分散オブジェクト環境ORCHESTRAの設計*

6F－1

尾形薫†, 八代将慶, 佐藤尚, 倉橋明宏, 近藤邦雄‡
埼玉大学§

1　序論

我々は分散環境下でのオブジェクト指向言語の実装についての研究[1] [2]を行なっている。これはCSCW[3]やグループウェア、CGなどの研究環境の整備を目的として行なわれており、この環境に対しては、ネットワークに関する深い知識なしに分散プログラムを記述できること、という要求があった。

ORCHESTRAはSCORE言語とその実行系からなるシステムである。SCORE言語はクラスオブジェクトとメソッドを記述することができるオブジェクト指向言語で、単一継承を許している。ORCHESTRAではメソッドは並行に動作でき、メッセージはブロックされることがない。クラスツリーは分散環境全体で唯一つしか存在しない。また、ORCHESTRAは複数者利用を前提にしているため、オブジェクトやメソッドに所有者と権限の概念が導入されている。

類似する分散オブジェクト環境としてはArgus[4]、分散化したSmalltalk-80[5]などがあげられるが、Argusは弱い意味でのオブジェクト指向言語であり継承などの機能を備えておらず、また、Smalltalk-80は基本的に単一利用者での使用が前提になっているため、複数利用者が使用した場合の所有権の問題が生じる、などの点でORCHESTRAと異なっている。

2　分散環境ORCHESTRA

分散環境ORCHESTRAはその環境下に唯一つのクラスツリーを保持し、オブジェクトに対する外部または内部からのメッセージにより処理を行なう。実際にはネットワークを介した複数個の仮想機械同士が通信することで分散環境が実現されている。すべての仮想機械は同一のクラスツリーを持ち、変更が生じる度に逐次情報を交換することで全体として統一を保ち続けることができる。

分散処理は基本的にインスタンスオブジェクトを仮想機械間で移動することで行なう。オブジェクトを移動すると、移動元の仮想機械には移動先が記録される。移動したオブジェクトは互いの仮想機械で共有されることになる。

2.1　環境情報

仮想機械が管理する情報は、広域/局所に分けられる。表1に示すように、クラスやメソッドの情報は広域的なため、すべての仮想機械で共有される。このような共有される情報は環境情報と呼ばれる。しかしクラスから派生したインスタンスオブジェクト、クラスやメソッドを管理するID、オブジェクト間のメッセージなどは仮想機械内部で処理される局所的存在のため共有されることはない。

クラス情報	メソッド情報
クラス名	メソッド名
スーパークラス名	クラス名
クラス変数量	ローカル変数量
オブジェクト変数量	引数の数

表1: 仮想機械の管理する環境情報

後述するオブジェクトの移送、他の仮想機械へメッセージ送りなどの事象は基本的に当事者となる仮想機械同士での情報のやりとりになるため、これらの情報は互いの局所情報として扱われる。

2.2　環境の構築

仮想機械はその起動時にすでに起動している他の仮想機械から環境情報を取得し、さらに起動ファイルに定義されたクラス/メソッド定義ファイルを読み込むことで自己の環境を構築する。構築されたクラス/メソッドの情報は他の仮想機械へ伝えられる。他の仮想機械からの環境情報により生成されたクラス/メソッドは仮想存在として扱われ、定義ファイルにより生成されたものは実体として扱われる。

*ORCHESTRA: A Distributed Object Oriented Environment
†OGATA Kaoru: kaoru@ke.ics.saitama-u.ac.jp
‡YASHIRO Masayoshi, SATO Hisashi, KURAHASHI Akihiro, KONDO Kunio
§SAITAMA University

仮想存在への操作を行なう場合には、自動的に実体への操作が行なわれる。例えば、クラス変数の操作を行なうと、そのクラスの実体を持つ仮想機械に対してクラス変数操作の要求が行なわれる。

2.3 環境の存続

仮想機械は任意の時点で自己の持つ環境情報のすべてを他に対して解放することができる。即ちクラスやメソッドの実体を他の仮想機械に委譲することができる。これにより環境を構成する仮想機械の一つが終了しても全体の環境が存続することができる。

2.4 全体環境と個人環境

分散環境 ORCHESTRA は複数利用者を前提にしている。この場合、ある利用者が開発途中のクラス / メソッドが他の利用者にも公開されてしまうのでは不都合が生じる。ORCHESTRA では利用者が作成するクラス / メソッドは原則として非公開とし、これらを個人環境の中に構築するようになっている。個人環境は全体環境の一部として含まれるもので、独立したものではない。単に他者から見えなくなるだけである。また、非公開クラス / メソッドは局所情報として扱われる。個人環境のクラス / メソッドは公開することもできる。公開されたクラス / メソッドは広域情報として取り込まれる。

クラスおよびメソッドは、C 言語で記述し仮想機械のコードの一部として組み込むか、もしくは SCORE 言語を用いることで利用者が自由に定義することができる。SCORE 言語は C 言語のサブセットにクラスとメソッドの定義およびメッセージ送りのための文法を加えた言語である。SCORE 言語は型なしであり、型検査はメソッドの実行時に行なわれる。図 1 は SCORE 言語による複素数型のプログラムの一部である。

```
ClassName  Complex
SuperClass Numeric
var real, im;
class method new(r, i) {
 let tmp;
 tmp := [super new()];
 [tmp setReal(r)]; [tmp setIm(i)];
 return tmp;
}
method setReal(r) { real := r; }
method real() { return real; }
method add(c) {
 return [Complex new(real+[c real],im+[c im])];
}
method sample()
{
 let a, b;
 a := [Complex new(3, 4)];
 b := [Complex new(10, 7)];
 return [a add(b)];
}
```

図 1: SCORE 言語による複素数型の定義

2.5 メッセージ送り

分散環境 ORCHESRA ではメッセージ送りに『過去型、現在型、未来型』の 3 つのメッセージ型を用いることができ、ローカル・リモートの区別なく送ることができる。これらのメッセージ型により並行動作を実現している。

またメッセージは待ちなしで直ちに処理される。メッセージはブロックされることはなく、これにより本質的にデッドロックを起こさないが、危機領域への同時突入を防ぐことができない。

2.6 仮想機械間の通信

仮想機械間の通信には UDP と TCP のハイブリッド方式を用いている。基本的に UDP により通信を行なうが、オブジェクトの移送などの発生により仮想機械間の関係が密になったと判断すると、自動的に相手側の仮想機械と TCP 接続に変更する。共有するオブジェクトが消滅するなど仮想機械間の関係が疎になると TCP 接続を廃止し、UDP に戻る。

3 まとめ

本論文では分散オブジェクト環境として、ORCHESTRA というモデルを提案した。現在 ORCHESTRA は SPARCstation 上に実装され、この環境上で CSCW アプリケーション (MULCO) などを作成して、システム全体の有効性の検証や　追加機能などの検討を行なっている。

今後の課題としては、実行中のメソッドの移動、利用者権限の問題、また、デバッガなど、開発環境の充実などがあげられる。

参考文献

[1] 尾形薫: オブジェクト指向型仮想機械 ORCHESTRA の設計, 埼玉大学情報工学科卒業論文 ICS-93-B-007

[2] OGATA Kaoru et al: ORCHESTRA: A Distributed Object Oriented Environment, 投稿中

[3] 倉橋明宏: 分散型オブジェクト指向言語を用いたグラフィック・コミュニケーション・システムの構築, NICOGRAPH'93, 1993

[4] 前川守他編: 分散オペレーティングシステム, 共立出版, 1991

[5] 所真理雄他 編: オブジェクト指向コンピューティング, 岩波書店, 1993

[6] 米澤明憲, 柴山悦哉: モデルと表現, 岩波書店, 1992