青梅商工会議所主催 RTM講習会

日時:2011年7月25日(月) 10:30~17:30 場所:産業技術総合研究所 中央第2 本部情報棟1F ネットワーク会議室



RTミドルウェア講習会



10.20	第1部:RTミドルウェアの概略紹介
10.30-	担当:神徳 徹雄(産業技術総合研究所)
	概要: RTミドルウェア, RTコンポーネントの概要説明
11.15	第2部:RTミドルウェアの概略,導入方法の紹介
11:13-	担当:栗原眞二(産業技術総合研究所)
	概要: サンプルシステムを用いた概略紹介. RTミドルウェアの導入方法について紹介
12.00	第3部: RTミドルウェアを用いたシステム構築方法の紹介
13:00-	担当:坂本武志(株式会社グローバルアシスト)
	概要:簡単なサンプルの動作(実習1)基本的な仕組み,機能の紹介
	第4部:RTコンポーネントの作成方法の紹介
14:15-	第4部:RTコンポーネントの作成方法の紹介 担当:坂本武志(株式会社グローバルアシスト),栗原眞二(産業技術総合研究所)
14:15- 15:15	第4部:RTコンポーネントの作成方法の紹介 担当:坂本武志(株式会社グローバルアシスト),栗原眞二(産業技術総合研究所) 概要:サンプルコンポーネントの作成(実習2)RTCの設計方法の紹介
14:15- 15:15	第4部:RTコンポーネントの作成方法の紹介 担当:坂本武志(株式会社グローバルアシスト),栗原眞二(産業技術総合研究所) 概要:サンプルコンポーネントの作成(実習2)RTCの設計方法の紹介 第5部:OpenRTM-aistコマンドラインツール rtshellの利用方法
14:15- 15:15 15:30- 16:00	第4部:RTコンポーネントの作成方法の紹介 担当:坂本武志(株式会社グローバルアシスト),栗原眞二(産業技術総合研究所) 概要:サンプルコンポーネントの作成(実習2)RTCの設計方法の紹介 第5部:OpenRTM-aistコマンドラインツール rtshellの利用方法 担当:Geoffrey Biggs(産業技術総合研究所)
14:15- 15:15 15:30- 16:00	第4部:RTコンポーネントの作成方法の紹介 担当:坂本武志(株式会社グローバルアシスト),栗原眞二(産業技術総合研究所) 概要:サンプルコンポーネントの作成(実習2)RTCの設計方法の紹介 第5部:OpenRTM-aistコマンドラインツール rtshellの利用方法 担当:Geoffrey Biggs(産業技術総合研究所) 概要:rtshellの紹介と、システムの実行. ログのとりかた、ログの再生方法の紹介
14:15- 15:15 15:30- 16:00	第4部:RTコンポーネントの作成方法の紹介 担当:坂本武志(株式会社グローバルアシスト),栗原眞二(産業技術総合研究所) 概要:サンプルコンポーネントの作成(実習2)RTCの設計方法の紹介 第5部:OpenRTM-aistコマンドラインツール rtshellの利用方法 担当:Geoffrey Biggs(産業技術総合研究所) 概要:rtshellの紹介と、システムの実行.ログのとりかた、ログの再生方法の紹介 第6部:RTミドルウェアの便利な機能
14:15- 15:15 15:30- 16:00 16:15- 17:30	第4部:RTコンポーネントの作成方法の紹介 担当:坂本武志(株式会社グローバルアシスト),栗原眞二(産業技術総合研究所) 概要:サンプルコンポーネントの作成(実習2)RTCの設計方法の紹介 第5部:OpenRTM-aistコマンドラインツール rtshellの利用方法 担当:Geoffrey Biggs(産業技術総合研究所) 概要:rtshellの紹介と、システムの実行.ログのとりかた、ログの再生方法の紹介 第6部:RTミドルウェアの便利な機能 担当:片見 剛人(富士ソフト株式会社)



第4部 RTコンポーネントの作成方法の紹介

株式会社 グローバルアシスト 坂本 武志 産業技術総合研究所 栗原 眞二



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++

> 0++

> 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

RTC Builder - Flip/RTC.xml - ファイル(E) 編集(E) ナビゲート(N) 検	Eclipse SDK 索(A) プロジェクト(P) 実行(P) ウインドウ(W) ヘルプ(H)	
📬 • 🔛 🛎 🏤 💁 •] 😂 🖋 •] 🔄 • 🎘 • 등 💠 • ⇒ •	🗄 🛃 RTG Builder 🛛 🛪
🗯 パッケージ・エクスプロ 😫 🦳 🗖	> Flip 23	- D
= 🔩 🏹	基本	
E 😂 Flip	▼ RT-Component Basic Profile	▼ E>F
· · · · · · · · · · · · · · · · · · ·	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コンボーネントを識別する名前を指定します。
	*モジュール名: Flip モジュール調査: Flip image component	しの名利なコンホーネントのハースインスタンス名にも100円にも1189。 使用できる文字はアルファベット、数字、ハイフン、アンダースコアのみです。
	*バージョン: 1.00	モジュール教要: RIコンボーネントが提供する機能の教要を入力します。 ASCII文字が使用できます。
	*ベッタ治: AIST *モジュールカテガリ・Category	バージョン: RTコンボーネントのバージョンを指定します。 xyzなyzは数字)の形式で入力してください。
	10月1日の10日 10日 10日 10日 10日 10日 10日 10日 10日 10日	ペンダ名: RTコンボーネントを作成した作者名、ベンダ名を指定します。 ASCII文字が使用できます。
	Pbティヒティ型: PERIODIC	モジュールカテゴリ: RTTンボーネントのカテゴリを入力します。 遅択敗ったい場合は仕意のカテゴリ名を入力することができます。 使用できる文字化、アルラマベット数字、ハイフン、アンダースTPのみです。
	コンポーネント種類: 🔽 DataFlow 🔲 FSM 🔲 MultiMode	コンボーネント型: RTコンボーネントの型を指定します。 ・STATIC 動物に工生成消除されないRTC ・UNDUE 動物に実成消除されるユニークなRTC
	最大インスタンス数:1	COMMUTATIVE 動的に生成可能なRTC アカティビティ型・ RTTンボーネント/NPDティビティ型を指定します。
	実行型: PeriodicExecutionContext ま行周期: 1000.0	・PERIODIC、実行周期が固定の周期実行型RTC ・SPORADIC、実行周期が固定の周期実行型RTC
	OpenCVライブラリのうち、cvFlipO関数を用いて画像の反転を行うコンポーネント	・EVENTDRIVEN:イベント駆動型RTC コンボーネント種類: RTコンボーネントの種類を指定します。
	概要:	- DataFlowComponent
	RTC Type :	最大インスタンス数:生成可能なインスタンス数を指定します。制限がない場合は0を指定します。 また可じたかには、
	▼ コート主体2パック =>10 コードの生成およびパッケージ化を行います。	実行型: 実行型を指定します。 実行周期: コンボーネントアクションの実行周期をIHz)単位で指定します。
	コード生成 パッケージ化	この感覚を値はテータフロー型コンボーネントのみで有効です。 概要: RTコンボーネントの簡単な説明を記述します。
	▼ ブロファイル情報のインボート・エクスボート	RTC Type: 特定機能を実現するRTCの種類を区別する必要がある場合に使用します。 値が省略された場合には通常のRTCとして解釈されます。
	プロファイル情報のインボートおよびエクスボートを行います。	
	インボート	コード生成: 設定した情報を惹くRTOのスクルトンコードを主成します。 パッケージ化: RTCのソースコード、実行用バイナリコードなどを各種形式でアーカイブします。
		インボート: RtcProfile形式で保存した情報を読み込みます。 パッケージング: 設定した情報をRtcProfile形式で出力します。
	基本「アクティビティ データボート サービスボート コンフィギュレーション ドキュメント生成 言語・環境 RTCxm	1
	BuildView 🛛	- 8
	🛛 originallmage	flippedImage
	l l l l l l l l l l l l l l l l l l l	lip





RTC Builder - Flip/RTC.xml - Eclipse SDK _ 🗆 🗡 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) 📑 • 🔛 🖻 🛛 🏠 🛛 💁 • | 🇀 🛷 • | ½ × ½ × ↔ ↔ • → • 🔛 💦 RTC Builder 🛪 🐣 🗖 🗖 🜔 🗡 Flip 🛛 📕 パッケージ・エクスプロ 😒 - -F 🕵 基本 🖃 彦 Flip ▼ RT-Component Basic Profile ヒント RTC.xml RTコンボーネントを識別する名前を指定します。 この名称はコンボーネントのベースインスタンス名にも使用されます。 使用できる文字はアルファベット、数字、ハイフン、アンダースコアのみです。 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: *モジュール名: Flip モジュール概要: Flip image component RTコンボーネントが提供する機能の概要を入力します。 ASCII文字が使用できます。 モジュール概要: *バージョン: 1.0.0 RTコンポーネントのバージョンを指定します。 x.y.z(x,y.z(は数字)の形式で入力してください。 *バンダ名: AIST バージョン: *モジュールカテゴリ: Category -RTコンポーネントを作成した作者名、ベンダ名を指定します。 ASCII文字が使用できます。 べンダ名: コンポーネント型: STATIC • RTコンボーネントのカテゴシを入力します。 選択肢にない場合は任意のカテゴン名を入力することができます。 使用できる文字は、アルファベット、数字、ハイフン、アンダースコアのみです。 モジュールカテゴリ: PERIODIC アクティビティ型: • RTコンボーネントの型を指定します。 ・STATIC: 動的に生成/削除されないRTC ・UNIQUE: 動的に生成/削除されるユニークなRTC コンポーネント型: コンポーネント種類: 🔽 DataFlow 🔲 FSM 🔲 MultiMode ・COMMUTATIVE動的に生成可能なRTC 最大インスタンス数: Lのアレー・ビディ型を指定します。 この周期実行型RTC 固定の周期実行型RTC 動型RTC アクティビティ型: -RTCプロファイルエディタ /ポーネント 4 コンポーネント種類: 定します。 実行型RTC 狀態遷移型RTC ・MultiMode: マルチモード型RTC パッケージ・ エクスプローラ RTC Type : 最大インスタンス数: 生成可能なインスタンス数を指定します。制限がない場合はOを指定します。 ▼ コード生成とパッケージ化 実行型: 実行型を指定します。 コンポーネントアクションの実行周期を[Hz]単位で指定します。 この設定値はデータフロー型コンポーネントのみで有効です。 実行周期: コードの生成およびパッケージ化を行います。 コード生成 パッケージ化 RTコンポーネントの簡単な説明を記述します。 概要: RTC Type: 特定機能を実現するRTCの種類を区別する必要がある場合に使用します。 値が省略された場合には通常のRTCとして解釈されます。 ▼ ブロファイル情報のインボート・エクスボート プロファイル情報のインポートおよびエクスポートを行います。 コード生成: 設定した情報を基にRTCのスケルトンコードを生成します。 インポート エクスポート パッケージ化: RTCのソースコード、実行用バイナリコードなどを各種形式でアーカイブします。 インポート: RtcProfile形式で保存した情報を読み込みます。 設定した情報をRtcProfile形式で出力します。 パッケージング: 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 ドキュメット生成 言語・環境 RTC xml 🕼 BuildView 🕺 🔀 originalImage flippedImage D Flip ビルドビュー



- 以下から「USBCamera.zip」をダウンロードします。 <u>http://www.openrtm.org/openrtm/ja/node/1677#document</u>
- 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。





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
-------------	------

● パースペクティブを開く	
 □C/C++ □CVS リボジトリー・エクスプローラー ② Java のごうオルト) ③ Java の型階層 ③ Planning ④ Plaewing ③ Planning ● Plydev IT Fisystem Editor ● RTO Builder ● チーム同期化 ③ デバッヴ ◇ デバッヴ ● アジウイン開発 ● リソース 	
Ок 🖓 🔺	ャンセル

※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

プロジェクト作成/エディタ起動



ペロミの場所にフロシェッ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: USBCamera

②「プロジェクト名」欄に入力し、「終了」

RT-Component Builder Project	
プロジェクト名(P): USBCameral	
✓ デフォルト・ロケーションの使用(D)	
ロケーション(L): C¥work¥workspace200¥USBCamera	参照(B)
② 完成	E) キャンセ

③指定した名称のプロジェクトを生成









①「基本」タブ下部の「インポート」ボタン をクリック







■ 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





RTCプロファイルエディタ



画面要素名	説明
基本プロファイル	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

≯*USBCamera S		4	•	5
基本				7
- RT-Componen	nt Basic Profile	* K21		
このセクションではR	Tコンポーネントの基本情報を指定します。	モジュール名: RTI	レポーネントを識別する名前を指定します。	
*モジュール名:	USBCamera	この: 使用	名称はコンボーネントのベースインスタンス名にも使用されます。 Tできる文字はアルファベット、数字、ハイフン、アンダースコアのみです。	
モジュール概要:	USB Camera component.	モジュール概要: RT=	レポーネントが提供する機能の概要を入力します。 ITウマネが使用できます	
*バージョン:	1.0.0	バージョン: RT	レポーネントのバージョンを指定します。	
*バンダ名:	AIST		は、y.zは数子200形式で入力しくなさい。 ℃/ポーネントを作成した作者名、べつダ名を指定します。	
*モジュールカテゴリ	: ImageProcessing	ASC ASC	江文字が使用できます。	
コンポーネント型:	STATIC	ビンユールカテコリ: RI_ 選択	1277-1	
アクティビティ型:	PERIODIC	使用 コンポーネント型: RT	🎇 モジュール名: 📙	SBCamera
コンポーネント種類	DataFlow FSM MultiMode	・51 ・UI ・C(マカチッパティボリ・ロエー		広告(IISB Comoro component)
最大インスタンス数	: 1		◎ モンユ―ル 例安・	TR: (USD Calliel a Cullipulielit)
実行型:	PeriodicExecutionContext	EV ·EV		
実行周期:	1000.0	コンボーネント種類: RTコ ・Da	LaFlot ハーンヨノ・I_U_U	
概要:	USB camera component using OpenCV		◎… ベンダ名:任意(/	AIST)
RTC Type :		実行型: 美行型: 美行型: 美行型: 美行型: ことの		UL 任音 (ImageDragonaling)
▼ コード生成とパッ	ッケージ化	概要: RT_	こう エンユールハノー	リ 在息(IIIIager I Ocessing)
コードの生成および	パッケージ化を行います。	RTC Type: 特定 値が		
コード生成 パッ	ケージ化		ゔ コノ小一个ノΓ空	
▼ ブロファイル情報	<u><u></u> <u> <u> </u> のインボート・Tクスボート</u></u>	コード生成: 設定		
プロファイル情報の	インボートおよびエクスボートを行います。	707 712. 100	◎ プクティヒティ空・	PERIUDIU
インボート エクス	スポート	インボート: Rtcl		1手 坐王・ □ - ↓ - □
		パッケーソンク: 設定	◎ コンホーネントの	不里 突見: しるtafiow
			┃最大インスタンス	
基本 アクティビティ デ	ータボート サービスボート 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		
			■ 軍行型: Periodic	-xecutionContext
			大门王····································	
			【宝行周期·1000	
				V

※エディタ内の項目名が赤字の要素は必須入力項目 ※画面右側は各入力項目に関する説明

アクティビティ・プロファイル



■ 生成対象RTCで実装予定のアクティビティを設定

アクティ	ビティ			-	1 ①設定対象のアクティビティ
▼ PDT1ET1			◆ ピント		
このセクションでは	は使用するアクションコールバックを指定します。		onInitialize	初期化処理です。コンポーネントライフサイクル開始時に一度だけ呼ばれます。常に有効。	を選択
	コンポーネントの初期化と終了処理に開	関するアクション	onFinalize	終了処理です。コンボーネントライフサイクルの終了時に1度だけ呼ばれます。 EventimeContentが実行す問題なするとき1度だけ取ざれます。	
onInitialize	onFinalize		onShutdown	ExecutionContextが実行を停止するとき1度だけ呼ばれます。	
	実行コンテキストの起動と停止に関	するアクション	onActivated	非アクティブ状態からアクティブ化されるとき1度だけ呼ばれます。	
onStartup	onShutdown		onDeactivated	アクティブ状態から非アクティブ化されるとき1度だけ呼ばれます。	onActivated N
	alive状態でのコンポーネントア	ゆション	onAborting	ERROR状態にへる前に1度だけ判認れます。 ERROR状態にいる問問期的に呼ばれます。	on Error
onActivated	onDeactivated	onAborting	onReset	ERROR状態がらリセットされ非アクティブ状態に移行するときに1度だけ呼ばれます。	
onError	onReset		onExecute	アクティブ状態時に周期的に呼ばれます。	
	Dataflow型コンポーネントのア	ウション	onStateUpdate	e onExecuteの後毎回呼ばれます。 オーローンがManagement Andrews	
onExecute	onStateUpdate	onRateChanged	on Rate Unange on Action	a ExecutionContextのFate/が変更されたことで呼ばれます。 対応する状態に応じた動作を実行するために呼ばれます。	(2) 使用/ 未使用を設定
	FSM型コンポーネントのアクシ	ອ້ອງ	onModeChang	ed モードが変更された時に呼ばれます。	
onAction			41.11.1 117 77		
	Mode型コンポーネントのアク	ション	動作	アクティビティの破壊説明を記述します。 アクティビティを実行する前に成立すべき事前条件を記述します。	
onModeChange	d		事後条件:	アクティビティを実行した後に成立すべき事後条件を記述します。	
👻 Documenta	tion				
このセクションでは 上段のアクション	t各アクションの概要を説明するドキュメントを記述し を選択すると、それぞれのドキュメントを記述できます	<i>」</i> ます。 す。			
アクティビティ名:	onInitialize	• ON • O	FF		
	コンボーネント自身の各種初期化処理	L	<u>~</u>		以下をチェック:
動作磁要:			×.		
	なし		<u></u>		
事前条件:					
			*		I onDeactivated
	ロンボーネントの初期化処理が正常に完了してい	NS	<u>_</u>		
事後条件:			-		
基本 [アクティビティ]	データボート サービスボート コンフィギュレーション	/ ドキュメント生成 言語・環境 RTC	×ml Mapping ID USB P	nP RTC-CANopen	

※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





、キキンル毛粘の個の「ヘ」ム」ーヒゟ、 そんにぃら

生成対象RTCに付加するDataPortの情報を設定

データポート	①該当性類の欄の「Auu」小ダンをクリック
→ DataPortプロファイル	□→▷▶ Ⅰ.ポートを追加後 直接入力で名称設定
このセクションではRTコンポーネントのDataPort(データポート)の情報を設定します。	
*ボート名 (InPort) Add Add Delete Add Delete	PoweiDiper iPorticOutPort InPort: RTコンボーネント 他のRTコンボー・・ OutPort: RTコンボーネント 他のRTコンボー・・ オート名: データ地上トを識
▼ Detail このとわい、スパーデ、5.米、1.年の概念★1900+7.1%といいしままします。	ホート24は、ID- ポート名に対して ASCI文字が使
とのセンションにはチーダホート毎の構要で認知するトイエンントが記述できます。 上のデータボートを選択すると、それぞれのドキュメントが記述できます。 ボート名: [image (OutPort)	データ型: データポート間で inPortとOutPort データ型はOpen 使用することがで
*データ型 RTC::CameraImage	^{変数名: データポートに関} 変数の名称は ポートの場所: RTSystemEdite 2設定する型情報を一覧から選択
え以名 Image 表示位置 RIGHT	このプロパティは2 ドキュメント: データボートに関 全てな見がする。
Documentation	レベルの情報を言うした。 レベルの情報を言うした。 レベアレの情報を言うした。 レベアレの情報を言うした。 レベアレの情報を言うした。
Capture images data from the camera 不要說明:	エのフーシスティーを通知すると、これにものフィエンフトが高した」できょう。 ポート名: image (OutPort)
データ型: RTC:CameraImage)	*データ型 RTC::Acceleration2D マ
デーク数:	変数名 RTC:BumperGeometry RTC:BumperGeometry
意味:	表示位置 RTC::CameraInfo RTC::Carlike

※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

BuildView 🛛 image USBCamera		- 8	
		image D	
	USBCamera		

サービスポート・プロファイル



生成対象RTCに付加するServicePortの情報を設定

サービスポート					
RI-Component Service Ports	Add Port Add Interface Delete	RT-Component S このセクションではRT *インターフェース名: 方向: インスタンス名: 変数名: *IDLファイル: *インターフェース型: IDLパス: ▼ Documentati	Service Port Interface Profile コンポーネントのService Interfaceの情報を設定します。 if_name Provided C:¥work¥MyService.idl MyService MyService2 MyService3 MyService5		Browse Browse
		概要說明: 		*	

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

コンフィギュレーション・プロファイル



■ 生成対象RTCで使用する設定情報を設定

RT-Comp	onent Configuration Parameter Definitions		* ビント		
このセクションで	ではRTコンボ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。		Config. Param.: RT	直接人刀で名称設定	
*名称		Add	再利	▼ RT-Component Configuration Parameter Definitions	
deviceNumb	ber	Delete	ハラス パラメータ名: コンフ	このセクションではRTコンボ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。	
			パラ: 名前	*名称 Add Add	
			データ型: コンフ	Cont_nameU	
			デフォルト値: コンフ		
			R1二 解釈		
🕶 Detail			変数名: コンフ 実際		
このセクションで	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。		単位: コンプ		
パラメータ名:	deviceNumber		制約条件: コンフ ・指知)詳細両面にて 刑情報 恋粉々たど	た記
*=- 5#0	1004		・100 ・重領	シ計杣凹山にて,主旧和,友奴石ると	
*ナーツ空			• 列3		1
*ナノオルド地 夜逝かを			ייייייייייייייייייייייייייייייייייייי	I 名称: deviceNumber	
夏 <u>奴</u> 石: 単位:			Step: 設定		
制約条件:				I データ型: int	
Widget:	text	~			
Sten:				デフェルト値・∩	
Description	11			/ / / / / l · U	
- Unclimenta	100			赤粉々・dovigoNumbo	J
				炙致石·ueviceivuilibe	
				制約粂件:	
				I Widget' text	

※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<=x<=100
 - 列挙型: (値1,値2,・・・)
 - ▶ 例) (val0, val1, val2)
 - 配列型: 值1, 值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

			► E2F
)セクションでは使用する言語 C++ Python Java Ruby	を指定します	Use old build environment.	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、プロファイル内にのみ保存されます。
粟境			
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		
ersion	OS	Add	
		Delete	
羊&田! 青幸辰			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
OS Version	AddCPU	Add	



RTコンポーネントの作成



RTコンポーネントの実装(概要)





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。

2011.7.25 青梅商工会議所主催 RTM講習会

KIコン小一インFの美装	
単体で動作確認済みのプログラムからRTコンポー	ーネントを作成
int main (int argc, char** argv) {	RTC::onInitialize() に実装
// キャプチャクラスのオブジェクトの初期化 cam->initialize();	RTC::onActivated() に実装
while(1) { // カメラからの画像キャプチャ処理 cam->capture();	RTC::onExecute() に実装

// 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::onDeactivated() に実装

USBカメラコンポーネントのソースファイル



RTコンポーネントの実装

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コンポーネントの実装



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;

RTコンポーネントの実装



```
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)



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

🗛 C Make 2.8.5 - C:/w	ork/workspace200/USI	BCamera/bu	uild	
File Tools Options He	elp			
Where is the source code:	C:/work/workspace200/USE	3Camera		Browse <u>S</u> ource
Where to build the binaries:	C:/work/workspace200/USE	3Camera/build	l 🗸	Browse <u>B</u> uild
Search:	Grouped A	Advanced 🖪	Add Entry	🗱 <u>R</u> emove Entry
Name				
<				>
Press Configure to update	and display new values in rec files.	d, then press G	àenerate to gene	rate selected build
Configure Generate	Current Generator: None			
	<u>चर्ल</u> ा			

②「Configure」を実行し、使用する プラットフォームを選択

Where is the source co	de: C:/work/workspace200/USBCamera Browse Source
Where to build the bina	ries: C:/work/workspace200/USBCamera/build 🛛 💽 Browse <u>B</u> uild
Search:	🔺 cmake-gui 🦳 🖓
Name	Opeolify the generator for this project
Press Configure to up Configure Gen	Visual Studio 9 2008 Ise default native compilers Specify native compilers Specify toolchain file for cross-compiling Specify options for cross-compiling

※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





④binary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

💐 USBCamera – Visual C++ 2008 Expre	ess Edition	
ファイル(E) 編集(E) 表示(V) プロジェクト(P)	ビルド(B) デバッグ(D) ツール(T) ウィンドウ(W) ヘルブ(H)	
🛅 • 🛅 • 😂 🖬 🖉 🕺 🗴 🗈 🛍 🔊 •	ガリューションのビルド(B) F7 ア	** ਦ
YJューション エクスプローラ - ALL_BU、 ・ Q × () () () () () () () () () () () ()	ソリュージョンのリビルド(B) Orti+Alt+F7 ソリューションのクリーン(Q) ALL_BUID のグルド(Q) ALL_BUID のグリドル(E) ALL_BUID のグリーン(Q) プロジェクトのみ(Q) パッチ ビルド(D). 構成マネージャ(Q)-	
出力		→ ₽ ×
出力元の表示(S):		
コード定義ウィンドウ き 呼び出しブラウザ コード マネカ マンド		

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







動作確認



- 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。

🚾 RT System Editor RCP				🚾 RT System Editor RC	CP				
<u>File Window H</u> elp				<u>File Window H</u> elp					
: 👯 📴 🔛 🗰 🍪 🔐				i 💀 öfe i 🖻 👹 🍪	1				
🏄 Name Servic 🕥 RepositoryVi 😑 🗖	🔐 *System Diagram 🔀] 🔲 プロパティー	~	🗯 Name Servic 🍈 Repos	ositoryVi 🗖 🗖	ön system Diagram 🔀	- 0	🔲 プロパティー	~ - 8
🖞 🗘 🖓 🗸 🗸		プロパティー	値 🛆					プロパティー	値
Grahost		😑 🔤 CameraViewer0						🖃 📴 USBCamera0	
ia op aist-sk host_cxt	· · · · · · · · · · · · · · · · · · ·	Path URI	localhos	ia oper ist−sk∦	host_cxt	· · · · · · · · · · · · · · · · · · ·		Path URI	localhost/o
- UCDO		Instance Name	Camera	a sraviewei	erU rtc			Instance Name	USBCamer-
		Type Name	Camera	USBCamerau)	JINC		1 I I	Type Name	USBCamer-
		Description	USB Ca			USBCanteral		Description	USB Came
		Version	1.0.0					Version	1.0.0
		Vendor	AIST					Vendor	ADT
	CameraViewer0	Category	example				DameraViewer0	Category	ImageProce
		State	INACTO					State	INACTIVE
		properties						properties	
		See SugardianCa						E Owned	or .
		ID	0						0
		State	DUNINTS					State	RUNNING
		Kind	PERIOD					Kind	PERIODIC
		Bate	1000.0					Rate	1000.0
	Confi ZT Mana ZT Com ZT Exec ZT RT L 🖓 🗖	n Mart	1000.0			Confi RT Mana RT Com RT E	kec 犹 RT L 🖓 🗖	OutPort	1000.0
		Name	in					Name	USBCamer-
	ComponentNa ConfigurationSet: default 編集	Data Type	IDL:RTC			Componentiva ConfigurationSet: default	編集	Data Type	IDL:RTC/C.
	active c name	Interface Typ	o corba ci			active c name	1 法田	Interface Ty	/p corba.cdr
	G d image_height 2	Dataflow Ty	pullpusł			G d deviceNumber	0 2000	Dataflow Ty	/p pull,push
	image_width 3 キャンセル	Subscription	Any				キャンセル	Subscription	n flush,new,pr
			1.5						
		🖃 🁥 OutPort							
		Name	.Key out						
		Data Type	IDI:RTC			(複製)(追加)前間	Î.	<	
Pos: (206,56) Size: (93,99)				Pos: (35,57) Size: (93,39)					

動作確認



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。

	RT System Editor RGP		
	Eile Window Hop Baie, Rome & Baie Name Serve (1) RepositoryVi	🔊 •System Diagram 🖸 👘 🗖	70/7/
	10 4 4 1 B B 8 # 7		プロパティー (値
rvic	K loahort Gonera Vermöhrt, skt	USBCame a	Rf Dyslen Dozion System D Kind ONLINE Create Date Usdate Date Composite None
		Confi. 🔠 Mana. 🐮 Con. 🐮 Exec. 🗱 RT L. 🔍 🗆	
		ComponentNe ConfigurationSet	
		active c name V 10/11	
		54725A	

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。

ing ing : 📂 📕 🔐 🔐	🔤 Connector Profile 🛛 🔀		
🕼 Name Servic 🔞 RepositoryVi 🐃 🗆	6	 10/15-r	~ - [
🚡 🗢 🔿 📑 🏶 🏓 🏹		プロパティー	億
a ft kontert 0 generative to text 0 generative of the 0 users 0 users	Connector Profile & APULT (CES), w Name : DISECaneral Image, In Data Tope : DISECaneral Image, IN Detailow Type : contra, of w Detailow Type : pouth w Subservation: Type : Inuth w Tush Ratefold : Tush Ra	II // System Decem System Decem Create Oute Update Oute Update Oute Composite	ONLINE

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

科 tabs.css



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	Q• 核杰
構成 構成表引 構成メンバ	
クラス USBCamera	RTC::ReturnCode_t USBCamera::onDeactivated (RTC::UniqueId ec_id) [virtual]
	Post-processing such as freeing allocated memory.
USB Camera component. [詳細] #include <usbcamera.h></usbcamera.h>	RTC::ReturnCode_t USBCamera::onExecute (RTC::UniqueId ec_id) [virtual]
すべてのメンバー覧	Capture images from the camera, and outputs the data from OutPort.
Public メソッド	
USBCamera (RTC::Manager *m constructor	int USBCamera::m_deviceNumber [protected]
~ USBCamera () destructor	Device number
virtual RTC::ReturnCode_t onInitialize () virtual RTC::ReturnCode_t onActivated (RTC::UniqueId ec.	Name: deviceNumber deviceNumber DefaultValue: 0
virtual RTC::ReturnCode_t onDeactivated (RTC::UniqueId virtual RTC::ReturnCode_t onExecute (RTC::UniqueId ec_i	e OutPort <cameraimage> USBCamera::m_m_imageOut [protected]</cameraimage>
Protected 変数	Capture images data from the camera
int m_deviceNumber CameraImage m_m_image	Type: RTC::CameraImage
OutPort< CameraImage > m_m_imageOut	
説明	C:/work/workspace199/USBCamera/USBCamera.h C:/work/workspace199/USBCamera/USBCamera.cpp
	usbcameraに出せてFri Jul 15 2011 21:41:19に生成されました。 (山の)ぶりの色前 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

00001	// -*- C++ -*-
00015	#ifndef USBCAMERA H
00016	#define USBCAMERA_H
00017	
00018	<pre>#include <rtm manager.h=""></rtm></pre>
00019	#include <rtm dataflowcomponentbase.h=""></rtm>
00020	#include <rtm corbaport.h=""></rtm>
00021	#include <rtm datainport.h=""></rtm>
00022	#include <rtm dataoutport.h=""></rtm>
00023	<pre>#include <rtm basicdatatypeskel.h="" idl=""></rtm></pre>
00024	<pre>#include <rtm extendeddatatypesskel.h="" idl=""></rtm></pre>
00025	<pre>#include <rtm idl="" interfacedatatypesskel.h=""></rtm></pre>
00026	
00027	<pre>// Service implementation headers</pre>
00028	<pre>// <rtc-template block="service_impl_h"></rtc-template></pre>
00029	
00030	//
00031	
00032	// Service Consumer stub headers
00033	// <rtc-template block="consumer_stub_h"></rtc-template>
00034	
00035	//
00036	
00037	using namespace RTC;
00038	
00050	class USBCamera
00051	: public RTC::DataFlowComponentBase
00052	
00053	public:
00058	USBCamera(RTC::Manager* manager);
00059	
00063	~USBCamera();
00064	
00065	<pre>// <rtc-template block="public_attribute"></rtc-template></pre>
00066	

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。



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>

RTCBuilderの補足



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



③USBCameraComp¥¥Debug内の USBCameraComp.exeを起動



②VisualStudioを用いたビルド

フェ(また) 原来(に) 本二の人 ゴロ	(1-h)(0)	21 (M) 21 (- 50)	0 01/100		
		100-100 100-000 7-000 07-000			
	2		· · · · · · · · · · · · · · · · · · ·	 O D, D X, M F1 + ² 	
四四日日前第三日日	1 20	SIT-STRUCTURE COMMUNICI			
リューション エクスプローラ - ● 単 ×	USBC	202-232000-2020			
Barrowski State Andrewski State Barrowski State State State Barrowski State State State Barrowski State State Barrowski State State Barrowski State State Barrowski State Barrowski State Barrowski State	00-10	TRACERCON 1475	×		
	B// -	選択範囲のパビルドロ			
	/#!	違抗範囲のクリーン(2)			
	1.1	1699 EAFD			
		構成マネージャ(2)			
	* \$1	(d3)			
	*/				
	line	lude "USBCamera.h"			
	// Mo	dule specification			
	// <r< td=""><td>to-template block="module_spec"></td><td></td><td></td><td></td></r<>	to-template block="module_spec">			
	1				
		implementation_id", "USBCamera",			
		description", "USB Camera co	apponent.",		
		version", "1.0.0",			
		category", "ImageProcessi	ng",		
		'activity_type', "PERIODIC", 'kind", "DataFloaCompo	nen!".		
		mex_instance", "1",			
		lang type". "compile".			
	1	/ Configuration variables			
1911- 10/17. E. 70/17.	<	confiderault.devicemumber, 0,			
873					
出力 中力量の表示(S)		• 10 10 0 1 m 10			•

ドキュメント作成(Windows)

①Doxygen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

③「Output」セクションにて、html出力を指定

no 1. Consolity the work inst disastery it	en which dessens will us
Analy Alassen	Caler
ep 2: Configure doxygen using the W	izard and/or Lipert tab, then switch to the Hun tab to generate the documentation
Wicard Expert Run	
opeita Project Mode Output Diserami	Select Se output for motiful to generate If TML plan HTML mith narregation parel proper for compressed HTML (chm) With search transform (Change color.)
	a sitemadiate format for hyperinead PDF a sitemadiate format for PDF a sitemadiate format for PDF a sitemadiate format for PDIFCript
	[Man pages] Roh Text Format Q(TF) 20%.
	Previous

Mode」セクションにて、 出力内容,使用言語を指定

S Doxygen GUI frontend +	
File Settings Help	
Step 1: Specify the working directory from w	hich daxyeen will run
C:/work/doxygen	Select
Step 2: Configure doxygen using the Wizard	and/or Expert tab, then switch to the Run tab to generate the documentation
Topics	
rises.	Select the desired extraction mode
- Mode	 Documented entities colv
Diaerama	All Entities
	Include cross-referenced source code in the output
	- Select programming language to optimize the results for
	Optimize for C++ output
	C Optimize to C / CL Opport
	Optimize for Java or C# output
	Optimize for C or PHP output
	O Optimize for Fortran output
	O Optimize for VHDL output
	Previous Next

③「Run」タブにて、「Run doxygen」を実行

■ DataPortIこ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

青梅商工会議所主催 RTM講習会

日時:2011年7月25日(月) 10:30~17:30 場所:産業技術総合研究所 中央第2 本部情報棟1F ネットワーク会議室

