
00198
00199
00200
00201
00202
00203
00204
00205
00206
00207
00208
00209
00210
00211
00212
00213
00214
00215
00216
00217
00218
00219
00220
00221
00222
00223
00224
00225
00226
00227
00228
00229
00230
00231
00232
00233
00234
00235
00236
00237
00238
00239
00240
00241
00242
00243
00244
00245
00246
00247
00248
00249
00250
00251
00252
00253
00254
00255
00256
00257
00258
00259
00260
00261
00262
00263
00264
00265
00266
00267
00268
00269
00270
00271
00272
00273
00274
00275
00276
00277
00278
00279
00280
00281
00282
00283
00284
00285
00286
00287
00288
00289
00290
00291
00292
00293
00294
00295
00296
00297
00298
00299
00300
00301
00302
00303
00304
00305
00306
00307
00308
00309
00310
00311
00312
00313
00314
00315
00316
00317
00318
00319
00320
00321
00322
00323
00324
00325
00326
00327
00328
00329
00330
00331
00332
00333
00334
00335
00336
00337
00338
00339
00340
00341
00342
00343
00344
00345
00346
00347
00348
00349
00350
00351
00352
00353
00354
00355
00356
00357
00358
00359
00360
00361
00362
00363
00364
00365
00366
00367
00368
00369
00370
00371
00372
00373
00374
00375
00376
00377
00378
00379
00380
00381
00382
00383
00384
00385
00386
00387
00388
00389
00390
00391
00392
00393
00394
00395
00396
00397
00398
00399
00400
00401
00402
00403
00404
00405
00406
00407
00408
00409
00410
00411
00412
00413
00414
00415
00416
00417
00418
00419
00420
00421
00422
00423
00424
00425
00426
00427
00428
00429
00430
00431
00432
00433
00434
00435
00436
00437
00438
00439
00440
00441
00442
00443
00444
00445
00446
00447
00448
00449
00450
00451
00452
00453
00454
00455
00456
00457
00458
00459
00460
00461
00462
00463
00464
00465
00466
00467
00468
00469
00470
00471
00472
00473
00474
00475
00476
00477
00478
00479
00480
00481
00482
00483
00484
00485
00486
00487
00488
00489
00490
00491
00492
00493
00494
00495
00496
00497
00498
00499
00500
00501
00502
00503
00504
00505
00506
00507
00508
00509
00510
00511
00512
00513
00514
00515
00516
00517
00518
00519
00520
00521
00522
00523
00524
00525
00526
00527
00528
00529
00530
00531
00532
00533
00534
00535
00536
00537
00538
00539
00540
00541
00542
00543
00544
00545
00546
00547
00548
00549
00550
00551
00552
00553
00554
00555
00556
00557
00558
00559
00560
00561
00562
00563
00564
00565
00566
00567
00568
00569
00570
00571
00572
00573
00574
00575
00576
00577
00578
00579
00580
00581
00582
00583
00584
00585
00586
00587
00588
00589
00590
00591
00592
00593
00594
00595
00596
00597
00598
00599
00600
00601
00602
00603
00604
00605
00606
00607
00608
00609
00610
00611
00612
00613
00614
00615
00616
00617
00618
00619
00620
00621
00622
00623
00624
00625
00626
00627
00628
00629
00630
00631
00632
00633
00634
00635
00636
00637
00638
00639
00640
00641
00642
00643
00644
00645
00646
00647
00648
00649
00650
00651
00652
00653
00654
00655
00656
00657
00658
00659
00660
00661
00662
00663
00664
00665
00666
00667
00668
00669
00670
00671
00672
00673
00674
00675
00676
00677
00678
00679
00680
00681
00682
00683
00684
00685
00686
00687
00688
00689
00690
00691
00692
00693
00694
00695
00696
00697
00698
00699
00700
00701
00702
00703
00704
00705
00706
00707
00708
00709
00710
00711
00712
00713
00714
00715
00716
00717
00718
00719
00720
00721
00722
00723
00724
00725
00726
00727
00728
00729
00730
00731
00732
00733
00734
00735
00736
00737
00738
00739
00740
00741
00742
00743
00744
00745
00746
00747
00748
00749
00750
00751
00752
00753
00754
00755
00756
00757
00758
00759
00760
00761
00762
00763
00764
00765
00766
00767
00768
00769
00770
00771
00772
00773
00774
00775
00776
00777
00778
00779
00780
00781
00782
00783
00784
00785
00786
00787
00788
00789
00790
00791
00792
00793
00794
00795
00796
00797
00798
00799
00800
00801
00802
00803
00804
00805
00806
00807
00808
00809
00810
00811
00812
00813
00814
00815
00816
00817
00818
00819
00820
00821
00822
00823
00824
00825
00826
00827
00828
00829
00830
00831
00832
00833
00834
00835
00836
00837
00838
00839
00840
00841
00842
00843
00844
00845
00846
00847
00848
00849
00850
00851
00852
00853
00854
00855
00856
00857
00858
00859
00860
00861
00862
00863
00864
00865
00866
00867
00868
00869
00870
00871
00872
00873
00874
00875
00876
00877
00878
00879
00880
00881
00882
00883
00884
00885
00886
00887
00888
00889
00890
00891
00892
00893
00894
00895
00896
00897
00898
00899
00900
00901
00902
00903
00904
00905
00906
00907
00908
00909
00910
00911
00912
00913
00914
00915
00916
00917
00918
00919
00920
00921
00922
00923
00924
00925
00926
00927
00928
00929
00930
00931
00932
00933
00934
00935
00936
00937
00938
00939
00940
00941
00942
00943
00944
00945
00946
00947
00948
00949
00950
00951
00952
00953
00954
00955
00956
00957
00958
00959
00960
00961
00962
00963
00964
00965
00966
00967
00968
00969
00970
00971
00972
00973
00974
00975
00976
00977
00978
00979
00980
00981
00982
00983
00984
00985
00986
00987
00988
00989
00990
00991
00992
00993
00994
00995
00996
00997
00998
00999
01000
```

# RTSystemEditorの補足

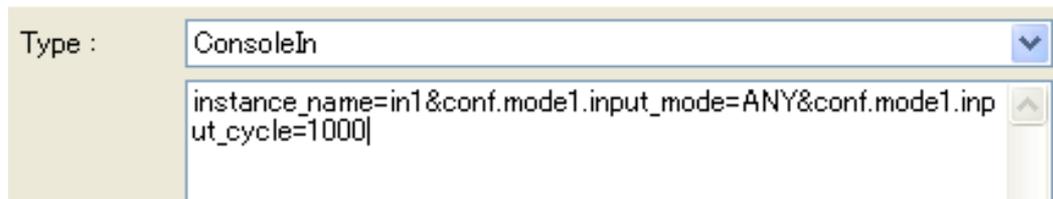


# マネージャビュー

## ■ RTコンポーネントの新規インスタンスの生成



- コンフィギュレーション指定パラメータ
  - `conf. [ConfigSet名]. [Configパラメータ名]=[設定値]`の形式にてConfigurationSetの値も設定可能



# オフラインエディタ

- RTコンポーネントの仕様を用いてRTシステムを構築
  - 実際のRTコンポーネントが動作している必要はない

コンフィギュレーションビュー

ComponentName	ConfigurationSet	active	config	name	Value
ImageProcess_1					

オフライン・システムエディタ

プロパティビュー

リポジトリビュー

# 設定画面

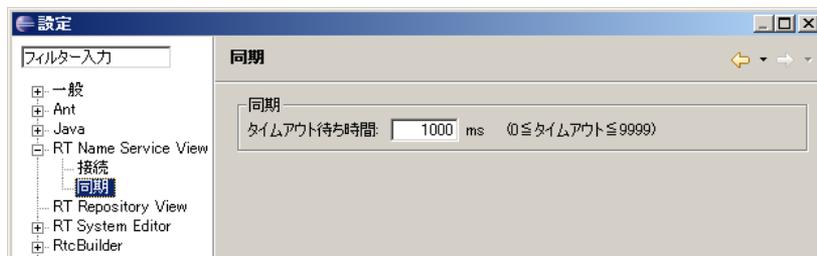
## ■ 「RT Name Service View」－「接続」【接続周期】

- ネームサービスビューが、ネームサーバに情報を問い合わせる周期



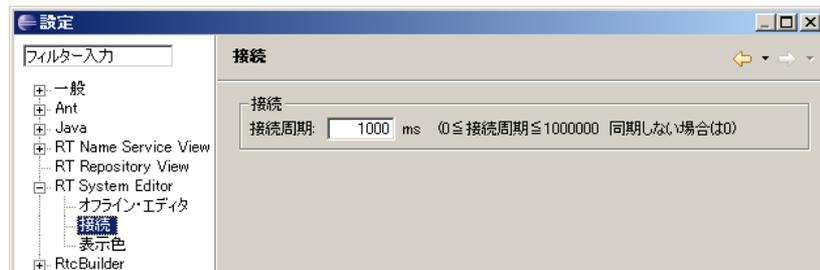
## ■ 「RT Name Service View」－「同期」【タイムアウト待ち時間】

- ネームサービスビューが、リモートオブジェクトのレスポンスを待つ時間



## ■ 「RT System Editor」－「接続」【接続周期】

- システムエディタが、ネームサーバに情報を問い合わせる周期



**【接続周期】をゼロに設定すると  
ネームサーバとの同期を行わない**

# その他のツールのご紹介



## ■ UMLを利用した設計情報から，RTCの雛形コードを生成

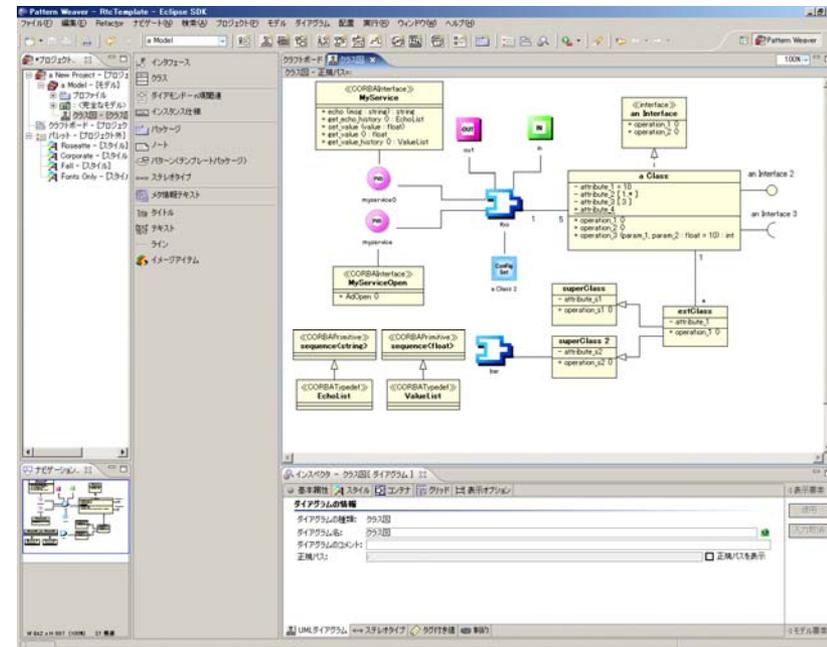
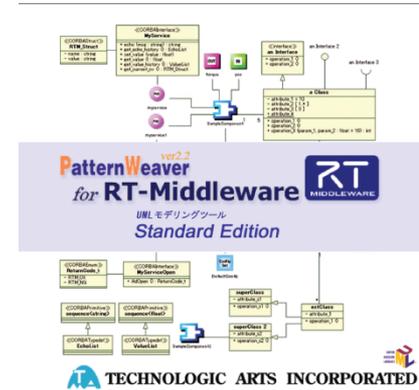
- OpenRTM-aist-1.0.0-RELEASEに対応
  - C++, Java, Python
- UMLを用いたコンポーネント設計/システム設計支援
- システム設計情報を基にした各種コードの自動生成
  - RTコンポーネント(C++, Java, Python)雛形コード
  - CORBA IDLコード
  - 関連クラス(C++, Java, Python)雛形コード
- RTC, RTシステム開発支援ツールのご提供
  - RTシステムローダー
  - プラグアンドプレイ設定ツール
  - 状態遷移設定ツール

※PatternWeaverサイト <http://pw.tech-arts.co.jp/>

お問い合わせ先 [pw@tech-arts.co.jp](mailto:pw@tech-arts.co.jp)



株式会社テクノロジックアート  
TECHNOLOGIC ARTS INCORPORATED



# システム構築ツールセット

## ■ RTコンポーネント, RTシステムを制御するためのPythonライブラリ群

- 各種操作の自動化, プログラムからの利用を可能に

### ■ rtctree

- 簡単なAPIでRTコンポーネントを管理するためのライブラリ
- CORBAのAPIを知らなくても, 他プログラムからRTCを管理可能
- RTCのactivate/deactivate, ポート間の接続を行うこと等が可能

### ■ rtcshell

- ネームサーバに登録されているRTコンポーネントをシェルから管理するツール
- RTCのactivate/deactivate, ポート間の接続を行うこと等が可能
- リソースの少ないシステム, GUIが利用できない環境でも利用可能

### ■ rtsprofile

- RTシステム仕様(RTSProfile)のインタフェースライブラリ
- システムの復元や管理が可能
- XML, YAMLを利用可能

# システム構築ツールセット

## ■ rtsshell

### ■ rtresurrect

- RTSPProfileに保存されたRTシステムを復元するためのツール
- 全ての接続, コンフィギュレーション設定を復元し, アクティブなコンフィギュレーションを設定可能

### ■ rtteardown

- RTSPProfileに保存されたRTシステムをシャットダウンするツール
- 全ての接続を切断

### ■ rtcryo

- 既存のRTシステムをRTSPProfileのファイルとして保存するツール
- 全てのコンポーネント, コンポーネント間の接続, 構成設定をファイルに保存

### ■ rtstart/rtstop

- コマンドラインでRTシステムを起動/停止するツール
- RTSPProfileに記載された実行順, 実行条件に従ってRTコンポーネントを制御

# RTミドルウェア講習会

