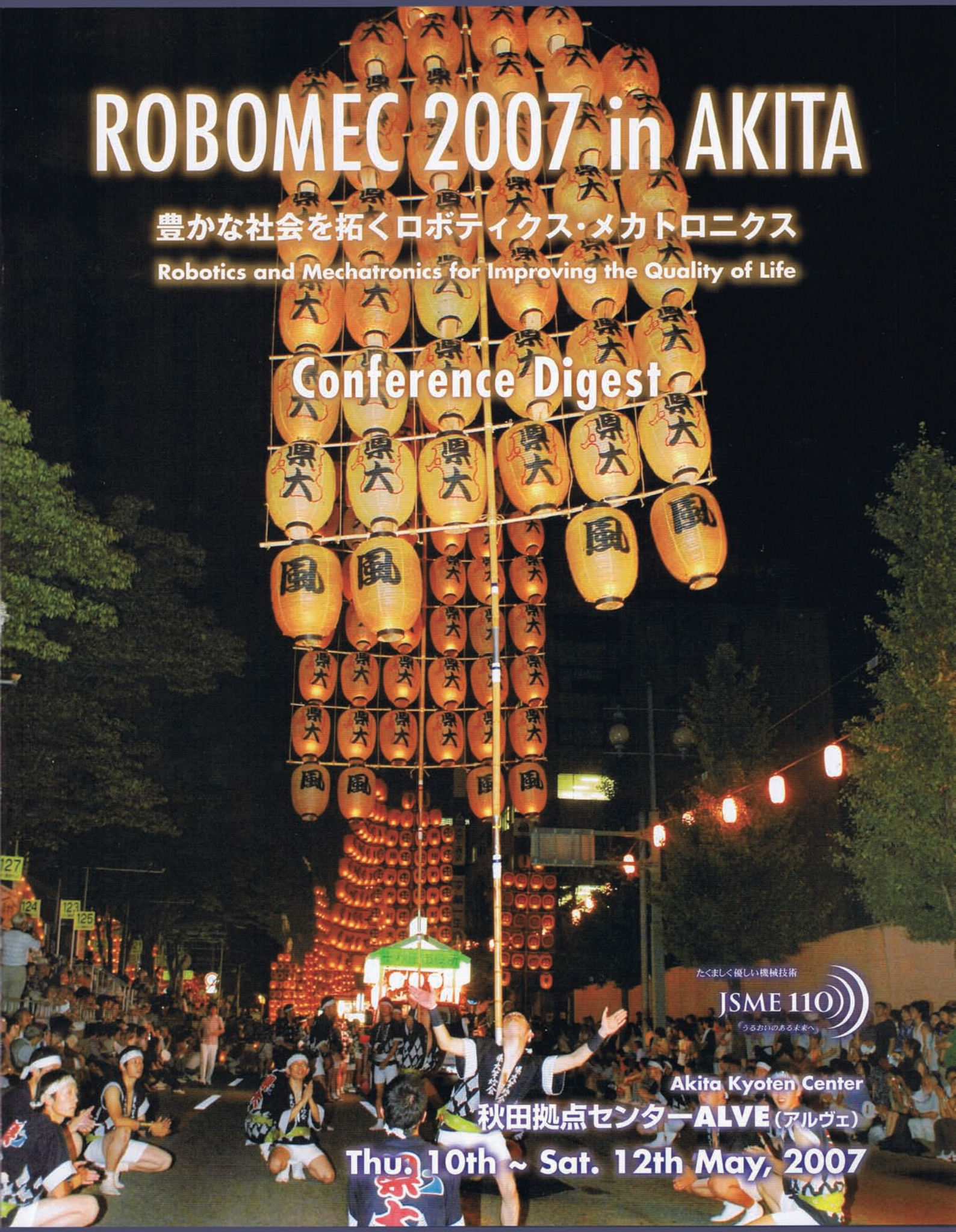
2007 JSME Conference on Robotics and Mechatronics
主催　社団法人 日本機械学会 ロボティクス・メカトロニクス部門

ROBOMEC 2007 in AKITA

豊かな社会を拓くロボティクス・メカトロニクス

Robotics and Mechatronics for Improving the Quality of Life

Conference Digest



たくましく優しい機械技術
JSME 110
「うるおいのある未来へ」

Akita Kyoten Center
秋田拠点センターALVE（アルヴェ）

Thu. 10th ~ Sat. 12th May, 2007

社団法人 日本機械学会
〒160-0016 東京都新宿区信濃町35番地 信濃町煉瓦館5階 TEL. (03) 5360-3500　FAX (03) 5360-3508

RT ミドルウェア用開発環境の構築

Building a development environment for RT-Middleware

正　安藤　慶昭（産総研）　　正　神徳　徹雄（産総研），
○坂本　武志（テクノロジックアート），　山下　智也（テクノロジックアート），
長瀬　嘉秀（テクノロジックアート）

Noriaki ANDO, Tetsuo KOTOKU, AIST
Takeshi SAKAMOTO, Tomonari YAMASHITA, Yoshihide NAGASE, TECHNOLOGIC ARTS

To utilize the merit of RT-Middleware, it is important to reduce development time and cost of RT-Component and to increase various RT-Components that everyone can use. So we built the development environment for RT-Middleware. This development environment is based on Eclipse that is Integrated Development Environment for multi-platform such as Windows, MacOS and Linux. And each function consists of various plug-in such as UML modeling tool Pattern Weaver. In this development environment, we can easily perform skeleton generation, assembling and operation check for RT-Component which is executed on OpenRTM-aist-0.4.0. In this paper, we propose a new development environment for RT-Middleware and introduce the details of functionality.

Key Words: RT-Middleware, Eclipse, Robot

1.　はじめに

急速な少子高齢化の進展による労働力不足や，個人生活の多様化といった背景を受けて，公共分野，医療・福祉分野をはじめとする非製造業分野や，エンターテイメント分野，ホームオートメーション分野といったパーソナル分野へのロボット技術の応用，いわゆる「サービスロボット」の実用化が期待されている．しかし，工場内のような定型的な環境の中で，ある一部の限られたユーザに利用される産業用ロボットと比較して，サービスロボットは，適用箇所，対象ユーザの範囲が格段に広がるため，これまで以上に多種多様な要求が発生することが予想される．

一方，現在の産業用ロボットの開発では，ロボット全体を一括して開発する，モノリシック的な開発方法が主流である．このため，多様な要求の発生が予想されるサービスロボットの開発に，従来の方法をそのまま適用するのは限界があり，より効率的な開発手法の確立が望まれている．

このような背景を受け，著者らはこれまで RT(Robot Technology)ミドルウェアの仕様策定，開発・実装を行ってきた[1]．RT ミドルウェアとは，ロボットの様々な機能要素を RT コンポーネントと呼ばれる部品単位に分割し，これらを自由に組み合わせることで，多様なロボットシステムの構築を可能にするための共通プラットフォームである．RT ミドルウェアを利用することで，RT コンポーネント単位での開発，再利用，入れ換えが可能となるため，多種多様な要求にも最小限の変更で柔軟に対応できるようになる．

しかし，RT コンポーネントを利用した開発手法のメリットを十分に活かすためには，Fig.1 に示すように，ソフトウェア資産としての RT コンポーネントを多数蓄積し，再利用できる RT コンポーネントの種類･数を充実させることが重要である．そして，そのためには RT コンポーネント自体の開発負荷を軽減し，既存の研究・開発成果，アルゴリズム，知識などを RT コンポーネント化する敷居を下げ，より開発効率を向上させる必要がある．

そこで本稿では，RT コンポーネントの開発負荷を低減するための開発環境の提案を行うとともに，その詳細について紹介する．

2．RT ミドルウェア

2．1．RT ミドルウェアによるシステム開発

RT ミドルウェアは，ロボットシステム向けの共通プラットフォームであり，RT コンポーネントと呼ばれる部品を基本とするコンポーネント・ベース開発を実現するためのフレームワークである．

コンポーネント・ベース開発のメリットは，提供する機能を定義する「インターフェース」の部分と，それを実現する「実装部分」を明確に分離することで，「実装部分」を変更した際の他コンポーネントへの影響を最小限に抑えられるとともに，適用範囲・用途に応じて「実装部分」の入れ替えが行える点である．更に，適切なコンポーネント化を行うことで，再利用性の向上や，抽象化による柔軟性を獲得することもできる．

しかしその一方で，コンポーネント単体のみでは，意味のあるソフトウェアとして動作させることは難しい場合も多く，動作確認・テストを実行するために他のコンポーネントとの接続が必要となるとともに，個々のオペレーションを呼び出すための何らかの仕組みも必要となってくる．

また，RT ミドルウェア上で動作する RT コンポーネントの開発を行う場合，開発者はコンポーネント・フレームワークに独自のロジックを埋め込む必要がある．この場合，フレー

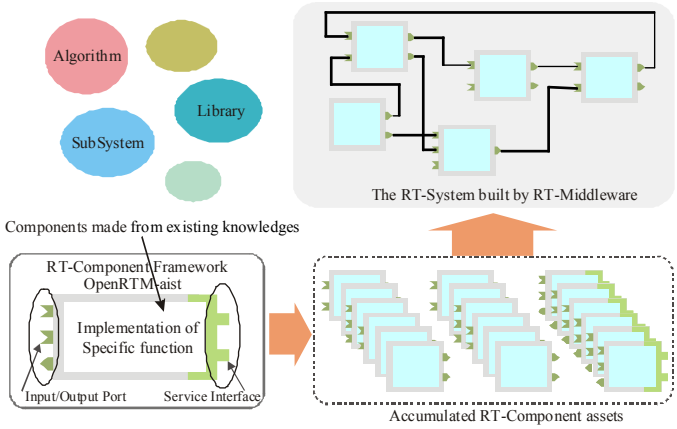


Fig.1 RT-Component-based development overview

ムワーク部分のコードは，定型的なものとなるため，コンポーネント仕様から自動生成することが可能である．

このような特徴を踏まえ，RT コンポーネントの開発効率を更に向上させるためには，RT コンポーネント向けの統合開発環境を構築することが有効である．一般に統合開発環境には，目的に応じて個別に使用していた各種ツール(エディタ，デバッガ，コンパイラなど)を統合し，統一された対話型インターフェースを提供することで操作性を向上させることができる．また，各種テストやデプロイ(配置)作業などの定型作業を自動化，簡略化することで，開発者の負担を下げることができる．といったメリットがある．これらの特徴は RT コンポーネント開発においても効果的であり，更に RT コンポーネント開発に固有の機能を付加することで，開発効率の向上が期待できる．

2. 2. OMG RTC 標準仕様

RT ミドルウェアの仕様を明確にするとともに，より一層広範囲への普及を図るために，オブジェクト指向技術の国際的な標準化団体である Object Management Group(OMG)にて，標準仕様 The Robotic Technology Component Specification[2](以下，OMG RTC Specification)の策定が進められている．本仕様は，モデル駆動型アーキテクチャ(Model Driven Architecture: MDA)の考え方に基づき，各種実装技術や開発プラットフォームに依存しないモデル(Platform Independent Model:PIM)と実際の実装技術(CORBA,C++)をターゲットとしたモデル(Platform Specific Model:PSM)を明確に定義しており，以下のような特徴がある．

・自律分散システム向けの OMG 標準仕様である Platform Independent Model(PIM) and Platform Specific Model(PSM) for Super Distributed Object(SDO) Specification[3]を拡張した構造の採用

・RT コンポーネントが最小限備えるべき機能(lightweightRTC)と，付加的な機能(Introspection)を分離

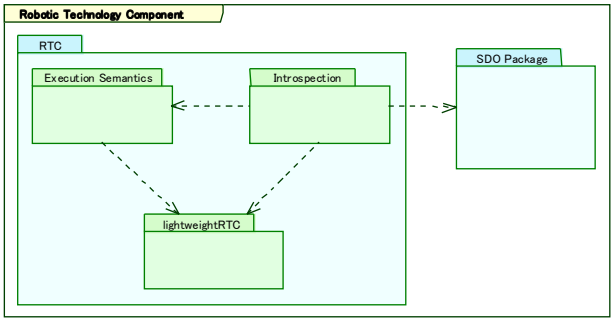


Fig.2 Robotic Technology Component package structure

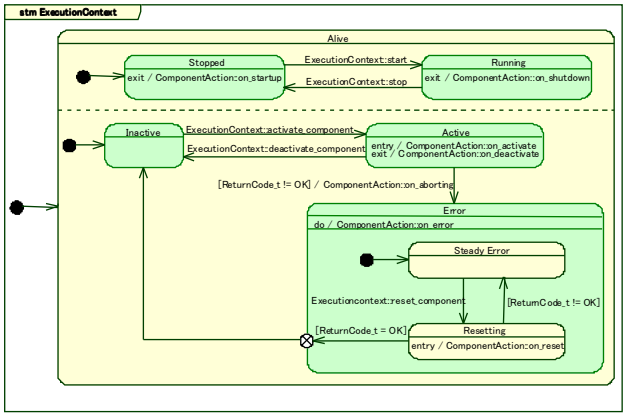


Fig.3 Executioncontext statemachine diagram

・機能を提供するコンポーネントと，論理的な実行主体(ExecutionContext)を明示的に分離

・コンポーネント，論理的な実行主体の状態遷移の改良

OMG RTC Specification の全体パッケージ構成を Fig.2 に，論理的な実行主体である ExecutionContext のステートマシン図を Fig.3 にそれぞれ示す．

なお，OMG RTC Specification は 2006 年 9 月に標準仕様案として採択され，現在最終的な仕様の検討作業中であり，2007 年秋頃に OMG 公式標準仕様として発行される予定である．

2. 3. OpenRTM-aist-0.4.0

OpenRTM-aist-0.4.0 は，著者ら(産業技術総合研究所)が公に配布している RT ミドルウェアの最新版であり，以下のような特徴がある．

・OMG RTC Specification 準拠

・データポート(InPort/OutPort)の改善

・サービスポートの導入

・Configuration インターフェースの導入

なお，今回提案する開発環境は OpenRTM-aist-0.4.0 上で動作する RT コンポーネントを対象としている．

3．開発プラットフォーム Eclipse

Eclipse は，オープンソース・コミュニティ Eclipse Foundation で開発が行われている統合開発環境で，以下のような特徴がある[4]．

・「Plug-in」を追加することで，使用目的に応じて柔軟に機能拡張が可能．

・Rich Client Platform（RCP）という仕組みを利用することで，スタンド・アローン・アプリケーションを作成することが可能．

・マルチプラットフォームに対応しており，Windows や MacOS, Linux など複数の OS 上での利用が可能．

一般に，ベース部分である Eclipse Platform が JavaVM 上で動作し，EclipseSDK として配布される標準パッケージ内に Java Development Tools(JDT)が含まれているため， Eclipse は Java 用の開発環境というイメージが強いが，内部的にはこの JDT も Plug-in として実現されている．そのため，Plug-in の追加により，Java 以外のプログラミング言語用開発環境を構築することも可能となっている．

RT ミドルウェアは，多様なロボットシステムの開発に利用可能な共通基盤であり，その開発プラットフォームは多岐に渡ることが予想される．また，ロボットシステム自体も非常に多様であるため，開発形態や開発時に必要となるツール・機能，開発言語も様々である．従って，RT コンポーネントの開発には，様々な開発プラットフォームで利用できるとともに，必要なときに必要な開発ツールを容易に追加できるような開発環境が有効である．

OpenRTM-aist-0.2.0 に付属していた RtcLink はコンポーネント接続・実行などが可能であり，RT コンポーネント開発において重要なツールであったが，ベースとなっている GUI ツールキット wxPython 上に直接構築されていたため，新たなツールの追加，機能のカスタマイズなどが難しかった．これに対して，統合開発環境 Eclipse は，ソフトウェア開発環境として広く普及しており，事実上のデファクト・スタンダードの地位を確立しつつあるとともに，上述の条件を全て満たしているため，RT コンポーネント用の開発環境を構築するのに適したプラットフォームであると考え，今回提案する統合開発環境の基盤として採用した．

4．RT コンポーネント開発環境

4. 1. RT コンポーネント開発サイクル

ソフトウェアの開発サイクルは，大きく分類して，要件定義・設計・実装・テストという工程からなり，統合開発環境は，これら各工程内の作業をアシストするとともに，工程間をシームレスに繋げる役割がある．今回提案する開発環境は，Fig.4 に示すように，それぞれの工程をサポートする以下のツールから構成されている．

設計：パターンウィーバー ＋rtc-template Plug-in
実装：Eclipse C/C++ Development Tooling - CDT
テスト：RtcLink

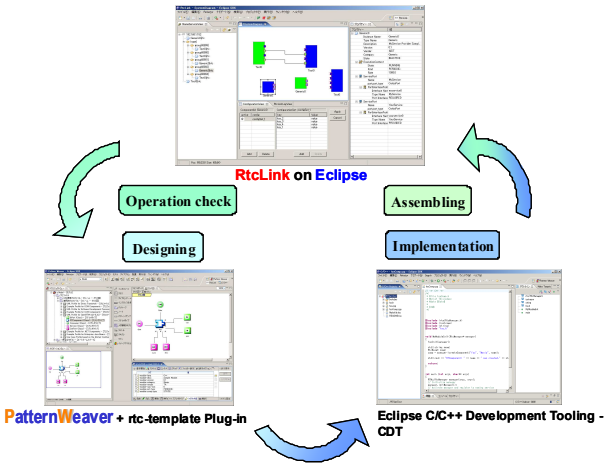


Fig.4 Relations among tools

4. 2. RtcLink

RtcLink は，RT コンポーネントを組み合わせてシステム構築，動作検証を行うための GUI ベースのツールである．ドラッグアンドドロップを中心とする簡単な操作で，使用する RT コンポーネントの配置，ポート間の接続，各 RT コンポーネントの動作状態の変更などを行うことができる．RtcLink の画面構成を Fig.5 に示す．

RtcLink の主な機能を以下に示す．

○コンポーネント・ツリー表示機能

ネーム・サーバへ接続し，登録されている RT コンポーネントをツリー形式で表示する機能．

OpenRTM-aist-0.4.0 は CORBA をベースとした仕様となっており，RT コンポーネントを管理・公開するためにネーミングサービスが利用されている．本機能は，各ネーム・サーバに登録されている RT コンポーネントを表示する機能である．複数のネーム・サーバへの接続が可能であり，各サーバ単位で登録内容をツリー形式表示する．また，一定周期毎にネーム・サーバから最新情報を取得し，ツリー表示の自動更新を行う．（ユーザによる手動更新も可能．自動更新周期は各ユーザが設定可能．）

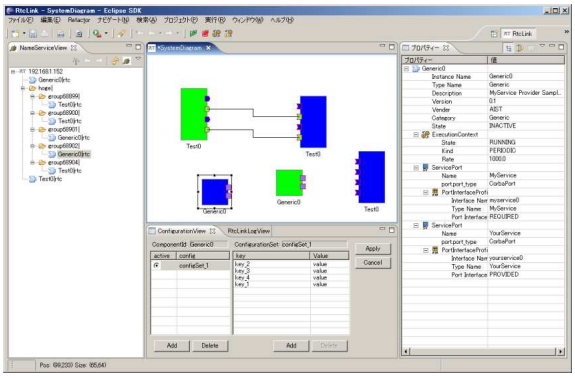


Fig.5 RtcLink

○コンポーネント組み立て機能

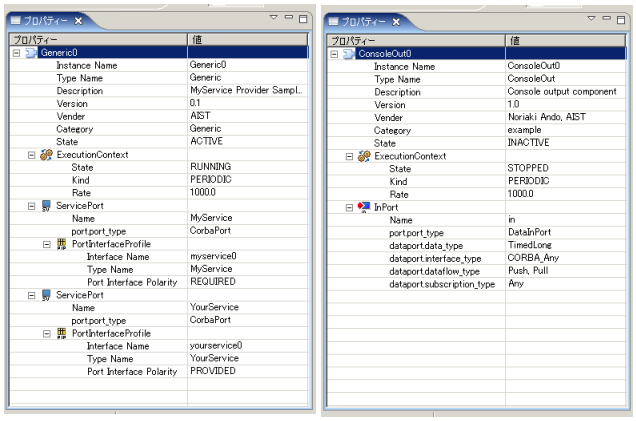
コンポーネント・ツリー内の RT コンポーネントを利用して，システムを構築する機能．

使用する RT コンポーネントは，コンポーネント・ツリーからドラッグアンドドロップで配置できる．また，配置した RT コンポーネントの移動，拡大・縮小，回転もマウス操作のみで実行できる．更に，配置した RT コンポーネントが持つポート間を接続することが可能で，接続時の接続可否判定や，接続情報（送受信するデータ型の指定，送受信の周期など）である ConnectorProfile の入力を行うことが可能である（ConnectorProfile は OpenRTM-aist-0.4.0 から新たに追加された機能）．

○コンポーネント・プロファイル表示機能

選択された RT コンポーネントに設定されたプロファイル情報を表示する機能．

各 RT コンポーネントは，設計（実装）時に固有のプロパティ情報が設定されている．本機能は各 RT コンポーネント固有のプロパティ情報を確認するための機能である．具体的には，RT コンポーネントのインスタンス名・型名・概要説明・状態など，さらに，実行主体である ExecutionContext の実行状態・実行種類・実行周期，Port の名称・データ型・インタフェース・データフロータイプ・サブスクリプションタイプ，Interface の型名・方向を確認することができる．



(a) Profile example of RT-Component having data port　(b) Profile example of RT-Component having service port

Fig.6 Profile View

○システムセーブ／ロード機能

構築したシステムの構成情報を XML 形式の外部ファイルに保存する機能．また，システム構成が保存された XML ファイルを読み込んで復元する機能．

外部ファイルには，システムを構成する RT コンポーネントの情報，各 RT コンポーネントに設定されたコンフィギュレーション情報，ポート間の接続関係および各接続の接続情報（ConnectorProfile），RT コンポーネントやポート間接続線の描画情報(位置，大きさなど)を保存することが可能である．

○コンポーネント・コンフィギュレーション機能

各 RT コンポーネントに設定されたコンフィギュレーション情報を表示，編集する機能．

RT コンポーネントのコンフィギュレーション機能は，OpenRTM-aist-0.4.0 から新たに追加された機能であり（SDO

で規定されている機能)，各 RT コンポーネント固有の設定情報（プロパティ）を ConfigurationSet という単位で複数保持することが可能となっている．

本機能は，RT コンポーネントへの ConfigurationSet の追加・編集・削除，各 ConfigurationSet 内のプロパティ情報の追加・編集・削除を行うことができる．また，アクティブな ConfigurationSet の切替も可能で，各 RT コンポーネントの設定を変更することもできる．

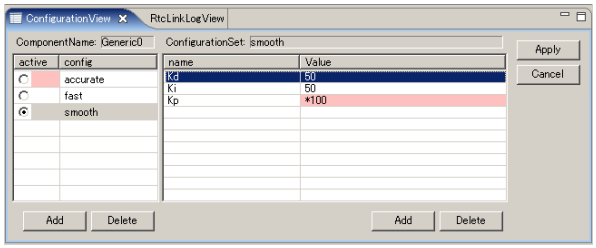


Fig.7 Configuration View

○コンポーネント動作制御機能

構築したシステム内の RT コンポーネントおよび実行主体である ExecutionContext の動作を制御する機能．

OpenRTM-aist-0.4.0 ではコンポーネントの状態遷移の変更に伴い，本機能には，従来の start/stop に対応する activate/deactivate に加えて，スレッドそのものの実行・停止を制御する start/stop オペレーションが追加された．また，選択した個々の RT コンポーネントの動作を変更する機能と，構築したシステム内に含まれる全 RT コンポーネントの動作を一斉に変更する機能も追加された．

各 RT コンポーネントは，内部状態に対応した表示色となっているため，動作変更結果（成功／失敗）は画面上で簡単に確認できる．

4. 3. rtc-template およびパターンウィーバー

パターンウィーバーは，著者ら(株式会社テクノロジックアート)が販売する UML モデリングツールである．全 13 種類のダイアグラムに対応した本格的 UML モデリングツールであり，Eclipse Plug-in として動作する[5]．今回の開発環境構築では，パターンウィーバーの UML プロファイル機能と，Plug-in 機能を利用し，RT コンポーネントのテンプレート・コード生成ツールを開発した．画面例を Fig.8 に示す．

UML プロファイルとは，特定のプラットフォームやドメイン固有の情報を追加するために，UML の既存要素(クラスやインタフェースなど)を拡張する仕組みである．追加情報はステレオタイプを用いて定義することができ，タグ付き値を利用した詳細情報の定義や，表示用アイコンの定義も行うことが可能である．パターンウィーバーでは，ユーザが独自に作成した UML プロファイルを使用する機能が存在するため，今回は，RT コンポーネントのテンプレート・コード生成のために必要な情報と，表示用アイコンを UML プロファイルとして新たに定義した．テンプレート・コード生成では，UML プロファイル定義に沿って設定された情報を基に，RT コンポーネントのプロファイル情報を埋め込んだクラスの雛形生成，各種 Port の生成と登録，新たに導入されたサービスポートの生成と登録を行う．また，タグ付けされたテキスト・ファイルを基に，テンプレート・コードを生成する仕組みを採用しているため，後から生成対象の言語を拡張することも可能である．

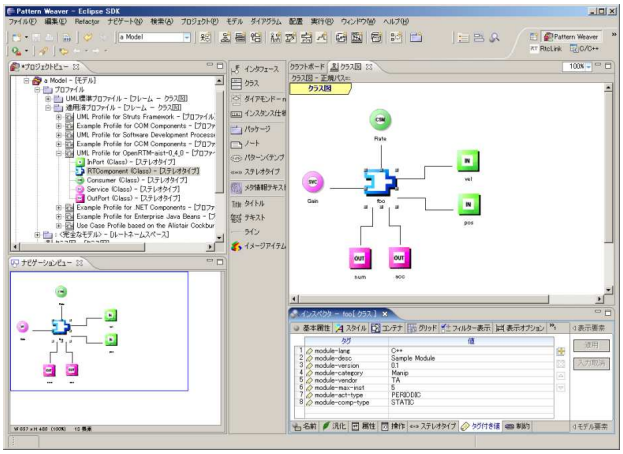


Fig.8 PatternWeaver + rtc-template Plug-in

5．まとめと今後の課題

今回提案した RT ミドルウェア用開発環境は，オープンソースの統合開発環境である Eclipse をベースとしており，開発に必要な様々なツールを Plug-in 形式で容易に追加できるため，多様なロボット開発環境を容易に構築することができる．

本稿の例では，RT コンポーネントの要件定義・設計・実装・テストという開発サイクルをサポートするために，以下のツールを用意した．

・設計ツール：rtc-template(新規作成)＋P.W.(既存 Plug-in)
・実装：CDT（既存 Plug-in）
・テスト：RtcLink（新規作成）

このうち新規に作成したのは rtc-template と RtcLink のみであり，その他は既存のものを利用することで容易に RT コンポーネントの開発環境を構築することができた．Eclipse の Plug-in の仕組みを利用することで，通常であれば多くの工数が必要な開発環境の構築を比較的容易に行うことができた．

しかし，今回構築した開発環境は未だ発展途上のものであり，多くの改良点が残されていると考えている．今後は，実際に本開発環境を使用したユーザからの要望をフィードバックしつつ，各ツールの更なる高機能化，新規ツールの開発を行っていきたい．特に，設計部分では，ベースとなっている UML モデリングツール　パターンウィーバーの機能を更に活用し，より高機能化を図る必要があると考えている．また，テスト部分に関しては，RT コンポーネント単体テストへの対応，動作情報や各種ログ情報の取得機能の追加が必要であると考える．

謝　辞

本研究の一部は，新エネルギー・産業技術総合開発機能(NEDO)次世代ロボット共通基盤開発プロジェクトの一環として実施されたことを記し，ここに感謝の意を表する．

文　献

[1] Noriaki ANDO, Takashi SUEHIRO, Kosei KITAGAKI, Tetsuo KOTOKU, Woo-Keun Yoon, "RT-Middleware: Distributed Component Middleware for RT (Robot Technology)", 2005 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS2005), pp.3555-3560, 2005.08, Edmonton, Canada
[2] The Robotic Technology Specification, http://www.omg.org/cgi-bin/doc?ptc/2006-11-07
[3] Platform Independent Model(PIM) and Platform Specific Model(PSM) for Super Distributed Object(SDO) Specification, http://www.omg.org/cgi-bin/doc?formal/2004-11-01
[4] Eclipse Fooundation のホームページ, http://www.eclipse.org/
[5] パターンウィーバーのホームページ,http://pw.tech-arts.co.jp/