ＯＺ：対象指向開放型分散システムアーキテクチャ

6H-5　－－－　オブジェクトの移動と分散ガベージコレクション　－－－

塚本享治（電総研）　田中伸明（松下電器）
吉江信夫（住友電工）　近藤貴士（シャープ）　中込昌吾（ＡＢＣ）

1．まえがき

ＯＺプロジェクトの目的は、分散環境での標準プロトコルのあり方を見直し、利用者に拡張可能な開放型分散システムを提供することである。筆者らは、ＯＺ言語と呼ばれるオブジェクト指向言語を利用者に提供することによって、利用者によるシステムの拡張を実現しようとしている。インタプリタの実行機能の基本部分についてはすでに報告したが、それに含まれていなかったオブジェクトの移動、分散環境下でのガベージコレクションなどについて述べる。

2．オブジェクトとプロセスの概念

2．1　グローバルオブジェクトとローカルオブジェクト

ＯＺシステムでは、オブジェクトは、同じ仮想マシン上のオブジェクトからしか参照されないローカルオブジェクトと、他の仮想マシン上からも参照されうるグローバルオブジェクトに分けられる。

グローバルオブジェクトには、システム内で唯一のｉｄが付けられている。仮想マシン外にあるオブジェクトを参照するときには、すべてこのｉｄを用いて参照を表す。

2．2　モニタとクラス

ＯＺシステムでは、通常のオブジェクトはモニタとクラスに分けられる。その違いを次に示す。

1)．グローバルとローカル

クラス：ローカルオブジェクト

モニタ：グローバルオブジェクト

2)．実行制御

クラス：メッセージは、送信したオブジェクトのプロセスのサブルーチンとして実行される。

モニタ：メッセージを受けると自分のプロセスを生成し、実行する。

2．3　プロセス

ＯＺシステムでは、モニタインスタンスへのメッセージパッシングが起こるとライトウェイトプロセス（以降プロセスと略す。）を生成して、メッセージを実行する。

１つのモニタインスタンスは、複数のプロセスを持つことができる。ある１つのモニタが持っているプロセスは、手続きとモニタに対して宣言された変数（インスタンス変数と呼ばれる）を共有する。各プロセスはプロセスのコンテキストと、手続き固有の変数（メソッド変数と呼ばれる。）を個別に持つ。

3．モニタインスタンスの移動

通常のメッセージパッシングでは、その引数中にモニタインスタンスがあると、そのｉｄのみが送信され、モニタインスタンスの本体は移動しない。

今回は、他のマシン上への、実体を含めたモニタインスタンスの移動の機能も実装した（Fig.1）。

モニタインスタンスの移動を実現するために次のことを実現することが必要となった。

3．1　モニタインスタンスへの参照関係の保持

モニタインスタンスの移動が行われた後、送信元の仮想マシン上には、移動先のモニタインスタンスへの参照を表すオブジェクトが生成され、モニタインスタンス自体は消去される。参照を表すオブジェクトへのメッセージパッシングは、モニタインスタンス本体へ転送される。

3．2　プロセスの移動

モニタインスタンスが移動するときには、いっし

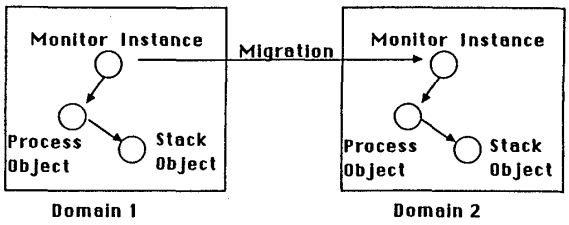

Fig.1 Monitor Migration

OZ: Object Oriented Open Distributed System Architecture -- Object Migration and Distributed Garbage Collection --
Michiharu TUKAMOTO(1), Nobuaki TANAKA(2), Nobuo YOSHIE(3), Takashi KONDO(4), Shogo NAKAGOME(5)
(1)Electrotechnical Laboratory (2)MATSUSHITA Electric Industrial Co.,Ltd.
(3)SUMITOMO Electric Industries,Ltd. (4)SHARP Corporation (5)ABC Co.,Ltd.

ょにプロセスオブジェクト，スタックオブジェクトも運ばれる。プロセスの移動においては、次の点が特徴的な処理である。

1) 手続きのロード

プロセスが実行している手続きのオブジェクト（タイプオブジェクトと呼ぶ）は、いっしょに送らず、送信先でプロセスがスケジューラによってディスパッチされたときに送ることとした（Fig.2）。これは送信先でプロセスが実行されずに、さらに転送されるときのことを考え、通信量を減らすためである。

2) プロセスコンテキスト、スタックの転送

実行効率向上のため、プロセスコンテキスト、スタックにはアドレスが含まれる。これらはすべて、スタックオブジェクト内、あるいはタイプオブジェクト内を指すアドレスなので、それぞれ、スタック、あるいはタイプオブジェクトの先頭からのオフセット値になおして転送する。受信した仮想マシンでは、スタック、あるいはタイプオブジェクトの転送が終了した後にオフセットからアドレスに変換して、実行を再開する。

4. ガベージコレクション

ＯＺシステムでは、他のマシン上にあるオブジェクトの参照を許すので、分散環境を意識したガベージコレクション（ＧＣと略す。）が必要である。

ＧＣは、ローカルＧＣとグローバルＧＣに分けて行われる。

・ローカルＧＣ

ドメインと呼ばれる仮想マシンごとに独立して行われるＧＣであり、他のドメイン上のオブジェクトから参照されていないオブジェクトを回収する。

・グローバルＧＣ

他のドメインから参照されている可能性のあるモニタインスタンスに対して行われるもので、次のような方法で行われる（Fig.3参照）。

1). ローカルＧＣが行われるときに、他の仮想マシン上のモニタインスタンスへの参照を表すオブジェクト（外部参照モニタインスタンスと呼ぶ）から、参照されるモニタインスタンスへ、グローバルＧＣの世代を伝える。これは、あるグローバルＧＣの世代の中では、最初にその外部参照モニタインスタンスが見つかったときに、１回だけ行う。また、そのグローバルＧＣの世代中に生成された外部参照モニタインスタンスについては、転送のときに世代を伝えるので、ここでは、伝搬をしなくてよい。

2). 各仮想マシン間でのグローバルＧＣの世代の伝搬の個数は、次第に減少する。ある時間区間内で、全ての仮想マシン上でローカルＧＣが起こり、しかも世代の伝搬がなくなったとき、グローバルＧＣの世代を、１つ増やす。

3). ローカルＧＣのときに、グローバルＧＣの世代がそのときの世代より２つ以上古いものも同時に回収する。

5. まとめ

モニタインスタンスの移動は可能になったが、今後はこれの活用方法を検討する必要がある。今回発表した方式のグローバルＧＣは、仮想マシン上でローカルＧＣが繰り返し起きることを前提としている。ローカルＧＣが発生しない場合の対策は、別の機会に述べたい。

6. 参考文献

［１］塚本、棟上：ＯＺ：対象指向開放型分散システムアーキテクチャ －アーキテクチャとその実現法－，情報処理学会マルチメディア通信と分散処理研究会３４－７（1987･7･24）

［２］塚本 他：ＯＺ：対象指向開放型分散システムアーキテクチャ，情報処理学会第３６回全国大会講演論文集

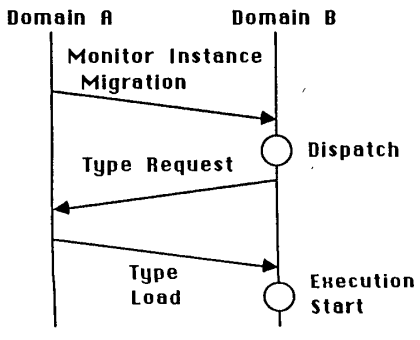

Fig.2 Type Load

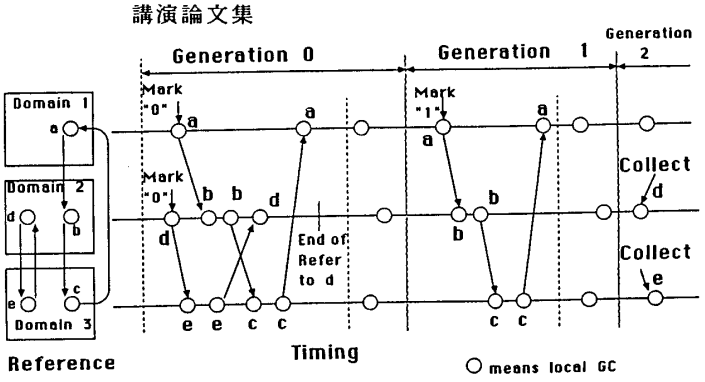

Fig.3 Global GC