

RTミドルウェアサマーキャンプ2011

日時:2011年8月30日(月) 14:40~17:00

場所:産業技術総合研究所 中央第2本部・情報棟1階 交流会議室



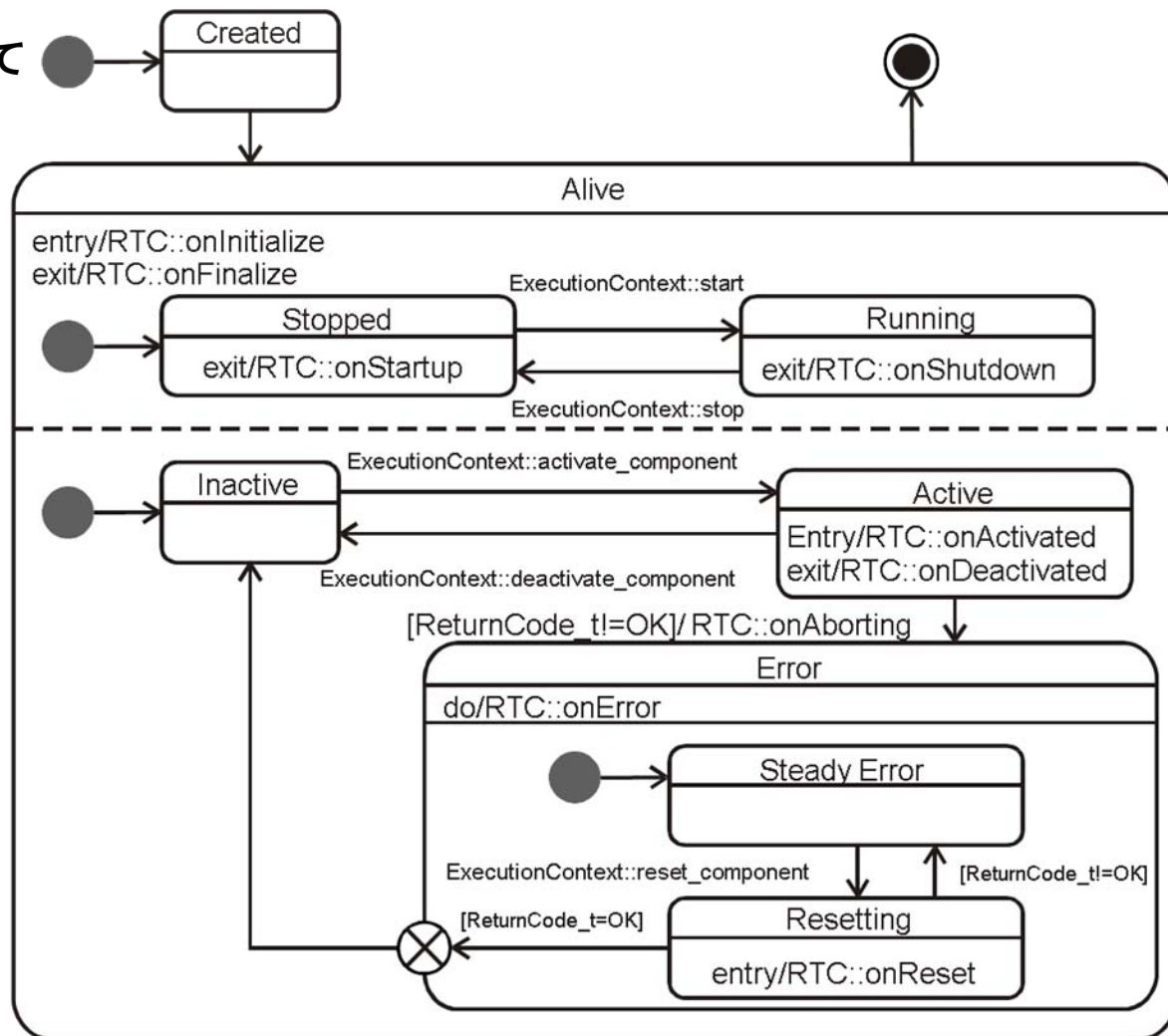
コンポーネント作成演習

産業技術総合研究所 栗原 眞二

- 以下から「USBCamera.zip」をダウンロードします。
<http://www.openrtm.org/openrtm/ja/node/1677#document>
- USBCamera.zipをC:¥に展開します。
 - ※ USBCameraが、スペースを含むパスに展開された場合、VC++でのビルド時にエラーが発生します。
“C:¥”でなくても、スペースを含まないところであれば構いません。

RTコンポーネントの実装(概要)

- RTコンポーネントの状態について
 - 生成状態(Created)
 - 活動状態(Alive)
 - 非アクティブ状態(Inactive)
 - アクティブ状態(Active)
 - エラー状態(Error)
 - 終了状態



RTCライフサイクル (UML ステートマシン図)

■ 予め決められた関数 (コールバック関数)について

関数名	概要
onInitialize	ライフサイクル初期化時に1度だけ呼ばれる。
onActivated	アクティブ化する際に1回呼ばれる。
onDeactivated	非アクティブ化する際に1回呼ばれる。
onExecute	アクティブ状態にあるとき周期的に呼ばれる。
onStateUpdate	onExecute の後に毎回呼ばれる。
onAborting	エラー状態に移行する際に1回呼ばれる。
onError	エラー状態にあるとき周期的に呼ばれる。
onReset	エラー状態から復帰する際に1回呼ばれる。
onShutdown	ECの駆動が停止する際に1回呼ばれる。
onStartup	ECの駆動が開始する際に1回呼ばれる。
onFinalize	ライフサイクル終了時に1度だけ呼ばれる。

■ 単体で動作確認済みのプログラムからRTコンポーネントを作成

```
int main (int argc, char** argv) {  
    // カメラからの画像をキャプチャするクラスの  
    // インスタンスを生成  
    ds_Camera *cam;  
    cam = new ds_Camera();  
  
    // キャプチャクラスのオブジェクトの初期化  
    cam->initialize();  
  
    while(1) {  
        // カメラからの画像キャプチャ処理  
        cam->capture();  
  
        // 画像を表示  
        cvShowImage("Capture", cam->getImage());  
        cvWaitKey(2);  
    };  
  
    // キャプチャクラスのオブジェクトの終了処理  
    cam->finalize();  
  
    // キャプチャクラスのオブジェクトの破棄  
    delete cam;  
  
    return 0;  
}
```

RTC::onInitialize() に実装

RTC::onActivated() に実装

RTC::onExecute() に実装

RTC::onDeactivated() に実装

USBカメラコンポーネントのソースファイル

■ RTコンポーネントにすると

```
RTC::ReturnCode_t USBCamera::onInitialize() {  
    // Set OutPort buffer  
    addOutPort("image", m_imageOut);  
    // Bind variables and configuration variable  
    bindParameter("deviceNumber", m_deviceNumber, "0");  
  
    //カメラからの画像をキャプチャするクラスの  
    //インスタンスを生成  
    cam = new ds_Camera();  
    return RTC::RTC_OK;  
}
```

```
RTC::ReturnCode_t USBCamera::onFinalize() {  
    // キャプチャクラスのオブジェクトの破棄  
    delete cam;  
    return RTC::RTC_OK;  
}
```

■ RTコンポーネントにすると

```
RTC::ReturnCode_t
USBCamera::onActivated(RTC::UniqueId ec_id) {
    // キャプチャクラスのオブジェクトの初期化
    if(cam->initialize())
        return RTC::RTC_OK;
    return RTC::RTC_ERROR;
}
```

```
RTC::ReturnCode_t
USBCamera::onDeactivated(RTC::UniqueId ec_id) {
    // キャプチャクラスのオブジェクトの終了処理
    cam->finalize();
    return RTC::RTC_OK;
}
```



```
RTC::ReturnCode_t USBCamera::onExecute(RTC::UniqueId ec_id) {
    // カメラからの画像キャプチャ処理
    if (cam->capture() < 0)
        return RTC::RTC_OK;

    // 画像サイズの取得
    int len = cam->getImageSize();
    CvSize size = cam->getSize();

    // アウトポート変数へ画像情報をセット
    m_image.pixels.length(len);
    m_image.width = size.width;
    m_image.height = size.height;

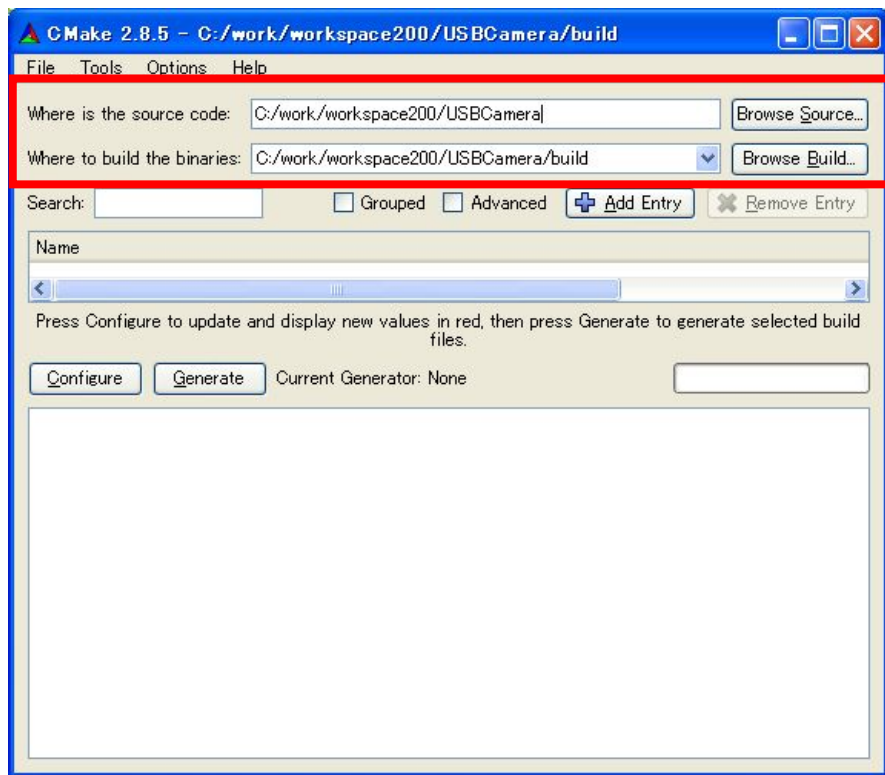
    // アウトポート変数へ画像データをセット
    memcpy((void *)&(m_image.pixels[0]), cam->getImageData(), len);

    // アウトポートからデータ出力
    m_imageOut.write();

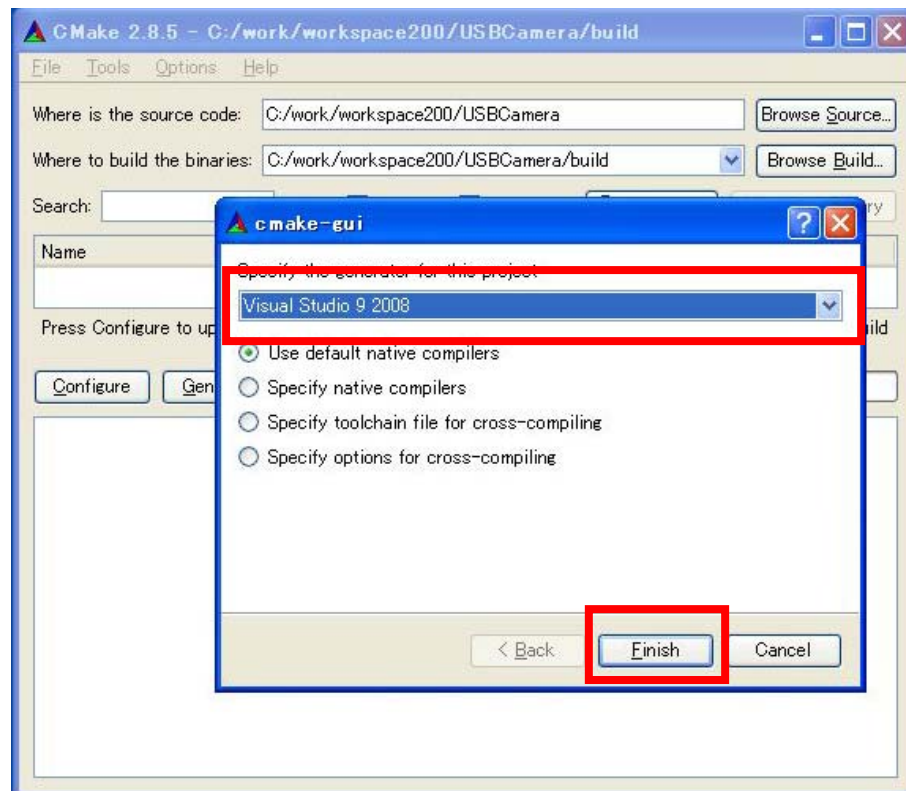
    return RTC::RTC_OK;
}
```

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

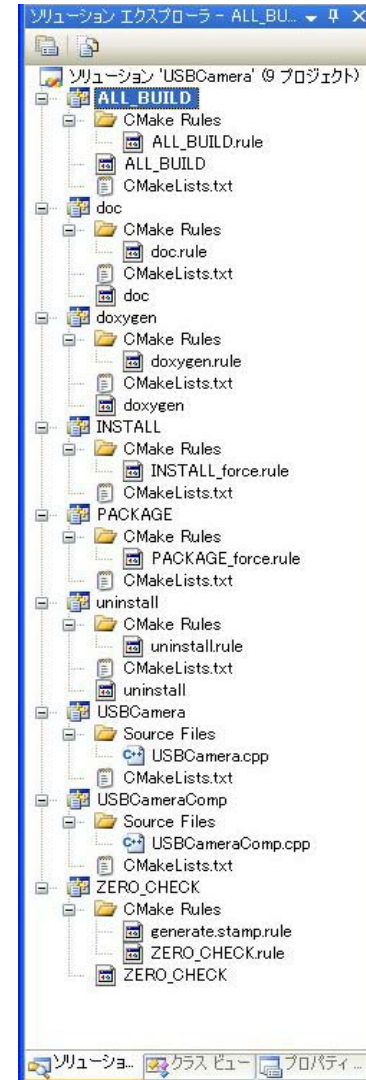
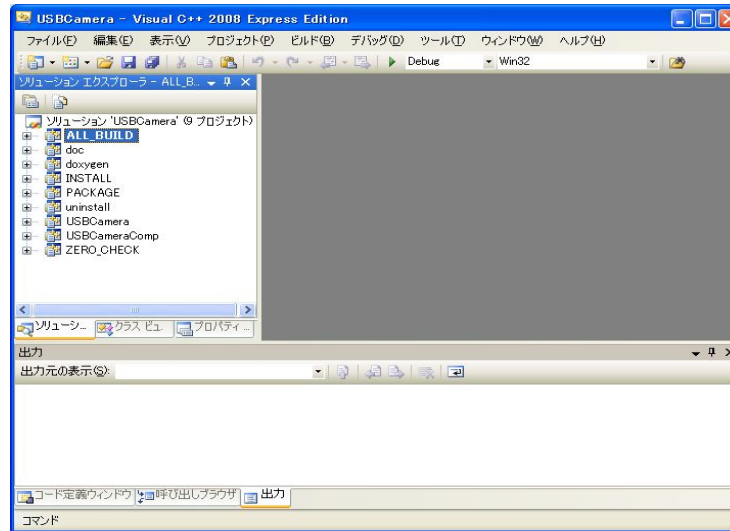
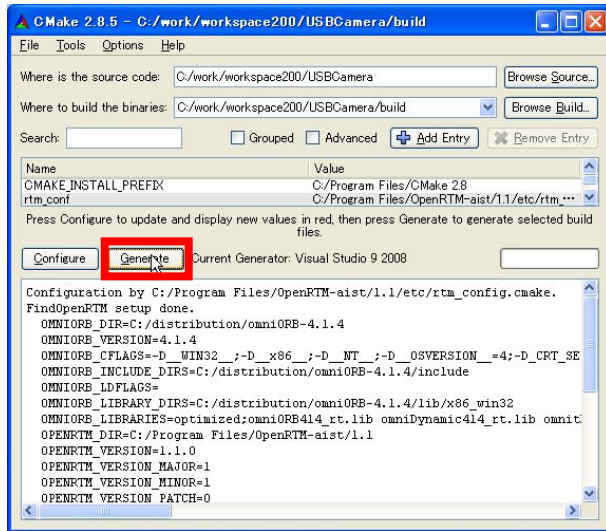


※binaryには, sourceとは別のディレクトリを指定する事を推奨

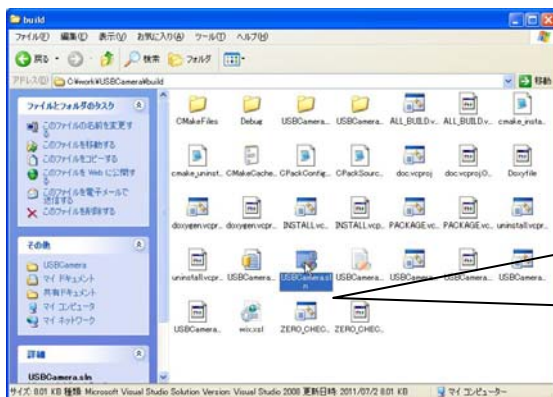
※日本語は文字化けしてしまうため英数字のみのディレクトリを推奨

コンパイル(Windows, CMake利用)

③ 正常終了後, 「Generate」を実行

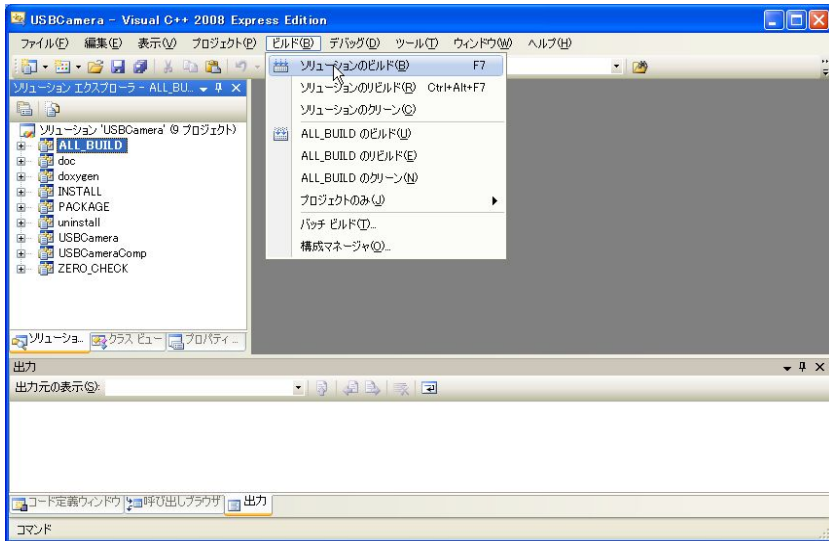


④ binaryとして指定したディレクトリ内にあるソリューションファイルを開く

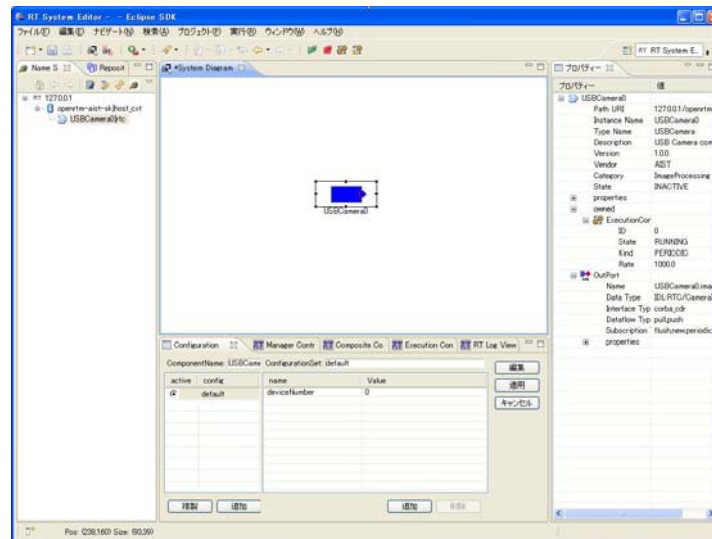
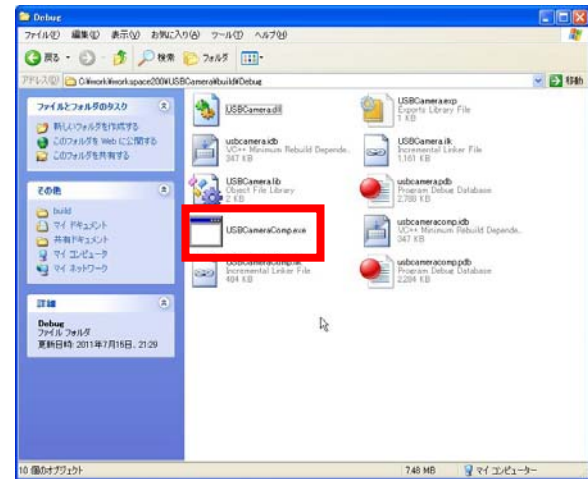


コンパイル・実行(Windows, CMake利用)

⑤ ソリューションをビルド



⑥ binaryにて指定したディレクトリ以下のDebug内のUSBCameraComp.exeを起動



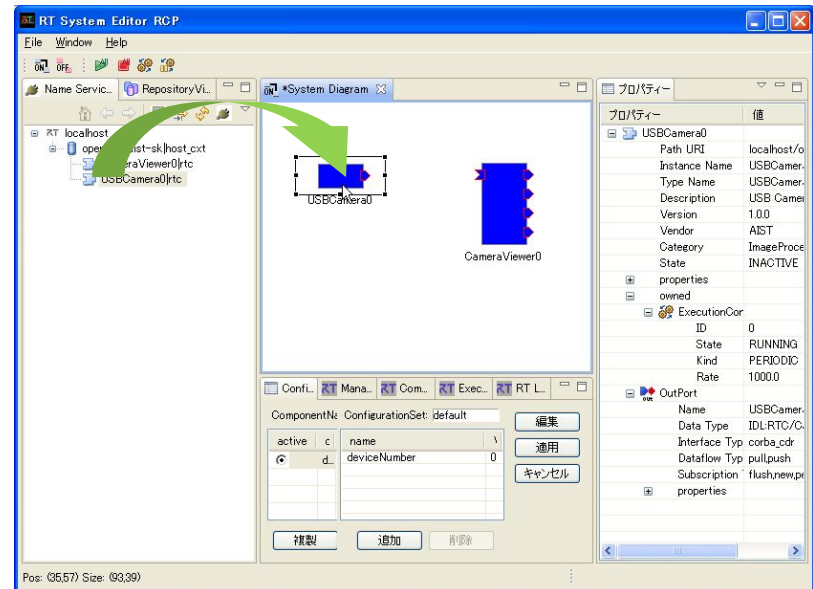
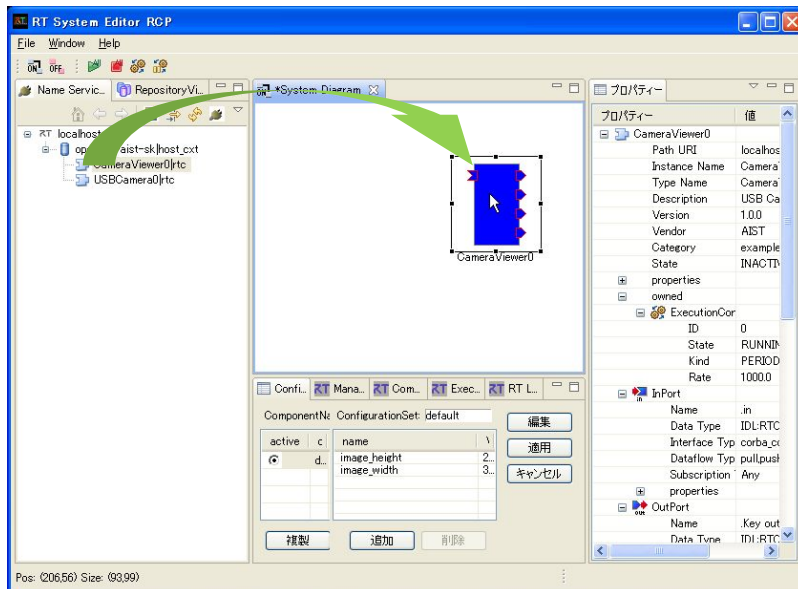
1. CameraViewerの起動

[スタート]メニューから起動

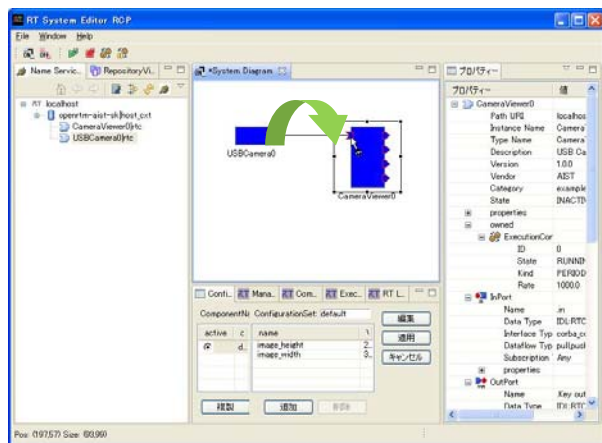
[プログラム]→[OpenRTM-aist 1.1]→[C++]→[components]
→[opencv-rtcs]→ [CameraViewerComp.exe]

2. コンポーネントの接続

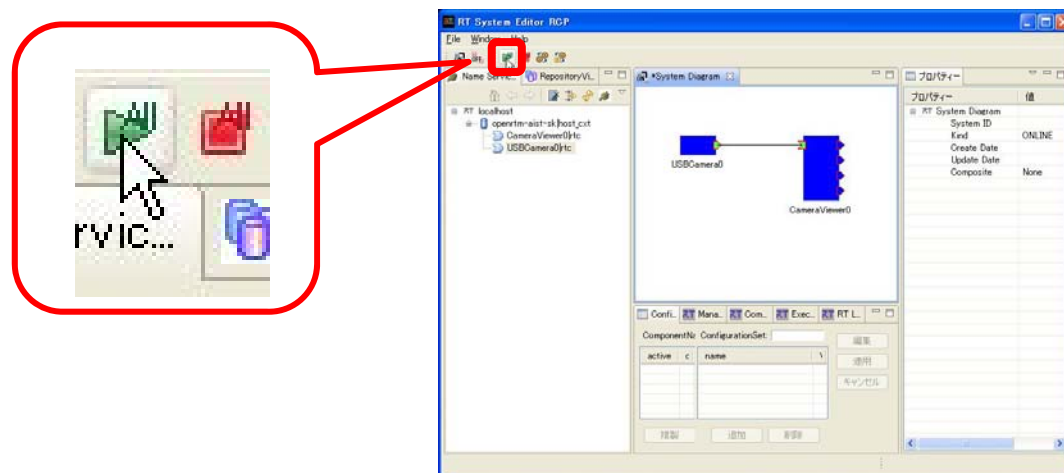
USBCameraとCameraViewerをシステムダイアグラムにドラッグ&ドロップ
します。



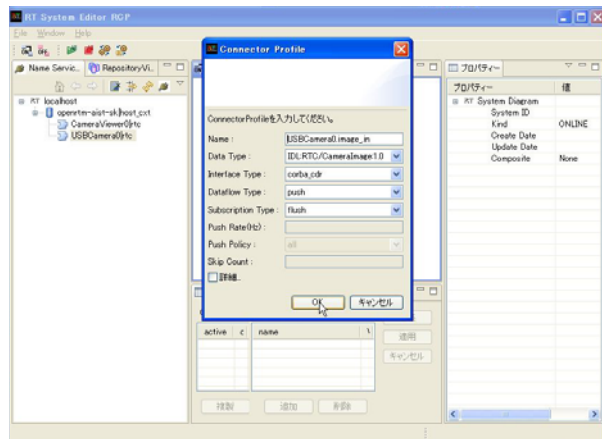
3. ポートを接続します。



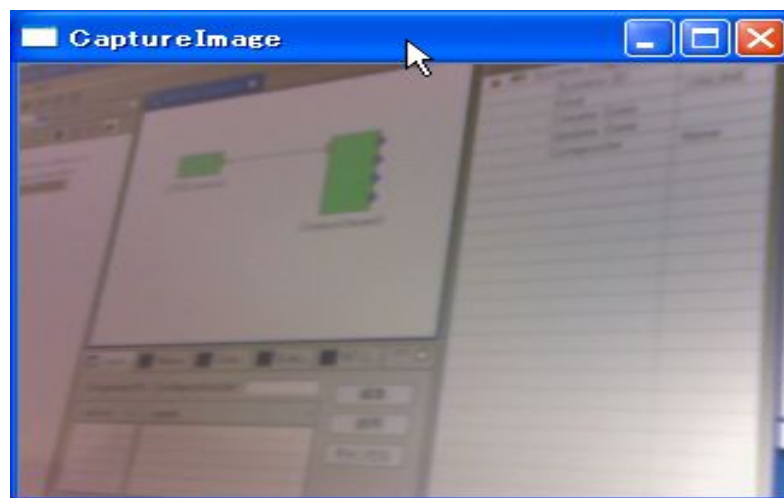
5. コンポーネントをアクティベートします。



4. 接続ダイアログでOKをクリックしポートを接続します。

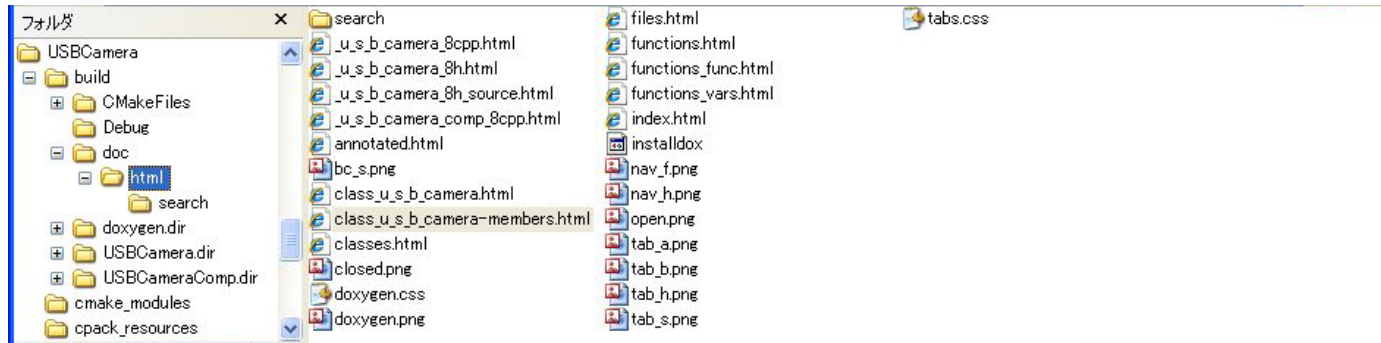


6. CameraViewerに画像が表示されます。



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以下にdoxygenにて生成したドキュメント



■ 生成されたドキュメントの例

usbcamera 1.0.0

メインページ クラス ファイル

検索

生成 生成索引 生成メソッド

クラス USBCamera

USB Camera component. [詳細]

```
#include <USBCamera.h>
```

すべてのメンバー一覧

Public メソッド

戻り値	メソッド名	説明
RTC::ReturnCode_t	onActivated (RTC::UniqueId ec_id)	
RTC::ReturnCode_t	onDeactivated (RTC::UniqueId ec_id)	
RTC::ReturnCode_t	onExecute (RTC::UniqueId ec_id)	

Protected 変数

型	変数名	説明
int	m_deviceNumber	
CameraImage	m_m_image	
OutPort< CameraImage >	m_m_imageOut	

説明

このクラスの説明は次のファイルから生成されました:

- C:/work/workspace199/USBcamera/USBcamera.h
- C:/work/workspace199/USBcamera/USBcamera.cpp

```
RTC::ReturnCode_t USBCamera::onDeactivated ( RTC::UniqueId ec_id ) [virtual]
```

Post-processing such as freeing allocated memory.

```
RTC::ReturnCode_t USBCamera::onExecute ( RTC::UniqueId ec_id ) [virtual]
```

Capture images from the camera, and outputs the data from OutPort.

変数

```
int USBCamera::m_deviceNumber [protected]
```

Device number

- Name: deviceNumber deviceNumber
- DefaultValue: 0

```
OutPort<CameraImage> USBCamera::m_m_imageOut [protected]
```

Capture images data from the camera

- Type: RTC::CameraImage

usbcamera1に対してFri Jul 15 2011 21:41:19に生成されました. [doxygen](#) 1.7.3

usbcamera 1.0.0

メインページ クラス ファイル

ファイル一覧

C:/work/workspace199/USBcamera/USBCamera.h

説明を見る。

```
00001 // -*- C++ -*-
00015 #ifndef USBCAMERA_H
00016 #define USBCAMERA_H
00017
00018 #include <rtm/Manager.h>
00019 #include <rtm/DataFlowComponentBase.h>
00020 #include <rtm/CorbaPort.h>
00021 #include <rtm/DataInPort.h>
00022 #include <rtm/DataOutPort.h>
00023 #include <rtm/idl/BasicDataTypesSkel.h>
00024 #include <rtm/idl/ExtendedDataTypesSkel.h>
00025 #include <rtm/idl/InterfaceDataTypesSkel.h>
00026
00027 // Service implementation headers
00028 // <rtc-template block="service_impl_h">
00029
00030 // </rtc-template>
00031
00032 // Service Consumer stub headers
00033 // <rtc-template block="consumer_stub_h">
00034
00035 // </rtc-template>
00036
00037 using namespace RTC;
00038
00039
00040 class USBCamera
00041 : public RTC::DataFlowComponentBase
00042 {
00043 public:
00044     USBCamera (RTC::Manager* manager);
00045     ~USBCamera ();
00046
00047     // <rtc-template block="public_attribute">
```

動作確認(コンポーネントの追加)

1. Flipコンポーネントの起動

[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]→ [FlipComp.exe]

2. コンポーネントの追加

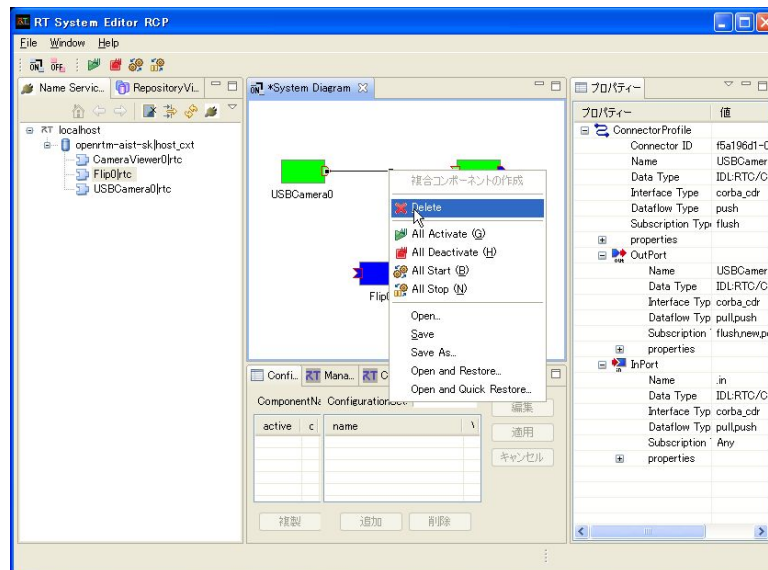
Flipをシステムダイアグラムにドラッグ & ドロップします。

3. USBCameraとCameraViewerのポートを切断します。

(1) ポートの接続線をクリックします。

(2) 接続線を右クリックします。

(3) “Delete”をクリックします。



動作確認(コンポーネントの追加)

1. コンポーネントの接続
USBCameraのOutPortとFlipのInPortを接続します。
2. コンポーネントの接続
FlipのOutPortとCameraViewerのInPortを接続します。
3. Flipコンポーネントをアクティベートします。
 - (1) Flipを右クリックします。
 - (2) 表示されたメニューにて“Activate(A)”をクリックします。

