動的変更と並列性を考慮したメッセージ委譲機構

2E−4

高田敏弘
NTT 基礎研究所

1 はじめに

従来、オブジェクト指向システムの多くは継承(inheritance)を用いて概念の抽象化やプログラムの部品化を行なってきた。それに対しLiebermanは、委譲(delegation)を用いることにより、従来の継承によるシステムに較べ、より柔軟で高い記述力を得る方法を示した[Lie86]。

しかしLiebermanはオブジェクトが受け取ったメッセージを他のオブジェクトへたらい回しにするという委譲の基本的な仕組みを示しただけであり、メッセージ委譲の制御方法などに関してはその後様々な研究がなされてきた[Alm89] [MR89]。

本稿では

- オブジェクトの性質の動的変更を容易にすること
- システム全体の並列性を向上させること

の2点を目的とした新たなメッセージ委譲機構を提案する。

2 「たらい回し」と「門前払い」

まず最初にオブジェクトの性質の動的変更を容易にするために、メッセージ委譲の概念の拡張を行なう。本稿では今までのLiebermanらの委譲機構を「たらい回し」と呼び、拡張した委譲機構のことを「門前払い」と呼ぶこととする。以下ではそれらについて説明する。

2.1 「たらい回し」

「たらい回し」とはLiebermanらの委譲と同様のものであり、自分では処理できないメッセージを他のオブジェクトに転送する機能をオブジェクトに与えるものである。

「たらい回し」を用いることによって、複数のオブジェクト間でその状態や挙動を共有することが可能になる。またメッセージのたらい回し先を動的に変更することにより、そのオブジェクトの性質を動的に変更することも可能となる。

2.2 門前払い

「たらい回し」が、自分では処理できないメッセージだけを他のオブジェクトに転送するのに対し、「門前払い」は、そのメッセージを自分で処理できるかどうかに関係なく他のオブジェクトに転送する機能をオブジェクトに与えるものである。

「たらい回し」を用いた場合、委譲先のオブジェクトを変更することによって、オブジェクトの挙動の追加や委譲を用いて実現されている機能の変更は可能である。しかしこの方法では、オブジェクトがメソッドを自分自身で持つことにより実現している挙動を変更することはできない。自分自身で実現している挙動の変更を行なうために、メソッド定義の動的な変更や削除を可能にするシステムも存在するが、本稿ではメソッド定義の直接的な変更ではなく、ここに述べた「門前払い」という拡張した委譲機構によってそれを解決する。

すなわち、オブジェクトOがメッセージMに対応するメソッドを持つ場合、メッセージMをオブジェクトO'へ「門前払い」することにより、メソッド自身を書き換えることなしにOのMに対する挙動を変更することができる。また全てのメッセージを他のオブジェクトに「門前払い」してしまうことによって、Actor[Agh87]のbecomeに相当する機能の実現も可能である。

3 複数のオブジェクトへのメッセージの委譲

継承においてクラス階層や多重継承といったものが存在するように、本稿で述べる委譲機構においても、委譲の階層や複数のオブジェクトへのメッセージの委譲(多重委譲)などを考えることが可能である。

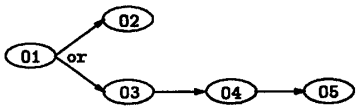
更に委譲を用いているために容易に実現可能な機能として、複数のオブジェクトへの並列なメッセージの委譲を考えることができる。本稿ではこのような複数のオブジェクトへの並列なメッセージの委譲を、or型, and型, seq型の3つに分類する。以下ではこれら3つの委譲方式について説明する。

また注意しておくが、たらい回し/門前払いとand/or/seq型は直交した概念であり、いずれの組み合わせも可能である。

3.1 or型の並列委譲

or型の並列委譲は、複数のオブジェクトに同時にメッセージの委譲を行ない、それらの中で一番最初にメッセージを受理したオブジェクトがその処理を行なう方式である。複数のオブジェクトがそのメッセージを受理可能な場合は、一番最初に受理した(これは非決定的に決まる)オブジェクトがその処理を行なう。

例えば、



という委譲の階層において$O2$と$O5$が(委譲された)メッセージを受理できる場合、$O2$と$O5$の内、先にこのメッセージを受理したオブジェクトが対応するメソッドを実行する[1]。

or型で複数のオブジェクトにメッセージが委譲される場合、その各メッセージの分配毎にシステム全体でuniqueなIDが与えられる。あるオブジェクトが委譲されてきたメッセージを受理可能な場合は、そのIDを基に、そのメッセージに対応するメソッドがまだ実行されていないかどうかを確認した上で[2]、メソッドの実行を行なう。

このようなor型の並列委譲を用いることにより、検索の並列化や非決定性を持つ問題の記述が可能になる。

Delegation with Object-Reconfiguration and Concurrency
Takada, Toshihiro
NTT Basic Research Laboratories

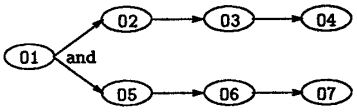
[1] メソッドを受理するか否かを判定する条件部の処理にかかる時間や、オブジェクトが分散環境上に展開されている場合の通信遅延を考えると、必ずしも$O2$が先にメッセージを受理するとは限らない。

[2] この確認をどのような方法で行なうかは実現上の課題である。

3.2 and 型の並列委譲

and 型の並列委譲は、**複数のオブジェクトに同時にメッセージの委譲を行ない、そのメッセージを受理したオブジェクトは全てその処理を行なう方式である。**

例えば、



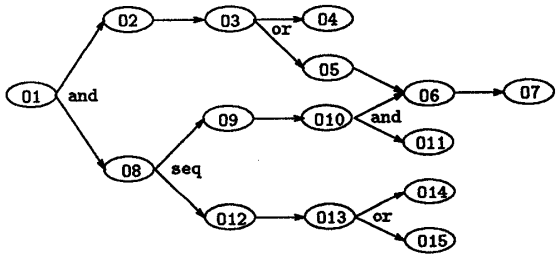
という委譲の階層において $O4$ と $O7$ が(委譲された)メッセージを受理できる場合、$O4$ と $O7$ の両方のオブジェクトでそのメッセージに対応するメソッドが実行される。and 型の並列委譲を用いることにより、複数の並列に動作する部品から構成されるオブジェクトの記述が可能になる。このような並列性を持つ部品の合成を継承を用いて行なうことは通常困難である。

3.3 seq 型の委譲

seq 型の委譲は、1つのメッセージが委譲によって複数のオブジェクトで実行される点では and 型と同じだが、その実行が逐次的に行なわれるという点で異なるものである。メッセージを委譲されるオブジェクト間でその実行の順序が意味を持つ場合、このような逐次実行の仕組みが必要となる。

4 委譲の階層の半順序構造

ここまでに示した複数のオブジェクトへのメッセージの委譲の機構は、下図に示すように自由に組み合わせて使用することができる。図中のオブジェクト $O6$ のように、複数の経路を辿ってメッセージが委譲されて来ることもあり得る。このように、あるオブジェクトへ同じメッセージが2回以上委譲されてきた場合も、メソッドの実行は1回だけ行なわれるものと規定する。



また本稿における委譲機構では委譲の階層構造がループを持つことは禁じられる。すなわち、オブジェクト O からオブジェクト O' へメッセージ委譲のリンクが張られていることを

$$O \succ O'$$

と表す時、関係 $\succ$ はシステム全体でオブジェクトに半順序関係を与えるものとする。

委譲のリンクの関係は、

- オブジェクトが新たに生成される時
- 既に存在するオブジェクトの委譲先を変更しようとした時

に動的にチェックされ、もし半順序関係を満たさないリンクを生成しようとした時は、システムが警告を行ない、オブジェクトの生成あるいは委譲先の変更は行なわれない。

5 実現

現在 TAO [TOO86] の process と object を用いて並列オブジェクト指向システムを試作中である。この試作システムに、本稿で述べた委譲の機構を入れる作業を行なっている。

6 課題

本稿で述べた委譲機構にはいくつかの課題が残されている。

- **並列委譲によるオブジェクトの並列実行の制御**
  現在、並列委譲によって生じるオブジェクトの並列実行部分の制御は、and/or/seq 型の組み合わせによるしか方法がない。しかし共有資源の排他制御など、より複雑な構造を記述しようとする場合は、委譲先のオブジェクト間に制約を導入するなどの、より高度な制御方法が必要となるであろう。
- **委譲階層の半順序関係の保持機構**
  委譲階層の半順序関係を保持するための効率の良いチェック機構が必要である。現在は、実際にリンクを辿り既にそれらのオブジェクトの間に逆向きのパスが存在するかどうかをチェックする、という方式をとっている。
- **委譲階層の探索の効率的手法**
  委譲を用いて動的にオブジェクトの階層構造を変更していく場合、継承におけるメソッドキャッシュのような手法は使い難い。委譲階層の探索の効率的な手法が必要である。

7 まとめ

本稿では、オブジェクト指向言語におけるメッセージ委譲機構の拡張とその制御機構として、

- たらい回しと門前払いの2つの委譲機構
- and 型, or 型, seq 型の並列委譲機構

を提案した。その目的は、

- オブジェクトの性質の動的変更への対応
- オブジェクトの実行時の並列性の向上

などの点にある。

参考文献

[Agh87] G. Agha. Actors: A Model of Concurrent Computation in Distributed Systems. MIT Press, 1987.

[Alm89] J. Almarode. Rule-based delegation for prototypes. In OOPSLA '89, pp. 363–370. ACM, 1989.

[Lie86] H. Lieberman. Using prototypical objects to implement shared behavior in object-oriented systems. In OOPSLA '86, pp. 214–223. ACM, 1986.

[MR89] N. H. Minsky and D. Rozenshtein. Controllable delegation: An exercise in law-governed systems. In OOPSLA '89, pp. 371–380. ACM, 1989.

[TOO86] I. Takeuchi, H. G. Okuno, and N. Ohsato. A list processing language TAO with multiple programming paradigms. New Generation Computing, Vol. 4, No. 4, pp. 401–444, 1986.