オブジェクト指向分散環境 OZ++ の言語の基本設計

5F−2

西岡利博 (三菱総研*) 藤野晃延 (富士ゼロックス情報システム)
音川英之 (シャープ*) 鈴木敬行 (シャープビジネスコンピュータソフトウェア*)
平川秀忠 (日本ユニシス*) 石川雅基 (三菱総研) 田代秀一 (電総研)
塚本 亨治 (電総研)

1 はじめに

OZ++ は、オブジェクトの交換と共有に基づく、オブジェクト指向分散環境である。現在その基本設計を進めている。基本的なオブジェクトモデル、および実行意味論は、電総研で開発された分散システム OZ+[1] を基本としている。

本稿では、 OZ++ 言語の基本設計の概要について述べる。

2 概要

OZ++ 言語は分散オブジェクト指向プログラミング言語である。ネットワーク上に分散した資源を利用する計算を記述するため、オブジェクトの位置に依らない、位置透過なプログラミングが可能である。

OZ++ 言語は C をベースとしているが、 C にはない、分散オブジェクト指向プログラミングのためのフィーチャ (クラス定義、メッセージパッシング、マルチスレッドプログラミングのための言語要素) が追加されている。一方、型システムの一部 (列挙型、構造体、共用体、ポインタ型、キャスティング、型定義など)、特殊なエクステント (静的変数、外部変数) を扱う言語要素、ユーザ定義の関数などの機能が削られている。

3 オブジェクトモデル

OZ++ 言語では二種類のオブジェクトモデルを提供する [3]。

- アクティブオブジェクト
  メソッドを実行するスレッドをそのオブジェクト自身で管理するオブジェクト。メソッドの起動はメッセージの送信によって行なわれる。

- パッシブオブジェクト
  オブジェクトの外側にあるスレッドからメソッドを実行されるオブジェクト。メソッドの起動は関数コールによって行なわれる。

いずれもメソッドの起動は動的束縛で行なわれる。あるパッシブオブジェクトは単一のアクティブオブジェクトの部品として存在する。計算は、パッシブオブジェクトを部品として持っているアクティブオブジェクトどうしが、互いにメッセージを送受信することによって進む。

4 型とクラス

OZ++ 言語は弱い型システムをサポートする。組み込みの型として、整数型、文字型、浮動小数点型を持つ。共用体、構造体、ポインタ、配列などの派生型はサポートされない。

* 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」研究員

Fundamental Design of Object-Oriented Distributed Programming Language in OZ++: An Object-Oriented Distributed Systems Environment

Toshihiro Nishioka (Mitsubishi Research Institute, Inc.*),
Terunobu Fujino (Fuji XEROX Information Systems Co., Ltd.),
Hideyuki Otokawa (Sharp Corporation*), Takayuki Suzuki (Sharp Business Computer Software Co., Ltd.*),
Hidetada Hirakawa (Nihon Unisys, Ltd.*), Masaki Ishikawa (Mitsubishi Research Institute, Inc.),
Shuichi Tashiro (Electrotechnical Laboratory), Michiharu Tsukamoto (Electrotechnical Laboratory),
*: Researcher of Research, Development and Evaluation of Open Fundamental Software Technology in Information-Technology Promotion Agency, Japan

クラスは、言語中では型としても振舞う。データを表す言語要素（変数、仮引数、メソッドの戻り値など）を、あるクラス（またはそのサブクラス）のインスタンスであると宣言することができる。コンパイラはこれらに対して静的な型チェックを行なう。弱い型システムを導入することで、送信しようとするメッセージに対応するセレクタがレシーバクラスに存在するかどうかをコンパイル時にチェックすることによって、実行時の method not found の少ない、信頼性の高いプログラミングを支援する。
また、OZ++ 言語では多重継承をサポートする。バグの少ないプログラミングにとって、概念モデルからプログラムへの写像が明解であることが最も重要であり、多重継承はそのための道具として有用である。

5 名称空間

言語で意識される名称空間は、クラス名の空間だけである。他のすべての言語要素の名称は、クラス名空間に依存して定義される。クラス名は、“クラス名空間名 + クラス名”の形式で参照される。
定数名、例外名は、その定数や例外が定義されているクラス（またはそのサブクラス）を前置して記述することにより、解決する。
多重継承によるメソッド名やインスタンス変数名の衝突は、別名を与えることで解決する。反復継承は禁止される。

6 実行意味論

OZ++ 言語はマルチスレッドプログラミングをサポートする。アクティブオブジェクトは、メッセージを受信すると、それを処理するスレッドを一つ生成する。アクティブオブジェクトは、それ自身モニタ [4]である。アクティブオブジェクト中の各スレッドは、条件変数によって同期をとる。

7 実行環境

OZ++ 言語は OZ++ 環境上のコンパイラでコンパイルされる。そのオブジェクトコードは OZ++ 環境にロードされ、その支援を受けながら計算を行なう。

8 関連研究

OZ++ の設計にあたって、OZ+[1] の研究成果を活用した。特に、基本的なオブジェクトモデルおよび実行意味論は、OZ+ のものを踏襲した。両者の相違は主に以下のような点である。

- OZ+ 言語が独自の文法を持った純オブジェクト指向言語であったのに対し、OZ++ は既存の言語をベースとしている。
- OZ+ がバイトコードで動作していたのに対し、OZ++ 言語はコンパイラがネイティブコードを出力することを前提にしている。
- OZ+ 言語が単一継承であったのに対し、OZ++ 言語は多重継承をサポートする。

9 まとめ

OZ++ 言語について、その基本設計について述べた。OZ++ 言語の特徴は以下の通りである。

- 分散オブジェクト指向プログラミングを支援する。
- 弱い型システムをサポートしている。
- 多重継承をサポートしている。

今後さらに以下の点について設計を進める計画である。

- トランザクション（処理のアトミシティ）
- 例外処理

本研究は、情報処理振興事業協会 (IPA) の「開放型基盤ソフトウェア研究開発評価事業」の一環として行なわれたものである。

参考文献

[1] 塚本, 浜崎他:「オブジェクト指向開放型分散システム OZ+ の研究開発」, 電総研彙報, vol. 56, No. 9, 1992.
[2] 浜崎 他 :「オブジェクト 指向開放型分散環境 OZ++ の通信機構の基本設計」, 情報処理学会第 46 回全国大会, 1993
[3] Chin, Chanson: *“Distributed Object-Based Programming Systems”*, ACM Computing Surveys, vol. 23, No. 1, 1991.
[4] 二木, 塚本他:「新形態プログラミング: 現状と展望」, 電総研調査報告第 210 号, pp. 95-98, 1984