

プロトタイプによるオブジェクト指向言語

3H-1

におけるリレーション

河田恭郎 アンドリュー・E・サントサ 前川 守

電気通信大学大学院 情報システム学研究科

1 はじめに

本稿では、我々が進めてきていた Eunice プロジェクトを簡単に紹介した後、その核となる Eunice97 言語の特徴について概観する。

2 Eunice プロジェクト

ソフトウェア工学の中心的課題とは、大規模ソフトウェア開発をいかに容易にするか、ということである。我々が進めている Eunice プロジェクトではこの課題に、仕様記述から設計、実装までを支援する単一言語を提供する、というアプローチで取り組む [1]。そのような言語として、我々は Eunice 言語を設計してきた。

その目的の違いから、通常の実装用プログラミング言語と比較して、Eunice 言語に対しては特に以下のような要請がある：

A. 開発の上流工程の支援

プログラミング言語は、ソフトウェア開発工程のうち実装段階を支援する。要求仕様抽出や仕様記述の段階などの開発の上流工程は通常、各種開発手法やそれに準拠した CASE ツールにより直接的・間接的に支援されている。しかし、Eunice はこれらの工程も支援しなくてはならない。

B. 拡張性

ソフトウェアのライフサイクルの中では、通常先に述べたように多くの言語（図式言語も含めて）が情報の表現媒体として利用される。我々は Eunice がこれら全てに代わることを目標としているが、単純にあらゆるものを取り込もうとすると言語仕様が膨大になってしまい、現実的ではない。そこで Eunice の基本言語仕様は極力小さく保つ一方、強力な拡張機能を持たせることで、さまざまな要求に答えられるようにする。

3 Eunice97

本説では、Eunice 言語の最新バージョン Eunice97 の暫定的な仕様について述べる。

Eunice97 は以下のような特徴を持つ。

1. プロトタイプによる (prototype-based) オブジェクト指向
2. メタオブジェクト・プロトコル (MetaObject Protocol, MOP)

Eunice97 は、言語自身の各種拡張と、開発環境の各種ツールとの連携 [2] のために MOP を利用する。

3. リレーションをファーストクラス・オブジェクト (first-class objects) して導入

他のオブジェクト指向言語・システムでは、MOP を支援するシステムでさえ、汎化/特化リレーション、あるいは類形化/個別化リレーションは、最初からその存在を仮定している。具体的には、前者は通常クラス間のインヘリタンス機構として実現し、後者は同じく通常クラスとそれから生成されるインスタンスとして実現している。Eunice97 においては、これら基本的なリレーションですら、汎用のリレーションの一種に過ぎない。

以上述べたような特徴により、Eunice97 は以下のような利点を持つことになる：

- 開発初期の支援が十分に行なえる (A.) : OMT [3] に代表されるオブジェクト指向開発手法では、最初にオブジェクトの抽出を行なうのが通常である。この段階では「名詞で表現されるのがオブジェクト、動詞で表現されるのがメソッド」などという大雑把な指標しかない。また、何がクラスで何がインスタンスなのかの区別も決して自明ではない。

しかし Eunice97 の世界では、1. と 3. から、この種の「誤り」はもはや本質的な誤りではない。クラスもインスタンスも、スロットもメソッドもリレーションも、最初は全て、「よくわからないがとにかくオブジェクト」という段階の認識も表現できる。また、一旦クラス (相当) と認識したオブジェクトを、インスタンス (相当) と認識し直

したりするようなことまで可能である。

このような柔軟性は、他のツール、システム、言語には見当たらない。

- 概念モデリングが行ないやすい (A.) : オブジェクトとその間に成立するリレーションによる概念/世界モデリングは、ERモデルや意味ネットワークとして現れたアイデアの自然な延長である。現に多くのオブジェクト指向開発手法で採用されており、その有効性は広く認められている。Eunice97は、通常のオブジェクト指向言語において、クラス間の継承、クラスをテンプレートとしたインスタンス生成、オブジェクトのスロット、などの機構を通じて陰に提供されている標準的なリレーションはもとより、任意のリレーションを支援する(3.)ことでこれを可能にする。リレーション間のリレーション、すなわち、高階リレーションをも表現できる。高階リレーションにより、推移性を持つリレーション、反射的なリレーションまで表現でき、その表現力は極めて豊かである。
- さまざまなインヘリタンス機構: インヘリタンスは、オブジェクト指向言語・システムの特徴の一つと通常数えられるが、実はその定義は多種多様であり、しかも、概念の特殊化、コードの再利用、インタフェースの共有など、通常複数の意味的関連を負わされている[4]。
どのようなインヘリタンスを用意するかが、オブジェクト指向言語・システムの設計において常に問題となっていたが、Eunice97においてはそれは問題にならない。なぜなら、表現したい意味的関連に1対1に対応するリレーションを定義すればよいからである。この機能により、Eunice97による記述は、非常に概念的に明確になる。
- 基本操作: オブジェクトに対する、copyのような基本的操作は、一般に shallow copy と deep copy を必要に応じて利用する。しかし、どちらがふさわしいかは、リレーションの性質として定義できるものである[5]。同様に、オブジェクトに対するそのような操作がどのように他のオブジェクトに伝搬するかを、リレーションの性質として記述できる。

3.1 関連研究

明示的にリレーションを採り入れたオブジェクト指向言語としては、DSM [6]がある。ただし、リレーシ

ョンはファーストクラス・オブジェクトではないし、メタ・プログラミングが可能なのでもない。しかしその他の点についてはEunice97はDSMより大きく影響を受けている。Eunice97は[7]にあげられている大半の問題を解決したことになる。

4 まとめ

本稿では、我々が進めてきていたEuniceプロジェクトの目的を述べた後、それを実現する上での核となるEunice97言語を概観した。その特徴は、MOPつきのプロトタイプ・ベースのオブジェクト指向を採用し、かつリレーションをファーストクラス・オブジェクトとしていることである。その期待される利点は多岐に渡る。

なお別に発表した[8]では、プログラマが、プロトタイプ・ベースのオブジェクト指向の視点と、クラス・ベースのオブジェクト指向の視点を随時切替えることができることの利点とその実現の方針について述べている。

参考文献

- [1] Y. Kawata, et. al. A perspective on a Eunice-based control system development environment. In Qiangnan Sun, et. al., ed., *Computer Applications in Production Engineering: Proceedings of CAPE '95*, pp. 540-547. International Federation for Information Processing (IFIP), Chapman & Hall, 1995.
- [2] 磯谷幸太郎, 河田恭郎, 前川守. 言語中心の開発支援環境におけるツール間の連携. 情報処理学会第55回全国大会予稿集, 1997. To appear.
- [3] James Rumbaugh, Michael Blaha, William Premerlani, Frederick Eddy, and William Lorenson. *Object-Oriented Modeling And Design*. Prentice Hall, 1991.
- [4] Antero Taivalsaari. On the notion of inheritance. *ACM Computing Surveys*, Vol. 28, No. 3, pp. 438-479, September 1996.
- [5] James Rumbaugh. Controlling propagation of operations using attributes on relations. *ACM SIGPLAN Notices*, Vol. 23, No. 11, pp. 285-296, November 1988. Norman Meyrowitz, ed., *Proceedings of OOPSLA '88*.
- [6] Ashwin V. Shah, James E. Rumbaugh, Jung H. Hamel, and Renee A. Borsari. DSM: An object-relationship modeling language. *ACM SIGPLAN Notices*, Vol. 24, No. 10, pp. 191-202, October 1989. Norman Meyrowitz, ed., *Proceedings of OOPSLA '89*.
- [7] James Rumbaugh. Relations as semantic constructs in an object-oriented language. *ACM SIGPLAN Notices*, Vol. 22, No. 12, pp. 466-481, December 1987. Norman Meyrowitz, ed., *Proceedings of OOPSLA '87*.
- [8] 河田恭郎, Andrew E. Santosa, 前川守. プロトタイプによるOOプログラミングとクラスによるOOプログラミングの融合. 情報処理学会夏のプログラミングシンポジウム予稿集, 1997. To appear.