分散オブジェクト開発用ツールキット

6D－8

乙川 進一　森 健　中澤 修
沖電気工業株式会社 マルチメディア研究所

1　はじめに

近年，計算機環境の分散化が進み，ミドルウェアとして分散オブジェクト環境が注目されている．分散オブジェクト環境は現在 OMG により標準化が進行中であり，分散オブジェクト間の通信機構については CORBA V1.1 として標準仕様 [1] が規定され，これに準拠した製品も発表されつつある．

このような分散オブジェクト環境を利用すると，分散アプリケーションの作成時につきものである，操作対象のデータの位置やプロセス間の通信の管理が簡単になる．これらの処理を新たに分散処理記述用の言語を作成し，それを用いて記述する方法もあるが，分散オブジェクト環境を利用するほうがより実際的である．しかし，我々が実際に分散オブジェクト環境上で分散アプリケーションの作成を行ってみた限りでは，まだ十分に簡単になったとは言い難い．特に単一言語環境に慣れたユーザにとっては，新たにインタフェース記述言語(Interface Definition Language: IDL)を覚えたり，オブジェクト登録作業が増えるために負担が大きい．

本稿では，以上の問題点を考慮し，多くの分散環境でアプリケーション作成の支援が可能であるような，CORBA V1.1 準拠の分散オブジェクト環境上の分散オブジェクト開発用ツールキットの概要について述べる．

2　分散オブジェクトの開発とその問題点

この節では，現在提供されている CORBA V1.1 に準拠した分散オブジェクト環境上[2] での，分散オブジェクトの開発と，その過程における問題点について述べる．

2.1　分散オブジェクト開発の現状

分散オブジェクトとそのオブジェクトをアクセスするクライアントの作成では，以下の手続きを行う必要がある．

オブジェクトのインタフェース定義の設計:
以下のような情報を IDL で記述し，オブジェクトタイプ定義としてインタフェースリポジトリに登録する．
- ・オブジェクト名，継承するオブジェクト
- ・オブジェクトの持つ属性
- ・オブジェクトの持つオペレーション(メソッド)

オブジェクトのインプリメンテーション(メソッド)の設計／実装:
メソッドの処理の記述は C 又は C++ で行う．これをコンパイルし，実行モジュールを作成する．メソッドの処理内容は以下のようになる．
- ・入力バッファからの入力引数の抽出
- ・他オブジェクトのメソッド呼出し，関数呼出し等
- ・出力バッファへの結果の出力

次に以下のようなメソッド登録情報を IDL で記述し，インタフェース及びインプリメンテーションリポジトリに登録する．
- ・オブジェクト名，オペレーション名，メソッド名
- ・メソッド実行環境
- ・実行モジュールファイル名

オブジェクトのロケーションサービスへの登録:
オブジェクトのインスタンスを作成し，ロケーションサービス(オブジェクトの位置を管理)に登録しておく．

クライアントの実装:
クライアントプログラムの処理を C 又は C++ で記述する．処理内容は以下のようになる．
- ・ドメインへのログイン
- ・リクエストの作成
- ・オブジェクトの獲得
- ・オブジェクトへのリクエストの送信／結果の受信
- ・ドメインからログアウト

なお，各処理の記述には分散オブジェクト環境が提供するルーチン群を用いる．

2.2　分散オブジェクト開発における問題点

実際の分散オブジェクト環境上でのアプリケーションの作成に際して，分散オブジェクト及びそれを用いたクライアント作成には次のような問題点がある．

A Distributed Objects Developer's Tool
Shin-ichi OTOKAWA, Takeshi MORI, Osamu NAKAZAWA
Oki Electric Industry Co.,Ltd.
550-1 Higashiasakawa, Hachioji, Tokyo 193, Japan

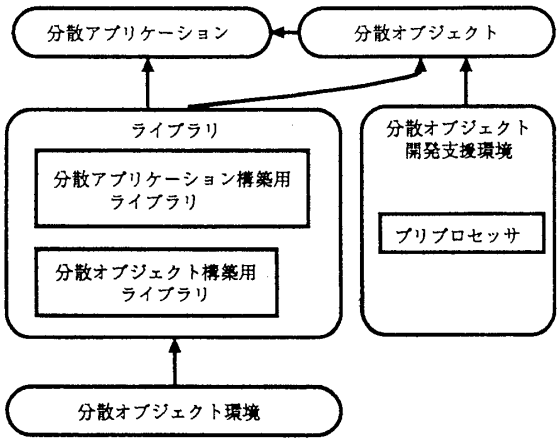

図 1: 分散オブジェクト開発用ツールキット

手順が煩雑:

通常のアプリケーション作成作業と比較して，オブジェクトの登録作業を行う必要がある．しかも，インタフェース／インプリメンテーションの二回にわたって行わねばならない．

プログラムの記述が煩雑:

オブジェクトのインタフェースの記述のために IDL という新たな言語を覚える必要がある．また，メソッドやクライアントの処理の記述でも IDL とのマッピング(データ型変換等)が必要である．さらに，各種の API ルーチンを使いこなさねばならない．

分散オブジェクトのメソッドの実行制御:

CORBA V1.1 では，オブジェクトのメソッドの起動は BOA(Basic Object Adapter) が行う．ユーザは BOA にメソッドの起動方針(Activation Policy)を指定し，メソッドを実行するサーバの動作(オブジェクトの共有／非共有，メソッド毎のサーバの起動等) を選択する．しかし，処理を行う実行主体(プロセス，スレッド等)の選択はできない．このため，例えばメソッド一つの処理にプロセスを一つ割り当てるようなことになってしまう．また処理を分散しようとしても，ネットワーク上のどの場所で処理を行うかの指定ができない．

3 分散オブジェクト開発用ツールキット

前節で述べた問題点を解決するために，プログラム記述の簡易化，分散オブジェクトの登録の簡易化，単一言語環境の提供，の 3 項目を実現する以下のツールキットを開発することとした．図 1 はこのツールキットの概観である．

プリプロセッサ:

C++ によって定義した，分散オブジェクト/メソッドの定義を，分散オブジェクトの定義 (IDL)とメソッドの定義 (C++) に変換するプリプロセッサを作成する．すなわち，分散オブジェクト/メソッドの定義を同時に行う．これによって，ユーザは IDL の記述，メソッドの記述を意識することなく，通常の C++ でのプログラミングと同じ感覚で分散オブジェクトの定義ができる．

クライアント作成のためのライブラリ:

クライアントの作成を簡単化するための C++ のライブラリを作成する．2.1 節で述べた，クライアントが行わなければならない手続きの簡易化を行う．つまり，分散オブジェクト環境が提供しているルーチン群を，クライアントが行う手続きごとにまとめ直す．

分散オブジェクトライブラリ:

分散オブジェクトの定義，クライアントの作成時に用いる C++ のクラスライブラリである．分散オブジェクト環境が提供する分散オブジェクトの定義，メソッドの定義を，C++ のクラスで再定義している．このクラスライブラリを用いることによって，より自然な形で分散オブジェクトを扱うことができる．

4 おわりに

分散オブジェクトの開発を支援するツールキットについて述べた．本ツールキットを利用することにより，CORBA V1.1 準拠の標準的な分散オブジェクト環境上で，より簡単に分散オブジェクト及びそれを利用したアプリケーションプログラムの作成が可能となる．現在ツールキットの試作中である．

今後の課題として，分散オブジェクトのメソッドの実行制御を行うためのプロセス/スレッド用の C++ のライブラリについても考慮中である．将来的にはプロセスツールキット [3] と融合し，分散オブジェクトのテスト／デバッグ環境まで整備していきたい．

参考文献

[1] "The Common Object Request Broker : Architecture and Specification",OMG Document Number 91.12.1 Revision 1.1 Draft 10,OMG,1991

[2] "The HyperDesk™ Distributed Object Management System™ : *A Technical Overview*", HyperDesk Corporation,1992

[3] "分散ソフトウェア開発用ツールキット—ライブラリ—", 鈴木 他，情報処理学会第 48 回（平成 6 年前期）全国大会,1994