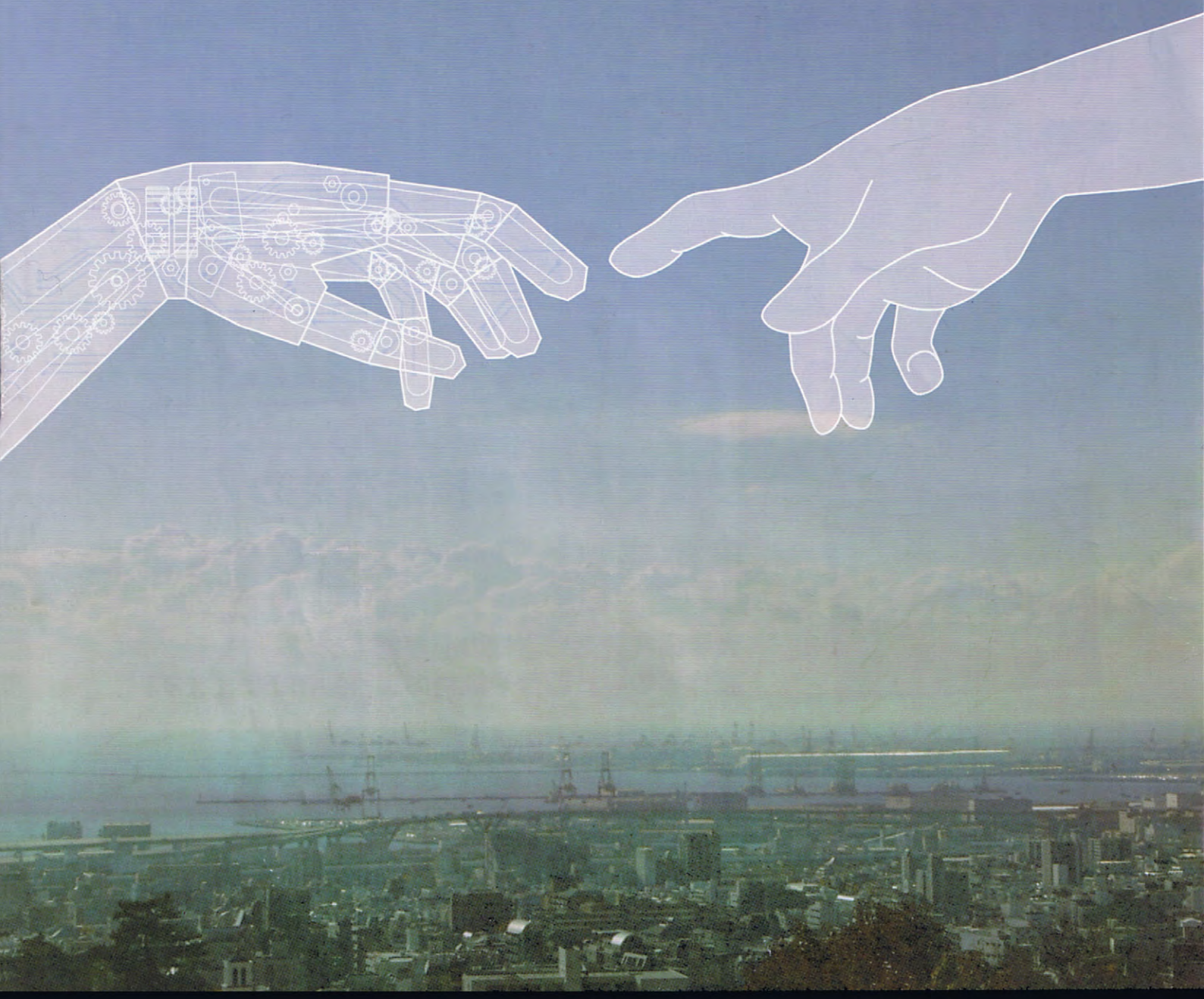
講演概要集

RSJ



第26回
日本ロボット学会学術講演会

THE 26TH ANNUAL CONFERENCE OF THE ROBOTICS SOCIETY OF JAPAN

[会期] 2008年9月9～11日 [会場] 神戸大学 [主催] 社団法人 日本ロボット学会 [共催] 神戸大学工学部・工学研究科

OpenRT Platform/RT ミドルウエア・ツールチェーンのためのモジュールおよびシステム仕様記述方式

○正 安藤慶昭 (産総研), 坂本武志 ((株) テクノロジックアート), 中本啓之 ((株) セック)

OpenRT Platform/Description Formats of Module and Systems Specifications for RT-Middleware Toolschain

*Noriaki ANDO (AIST), Takeshi SAKAMOTO (Technologic Arts Co. Ltd.),
Hiroyuki NAKAMOTO (SEC Co. Ltd.)

Abstract—In this paper, a specification-oriented RT system development is proposed. The RT system development consists of several process such as component development, debugging, system design and system development. OpenRT Platform, which is being developed as an integrated RT-system development platform, provides a tool-chain in order to support this development process. RT-Component Profile (RTC Profile) and RT-System Profile (RTS Profile) is proposed as description formats to be used by these tools. Two tools RTC builder and RT system editor are developed and cooperation between these tools by specification description formats is shown.

Key Words: Software platform, RT-Middleware, System Integration, Integrated Development Environment

1. はじめに

OpenRT Platform は RT ミドルウエア (RTM)[1] を基盤とし, モジュール化の基本単位として RT コンポーネント (RTC)[2] を採用した, RT システム開発を支援するコンポーネント指向ソフトウエアプラットフォームである.

RT ミドルウエアを利用した開発では, 複数の開発者により作成された多数の RT コンポーネントを, システムインテグレータが組み合わせることでシステムを構成する. RTC によるモジュール化の利点を生かすには, システムインテグレータは個々のコンポーネントの内部の詳細を知ることなく, コンポーネント自体が提供する機能のみに着目してシステムを設計・構築すべきである. このように, システムは多くの人の手により構築されるため, モジュールやシステムの仕様を明確に記述する方式を定める必要がある.

本稿では, 仕様に基づくロボット開発 (Specification Oriented RT-System Development) とそれを実現する仕様記述方式として, モジュールの仕様を記述する, RT-Component (RTC) Profile および, システムの仕様を記述する RT-System (RTS) Profile を提案する. これらの仕様記述方式は, RT システムの開発において使用される様々なツール (RT ミドルウエア・ツールチェーン) 間のデータ交換フォーマットとして利用される. また, これらの仕様記述方式を使用するツールとして RTC ビルダおよび RT システムエディタを開発した.

2. RTM ツールチェーン

RT ミドルウエアを用いてシステムを開発する際には, 図 1 に示すように, システム設計, コンポーネント開発, システム開発, システム運用といったプロセスを経る.

各プロセスにおける作業に対しては専用のツールが

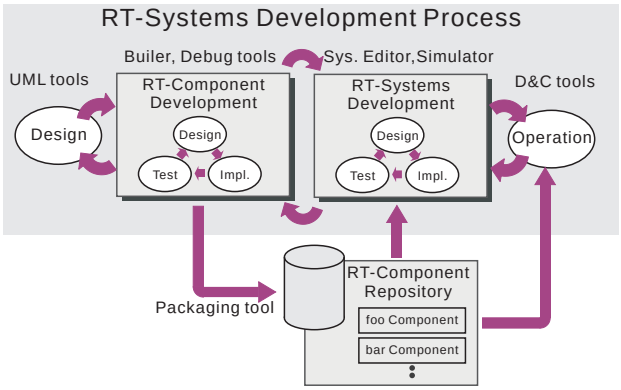


Fig.1 RT-Systems Development Process. The OpenRT Platform provides a development environment and tool-chain to support this development process.

提供され, 前段のツールが生成するファイルやコードを後段のツールが受け取り, 一連の RT システム開発作業が進められる. このようなツール群を RTM ツールチェーンと呼ぶ.

OpenRT Platform においては, 以下に述べるような開発プロセスを想定する.

システムを開発する場合, まず, システム設計に基づき必要なコンポーネントを揃える. 必要となるコンポーネントの大部分は, RT リポジトリと呼ばれる既存コンポーネントを蓄積するデータベースに存在し, これを利用する. リポジトリに存在しないコンポーネントは新たに作成する必要がある. 新たなコンポーネントを作成する際には, RT コンポーネントのフレームワークを利用するため, 大部分のコードを RTC ビルダにより自動的に生成することが可能である. 開発者は独自のロジックをフレームワークに埋め込む形でコンポーネン

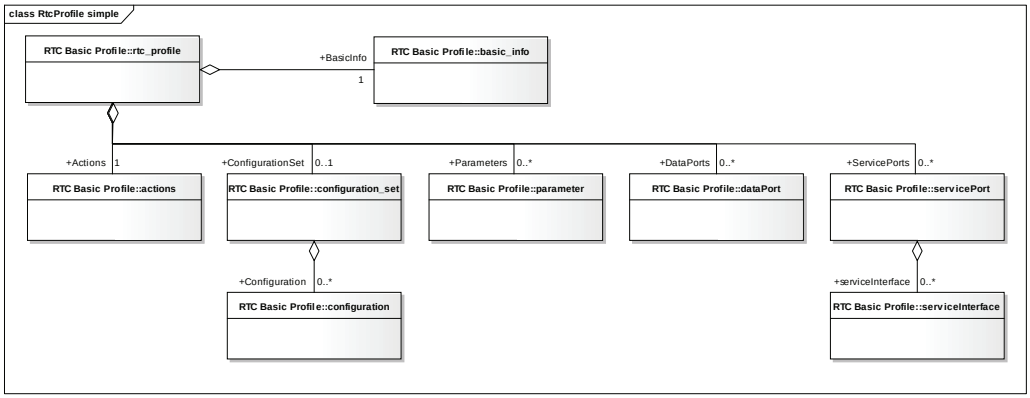


Fig.2 RTC Profile Class Diagram. RTC Profile defines data structure for a component description.

トを作成する．作成したコンポーネントをシステムに組み込む際には, 事前に単体でテストを行い, 仕様通りの動作を行うかどうかを, RTC デバッガによりテストを行う．こうして必要なコンポーネントが揃った後, コンポーネントを組み合わせて RT システムエディタによりシステムを構築する．コンポーネントの接続情報やパラメータ設定情報は XML 等の構造化されたフォーマットで記述され, 運用時にはこのファイルに基づきコンポーネントの起動, パラメータ設定, 接続を行い, システムが実際に動作する．

OpenRT Platform においては, こうした開発プロセスを支援する様々なツールが提供される．こうしたツール間の情報交換を, 共通化されたデータフォーマットに従い行うことで, ツール間の連携や新たなツールの導入が容易になる．本稿では, こうした共通データフォーマットのうち, RTC 仕様記述方式および RT システム仕様記述方式を提案する．

3. 仕様記述方式

3·1 MDA (Model Driven Architecture)

仕様記述方式を定めるにあたり, MDA を採用した．MDA (Model Driven Architecture) は OMG が提唱する, モデリング主導のシステムの開発, ライフサイクルの管理を実現するための参照アーキテクチャである [3]．中心となるモデルは, プラットフォームに非依存な PIM (Platform Independent Model) と プラットフォーム依存モデル PSM (Platform Specific Model) の 2 階層から構成される．

PIM はプラットフォームに依存しないシステムのモデルであり, UML により特定の言語や OS, ミドルウエアなどに依存せず, かつ曖昧さを排除して構築されたモデルを指す．PSM は PIM から生成される各プラットフォームに特化したモデルであり, 開発者はこのモデルを元に実装を行う．モデルを実装依存部分と非依存部分に分け定義することで, 技術トレンドが変化しても, PIM から PSM へのマッピングを定義し直すだけで, 新たな技術に対応できる利点がある．

本稿で提案する仕様記述方式は, PIM で定義されたデータモデルを, XML に対してマッピングし, XML スキーマを生成することでツール等で利用する．

3·2 RTC 仕様記述方式

RTC 仕様記述方式は, OMG RTC 標準で定められている RTC モデル [2] に基づき, さらにドキュメント記述および拡張記述が可能な表現力を強化したモデルとした．これを RTC Profile と呼ぶ．RTC Profile の利用方法としては以下のものが考えられる．

仕様記述フォーマットとして利用 システム設計時に, 機能要素をモジュール分割し各要素の詳細設計を行う際の, 各モジュールの記述方式として利用する．

コード生成での利用 RTC のモデルに必要な情報がすべて含まれているため, この情報を元に RTC のひな型コードを生成することができる．

システム構成での利用 RTC Profile に含まれるポートやコンフィギュレーション情報を元に, RTC 間の接続・設定・配置情報を仕様レベルで検討し, システムの設計を行うことができる．

検索情報としての利用 RTC リポジトリは, RTC をサーバ上に蓄積し, 必要な時に必要なコンポーネントを検索・ダウンロード・配置を行うための機能を提供するサーバである．RTC Profile に含まれる情報を利用して, 適切な RTC を検索することができる．

既存仕様の再利用 すでに存在するコンポーネントと類似のコンポーネントを作成する際には, 既存の RTC Profile を参照することで, より再利用性の高いコンポーネントを作成することができる．

ドキュメントとしての利用 利用者が RTC の内部を詳細に調べなくても利用できるだけの十分な情報を提供することで, モジュールの再利用性が向上する．

RTC Profile は RTC Basic Profile, RTC Document Profile, RTC Extended Profile の 3 つのパッケージから構成される．

図 2 に, RTC Basic Profile のデータモデルを示す．

3·2.1 RTC Basic Profile

RTC 基本プロファイルは RTC の基本メタ情報を含むプロファイルである． RTC Basic Profile には, 図 2 に示すように, `rtc_profile` を root ノードとして, `basic_info`, `actions`, `configuration_set`,

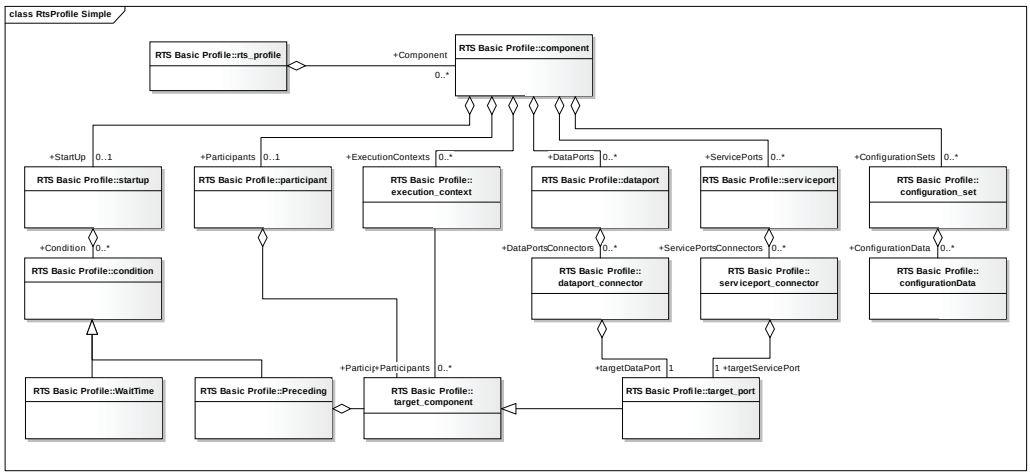


Fig.3 RTS Profile Class Diagram. RTS Profile defines data structure for a system description.

dataport, serviceport, parameter 等の要素から構成される.

一例として, 表 1 に basic_info の一部の属性および意味を示す. basic_info には RTC が持つ基本的なプロファイル情報が保持されている. これらの属性は主に OMG RTC Specification の ComponentProfile で定義されている値を含む. 他の要素も同様に属性を持つが, 紙面の都合上ここでは割愛する.

Table 1 RTC Basic Profile::basic_info

要素名	概要
name	RTC の名称
componentType	RTC の型名
category	RTC のカテゴリ
description	RTC の概要情報
executionRate	RTC のアクションの実行周期
executionType	RTC のアクションの実行タイプ
vendor	RTC の作成ベンダ名
version	RTC のバージョン番号

3·2.2 RTC Document Profile

モジュールの再利用性を向上させるには, インターフェースの適切な定義と, 十分なドキュメントを提供することが重要である.

RTC Document Profile では, RTC Basic Profile には含まれない, セマンティックなドキュメント情報の記述を支援する. たとえば doc_baisc::algorithm という要素には, そのコンポーネントが提供する機能やそれを実現するアルゴリズムを記述することにより, 利用者に対して RTC の動作に関する作成者の意図を伝える.

RTC Document Profile を含む RTC 仕様の XML ファイルを, 適当なテキストプロセッサに通すことで, 人間が読みやすい形式に変換することも可能である.

3·2.3 RTC Extended Profile

RTC Extended Profile は RTC に関する付加情報を記述する. RTC のバージョンアップを行う際に, 各バージョンに関する説明を記述する VersionUpLog, Configuration の付加情報に関するメタ情報を記述する Configuration_ext::Properties, などが含まれている.

3·3 RTS 仕様記述方式

RTS 仕様記述方式は, OMG RTC 標準で定められている RTC モデル [2] に基づき, RTC の組合せによりシステムを記述するモデルとした. これを RTS Profile と呼ぶ. RTS Profile データは以下の場面での利用が想定される.

システム仕様記述フォーマットとして利用 RTC で構成されるシステムにおいては, システム構成情報は, コンポーネントの配置情報・接続情報・コンフィギュレーション情報等から構成される. このシステム構成情報を記述するフォーマットとして RTS Profile を利用することができる.

システム検証での利用 RTS Profile データに基づき, シミュレータでシステムを構成し検証したり, 実際のシステムを構成し検証することができる.

システム運用時に利用 RTS Profile データに基づき, 実際のシステム運用時に RTC の起動・接続・設定 (Deployment and Configuration: D&C) を行うために利用することができる.

既存仕様の再利用 すでに存在するシステムと類似のシステムを作成する際には, 既存の RTS Profile を参照することで, システム開発のコストを低減することができる.

RTS Profile は RTS Basic Profile, RTS Extended Profile の 2 つのパッケージから構成される.

図 3 に, RTS Basic Profile のデータモデルを示す.

3·3.1 RTS Basic Profile

RTS Basic Profile は, RT システムの基本メタ情報を含む. RTS Basic Profile には, 図 3 に示すように, rts_profile ノードから始まり, component, execution_context, dataport_connector, serviceport_connector などの実行コンテキスト情報, 接続情報等から構成されるシステム情報が含まれる.

3·3.2 RTS Extended Profile

RTS 拡張プロファイルは, RT システムの本質的な機能にかかわらない付加的な情報を記述するために用意されているプロファイルである.

4. ツール

上述した RTC Profile および RTS Profile を生成または利用するツールとして, RTC ビルダおよび RT システムエディタを Eclipse plug-in として開発した.

Eclipse は Eclipse Foundation が開発するオープンソースの Java 言語およびその他言語のための統合開発環境であり, 同時に開発環境のためのプラットフォームでもある [4].

4·1 RTC ビルダ

図 1 の開発フローの中において, RTC を作成するツールとして RTC ビルダを開発した. RTC ビルダは, RTC の仕様を入力することで, RTC Profile XML ファイルを生成するとともに, C++, Java, Python といった言語用のひな型コードを作成するツールである.

図 4 に RTC ビルダの編集画面を示す. 中央のエディタ画面に RTC の仕様を入力する. 入力項目としては, 基本メタ情報, データポート, サービスポート, コンフィギュレーション, 言語・環境, ドキュメント等の情報があり, これらの項目を順に入力することで, RTC のひな型コード生成を行うことができる.

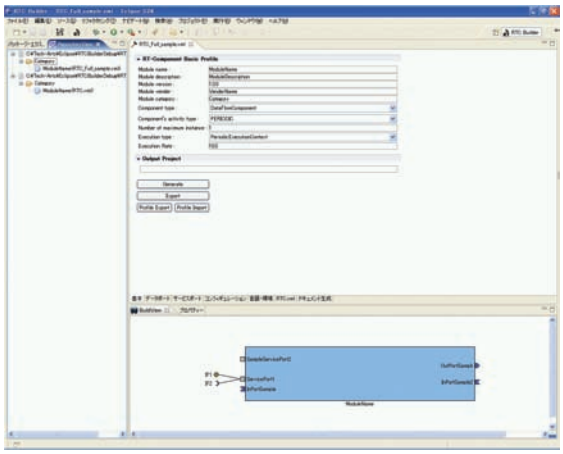


Fig.4 RTC Builder. RTC Builder is an RTC development tool based on component specification.

仕様入力後, "Generate" ボタンを押すと, 選択した言語のひな型コードが生成される. Eclipse に当該言語の開発環境 plug-in がインストールされていれば, 同一の画面内で引き続き RTC のロジックの実装作業を行うことができる.

4·2 RT システムエディタ

図 1 の開発フロー中の, RT システムを開発するツールとして RT システムエディタを作成した. 図 5 に RT システムエディタ (オンラインエディタ) の編集画面を示す. RT システムエディタは, RT ビルダで作成された RTC Profile を読み込みオフラインでシステム設計を行うオフラインエディタと, 実際に動作している RTC を接続・制御しシステムの開発・検証を行うオンラインエディタから構成される. 中央のエディタ画面に, RTC を配置し, ポートの接続, 設定等を行うことでシステムを構築する. いずれのツールも, 作成したシステムを RTS Profile XML ファイルとして保存, ロード, 再構成を行うことができる.

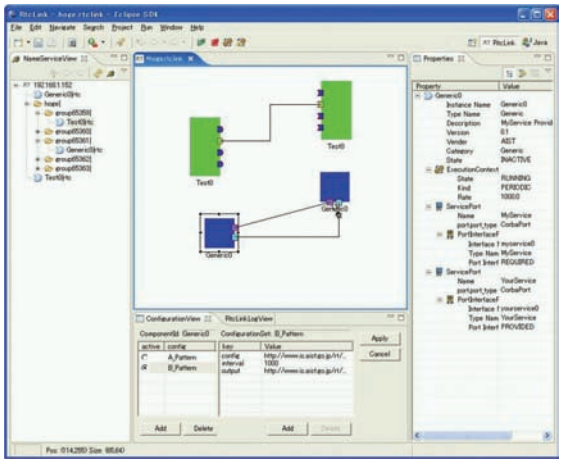


Fig.5 RT System Editor. RT System Editor is an RT system development tool.

5. おわりに

本稿では, 仕様に基づくロボットシステム開発 (Specification Oriented RT-System Development) および, これを実現する仕様記述方式と RT ミドルウエアツールチェーンを提案した. ツールチェーンの連携を効果的に行うためには, ツール間で共有されるデータフォーマットを定義することが重要である. 我々はモジュールの仕様記述フォーマットとして, RTC 仕様記述方式, システムの仕様記述フォーマットとして, RT システム仕様記述方式を MDA に基づき定義した. これらの仕様記述フォーマットを利用するツール, RTC ビルダおよび RT システムエディタを開発し, 実際に仕様記述フォーマットによりツールの連携が可能になることを示した. 今後, これらの仕様記述方式を利用する, RTC デバッガや RT リポジトリなどのツールを実現する予定である.

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) 次世代ロボット知能化技術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.

参考文献

[1] Noriaki ANDO, et.al, "RT-Middleware: Distributed Component Middleware for RT (Robot Technology)", IROS2005, pp.3555-3560, 2005

[2] Object Management Group, "Robotic Technology Component Specification Version 1.0", formal/2008-04-04

[3] "Model Driven Architecture: Applying MDA to Enterprise Computing", David S. Frankel, John Wiley & Sons, ISBN 0-471-31920-1

[4] Eclipse Fooundation, http://www.eclipse.org/