黒板モデルを用いたオブジェクト協調の実現方法

2J−3

小野寺　浩　　川幡　和利

富士通エフ・アイ・ピー株式会社

1.はじめに

オブジェクト指向パラダイムが、システム開発の各段階で適用されている。それが広く用いられるようになってきたのは、そもそも実世界をコンピュータ上に写像するうえで、そのモデル化に適している場合が多い、ということが大きな要因のひとつである。

しかし、問題の分析・モデル化の段階で、オブジェクト指向システムとしての恩恵を十分得られるようにオブジェクトやその属性を決定することは、さほど容易なことではない。Coad-Yourdon法やOMTなど、いくつかのオブジェクト指向分析手法が提唱されているが、それらは、決められた手順に従えば結果が得られるようなものではなく、分析に必要な個々の作業は互いに依存関係にあるうえ、それらを並列的に処理しなければならないが、これらはすべて分析者自身の手に委ねられている。[1]

こうした作業を支援するシステムの構築では、実行中に大量に生ずる個々の作業間の相互依存関係をうまく扱う必要があるが、全体を見渡して管理することのできる複雑さには限界がある。そこで、近年注目されている協調アーキテクチャの手法を適用して実現を試みた。協調アーキテクチャでは、特定の能力をもち自律的に動作可能なプログラム単位（エージェント）を多数集め、それらが互いに協調（競争・折衝・協力）することにより処理が行われる。[2]

本稿は、このエージェントに相当するものとして、独立動作可能なオブジェクトを用い、オブジェクト間の協調を黒板モデルを用いて実現する方法について論ずる。なお、実装は、富士通S-4/2(SunOS-4.1.1-JLE1.1.1RevB)上に、記述言語はAT&T C++(Ver.2.1)を用いて行った。

2.協調システムとしての要件

まず最初に、協調を実現するシステムとして備えるべき要件を整理した。

A mechanism for cooperation in object-oriented system
Hiroshi ONODERA, Kazutoshi KAWAHATA
Fujitsu Facom Information Processing Corp.

(1)エージェント
実際の処理を行うプログラム単位であるが、自律性が必要であることから、システム全体の制御とは独立に、単独で動作できる必要がある。

(2)協調相手の探索
協調システムでは、他のどのエージェントが自己の作業に関係するのか、あらかじめわからない場合が多い。このため、協調を求めるエージェントは、存在するすべてのエージェントの中から、協調可能な相手を探し出すことができなければならない。

(3)協調作業
具体的にどのような作業を行えば協調といえるのか、またそれらは一般性をもつのか、といった問題に答えるのは難しい。そこで、今回は、いくつかの例題を検討することにより経験的に得られた以下の項目について、エージェント間で段階的にチェックしあうことを協調作業としてとらえ、その実現を目指すこととした。

Step 1: 静的に設定した実行優先度のチェック
Step 2: 実行履歴のチェック
Step 3: 制約条件のチェック
Step 4: グループ情報のチェック
Step 5: 動的に変化する補助実行優先度のチェック
Step 6: 以上の段階で協調できない場合の利用者の介在

これらの要件をふまえて検討を行った結果、エージェントを独立したUNIX上のプロセスとし、エージェント間の協調に必要な通信のため黒板モデルを適用したうえで、さらにブロードキャスト部とネゴシエーションボード部の2つの黒板領域を用意することで協調相手の探索と協調作業を実現する方法を考えた。
以下、その詳細について述べる。

3.エージェントの実現方法

3.1.独立動作オブジェクトの作成

実装に用いた言語C++は、言語仕様としてオブジェクトの並列動作を記述する機能をもたない。そこで、個々のオブジェクトは、各々main関数をもつ別個のプログラム内で生成されるインスタンスとし、プロセス間通信機能を実現するクラスを利用してオブジェクト間のメッセージ・パッシングを行うよう

にした。つまり、各オブジェクトはUNIX上のプロセスとして独立に動作することになる。

3.2.協調に必要な情報の設定

次に、この独立動作オブジェクトに対し、協調を行う上で使用する以下の情報を追加し、これをエージェントとした。

・静的な実行優先度
　優先度は、エージェント構成の変更に対応できるよう、実数値で設定する
・制約条件
　再実行可能かどうかなどの条件を記述する
・グループ情報
　処理対象や処理内容が似たものをあらかじめグループ化しておき、同じグループどうしでは優先して処理を行おうとするものである
・動的に変化する補助実行優先度
　実行されるごとに重みづけされる優先度である
・利用者の介在する協調のための情報
　利用者が判断するための情報を自然言語で記述する

4.協調相手探索の実現

協調相手の探索は、黒板とコントラクトネットをモデルとして実現した。まず、黒板上のジョブ提示部にジョブという形で協調を必要とする問題が提示される（図1の①）。他の独立動作オブジェクト（エージェント）は、定期的にこのジョブ提示部を参照しており（同②）、提示されたジョブが自分の局所的評価により協調可能であると判断したら、黒板上の入札部へ入札を行う（同③）。ジョブ提示を行ったエージェントは、入札部を参照することにより、結果的に協調相手のエージェントを知ることができる（同④）。なお、ジョブの局所的評価としては、実行優先度・実行履歴・制約条件のチェックを行っている。

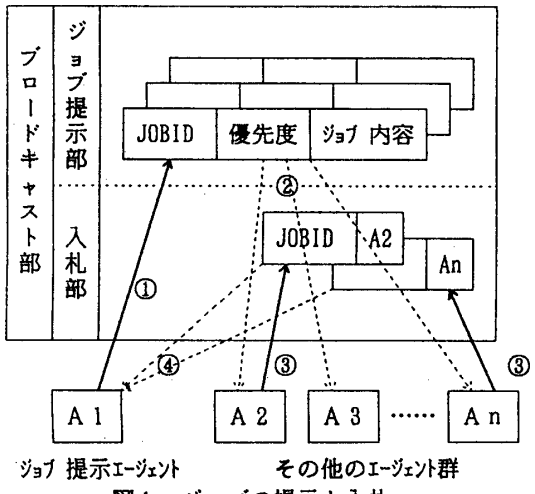

図1　ジョブの提示と入札

5.協調作業の実現

独立動作オブジェクト間で、協調に必要な情報を交換しあうのも、やはり黒板を用いて実現した。まず、協調のための情報を黒板に出力し（図2の①）、次に協調相手の独立動作オブジェクトへシグナルを送る（同②）。シグナルを受信した相手は、自分あてのメッセージを黒板上からピックアップし（同③）評価（先に述べた協調作業の項目のチェック）を行う。

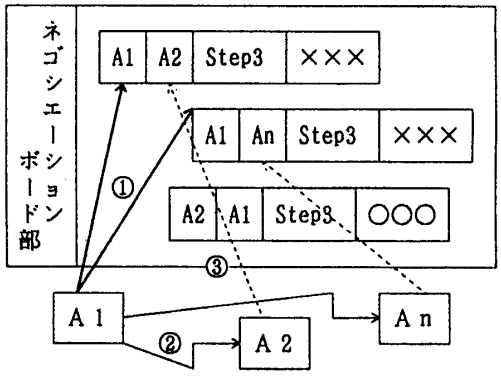

図2　協調に必要な情報の交換

6.黒板の実装

複数の独立動作オブジェクトから非同期にアクセスされる黒板は、メッセージを実際に書き込むための共用メモリと、排他制御用のセマフォを用いて実装した。2個のセマフォをペアで使うことにより、参照のみのアクセスについては複数のオブジェクトの同時参照を可能としているほか、更新アクセスの実行が、他の参照アクセスの実行に追い越されないようにしている。[3]

7.評価

オブジェクトを独立したプロセスとしたことで、情報の隠蔽度はかえって高まる結果となった。また協調のために必要な情報はすべて一旦黒板上に出力されるため、これをモニタすることによりシステム全体の動作状況を把握するのに便利であった。

なお、本稿で述べた実現方法は、目的や用途を限定したものではないため、一般的な協調問題解決システムへの応用が可能であると考えている。

8.参考文献

(1)J.Rumbaugh他：オブジェクト指向方法論OMT，トッパン，1992
(2)中島他：協調アーキテクチャ関連研究の現状，電子技術総合研究所調査報告，221，1991
(3)富士通：SX/Aカーネル説明書，1989