

RTミドルウェア講習会

日時:2012年7月19日(金) 13:00~17:00
場所:名城大学 天白キャンパス 研究実験棟Ⅱ
メカトロニクス工学科会議室



OpenRT Platform



- **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

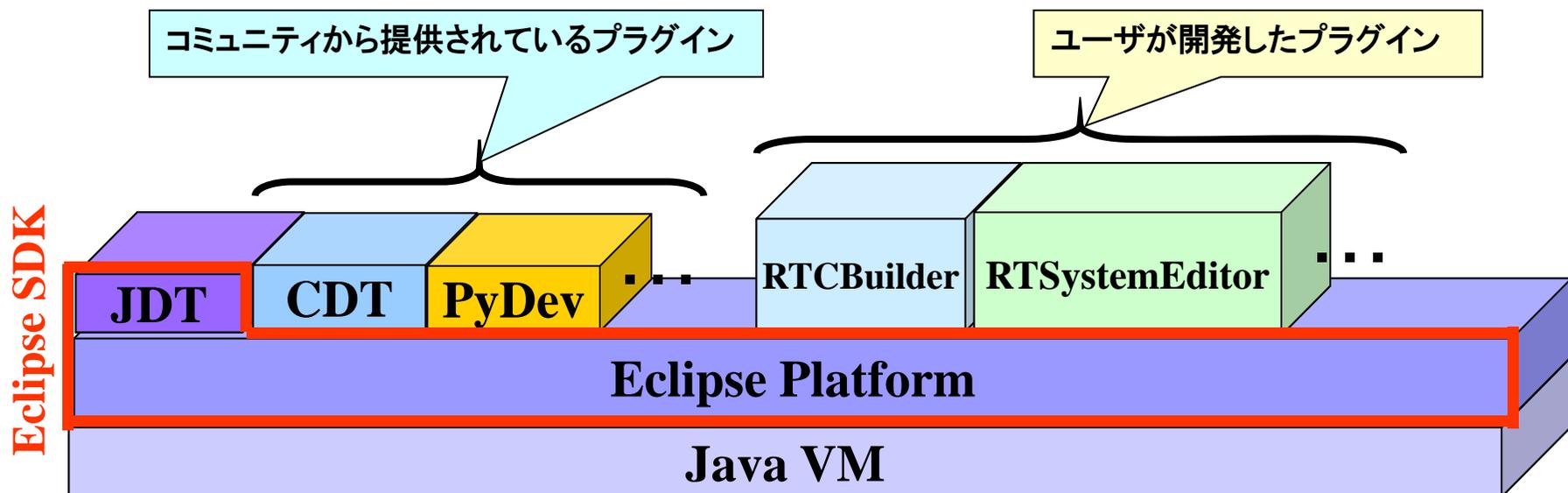
- **RTCビルダ**
 - **RTCデバッガ**
 - **RTシステムエディタ**
 - **ロボット設計支援ツール**
 - **シミュレータ**
 - **動作設計ツール**
 - **シナリオ作成ツール**
- など



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. The main content area is titled "OpenRTM Eclipse tools 1.1.0-RC2" and includes a "Table of contents" section with links to "全部入りパッケージ", "バイナリ", "Eclipse/JDK/JRE等", and "過去のバージョン". Below this is a section for "全部入りパッケージ" with a table of download links. The table has columns for the package name, the download link, and the date. The date shown is 2011.07.22.

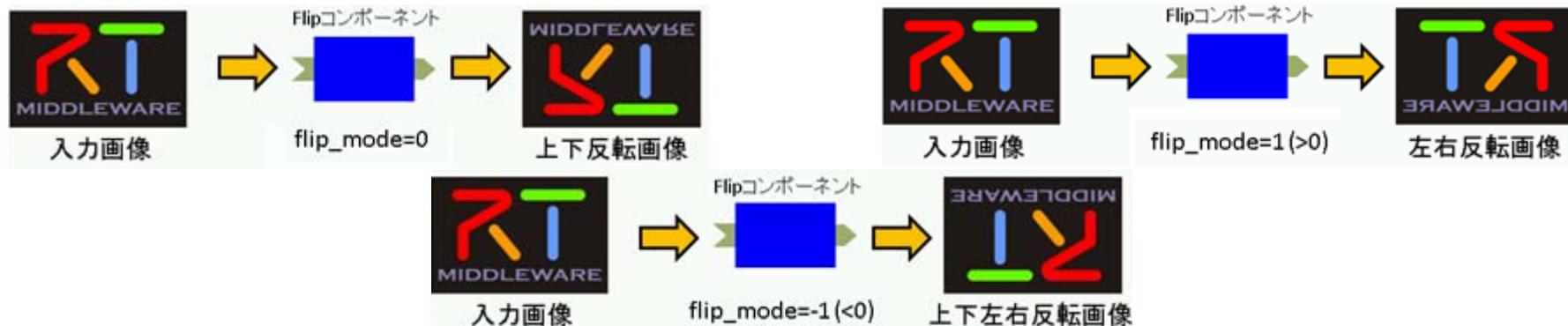
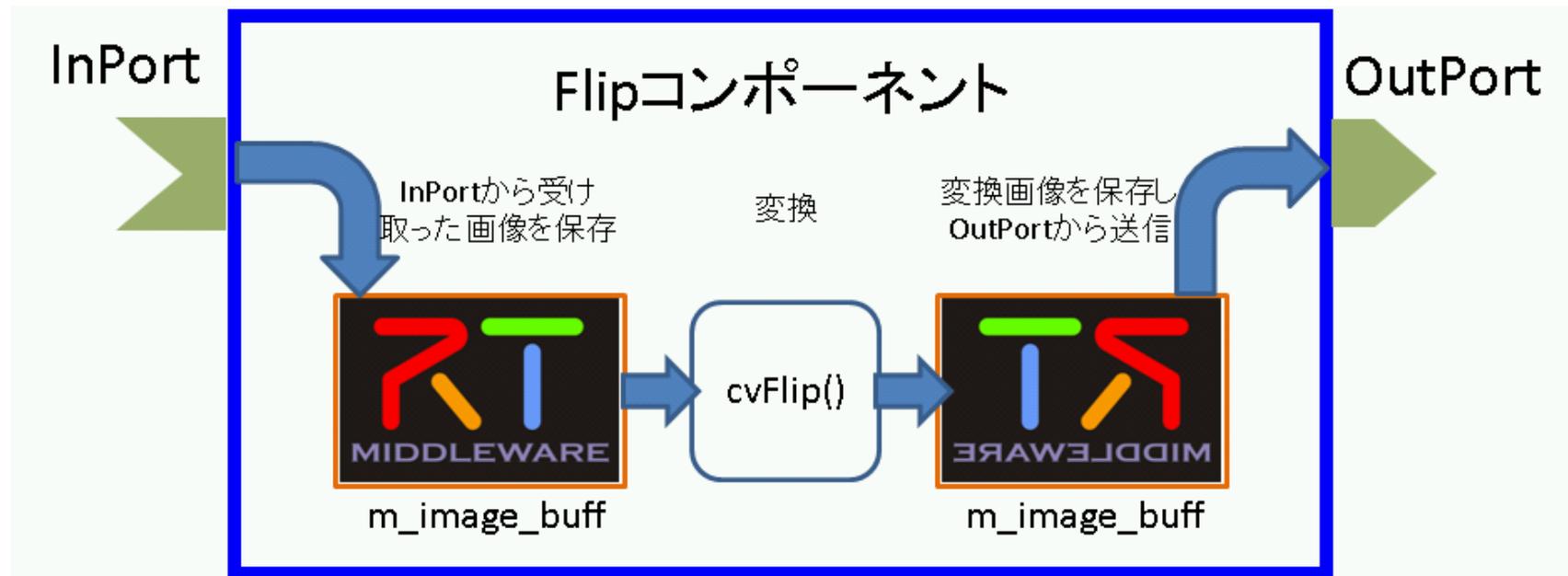
Eclipse-3.4.2 [Ganymede SR2]		
Eclipse3.4.2+RTSE+RTCB Windows用全部入り	eclipse342_rtmtools110-rc2_win32_ja.zip MD5:2e6f9fa3e370b6e7ac1f9340d36c7abf	2011.07.22

• 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 1.9.0.8+nobinonly-0ubuntu2 XIII + XPCOM application runner
```

Flipコンポーネントについて

- 入力画像を反転して出力するコンポーネント
 - OpenCVのcvFlip関数を利用

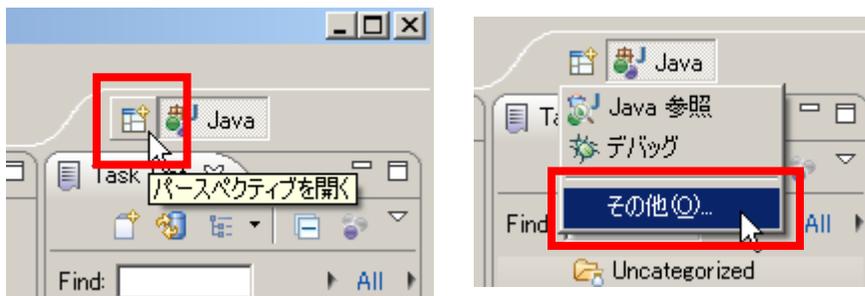


システム構築支援ツール RTSystemEditorについて

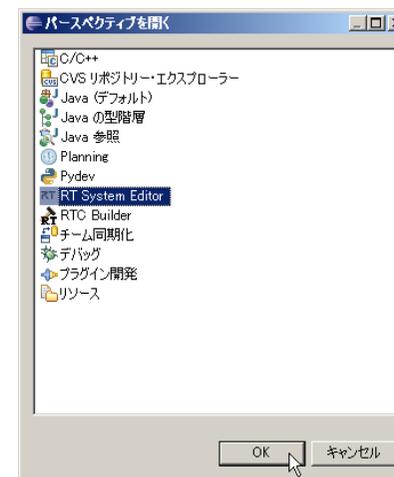


■ パースペクティブの切り替え

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

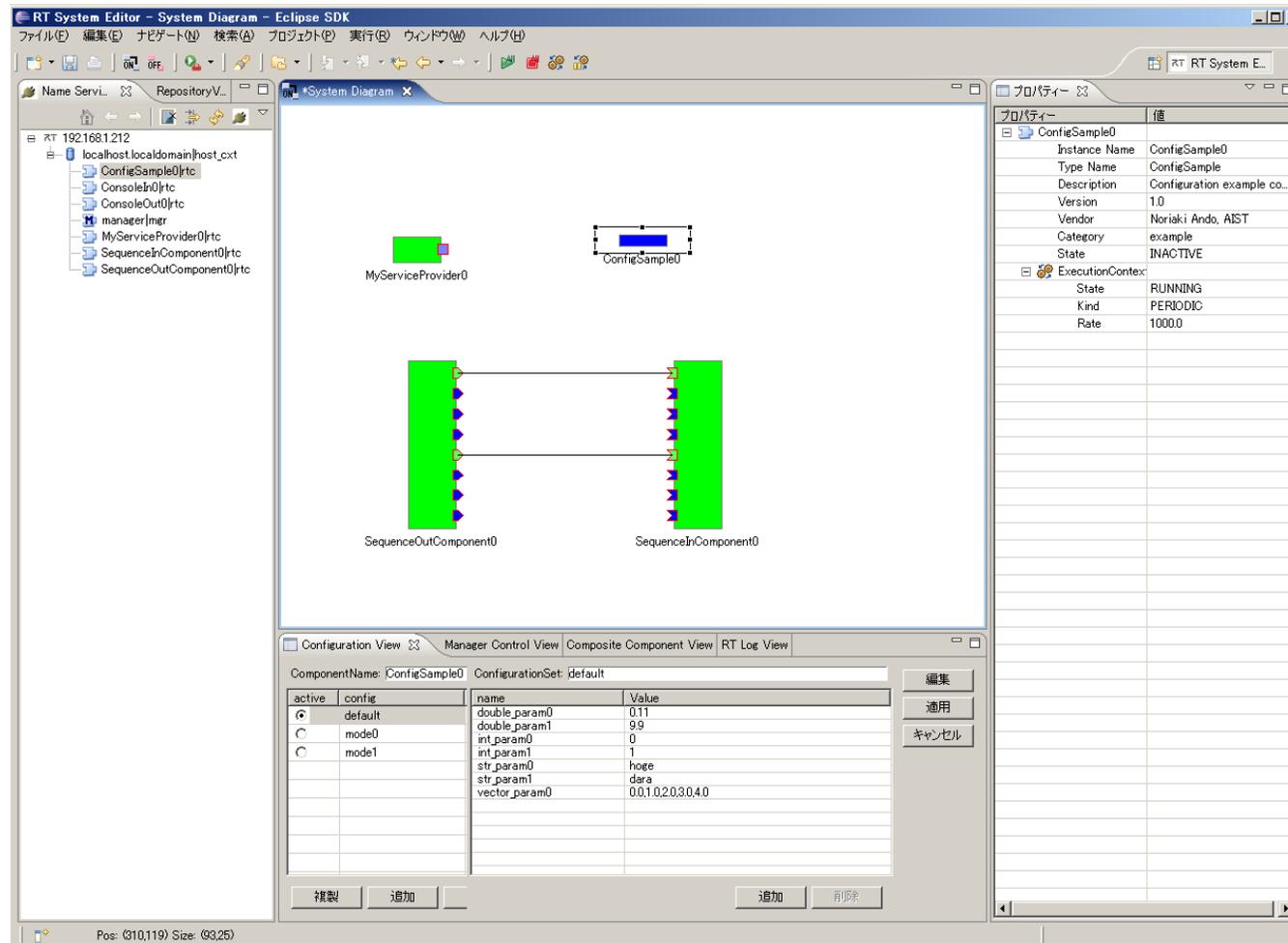


※パースペクティブ
Eclipse上でツールの構成を管理する単位
メニュー、ツールバー、エディタ、ビューなど
使用目的に応じて組み合わせる
独自の構成を登録することも可能

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



画面構成

The main screenshot shows the RT System Editor interface. A blue dashed box highlights the central area labeled **システムエディタ** (System Editor), which contains a diagram of components like MyServiceProvider0, ConfigSample0, and SequenceInComponent0. A pink box on the left points to the **ネームサービスビュー** (Name Service View) showing a tree structure of components. An orange box on the right points to the **プロパティビュー** (Property View) showing a table of configuration parameters for ConfigSample0. A red box at the bottom points to the **コンフィギュレーションビュー** (Configuration View) showing a table of configuration sets.

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

This section shows four sub-views of the RT System Editor. A green box on the left points to the **マネージャビュー** (Manager View) showing module management options. A yellow box at the bottom points to the **複合コンポーネントビュー** (Composite Component View) showing a table of components and their ports. A red box in the center points to the **実行コンテキストビュー** (Execution Context View) showing execution context details for ConsoleOut0. A blue box on the right points to the **ログビュー** (Log View) showing a table of system logs.

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

カメラ関連コンポーネントの起動

- Naming Serviceの起動

- [スタート]メニューから

- [プログラム]→[OpenRTM-aist 1.1]→[C++]→[tools]→[Start Naming Service]

- CameraViewerCompの起動

- [スタート]メニューから起動

- [プログラム]→[OpenRTM-aist 1.1]→[C++]→[components]

- [opencv-rtcs]→ [CameraViewerComp.exe]

- DirectShowCamCompの起動

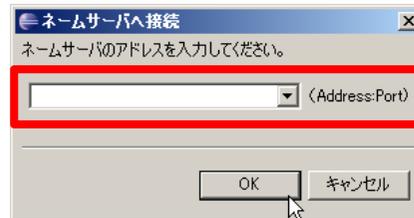
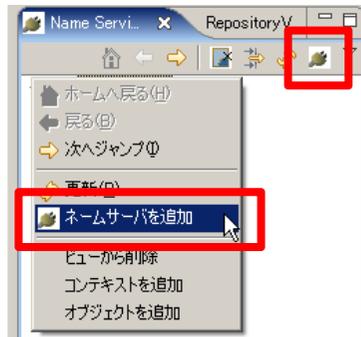
- [スタート]メニューから起動

- [プログラム]→[OpenRTM-aist 1.1]→[C++]→[components]

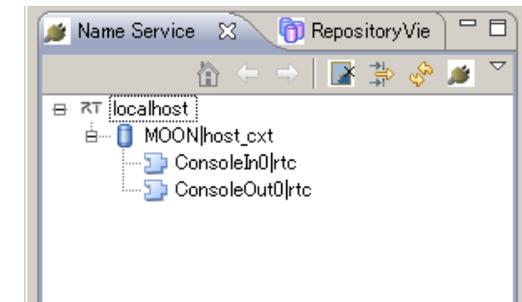
- [opencv-rtcs]→ [DirectShowCamComp.exe]

RTシステム構築の基本操作

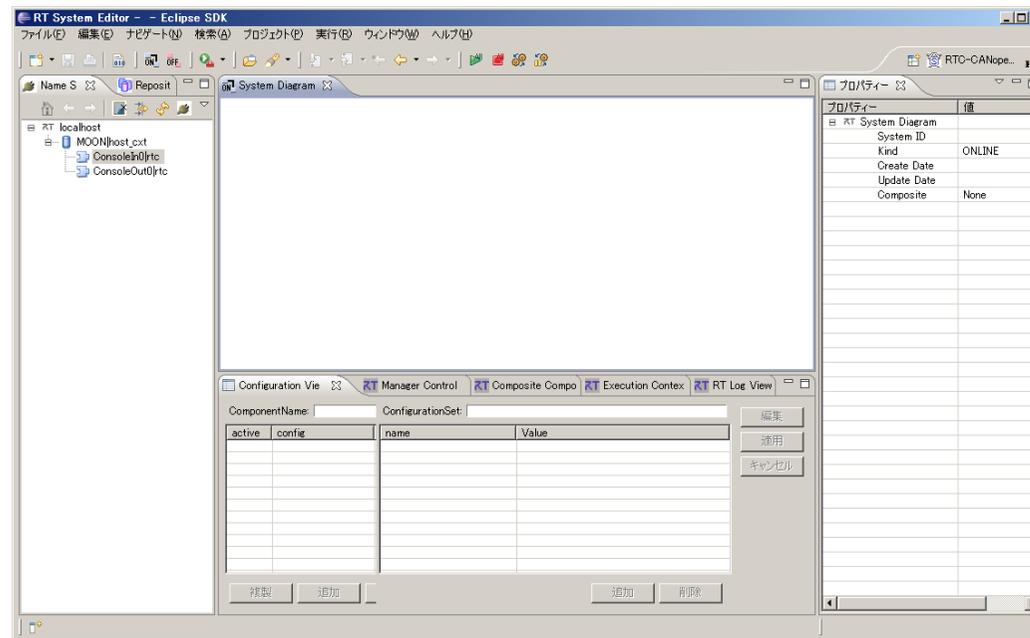
■ ネームサービスへ接続



※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

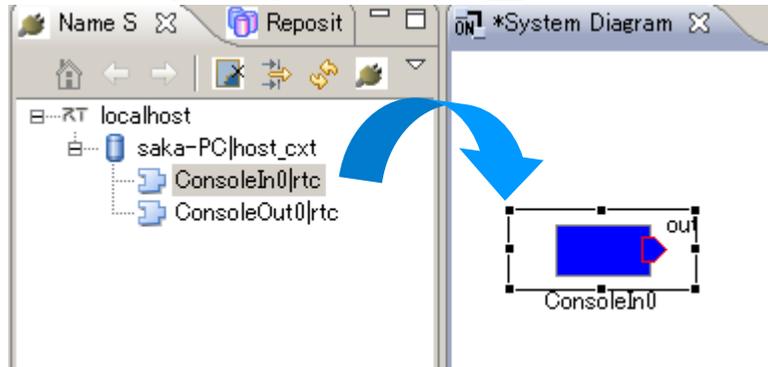


■ システムエディタの起動

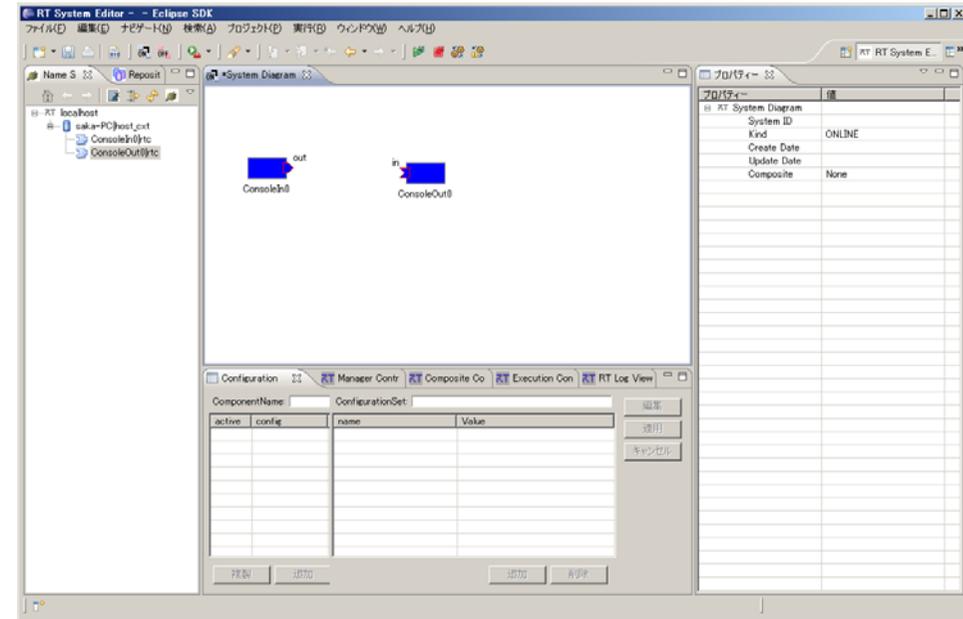


RTシステム構築の基本操作

■ RTコンポーネントの配置

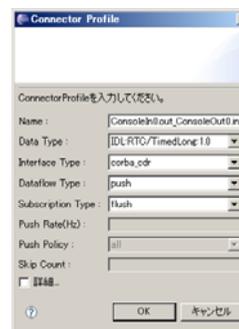
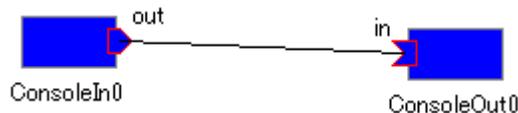


※ネームサービスビューから対象コンポーネントをドラッグアンドドロップ

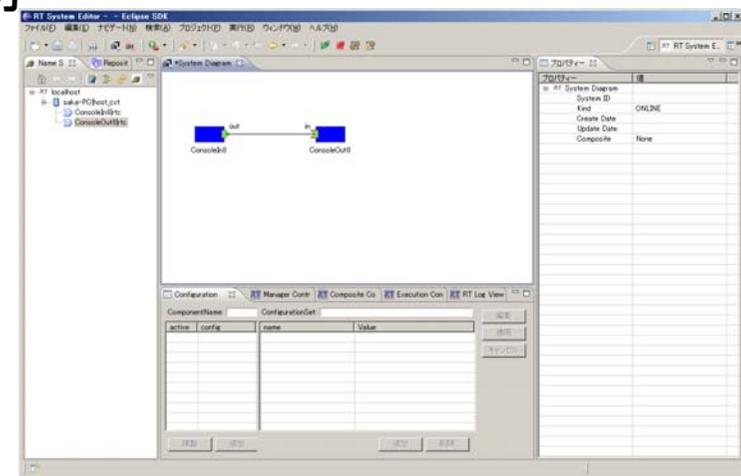


■ ポートの接続

①接続元のポートから接続先の②接続プロファイルを入力
ポートまでドラッグ



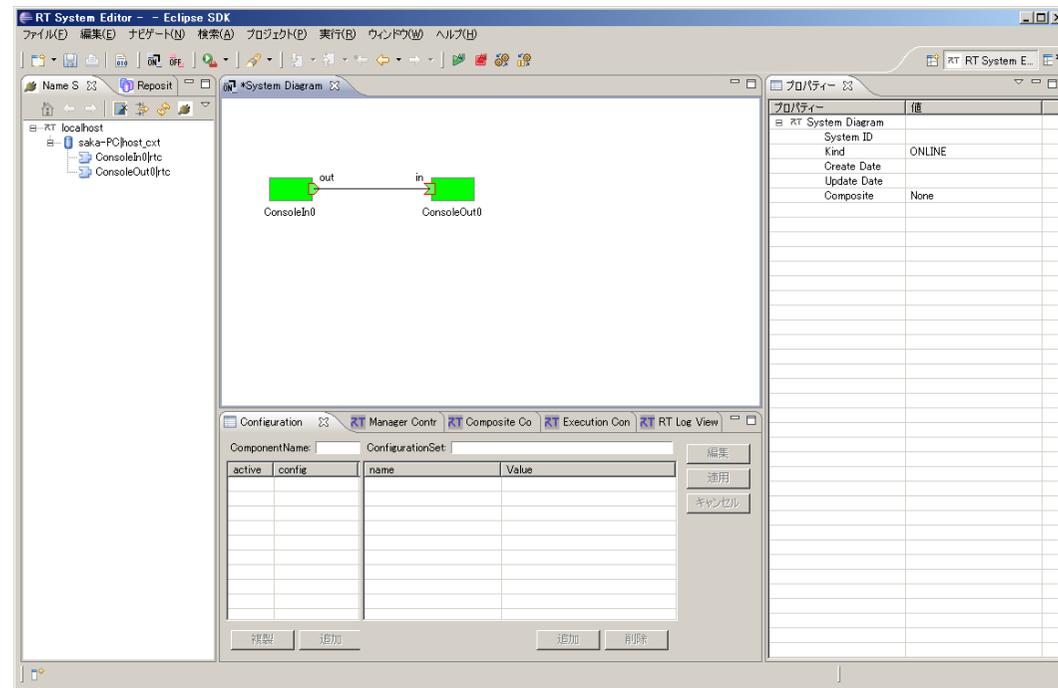
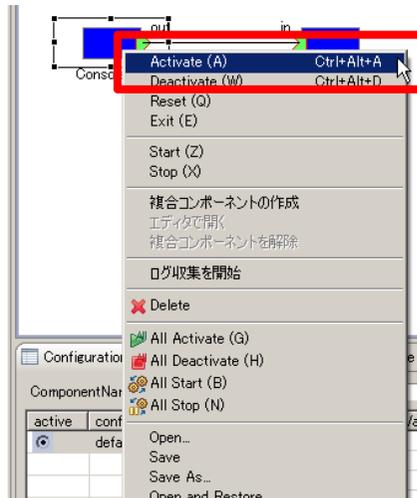
※ポートのプロパティが異なる場合など、
接続不可能なポートの場合にはアイコンが変化



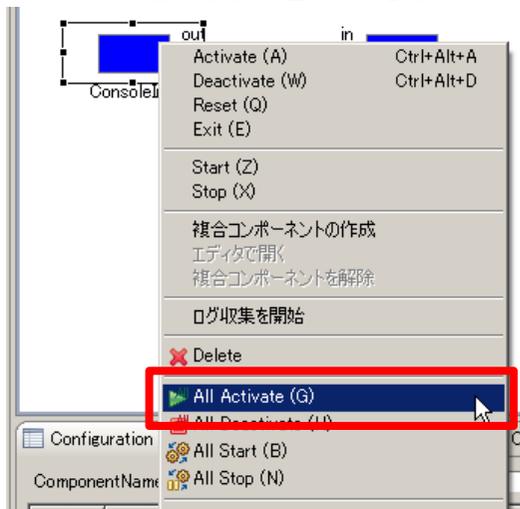
RTシステム構築の基本操作

■ コンポーネントの起動

※各RTC単位で起動する場合



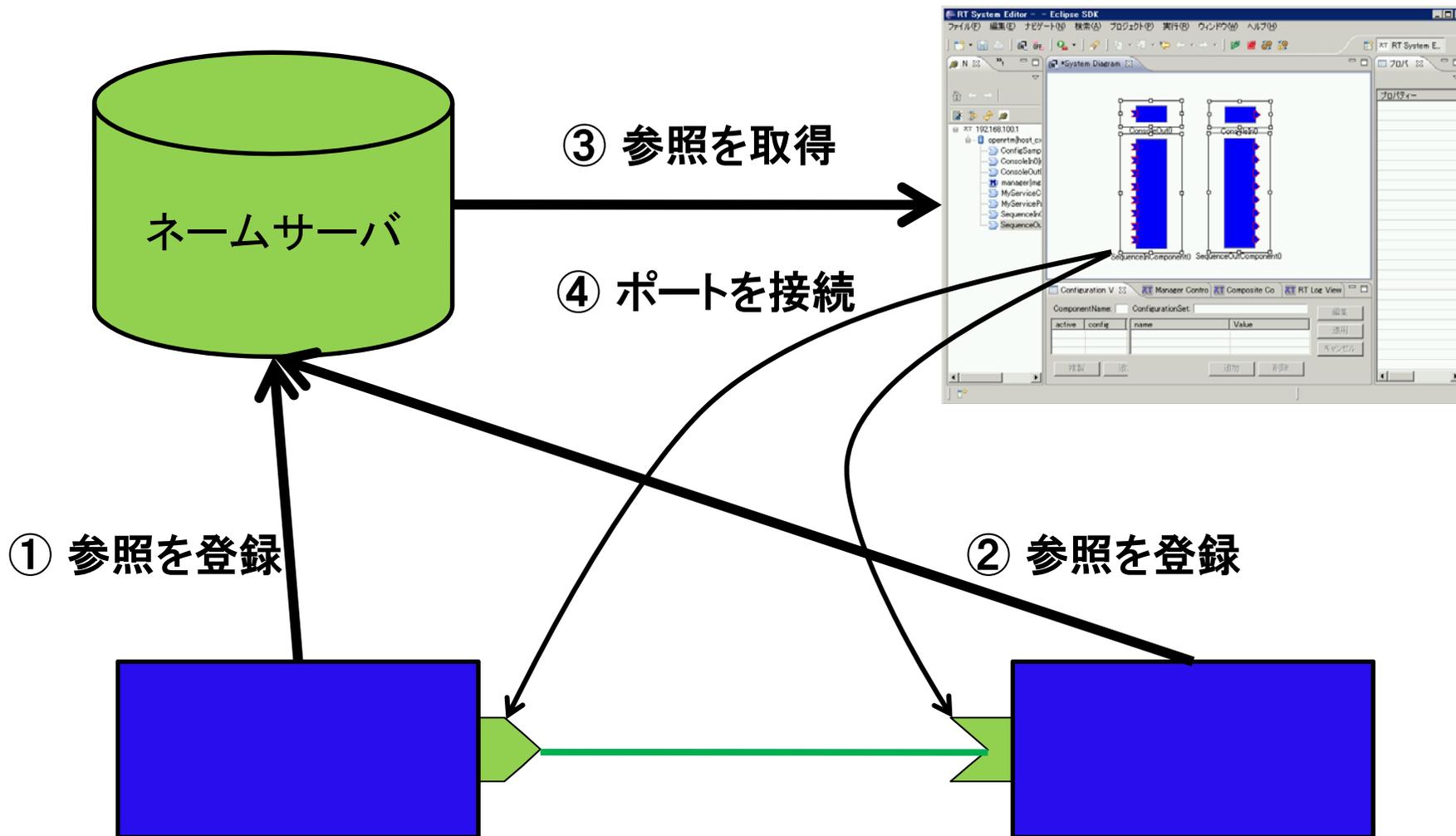
※全てのRTCを一括で起動する場合



※停止はDeactivateを実行

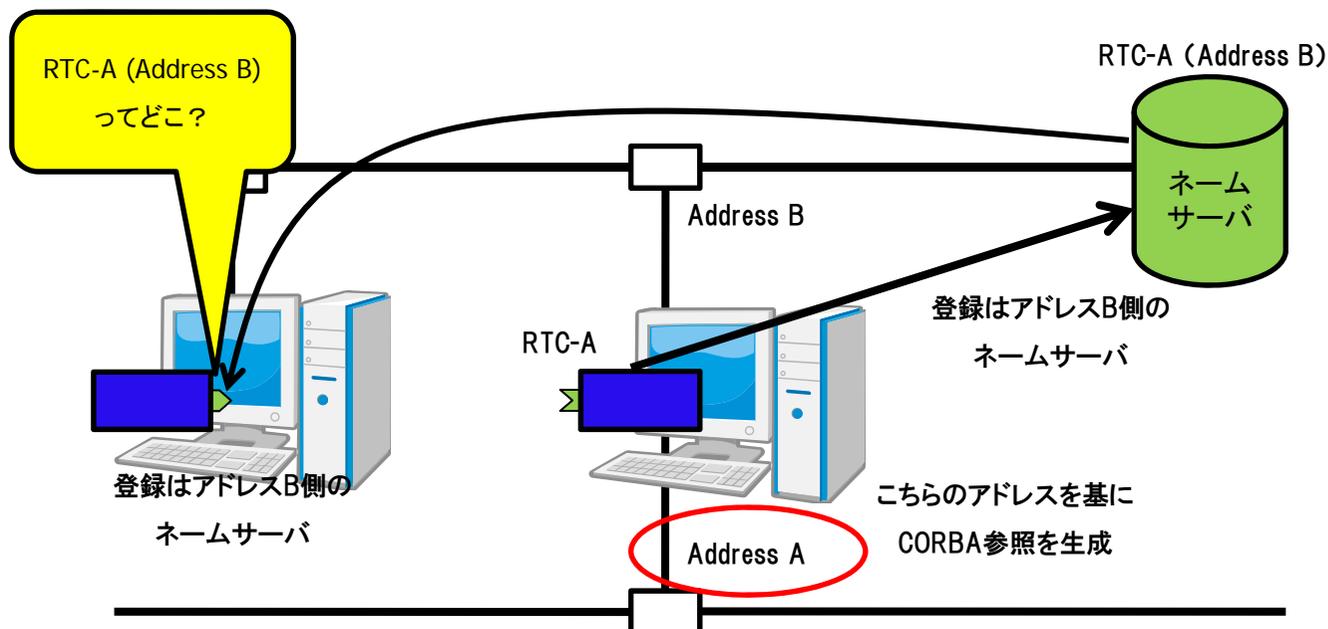
※RTC間の接続を切る場合には接続線をDelete
もしくは、右クリックメニューから「Delete」を選択

RTコンポーネントの動作シーケンス



ネームサービスに接続できない場合

■ 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:

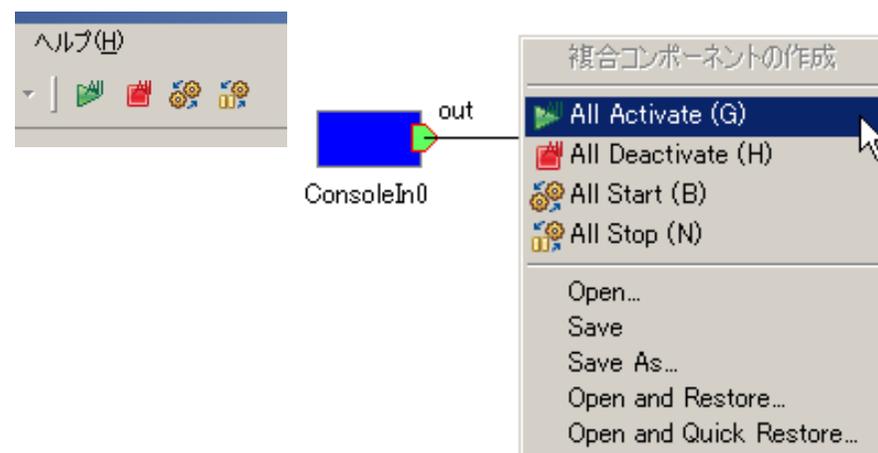
RTコンポーネントの動作

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

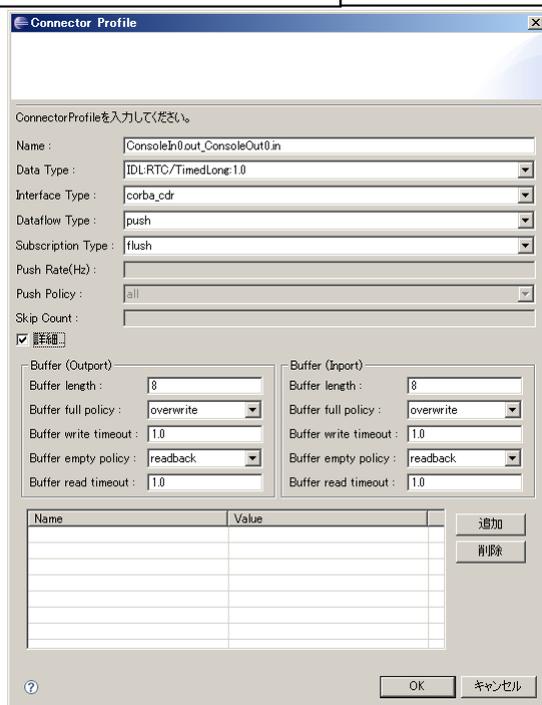
接続プロファイル(DataPort)について

項目	設定内容
Name	接続の名称
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

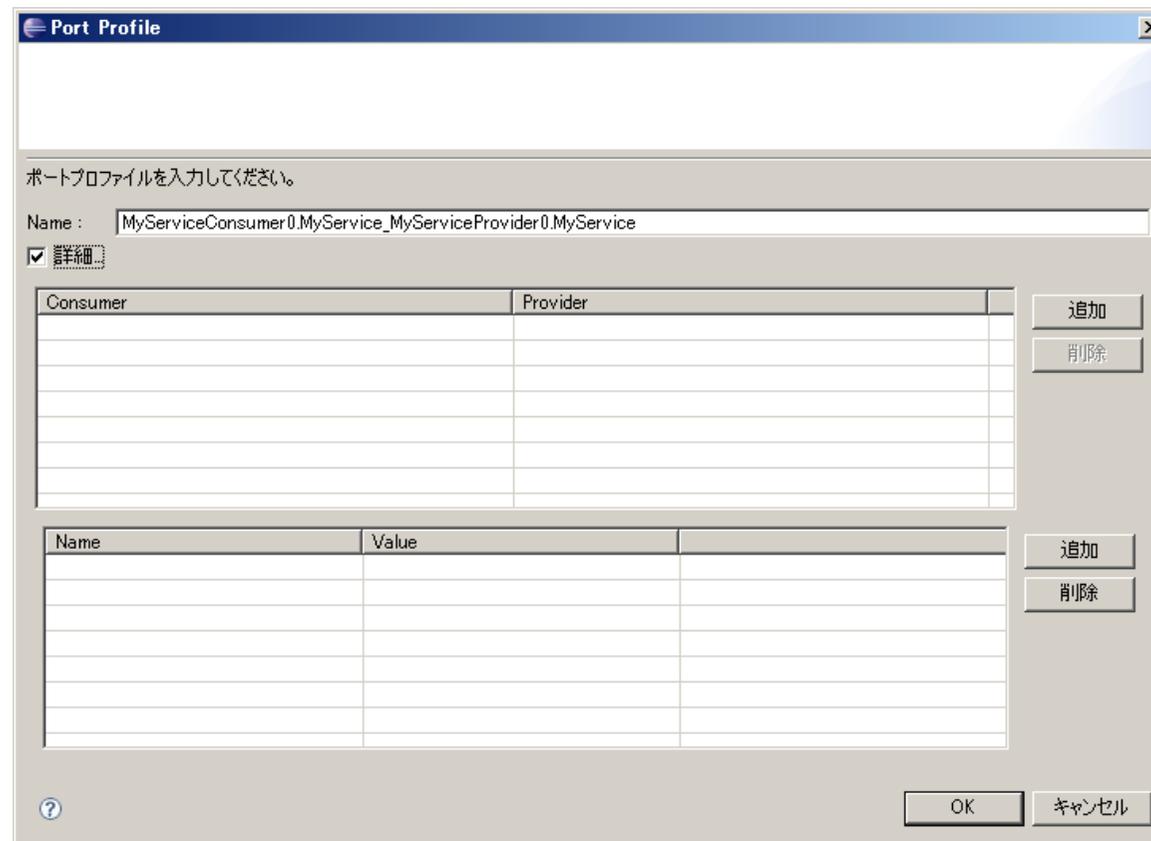
■ Buffer Policy

- 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



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

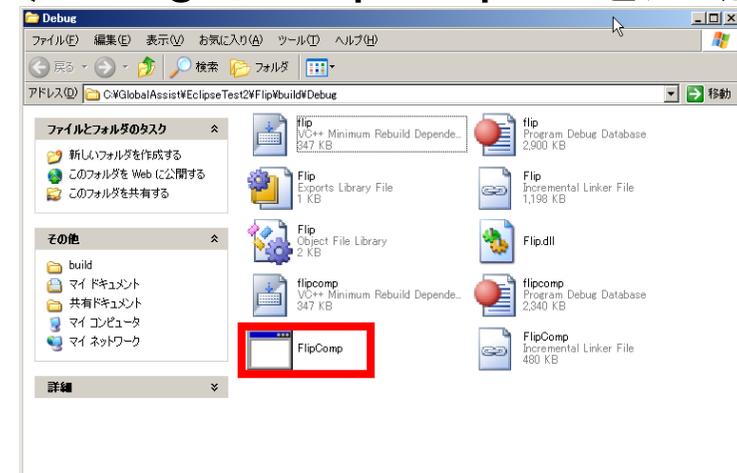
画像処理関連コンポーネントの起動

■ 画像処理用コンポーネントの起動

■ Flipコンポーネントの起動

先ほどコンパイルしたコンポーネントの起動

binaryにて指定したディレクトリ以下のSrc/Debug内のFlipComp.exeを起動



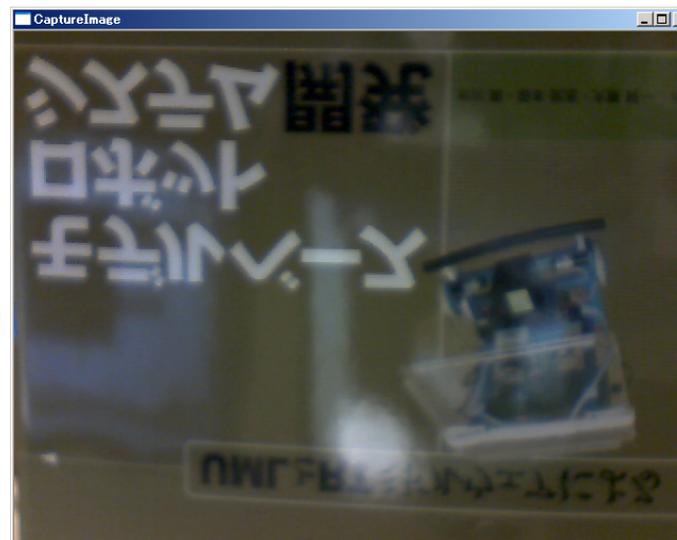
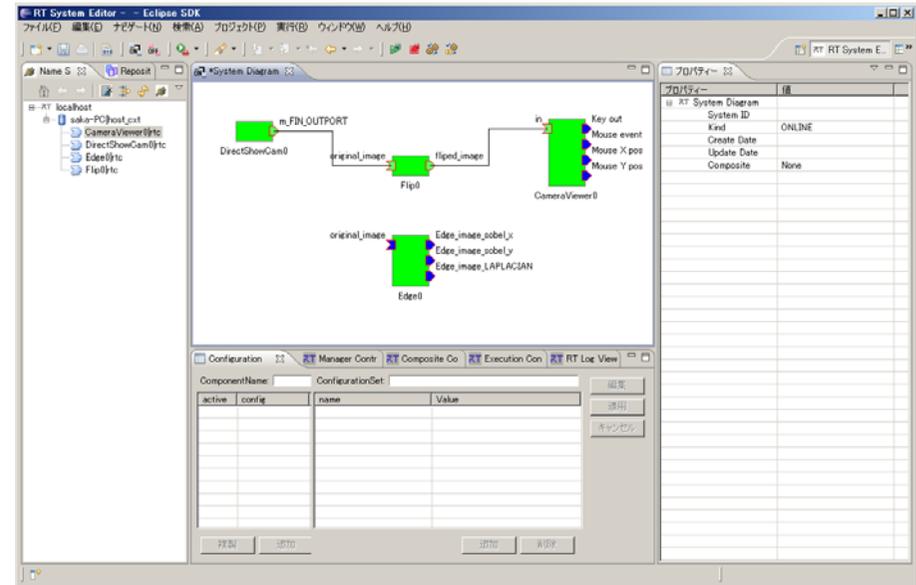
([プログラム]→[OpenRTM-aist 1.1]→[C++]→[components]
→[opencv-rtcs]→ [FlipComp.exe])

■ [スタート]メニューから起動

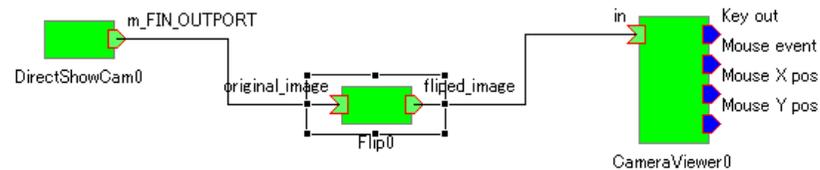
[プログラム]→[OpenRTM-aist 1.1]→[C++]→[components]
→[opencv-rtcs]→ [EdgeComp.exe]

システムの構成

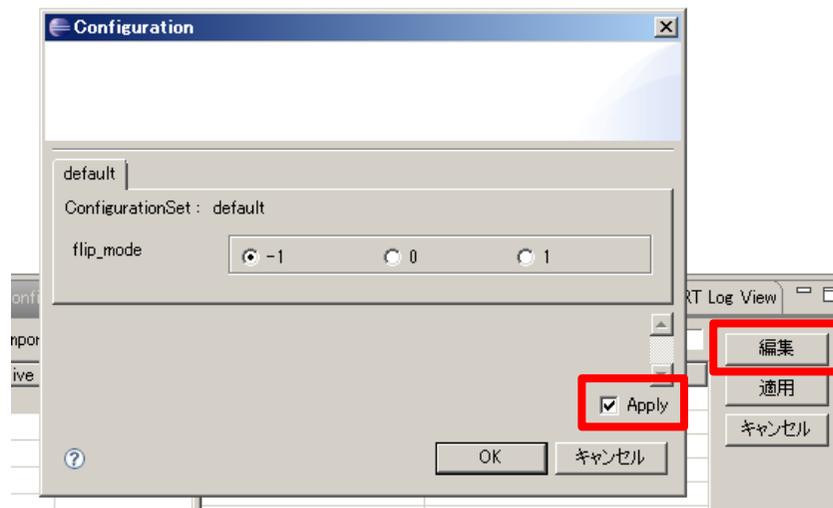
- Flip側との接続
 - DirectShowCam → Flip
 - CameraViewerと接続
(接続プロファイルはデフォルト設定)
 - AllActivateを実行



コンフィギュレーションの変更



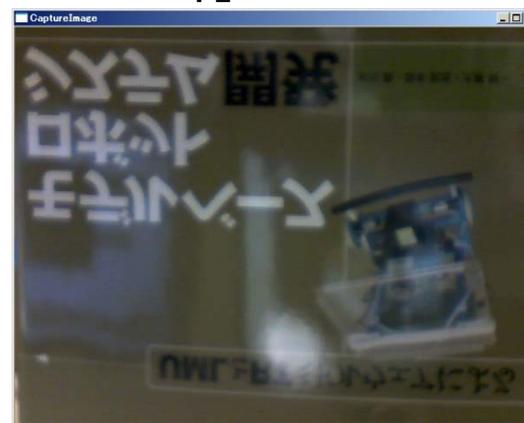
- ConfigurationViewの「編集」
- 表示されたダイアログ内で「flip_mode」の値を変更
- 「Apply」のチェックボックス



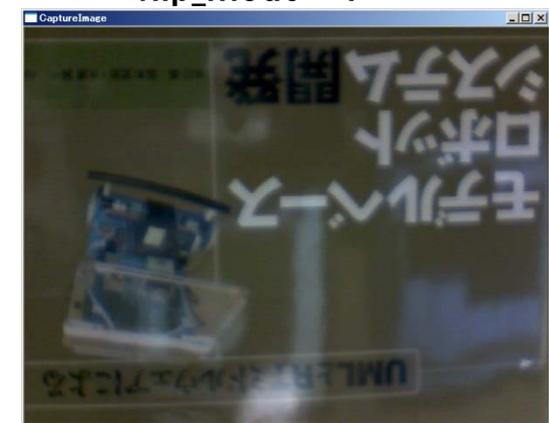
flip_mode=1



flip_mode=0

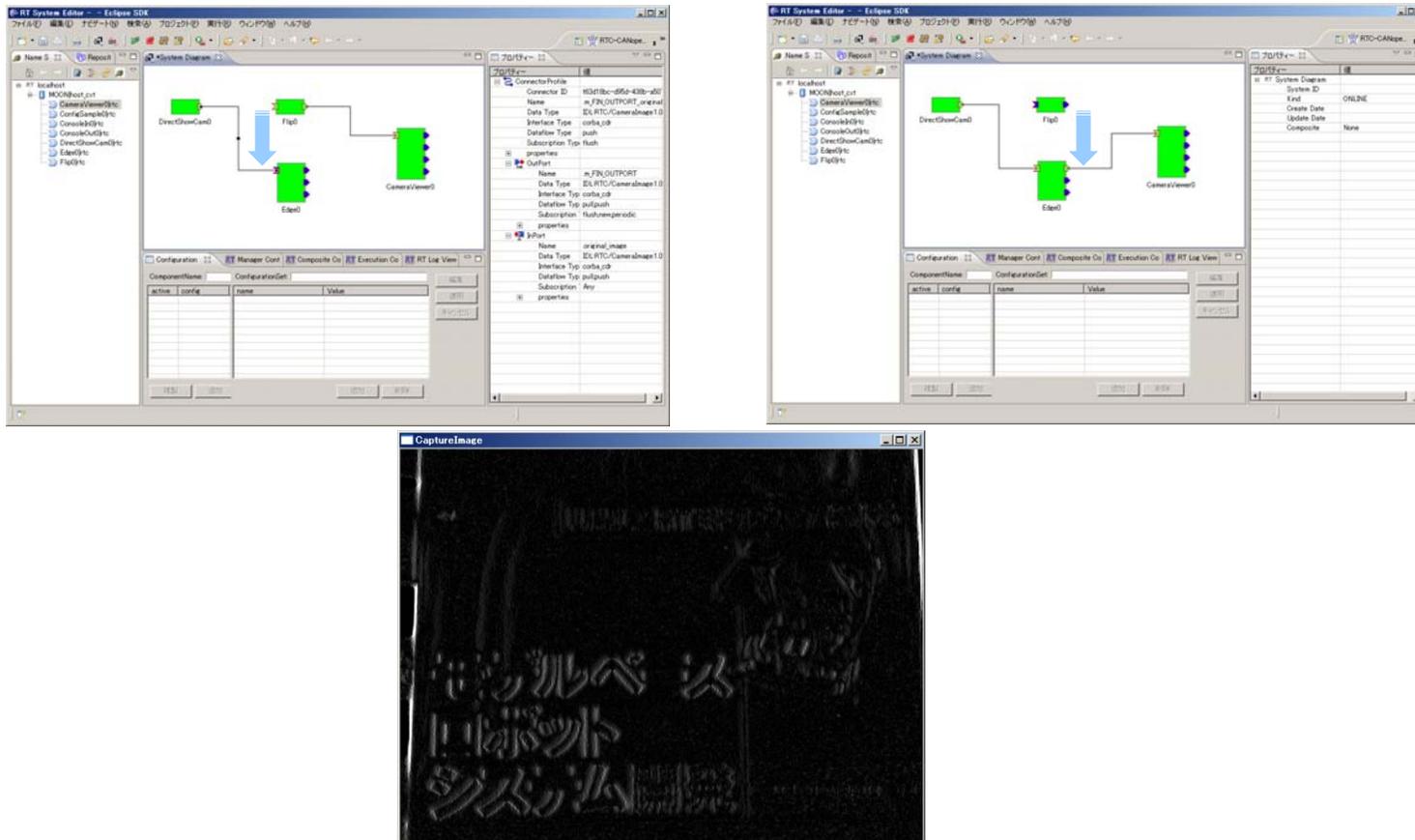


flip_mode=-1



システム構成の変更

- Edge側への差し替え
 - Flipに繋がっている接続線を選択
 - Flip側のPort部分に表示されているハンドルをEdge側のPortに繋ぎ替え
 - 接続プロファイルはデフォルト設定のまま



コンポーネント開発ツール RTCBuilderについて



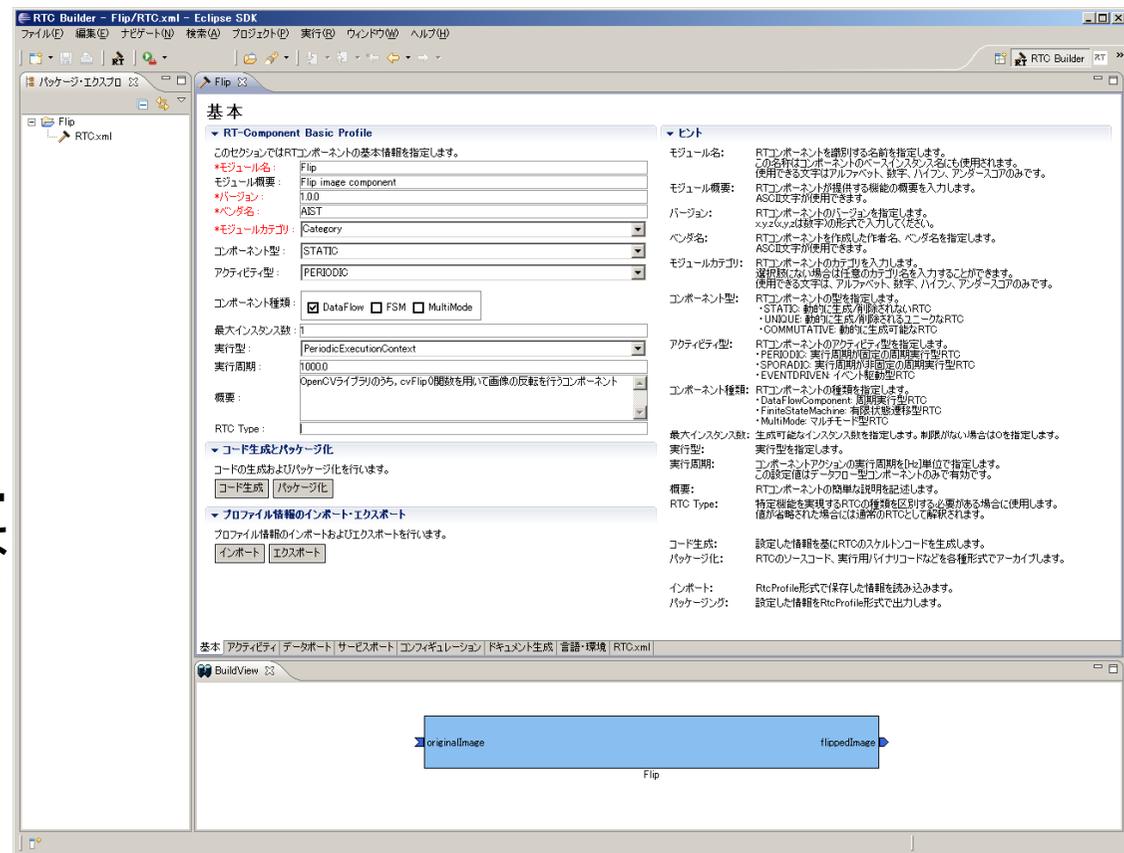
RTCBuilder概要

■ 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 interface for RTC Builder. The main window is titled "RTC Builder - Flip/RTC.xml - Eclipse SDK". The interface is divided into several panes:

- Package Explorer (パッケージ・エクスプローラ):** Located on the left, showing the project structure with "Flip" and "RTC.xml".
- RTC Profile Editor (RTCプロファイルエディタ):** The central pane, divided into "基本" (Basic) and "ヒント" (Hints) sections. The "基本" section contains fields for "RT-Component Basic Profile" such as Module Name (Flip), Version (1.0.0), Vendor (AIST), Component Type (STATIC), and Activity Type (PERIODIC). The "ヒント" section provides detailed instructions for each field.
- Build View (ビルドビュー):** Located at the bottom, showing a diagram of the "Flip" component with two data flows: "originalImage" and "flippedImage".

Callout boxes highlight these key areas: "パッケージ・エクスプローラ" (Package Explorer) in purple, "RTCプロファイルエディタ" (RTC Profile Editor) in blue, "ヒント" (Hints) in green, and "ビルドビュー" (Build View) in pink.

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

- 初期画面のクローズ
 - 初回起動時のみ

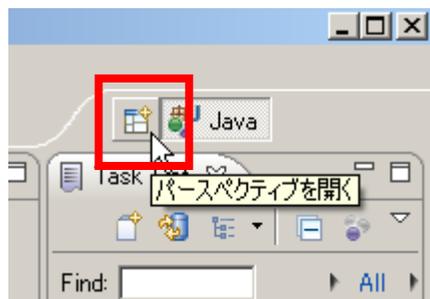


※パースペクティブ

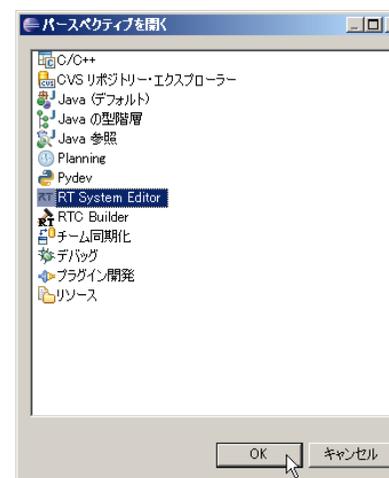
Eclipse上でツールの構成を管理する単位
メニュー、ツールバー、エディタ、ビューなど
使用目的に応じて組み合わせる
独自の構成を登録することも可能

■ 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

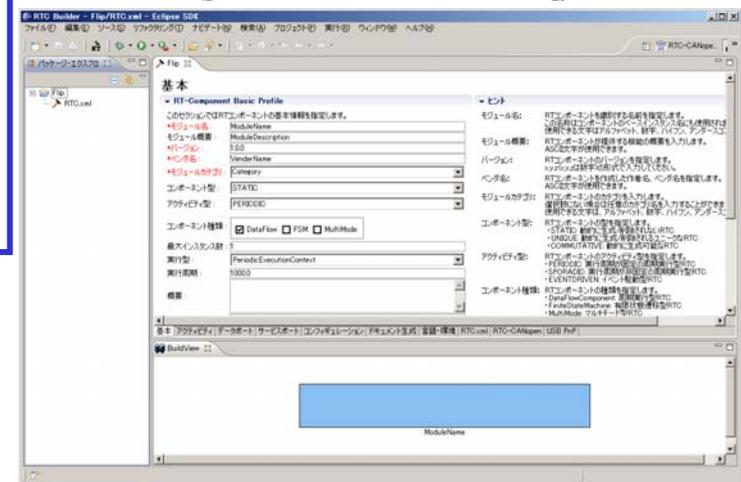


- ※メニューから「ファイル」-「新規」-「プロジェクト」を選択
【新規プロジェクト】画面にて「その他」-「RtcBuilder」を選択し、「次へ」
- ※メニューから「ファイル」-「Open New Builder Editor」を選択

- ※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

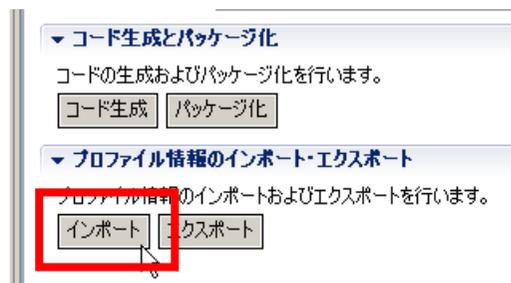
プロジェクト名: Flip

② 「プロジェクト名」欄に入力し、「終了」

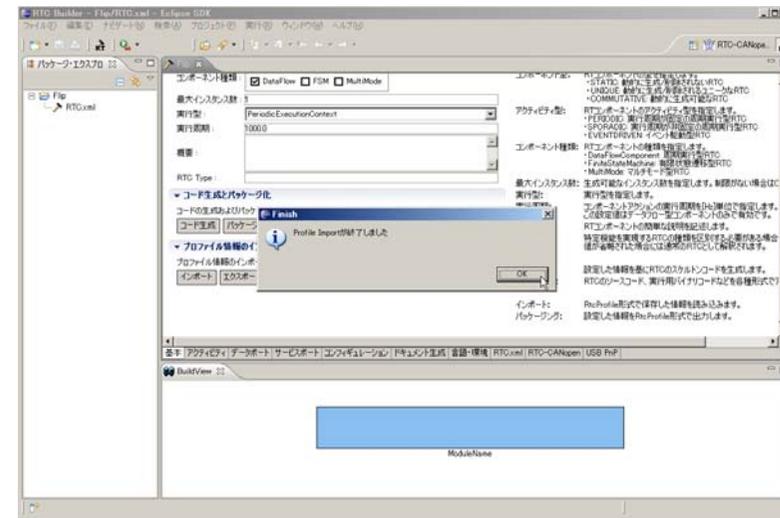


プロファイル インポート

①「基本」タブ下部の「インポート」ボタンをクリック



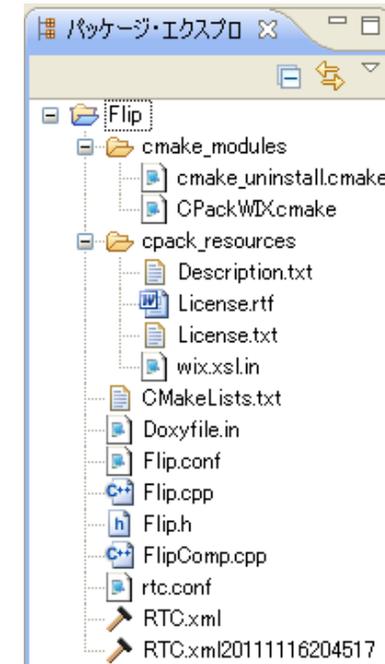
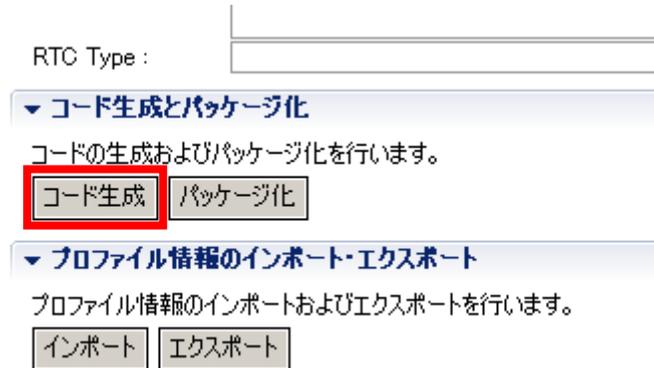
②【インポート】画面にて対象ファイルを選択



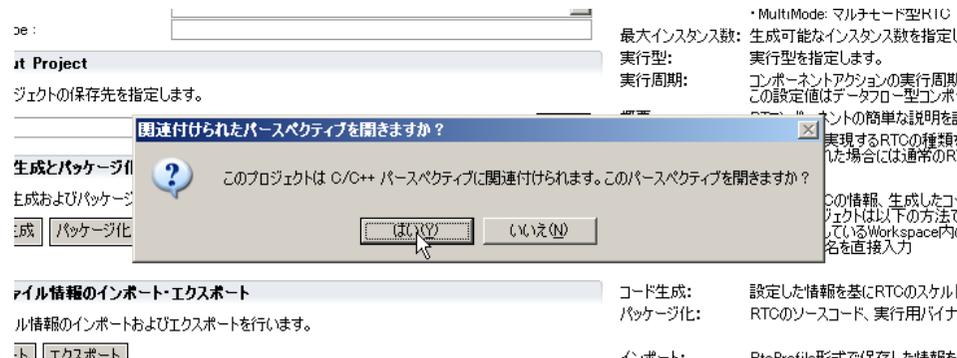
- 作成済みのRTコンポーネント情報を再利用
 - 「エクスポート」機能を利用して出力したファイルの読み込みが可能
 - コード生成時に作成されるRtcProfileの情報を読み込み可能
 - XML形式, YAML形式での入出力が可能

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

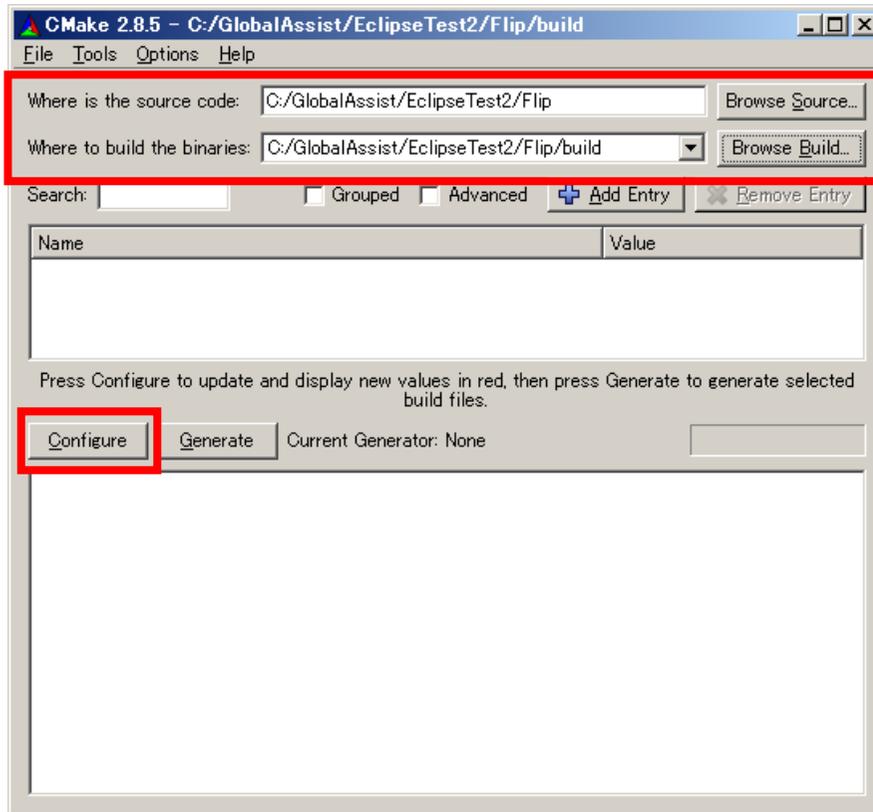
Java版RTC → JDT

(デフォルトインストール済み)

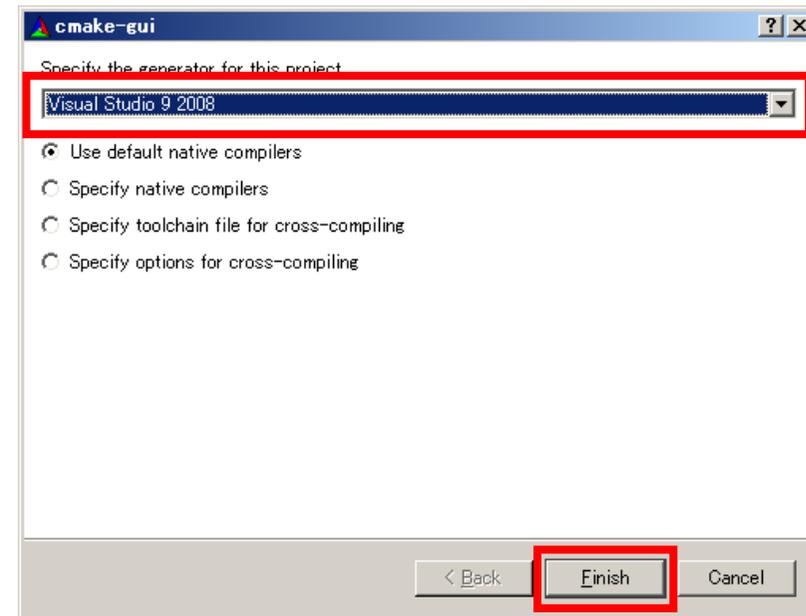
Python版 → PyDev

コンパイル(Windows,CMake利用)

① GUI版Cmakeを起動し, source, binaryのディレクトリを指定



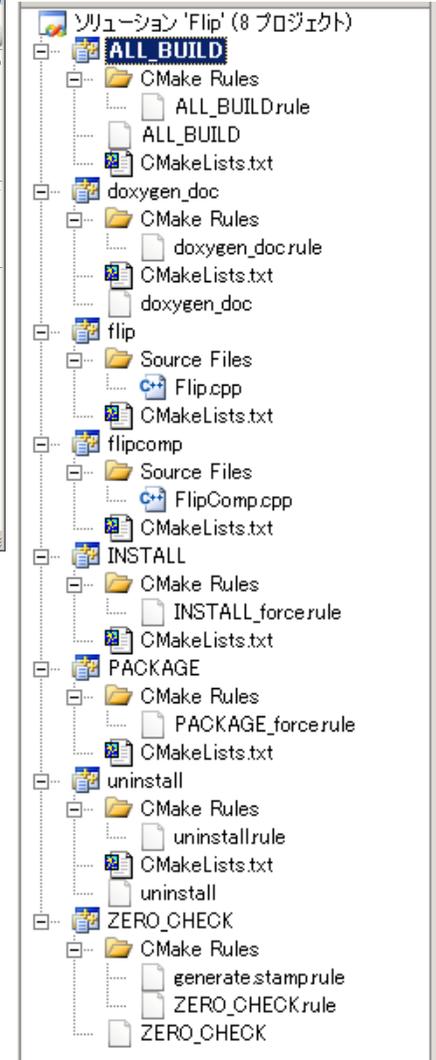
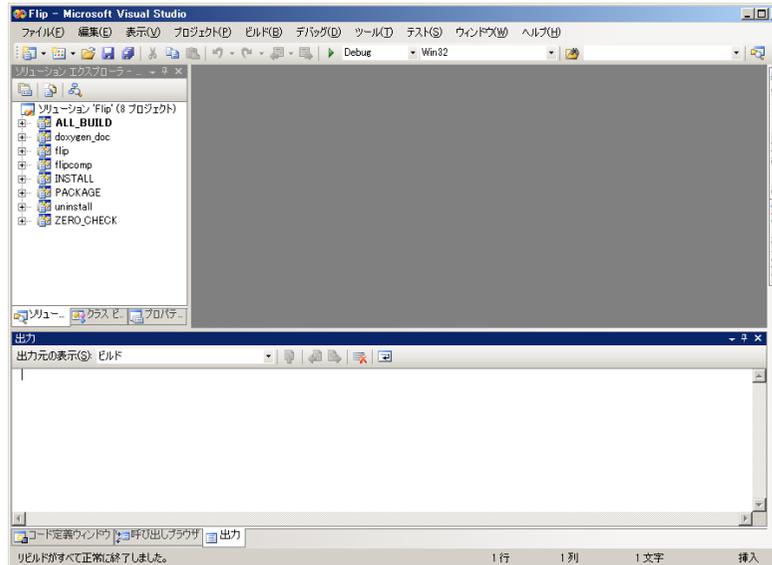
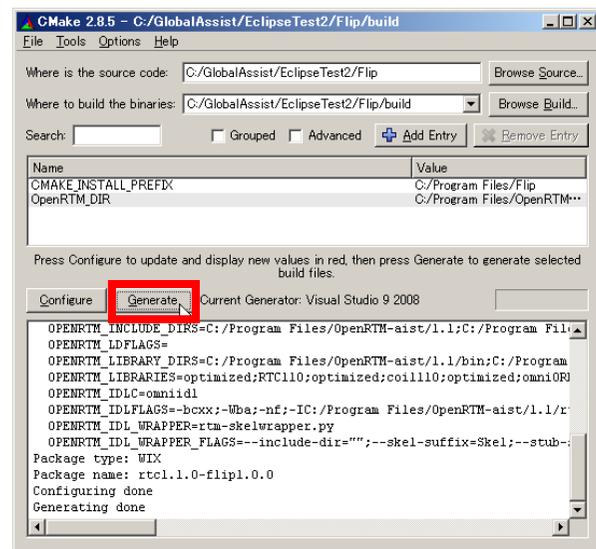
② 「Configure」を実行し, 使用するプラットフォームを選択



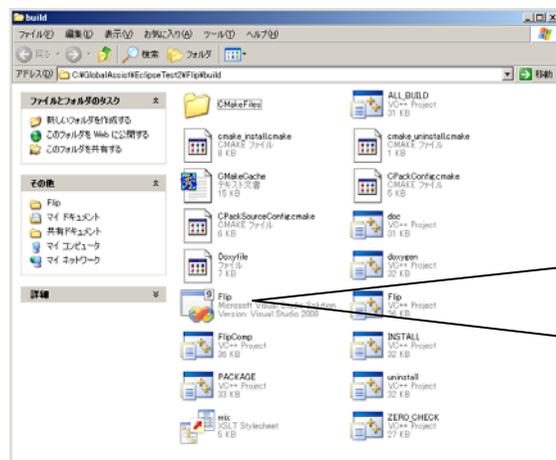
- ❌ binaryには, sourceとは別のディレクトリを指定する事を推奨
- ❌ 日本語は文字化けしてしまうため英数字のみのディレクトリを推奨

コンパイル(Windows, CMake利用)

③ 正常終了後、「Generate」を実行



④ binaryとして指定したディレクトリ内にあるソリューションファイルを開き、「ソリューションをビルド」を実行



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

■ RTコンポーネントの名称など, 基本的な情報を設定

The screenshot shows the configuration window for an RT component named 'Flip'. The 'Basic Profile' section is expanded, showing various fields. A red box highlights the following values:

- 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: 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

■ 生成対象RTCで実装予定のアクティビティを設定

アクティビティ

このセクションでは使用するアクションコレクションを指定します。

コンポーネントの初期化と終了処理に関するアクション

onInitialize onFinalize

実行コンテキストの起動と停止に関するアクション

onStartup onShutdown

alive状態でのコンポーネントアクション

onActivated onDeactivated onAborting

onError onReset

Dataflow型コンポーネントのアクション

onExecute onStateUpdate onRateChanged

FSM型コンポーネントのアクション

onAction

Mode型コンポーネントのアクション

onModeChanged

Documentation

このセクションでは各アクションの概要を説明するドキュメントを記述します。上段のアクションを選択すると、それぞれのドキュメントを記述できます。

アクティビティ名: onInitialize ON OFF

動作概要: コンポーネント自身の各種初期化処理

事前条件: なし

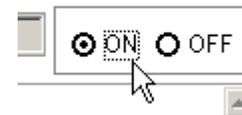
事後条件: コンポーネントの初期化処理が正常に完了している

基本 | アクティビティ | データポート | サービスポート | コンフィギュレーション | ドキュメント生成 | 言語・環境 | RTC.xml | Mapping ID | USB PrP | RTC-CANopen

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

■ 生成対象RTCに付加するDataPortの情報を設定

データポート

ヒント
 データポート: RTコンポーネント間でデータを出力するOutPortとInPortを接続
 InPort: RTコンポーネントに他のRTコンポーネント
 OutPort: RTコンポーネントから他のRTコンポーネント
 ポート名: データポートを識別するポート名は、同一のコンポーネント内で一意ASCII文字が使用でき
 データ型: データポート間でやり取りするデータの型は、OpenRTMで使用することができます
 変数名: データポートに関連付けられた変数の名称は言語に
 ポートの場所: RTSystemEditorなどこのプロファイルはオプション
 ドキュメント: データポートに関する情報を記述する必要はレベルの情報を記述す

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

※ 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

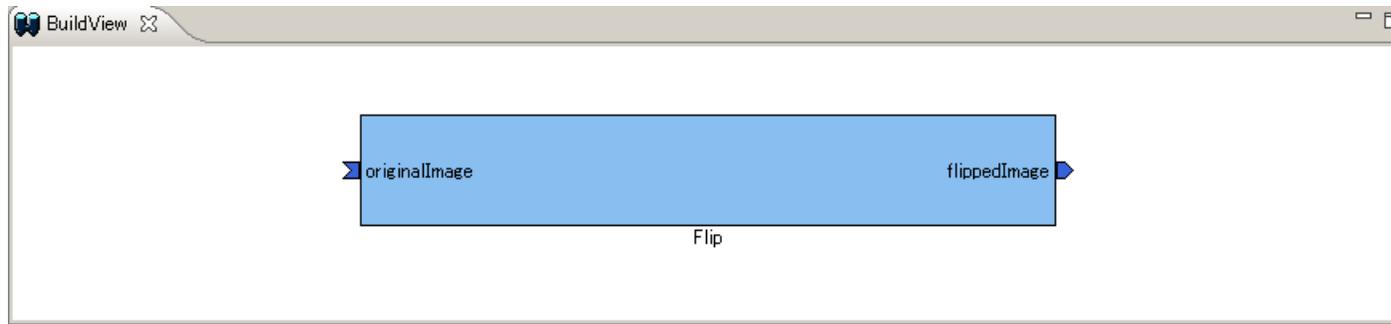
※ OpenRTM-aistにて事前定義されている型については、デフォルトで使用可能
 → [RTM_Root]rtm/idl 以下に存在するIDLファイルで定義された型

※ 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

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

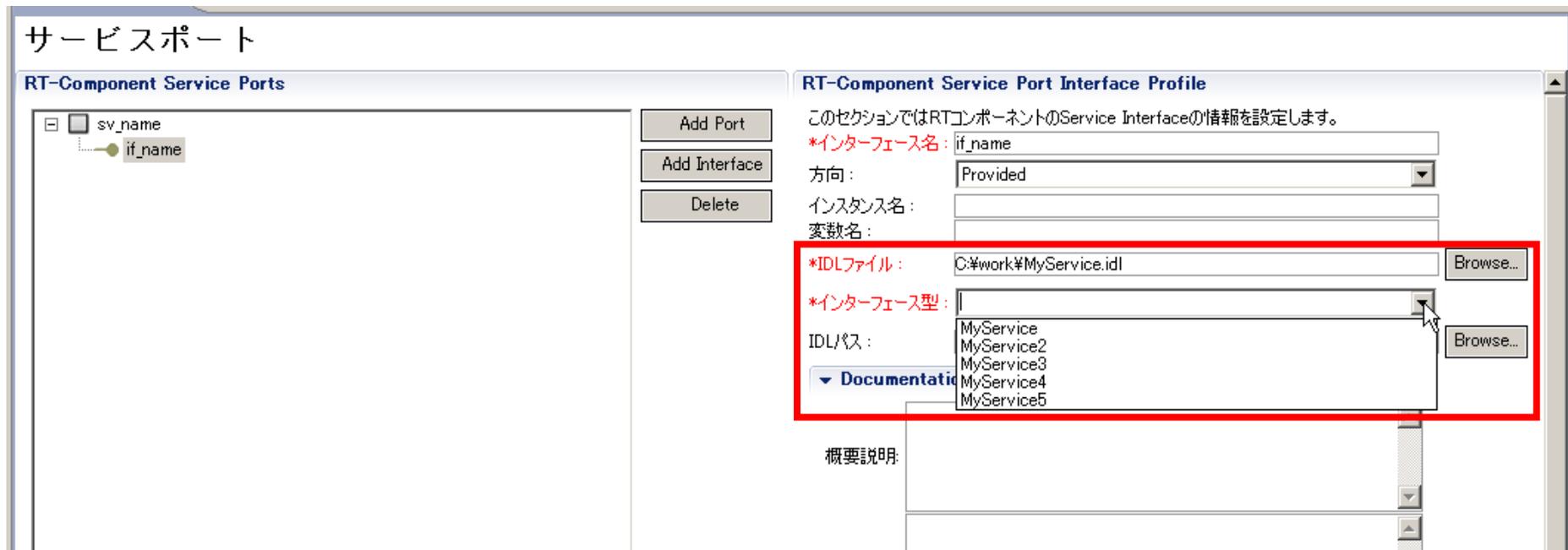
ポート名: **flippedImage**

データ型: **RTC::CameraImage**

変数名: **flippedImage**

表示位置: **right**

■ 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

■ 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

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

▼ 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: 上下反転しない場合
1: 左右反転しない場合
-1: 上下左右反転しない場合

ヒント

Config. Param.: RTコン
再利
パラメ
パラメータ名: コンフ
パラメ
名前
データ型: コンフ
基本!
デフォルト値: コンフ
RTコ
解釈
変数名: コンフ
実際
単位: コンフ
制約条件: コンフ
指定
・100%
・順序
・列挙
・配列
・ハッシュ
Widget: コンフ
設定

①「Add」ボタンをクリックし、追加後、直接入力で名称設定

②詳細画面にて、型情報、変数名などを設定

名称: flipMode
データ型: int
デフォルト値: 0
変数名: flipMode
制約条件: (-1, 0, 1)
Widget: radio

※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

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

■ 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、プロファイ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

詳細情報

OS Version	CPU

Add
Delete

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

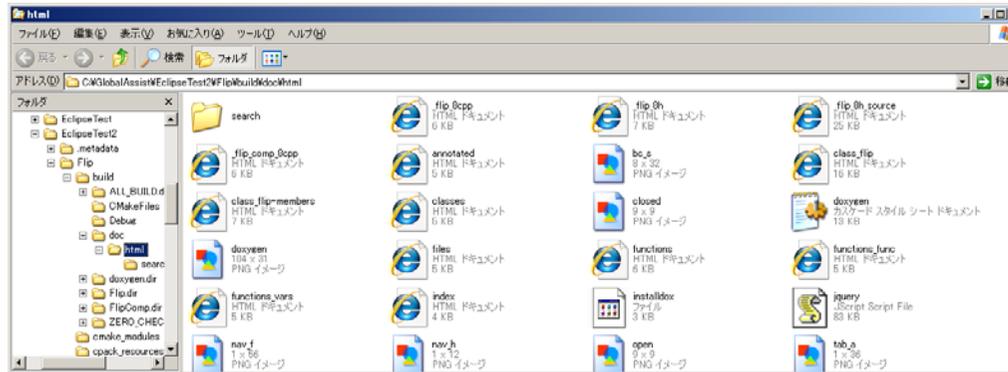
「C++」を選択

RTCBuilder補足説明



ドキュメント作成 (Windows, CMake利用)

※binaryにて指定したディレクトリ以下のdoc/html/doxygen/html以下にドキュメント



■ 生成されたドキュメントの例

flip 1.0.0

構成	構成索引	構成メンバ
クラス Flip		
Flip image component. [詳細]		
#include <Flip.h>		
すべてのメンバー一覧		
Public メソッド		
	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 変数		
int	m_flipMode	
CameraImage	m_originalImage	
InPort< CameraImage >	m_originalImageIn	
CameraImage	m_flippedImage	
OutPort< CameraImage >	m_flippedImageOut	
説明		
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利用してflipModeとしる前のカメラメータで指定します。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。		
<ul style="list-style-type: none"> •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の入出力仕様は以下のとおりです。		

関数

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の画面サイズ初期化

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

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

RTC::ReturnCode_t Flip::onInitialize () [virtual]

コンポーネント自身の各種初期化処理

変数

int Flip::m_flipMode [protected]

画像の反転方法を指定するパラメータ

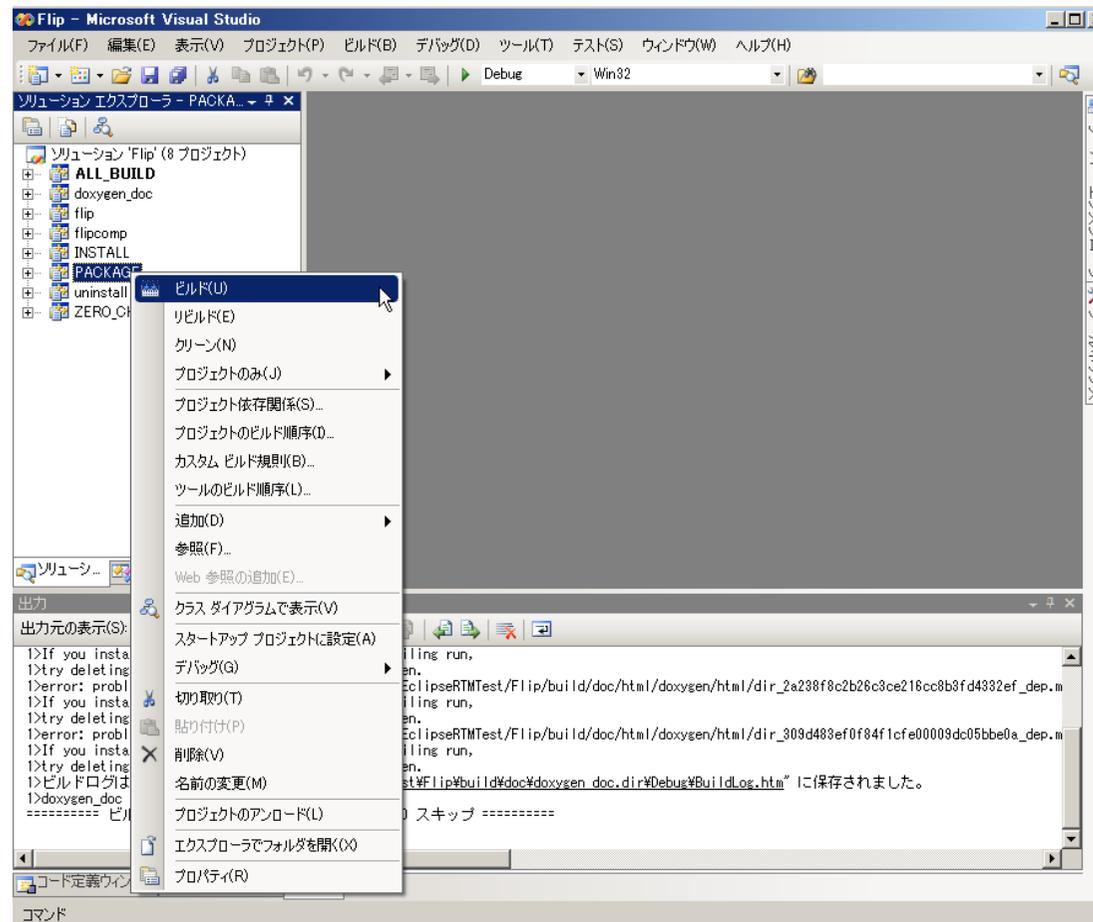
- Name: flipMode flipMode
- DefaultValue: 0
- Unit: なし

flip 1.0.0

メインページ	クラス	ファイル
ファイル一覧		
C:/GlobalAssist/EclipseTest2/Flip/Flip.h		
説明を見る。		
<pre> 00001 // -*- C++ -*- 00002 #ifndef FLIP_H 00003 #define FLIP_H 00004 00005 00006 #include <rtc/Manager.h> 00007 #include <rtc/DataFlowComponentBase.h> 00008 #include <rtc/CorbaPort.h> 00009 #include <rtc/DataInPort.h> 00010 #include <rtc/DataOutPort.h> 00011 #include <rtc/idl/BasicDataTypesSkel.h> 00012 #include <rtc/idl/ExtendedDataTypesSkel.h> 00013 #include <rtc/idl/InterfaceDataTypesSkel.h> 00014 00015 00016 // Service implementation headers 00017 // <rtc-template block="service_impl.h"> 00018 00019 // </rtc-template> 00020 00021 // Service Consumer stub headers 00022 // <rtc-template block="consumer_stub.h"> 00023 00024 // </rtc-template> 00025 00026 using namespace RTC; 00027 00028 class Flip 00029 : public RTC::DataFlowComponentBase 00030 { 00031 public: 00032 Flip (RTC::Manager* manager); 00033 00034 ~Flip (); 00035 00036 // <rtc-template block="public_attribute"> 00037 00038 // </rtc-template> 00039 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 01001 01002 01003 01004 01005 01006 01007 01008 01009 01010 01011 01012 01013 01014 01015 01016 01017 01018 01019 01020 01021 01022 01023 01024 01025 01026 01027 01028 01029 01030 01031 01032 01033 01034 01035 01036 01037 01038 01039 01040 01041 01042 01043 01044 01045 01046 01047 01048 01049 01050 01051 01052 01053 01054 01055 01056 01057 01058 01059 01060 01061 01062 01063 01064 01065 01066 01067 01068 01069 01070 01071 01072 01073 01074 01075 01076 01077 01078 01079 01080 01081 01082 01083 01084 01085 01086 01087 01088 01089 01090 01091 01092 01093 01094 01095 01096 01097 01098 01099 01100 01101 01102 01103 01104 01105 01106 01107 01108 01109 01110 01111 01112 01113 01114 01115 01116 01117 01118 01119 01120 01121 01122 01123 01124 01125 01126 01127 01128 01129 01130 01131 01132 01133 01134 01135 01136 01137 01138 01139 01140 01141 01142 01143 01144 01145 01146 01147 01148 01149 01150 01151 01152 01153 01154 01155 01156 01157 01158 01159 01160 01161 01162 01163 01164 01165 01166 01167 01168 01169 01170 01171 01172 01173 01174 01175 01176 01177 01178 01179 01180 01181 01182 01183 01184 01185 01186 01187 01188 01189 01190 01191 01192 01193 01194 01195 01196 01197 01198 01199 01200 01201 01202 01203 01204 01205 01206 01207 01208 01209 01210 01211 01212 01213 01214 01215 01216 01217 01218 01219 01220 01221 01222 01223 01224 01225 01226 01227 01228 01229 01230 01231 01232 01233 01234 01235 01236 01237 01238 01239 01240 01241 01242 01243 01244 01245 01246 01247 01248 01249 01250 01251 01252 01253 01254 01255 01256 01257 01258 01259 01260 01261 01262 01263 01264 01265 01266 01267 01268 01269 01270 01271 01272 01273 01274 01275 01276 01277 01278 01279 01280 01281 01282 01283 01284 01285 01286 01287 01288 01289 01290 01291 01292 01293 01294 01295 01296 01297 01298 01299 01300 01301 01302 01303 01304 01305 01306 01307 01308 01309 01310 01311 01312 01313 01314 01315 01316 01317 01318 01319 01320 01321 01322 01323 01324 01325 01326 01327 01328 01329 01330 01331 01332 01333 01334 01335 01336 01337 01338 01339 01340 01341 01342 01343 01344 01345 01346 01347 01348 01349 01350 01351 01352 01353 01354 01355 01356 01357 01358 01359 01360 01361 01362 01363 01364 01365 01366 01367 01368 01369 01370 01371 01372 01373 01374 01375 01376 01377 01378 01379 01380 01381 01382 01383 01384 01385 01386 01387 01388 01389 01390 01391 01392 01393 01394 01395 01396 01397 01398 01399 01400 01401 01402 01403 01404 01405 01406 01407 01408 01409 01410 01411 01412 01413 01414 01415 01416 01417 01418 01419 01420 01421 01422 01423 01424 01425 01426 01427 01428 01429 01430 01431 01432 01433 01434 01435 01436 01437 01438 01439 01440 01441 01442 01443 01444 01445 01446 01447 01448 01449 01450 01451 01452 01453 01454 01455 01456 01457 01458 01459 01460 01461 01462 01463 01464 01465 01466 01467 01468 01469 01470 01471 01472 01473 01474 01475 01476 01477 01478 01479 01480 01481 01482 01483 01484 01485 01486 01487 01488 01489 01490 01491 01492 01493 01494 01495 01496 01497 01498 01499 01500 01501 01502 01503 01504 01505 01506 01507 01508 01509 01510 01511 01512 01513 01514 01515 01516 01517 01518 01519 01520 01521 01522 01523 01524 01525 01526 01527 01528 01529 01530 01531 01532 01533 01534 01535 01536 01537 01538 01539 01540 01541 01542 01543 01544 01545 01546 01547 01548 01549 01550 01551 01552 01553 01554 01555 01556 01557 01558 01559 01560 01561 01562 01563 01564 01565 01566 01567 01568 01569 01570 01571 01572 01573 01574 01575 01576 01577 01578 01579 01580 01581 01582 01583 01584 01585 01586 01587 01588 01589 01590 01591 01592 01593 01594 01595 01596 01597 01598 01599 01600 01601 01602 01603 01604 01605 01606 01607 01608 01609 01610 01611 01612 01613 01614 01615 01616 01617 01618 01619 01620 01621 01622 01623 01624 01625 01626 01627 01628 01629 01630 01631 01632 01633 01634 01635 01636 01637 01638 01639 01640 01641 01642 01643 01644 01645 01646 01647 01648 01649 01650 01651 01652 01653 01654 01655 01656 01657 01658 01659 01660 01661 01662 01663 01664 01665 01666 01667 01668 01669 01670 01671 01672 01673 01674 01675 01676 01677 01678 01679 01680 01681 01682 01683 01684 01685 01686 01687 01688 01689 01</pre>		

配布用パッケージ作成(Windows,CMake利用)

■ ソリューション中の「PACKAGE」をビルド

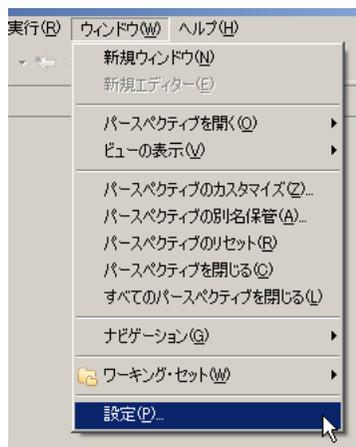


- binaryにて指定したディレクトリ直下にmsi形式のインストールパッケージを生成
 - コンポーネントのインストール先
C:¥Program Files¥OpenRTM-aist¥1.1¥components¥<言語>/<パッケージ名>

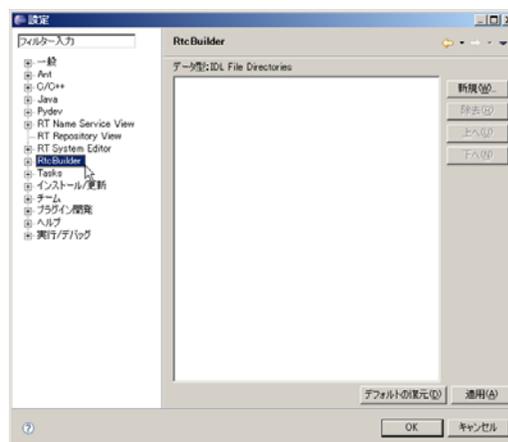
■ DataPortにて利用するデータ型の指定

→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

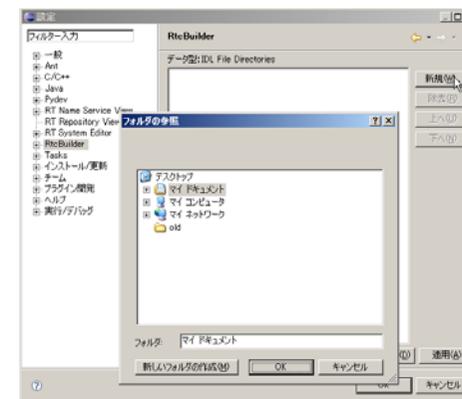
①メニューから
「ウィンドウ」-「設定」



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



※独自に定義したデータ型を使用する場合のみ必要な設定

OpenRTM-aistにて標準で用意されている型のみを使用する場合には設定不要

・標準型の定義内容格納位置：[RTM_Root]rtm/idl

→BasicDataType.idl, ExtendedDataTypes.idlなど

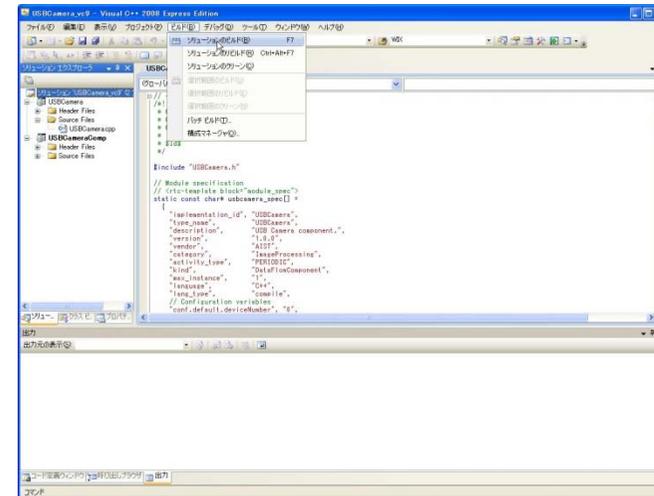
→デフォルト設定では，[RTM_Root]=C:/Program Files/OpenRTM-aist/1.1/

コンパイル・実行(Windows)

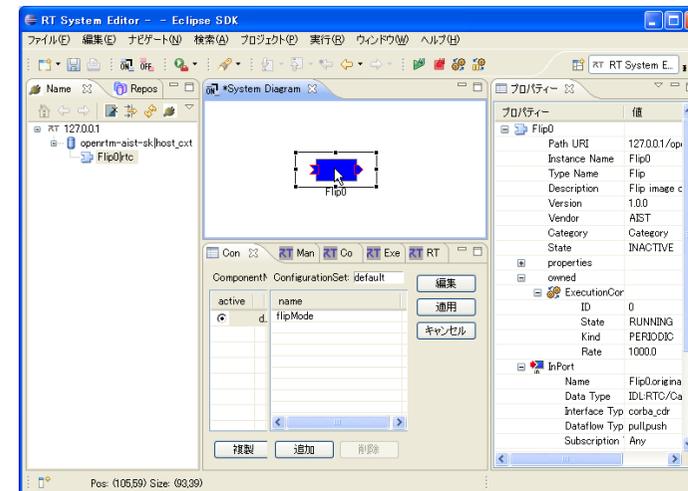
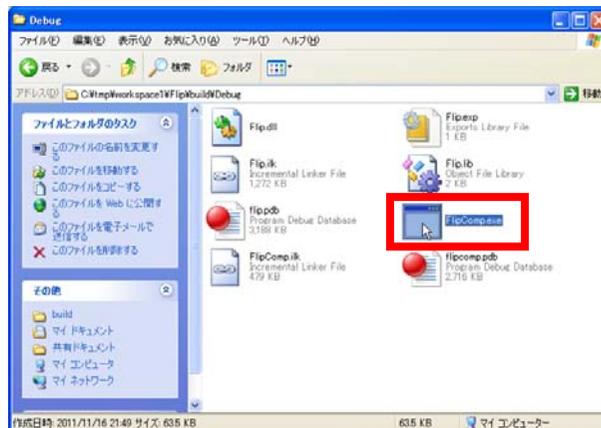
①コード生成先ディレクトリ内の「copyprops.bat」をダブルクリックして、設定ファイルをコピー



②VisualStudioを用いたビルド

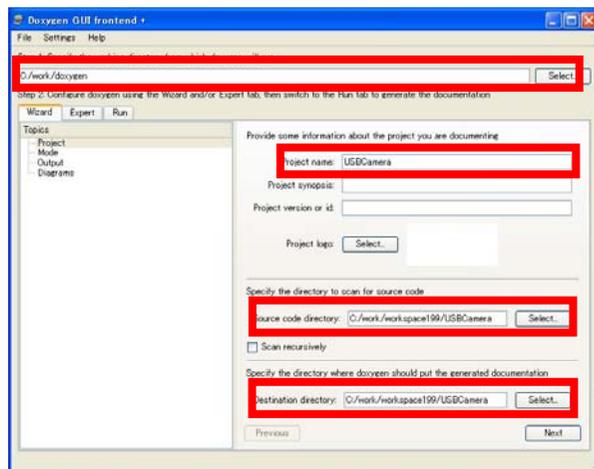


③FlipComp¥¥Debug内のFlipComp.exeを起動

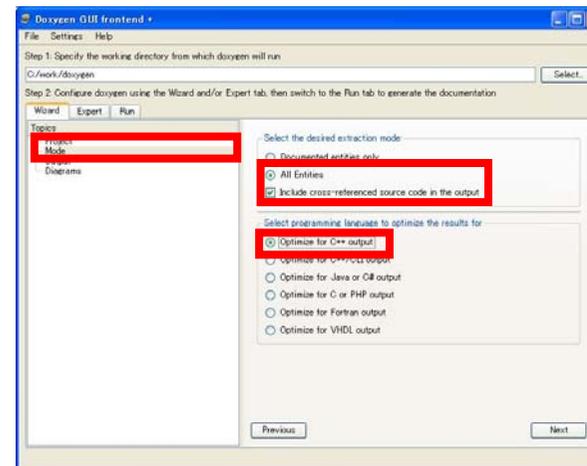


ドキュメント作成(Windows)

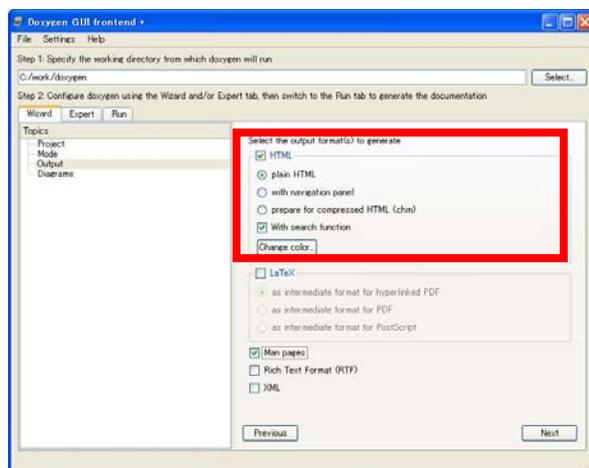
- ① Doxygen用GUIツールを起動
作業用ディレクトリ,ソース格納場所,
生成ファイル出力先,プロジェクト名を指定



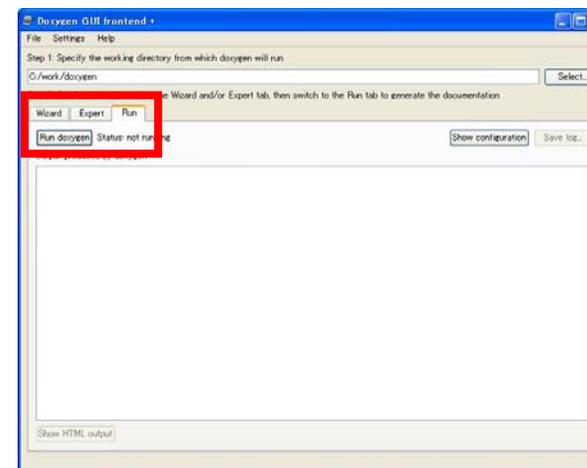
- ② 「Mode」セクションにて,
出力内容,使用言語を指定



- 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



RTSystemEditor補足説明



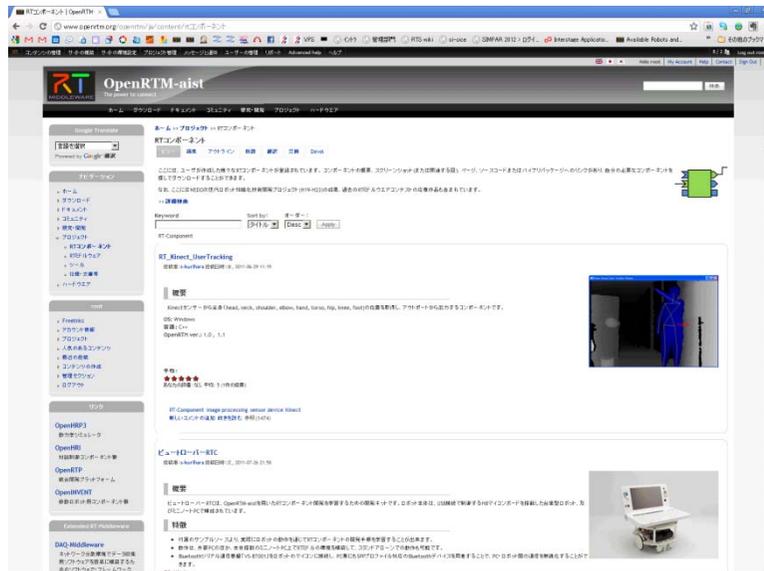
■ プロジェクトとは

- ユーザが作成した様々なコンポーネントやツールの公開場所
- ユーザ登録すれば、誰でも自分の成果物の紹介ページを作成可能
- 他のユーザに自分のコンポーネント等を紹介することができる

■ プロジェクトのカテゴリ

- RTコンポーネント: 1つのコンポーネントまたは複数のコンポーネント群などが登録されています。
- RTミドルウェア: OpenRTM-aistや他のミドルウェア、ミドルウェア拡張モジュール等が登録されています。
- ツール: 各種ツール(RTSystemEditorやrtshellを含む)ツールはこのカテゴリになります。
- 関連ドキュメント: 関連ドキュメントとは、各種インターフェースの仕様書やマニュアル等を含みます。

プロジェクトページ



タイプ	登録数
RTコンポーネント群	638
RTミドルウェア	29
ツール	39
仕様・文書	4
ハードウェア	30

既存コンポーネントの再利用

■ プロジェクトから対象コンポーネントを取得

■ 「顔検出コンポーネント」

<http://www.openrtm.org/openrtm/ja/project/facedetect>

対象コンポーネントをダウンロード

OpenRTM-aist
The power to connect

ホーム >> プロジェクト >> RTコンポーネント >> 顔検出コンポーネント

顔検出コンポーネント

投稿者: s-kurihara 投稿日時: 火, 2011-11-15 18:20

同合先(メールアドレス): openrtm@openrtm.org

InPortから入力された画像から顔を検出し、入力画像に検出結果を描画した画像をOutPortから出力するとともに、別のOutPortでは、顔の位置や検出した顔の数を出力するコンポーネントです。

詳細は、[こちら](#)をご覧ください。

バージョン	Downloads	日付	Links
0.1	Download (17.44 MB)	2011-11-15	Notes

OS: Windows
言語: C++
OpenRTM ver.: 1.1
☆☆☆☆☆

既存コンポーネントの再利用

- ダウンロードしたファイル(FaceDetect.zip)を解凍
- 解凍したディレクトリ内の以下のファイルを実行し、システムエディタ上に配置
\$(FaceDetect_Root)/build/Release/FaceDetectComp.exe

The screenshot displays the RT System Editor interface. The main window shows a system diagram with components: DirectShowCam0, Flip0, FaceDetect0, Edge0, and CameraViewer0. A configuration window for FaceDetect0 is open, showing the following table:

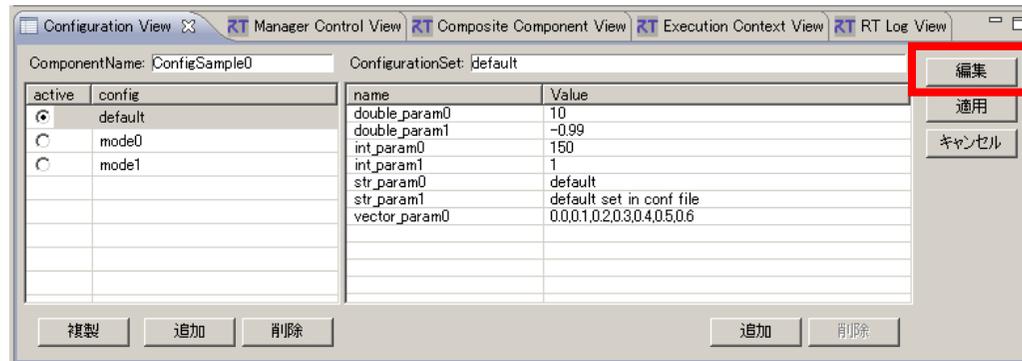
ComponentName	ConfigurationSet	name	Value
FaceDet	default	downscale	1.2
		haarcascade	../data/haarcascades/haarc...
		min_object_height	30
		min_object_width	30

On the right, the Properties window shows details for FaceDetect0, including Path URI, Instance Name, Type Name, Description, Version, Vendor, Category, State, and ExecutionCor. An inset window titled 'Capture Image' shows a video feed of a person with a red bounding box around their face, demonstrating the face detection component's output.

ネットワーク上の他のRTCとの接続

- IPアドレスの確認
 - スタートメニュー中の「全てのプログラム」-「アクセサリ」-「コマンドプロンプト」
 - コマンド「ipconfig」を実行
- 他PC上で動作するRTCとの接続
 - 隣の方のIPアドレスを聞く
 - RTSystemEditorの「ネームサーバを追加(コンセントのアイコン)」をクリックして、上記のIPアドレスを入力する
 - 隣の方のネームサーバ内の階層化にあるDirectShowCamをシステムエディタにDnDする
 - 上記でDnDしたDirectShowCamと自分のPC上で起動したCameraViewerのデータポートを接続する

■ 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

※コンフィギュレーション情報を複数保持している場合、上部のタブで編集対象を切り替え



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]

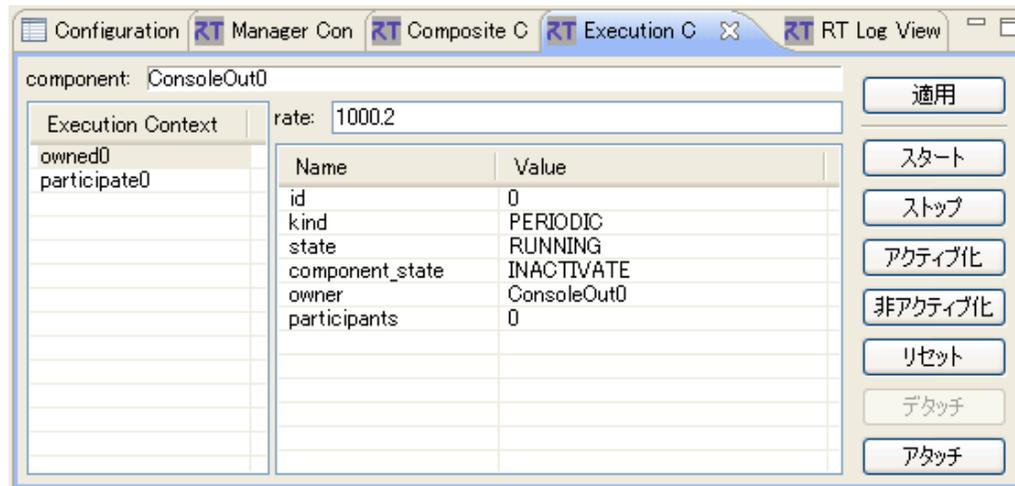
※例) conf.__mode1.str_param0: (bar2,foo2,dara2)

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

実行コンテキストビュー

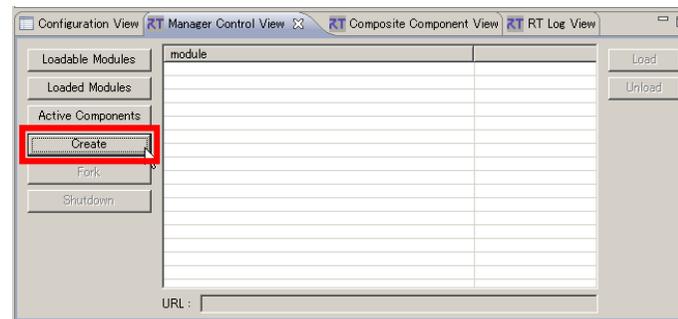
- RTコンポーネントが属する実行コンテキスト(EC)を一覧表示



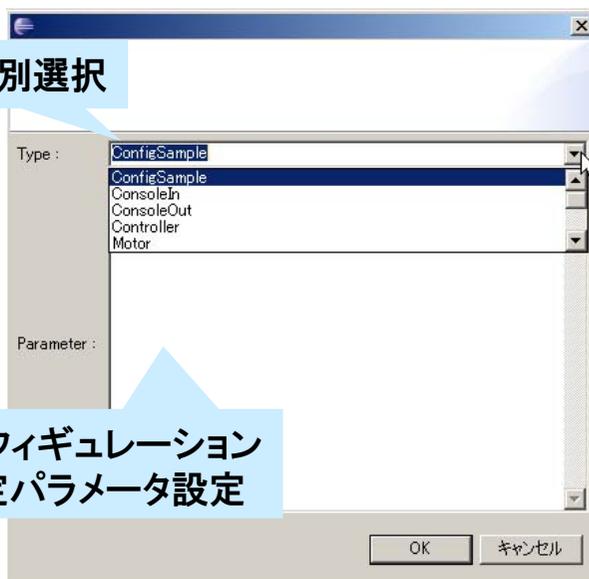
属性名	説明
id	ECのID. オンラインの場合には, context_handleを表示
kind	ECの種別(PERIODIC/EVENT_DRIVEN/OTHER)
state	ECの状態(RUNNING/STOPPING)
component state	対象RTCの状態(ACTIVE/INACTIVE/ERROR)
owner	対象ECを所有しているオーナーRTCのインスタンス名
participants	対象ECに参加中のRTCの数

※対象ECの実行周期の変更, EC自身の動作開始/終了, 新規RTCへのアタッチ, アタッチ済みRTCのデタッチも可能

■ RTコンポーネントの新規インスタンスの生成

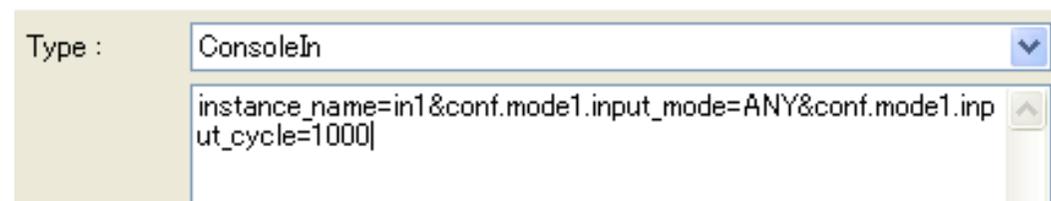


RTC種別選択



コンフィギュレーション
指定パラメータ設定

- 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



■ 選択したRTCから収集したログ情報を一覧表示

component	time	level	component	logger	message
<input checked="" type="checkbox"/> Notify0	2011-04-28 ...	ERROR	Notify0	RTC	test log!
<input checked="" type="checkbox"/> Notify1	2011-04-28 ...	ERROR	Notify1	RTC	test log!
	2011-04-28 ...	ERROR	Notify0	RTC	test log!
	2011-04-28 ...	ERROR	Notify1	RTC	test log!
	2011-04-28 ...	ERROR	Notify0	RTC	test log!
	2011-04-28 ...	ERROR	Notify1	RTC	test log!
	2011-04-28 ...	ERROR	Notify0	RTC	test log!
	2011-04-28 ...	ERROR	Notify1	RTC	test log!
	2011-04-28 ...	ERROR	Notify0	RTC	test log!
	2011-04-28 ...	ERROR	Notify1	RTC	test log!
	2011-04-28 ...	ERROR	Notify0	RTC	test log!
	2011-04-28 ...	ERROR	Notify1	RTC	test log!
	2011-04-28 ...	ERROR	Notify0	RTC	test log!
	2011-04-28 ...	ERROR	Notify1	RTC	test log!
	2011-04-28 ...	ERROR	Notify0	RTC	test log!
	2011-04-28 ...	ERROR	Notify1	RTC	test log!
	2011-04-28 ...	ERROR	Notify0	RTC	test log!
	2011-04-28 ...	ERROR	Notify1	RTC	test log!

※近日機能追加予定

● ログ収集の開始/停止

Activate (A)	Ctrl+Alt+A
Deactivate (W)	Ctrl+Alt+D
Reset (Q)	
Exit (E)	

Start (Z)	
Stop (X)	

複合コンポーネントの作成	
エディタで開く	
複合コンポーネントを解除	

ログ収集を開始	
Create Profile Table	

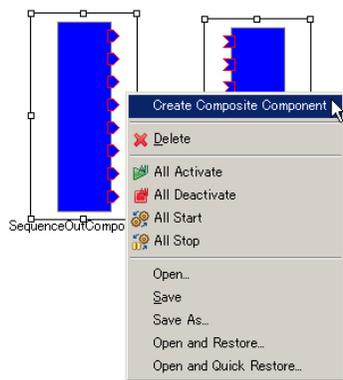
● ログ情報のフィルタリング

time	level	component	logger	message
2011-04-28 ...	ERROR	Notify0	RTC	test log!
2011-04-28 ...	ERROR	Notify1	RTC	test log!
2011-04-28 ...	ERROR	Notify0	RTC	test log!
2011-04-28 ...	ERROR	Notify1	RTC	test log!
2011-04-28 ...	ERROR	Notify0	RTC	test log!
2011-04-28 ...	ERROR	Notify1	RTC	test log!
2011-04-28 ...	ERROR	Notify0	RTC	test log!
2011-04-28 ...	ERROR	Notify1	RTC	test log!
2011-04-28 ...	ERROR	Notify0	RTC	test log!
2011-04-28 ...	ERROR	Notify1	RTC	test log!
2011-04-28 ...	ERROR	Notify0	RTC	test log!
2011-04-28 ...	ERROR	Notify1	RTC	test log!
2011-04-28 ...	ERROR	Notify0	RTC	test log!
2011-04-28 ...	ERROR	Notify1	RTC	test log!
2011-04-28 ...	ERROR	Notify0	RTC	test log!
2011-04-28 ...	ERROR	Notify1	RTC	test log!
2011-04-28 ...	ERROR	Notify0	RTC	test log!
2011-04-28 ...	ERROR	Notify1	RTC	test log!
2011-04-28 ...	ERROR	Notify0	RTC	test log!
2011-04-28 ...	ERROR	Notify1	RTC	test log!

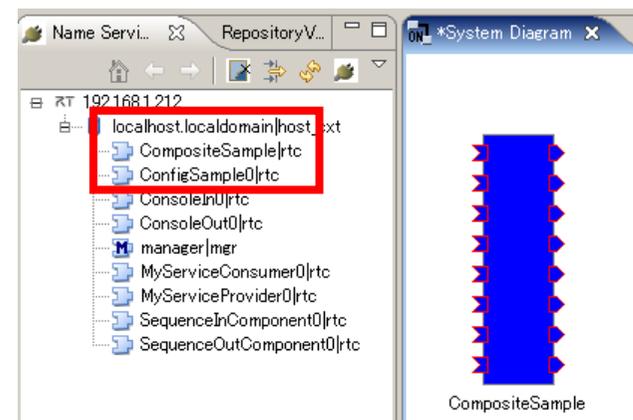
複合コンポーネント

- 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

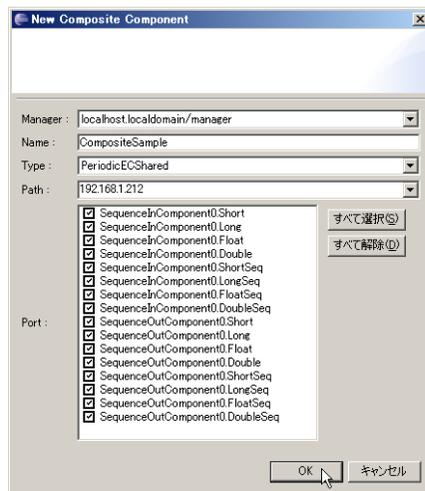
① 複数RTCを選択している状態で右クリック



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

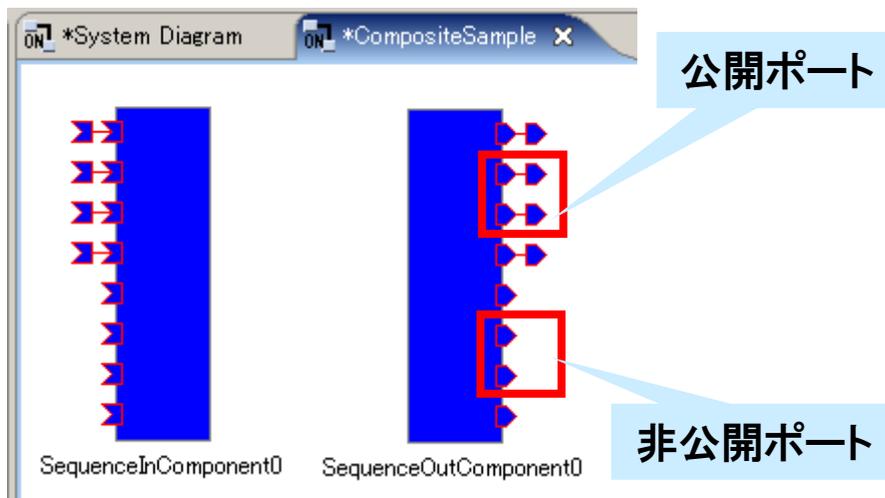
複合コンポーネント

■ 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



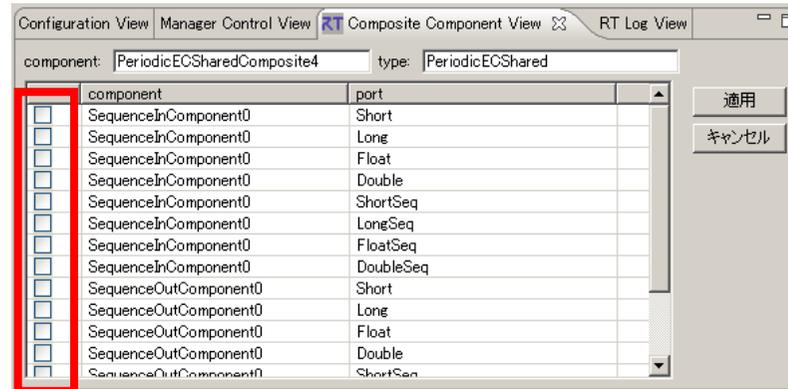
- ※エディタ内に別RTCをDnDすることで、子コンポーネントの追加が可能
→追加したRTCのポートは全て非公開に設定
- ※エディタ内のRTCを削除することで、子コンポーネントの削除が可能
→削除されたRTCは、親エディタに表示

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

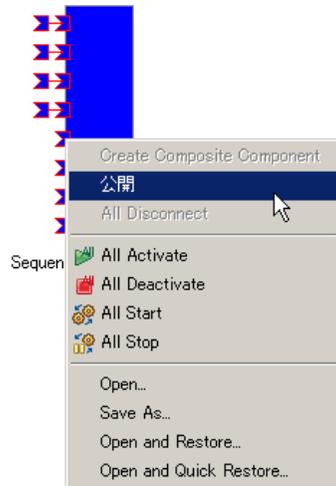
ポート公開情報



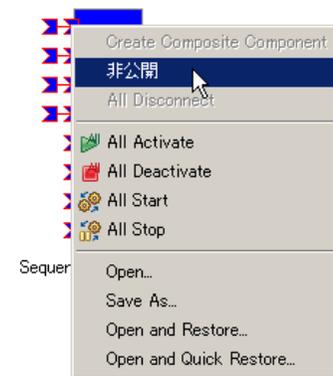
※ポート公開情報を変更し、「適用」をクリック

● 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」

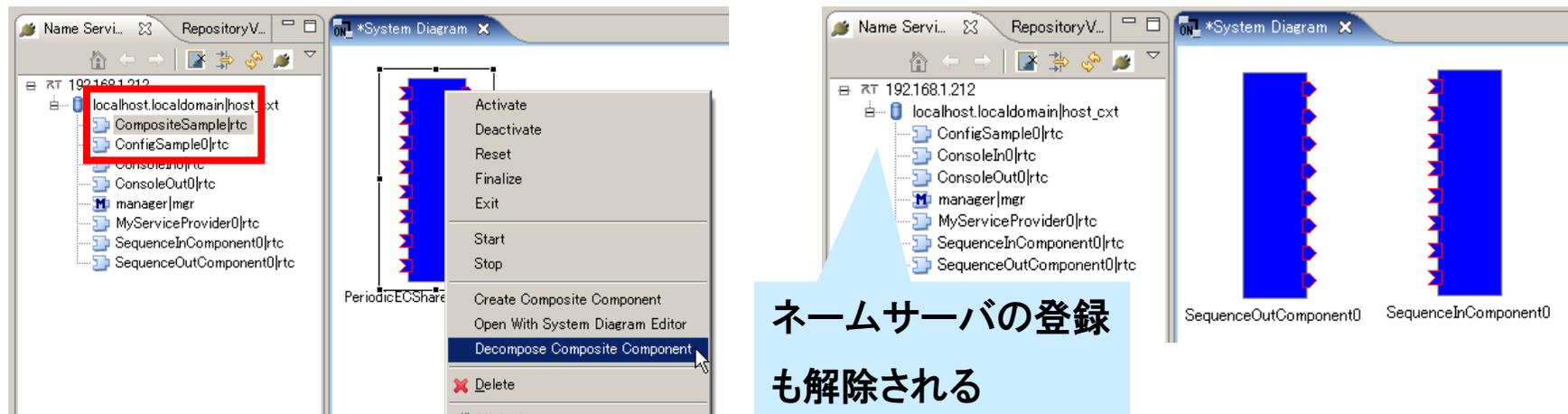


外部コンポーネントと接続されているポートを「非公開」に設定することはできません

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

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

リポジトリビュー

オフライン・システムエディタ

プロパティビュー

Configuration View

ComponentName	ConfigurationSet	active	config	name	Value
ImageProcess_1					

■ 接続一状態通知オブザーバ

- RTCの生存確認用オブザーバに関する設定
 - RTSE側から生存確認を行うのではなく, RTC側から通知(ハートビート)を行う形
 - OpenRTM-aist-1.1以降で対応



- ハートビート有効化:ハートビートによる生存確認機能の有効化
- ハートビート受信間隔:ハートビートの受信間隔. この間隔以内にRTC側からハートビートが送られてこないと生存確認失敗と判断
- ハートビート受信回数:この回数を超えて生存確認に失敗した場合, 対象RTCに異常が発生したと判断

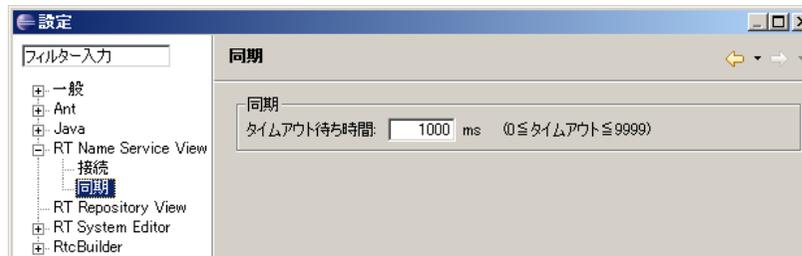
■ 「RT Name Service View」－「接続」【接続周期】

- 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



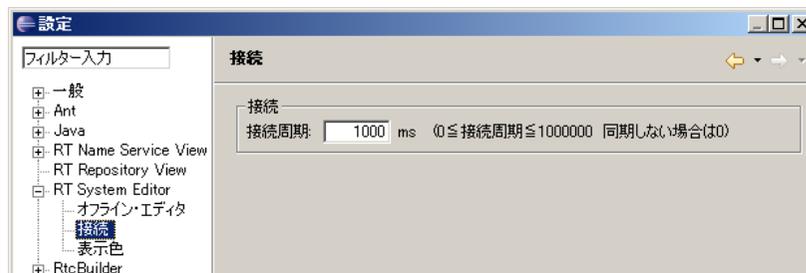
■ 「RT Name Service View」－「同期」【タイムアウト待ち時間】

- 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】

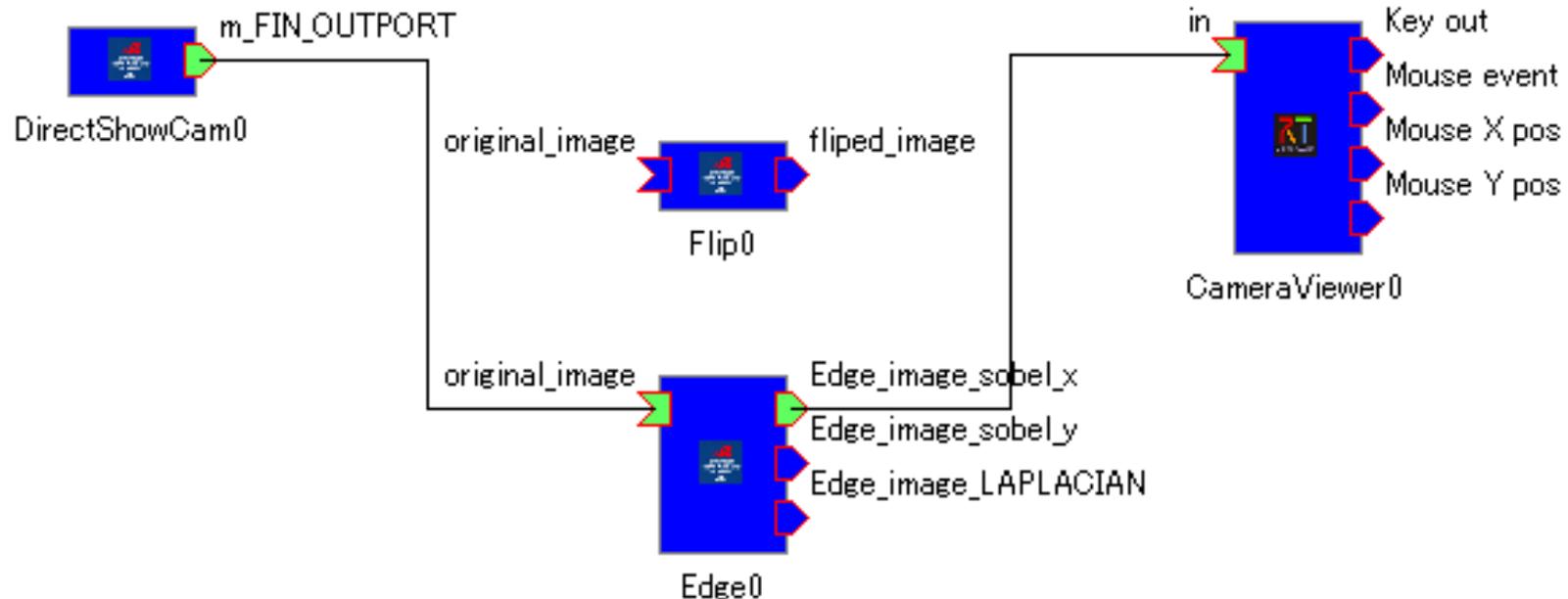
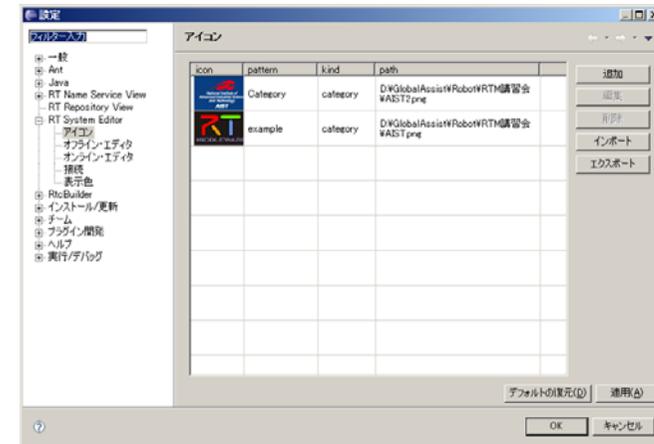
- システムエディタが、ネームサーバに情報を問い合わせる周期



**【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**

設定画面

- 「RT System Editor」-「アイコン」【表示アイコン】
 - RTC内に表示するアイコンを指定可能
 - カテゴリ単位, RTC名称単位で設定が可能



RTミドルウェア講習会

