

RSJ

RSJ 2003
THE ROBOTICS SOCIETY OF JAPAN

日本ロボット学会第21回学術講演会

講演概要集

2003/9.20-22

会場 | 東京工業大学



主催 (社)日本ロボット学会

RT 要素のモジュール化に関する検討

－ RT ミドルウェアの基本機能に関する研究開発 (その 1) －

末廣尚士 北垣高成 神徳徹雄 尹祐根 安藤慶昭
(独) 産業技術総合研究所 知能システム研究部門

Designing a Framework of RT Software Component

*Takashi SUEHIRO, Kosei KITAGAKI, Tetsuo KOTOKU,
Woo-keun YOON and Noriaki ANDO

National Institute of Advanced Industrial Science and Technology(AIST)

Abstract—

A new project of the METI was begun for three years from 2002 fiscal year. The purpose is a basic research and development which enables us to construct a robotic system with a new function easily by modularizing elements as software components and combining them freely. This paper discusses how to modularize elements of robot technology, a necessary function of modules, the structure of modules, use of a distributed object system, etc., and proposes "RT software component" as a basic framework of the modularization of robotic functional elements.

Key Words: RT(Robot Technology), software component, Robot system

1. はじめに

経済産業省のプロジェクト「ロボット機能発現のために必要な要素技術開発」が平成 14 年度より 3 年計画で開始された^{1, 2, 3)}。

本プロジェクトの目的は、ロボットシステムを構成する要素を、ソフトウェア的にモジュール化し、それを部品として自由に組み合わせることにより、新しい機能を持ったロボットシステムを容易に構築できるようにするための基礎技術を確立することにある。

このようにして、新しい機能を持ったロボットシステムを自由に作れるようになれば、現在は対応しきれないユーザごとの個別にニーズに答えることができ、従来は、主に製造業分野に限られていたロボット産業が非製造業分野へと拡大して行くことが期待されている⁴⁾。

産総研では、「RT ミドルウェアの基本機能に関する研究開発」という位置づけでこのようなロボット技術 (Robot Technology, RT¹⁾) のモジュール化のフレームワークの確立と、それを支える RT ミドルウェアの基本部分の開発を行う。

本研究開発では、分散オブジェクト技術 (CORBA) を用いて、ロボットの機能要素をモジュール化し、ユーザ幅広いニーズに合わせた新しいロボットシステムを容易に構築できる枠組を提供しようとしている。しかし、ここで注意しなくてはならないことは「ロボット

の機能要素のモジュール化」=「ロボットの機能要素の分散オブジェクト化」ではないということである。

本稿では、これらの違いを明確化し、モジュール化に必要な機能をどのような形で実現するかという、RT モジュールの基本形の概念設計を行った。

具体的には、ロボットの機能要素のモジュール化の考え方として「RT ソフトウェアコンポーネント」を提案し、必要な機能、構造およびその分散オブジェクトを用いた実現について検討を行った。

2. RT 要素のモジュール化に対する要請に関する検討

RT 要素のモジュール化に対する要請に関しては文献^{1, 2, 3)}において検討、提案が行なわれている。

このうち開発のベースに CORBA を採用したことにより他様なプラットフォームへの対応やネットワーク透過性などの分散オブジェクト環境は既に実現されていることになる。

したがって本研究開発の重点は RT システムの特有な機能の実現であり、以下のようなものは分散オブジェクトミドルウェアを採用したというだけではすぐには解決できないものとなっている。

- パッシブなモジュールだけではなく、アクティブなモジュールを実現する機能
いわゆるオブジェクトというのは、コマンドとして送られて来たメッセージに対して応答を返すというパッシブな動作を基本としている。これに対してロボットの要素はサーボなど、それ自身のタスクを持つだけでなく、必要なデータを自ら収集したり、またイベントの発生などの通知を行った

¹ここでいう RT(Robot Technology) とは、「ロボット技術を活用した、実世界に働きかける機能を持つ知能化システム」に関する技術の総称である。すなわち移動ロボットやヒューマノイドロボットといった単体のロボットだけではなく、センサ、アクチュエータを分散配置させ、生活支援や介護を行なうといった一見ロボットには見えないがロボットの技術を広く包含するものである⁴⁾。

りする必要がある。

- 通信障害に対する動作の安全性を確保するための機能

モータなどのアクチュエータを動作させる場合、動作途中で通信障害が発生した場合にも安全に動作させる工夫が必要となる。

- モジュールの周期動作やモジュール間の時刻同期などの時間管理機能

サーボ制御では安定した周期動作が必要である。また複数のモジュールを協調させるためには時刻同期が必要となる。センサデータの収集においても複数データ間の同時性、時間関係の把握は極めて重要なことである。

- モジュール間的高速で密な連携を実現する機能
RTシステムではサーボなど、家電ネットワークやプロセス制御などと比べるとモジュール間的高速で密な連携が要求される。

これら RT 要素特有の機能の実現を考慮しながら、RT 分野のアプリケーション全体に広く共通的に使われる機能を抽出して、それを整理、整備することにより RT 要素のモジュール化のフレームワークを確立する必要がある。

3. 分散オブジェクトによるモジュール化に関する検討

3.1 分散オブジェクトによるサ - ボの記述例

具体例として、センサ、モ - タ、サ - ボからなる制御システムを考える。センサオブジェクトの IDL 記述は、

```
interface Sensor {  
    long get();  
};
```

モ - タオブジェクトの IDL 記述は、

```
interface Motor {  
    void put(in long vel);  
};
```

などとなる。さてサ - ボはというと、ここで既に微妙に考えが分かれるところになる。ここでは、まずワンショットの制御則を実行する制御則オブジェクトを考える。その IDL 記述は、

```
interface Control {  
    long calc_vel(in long cur, in long ref);  
};
```

これらを使ってサ - ボの本体の部分コードを書くと、

```
for(;;) {  
    if(on_flag) {  
        cur = sen->get();  
        vel = con->calc_vel(cur, ref);  
        mot->put(vel);  
    }  
}
```

となる。また、この本体のコードをもつサ - ボ自身もオブジェクトとしてたとえば以下のように IDL 記述ができる。

```
interface Servo {  
    void on();  
    void off();  
    void set_sensor(in Sensor ss);  
    void set_motor(in Motor mm);  
    void set_control(in cc);  
};
```

これを図にすると Fig.1 のようになる。

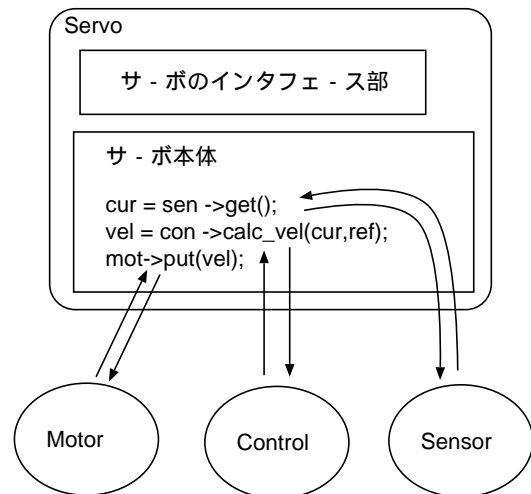


Fig.1 サーボのプログラム

この例から以下のようなことが示唆される。

- サ - ボ本体の働きは IDL では記述できない。これは当り前のようだが Sensor や Motor などの機能がほぼ IDL で記述され尽くされているように見えることは対照的である。さらに、このサーボ本体の働きは、サーボのモジュールにとって最も重要な機能である。この例ではこの機能がプログラムの形でしか表れていないが、モジュール化された「サーボ」の利用者にとっては、最低限そこで行われる他モジュールとの入出力は IDL のようなモジュールの仕様記述として表現されていることが必要となる。
- サ - ボ本体は、裏で常に活動し続けている。IDL は分散オブジェクトとしての静的なインタフェースを表現しているが、その裏ではサーボ本体が常に活動して内部の状態を変更している。したがって他のモジュールとデータやコマンドなどの情報をやり取りするときに、その時間的な整合性を保証することが重要となる。
- センサデータはセンサへの要求の返り値として取得されている。この例では、センサデータを取得するためには、まず、サーボからセンサにデータ要求メッセージを送る。そして、その返り値としてセンサデータを得る。これはセンサデータを得るのにネットワーク上の往復の時間が必要であることを意味する。

これは高速な制御を行う場合にネックになる。また、このような常に働いているループの場合には2倍の通信量になるだけだが、イベントを発生するセンサの場合、ネットワーク越しに、このようなポーリングを行うのはばかげたことといえる。

- モータへの出力要求も戻りを待つ。同様に、分散オブジェクトの呼び出しでは一般に oneway call は推奨されていないため、モータへの出力も呼び出しからの戻りを待つ必要がある。

3.2 RT 機能のモジュール化の課題

前項の考察から RT 機能のモジュール化に関して以下のような課題が抽出される。

- モジュール本体の働きの定式化
ロボットの機能要素のモジュール化に関してその本体の働きの定式化は重要である。その実装の問題を除いたとしても、インタフェース部以外で発生する外部との入出力の定式化および仕様記述は行わなくてはならない。
- モジュールの構造の定式化
ロボットの要素機能のモジュールの構造に関しては、最低限インタフェース部とアクティビティ部という構造があり、そのようなモジュールを簡単に作れるようなデザインパターン、支援ライブラリを作成する必要がある。
- ネットワークを意識した通信の効率化
分散オブジェクトの枠組の中では、各オブジェクトはどこにあっても(同じ1つのプログラム場合も含めて)扱いに差がないのが理想である。しかし、ロボットの場合、ハードウェアに直結し場所を動かさないモジュールの存在や、ネットワークの通信時間の影響が重要な場合がある。また、リアルタイム性が要求される場合にはいろいろな手段で通信を効率化する必要がある。

4. RT ソフトウェアコンポーネント

前節で説明したように分散オブジェクトとは IDL など記述された明示的なインタフェース (interface) を通してのみクライアントにサービスを行うサーバオブジェクトである。これはコマンドとして送られて来たメッセージに対して応答を返すというパッシブな動作の側面のみをサポートするものであり、そのオブジェクト内での処理については基本的には立ち入らない。

これに対してロボットの要素の多くはサーボなど、それ自身のタスクを持つだけでなく、必要なデータを自ら収集したり、またイベントの発生などの通知を行ったりなどの固有の処理 (アクティビティ) を持っている。その固有の処理の中では、自身のインタフェースで要求されるデータを随時書き換えたり、他の RT モジュールへクライアントとして要求を出したり、などが行われることになる。

ロボットの要素を部品として再利用できる形でモジュール化するためには、上記の機能をひとまとまりとした単位で考える必要がある。これをオブジェクトと区別して RT ソフトウェアコンポーネントと呼びロボットの要素のモジュール化の単位とする。(簡単のため、以後、本稿では RT コンポーネントまたは単にコンポー

ネントと呼ぶ)。

4.1 RT コンポーネントの動作パターンによる分類

RT コンポーネントの機能の面から分類すると、典型的には前述のセンサ、モータ、コントロール、サーボなどの分類が考えられる。しかし、ここでは、この分類の詳細には踏み込まず、より一般的な RT コンポーネントの動作パターンから大きく Fig.2 の3つの分類を検討した。

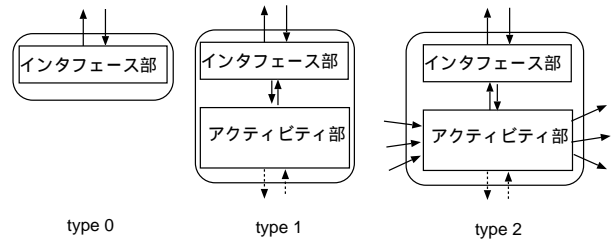


Fig.2 RT コンポーネントの典型的動作

- type 0
並列動作のアクティビティ部なし。単純な分散オブジェクトとして扱える。
- type 1
インタフェース部からのみ受動的に外部とのやりとりをする。固有ハードウェアとのインタフェースはあるとしても、アクティビティ部でのモジュール間通信はなくひたすら内部処理を行う。外部的には、普通の分散オブジェクトとして扱える。
- type 2
アクティビティ部から能動的に外部にリクエストを出す。アクティビティ部は分散オブジェクトとしてはクライアントになる。

4.2 RT コンポーネント間の通信についての検討

これと同時に RT コンポーネント間の通信についても検討を行った。RT コンポーネント間の通信は大きくインタフェース部を通しての通信とアクティビティ部で行う通信に分けられる。インタフェース部を通じた通信では、主に、インタフェース部への通常のメッセージやアクティビティ部へのコマンドなどがやり取りされ、アクティビティ部で行う通信では、主に、アクティビティ部からのデータ取得やアクティビティ部へのデータ書き込みなどアクティビティ部で必要とされるデータがやり取りされると考えられる。しかし、これについては完全にそのように切り分けられるのか、またはそのように切り分けたほうが良いのかということなど、引き続き検討が必要な項目が数多くあることが分かった。

たとえば、センサなどは type0 のコンポーネントとして扱うことができるが、その場合センサデータのやり取りはインタフェース部を通して行うことになる。しかし、もしセンサが push 型のデータ発信を行う必要がある場合には、アクティビティ部を通さなくてはならなくなる。

またこれらの通信に関しては以下のような問題点、課題が抽出された。

- アクティビティ部がデータを書き換えるため、データのコンシステンシーを保つ必要がある。
- データ取得を通常の分散オブジェクトの呼び出しとして定期的に行う場合、通信遅延が大きくなる。
- データ送信を通常の分散オブジェクトの呼び出しとして定期的に行う場合、不要な返答を待つ必要が生じる。
- 多数のデータ入出力要求を逐次的に行うと各々の遅延が加算される。

5. システム統合法の検討

RT コンポーネントを組み合わせてロボットシステムを構築する際の統合手法について検討を行った。3節でも扱ったサーボの例を考えると、統合の形態としては大きく以下の3種類がある (Fig.3(a),(b),(c)) ことが分かった。

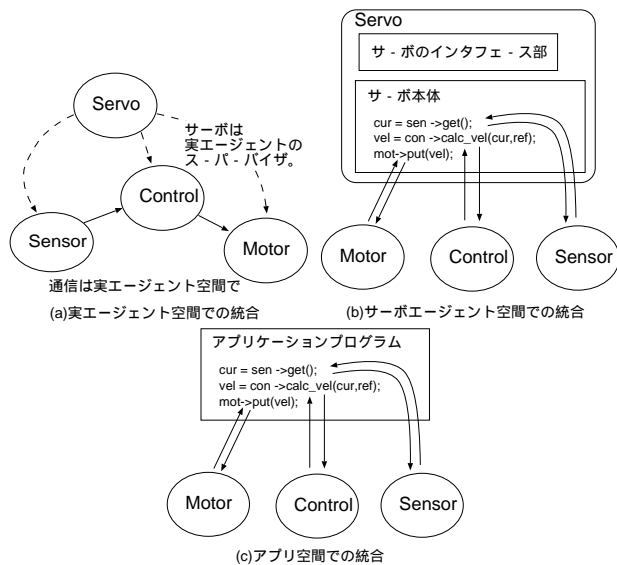


Fig.3 統合パターンの分類

(a) 実コンポーネント空間での統合 (Fig.3(a))

これはセンサ、コントロール、モータのコンポーネントが直接通信することによってサーボ制御を実現するものである。サーボコンポーネントの役割は上記3つのコンポーネント間の接続や制御動作の起動、停止、状態監視などになる。この統合手法では、アクティビティ部での入出力、push形の非同期通信が必須となると考えられる。この手法の欠点は、サーボコンポーネントはサーボ制御に必要な構造を設定するだけなので、そのプログラム上の記述からでは実コンポーネント空間で行われるサーボ制御の実体がどのようなものかが分かりにくくなる可能性が大きい。

(b) サーボコンポーネント空間での統合 (Fig.3(b))

これは、前述の (a) と後述の (c) との中間的な形態である。センサ、コントロール、モータの各コンポーネントは直接は通信を行わない。サーボコンポーネントがプロキシを通して各エージェントへ要求を出し、仲介することによりサーボ制御を実現する。後述の (c) との違いは、そのようなサーボプログラムも再利用可能なコンポーネントとしてモジュール化するという点で

ある。この統合手法では、アクティビティ部で入出力を行うサーボのような統合的なエージェントと必ずしもそれが不要でないセンサ、コントロール、モータなどの基本的なエージェントとを分けて考えることができる。ただし、イベント通知や通信遅延の問題を考えた場合には、基本的なコンポーネントにも非同期通信の機能が必要となる。

(c) アプリケーション空間での統合 (Fig.3(c))

この統合手法では、アクティビティ部を持つようなものは原則としてモジュール化の枠組みに含めない。すなわち RT のモジュールは複雑なアクティビティを持つ RT コンポーネントではなく分散オブジェクトとして扱えるものだけと割り切って考えることになる。具体的にはセンサ、コントロール、モータといった分散オブジェクトとして切り出せるものを基本的なモジュールとして、他のソフトウェアはすべてそれを利用したアプリケーションとして考えることになる。この方法の利点は、ネットワークやコンポーネントの接続といったものを意識しないで、アプリケーションプログラムを書くことができ、実際の振る舞いも完全にそのプログラムと同期して行われるという点にある。一方で同期型通信のみでは通信の遅延やトラフィック量が大きな問題になる場合もあるし、中途半端に非同期通信を導入すると先に述べた利点が失われる可能性がある。

6. おわりに

本稿ではセンサ、モータ、コントローラ、サーボシステムなどの典型的な RT 基本要素に必要な機能、課題、インタフェース仕様などの検討を行い、RT 基本要素を分散オブジェクトシステムを利用して RT コンポーネントという形でモジュール化することを提案し、また RT コンポーネントの統合の形態の分類および RT コンポーネントとしてモジュール化するために必要な機能の抽出を行った。

今後は、マニピュレータの力制御、遠隔制御、ビジュアルフィードバック制御などを具体例として、必要な RT コンポーネントの実装を行う。またその過程で、RT コンポーネントの構造の確立、そのプログラミングに必要な機能のライブラリ化、仕様化を行い、RT ミドルウェアの研究開発を進める予定である。

本研究は、NEDO「ロボットの開発基盤となるソフトウェア上の基盤整備」の依頼を受けて行ったものである。

参考文献

- 1) 北垣、末廣、神徳、平井、谷江: RT ミドルウェア技術基盤の研究開発について -ロボット機能発現のために必要な要素技術開発-、ロボティックシンポジウム予稿集、pp.487-492, 2003.
- 2) 北垣、末廣、神徳、平井、谷江: RT ミドルウェア技術基盤の確立に向けて、ロボメカ、2003.
- 3) 平井、末廣、北垣、神徳、谷江: ロボットシステムのソフトウェアに基づくモジュール化に関する一考察、SICE SI 部門講演論文集、vol.2, pp.53-54, 2002.
- 4) 21 世紀におけるロボット社会創造のための技術戦略調査報告書、(社)日本機械工業連合会、(社)日本ロボット工業会、2001.