第24回
日本ロボット学会
学術講演会

The 24th Annual Conference of the Robotics Society of Japan

講演概要集

会期▶2006年9月14～16日

会場▶岡山大学津島キャンパス

主催（社）日本ロボット学会

RT ミドルウエアを用いた分散コンポーネントフレームワーク —RT ミドルウェアの機能拡張の検討—

○神徳徹雄，安藤慶昭，末廣尚士 (産総研)

Distributed Component Framework by using RT Middleware — Functional Enhancement of the RT-Component Model —

*Tetsuo Kotoku, Noriaki Ando, Takashi Suehiro (AIST)

Abstract— This paper reports the progress of the development of distributed component framework technology. For the efficient software development of robotic systems, we are planning to develop a common robot architecture based on the modularization of software. In the previous project, we have developed a framework of RT-Component which promotes application of Robot Technology (RT). In the project of "A Robot Simulator constructed on Distributed Object Modules", we are planning to make some functional enhancements of the RT-Component for the integrated robot simulator system.

Key Words: RT middleware, RT (Robot Technology), dynamics simulator, component model, standardization

1. はじめに

次世代ロボットのソフトウエア開発を効率化するために，ロボット実機コンポーネントとシミュレータコンポーネントとを容易に交換可能な分散コンポーネントフレームワークの実現が求められている．そこで，ロボット用ソフトウェアのモジュール化による開発効率向上を目指した RT ミドルウエアプロジェクト [1] で開発された RT ミドルウエアのコンセプト検証用プロトタイプをベースにして，自由に拡張可能な統合ロボットシミュレータを構築するために必要な機能を追加することで，分散シミュレータに対応する分散コンポーネントフレームワークの実現を目指している．

現在，国際的なソフトウエア標準化団体である OMG において，ロボット用のコンポーネントモデルの標準化を進めており [2, 3]，将来的にその標準仕様に準拠させる予定である．

本稿では，この分散コンポーネントフレームワークの開発状況について報告する．

2. コンポーネントフレームワークの仕様

ソフトウエアシミュレーションおよび実機のハードウエア研究開発をコンカレントに行うことができる統合シミュレーションシステムを， RT ミドルウエアにより構築するために必要とされる機能を検討し，開発の自由度，再利用性，インテグレーションおよび実機との連携などの視点からモジュール化されたソフトウエアに求められるインタフェースを整理した．この検討に基づき Fig.1 のように，従来の RT コンポーネントのモデルに拡張性を高めるサービスとそのコンシューマのインタフェースおよびコンフィギュレーションインタフェースを追加したモデルを新たに提案した [4].

今後開発すべき分散シミュレータのためのコンポーネントフレームワーク仕様をまとめたものを以下に示す．

(開発自由度) RT コンポーネントは，開発者が自由にインタフェースを定義し機能を外部に提供する RTComponent Service フレームワークを提供する．

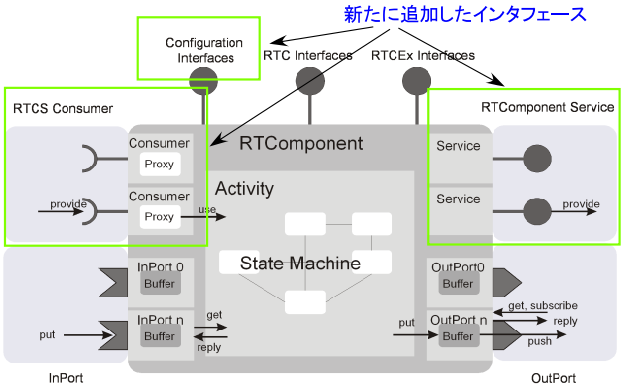

Fig.1 モデルに新たに追加したインタフェース

(再利用性) RT コンポーネントは，プログラムを再コンパイルすることなしに設定を外部から変更できる Configuration インタフェースを実装する．

(インテグレーション) シミュレータを構成する RT コンポーネントは，グラフィカルユーザインタフェース，モデルパーザ，干渉チェッカ，動力学シミュレータ，センサーシミュレータ，コントローラを構成コンポーネントとする．これらのコンポーネントはそれぞれ 1 つあるいは複数の RTComponent Service として実装する．

(実機との連携) シミュレータと実機がコンポーネントとして交換可能であり，相互にシームレスに利用可能な枠組みを RT コンポーネントおよび RT コンポーネントマネージャに実装する．

(稼働環境) Linux および Windows の両プラットフォームで利用可能な RT コンポーネントフレームワークを実装する．

3. RT ミドルウエアとしての実装

上記の仕様に基づいて，RT ミドルウエアとしての実装を開始している．具体的には RTComponent Service および Service Consumer フレームワークのためのインタフェース設計および IDL の記述を検討し，一部について試験的に実装を進めている．また，Configuration インタフェースの設計および IDL の記述が完了した．さらに，Windows および Linux の両プラットフォームで利用可能な RT コンポーネントフレームワークおよび相互運用可能性に関し調査し，C++/Python および Windows/Linux の相互運用性を確認したところである．

技術検証のために実装した RT コンポーネントは以下のとおりである．

3·1 RTComponent

RT コンポーネントの本体であり，コンポーネントの単位としての基本オブジェクトである．RT コンポーネントは，Activity，状態遷移，InPort/OutPort，コマンドインターフェース群などを持つ．

3·2 Activity と状態遷移

様々なコンポーネントデベロッパによって作成された多数のコンポーネントを統一的に扱うには，コンポーネントのライフサイクルにおいて共通化された状態を持つことが望ましい．そのため，RT コンポーネントは Activity と呼ばれる状態遷移を有する処理単位を持つ．Activity は，コンポーネント毎に固有なコアロジックを処理する部分である．RT コンポーネントは唯一の Activity を持ち，外部または内部のイベントに応じてその状態を遷移させる．Activity は次に述べる InPort/OutPort とは独立した処理単位が割り当てられ，原則として入出力とは非同期に処理を行う．RT コンポーネントの Activity は 11 の内部状態 (BORN, INITIALIZE, READY, STARTING, ACTIVE, STOPPING, ABORTING, ERROR, EXITING, FATALERROR, UNKNOWN) を持ち，それぞれ行うべき処理を定める．

3·3 InPort/OutPort

他の RT コンポーネントからの出力を受け取り処理する部分を入力ポート (InPort) と呼ぶ．データのストリームを受け取る必要がある RT コンポーネントはこの InPort を持つ．コアロジックは連続的に送られてくるデータに対して処理を行うため，基本的に周期動作を行うコンポーネントのみがこの InPort を持つ．また，他の RT コンポーネントへ処理結果のデータストリームを渡す部分を出力ポート (OutPort) と呼ぶ．受け取る側のコンポーネントから値を取得する pull 型のデータ出力と，subscribe することにより受け取り側へ能動的にデータを送る push 型の動作がある．原則として InPort/OutPort は Activity の処理とは非同期に動作することを前提としており，Pub-lisher/Subscriber モデルを採用している．通常の分散オブジェクトでは，データのやり取りは基本的にインタフェースのオペレーション呼び出しを介して行われる．これに対して，ロボットの制御における入出力データストリームは，データ型が重要な意味を持つ．RT コンポーネントはこういったデータ交換のために，データ型が同じであるポート同士を接続可能にするデータ型で区別される共通の入出力ポートオブジェクトとして InPort/OutPort を定義する．

3·4 RTC Service と Consumer

コンポーネントは，内部の振る舞いの詳細をあらかじめ定義されたインタフェースから変更できる必要がある．コンポーネントは原則としてバイナリモジュールとしての再利用性が求められ，再コンパイルすることなしに様々なシステムに利用できなければならない．RT コンポーネントにおいては，これを RT コンポーネント･サービス (RTCService) とよび，コンポーネント開発者が任意にインターフェースセットを追加できる．例えば，ロボットマニピュレータをコンポーネント化する際に，制御モード，制御パラメータ，座標系を指定したり，ヤコビやマニピュレータ固有のステータスを取得する必要がある．こうした，コンポーネントのカテゴリ別に特定のインターフェースセットを通して各種サービスを提供することを目的としている．また，あるコンポーネントが他のコンポーネントのサービスを利用して動作する場合，そのサービスへの参照を保持する必要がある．その参照を取得する方法はいくつか考えられるが，参照を外部から与える一つの方法として RTCServiceConsumer を定義している．RTCServiceConsumer はそのコンポーネントが依存するサービスへの参照のみを保持し，それはコンポーネント内のコアロジック内で利用される．こうする事で，コンポーネント同士のサービスの依存関係を実行時に柔軟に構成することが出来る．

4. おわりに

プロジェクトの初年度では，拡張可能な統合ロボットシミュレータを構築するために，統合シミュレーションシステムの実現に求められる RT ミドルウエアの拡張機能の機能設計を進めてきた．具体的には，開発者が自由にインタフェースを定義して機能を外部に提供するために，RT ミドルウエアの仕様として RTComponentService を追加することを検討した．

今後，ここで検討した基本仕様を実現するシミュレータ用の機能拡張版 RT ミドルウエアの実装を進め，そのフレームワークのもとでシミュレータ用の標準インタフェース仕様の策定を行う予定である．

謝辞

本研究は，総合科学技術会議科学技術連携施策群の効果的・効率的な推進事業の一環として，(独）科学技術振興機構からの委託により実施中のものである．

参考文献

[1] RT ミドルウエアプロジェクトのホームページ: http://www.is.aist.go.jp/rt/
[2] 神徳徹雄，水川真: “RT ミドルウエア標準化活動への誘い—OMG タンパ会議の報告”, JSME ロボティクス・メカトロニクス講演会 2006, 1P1-C22, 2006.
[3] 安藤慶昭，神徳徹雄，ルメアオリビエ，北垣高成，末廣尚士: “SDO(Super Distributed Object) に基づいた RT-Component オブジェクトモデル”, JSME ロボティクス・メカトロニクス講演会 2006, 1P1-C23, 2006.
[4] 安藤慶昭，末廣尚士，ルメア オリビエ，神徳徹雄，北垣高成: “RT ミドルウエアの PIM, PSM および実装—RT ミドルウエアの標準化に向けて—”, 第 11 回ロボティクスシンポジア論文集，pp.432-437, 2006.