分散環境のための並行オブジェクト指向言語処理系の実行時環境

1K－4

崔梗淦 原田賢一
慶應義塾大学理工学部

1 はじめに

分散環境におけるオブジェクト指向計算モデルでは，オブジェクトを並行実行の単位として，オブジェクト間で通信を行い，メッセージの受理によって各々のオブジェクトが対応する処理を行い，全体で一つの仕事を進めていく．UNIX などの既存の環境で，C や C++ などのプログラミング言語を用いて，分散オブジェクトを実現するには，OS の提供するより原子的な機能を用いてプログラミングせざる得ない．そこで，本研究では，C++ から，ネットワークで接続されている複数のワークステーション (ノード) 上に分散したオブジェクトを，利用するための実行時環境と，拡張された部分を含むコードを C++ のコードに変換する前処理系を設計・実装した．

2 実行時環境

2.1 並行オブジェクトと逐次オブジェクト

我々が実装した実行時環境の目的は，並行オブジェクトと逐次オブジェクトという二種類のオブジェクトを，容易に利用できるようにすることである．並行オブジェクトは，並行実行の単位となって，UNIX のプロセスへと写像される．つまり，後で述べるように並行オブジェクトとして記述されたクラスは，前処理系によって別の一つのプログラムに翻訳される．また，並行オブジェクトは能動的な振る舞いをするために，順路式の概念を取り入れたシナリオをもつことができる．シナリオによってどのメッセージを受理し処理を行うかが決定される．

並行オブジェクト間の通信は，主に非同期に行われる．これを実現するために，並行オブジェクトでは，

- メッセージを送信するスレッド
- メッセージを受信するスレッド
- シナリオを実行するスレッド

の三種のスレッドが協調して動作する．メッセージを送信するスレッドは，必要に応じて生成され，シナリオおよびメソッドから見て非同期の送信を実現する．メッセージを受信するスレッドは，他からのメッセージを常に待ち受け，到着したメッセージを待ち行列に挿入する．シナリオを実行するスレッドは，順路式に従って，待ち行列から受理可能なメッセージを取り出しては対応する処理を行う．

また，メッセージをあるオブジェクトに送信した後，その結果として値を受け取りたい場合がある．非同期の双方向通信は，以下のような仕組みで結果を受け取ることができる．まず，最初にメッセージを送る前に結果受信を仲介する並行オブジェクト (宛先オブジェクト) を生成しておいて，メッセージ送信時に他の引数とともにこのオブジェクトの識別子を相手に送る．メッセージの受信者は，処理の結果をこの宛先オブジェクトに向かって送信する．その後，もとのメッセージの送信者はこの宛先オブジェクトと交信して結果を取り出す．同期通信の場合は，内部的にはメッセージの送信の直後に結果を取り出すように処理する．これによって，結果を必要としない単方向の通信，結果を利用することのできる非同期・同期両方の双方向通信が実現される．

逐次オブジェクトは通常の C++ のオブジェクトと同じように振る舞い，すべてを並行オブジェクトとすることによるオーバヘッドを軽減することができる．

2.2 オブジェクトマネージャ

オブジェクトマネージャは，各ノード上で一つだけ存在し，他の並行オブジェクトからのさまざまな要求を受け付ける．受け付けられる要求には，並行オブジェクトの生成・消滅，ノード間でのオブジェクトの位置の決定，そして，複数のオブジェクトからなるグループの管理などのためのものがある．各ノード上のオブジェクトマネージャは，分散したオブジェクトが協調して動作するための互いに情報を交換する．

3 前処理系

前節で述べた実行時環境を，C++ から容易に利用できるようにするための前処理系を実装した．この前処理形では，C++ でのクラスの定義で `class` を使う代わりに，`concurrent class` を利用して定義されたクラスを，並行オブジェクトを生成するためのクラス (並行クラス) としてコードを変換する．その際に，各並行クラスに，`void scenario()` というメンバ関数が存在すれば，これを並行クラスのシナリオとして扱う．`scnerio()` は形のうえでは，メンバ関数のように記述されるが，順路式の概念実現のための記述ができる．シナリオが無い場合は，解釈できる全てのメッセージを受け取り，メッセージを到着順に処理するものを既定のシナリオとして，自動的に生成す

Runtime Environment of Object-Oriented Concurrent Programming Language for Distributed Environment
Kyong-Rok Choi, Ken'ichi Harada
Dept. of Computer Science, Keio University, 3-14-1 Hiyoshi, Kouhoku-ku, Yokohama 223, JAPAN

る．並行オブジェクトの記述に含まれる他のメンバはすべて，一つのC++のクラスとしてまとめて変換されるので，C++の継承にわずかな制約を加えるだけで，並行クラスでの継承も実現できる．

また，並行クラスから上記のようにプログラムを生成すると同時に，このプログラム(並行オブジェクト)と通信するために必要なコードをまとめたC++のクラスも生成される．C++プログラム中でこのクラスのインスタンスを作ることによって，並行オブジェクトに対応したプログラムが起動される．その並行オブジェクトとの通信は，このクラスのメンバ関数を呼び出すことで行われる．この際に，メッセージの引数はマーシャンリングされ，受信側のメソッドでアンマーシャリングされる．

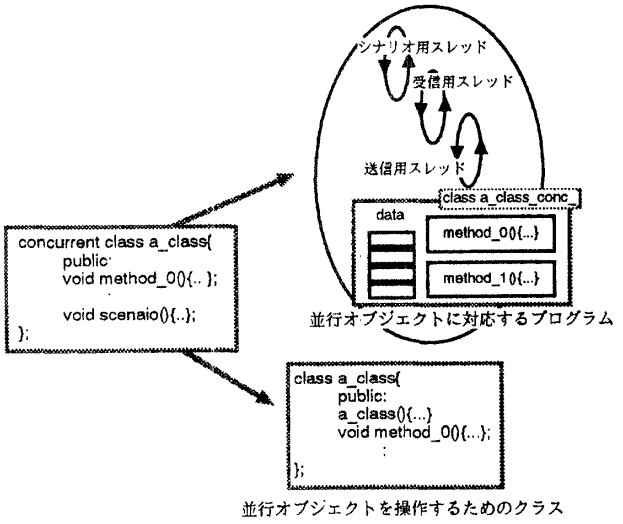

図 1: 並行クラス記述からの変換

ただし，並行オブジェクトを取り入れたことにより，次のような，C++には存在しなかった制約が生じてしまう．

- 並行オブジェクトをまたがるポインタ・参照の利用ができない．
- シナリオの継承ができない．
- 並行クラスのメンバーの可視性がC++と異なる．

ポインタについては，各々の並行オブジェクトがUNIXのプロセスとして実行され，異なるアドレス空間をもつために，その意味を失う．分散共有メモリなどを利用すれば，ある程度はこの問題を解決できるが，現在の実行時環境ではサポートしていない．

シナリオの継承については，とくに解決する方法が無く，既定のシナリオを利用しない並行クラスは必ずシナリオを記述しなければいけない．このため，継承を用いて並行クラスのシナリオで親クラスのメソッドなどを直接呼び出すには，親クラスのシナリオに関する知識の必要性が生じる可能性がある．

並行オブジェクトでのデータメンバには，他の並行オブジェクト内からは直接アクセスできないので，すべてメッセージのやりとりが必要となる．

4 実装

SparcStation5で動作するUNIX系のOSである，SunOS4.1.4上でC++を用いて，上述の実行時環境，前処理系を実装した．実行時環境は，並行オブジェクトに対応するプログラムにリンクするライブラリとオブジェクトマネージャ，オブジェクトマネージャを操作するいくつかのプログラム，および宛先オブジェクトからなる．

ライブラリには，並行オブジェクトを実現するための機能のうち，主に，通信を支援するようなコードや，並行オブジェクトと通信するために生成されるクラスの基底クラスとなるクラスなどが含まれている．オブジェクトマネージャや宛先オブジェクトなどもこの実行時環境のライブラリを用いて実現されている．

オブジェクトの生成は，オブジェクトマネージャへ生成を要求するメッセージを送信することで行われる．オブジェクトマネージャは，要求を受け取ると，fork, execの両システムコールを用いて，要求された並行クラスに対応するプロセスを生成する．オブジェクトマネージャも他の並行オブジェクトと同じ実行時環境を利用しているが，生成した並行オブジェクトに対応するオブジェクトの識別子を要求者に返す際には，前述にように宛先オブジェクトを介して結果をやり取りすることはせず，直接交信を行う．

並行オブジェクト中の三つのスレッドは，SunOS 4.1.4の軽量プロセス(Light Weight Process)ライブラリを利用して実現している．

5 まとめ

既存のUNIXでの分散環境でのプログラミングを容易にする，実行時環境と，それを利用するためのC++言語の前処理系を実装した．C++プログラムから実行時環境を直接利用するようにプログラムを書くことも可能であるし，拡張された構文を使ったプログラムを前処理形を用いて変換することができる．これによりネットワーク上に分散した資源を活用するプログラムの作成が簡単になる．

参考文献

[Koj93] Kojima, K. and M. Noro: "The Design of the Object-Oriented Concurrent Programming Language y," in *Proceedings of the Joint Conference on Soft. Eng. '93*, IPSJ, pp. 59–66.

[Cho95] 崔 梗洙, 小島 一人, 野呂 昌満, 原田 賢一: "並行オブジェクト指向言語yの通信機能," 情報処理学会第50回全国大会講演論文集.

[Koj95] 小島 一人, 野呂 昌満, 崔 梗洙, 原田 賢一: "並行オブジェクト指向言語yにおけるクラス階層の定義と継承," 情報処理学会第50回全国大会講演論文集.

[Cho96] 崔梗洙, 野呂昌満, 原田賢一: "並行オブジェクト指向言語yの実行時環境," 情報処理学会第52回全国大会講演論文集.