自由課題



自由課題で使用する機材:RaspberryPiとWebカメラ

概要

Raspberry Pi に接続された WEB カメラから画像データを取得し、OutPort で出力するコンポーネント を作成します。自由課題では、ここまで行ってきた課題を理解していることの確認と他の機器で使用す るコンポーネントの作成の仕方を学びます。これができれば、自分でコンポーネントを作成できるよう になります。みなさん、チャレンジしてみてください。

内容

1.	コンポーネントの仕様	.2
2.	コンポーネントの雛型の生成	.3
3.	ヘッダ、ソースの編集1	10
4.	CMake によるビルドに必要なファイルの生成1	1
5.	Visual Studio でビルド1	$\lfloor 2 ightharpoonup$
6.	コンポーネントの動作確認1	$\lfloor 2 ightharpoonup$
7.	コンポーネントを Raspberry Pi ヘコピー1	16
8.	Raspberry Pi 上でコンポーネントをビルド1	L7
9.	コンポーネントの接続1	18
10.	他コンポーネントとの接続2	20
11.	RTC-Library-FUKUSHIMA	21

この講習会テキストは下記ページを参考にしています。

・チュートリアル(画像処理コンポーネントの作成 Windows 編)

http://www.openrtm.org/openrtm/ja/node/5022 (2016/7/28 アクセス)

※文中の「x.y」や「x.y.z」の表記は使用環境の OpenRTM-aist のバージョンに読み替えてください。

1. コンポーネントの仕様

 作成するコンポーネントについて 作成するコンポーネントは Raspberry Pi に接続された WEB カメラから画像を取得し OutPort で画像データを出力するコンポーネントです。

コンポーネントの動作は、OpenCV を使用し WEB カメラから画像を取得し画像データを OutPort(CameraImage 型)から出力します。

さらに外部パラメータを使用し、取得する画像の縦幅、横幅を変更することが出来ます。 従って、このコンポーネントは OutPort(CameraImage 型)を一つ持ち、コンフィギュレー ションで画像サイズを変更するパラメータを持ちます。

このコンポーネントは Raspbery Pi上で起動しデータを取得します。取得したデータは PC で起動されたビューアコンポーネントに送られ PC 側で表示されます。従ってシステムのイ メージは下記の様になります。



② 仕様まとめ

以上のことをまとめるとコンポーネントは以下の様になります。

コンポーネント名称	RaspWEBCamera	
OutPort		
ポート名	WebCameraImageOut	
型	CameraImage	
Configuration		
パラメータ名	height	width
型	int	int

この仕様を元にコンポーネントを作成していきます。

2. コンポーネントの雛型の生成

コンポーネントの雛型の生成のために RTCBuilder を起動してください。

- 新規プロジェクト 新規プロジェクトを作成します。プロジェクト名は[RaspWEBCamera]とします。 手順に関しては、第二部の Flip コンポーネント作成を参考にしてください。
- ② プロファイル情報入力とコードの生成

ー番左の「基本」タブを選択し、基本情報を設定します。コンポーネントの名前や概要など を記入します。ラベルが赤文字の項目は必須項目です。その他はデフォルトのままで大丈夫 です。

モジュール名: RaspWEBCamera モジュール概要: get imageDate in WEBCamera バージョン:1.0.0 ベンダ名:Aizu モジュールカテゴリ: Camera 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:DataFlowComponent 最大インスタンス数:1 実行型:PeriodicExecutionContext 実行周期:1000.0 ファイル(E) 編集(E) ナビゲート(N) 検索(A) プロジェクト(P) 実行(R) ウィンドウ(W) ヘルプ(H) クイック・アクセス 🖹 🖹 Java 💦 RTC Builder 📸 = 🗒 📾 🛃 🍓 = 🛷 = 🖢 = 🕸 = 🆛 + = + 😫 パッ... 🕱 🗖 🗖 🕕 *RaspWEBCamera 😒 - -□ 🙀 🔻 基本 A 🗁 RaspWEBCamera ▼ RT-Component Basic Profile + ヒント > RTC.xml 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: RTコンオ この名称 *モジュール名: RaspWEBCamera 使用でき モジュール概要: get imageDate in WEBCamera RTコンオ モジュール概要: ASCII文: *バージョン: 1.0.0 バージョン: RTコンオ *ベンダ名: Aizu x.v.z(x.v *モジュールカテゴリ: Camera ベンダ名: RTコンオ ASCII文 コンポーネント型: STATIC モジュールカテゴリ: RTコンオ 選択肢に アクティビティ型: PERIODIC 使用でき コンポーネント型: RTコンオ コンポーネント種類: 🔽 DataFlow 🔤 FSM 📃 MultiMode · STATI 最大インスタンス数: 1 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 - 0 BuildView 🔀 WebCameraIma RaspWEBCamera

次に、「アクティビティ」タブを選択し、使用するアクションコールバックを指定します。 WEB カメラコンポーネントでは、onActivated0,onDeactivated0,onExecute0コールバッ クを使用します。下図のように赤枠の onAtivated をクリック後に赤枠のラジオボタンにて "on"にチェックを入れます。onDeactivated,onExecute についても同様の手順を行います。

C RTC Builder - RaspWEB	Camera/RTC.xml - Eclipse SDK		frame.	and the state of the		
ファイル(E) 編集(E) ナヒ	ビゲート(<u>N</u>) 検索(<u>A</u>) プロジェクト(<u>(P)</u> 実行(<u>R</u>) ウィンドウ(<u>W</u>)	ヘルプ(<u>H</u>)			
	• ∦ • 2 • 5 • • ←	• • •	21	イック・アクセス	😭 🖓 Java 💦 F	RTC Builder
ا 🗆 🗠 الا	≯*RaspWEBCamera ⋈					
📄 🔄 🗢 🖌 🕞 RaspWEBCamera	onActivated onError	onDeactivated onReset	onAborting		onError onReset	ERROR状。 ERROR状
RTC.xml		Dataflow型コンポーネントのフ	7クション		onExecute onStateUpdate	アクティ: onExecut
	onExecute	onStateUpdate	onRateChange	ed	onRateChanged	Execution
	onAction	FSM型コンホーネントのアク	ション		onAction onModeChanged	刃応する モードが
		Mode型コンポーネントのアク	フション		動作概要:	アクティ
	onModeChanged				事前条件:	アクティー
	✓ Documentation このセクションでは各アクション 上段のアクションを選択すると、	の概要を説明するドキュメントを それぞれのドキュメントを記述で 」	記述します。 さます。		争 後条件:	アクティー
	アクティビティ名: ONACtivated	1		ON OFF		-
			2 - 2. R+ - 42.4		TC yml	F.
	A 1071 ET1 7-9/1-1			、生成言語・環境へ	re.xmi	
	💭 BuildView 🔀		WehCar			
		Rasp	WEBCamera			
		1				

4

「データポート」タブを選択し、データポートの情報を入力します。 先ほど決めた仕様を 元に以下のように入力します。

・OutPort ポート名: WebCameraImageOut データ型: RTC::CameraImage 変数名: WebCameraImage 表示位置: right



「コンフィギュレーション」タブを選択し、先ほど決めた仕様を元に、Configurationの情報を入力します。制約条件および Widget とは、RTSystemEditor で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表示する際に GUI で値の変更を行うための形式を表すものです。

ここでは、カメラのサイズをコンフィギュレーションで操作出来る様にするので、[height] と[width]を設定します。



C RTC Builder - RaspWEB	Camera/RTC.xml - Eclipse SDK	
ファイル(E) 編集(E) ナヒ	2ゲート(N) 検索(A) プロジェクト(P) 実行(B) ウィンドウ(W) ヘルプ(H)	
9 🛧 🖬 🕼 🗠 🔒	▲ ▼ パ マ マ マ マ マ マ マ マ マ マ マ マ マ マ マ マ マ マ	RTC Builder
I Ify 23 □ AspWEBCamera ARTC.xml	★*RaspWEBCamera ☆ コンフィギュレーション・パラメータ ◆RT-Component Configuration Parameter Definitions このセクションではRTコンポーネントのコンフィギュレーション・パラメータを指定します。 *名称 height width Delete	► ヒント Config. Para パラメータ名
	 ▼ 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: width 	デフォルト値 変数名:
	< m	۰. ۲
	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	J
	BuildView 😒 WebCameraImaneOut RaspWEBCamera	

「言語・環境」タブを選択し、プログラミング言語を選択します。 ここでは、C++(言語)を 選択します。言語・環境はデフォルトでは設定されていないので、指定し忘れるとコード生 成時にエラーになりますので、 必ず言語の指定を行うようにしてください。

C RTC Builder - RaspWEBC	amera/RTC.xml - Eclipse SDK		South to make the	
ファイル(<u>E</u>) 編集(<u>E</u>) ナビ	ゲート(<u>N</u>) 検索(<u>A</u>) プロジェクト(<u>P</u>)	実行(<u>R</u>) ウィンドウ(<u>W</u>) ヘルプ(<u>H</u>)		
📬 - 🖪 🕲 🖕 🍕	• % • <u>8</u> • 9 • 6 • •	⇒ •	クイック・アクセス	😭 💐 Java 💦 RTC Builder
תיא צ ם ם ה אינייייייייייייייייייייייייייייייייייי	★ *RaspWEBCamera ⊗			
✓ ➢ RaspWEBCamera	言 前 ・ 項 項 マ 画調 このセクションでは使用する言語を計 ● C++ ● Python ● Java ● Ruby	皆定します	Use old build environment.	✓ ヒント 言語: RTコンポーネントを(= 環境: 言語ごとのライブラし 詳細情報で設定した内
	▼ 環境 このセクションでは依存するライブミ	ラリや使用する05などを指定します		
	version	US	Add Delete	-
	< 其本 アクティビティ データポート サ	… ナービスポート コンフィギュレーション	ドキュメント生成 言語・環境 PT	F
	BuildView 3	RaspWEBCar	WebCameraImaneOut	
		1		

全ての設定が完了しましたら、「基本」タブに戻りコード生成ボタンをクリックします。問 題がなければコンポーネントの雛型が生成されます。

C RTC Builder - RaspWEB	Camera/RTC.xml - Eclipse SDK				
ファイル(E) 編集(E) ナヒ	ビゲート(<u>N</u>) 検索(<u>A</u>) プロジェクト(<u>I</u>	P) 実行(R) ウィンドウ(W) ヘルプ([<u>H</u>)		
📑 - 🛛 🕲 🛃 🧣	⊾ • ∦ • ∦ • (• • •	• ⇒ •	クイック・アクセス	👔 sva 👔	RTC Builder
 Image: Second second	 * ペーシー・シー・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・	 ▼ → ▼ を行います。 ・エクスポート よびエクスポートを行います。 ザービスポート コンフィギュレーショ 	クイック・アクセス ン ドキュメント生成 言語・環境	 (1) します (1) します	RTC Builder ・ MultiM ・ 生成可能 実行型を コンポー この設定 RTコンオ 特定機能 健定した RTCのソ RtCProfil 設定した
		RaspWEBC	WebCameraImageOut	D	
		1			

③ 仮ビルド

ここまでの作業でWEBカメラコンポーネントの雛型が生成されました。 次の作業として CMake を利用してビルド環境の Configure を行います。 スタートメニューなどから CMake (cmake-gui)を起動します。

CMake 3.2.1 - C:/rtcv	ws/RaspWEBCamera/build		-		
Where is the source code:	C:/rtcws/RaspWEBCamera				Browse Source
Where to build the binaries:	C:/rtcws/RaspWEBCamera/E	uild	21		Browse Build_
Name			Value	Carouped C Movanced	Add Entry
Configure Generate	Press Configure to upda	te and display new values	in red, then press Gene	erate to generate selected build	d files.

画面上部に以下のようなテキストボックスがあります。

- $\boldsymbol{\cdot}$ Where is the source code
- Where to build the binaries

「Where is the source code」に CMakeList.txt が有る場所、「Where to build the binaries」 にビルドディレクトリを指定します。

CMakeList.txt はデフォルトでは<ワークス ペースディレクトリ>/ RaspWEBCamera になります。

ビルドディレクトリとは、ビルドするためのプロジェクトファイル やオブジェクトファイル、バイナリを格納する場所のことです。場所は任意ですが、この場合 <ワークスペース ディレクトリ>/RaspWEBCamera/build のように分かりやすい名前をつけた RaspWEBCamera のサブディレクトリを指定することをお勧めします。

ディレクトリは自動で作成されるので指定前に作成する必要はありません。

今回は以下の様になるはずです。

Where is the source code	C:¥rtcws¥RaspWEBCamera
Where to build the binaries	C:¥rtcws¥RaspWEBCamera¥build

指定したら、下の Configure ボタンを押します。すると下図のようなダイアログが表示さ れますので、生成したいプロジェクトの種類を指定します。



Visual Studio バージョン	32/64 bit	生成したいプロジェクトの種類
Visual Studio 2013	32 bit	Visual Studio 12 2013
	64 bit	Visual Studio 12 2013 Win 64
Visual Studio 2015	32 bit	Visual Studio 14 2015
	64 bit	Visual Studio 14 2015 Win 64

ダイアログで Finish を押すと Configure が始まります。問題がなければ下部のログウイン ドウに Configuring done と出力されますので、続けて Generate ボタンを押します。 Generating done と出ればプロジェクトファイル・ソリューションファイル等の出力が完 了します。

次に先ほど指定した build ディレクトリの中の RaspWEBCamera.sln をダブルクリッ クして Visual Studio 2013 を起動します。

起動後、ソリューションエクスプローラーの ALL_BUILD を右クリックしビルドを選 択してビルドします。特に問題がなければ正常にビルドが終了します。



3. ヘッダ、ソースの編集

 ヘッダファイル (RaspWEBCamera.h)の編集 OpenCV のライブラリを使用するため、OpenCV のインクルードファイルをインクルード します。下記内容をインクルードしている所に追加してください。

※ソースコードは一行で表示するためにフォントを小さくしています。

```
//OpenCV 用インクルードファイルのインクルード
#include <opencv2/core/core.hpp>
#include <opencv2/highgui/highgui.hpp>
#include <opencv2/imgproc/imgproc_c.h>
#include <opencv2/imgproc/imgproc.hpp>
```

画像の保存用にメンバー変数を追加します。下記内容を class の private:内(// <rtc-template block="private_attribute">の下)に追加してください。

```
int width;
int height;
cv::VideoCapture cap;
cv::Mat mat_image;
```

② ソースファイル (RaspWEBCamera.cpp) の編集

下記のように、onActivated(),onDeactivated(),onExecute()を実装します。

onActivated()

```
RTC::ReturnCode_t RaspWEBCamera::onActivated(RTC::UniqueId ec_id)
{
std::cout << "Active" << std::endl;
// カメラ接続
cap.open(0);
if (cap.isOpened0)std::cout << "USB Camera Connect" << std::endl;
else std::cout << "USB Camera UnConnect" << std::endl;
// 画面サイズ設定
cap.set(CV_CAP_PROP_FRAME_WIDTH, m_width);
cap.set(CV_CAP_PROP_FRAME_HEIGHT, m_height);
return RTC::RTC_OK;
```

```
onDeactivated()
```

onExecute()

```
RTC::ReturnCode_t RaspWEBCamera::onExecute(RTC::UniqueId ec_id)
         if (cap.isOpened()){
                  if (width != m_width || height != m_height)
                  {
                            // 画面サイズ再設定
                            cap.set(CV CAP PROP FRAME WIDTH, m width);
                            cap.set(CV_CAP_PROP_FRAME_HEIGHT, m_height);
                            std::cout << "Width:" << m_width << " Height:" << m_height << std::endl;</pre>
                  //画像取得
                  cap.read(mat_image);
                  IplImage frame = mat_image;
                  int len = frame.nChannels * frame.width * frame.height;
                  // 画面サイズ情報を入れる
                  m_WebCameraImage.pixels.length(len);
                  m WebCameraImage.width = frame.width;
                  m_WebCameraImage.height = frame.height;
                  //画像データを OutPort にコピー
                  memcpy((void *)&(m_WebCameraImage.pixels[0]), frame.imageData, len);
                  //画像データ出力
                  m_WebCameraImageOut.write();
                  //コンフィギレーションの値を保存
                  height = m height;
                  width = m_width;
         }
         return RTC::RTC_OK;
```

4. CMake によるビルドに必要なファイルの生成

C:¥rtcws¥RaspWEBCamera¥src¥CMakeLists.txt を編集します。

このコンポーネントでは OpenCV を利用していますので、OpenCV のヘッダのインクルード パス、ライブラリやライブラリサーチパスを与えてやる必要が有ります。以下の 2 点を追加・ 変更するだけで OpenCV のライブラリがリンクされ使えるようになります。

- ・find_package(OpenCV REQUIRED)を追加
- ・最初の target_link_libraries に \${OpenCV_LIBS} を追加
 - ・target_link_libraries は2ヶ所あります。
 - ・追加するときは\${OpenCV_LIBS}の前に半角スペースを入れてください。

```
set(comp_srcs RaspWEBCamera.cpp)
set(standalone_srcs RaspWEBCameraComp.cpp)
find_package(OpenCV REQUIRED) ~この行を追加
: 中略
add_dependencies(${PROJECT_NAME} ALL_IDL_TGT)
target_link_libraries(${PROJECT_NAME} ${OPENRTM_LIBRARIES} ${OpenCV_LIBS}) ←OepnCV_LIBS を追加
: 中略
add_executable(${PROJECT_NAME}Comp ${standalone_srcs}
${comp_srcs} ${comp_headers} ${ALL_IDL_SRCS})
target_link_libraries(${PROJECT_NAME}Comp ${OPENRTM_LIBRARIES} ${OpenCV_LIBS}) ←OepnCV_LIBS
を追加
```

5. Visual Studio でビルド

CMakeList.txt を編集したので、再度 CMake GUI で Configure および Generate を行います。 CMake の Generate が正常に終了した事を確認し、RaspWEBCamera.sln ファイルをダブル クリックし、Visual C++ 2013 を起動します。

Visual C++ 2013 の起動後、下図のように右クリックでコンポーネントのビルドを行います。



6. コンポーネントの動作確認

コンポーネントが仕様通りに動くか確認をします。そのために PC 内で WEB カメラコンポー ネントとビューアコンポーネントを起動し実際に WEB カメラから画像を取得します。

① NameService の起動

コンポーネントの参照を登録するためのネームサービスを起動します。

[スタート]メニューから[すべてのプログラム]→[OpenRTM-aist x.y.z]→ [tools]→[Start Naming Service]をクリックして下さい。

※Windows8の場合下記パスを参考になります。

C:¥ProgramData¥Microsoft¥Windows¥Start Menu¥Programs¥OpenRTM-aist x.y.z ¥Tools ※[Start Naming Service]をクリックしても omniNames が起動されない場合は、フルコン ピュータ名が 14 文字以内に設定されているかを確認してください。

- ② WEB カメラコンポーネントの起動
 C:¥rtcws¥RaspWEBCamera¥build¥src¥Debug の RaspWEBCameraComp.exe をダブ ルクリックで起動させます。
- ③ ビューアコンポーネントの起動
 InPort で受け取った画像を画面に表示する CameraViewerComp を起動します。

 $[スタート]メニューから[すべてのプログラム] \rightarrow [OpenRTM-aist x.y.z] \rightarrow [C++] \rightarrow$ [Components] \rightarrow [OpenCV-Examples] 内にあるのでダブルクリックで起動してください。

- ④ WEB カメラの接続
 WEB カメラを PC に接続してください。
- ⑤ コンポーネントの接続

RTSystemEditor の左側の Name Service View のコンセントアイコンをクリックし、ネームサーバに接続します。



自ホストのネームサーバに接続します。接続ダイアログに localhost と入力します。

ネームサーバのアドレスを入力	りしてください。
localhost	✓ (Address:Port)
ОК	キャンセル

下記の様に表示されるのでその後、メニューバーの online エディタアイコン(ON と書か れたアイコン)をクリックし、SystemEditor を開きます。

Ele Window Help	RT System Editor RCP	_					
Name Ser. ① Repositor □ 〕	<u>File Window Help</u>						
Mame Ser ● Repositor ● □ プロパティー ● ■ プロパーー 『	572 5FE						
	🏄 Name Ser 🕜 Repositor 🖓 🗖				- 0	🔲 プロパティー	~
* Tr localhost * NB1503013 host_cxt CameraViewer0]rtc RaspWEBCamera0[rtc Configur	🗸 🗞 🐇 🕷 (中 中 🖞						
NBI503013[host_cxt ComeraViewer0]rtc RaspWEBCamera0]rtc Configur Manager Composi Executio RT Log ComponentName: ConfigurationSet: 編集 active config name value 通用 キャンセンル 視題 追 通加 削除 詳細	א דד localhost						
CameraViewer0)rtc RaspWEBCamera0]rtc □ Configur 截 Manager 截 Composi 截 RT Log □ ComponentName: ConfigurationSet: 編集 active config name value 通用 +ヤンセル 後題 通加 削除 詳細	INB1503013 host_cxt						
RaspWEBCamera0Irtc Configur 配 Manager 配 Composi 配 RT Log ロ ComponentName: ConfigurationSet: 編集 active config name value 通用 キャンセル 後期 通加 削除 詳細	CameraViewer0 rtc						
■ Configur 配 Manager 配 Composi 配 Executio 配 RT Log □ ComponentName: ConfigurationSet: 編集 active config name value 通用 キャンセル 夜裂 追加 削除 詳細	RaspWEBCamera0 rtc						
□ Configur 집 Manager 집 Composi 집 Executio 집 RT Log □ □ ComponentName: ConfigurationSet: 編集 active config name value 通用 キャンセル 液製 追加 副除 詳細							
□ Configur 【 Manager 【 Composi 】 Executio 】 【 RT Log □ □ ComponentName: ConfigurationSet: 和集 active config name value 通用 キャンセル 後期 通加 削除 〕詳細							
Configur 社 Manager 社 Composi 社 Executio 社 RT Log ロロ ComponentName: ConfigurationSet: 福集 active config name value 通用 キャンセル 後期 追加 削除 詳細							
Configur 配 Manager 配 Composi 配 Executio 配 RT Log ロロ ComponentName: ConfigurationSet: 編集 active config name value 通用 キャンセル 後裏 道士 道加 削除 詳細							
Configur 配 Manager 配 Composi 配 Executio 配 RT Log ロロ ComponentName: ConfigurationSet: 編集 active config name value 通用 キャンセル 復調 道北 道加 削除 詳細							
Configur 配 Manager 配 Composi 配 Executio 配 RT Log 日 ComponentName: ConfigurationSet: 編集 active config name value 通用 キャンセル 復調 道士 道加 削除 詳細							
Configur 配 Manager 配 Composi 配 Executio 配 RT Log ロロ ComponentName: ConfigurationSet: 編集 active config name value 通用 年ヤンセル 復調 道北 道加 剤除 詳細							
□ Configur 집 Manager 집 Composi 집 Executio 집 RT Log □ □ ComponentName: ConfigurationSet: 編集 active config name value 通用 年マンセル 後期 道北 道加 削除 詳細							
Configur るT Manager るT Composi るT Executio るT RT Log つ ComponentName: ConfigurationSet: 名氏							
Configur Manager ConfigurationSet: ConfigurationSet: AugumentName: ConfigurationSet: Augument Aug							
ComponentName: ConfigurationSet: 編集 active config name value 適用 キャンセンレ 復調 道北 通加 削除 詳細							
ComponentName: ConfigurationSet: 福集 active config name value 通用 キャンセル 後期 道北 道加 削除 詳細							
□ Configur 전 Manager 전 Composi 전 RT Log □ □ ComponentName: ConfigurationSet: 編集 active config name value 通用 キャンセル 後期 道士 道加 削除 詳細							
ComponentName: ConfigurationSet: 編集 active config name value 適用 年マンセル 通知 削除 詳細		Configur	anager 🔭 Composi	. RT Executio RT RT	Log 🗖 🗖		
active config name value 通用 進用 第日 キャンセル 復製 追加 削除 詳細		ComponentName:	ConfigurationSet:		編集		
通用 通用 接顧 追加 削除 詳細		active config	name	value			
複製 追加 削除 詳細					週用		
					キャンセル		
		複製 追り	追加	削除□詳細			

左側の Name Service View から各コンポーネントをドラックアンドドロップで SystemEditor 上にコンポーネントを配置します。そしてデータポートを下記図の様に接続 します。

le <u>W</u> indow <u>H</u> elp				
al aff 🖉 🖉 🔐 🖓				
🛊 Name Ser 🎁 Repositor 🗁 🗆	🚮 *System Diagram 🙁	- 8	□ プロパティー	~ -
🗄 🗘 🖓 📑 🖓 🌽 🎽			プロパティー	値
דא localhost			ਸ਼ System Diagram	
NB1503013 host_cxt			System ID	
🔁 CameraViewer0 rtc			Kind	ONLIN
RaspWEBCamera0 rtc			Create Date	
	WebGene		Update Date	
	Webcame	Key_out	Composite	None
	RaspWEBCamera0	Mouse_event		
	hasp in Ebecarier as	Mouse_X_pos		
		Mouse_Y_pos		
		CameraViewer0		
	Configur KT Manager KT Con	mposi 🔣 Executio 🕅 RT Log 🗮 🗆		
	ComponentName: ConfigurationSe	et: 編集		
	active config name	value		
	-	週用		
		キャンセル		
	複製 追加	追加 削除 二詳細		
			4	

⑥ コンポーネントの Activate

RTSystemEditor の上部にある緑の「ALL」というアイコンをクリックし、 全てのコンポ ーネントをアクティブにします。正常にアクティブになると、下図のように黄緑色でコンポ ーネントが表示されます。

RT System Editor RCP	Restriction of the second seco		
<u>File Window H</u> elp			
🗑 💀 📂 🗰 🦛			
🍺 Name Ser 🎁 Repositor 🖆 🗇	💀 *System Diagram 🛞 🗖 🗖	コノロパティー	~
🖞 (+ - +) 📑 🍃 🤣 🗡		プロパティー	値
אד localhost		אד System Diagram	
Image: Big NB1503013 host_cxt		System ID	
CameraViewer0 rtc		Kind	ONLINE
RaspWEBCamera0 rtc		Create Date	
		Update Date	
	Key_out	Composite	None
	RaspWEBCamera0		
	Mouse_X_pos		
	Mouse_Y_pos		
	CameraViewer0		
	🔲 Configur 💦 Manager 💦 Composi 🥂 Executio 🥂 RT Log 🗮 🗆		
	ComponentName: ConfigurationSet:		
	in the second		
	active config name value 適用		
	++>ZL		
	複製 追加 〕 単細		
		•	۲.

⑦ 動作確認

System Diagram 上の RaspWEBCamera0 をクリックするとコンフィギュレーションビュ ーが表示されます。編集のボタンを押すと height と width が表示されますので変更して、 画像のサイズが変更されるのを確認してください。

a Window Unla					
e <u>window</u> Help					
52 56 🕑 🖻 🔗	i P				
Name Ser 🕅 R	epositor 🗖 🗖	🔂 *System Diagram 🐹	- 8	□ プロパティー	~ - 6
	🖹 🌦 🤣 🎽 🏹			プロパティー	値
মন localhost				🔄 RaspWEBCamera	c
NB1503013	host_cxt			Path URI	localhost
🔁 CameraV	/iewer0 rtc			Instance Nam	e RaspWE
🔁 RaspWEE	3Camera0 rtc			Type Name	RaspWE
		WellComproTecoport		Description	get imag
			Key_out	Version	1.0.0
		RaspWEBCamera0	Mouse_event	Vendor	Aize
			Mouse X nos	Category	Camera
ConfigurationSet :					
	default				
height	default 240				
height width	default 240 320				
height width	default 240 320				
height width	default 240 320				
height width	default 240 320				
height width	default 240 320				
height width	default 240 320				
height width	default 240 320				Ž Apply
height width	default 240 320		ОК	[=ャンセル	Ž Apply

確認を終えたら、RTSystemEditor の上部にあります赤色の「ALL」というアイコンをクリ ックし、コンポーネントをディアクティブ状態にします。その後、Name Service View の コンポーネント一覧から RaspWEBCamera0 を右クリックで選択し、Exit でコンポーネン トを削除します。その後 WEB カメラを PC から取り外し、使用する Raspberry Pi に接続 してください。

7. コンポーネントを Raspberry Pi ヘコピー

作成したコンポーネントを Raspberry Pi にコピーする方法を記載します。前段階として Raspberry Pi に Tera Term でアクセスしてください。

 build 以下を削除 Raspberry Pi にコンポーネントをコピーするときは CMake で作成した build 以下を削除 します。これは、容量を減らすためと Windows 上で作成された build 以下は Raspberry Pi 上では使用できないためです。

② フォルダを zip で圧縮し Raspberry Pi ヘコピーする。
 RaspWEBCamera のフォルダを zip で圧縮してください。今回コピーするには Tera Term の「SSH SCP ...」を使用します。この機能で送れるファイルは1回につき1ファイルなのでフォルダを送るには圧縮して1ファイルにする必要があります。

次に Tera Term の「ファイルメニュー」→「SSH SCP ...」を選択します。

TTSSH:	Secure File Copy	×
Erom:		 Send
To:	~/	Cancel
	You can drag the file to this window.	
	~	
From:		Receive
To:	C:¥Program Files (x86)¥teraterm	

上の方の From にさきほど圧縮した WEB カメラコンポーネントを選択し、Send ボタンを クリックします。

8. Raspberry Pi上でコンポーネントをビルド

Raspberry Pi上で使用できるように再ビルドを行います。

コピーされたファイルを解凍する。
 以下のコマンドを入力し、圧縮ファイルの解凍を行います。

\$ unzip RaspWEBCamera.zip

unzip: 圧縮ファイルを復元する。

② コンポーネントをビルドする。

以下のコマンドで Raspberry Pi上で使用できる様にビルドします。

\$ cd RaspWEBCamera
\$ mkdir build
\$ cd build
\$ cmake ../
\$ make

cd:カレントディレクトリを変更する。 mkdir:ディレクトリを作成する。 cmake:プログラムをコンパイルするための Makefile を生成する。 make: プログラムをコンパイルする。

③ NameServer とコンポーネント起動。以下のコマンドで NameServer とコンポーネントを起動します。

\$ rtm-naming
\$ cd ~/RaspWEBCamera/build/src/
\$./RaspWEBCameraComp

エラーが出ずに起動出来たら完了です。

9. コンポーネントの接続

PC 側、Raspberry Pi 側で NameServer とコンポーネントが起動したので RTSystemEditor でコンポーネント同士を接続します。

① PC で RTSystemEditor を使用し各コンポーネントを Active にする。

RTSystemEditor の Name Service View の接続アイコンをクリックし Raspberry Pi のホ スト名+.local 、または、Raspberry Pi の IP アドレスをダイアログに入力します。すると Name Service View に RaspWEBCamera0 が表示されます。

RT System Editor RCP						
Eile <u>W</u> indow <u>H</u> elp						
n in						
Mame Ser () Repositor () Name Ser () Repositor () NBL503013 host_cxt CameraViewer0 rtc x TraspS1.local () RaspWEBCamera0 rtc	 Regeneration of the second se	図 anager 配 Composi. ConfigurationSet: name	·· 社 Executio 社 value	RT Log 日日 編集 道用 手ヤンセル 細	□ J□/(ティー)	

Name Service View から各コンポーネントを SystemEditor 上にドラッグアンドドロップ し、データポートを接続します。その際に Connector Profile の設定はデフォルトでなく下

記画像の様に、	Subscription Type と Push Policy を new に変更して接続してください	١°

ConnectorProfileを入	カしてください。
Name :	RaspWEBCamera0.WebCameraImageOut_CameraViewer0.in
Data Type :	IDL:RTC/CameraImage:1.0
Interface Type :	corba_cdr 🗸
Dataflow Type :	push 🗸
Subscription Type :	new 🗸
Push Rate(Hz) :	
Push Policy :	new 🗸
Skip Count :	
📄 詳細	

データポートの接続が完了したら、「ALL」という緑のアイコンをクリックし全てのコンポ ーネントをアクティブにしてください。

Raspberry Pi に接続した WEB カメラから画像を取得出来たら完了です。

10. 他コンポーネントとの接続

WEB カメラコンポーネントとビューアコンポーネントの間にサンプルコンポーネントを 挿むことによって取得した画像に色々な変化をもたらすことが出来ます。 ここでは一例を紹介します。

サンプルコンポーネントは下記場所にあります。

[スタート] メニューから[すべてのプログラム] → [OpenRTM-aist x.y.z] → [C++] → [Components] → [OpenCV-Examples]

• FlipComp

入力された画像を反転して出力します。反転の種類は左右反転、上下反転、上下左右反転の 3種類あります。どれにするかはコンフィギュレーションで決定します。



$\cdot \operatorname{AffinComp}$

入力された画像にアフィー変換をかけ平行四辺形にして出力します。



11. RTC-Library-FUKUSHIMA

① RTC-Library-FUKUSHIMA について

RTC-Library-FUKUSHIMA とはロボット産業振興のために作成された **RTC** ソフトウェ アライブラリーです。

主にコンポーネントの登録やダウンロードしての再利用などが出来ます。

			Google"カスタム検索		検索
UNIVERSITY OF AIZU		ミドルウェア	ライブラリ	ドキュメント	フォーラム
RTC-Library-FUKUSHIMA OpenRTM-aistを利用した、ロボット・ティ ・ RTCライプラリ ・ OpenRTM-	ウノロジー・コンポーネ aist	ント (RTC) ライフ	ガラリ		
お知らせ イベント	2015年07月24日	RTミドルウェア 習会が行われまし	<u>強化月間2015</u> した	in 中央大学・RT3	ミドルウェア講
▶ お知らせ一覧を見る 案内	2015年07月09日	<u>Fedora22のOp</u> e しました	enRTM-aist (C	C++/Python)パ	ッケージを公開

② コンポーネントをアップロード

RTC-Library-FUKUSHIMA へのコンポーネントのアップロードの仕方を説明します。

① ログイン

RTC-Library-FUKUSHIMA へは下記 URL でアクセスします。 RTC-Library-FUKUSHIMA : <u>http://192.168.11.101/</u> ・ログイン ID ID: guestuser PW : notrtclibguest

今回の講習会では本番の環境を使わずにローカルの環境を使用します。



サイトにアクセス出来たらサイト上部のログインをクリックしてください。 ログイン画面に移行しユーザー名またはメールアドレスとパスワードを入力する欄が ありますので入力してログインをしてください。

		● <u>ログイン</u>	Google"カスタム検索		検索
UNIVERSITY OF AIZU		ミドルウェア	ライブラリ	ドキュメント	フォーラム
ログイン					
<u>トップページ</u> > ログイン					
ユーザ- パスワ-	-名またはメールアドレス -ド:				

② コンポーネントのアップロード手順

コンポーネントをアップロードするにはログイン後、トップページから「ライブラリ」 を選択し、「コンポーネント登録/パッケージ登録」のタブを選択します。そして「コン ポーネント登録」を選択します。

<u>トップページ</u> > ラ-	イブラリ		
ライブラリ検索	コンボーネント登録/パッケージ登録		
コンポーネン	ト登録/パッケージ登録		
RTコンポーネントま 利用規約などに同意	たはパッケージの登録を行います。 の上、ガイドラインに従って登録をお願いい	たします。	
RTコンポーネントま 未登録の場合は、先	たはパッケージの登録には、会員登録および に会員登録を行っていただき、ログインをお	グログインが必要です。 3願いいたします。	
コンポーネント	登録 ▶ パッケージ登録		
▶ 登録ガイドライ	2		

下記登録画面に遷移したことを確認してください。下記画面で登録を行います。

	🌢 <u>ようこそ、もうえ様</u>		Google"カスタム検索		検索
UNIVERSITY OF AIZU		ミドルウェア	ライブラリ	ドキュメント	フォーラム
コンポーネント登録					
<u>トップページ</u> > <u>ライブラリ</u> > コン	ポーネント登録				

今回は下記項目を登録します。

RTC.xml ファイル読み込み

WEB カメラコンポーネントで作成された RTC.xml を指定します。指定後、「RTC.xml ファイル読み込み」のボタンを押してください。 RTCBuilder で設定したコンポーネントの情報が登録されます。

コンポーネント登録情報入力

・コンポーネント名:RaspWEBCamera
・概要:get imageDate in WEBCamera
・カテゴリ:カメラ
・タグ:C++、OpenCV、画像処理
・ファイルアップロード:コンポーネントを zip に圧縮してアップロードします。その際、build 以下は削除か退避しておいてください。
・同意する:チェックを入れてください。
・私はロボットではありません:チェックを入れてください。※ローカル環境ではなし

入力が終わりましたら、「確認」のボタンを押し登録情報確認ページに遷移してくださ い。