情 報 学 基 礎 31－3
（1993．9．14）

分散システムに於けるオブジェクト管理の一考察

吉屋 英二　　藤野 晃延

富士ゼロックス情報システム

オブジェクト指向の概念が提供する様々な長所を、それを損なうことなく分散システム環境に適用しようとした場合に直面する、幾つかの普遍的な課題について検討を行った。課題を解決することは、2つの位相の異なるコンピューティングモデルを、可能な限り両概念の長所を残しつつ、それぞれの概念領域から他方のそれに写像し、統合することに等しい。ここではオブジェクト指向の大きな特徴の1つである継承機構を、分散環境の特徴である透過性の基で実現するためのモデルと、その際のオブジェクト管理のありかたについて考察した。

A novel approach to represent object management in a distributed system

Eiji Yoshiya　　Terunobu Fujino

Fuji Xerox Information Systems

Without having loss the strength of object-orientation, charenges are the effective implementation of object management facilities in a distributed system while applying object-orientation concept into distributed environment. Both object-orientation and distributed computing have excellent characteristics, the goal is to unify these 2 different computing aspects into one by mapping the basic factors of each contents, specifically inheritance mechanism of object-orientation and transparency representation of distributed environment, from one side to another as well as keeping its potential usefulness.

1 はじめに

オブジェクトベースド[7]による分散システムの実現は既に幾つか試みられている[1][6][2][5]が、継承機構を含みオブジェクト指向の概念をできるだけそのまま分散システムに適用するべく、その際の課題について検討を行った。

具体的には、分散システムの特徴である高い透過性とマルチユーザ環境という利点を活かしつつ、オブジェクト指向における継承機構の実現形態と、実行単位であるオブジェクトの管理方式について考察した。

2 分散システムにおけるオブジェクト指向の課題

初めに分散システムの特質について言及する。分散システムに固有な特質としては以下に示すものがある。

- 大域的情報の存在を前提とすることができない
- 常にどこかで部分的障害が発生している可能性がある

これらの特質から、分散システムにおいてオブジェクト指向を実現するためには、以下に示す課題を解決しなければならない。

(a) **クラスライブラリを共有するための機構**

分散システムでは、マルチユーザ環境のため、全てのユーザに共通する大域的なクラスライブラリの存在を前提とすることができない。仮に初期段階に共通なクラスライブラリを提供したとしても、複数のユーザがそれぞれ独立にそのクラスライブラリに対して改変を行っていくため、結局、共通なクラスライブラリは存在しなくなる。

一方で、オブジェクト指向の利点は単に実行環境におけるコンピューテーション機構という側面のみではなく、ソフトウェアの開発において「巨人の肩に乗る」[3]と形容される、継承機構を用いたソフトウェアの高い再利用性にもある。このオブジェクト指向の恩恵を分散システムにおいても変わらず享受するためには、クラスライブラリを共有するための機構が不可欠である。

(b) **マルチユーザ環境下における定義クラス名の衝突回避**

クラスライブラリを共有するという要求の一方で、各ユーザが自由に独自のクラスを定義し、改変できるという自由度も保証できることが望ましい。

例えば試行中のある部分的なクラス群についての変更や改変は、他のユーザに何ら影響を及ぼすことなく、その該当ユーザの環境内においてのみ有効であり、試行が完了した後、そのクラス群を公開して共有できるとよい。

既に公開されているクラスライブラリを再利用しつつ、公開されていないクラス群においても、ユーザ毎にクラスの名前付けやその階層構造を自由に定義可能とするには次の機構を備えなければならない。

- 多重化されたスコープによる名前空間管理
- 公開されているクラスライブラリと、非公開の各部分（クラスの部分ツリー）を合成することによる、クラス階層構造の導出

このような課題を解決するための機構とそのオブジェクト管理方式について述べる。

3 前提とする分散システム環境モデル

前項の課題に対する検討の前に、前提とする分散システムの環境モデルについて概要を示す。

ここでは便宜的に分散システム環境を物理的構成と論理的構成と分けて考える。取り敢えず物理的構成として1つのオペレーティングシステムをプラットフォームと捉えるものとする。その時、論理的構成は以下の要素からなる。

ワールド その分散システム全体のこと。分散システムとしてのサービスはこのワールドを境界とする。

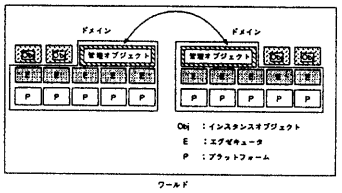
ドメイン ワールド内に存在する論理的な自治管理単位。1つの自治管理単位ごとにアカウントやセキュリティ等に関する管理ポリシーが唯一存在する。

エグゼキュータ オブジェクトを実行する主体。各プラットフォーム上に生成される。

管理オブジェクト システムによりあらかじめ提供されるオブジェクト群。他のオブジェクトを管理し、協調動作を可能とする。

インスタンスオブジェクト 管理オブジェクトを除いた一般のオブジェクト。

この分散システムの環境モデルを以下に示す。



分散システムの環境モデル

上に示した管理機構では、全てのオブジェクトはIDにより管理される。あるオブジェクトの提供するサービスを利用するためには、該当するそのオブジェクトのIDが判明すればよい。

この環境では、管理オブジェクトの1つであるがネームサーバに問い合わせることで必要とするサーバ(それ自身オブジェクトである)のIDを得ることができ、必要とするサービスを得ることができる。ネームサーバの位置は各エグゼキュータが生得的に把握している。

4 管理機構

前述の環境モデルにおける広義のオブジェクト管理機構として、① 版管理情報を含むクラス情報を提供するクラス管理、② クラス以外の名称とそれに対応するオブジェクトとの関係情報を提供する名称管理、③ アカウント等のユーザに関する情報を司るユーザ管理、④ オブジェクトの生成や管理を行うオブジェクト管理等がある。以下にこれら管理機能を提供する管理オブジェクトを示す。管理オブジェクトは、ドメイン内において、複数のエグゼキュータにまたがって一貫した管理機能の提供をする。

- クラス管理
  コンパイル時に定義情報やクラス名称の解決に必要な情報を提供し、実行時には実行コードを検索し、提供する。[12]
  スクールサーバ
  個別のクラス名空間を提供する対応表（スクール）を、登録/管理するサーバ。
  クラスバージョンサーバ
  版管理を可能とするためクラスサーバを管理するサーバ。現在有効であるクラスサーバへの参照情報を保持している。
  クラスサーバ
  クラスの実行可能コード等を検索、提供するサーバ。

- 名称管理
  クラス以外の名称を管理/提供する。
  ネームサーバ
  名前やクラスのIDから、オブジェクトのIDリストを返すサービスを提供するサーバ。管理する名前の種類によりタイプが分けられる。例えば、① ユーザ名、② プラットフォーム名、③ ドメイン名、④ サーバ名等がある。
  トレーディングサーバ
  負荷情報やネットワーク構造のような情報を持っていて最適な選択を提供したり、抽象的な名称からの検索を提供したりするサーバ。

- オブジェクト管理
  インスタンスオブジェクトを生成/管理する。
  オブジェクトマネージャ
  エグゼキュータ毎にあり、インスタンスオブジェクトの生成と管理を行うオブジェクト。

5 クラスのバージョン

クラスの利用者にとって、利用しているクラスが変更された場合、再コンパイルが必要かどうかは一概には判定できない。クラスの2つの利用形態、即ちメソッド呼び出しによるサービス利用の場合と、静的に構造及び機能を継承する場合の二種類に分けて考え、それぞれについて再コンパイルの必要性を判別できることが望まれる。

- メソッドを利用している場合には、メソッドのシグネチャに変更が生じた時、

- 継承している場合には、メソッドのシグネチャ又はインスタンス変数に変更が生じた時、

それぞれ再コンパイルを実施する必要がある。

6 クラスサーバ

クラスサーバとは、コンパイル時にソースコードが参照しているクラスの情報を提供したり、実行時にクラスの実行コードを提供するオブジェクトである。クラスサーバはクラスバージョンサーバにより管理され、クラスバージョンサーバは自分の子の中の一つを公開バージョンとして持っている。コンパイル時や実行時には、クラスバージョンサーバにクラスの情報を問い合わせるので、公開バージョンの変更を行うことにより、動的にクラスのバージョンを決定することができる。

7 スクールサーバ

小規模の環境と異なり、(特にマルチユーザの)分散環境では、クラス名が容易に衝突してしまうため、これを大域的にユニークなものとして扱うことは難しい。この問題を解決するため、クラス名と実際のクラスとの対応表(**スクール**と言う)を環境中に維持する。この役を行なうオブジェクトを**スクールサーバ**という。クラス名の指定に際してスクールを指定することができ、出所の異なるライブラリを同時に利用してもクラス名の衝突が起きないようにし、同時にクラス階層構造の合成を行なうことができる。

8 コンフィグレーション

これまで述べてきたように、クラスの名称は

「スクール名」+「クラス名」+「バージョン」

で一意に決定できる。

しかし、ソースコード上でクラスを一意に指定すると、改変され新しく公開されたスクールを使用したい場合にソースコードを変更しなければならない。そこで、ソースコード上ではスクール名を直接指定せずシンボリックな名称を与え、コンパイル時にこの名称と実際のスクールを対応させることにする。この対応をコンフィグレーションと呼ぶ。

クラスのバージョンの指定は、スクール中でクラス名とクラスサーバを対応させることで行なう。しかし、前に述べたようにバージョンは省略可能である。この場合はクラスサーバの代りにクラスバージョンサーバをクラス名に対応させる。バージョンが省略された場合は実行時に動的にバージョンが決定される。

9 まとめ

分散システムにおけるオブジェクト指向の実現手段を、特にクラス管理を中心に考察を行ない、管理機構として必要なサービスを明らかにすると共に、オブジェクト管理のための名称空間の基本構成について述べた。

クラスの名称を、スクール名、クラス名、バージョンで決定し、コンパイル時のコンフィグレーション、スクールの適用と切替えにより、柔軟な名前付けを行なうことができ、クラス名の衝突の回避や、クラスのバージョンの動的な指定や静的な指定を可能とすることができる。

尚、本論文中に参照したOZ++[8]についての内容は、情報処理振興事業協会「開放型基盤ソフトウェア研究開発評価事業」の一環として行われている研究に基づいている。

参考文献

[1] R.J. van der Linden, and J. Sventek, *The ANSA Trading Service*, IEEE Distributed Processing Technical Committee Newsletter, Vol. 13(4), 1991

[2] B. Liskov, *Distributed programming in Argus*, CACM, March 1988, Vol. 31, No. 3

[3] B. Meyer, *Lessons from the design of Eiffel libraries*, CACM, September 1991, Vol. 33, No. 9

[4] Sape Mullender (Eds.), *Distributed Systems*, ACM Press , New York NY, 1989

[5] S.J. Mullender and A.S. Tanenbaum, *The design of a capability-based distributed operating system*, Comput. J. , Aug. 1986, Vol. 29, No. 4

[6] Object Management Group, *The Common Object Request Broker:Architecture and Specification*, OMG Document No. 91.12.1, Rev. 1.1, 1991

[7] P. Wegner, *Dimensions of object-based language design*, ACM Proceedings of OOPSLA'87, pp. 168-182, 1987

[8] 塚本他:「OZ++ 開発計画」、情報処理学会第45回全国大会、Oct. 1992

[9] 濱崎他:「オブジェクト指向分散環境 OZ++ の通信機構の基本設計」, 情報処理学会第 46 回全国大会, Mar. 1993

[10] 西岡他:「オブジェクト指向分散環境 OZ++ の言語の基本設計」, 情報処理学会第 46 回全国大会, Mar. 1993

[11] 濱崎他:「オブジェクト指向分散環境 OZ++ の基本設計」, SWoPP '93, Aug. 1993

[12] 吉屋他:「オブジェクト指向分散環境 OZ++ のクラス管理方式」, 情報処理学会第 47 回全国大会, Oct. 1993

[13] 新部他:「オブジェクト指向分散環境 OZ++ の実行モデル」, 情報処理学会第 47 回全国大会, Oct. 1993