

オブジェクト指向交換プログラムの分散環境での実行制御

田中 聡 久保田 稔 丸山 勝己

NTT 交換システム研究所

複数のモジュールからなる分散オブジェクト指向交換システムにおける、オブジェクト識別子の管理手法、メッセージ通信の手法、メッセージの転送制御について述べる。本手法では、オブジェクト識別子としてモジュールの識別子とモジュール内のローカルな識別子から構成される論理識別子を用い、分散を意識しないメッセージ通信を可能にしている。さらに、オブジェクトの移動、置き換え時に、論理識別子からオブジェクトの物理アドレスに変換する過程で、メッセージの転送制御を行う。これらにより、電話サービスを中断することなく、システム構成の変更、オブジェクトの移動、置き換えを可能にしている。

Execution Control for an Object-Oriented Program in a Distributed Switching System

Satoshi Tanaka, Minoru Kubota and Katsumi Maruyama

NTT Communication Switching Laboratories

3-9-11, Midori-Cho, Musasino-Shi, Tokyo, 180 Japan

This paper describes object identifier management, inter-object communication and message forwarding mechanism for an object-oriented program in a distributed switching system consisting of multiple hardware modules. Using an object identifier consisting of a module identifier and a local object identifier, a message for an object in a remote module can be handled in the same way as a message for a local object. When an object migrates to another module or replaced with a new one, messages sent to the old location can be forwarded to the original object. This allows continuous telecommunication services even when the system is reconfigured.

1 はじめに

ISDN、インテリジェントネットワークの新サービスユーザーが増加し、電話サービスへの要求はますます多様化している。このような要求に対応するため、

- (1) ニーズに応じた処理能力の向上
- (2) サービスや機能の追加、変更の容易性

が、交換システムに求められている。これらを実現するため、複数のモジュールから構成される分散交換システムに、オブジェクト指向に基づく交換プログラムを適用することを検討している [1]。この分散オブジェクト指向交換システムは、以下の条件を満たすことを狙いとしている。

- (1) 高い処理能力と実時間性
- (2) 高い信頼性
- (3) 種々の交換システムの構成に対する柔軟な対応
- (4) 容易なプログラムの追加、変更

これらを実現するためには、交換プログラムを構成するオブジェクトを柔軟に管理し、分散システム上での効率良いオブジェクト間メッセージ通信処理を実現する必要がある [2][3]。本稿では、オブジェクトの位置情報を隠蔽し、オブジェクトの移動、置き換えを容易とすることを狙いとした、オブジェクト識別子の管理手法、メッセージ通信の手法、オブジェクト移動・置換時のメッセージの制御手法について検討した結果について報告する。

2 分散オブジェクト指向交換システム

2.1 分散交換システムの構成

本稿で対象とする分散交換システムは、図1に示すように、特定の交換機の機能を実現する複数のモジュールがモジュール間結合機構により結合したシステムである。この分散交換システムをノードと呼び、これがネットワークで結合されて通信網を構成するものとする。本稿では、一つのノード内での分散処理方式を対象とする。

モジュールは、加入者回路、通話路、トランクなどの交換処理装置とそれを制御するプロセッサ及びメモリから構成される。ソフトウェアモジュールと区別するため、以後これをシステムモジュールと呼ぶ。

システムモジュールは特定の交換機能あるいは信号機能を実現し、それ自身で小型の交換機と考えることができる。システムモジュールの具体例としては、アナログ加入者線システムモジュール、ISDN 加

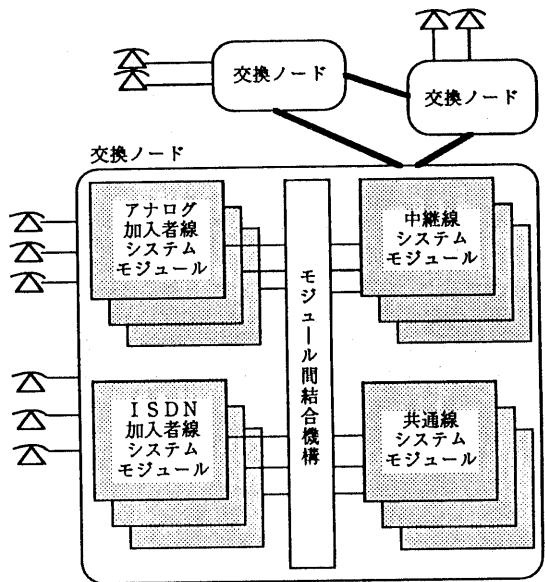


図1: 分散交換システム構成

入者線システムモジュール、中継線システムモジュール、共通線信号システムモジュールなどがある。モジュール間結合機構は、これらのシステムモジュール間を結び、情報や信号の転送を行う。

2.2 オブジェクト指向交換プログラムの構成

各システムモジュール上では、オブジェクト指向の考えを適用して設計された交換プログラムと、それらの実行を制御する実行制御システムが動作する。

この交換プログラムは、図2に示すようなオブジェクトから構成される。各種交換サービスを実現しているサービスシナリオオブジェクト、数字分析を行うサービス分析オブジェクト、呼処理を実行する呼処理オブジェクト、交換処理装置の制御を行うリソースオブジェクト(加入者線オブジェクト、通話路オブジェクト、トランクオブジェクトなど)である。

実際の呼処理は、呼ごとに生成される Caller オブジェクト(発側呼処理オブジェクト)、Callee オブジェクト(着側呼処理オブジェクト)によって進められる。これらのオブジェクトが並行に実行され、相互に情報を通信しながら、呼の接続処理、各種交換サービスを実行する(図3)。

このような交換プログラムでは、Caller クラス、Callee クラス、サービスシナリオクラスなど、静的なオブジェクトは設計時に決めることができ、また

システムモジュール

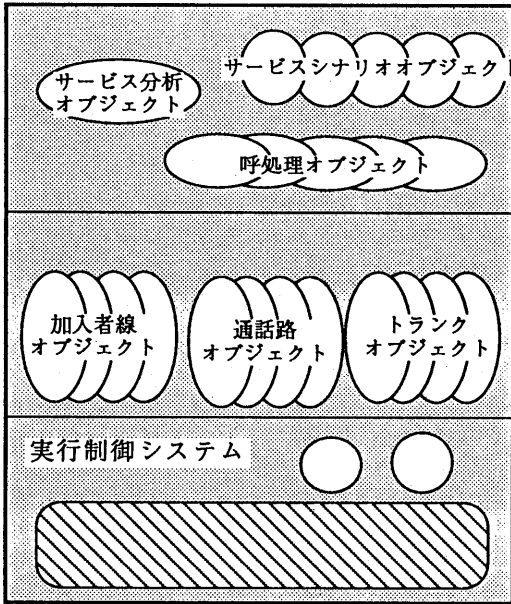


図 2: オブジェクト指向交換プログラムの構成

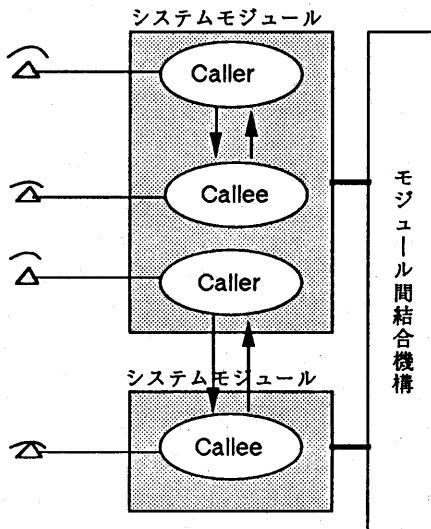


図 3: 呼処理の概要

数も多くはない。一方、呼処理を行う Caller オブジェクト、Callee オブジェクトは、それぞれのクラスのインスタンスとして実現されている。

3 オブジェクト識別子の管理手法

3.1 要求条件

本分散オブジェクト指向交換システムでは、呼ごとに生成されるオブジェクトは、局用交換システムの場合には、同時に数千個となる。また、呼接続には一定の遅延時間が規定されており、各呼処理オブジェクトは定められた時間内に処理を終わらなければならない。さらに、交換機は原則として運転停止が許されず、障害になったとしてもその影響範囲は極力最小限にとどめなければならない。

このように超多重性、実時間性、高信頼性を必要とする分散交換システムに追加・変更の容易化を狙いとするオブジェクト指向交換プログラムを適用する際、実行制御システムには次の機能が要求される。

- (1) オブジェクトの位置を隠蔽したメッセージ通信
- (2) 呼処理を継続しながらのオブジェクトの移動、置き換え

各オブジェクトは、相手先のオブジェクトの識別子を指定してメッセージを送ることができる。よって、実行時に通信相手のオブジェクトの物理的な識別子を得るための機構と、通信先が移動したり入れ替え中でも目的のオブジェクトにメッセージを送るための制御が必要となる。

この時、オブジェクト識別子管理には、次のような条件が必要である。

- (a) 交換システムは専用の組み込みシステムであるため、静的なオブジェクト（例えばクラス）の識別子が設計時に決められること。
- (b) 高い処理能力と実時間性の要求を満たすため、識別子の管理を行なう処理には十分な実行効率を達成していること。
- (c) 交換システム全体としてサービスを継続しながら、オブジェクトの移動、置き換えを行う時、識別子の衝突を避けられること。
- (d) 信頼性を高め、障害に対して柔軟に対処する方法として、オブジェクトの冗長構成（2重化）、オブジェクトの移動をサポートできること。

以降、オブジェクトから見て、他のシステムモジュールに存在するオブジェクトをリモートオブジェクト、自分と同じシステムモジュール上にあるオブジェクトをローカルオブジェクトと呼ぶことにする。

3.2 メッセージ通信時のオブジェクトの識別

オブジェクトを識別するために、次の三つの識別子を考える。

- 外部識別子

プログラマが与える識別子で、一意性は設計時に保証する。

- 論理識別子

実行制御システムが個々のオブジェクトに付与、管理する識別子。一意性は実行制御システムが保証する。

- 物理識別子

オブジェクトを具体的にアクセスするための物理アドレスを含んだ識別子。

この時、メッセージ通信のために指定するオブジェクト識別子として、これらのうちいずれかを用いることができる。

メッセージ通信の相手先オブジェクトの指定に外部識別子を用いると、オブジェクト管理の柔軟性は高い。しかし、毎回、外部識別子からオブジェクトの物理アドレスへのマッピングを行う必要があり、そのオーバーヘッドが大きい。さらに、トラヒックに応じて生成されるオブジェクトすべてに設計時に固有の名前を付けることは困難である。

物理識別子を用いると、オブジェクトを移動、置換する制御が難しい。

論理識別子を用いると、外部識別子を論理識別子に変換する機能や、相手のオブジェクトの論理識別子を実行時に知る機構が必要になる。しかし、それ以降はオーバーヘッドなしに論理識別子からオブジェクトの物理アドレスへのマッピングを行うことができる。さらに、オブジェクトの移動、置換時には、論理識別子からオブジェクトの物理アドレスに変換する過程で、メッセージの転送、振り分けなどの制御が可能である。

したがって、本システムでは論理識別子を用いてメッセージ通信を行う方法を取る。

3.3 オブジェクト識別子の内部構造

メッセージ通信の相手先オブジェクトの指定に論理識別子を用いる時、分散を意識しないメッセージ通信を実現するための論理識別子の内部構成として、次の二つの案が考えられる。

案1 各システムモジュールでオブジェクトテーブルを作り、そのインデックスを論理識別子とする(図4)。

案2 論理識別子をオブジェクトの存在するシステムモジュールの識別子とそのシステムモジュール内でのローカルな識別子の組として構成する(図5)。

案1では、論理識別子は単純なインデックスとなるので、取り扱いが容易である。しかし、ローカルオブジェクトに加えて、通信中のリモートオブジェクトの情報もテーブルに登録しなくてはならない。さらに、一連のメッセージ通信が終了した時に、リモートのオブジェクトをテーブルから削除するための処理が必要になる。しかし、リモートオブジェクトの位置情報の変更は一斉に行うことができる。

案2は、案1に比べて取り扱いが難しくなるが、実行制御システムは自システムモジュール内のオブジェクトのローカル識別子だけを管理すればよい。一連のメッセージ通信が終了しても、特に処理をする必要はない。しかし、論理識別子はオブジェクトのローカルデータとなるので、案1のように実行制御システムがリモートオブジェクトの位置情報を一斉に書き換えることはできない。加えて、通信先のオブジェクトが移動すると、メッセージはオブジェクトの元あった位置に送られるので、さらに転送する制御が必要である。

交換システムにおいては、交換装置の故障、ソフトウェアの障害に対して、通話中の呼が影響を受けないようにすることが求められる。すなわち、故障を検出したシステムモジュールが初期設定された後も、通話中の呼に関する処理は極力継続させなければならない。そのため、呼処理プログラムは通話中の呼に関する情報をまとめて、二次記憶上にバックアップをとる。そして、システムモジュールが処理を再開する時に、この情報をもとに適当なオブジェクトが生成され、呼の処理を継続する手法を取る。

この時、案1はシステムモジュール再開時にリモートオブジェクトの情報が必要となるので、実行システム中のオブジェクトテーブルの情報を二次記憶上にバックアップする機能を持つ必要がある。案2は再開時に生成されたオブジェクトがリモートオブジェクトに新しい自分の論理識別子を教えるか、また、再開時にオブジェクトを生成するとき、もとの論理識別子を指定できるようにすれば、実行制御システムとしてバックアップの機能を持たなくてもよい。

以上の議論から、管理のオーバーヘッドの小ささ、移動、置き換えの制御の柔軟性を重視し、本稿では案2の方式を取る。以降、論理識別子を構成するシステムモジュールの識別子をシステムモジュール識別子、ローカルなオブジェクトの識別子をローカルオブジェクト識別子と呼ぶ。

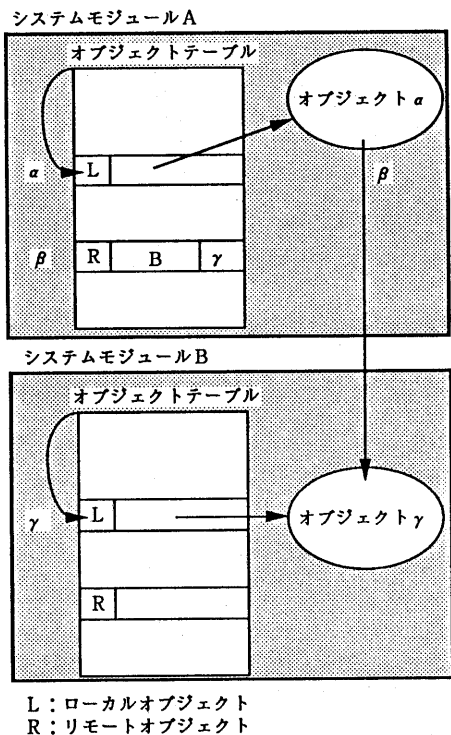


図4: オブジェクトテーブルのインデックスを論理識別子とする方法

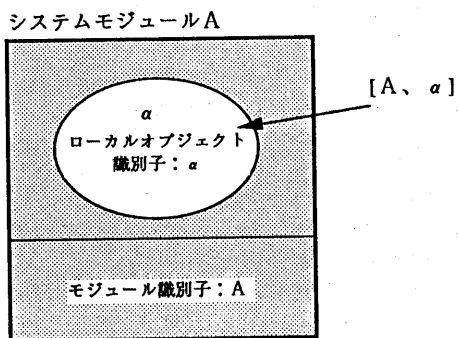


図5: システムモジュール識別子とローカルオブジェクト識別子で論理識別子を構成する方法

3.4 識別子の変換

実行時に実行制御システムが付与・管理する論理識別子を用いて、リモートオブジェクト、ローカルオブジェクトの区別なくメッセージ通信を行えるようにすると、実行時に外部識別子を論理識別子に変換する機構が必要となる。この外部識別子と論理識別子を管理する手法として二つ考えられる。

案1 分散交換システムで一つのデータベースオブジェクトが管理する。

案2 各システムモジュールに分散配置されたデータベースオブジェクトにより管理する。

案1は、データベースオブジェクトの論理識別子を固定的に決める必要があり、データベースオブジェクトの存在するシステムモジュールの故障がシステム全体の停止につながる可能性が強い。

案2では、各システムモジュールで、データベースオブジェクトの論理識別子を固定にすればよく、分散したデータベースオブジェクト間も設計時に定められた論理識別子でメッセージ通信することができる。さらに、一部のシステムモジュールの故障を隠蔽してシステム全体の停止を避けることができる。

したがって、本システムでは案2の方向で検討することとする [4]。以降、このデータベースオブジェクトを識別子管理オブジェクトと呼ぶ。

3.5 静的識別子と動的識別子

識別子管理オブジェクトや各種クラスオブジェクトのように、交換プログラムでは、設計時に論理識別子を決定できるものが多い。したがって、インスタンスのように実行時に動的に割り当てられる識別子に加えて、静的に割り当てられる識別子を取り扱う必要がある。

これらの静的な論理識別子は、各システムモジュールの実行制御システムが立ち上がった時に設定される必要がある。したがって、静的識別子にオブジェクトを割り当てる機能が必要である。システムモジュールの初期設定は、再開のレベル (全体初期設定か部分初期設定か) やシステムの変更などの理由でかわる必要があるため、システム初期設定を専門に行うオブジェクトを設け、これが静的識別子の設定を行う。以降、この機能を行うオブジェクトをシステム初期設定オブジェクトと呼ぶ。

システム初期設定オブジェクトと、先に述べた識別子管理オブジェクトには、システム固定の静的識別子を割り当てる。

4 メッセージ通信制御

各システムモジュールは論理的なシステムモジュール識別子を持ち、モジュール間通信に使用するシス

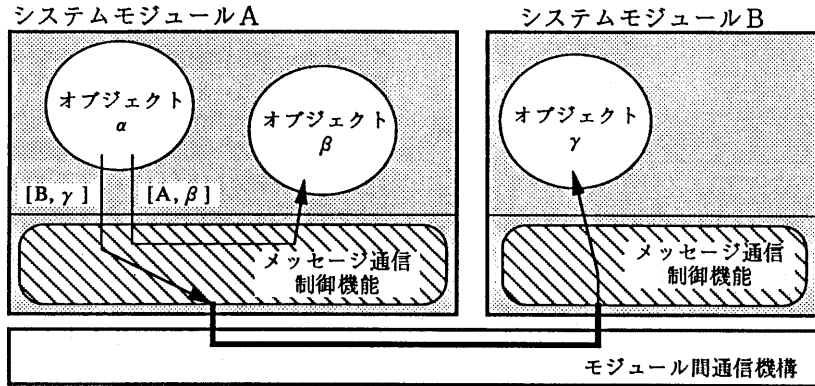


図 6: メッセージ通信制御

システムモジュールのアドレスと 1 対 1 に変換可能であるとすると、このとき、各システムモジュールの実行制御システムは次の情報を管理する必要がある。

- 自システムモジュールのシステムモジュール識別子
- 同一ノード内のシステムモジュールのシステムモジュール識別子とシステムモジュールアドレスの変換表
- 自システムモジュール内のローカルオブジェクト識別子からオブジェクトの物理アドレスへの変換表

この時、実行制御システムのメッセージ通信機能が次のようにオブジェクト間のメッセージ通信を制御する(図 6)。

4.1 送信メッセージの処理

メッセージ通信機能は、送信先オブジェクトの論理識別子内のシステムモジュール識別子を自システムモジュール識別子と比較し、それに応じて次のように処理する。

- (1) ローカルオブジェクトの時
 - ローカルオブジェクト識別子をオブジェクトの物理アドレスに変換する。
 - (1-1) オブジェクトが移動、置換の制御をしている時、メッセージを転送する。
 - (1-2) それ以外の時、物理アドレスによりメッセージの送信処理を行う。
- (2) リモートオブジェクトの時
 - システムモジュール識別子から相手のシステム

モジュールアドレスに変換し、送信元の論理識別子、送信先の論理識別子、送信メッセージの本体を組にして、モジュール間結合機構により相手先システムモジュールに転送する。

4.2 受信メッセージの処理

また、他のシステムモジュールからのメッセージを受信した時、メッセージ通信機能は受信したメッセージのシステムモジュール識別子が自システムモジュール識別子と同じかどうかチェックし、次のように処理する。

- (1) 送信先論理識別子内のローカルオブジェクト識別子を物理アドレスに変換し、移動、置換の制御をしていない時、メッセージの送信処理を行う。
- (2) 移動、置換の制御をしている時は、メッセージを転送する。

4.3 オブジェクト間のアソシエーション手法

分散オブジェクト指向交換プログラムにおいては、Caller オブジェクト、Callee オブジェクトという対等な関係のオブジェクトが呼(コール)で関係付けられる。このような動的に生成されるオブジェクトがメッセージ通信を行うためには、互いの識別子を実行時に知る機構が必要になる。そこで、これまで述べてきたようなオブジェクト識別子を用いた実行制御システムの上で、互いのオブジェクトの論理識別子を次のようにして知ることができる。その具体例を図 7 に示す。

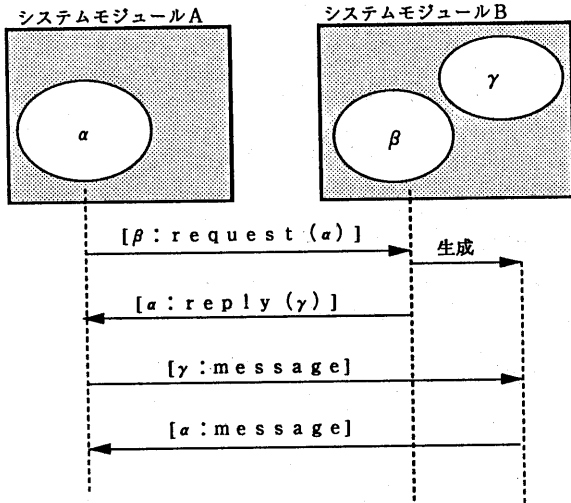


図7: オブジェクト間のアソシエーション手法

α 、 β 、 γ をオブジェクトの論理識別子、A、Bをシステムモジュールとする。Callerオブジェクト α は、システムモジュールBのCalleeクラス β に対して自分に対応してくれるCalleeオブジェクトの生成を要求する。Calleeクラス β はCalleeオブジェクト γ を生成し、その論理識別子をCallerオブジェクト α に返す。以後は、Callerオブジェクト α とCalleeオブジェクト γ とがメッセージを通信する。なお、Calleeクラス β は識別子管理オブジェクトにより知ることができる。

5 メッセージ転送制御

5.1 オブジェクト移動時のメッセージ転送制御

本オブジェクト指向交換プログラムにおいて、オブジェクトが移動するのは次のような場合である。

- (1) システムモジュールは正常に動作しているが、負荷分散などのため一部のオブジェクトが移動する。
- (2) システムモジュールの正常動作ができなくて、他のシステムモジュールにすべて移動する。

この時、処理を中断しないでオブジェクトの移動を行うためには、オブジェクトの元あった位置に到着したメッセージを転送する制御が必要になる [5]。

(1)の場合、メッセージ転送制御にはさらに二つの場合があり、オブジェクトの元あったシステムモジュールのメッセージ通信制御機能により次のように転送される(図8)。

(1-1) オブジェクト移動中

メッセージは転送キューに入れられ、移動完了後転送される。

(1-2) オブジェクト移動完了後

移動先の論理識別子により、メッセージが転送される。

また、(2)の場合には、まず各システムモジュールは移動中システムモジュールへのメッセージ通信を一時ストップする。その間にシステムモジュール上のすべてのオブジェクトを移動させる。その後、移動先の実行制御システムは、新しいシステムモジュールアドレスを各システムモジュールに通知し、メッセージ通信を開始する。この手法により、システムモジュールごとのオブジェクトの移動に対して、メッセージを転送することができる。この手法は、新しくシステムモジュールを追加したり、削除したりするときにも使うことができる。

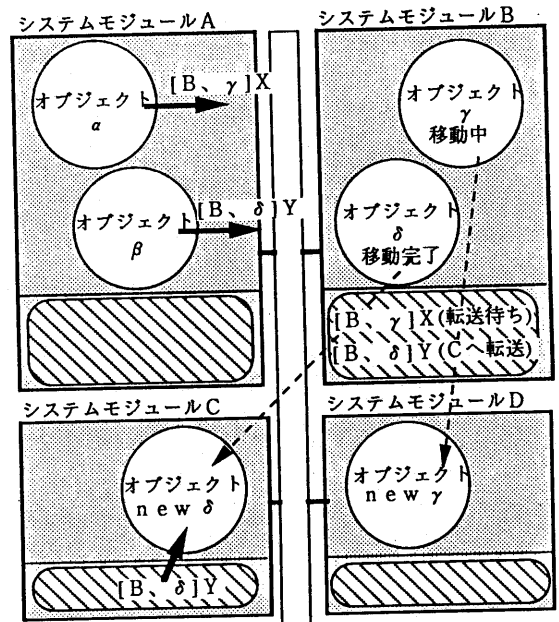
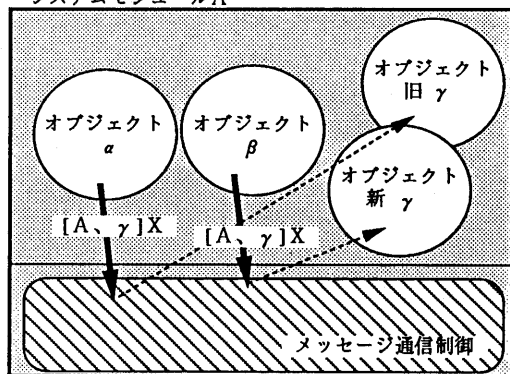


図8: オブジェクト移動時のメッセージ転送制御

5.2 オブジェクト置き換えのメッセージ転送制御

交換システムは連続したサービスの運用が求められるため、処理能力を増やしたり、機能の追加、変更のためにシステムを停止することができない。そのため、本オブジェクト指向交換プログラムでは、旧プログラムを運転中に新しいオブジェクトをロードし、古いオブジェクトを新しいオブジェクトで置

システムモジュールA



オブジェクト γ は置き換え制御中
 オブジェクト a はオブジェクト旧 γ と通信中
 オブジェクト β はオブジェクト新 γ と通信中

図9: オブジェクト置換時のメッセージ転送制御

き換えることによって、機能の追加、変更の容易性向上を目指している。

オブジェクトを置き換える場合、古いオブジェクトが持っていた内部データを新しいオブジェクトに引き継ぐなどの制御が必要になる。特に、交換プログラムの場合には連続して実行されているオブジェクトが多い。そのため、瞬間的な置き換えが難しく、新しいオブジェクトと古いオブジェクトを同時に機能させながら置き換えて行く制御が必要である。

これには、同じ論理識別子に新旧二つのオブジェクトを同時走行させながら、送られてきたメッセージを振り分ける機構が有効である(図9)。これにより、メッセージ通信時に相手先の論理識別子から物理アドレスに変換するときに振り分けをすることが可能である。メッセージには送信元のオブジェクト識別子とメッセージの種類の情報とが載っているため、これらの情報を基に、新旧どちらのオブジェクトにメッセージを送るかを定めることができる。

6 まとめ

超多重性、実時間性、高信頼性を必要とする分散交換システムにオブジェクト指向交換プログラムを適用する際、電話サービスを中断せずに、システムの構成を変えたり、オブジェクトの移動、置き換えを可能とするオブジェクト識別子の管理手法、メッセージ通信の手法について報告した。また、オブジェクトの移動、置き換え時のメッセージの転送制御についても述べた。

今後は、UNIX上の分散交換システムのシミュレーション環境[6]上で本手法を実現し、具体的な評価を行う予定である。またオブジェクトの移動、置き換えの具体的な機構について検討を進める。今後の検討課題として次のものがある。

1. オブジェクト移動の機構。
2. 分散した識別子管理オブジェクトの実現、オブジェクトのグループ化を実現するための、ブロードキャスト、マルチキャストの機能。
3. オブジェクト置換時の制御とその手法を生かした呼処理プログラムの構造。
4. プログラムによってメッセージ転送を制御するデレゲーション(委譲)機構。

これらの検討を基に、ノードを含む通信網全体に分散システムを適用するときのソフトウェア制御技術の検討を行う予定である。

謝辞

オブジェクトの置き換えについて議論していただいた当研究所大崎社員に感謝します。

参考文献

- [1] T.Kai, K.Maruyama and K.Koyanagi: *Program Structure Based on a Concurrent Object Model for Switching Systems*, Proc. Software Engineering for Telecommunication Switching Systems, IEE, pp.34-38 (1989).
- [2] D.R.Cherton: *The V Distributed System*, Comm. ACM, Vol.31, No.3, pp.314-333 (1988).
- [3] D.R.Cherton: *VMTP: A Transport Protocol for the Next Generation of Communication Systems*, Proc. SIGCOMM 86, ACM, pp.406-415 (1986).
- [4] S.Tanaka, M.Kubota and K.Maruyama: *Execution Control for Distributed Object-Oriented Switching Programs*, IEICEJ Technical Report, SSE89-107, pp.71-76 (1989).
- [5] M.L.Powell and B.P.Miller: *Process Migration in DEMOS/MP*, Proc. 9th SIGOPS Symp., ACM, pp.110-119 (1983).
- [6] 田中、久保田: 分散処理交換システムのシミュレーション環境の実現、第39回後期情処全国大会、2R-3 (1989).