2W−2

# 広域分散システム環境における ノンストップ・バージョンアップ手法

島田　優,　村田　悟,　齊藤雅彦,　横山孝典
(株)日立製作所　日立研究所

## 1．はじめに

分散処理システムが広域化、高信頼化するに従い、実行時環境の保守／管理といった問題は、従来の単一の大型計算機による集中処理に比べ、はるかに困難なものになる。我々は既に、広域分散システムに対して有効な多重世界モデルを提案している1)2)。本講では、この多重世界モデルに基づくシステムにおいて、全システムを停止させることなくバージョンアップを行う、ノンストップバージョンアップ実行時にシステムの正しい動作を保証するための手法について述べる。

## 2．多重世界モデル

我々が提唱している多重世界モデルにおいては、ユーザーから見たシステムは「オブジェクト」と「世界」から構成される。これらオブジェクトは互いにメッセージの交信を行うことにより処理を実行してゆく。また、世界はオブジェクトの実行環境または実行範囲を規定するものである。そしてオブジェクトは自己が属する世界内の他のオブジェクトのみ直接認識可能である。従って、あるオブジェクトが直接通信可能な他のオブジェクトが存在する範囲も、自己が属する世界内に限られる。このため、他の世界に属するオブジェクトとの間の通信は、そのオブジェクトが属している世界を抽象化しているオブジェクトと世界を介することによって間接的に行われることになる。

一方個々のオブジェクトに関しては、一つのオブジェクトを、ユーザが記述する通常のプログラム本来の処理を実行するオブジェクトレベルと、オブジェクトの状態／実行を管理するメタレベルに分離している。これによりシステム構成に依存する処理はメタレベルのオブジェクト（メタオブジェクト）により吸収され、オブジェクトレベルからは知る必要が無くなる。

オブジェクトが他のオブジェクト又は世界と通信する場合には、このオブジェクトが実際に通信を実行するのではなく、このオブジェクトのメタオブジェクトに対して通信を依頼することにより通信が実行される。従って通信は通常オブジェクトからメタオブジェクトに対するメタコールの形で起動される。オブジェクトから通信の依頼を受けたメタオブジェクトは、オブジェクトが指定する通信形態に従い通信を実行する。

## 3．バージョンアップ

多重世界における実行系のバージョンアップは、システム管理者がバージョンアップの対象となるオブジェクトに対して、バージョンアップ命令を通知することにより各オブジェクトのバージョンアップが開始される。なお分散システムの性質上、システムは広域に分散した計算機群から構築されていて、システムの保守管理のために全ての計算機を同時に停止させ、プログラムのバージョンを一斉に変更することが出来ないものとする。

分散処理システムの場合、一般に複数のオブジェクトが相互に通信を行うことにより、一つの作業を協調して行う。そこでシステムが正しく動作するためには、各々のオブジェクトが正しく動作することに加えて、処理を依頼されたオブジェクトが、依頼した側の要求通りに正しく動作する必要がある。

ここで言う「正しく動作する」とは、オブジェクトの動作が作成者の仕様と、又はユーザが意図した動作と同じ動作をすることを意味する。

互いに依存するオブジェクトが正しく動作するためには、まずオブジェクト間の通信インターフェースが一致している必要がある。

バージョンアップ時にオブジェクトが有していたそれまでの関数のインターフェースを、異なるインターフェースに変更する場合には、従来の関数名と異なる名前にした関数を新たに追加し、メタオブジェクトにより起動する関数をインターフェースにより選択することにより対応可能である。

そこで、以下各オブジェクトの通信インターフェースは常に一致するものと仮定する。

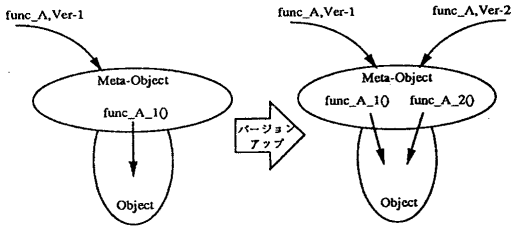


図1　新規インターフェースの追加

## 4．ノンストップバージョンアップの問題点

オブジェクトのインターフェースがバージョンアップの前後で同一ならば、バージョンアップ後のオブジェクトの動作の意味が、バージョンアップ前の動作の意味と異なっている場合でも、サーバ側のオブジェクトはバージョンに関わらず通信を受け取り実行してしまう。従って、異なるバージョンでは動作の意味が異なるオブジェクトの場合、バージョンアップ後のシステムは正しい動作が保証されず、意図しない動作を行う危険がある。

また、バージョンアップ完了後はシステムの正しさが保証されるような場合でも、バージョンアップが行われる際には、関連するオブジェクト全てが同時にバージョンアップされるわけではない。その為、過渡的にシステム内に複数のバージョンのプログラムが存在することがある。更に、バージョンアップ中にも関わらず、世界外から通信を送ってくるオブジェクトも有り得る。

この為、一連の処理の中で複数回依存オブジェクトを呼び出す時に、時間によって依存オブジェクトのバージョンが異なる場合が起こり得る。この時、次のようなケースが発生する。

(a)非同期変更：クライアント側が一連の処理を行うため複数回通信を行う場合、通信と通信の間に依存オブジェクトのバージョンが変化してしまうことが有る。

(b)間接変更：クライアント側が直接通信する依存オブジェクトと、依存オブジェクトが通信するオブジェクトが同じ機能を持つ場合、これらの依存オブジェクトのバージョンが異なる場合がある。

この様な時にシステムが正しく動作しているかどうか確認するため、オブジェクトのバージョンとその動作の意味を管理しておく必要がある。

---

A Study of Non-Stop Version up under Widely Distributed Computing Environment
Masaru SHIMADA, Satoru MURATA, Masahiko SAITOH, Takanori Yokoyama
Hitachi Research Laboratry, Hitachi, Ltd.

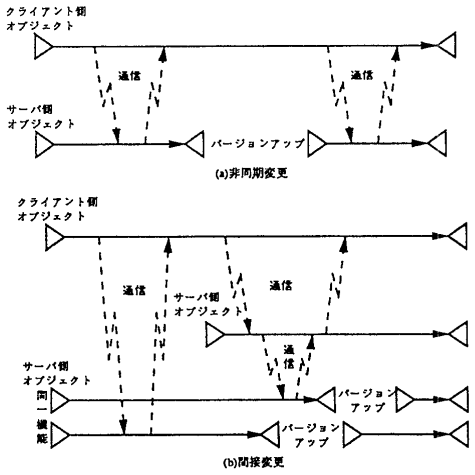


図2　バージョンアップ時の問題点

しかし、単純にバージョンアップ時にバージョン番号を変更し、通信時にクライアントとサーバのバージョン番号を単に比較するだけでは、一方のバージョン番号のみを変更すると通信が出来なくなるといった問題が発生する。従って、た管理方式が必要になる。

## 5．動的バージョン情報判定

以下では、バージョンアップしたオブジェクトを作成する時には、その時点までの他のオブジェクトとの関係が既に分かっているものと仮定する。

送信元オブジェクトの動作が正しいことは、依存オブジェクトに対し送信元オブジェクトが依頼してきた機能に関して満足しなければならない条件（要求バージョン情報）を満足する場合のみ、保証される。なおここで言うバージョン情報とは特定のオブジェクトのバージョンを表す情報ばかりではなく、バージョンの集合を列挙したり範囲として表したものである。

この情報を用いて以下の様な手順で通信を行うことにより、オブジェクトのバージョンと動作の意味の間により柔軟性を持たせる。

### 5．1　送信

メタオブジェクトはこの送信元オブジェクトにおける要求バージョン情報を通信のパケットにこれらの情報を付加して送信する。これらオブジェクト識別名と要求バージョン情報の対応は予めユーザーが記述しておくことになる。

そしてこの通信に対して依存オブジェクトでバージョン不一致が検出された場合、オブジェクトデータベースで要求バージョン情報が通信の要求バージョン情報を満足させる、他の依存オブジェクトを検索して、再びこの新しく検索したオブジェクトに送信する。

### 5．2　受信

次に、この通信を受信した依存オブジェクトのメタオブジェクトでは、受信した通信パケットから要求バージョン情報を取り出す。そしてメタオブジェクトは依存オブジェクトの自己バージョン情報が要求バージョン情報の条件を満足するかどうかを調べる。

この比較が満足される場合には送信元オブジェクトの要求に対する依存オブジェクトの動作の正しさが保証されるために、その通信を受理する。

しかし自己バージョン情報が要求バージョン情報の条件を満足しない場合には、依存オブジェクトの等価動作バージョン情報を調べ、この条件を満足させるかどうかを調べる。そして満足させる場合には、やはり送信元オブジェクトの要求に対する動作の正しさが保証されるため通信を受理する。しかし、等価動作バージョン情報が要求バージョン情報の条件を満足しない場合には、送信元オブジェクトにバージョン不一致のエラー発生を通知する。

この、依存オブジェクトがその動作と等価な動作を保証する他のオブジェクトのバージョン情報である等価動作バージョン情報は予めユーザーが記述しておく。

依存オブジェクトがこの受理した処理を終了して送信元オブジェクトに応答を返す際、メタオブジェクトは応答の通信パケットに自己バージョン情報を付加してから送信元オブジェクトに対して応答通信を返す。

### 5．3　返信

応答通信パケットを受け取った送信元オブジェクトのメタオブジェクトは、応答通信の通信パケットから依存オブジェクトの実行オブジェクトバージョン情報を取り出し、要求バージョン情報を更新する。更に必要に応じて世界に対しオブジェクトデータベースの要求バージョン情報の更新を要求する。

この様に、クライアントの要求バージョン情報と、サーバの等価動作バージョン情報に幅を持たせることにより、バージョンと動作の意味の関係をより柔軟に管理することが出来る。

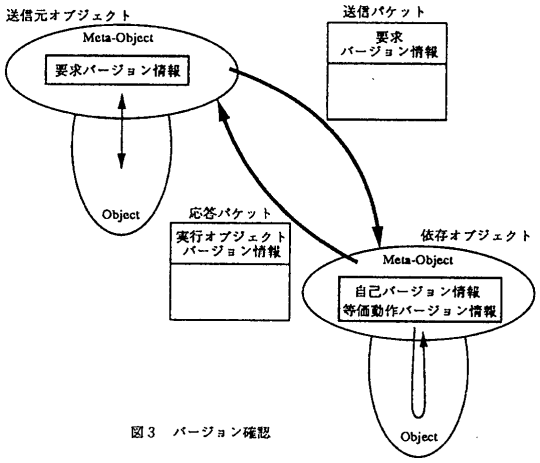


図3　バージョン確認

## 6．おわりに

本報告では、分散処理システムを構成する各オブジェクトについて、クライアント側オブジェクトが必要とするサーバ側オブジェクトの条件と、サーバ側オブジェクトが満足させるクライアント側オブジェクトの条件を予め宣言しておき、これらを実際の通信時にメタオブジェクトによって動的に更新させる方式を提案した。この方式により、システムの保守に伴う各プログラムが正しく、意図した通りに動作するための条件の変化に対応し、システム全体のの正しい動作を保証している。

[参考文献]
1）横山他：リフレクティブアーキテクチャに基づく分散処理環境（1）（2）（3），情報処理学会第４３回全国大会
2）齊藤他：広域分散システム環境の高信頼化に関する一手法，情報処理学会第４５回全国大会