効率よいRTシステム運用法

宮本 信彦
国立研究開発法人産業技術総合研究所
ロボットイノベーション研究センター
ロボットソフトウェアプラットフォーム研究チーム

資料のダウンロード

- RTミドルウェアサマーキャンプ2016のページから2016SummerCamp-07.zipをダウンロード

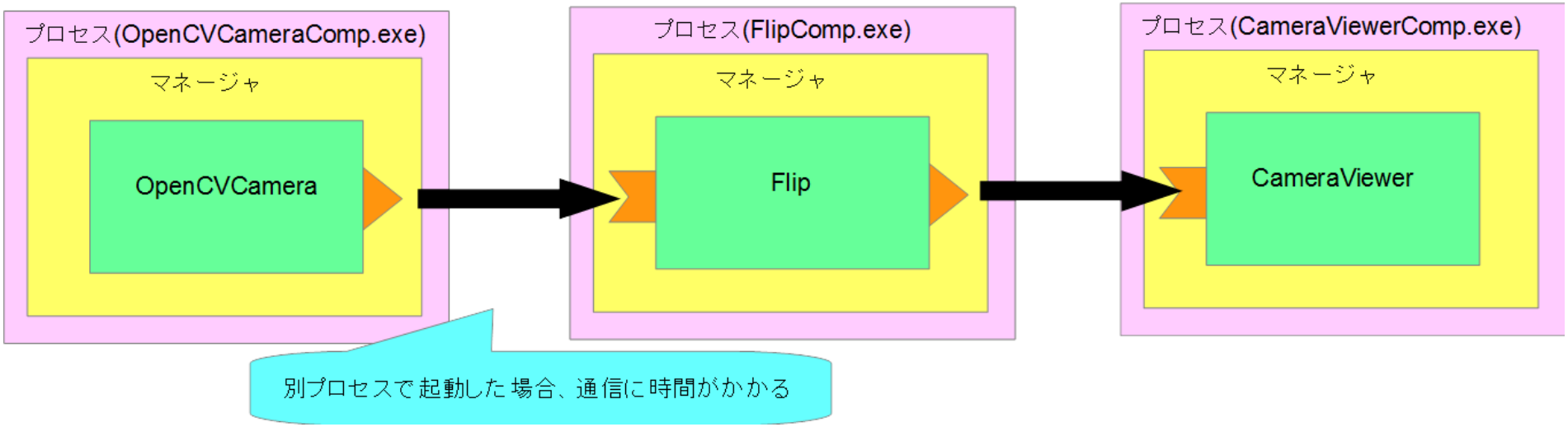
16:20 - 17:50	産総研見学会	調整中
18:00 -	懇親会@さくら館	-

8月2日(火)	第2日目	-
09:00 - 10:00	自習時間	-
10:00 - 10:30	講義4: 利用可能なRTコンポーネント	菅 佑樹 (SUGAR SWEET ROBOTICS)
10:30 - 11:00	講義5: ROS-RTM相互運用とJSKでの取り組み (仮)	岡田 慧 (東京大学)
11:00 - 11:30	講義6	調整中
12:00 - 13:00	昼休み	-
13:00 - 16:30	SysMLモデリングおよび実習	-
16:30 - 17:00	SysMLモデリング結果発表・講評	-

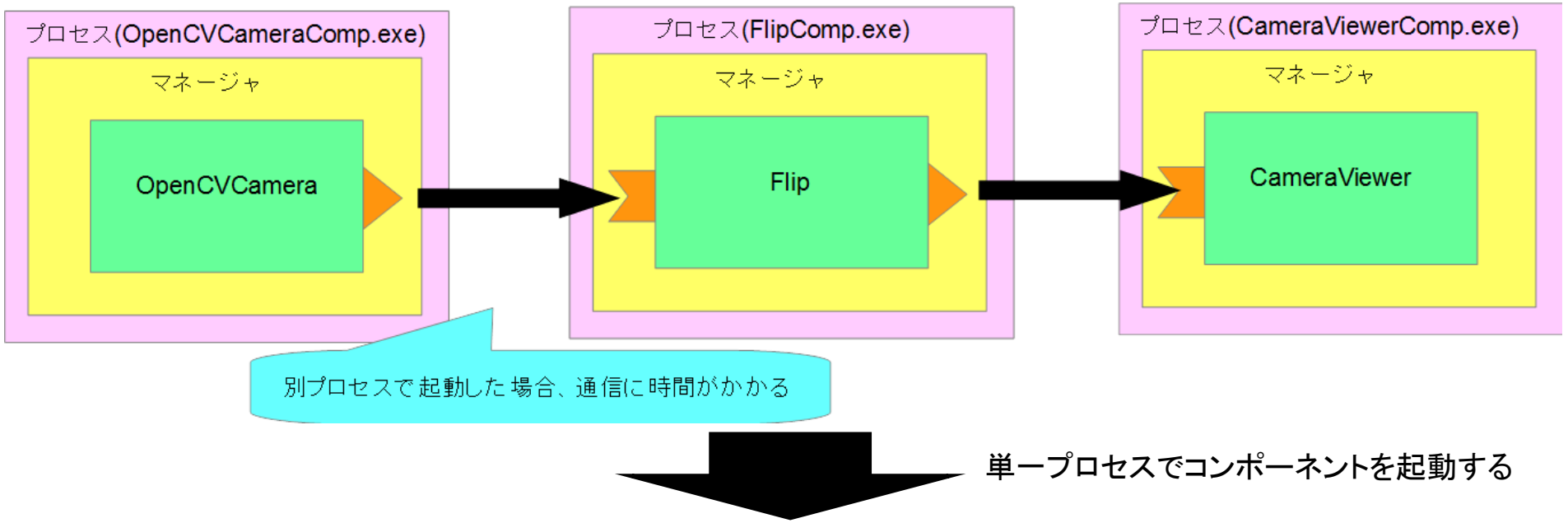
8月3日(水)	第3日目	-
09:00 - 10:00	自習時間	-
10:00 - 10:30	講義7: 効率よいRTシステム運用法 講義資料:2016SummerCamp-07.zip	宮本 信彦 (産業技術総合研究所)
10:30 - 11:00	講義8: rtshell	ジェフ ビグス (産業技術総合研究所)
11:00 - 16:30	実習	-
16:30 - 17:00	各チームの進捗報告	-

RTシステムを作成する手順について

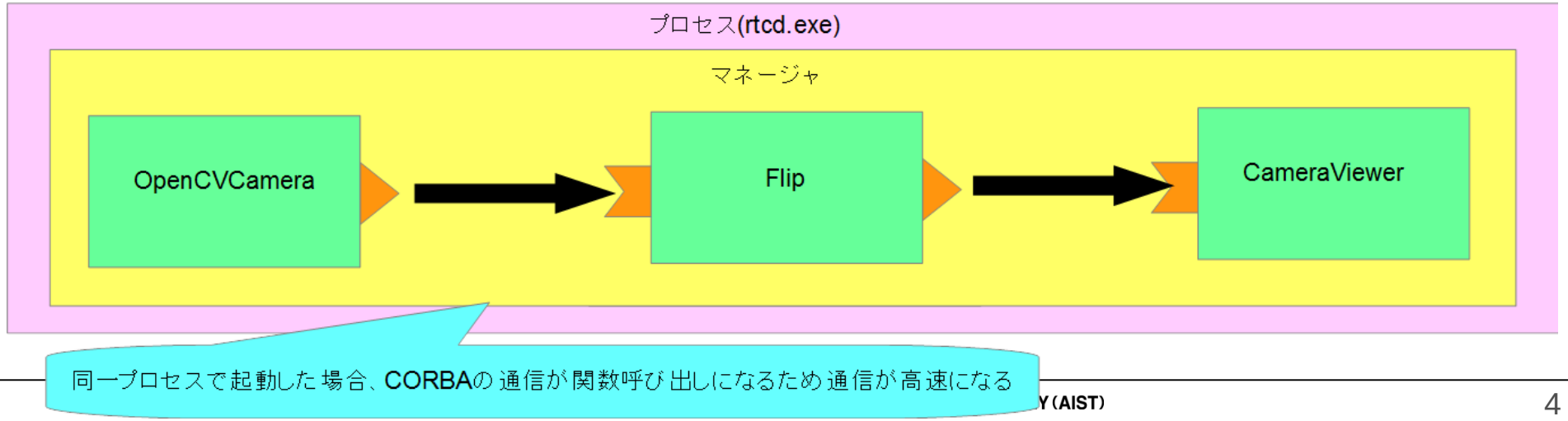
- 講習会のFlipコンポーネント動作確認の手順でRTシステムを作成した場合
  - コンポーネントをそれぞれ別プロセスで起動するため、通信に時間がかかる



RTシステムを作成する手順について

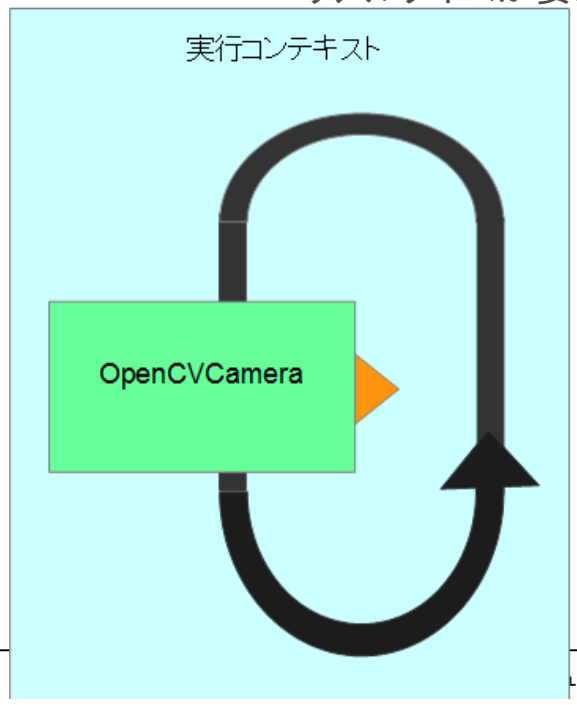


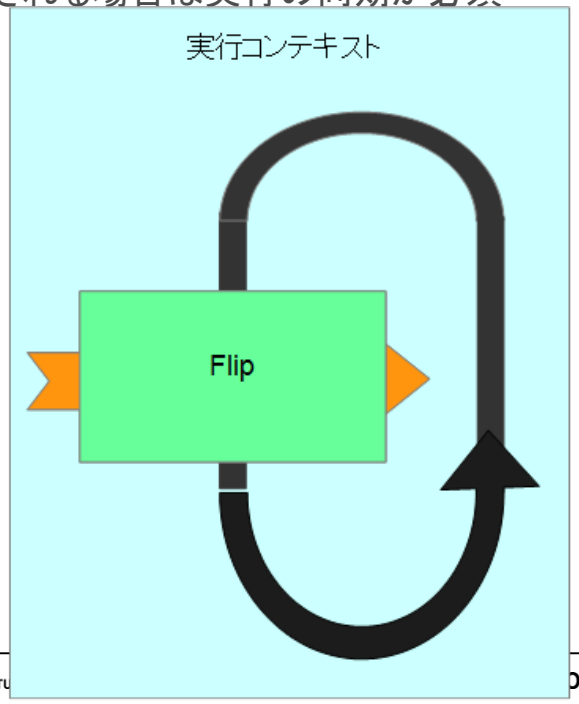
単一プロセスでコンポーネントを起動する

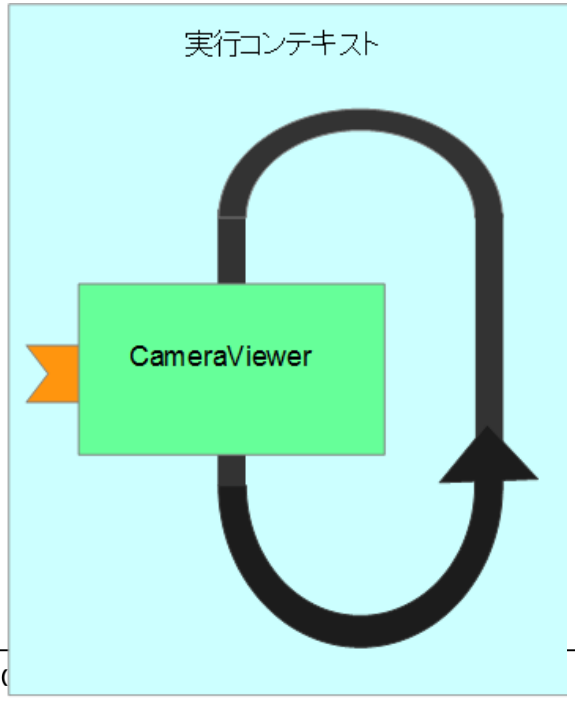


RTシステムを作成する手順について

- 講習会のFlipコンポーネント動作確認の手順でRTシステムを作成した場合
  - コンポーネントをそれぞれ別の実行コンテキストで非同期に駆動するため、Flipがデータを受信していないタイミングでonExecuteコールバックを呼び出すことがある
  - OpenCVCameraで画像をキャプチャしてからCameraViewerで画像を表示するまでの時間が不明
    - 実行のタイミング次第では時間がかかる可能性もある
    - リアルタイムが要求される場合は実行の同期が必須

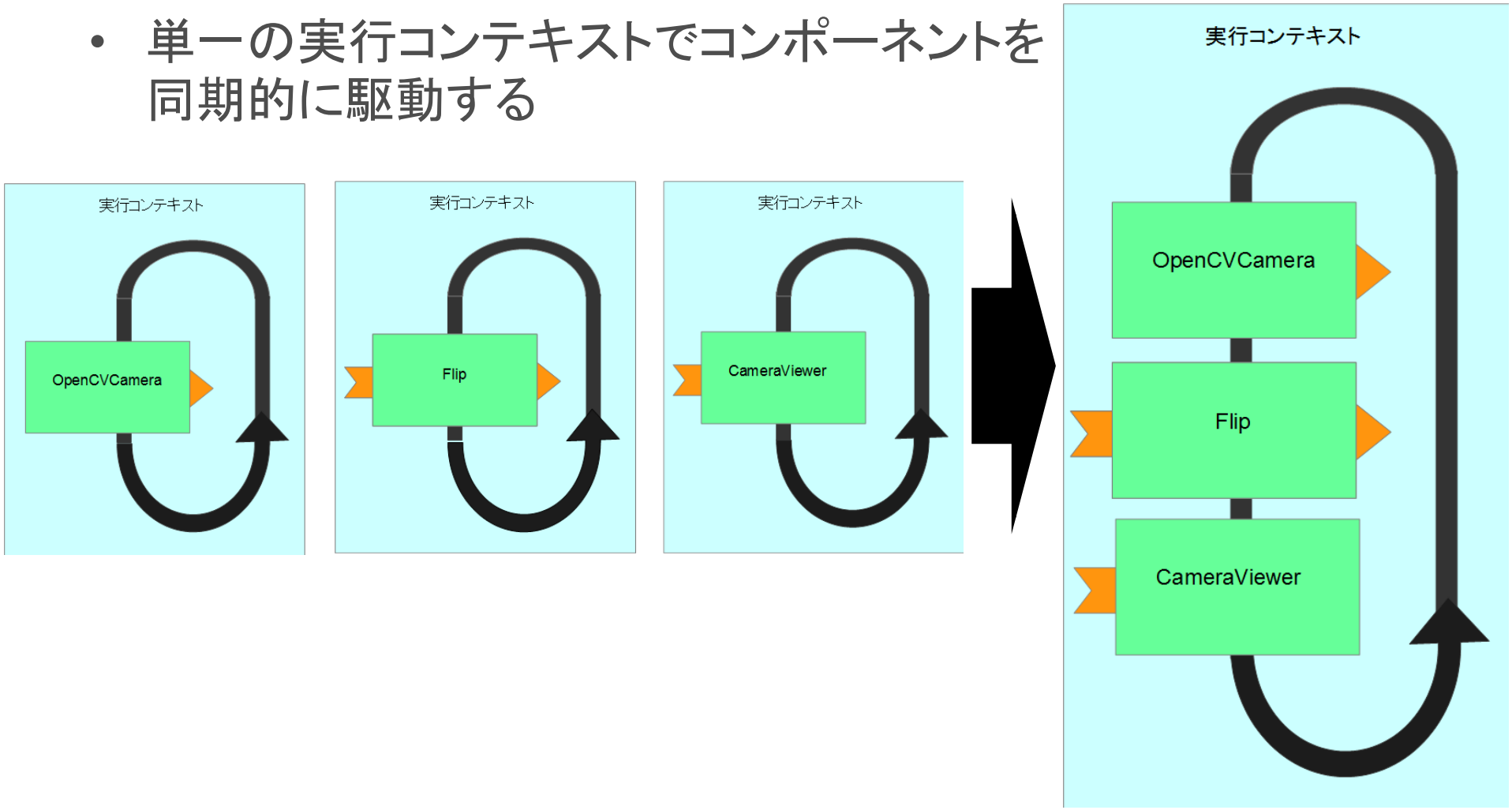






RTシステムを作成する手順について

- 単一の実行コンテキストでコンポーネントを同期的に駆動する

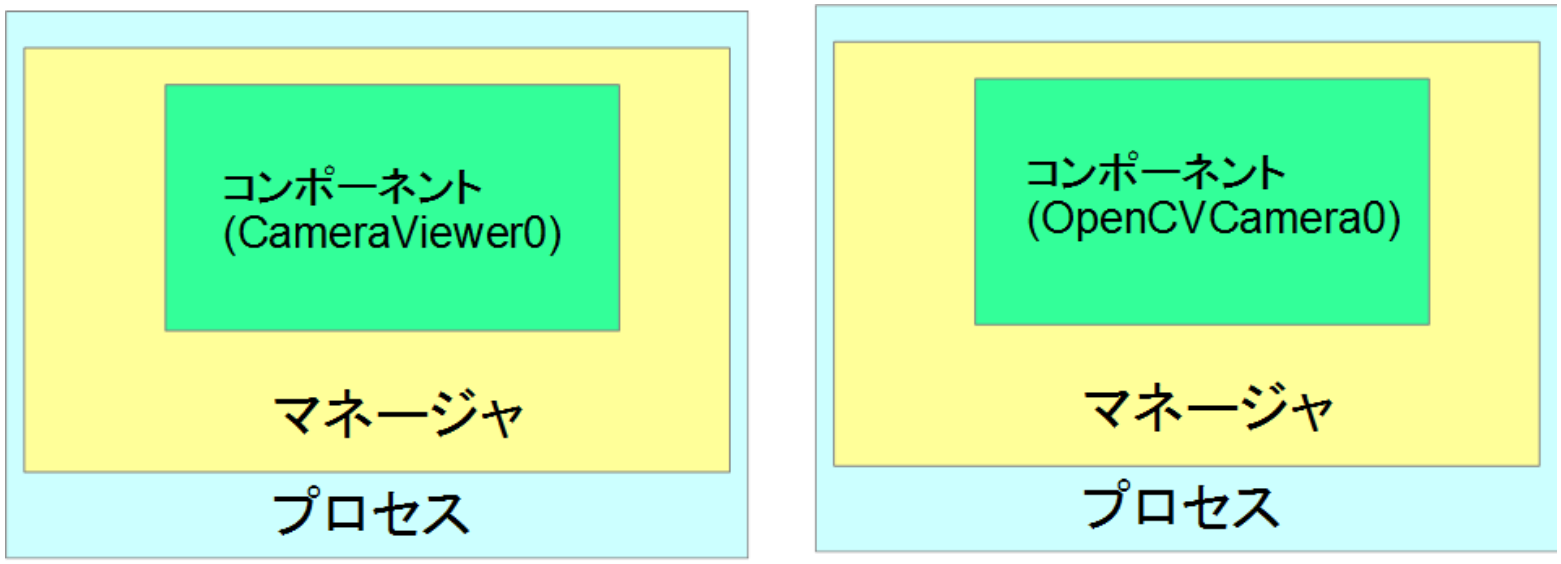


目次

- 同一プロセスでコンポーネントを起動する方法について
- 複数のコンポーネントを単一の実行コンテキストで駆動させる方法について
- 複合コンポーネントの生成方法について
- ツールの紹介
  - 複合コンポーネント作成支援ツール
  - 実行順序設定可能な実行コンテキスト

同一プロセスでコンポーネントを起動する方法

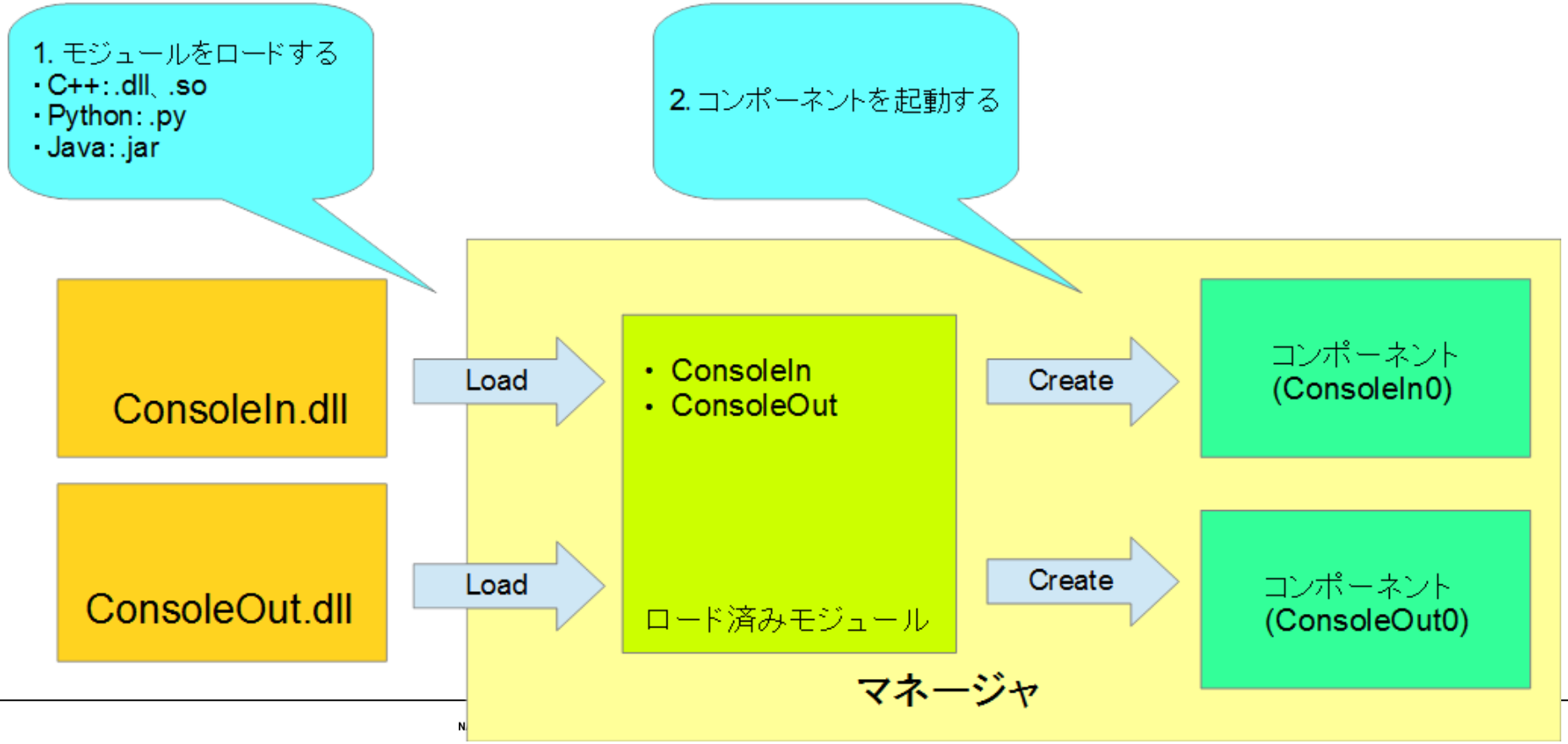
- マネージャについて
  - RTCの管理を行うサービス
    - モジュールのロード
    - コンポーネントの起動、破棄等
  - 1つのプロセスで1つのマネージャが起動



基本的にマネージャは各プロセスに1つ起動する。
マネージャがコンポーネントを起動する

同一プロセスでコンポーネントを起動する方法

- マネージャについて
  - ダイナミックリンクライブラリ(.dll、.so)、Pythonファイル(.py)、JARファイル(.jar)を読み込み、ロードしたモジュールからコンポーネントを起動する。
    - → 同一プロセスでコンポーネントを起動できる
      - 同一プロセスでコンポーネントを起動することで、データポート・サービスポートによる通信が高速になる



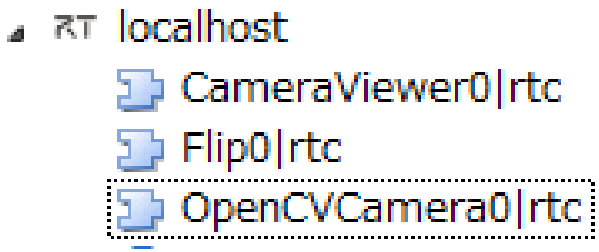
同一プロセスでコンポーネントを起動する方法

- rtc.confの編集
  - RTCビルダによるスケルトンコード作成時に生成されている
  - RTコンポーネント、マネージャ、ロガーなどの設定を行う
    - マネージャ起動時に読み込む
  - モジュール探索パス、起動時にロードするモジュール、起動するコンポーネントの設定を行う
    - モジュール探索パス
      - manager.modules.load_path: ., components/
    - 起動時にロードするモジュール
      - manager.modules.preload: OpenCVCamera.dll, CameraViewer.dll, Flip.dll
    - 起動するコンポーネント
      - manager.components.precreate: OpenCVCamera, CameraViewer, Flip
  - 今回は見本を作っているのでコメント記号(#)を消して動作確認を行う
    - 資料の「動作確認」→「Windows」、「Ubuntu」フォルダ内
    - 該当箇所のコメント記号(#)を消してください

```
4  ↓
5  ↓
6  #同一プロセスで複数のコンポーネントを起動するのに必要な設定↓
7  #モジュール探索パスの設定↓
8  #manager.modules.load_path: ., components/↓
9  #事前にロードするモジュールの設定↓
10 #manager.modules.preload: OpenCVCamera.dll, CameraViewer.dll, Flip.dll↓
11 #起動時に生成するコンポーネントの設定↓
12 #manager.components.precreate: OpenCVCamera, CameraViewer, Flip↓
13 ↓
```

同一プロセスでコンポーネントを起動する方法

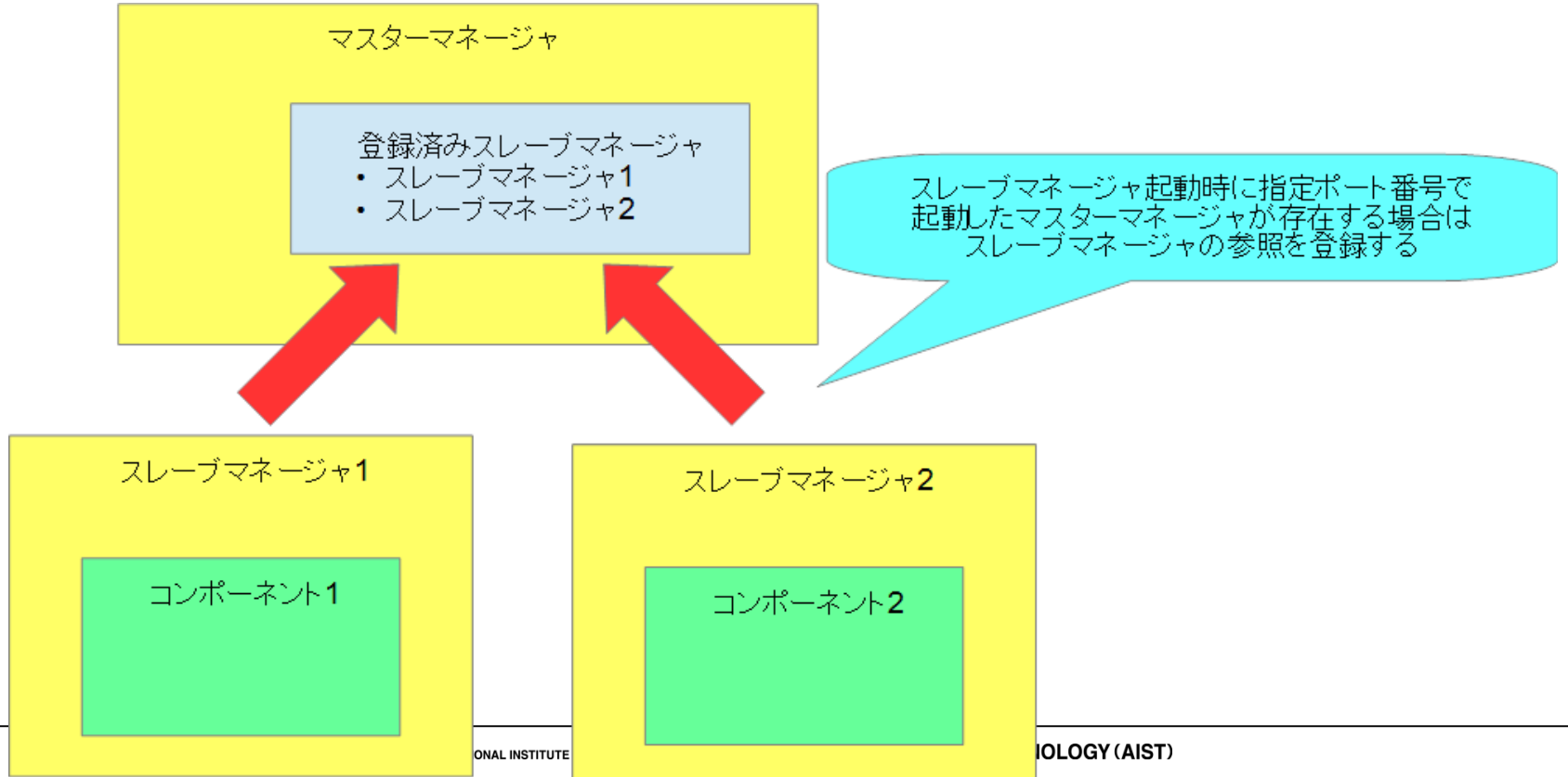
- rtc.confの編集
  - 編集が終わったらrtcd.exeを起動する
    - OpenCVCamera、CameraViewer、Flipが同時に起動する



  - 起動したことを確認したら、コンポーネントをexitして一旦終了してください。

同一プロセスでコンポーネントを起動する方法

- マスターマネージャとスレーブマネージャ
  - マスターマネージャ
    - 外部のツール(RTシステムエディタ等)から操作可能なマネージャ
  - スレーブマネージャ
    - 何も設定しなければ起動するマネージャ
    - マスターマネージャから操作が可能
    - 基本的にはマスターマネージャを介さない限り外からの操作はできない



同一プロセスでコンポーネントを起動する方法

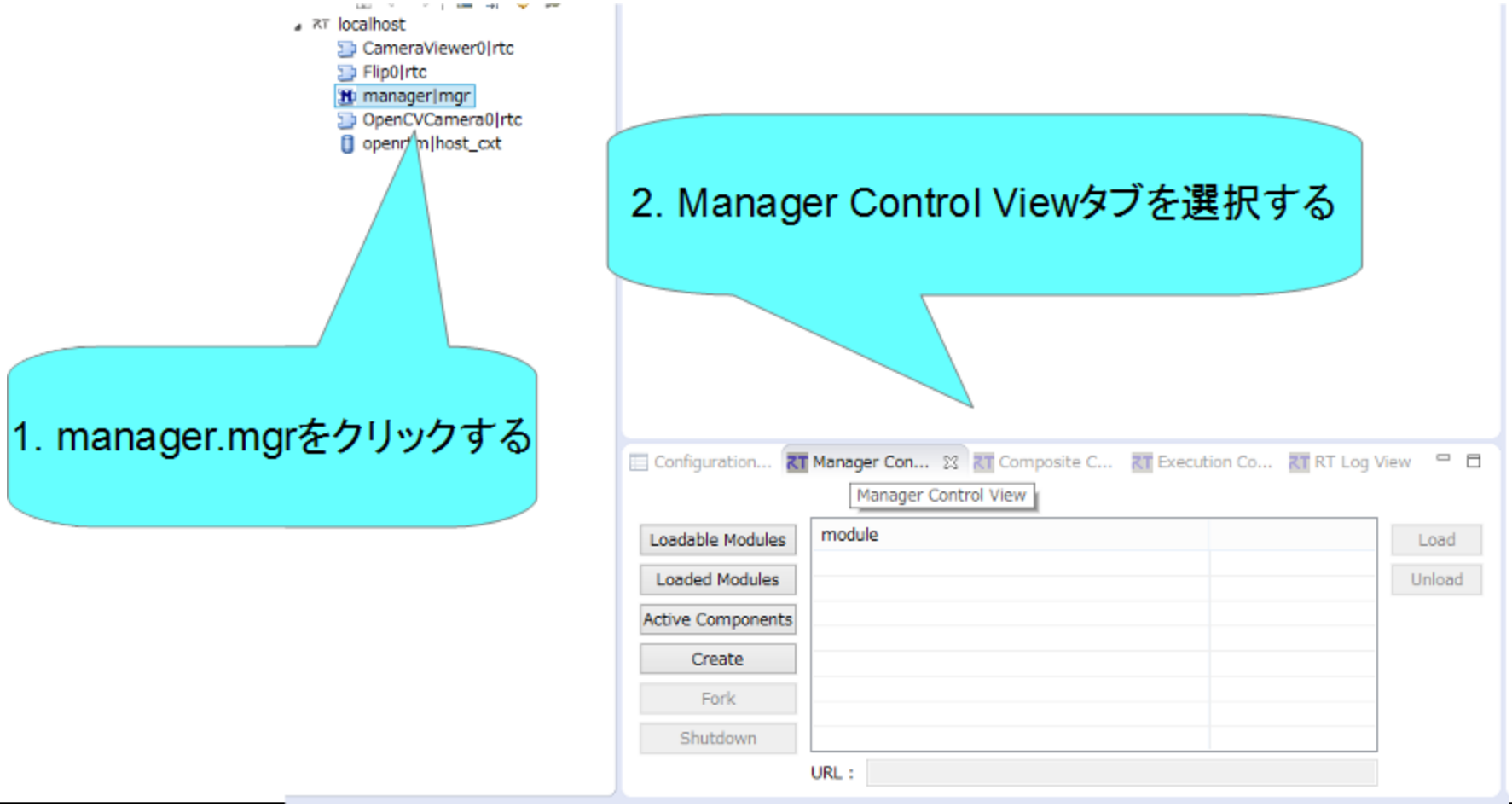
- マスターマネージャの起動手順
  - rtc.confを編集する
    - manager.is_master: YES
    - 先ほどと同じく、該当箇所のコメント記号(#)を削除

```
17 #マネージャに関する設定↓
18 #マネージャのネームサーバーへの登録フォーマット設定↓
19 manager.naming_formats: %n.mgr↓
20 #マスターマネージャに設定↓
21 #manager.is_master: YES↓
```

  - rtcd.exeを起動する
    - ネーミングサービスにmanager.mgrが登録される
    - corbaloc形式によるアクセスが可能になる
      - obj = orb.string_to_object("corbaloc::localhost:2810/manager")
      - ネームサーバーと同様に、アドレスとポート名でアクセス

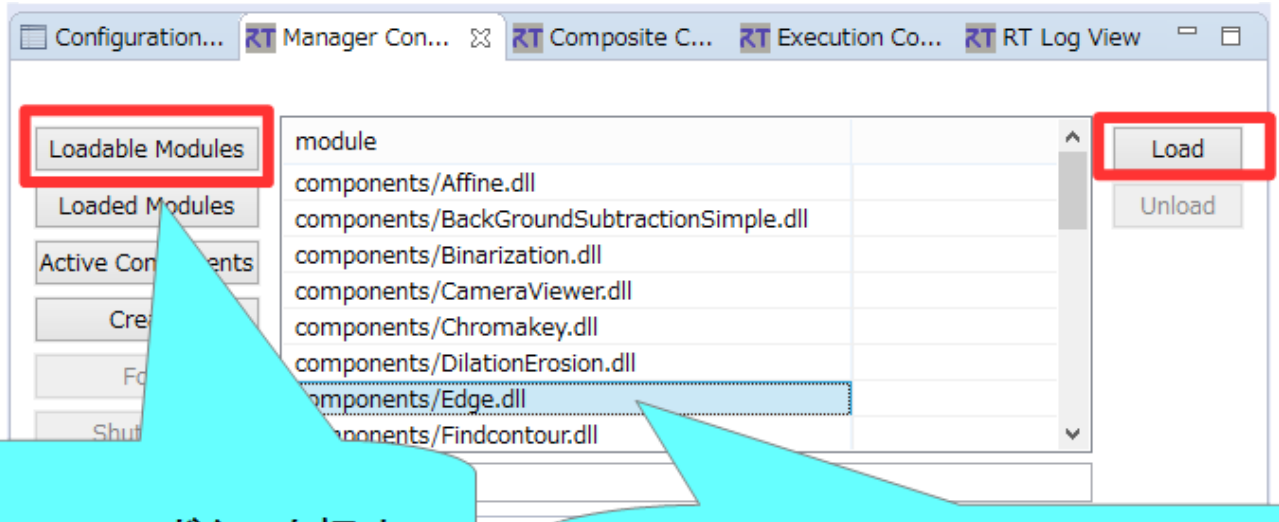
同一プロセスでコンポーネントを起動する方法

- RTシステムエディタによるマスターマネージャの操作手順
  - Manager Control Viewを開く



同一プロセスでコンポーネントを起動する方法

- RTシステムエディタによるマスターマネージャの操作手順
  - モジュールをロードする

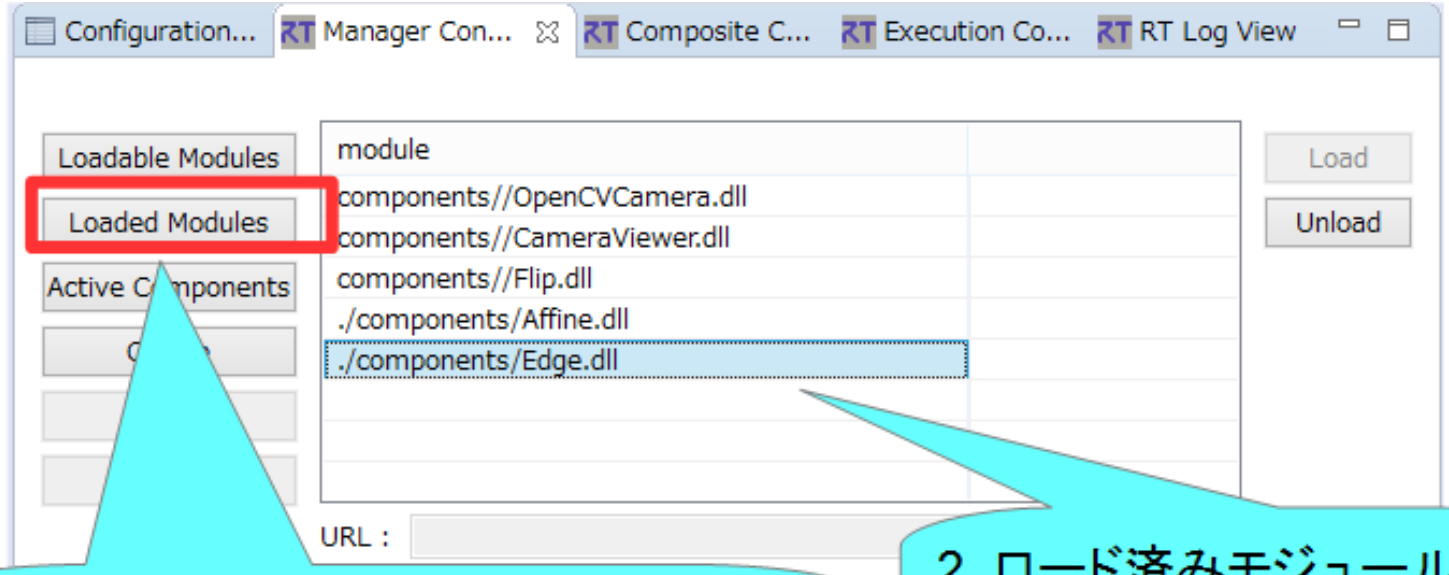


1. Loadable Modulesボタンを押す

2. ロード可能なモジュール一覧からロードしたいモジュールを選択してLoadボタンを押す
今回はEdge.dllをロードする

同一プロセスでコンポーネントを起動する方法

- RTシステムエディタによるマスターマネージャの操作手順
  - ロード済みのモジュールを確認する

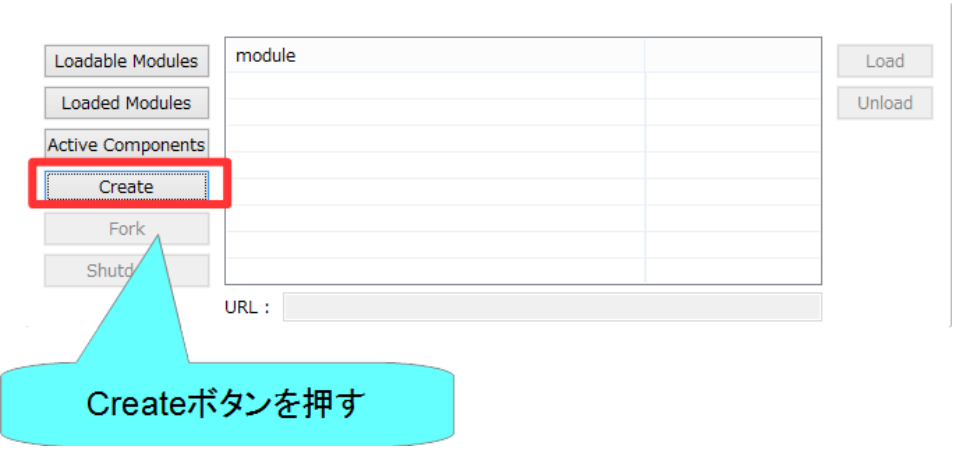


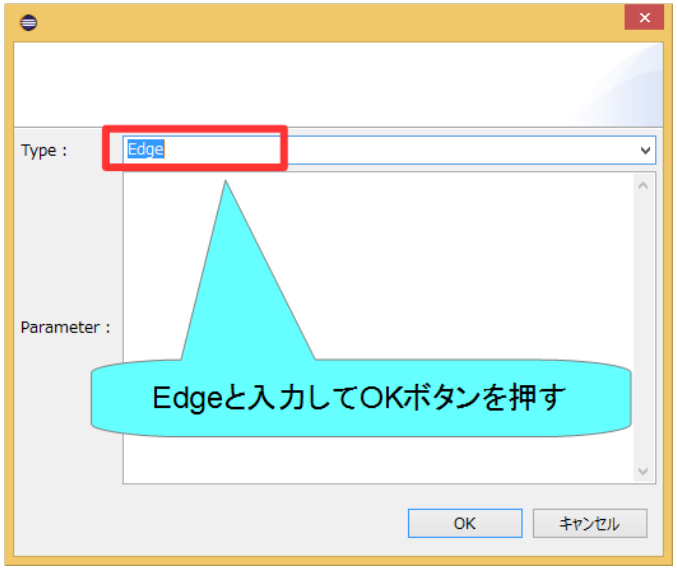
1. Loaded Modulesボタンを押す

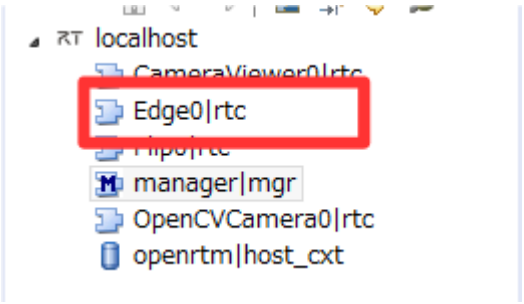
2. ロード済みモジュールの一覧が表示
先ほどロードしたEdge.dllが存在するかを
確認してください

同一プロセスでコンポーネントを起動する方法

- RTシステムエディタによるマスターマネージャの操作手順
  - コンポーネントを生成する

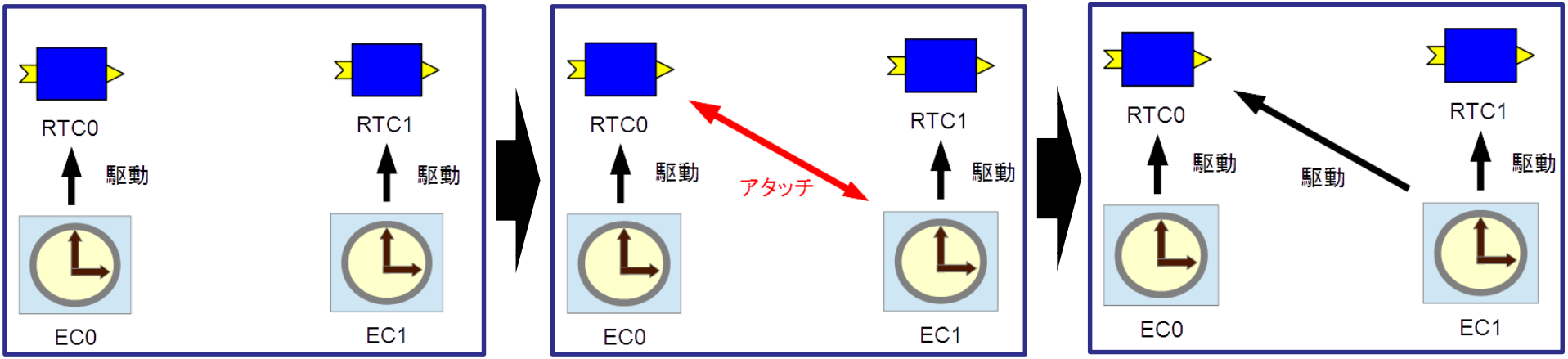






複数のコンポーネントを単一の実行コンテキストで駆動させる方法について

- 実行コンテキストとの関連付け
  - 何も設定をしない場合、RTコンポーネントはそれぞれ別々の実行コンテキストによって駆動される
    - コンポーネント生成時に実行コンテキストも生成するため、通常はこの実行コンテキストに関連付けられている
  - RTコンポーネントと実行コンテキストを関連付けることで実行を同期させることができる
    - コンポーネントは関連付けた実行コンテキストごとに別々の状態を持っている
      - RTシステムエディタのSystem Diagram上で表示しているのはIDが0の実行コンテキストでの状態



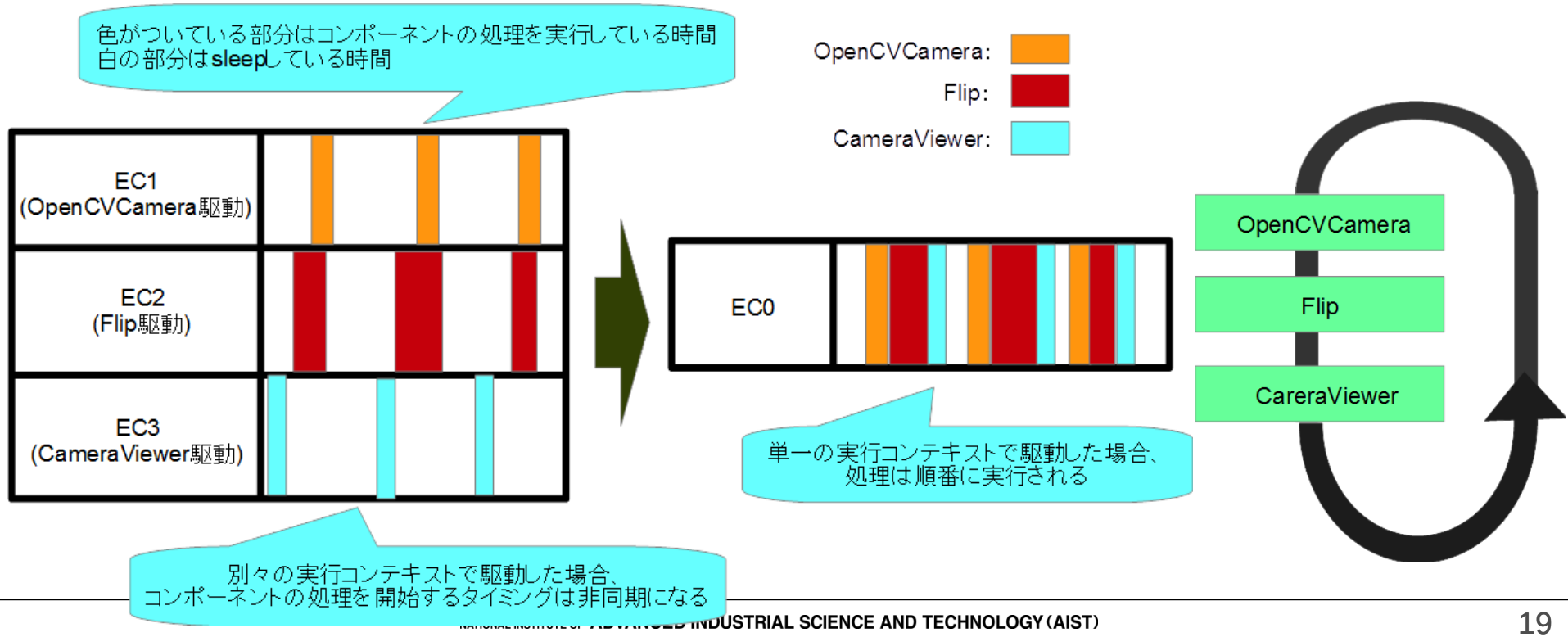
RTC0とRTC1は別々の実行コンテキストで駆動

RTC0とEC1を関連付ける

EC1がRTC0も駆動する

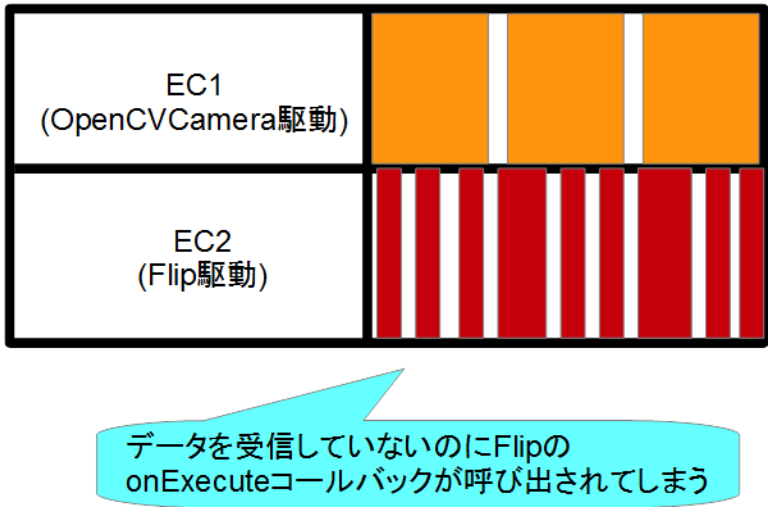
複数のコンポーネントを単一の実行コンテキストで駆動させる方法について

- 実行コンテキストとの関連付け
  - PeriodicExecutionContext(周期実行の実行コンテキスト)の場合、直列にコンポーネントのロジックが実行される
    - OpenCVCamera、Flip、CameraViewerを直列に実行する
      - OpenCVCameraでキャプチャした画像をOutPortから出力する
      - Flipでインポートから画像データを受信し、反転画像をアウトポートから出力する
      - CameraViewerでInPortから受信した画像データを表示する

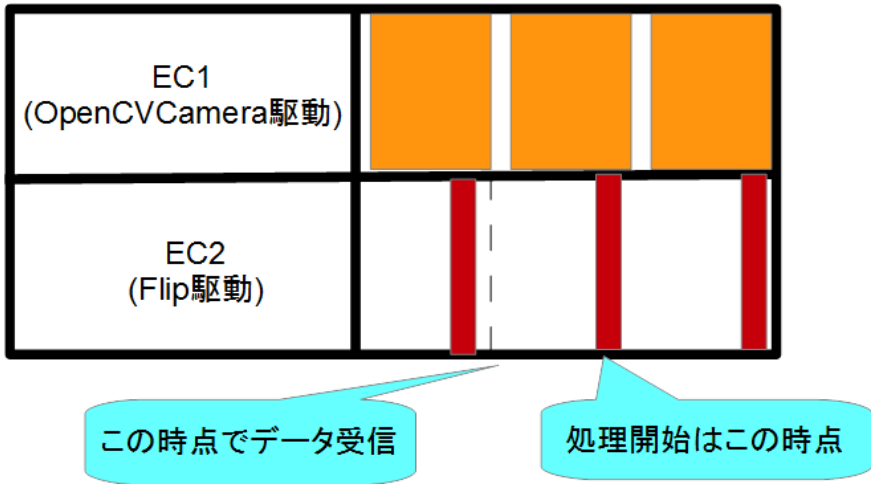


複数のコンポーネントを単一の実行コンテキストで駆動させる方法について

- 別々の実行コンテキストで駆動した場合の問題点
  - Flipコンポーネントの動作に比べて OpenCVCameraの処理が**遅い**場合
    - データを受信していないのにonExecuteコールバックが呼び出されるため効率が悪い

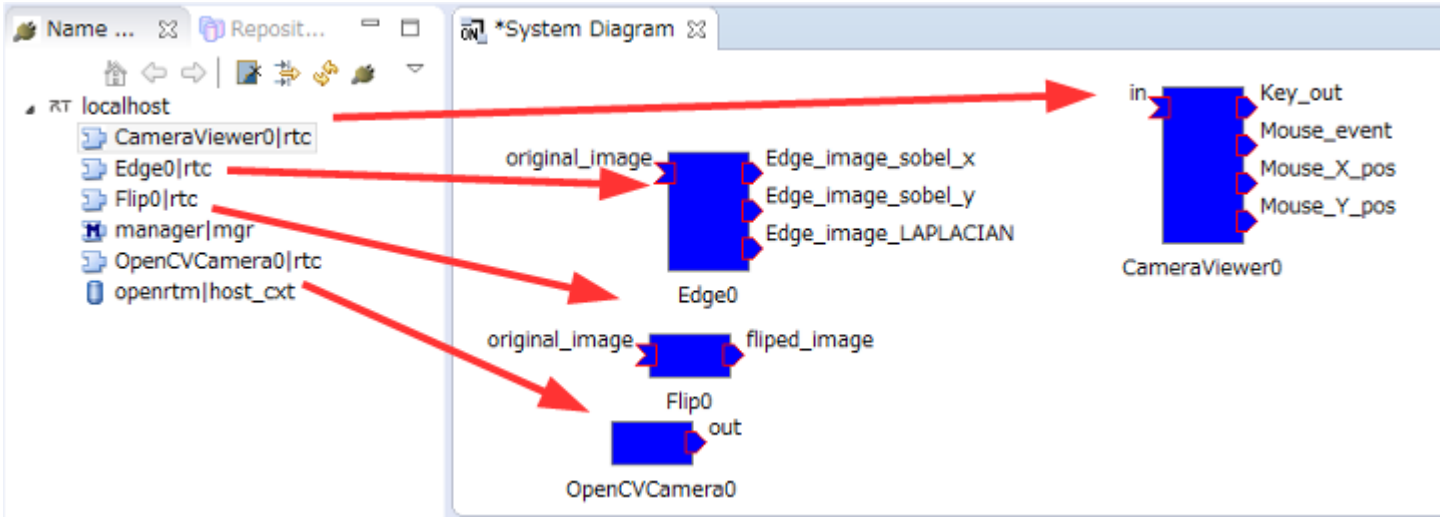


- 実行周期を処理に時間がかかる側に合わせた場合、処理の開始が遅れる可能性がある
  - → OpenCVCameraの処理直後にFlipを実行したい



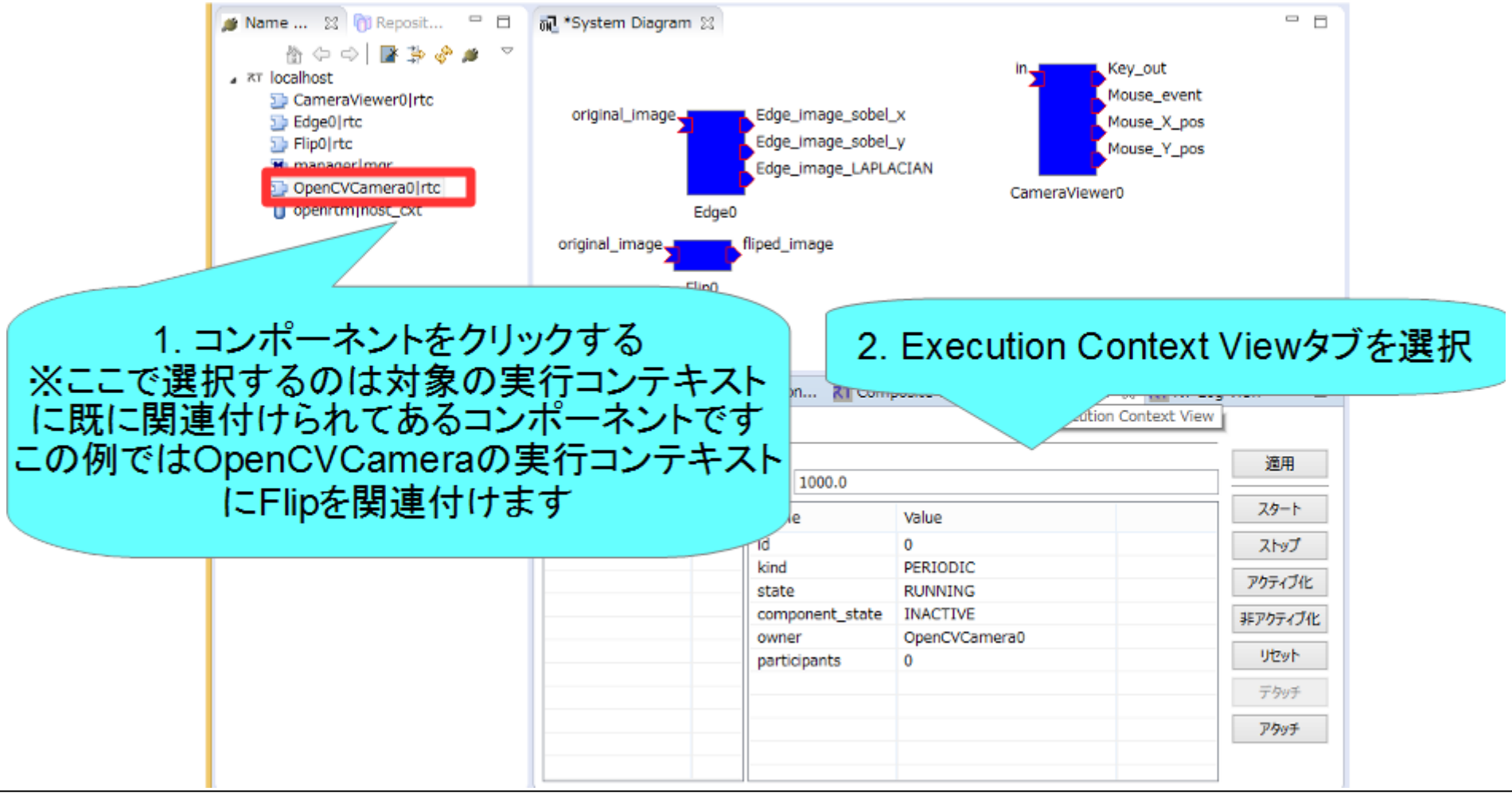
複数のコンポーネントを単一の実行コンテキストで駆動させる方法について

- RTシステムエディタの操作
  - System Diagram上にコンポーネントをドラッグアンドドロップする
    - ※理由は分かりませんが、System Diagram上に表示しないとアタッチするコンポーネント一覧にコンポーネント名が表示されません。



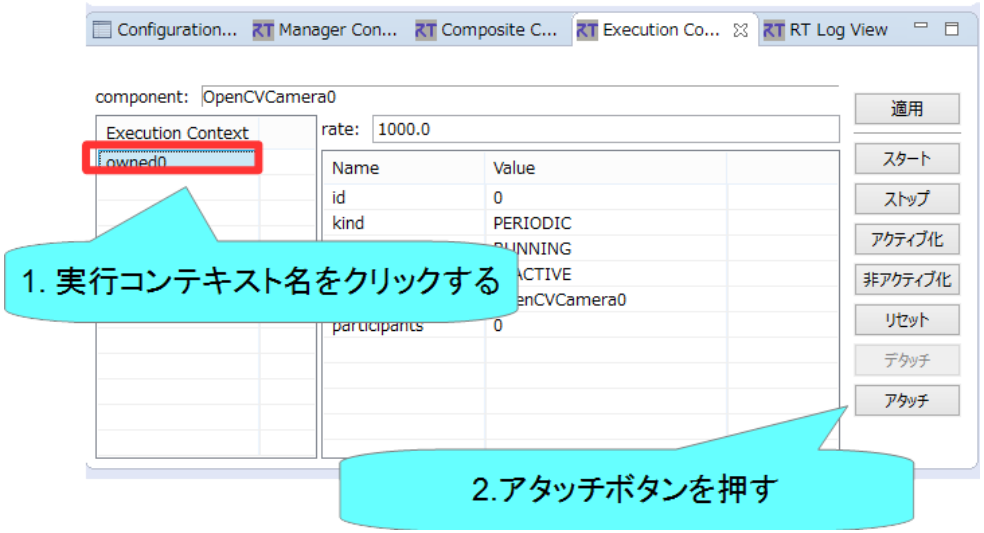
複数のコンポーネントを単一の実行コンテキストで駆動させる方法について

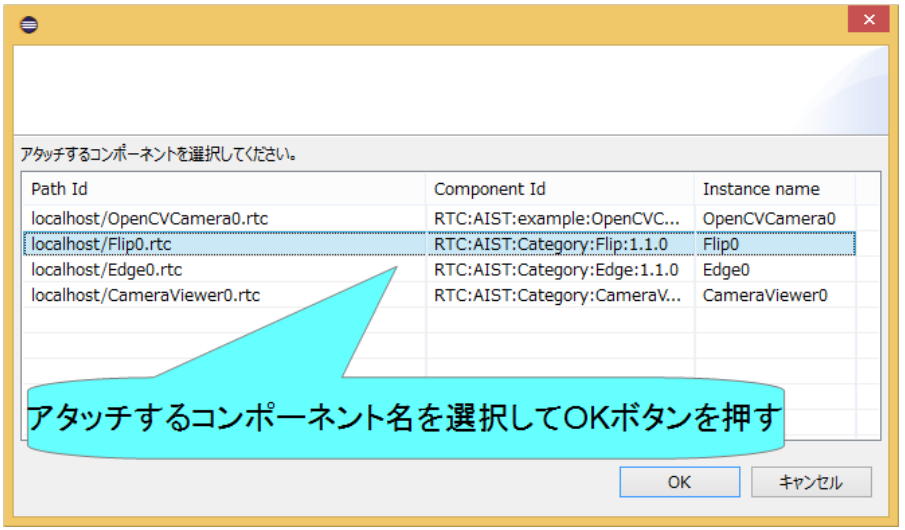
- RTシステムエディタの操作
  - Execution Context Viewを開く



複数のコンポーネントを単一の実行コンテキストで駆動させる方法について

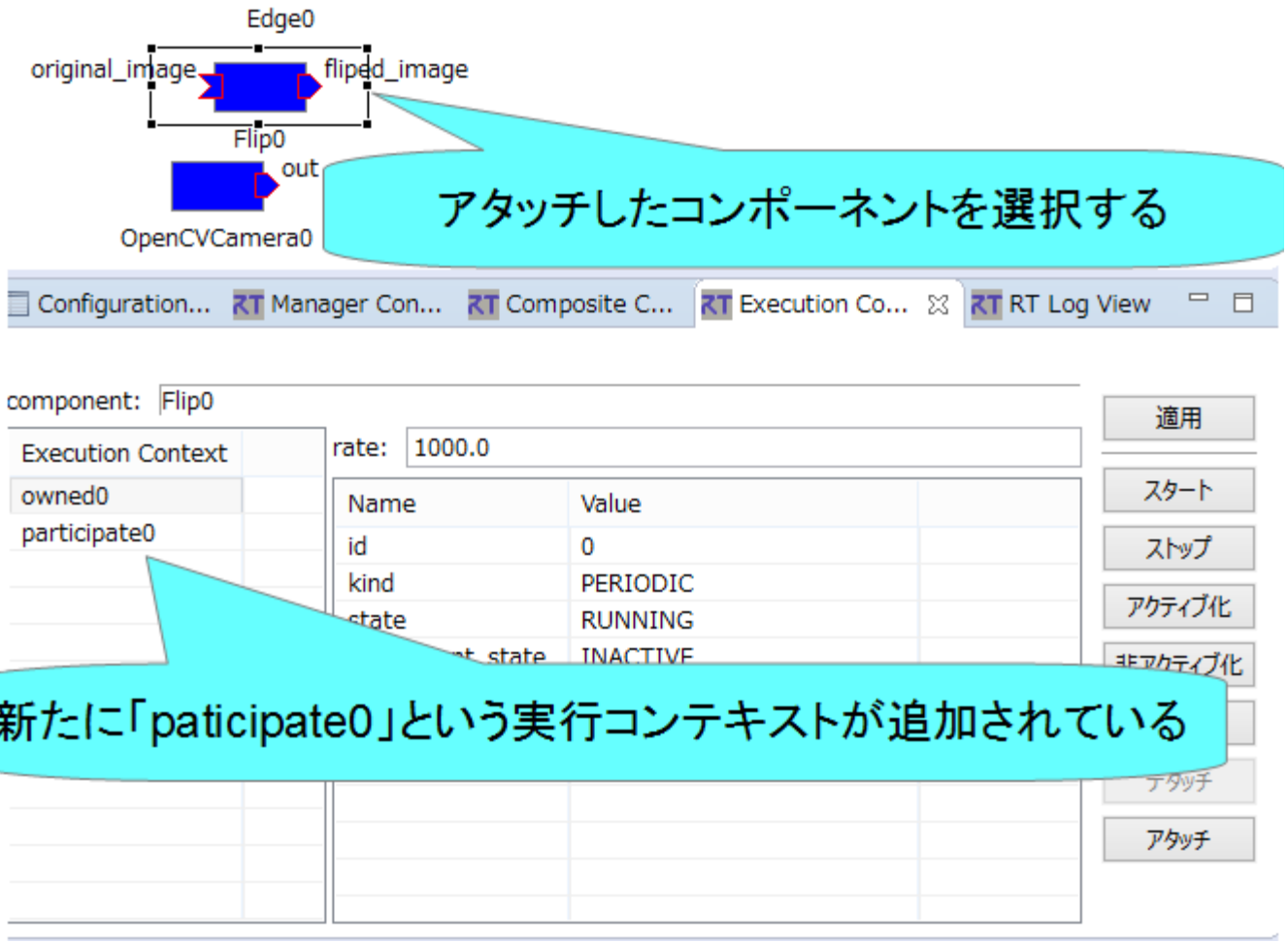
- RTシステムエディタの操作
  - 実行コンテキストにコンポーネントをアタッチする





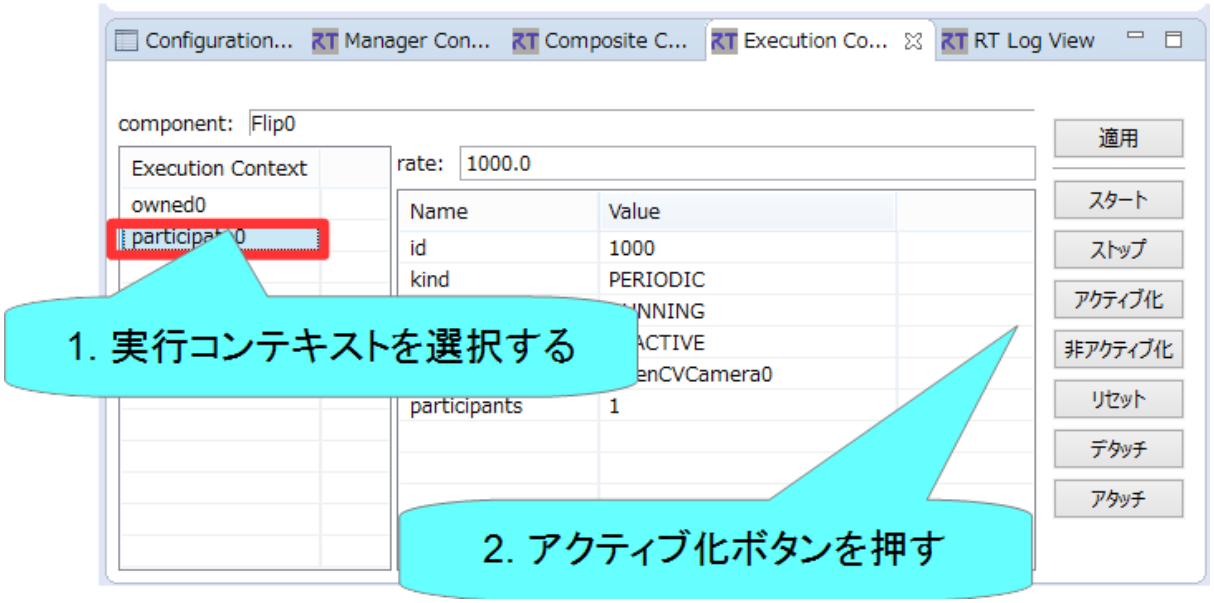
複数のコンポーネントを単一の実行コンテキストで駆動させる方法について

- RTシステムエディタの操作
  - 実行コンテキストにコンポーネントをアタッチできたかの確認



複数のコンポーネントを単一の実行コンテキストで駆動させる方法について

- RTシステムエディタの操作
  - コンポーネントのアクティブ化
    - アタッチした実行コンテキストでコンポーネントをアクティブ化する場合、Execution Context Viewから操作する必要がある
      - ※All Activateボタン等でアクティブ化されるのはIDが0の実行コンテキストでの状態なので注意
      - System Diagram上の表示が青いので一見してアクティブ状態に遷移していないように見えますが、実行コンテキストparticipate0上ではアクティブ状態に遷移しています

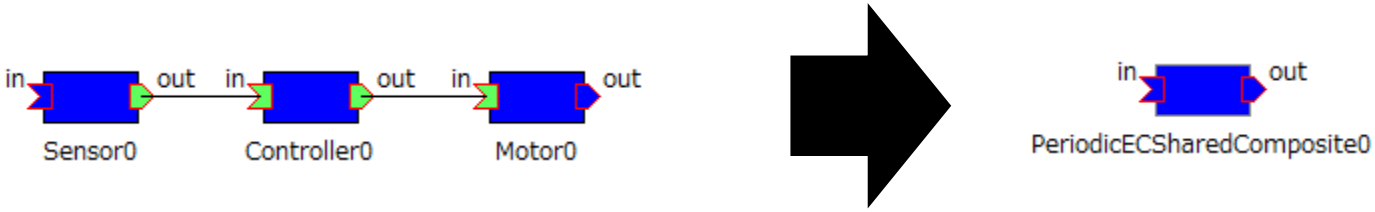


補足

- 実行コンテキストの指定方法について
  - コンポーネントはCreated状態の時に実行コンテキストを生成する
  - 生成される実行コンテキストはrtc.confで指定する
    - exec_cxt.periodic.type: PeriodicExecutionContext
  - 以下のような実行コンテキストを指定可能
    - PeriodicExecutionContext
      - 周期実行の実行コンテキスト
    - ExtTrigExecutionContext
      - 外部トリガによる実行コンテキスト
    - SynchExtTriggerEC
      - 外部トリガによる実行コンテキスト
    - ArtExecutionContext
      - ART-Linuxによる実時間周期実行の実行コンテキスト
    - RTPreemptEC
      - RT-Preemptive Linuxによる実時間周期実行の実行コンテキスト
    - TkPeriodicExecutionContext
      - T-Kernel上で動作するOpenRTM-aist用の実行コンテキスト(詳細はよく知りません)
    - StepwiseEC
      - 他のRTCと協調動作を行うための実行コンテキスト
    - ArtDataSyncEC
      - 使ったことないので内容は不明です
    - EmptyExecutionContext
      - OpenRTM.NETに存在する実行コンテキスト

複合コンポーネントの生成方法について

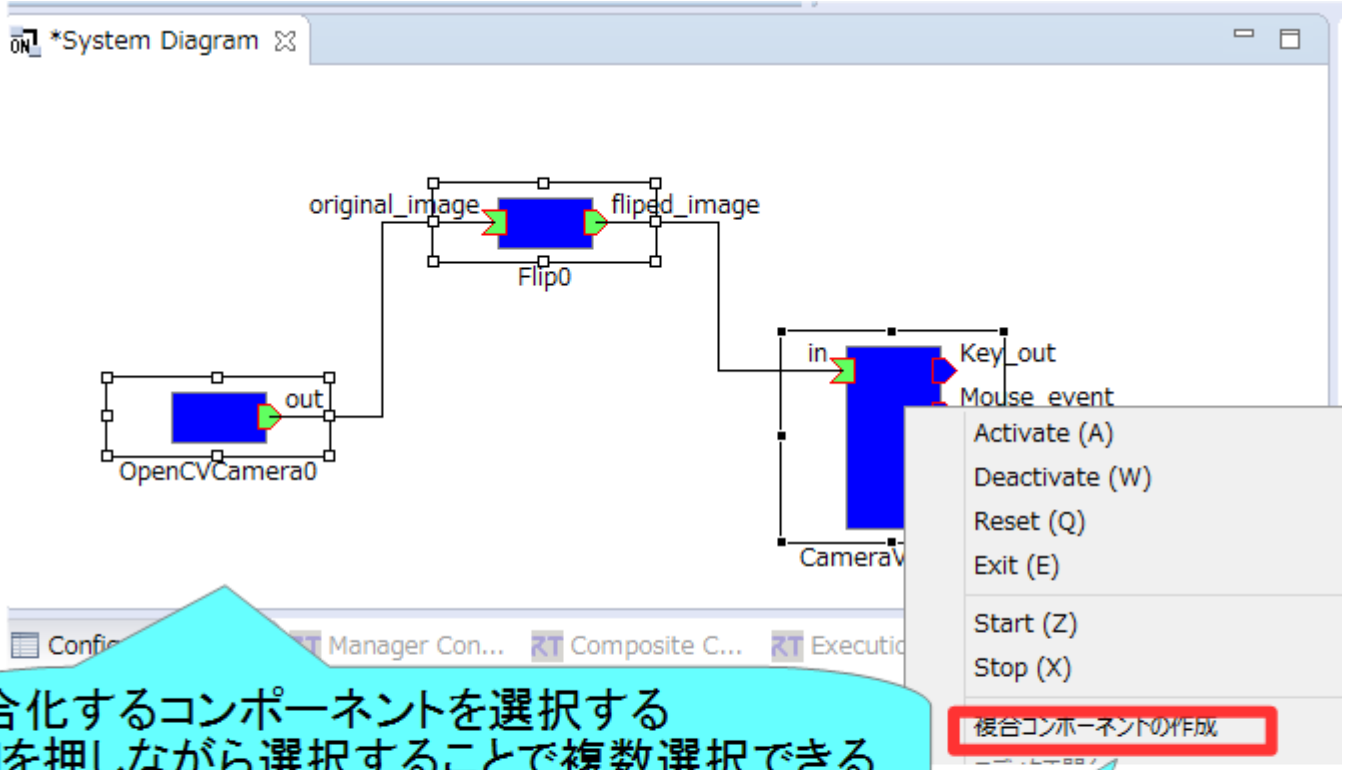
- 複合コンポーネントについて
  - 複数のコンポーネントを合成してひとつのコンポーネントにする手法



  - 子コンポーネントの指定のポートのみ複合コンポーネントで表示
    - 内部の複雑なシステムを隠蔽して、必要な部分のみを見えるようにする
  - 複合コンポーネントの実行コンテキストに子コンポーネントに関連付ける
    - 実行の同期

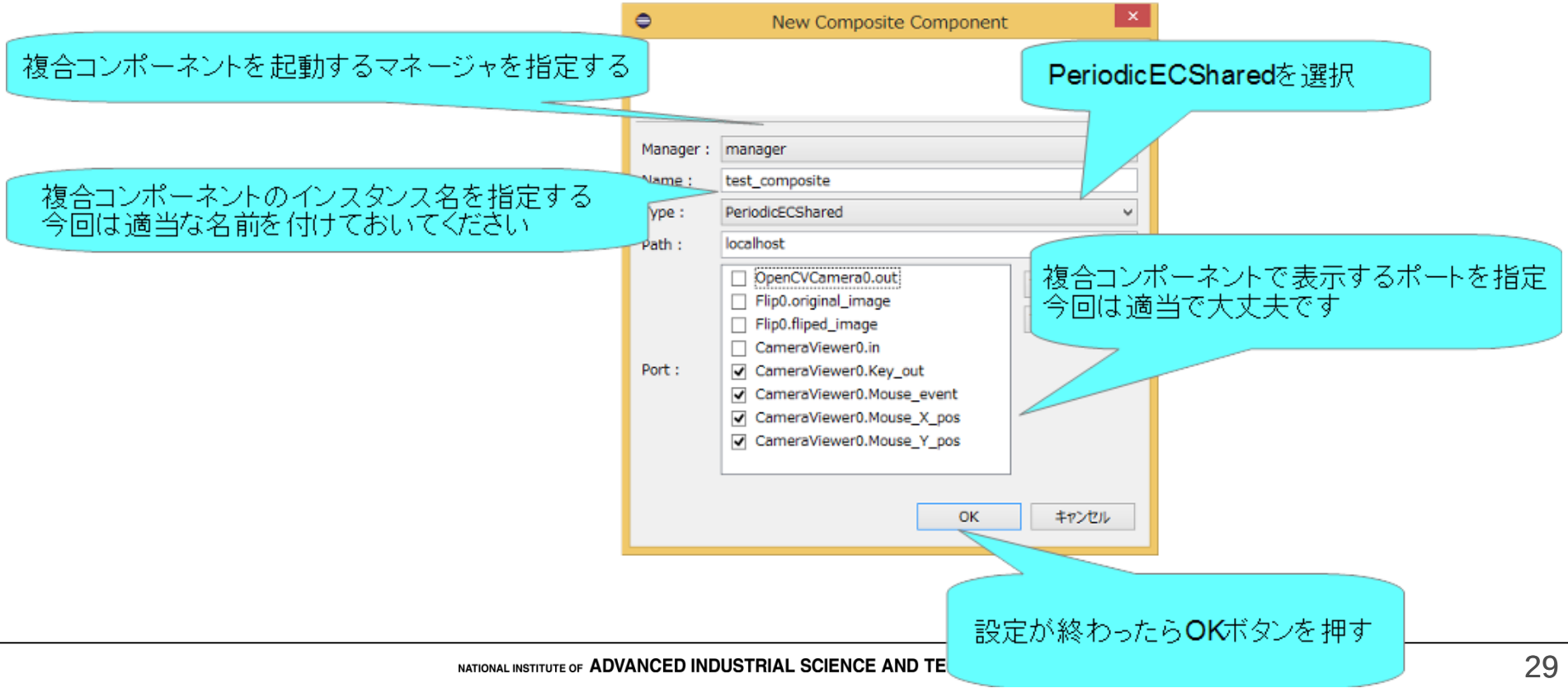
複合コンポーネントの生成方法について

- RTシステムエディタの操作手順
  - 複合コンポーネントの生成



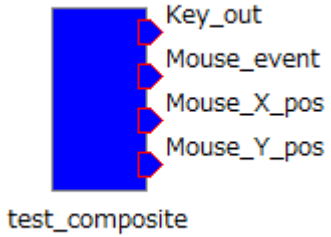
複合コンポーネントの生成方法について

- RTシステムエディタの操作手順
  - 複合コンポーネントの生成
    - 複合コンポーネントの種類について
      - 以下の3種類から指定可能
        - » PeriodicECShared(実行コンテキスト共有)
        - » PeriodicStateShared(実行コンテキスト、状態共有)
        - » Grouping(見た目のみ複合化)

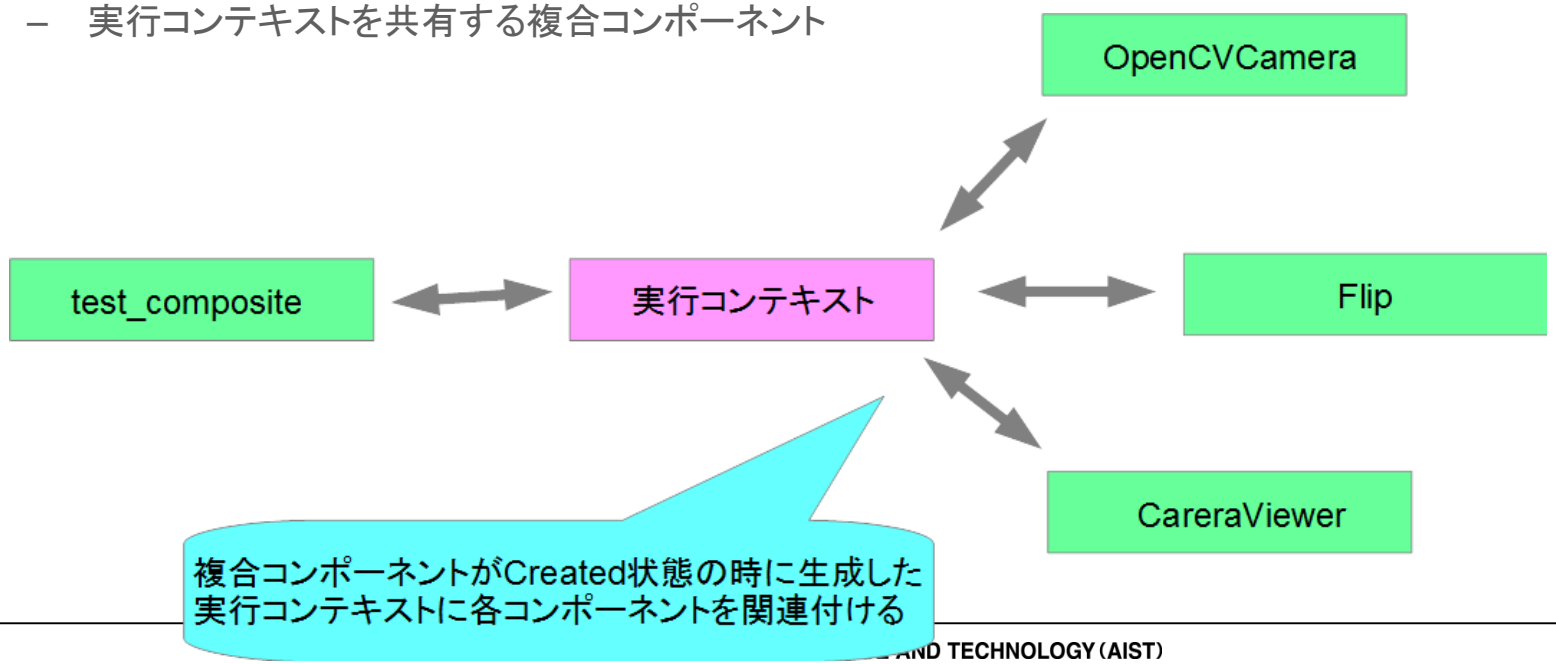


複合コンポーネントの生成方法について

- RTシステムエディタの操作手順
  - 以上の手順で複合コンポーネントが生成される

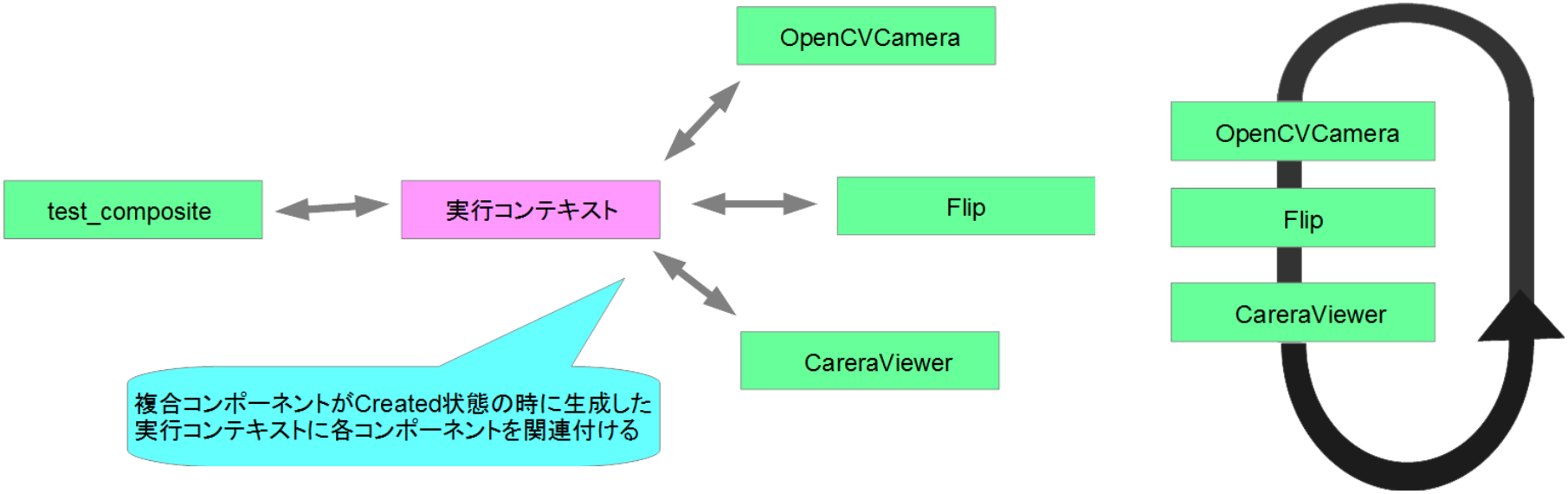


- PeriodicECSharedの複合コンポーネントについて
  - 実行コンテキストを共有する複合コンポーネント



複合コンポーネントの生成方法について

- RTシステムエディタの操作手順
  - 複合コンポーネントをアクティブ化すると、子コンポーネントもアクティブ状態に遷移する
  - 実行コンテキストを共有しているため、OpenCVCamera、Flip、CameraViewerを直列実行する



複合コンポーネントの生成方法について

- rtc.confで設定する方法
  - OpenRTM-aist付属の複合コンポーネントのサンプルはこの手順で複合コンポーネントを起動している
  - 複合コンポーネントを起動する
    - manager.components.precreate: OpenCVCamera, CameraViewer, Flip, PeriodicECSharedComposite?&instance_name=test_composite
    - PeriodicECSharedCompositeが複合コンポーネント
    - ?&instance_name=＊＊＊でインスタンス名を指定
  - 複合化する子コンポーネントの設定
    - composite.PeriodicECShared.PeriodicECSharedComposite.conf.default.members: OpenCVCamera0, CameraViewer0, Flip0
    - コンフィギュレーションパラメータmemberで設定
  - 表示するポートの設定
    - composite.PeriodicECShared.PeriodicECSharedComposite.conf.default.exported_ports: CameraViewer0.Key_out, CameraViewer0.Mouse_event, CameraViewer0.Mouse_X_pos, CameraViewer0.Mouse_Y_pos
    - コンフィギュレーションパラメータexported_portsで設定
  - 該当箇所のコメント記号(#)を削除する

```
31 ↓
32 #複合コンポーネントに関する設定↓
33 #起動時に生成するコンポーネントの設定↓
34 #manager.components.precreate: OpenCVCamera, CameraViewer, Flip, PeriodicECSharedComposite?&instance_name=test_
35 #子コンポーネントの設定↓
36 #composite.PeriodicECShared.PeriodicECSharedComposite.conf.default.members: OpenCVCamera0, CameraViewer0, Flip0
37 #表示するポートの設定↓
38 #composite.PeriodicECShared.PeriodicECSharedComposite.conf.default.exported_ports: CameraViewer0.Key_out, Camera
```

補足

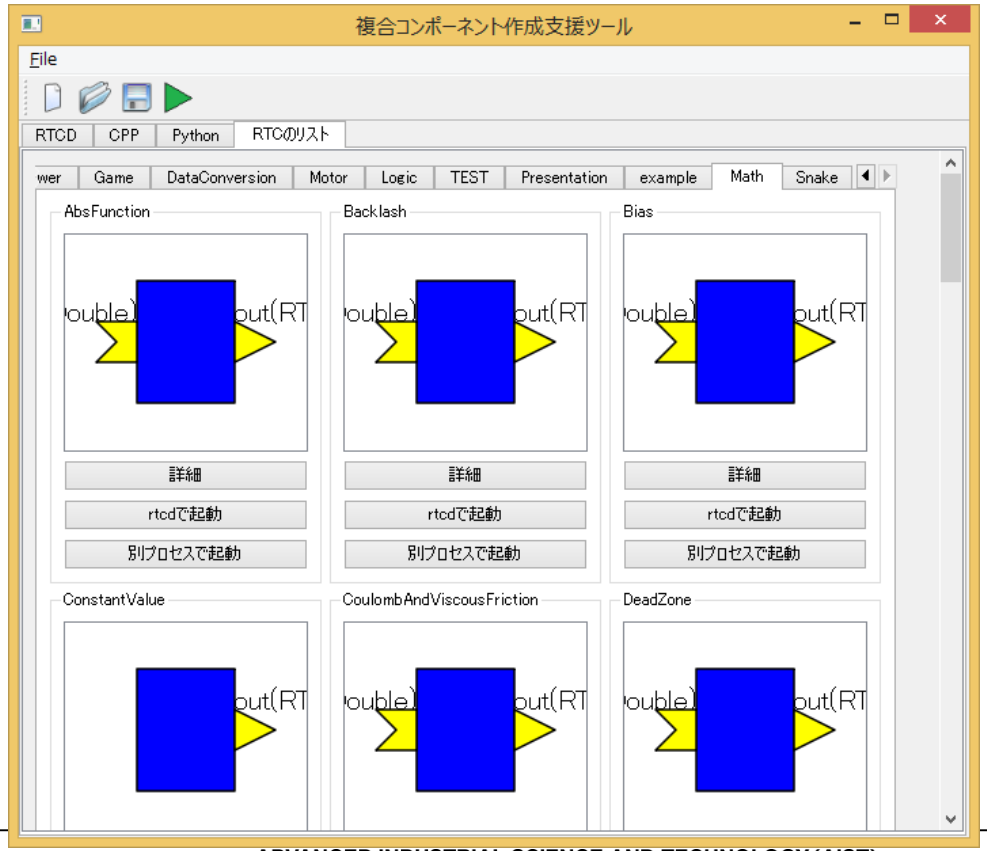
- マスターマネージャの終了手順について
  - 現在のところ、Manager Control ViewのShutdownボタンは絶対に押せないようになっているようなので、自分で操作するプログラムを書くしかありません。
  - 以下はPythonの例
    - corbaloc形式でアクセスする

```
#!/usr/bin/python
# -*- coding: utf-8 -*-
import sys


from omniORB import CORBA
import CosNaming
import RTC,RTM,RTM__POA

orb = CORBA.ORB_init(sys.argv, CORBA.ORB_ID)
obj = orb.string_to_object("corbaloc::localhost:2810/manager")
manager = obj._narrow(RTM__POA.Manager)
manager.shutdown()
```

  - マネージャを終了させると、コンポーネントもまとめて終了します

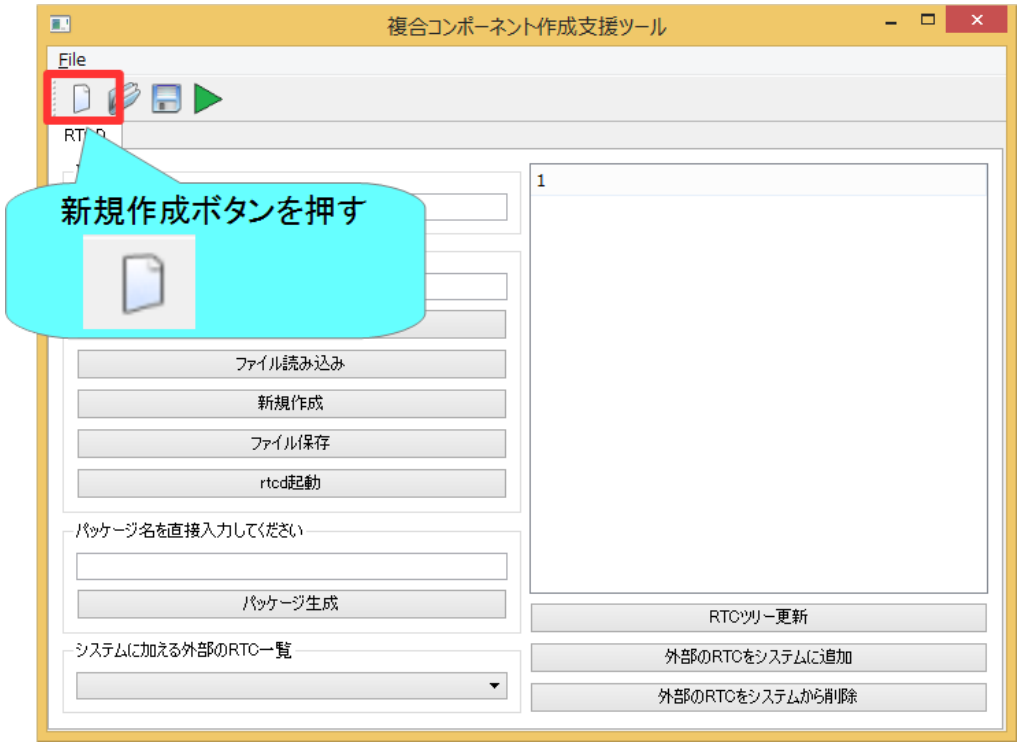
複合コンポーネント作成支援ツールについて

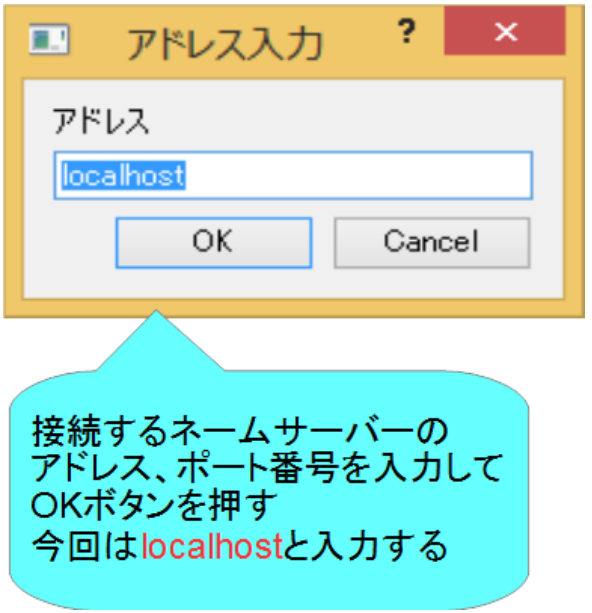
- コンポーネントの起動、各種設定、複合化、RTシステムの復元までを自動化するツール
  - 資料の「複合コンポーネント作成支援ツールのマニュアル」フォルダ内のショートカットからマニュアルを見ることができます
    - マニュアルのリンクからダウンロードできます
    - 適当な場所にZIPファイルを展開してください



複合コンポーネント作成支援ツールについて

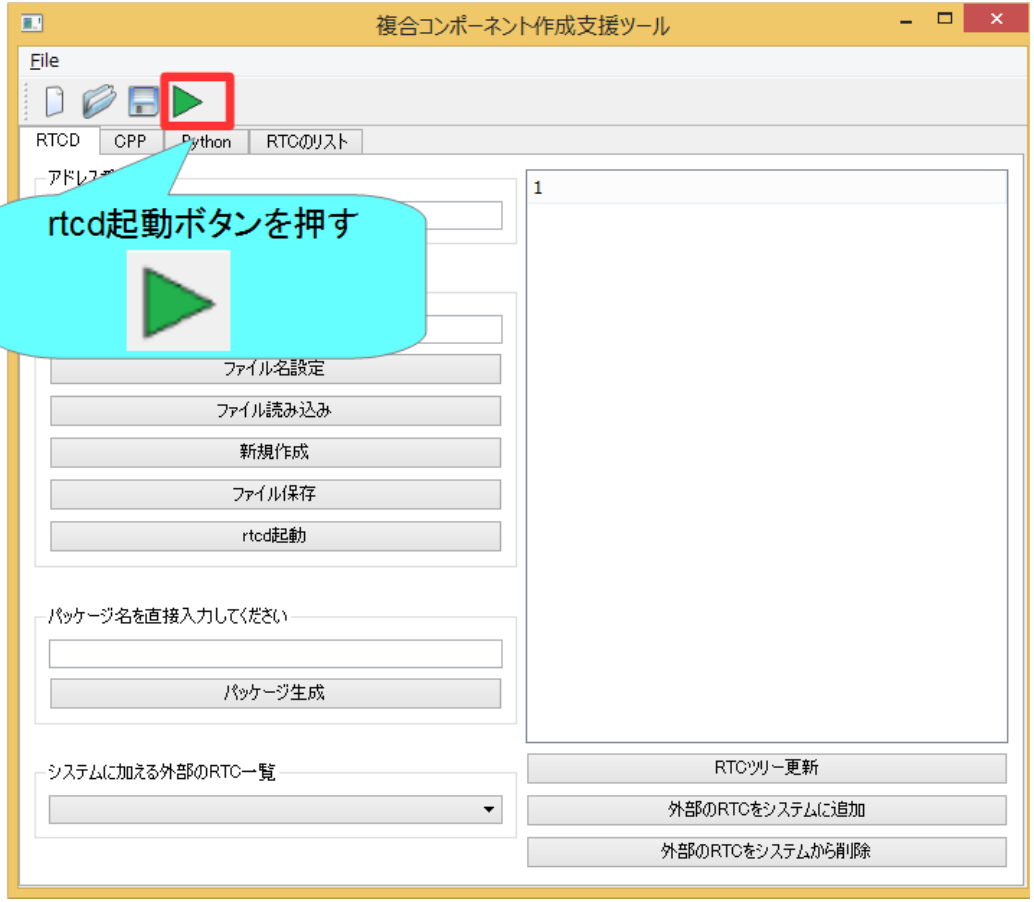
- プロジェクトの新規作成

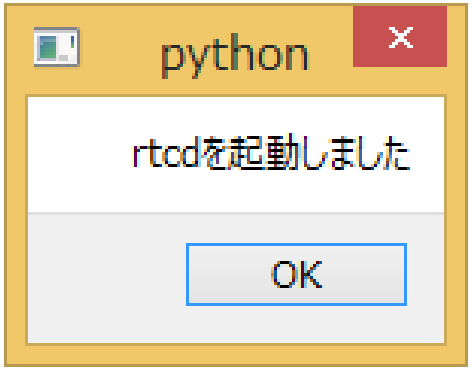




複合コンポーネント作成支援ツールについて

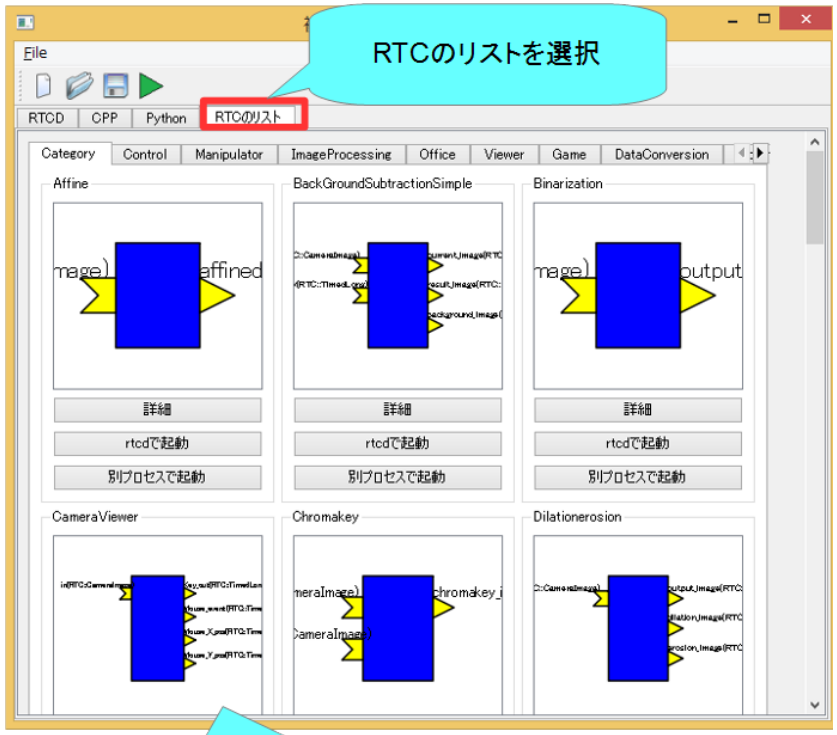
- rtcdを起動する





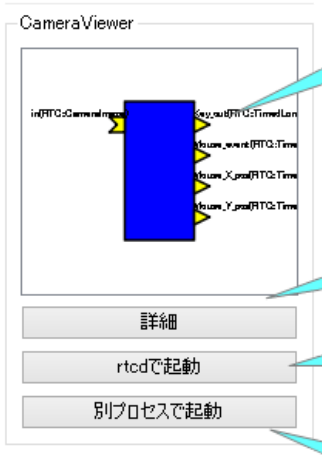
複合コンポーネント作成支援ツールについて

- コンポーネントを起動する
  - OpenCVCamera、Flip、CameraViewerを起動する
    - Flip、OpenCVCameraはCategory、CameraViewerはexampleのタブを開いたら見つかります
    - ※コンポーネントのリストはRTCビルダで最初に設定したカテゴリ別に分けてあります。このように適当にカテゴリを設定すると、カテゴリでコンポーネントの分別ができなくなるのでコンポーネント作成の際には注意するようにしてください。



RTCのリストを選択

RTCを一覧から選択して起動する
今回はrtcdで起動する



ダブルクリックすると拡大表示

詳細ボタンを押すと
コンポーネントの詳細表示

rtcdで起動を選択すると
dll等を読み込んでrtcdで起動する

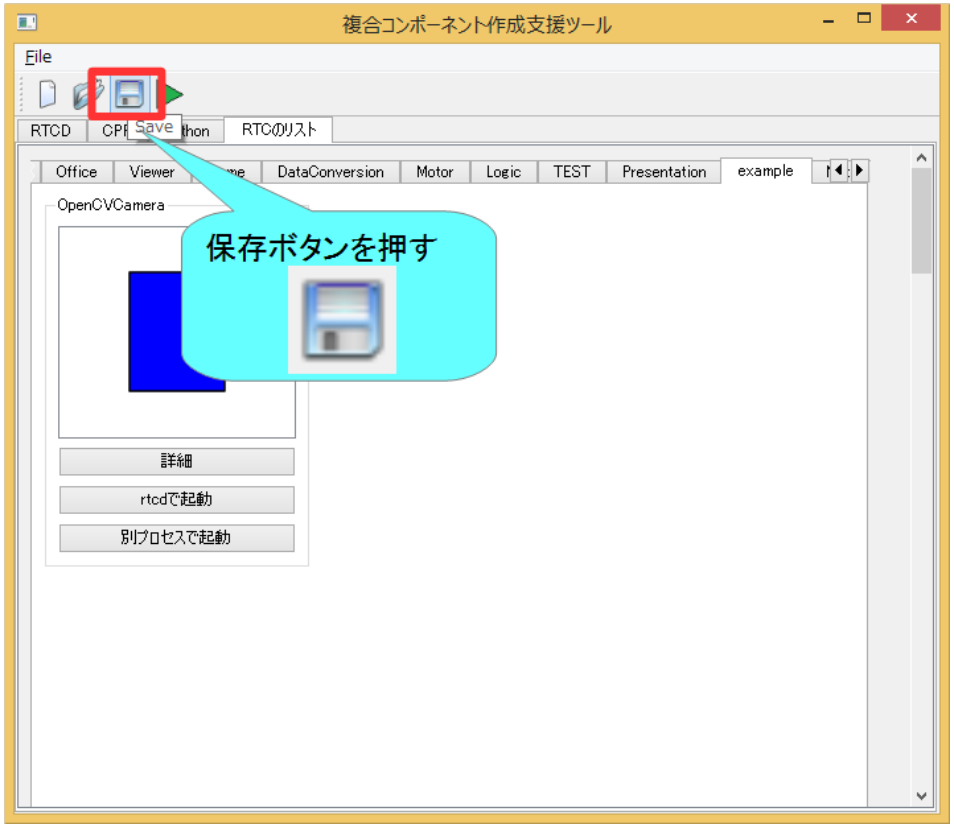
別プロセスで起動を選択すると
exeファイル等で別プロセスで起動する

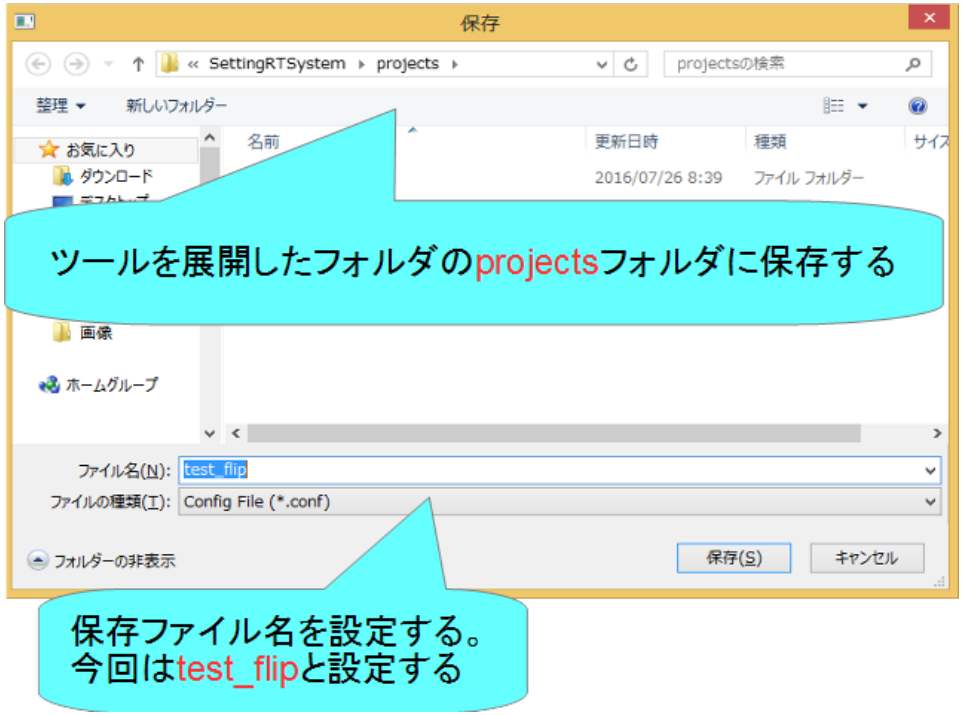
複合コンポーネント作成支援ツールについて

- システムを作成する
  - RTシステムエディタでポートの接続を行ってください。
  - 先ほどと同じ手順で複合コンポーネントを作成してください。

複合コンポーネント作成支援ツールについて

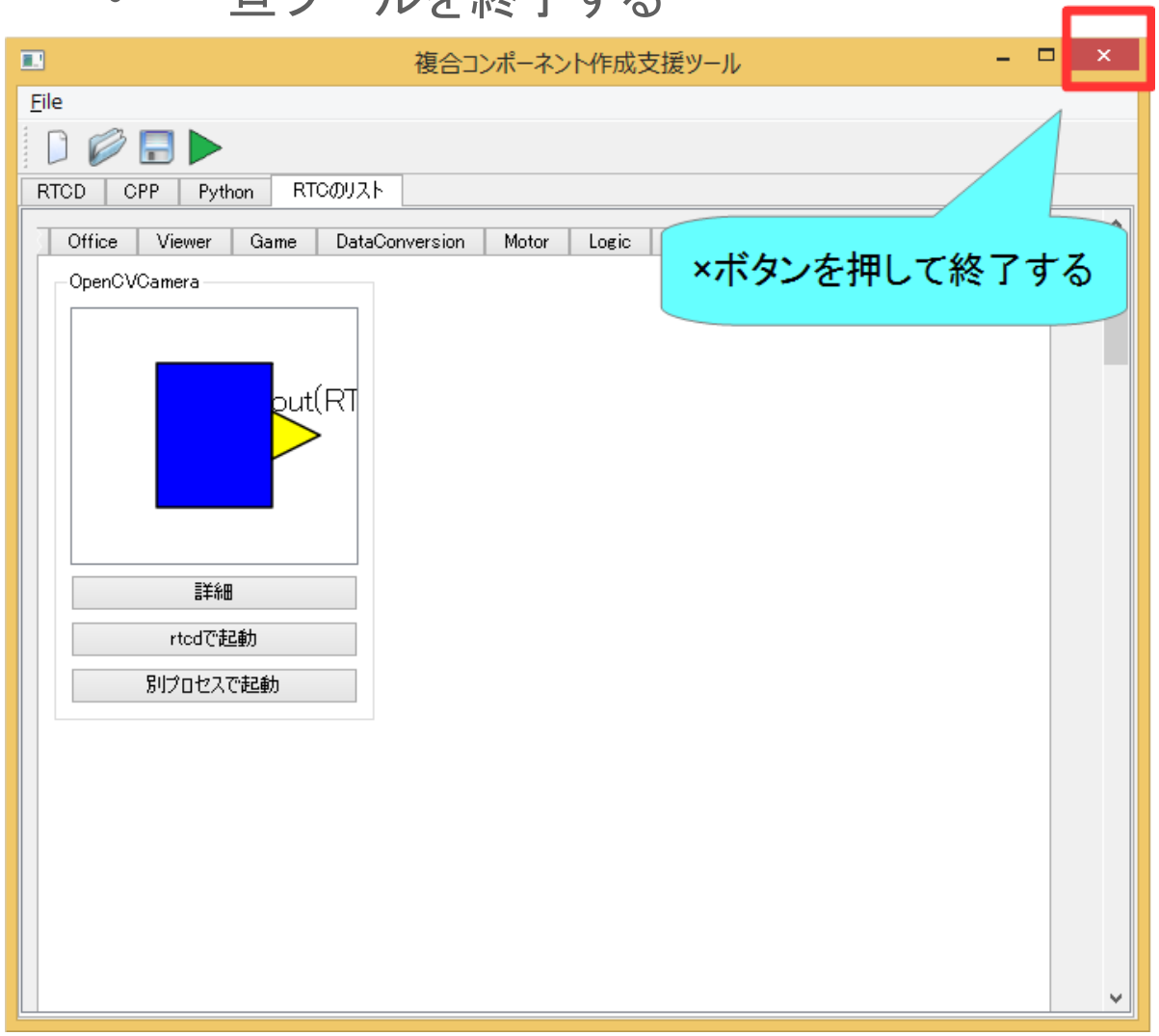
- システムを保存する

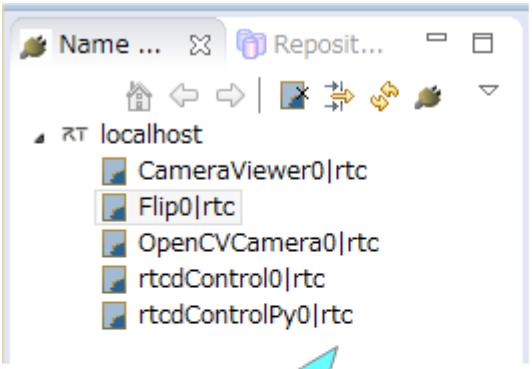




複合コンポーネント作成支援ツールについて

- 一旦ツールを終了する

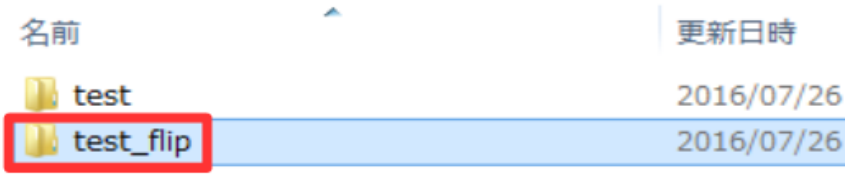




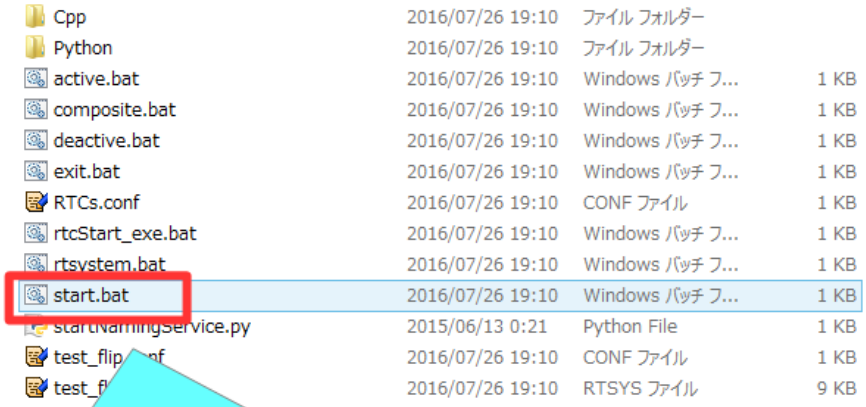
不具合でゾンビが残るので
RTシステムエディタで消す

複合コンポーネント作成支援ツールについて

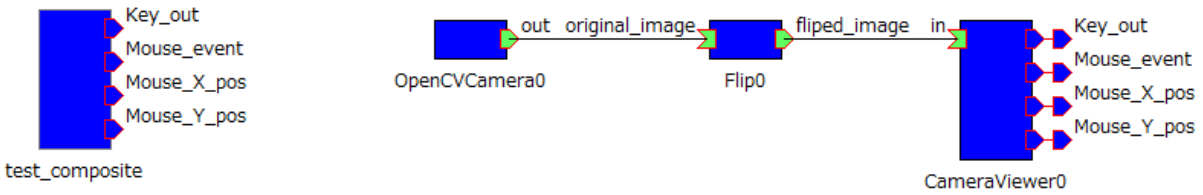
- システムを復元する



先ほど保存先に指定したprojectsフォルダに
test_flipフォルダが生成されているので開く



start.batをダブルクリックして実行する

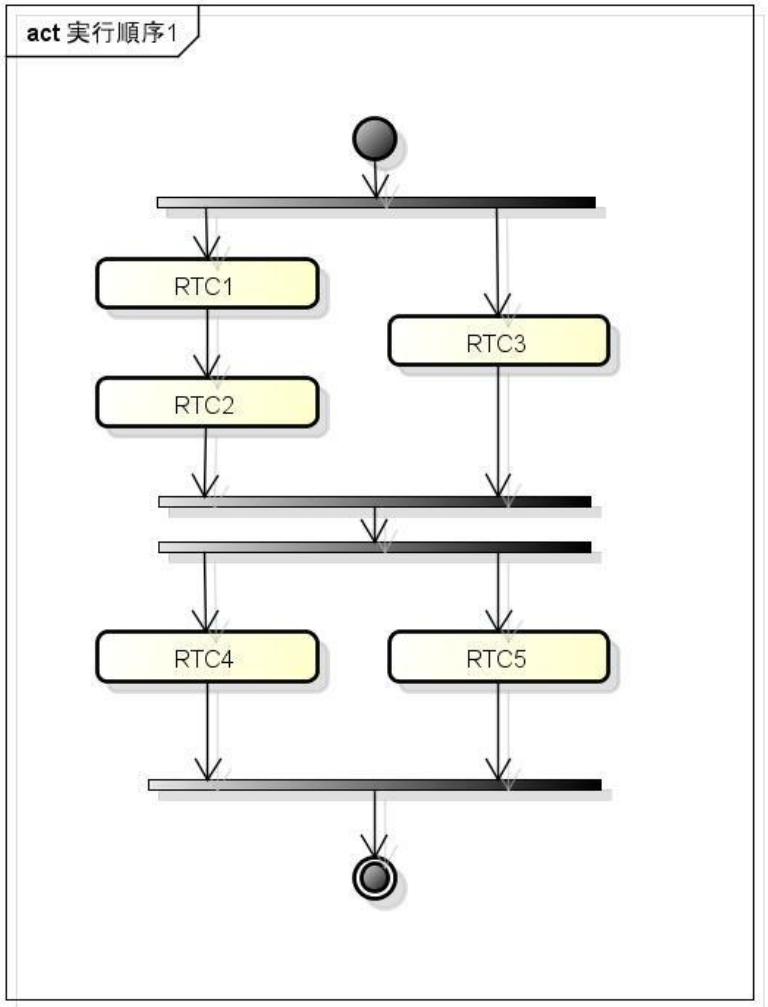


バッチファイルを実行するだけで、
- コンポーネントの起動
- 複合コンポーネントの生成
- システムの復元

を行う

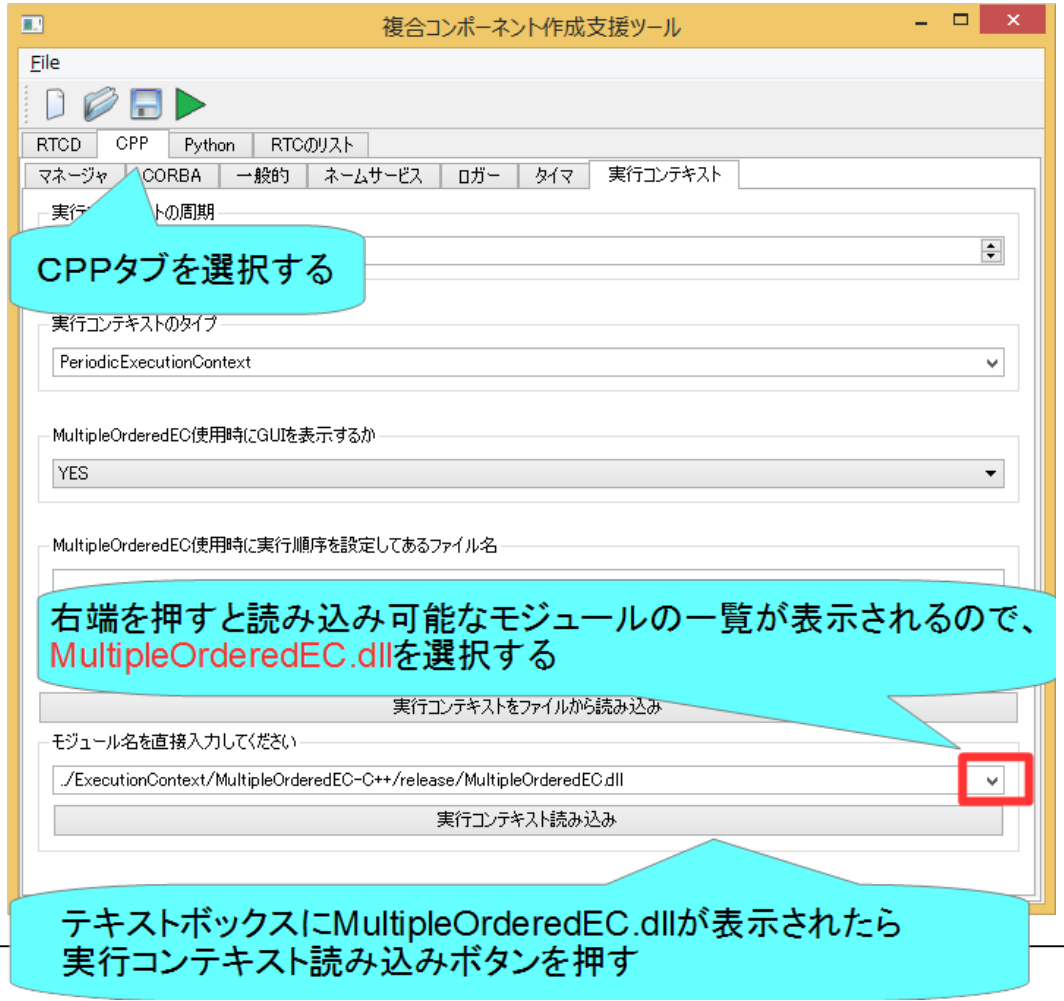
実行順序設定可能な実行コンテキストについて

- 複合コンポーネント作成支援ツールに付属
  - コンポーネントの実行順序を設定可能な実行コンテキスト
  - 直列実行、並列実行どちらも可能



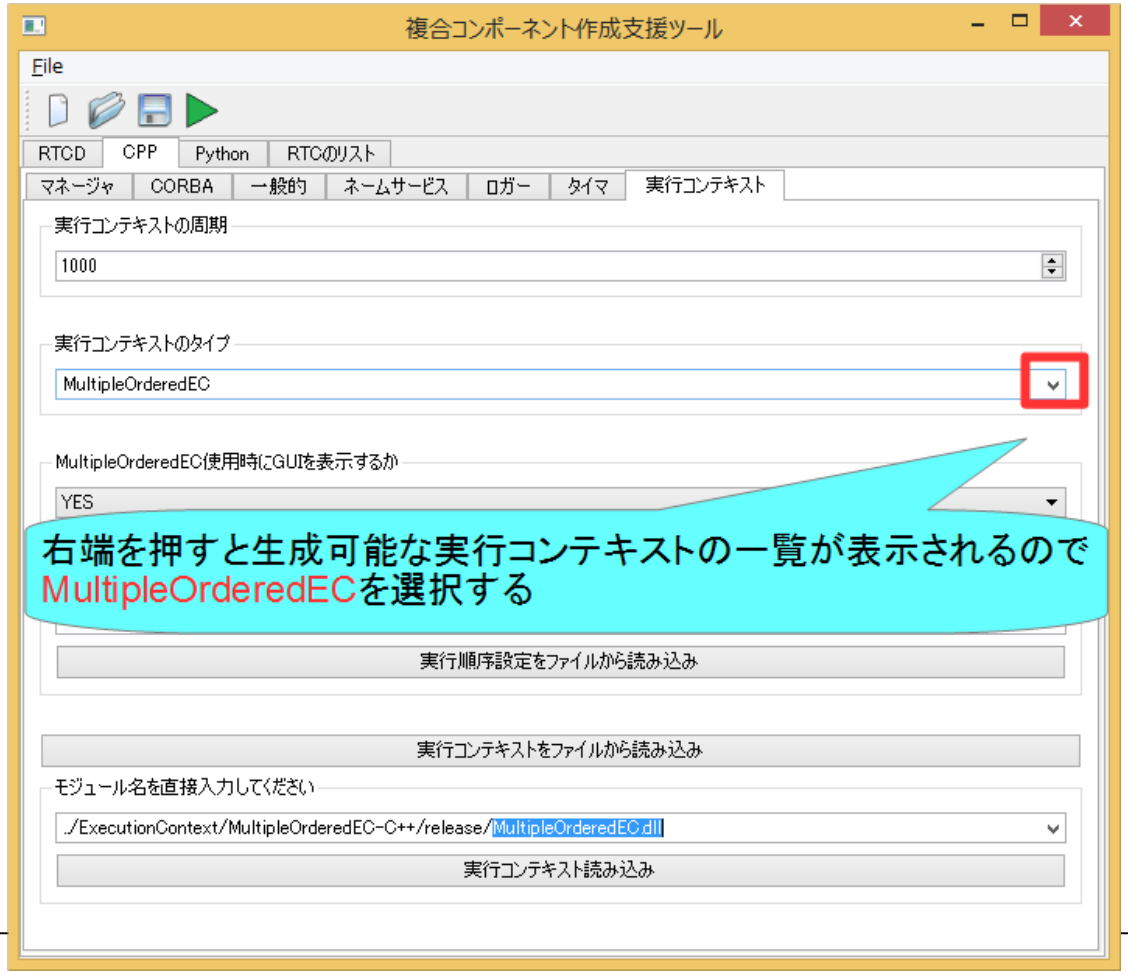
実行順序設定可能な実行コンテキストについて

- 複合コンポーネント作成支援ツールに付属
  - rtcd起動前に実行コンテキストの設定を行う
    - 独自に作成した実行コンテキストはモジュールの読み込みが必要



実行順序設定可能な実行コンテキストについて

- 複合コンポーネント作成支援ツールに付属
  - rtcd起動前に実行コンテキストの設定を行う
    - 設定が終わったら先ほどと同じ手順でrtcd起動、複合コンポーネントの作成を行い保存する



実行順序設定可能な実行コンテキストについて

- start.batを実行する
  - 実行順序設定、確認用GUIが起動する