3K－1　オブジェクト指向による アーキテクチャ評価シミュレータ設計手法の検討*

鈴木研司†　本村真人　井上俊明　小長谷明彦
NEC 情報システムズ†　NEC

1　はじめに

オブジェクト指向とは、対象を”もの (＝オブジェクト)”として見なす考え方であり、複数のオブジェクトを連係させることでシステム全体を構成する設計手法をオブジェクト指向設計と呼ぶ。オブジェクト指向設計は、オブジェクトが内蔵する属性と機能の独立性を高める設計方法であるため、システムの進化に伴って生じるオブジェクト単位の変更に対して対処が容易な設計手法だと言われている。

今回、我々は新プロセッサアーキテクチャのパイプラインアーキテクチャシミュレータ (PA シミュレータ)[1] を作成するにあたり、オブジェクト指向設計手法の適用を試みた。一般に、設計途上のプロセッサアーキテクチャは、設計の変更 / 詳細化などにより絶えまなく変化し続ける。オブジェクト指向設計手法により、このような変化に応じて柔軟にその内部構造を進化させ続けることが可能な PA シミュレータの実現が期待される。

本稿では、オブジェクト指向設計手法による PA シミュレータ設計における 3 つの問題を論じ、それぞれの解決方法について報告する。

2　オブジェクト指向シミュレータ

はじめに、アーキテクチャモデルとシミュレータの関係を図 1に示す。シミュレータのエンジン部は、アーキテクチャの概念ブロックを基に、必要に応じて適度に抽象化したアーキテクチャオブジェクトによって構成されている。このようなオブジェクトを組み合わせてシミュレータを構築するには、ハードウェアの特性とオブジェクト指向とをうまく親和させる必要がある。以下では、その一般的な問題点と本シミュレータにおける解決方法を述べる。

*An Object-Oriented Design Approach for Architecture Simulator Development
†Kenji SUZUKI, Masato MOTOMURA, Toshiaki INOUE, Akihiko KONAGAYA
†NEC Informatec Systems, NEC Corporation

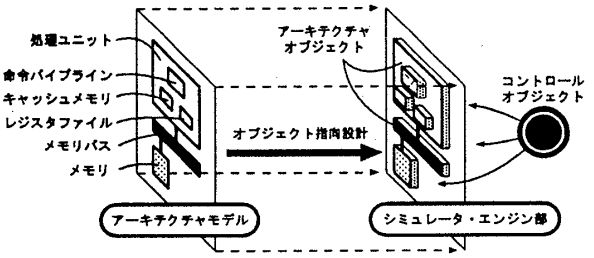

図 1: アーキテクチャモデルとシミュレータ: 概観

2.1　複雑なモデル構造の階層的抽象化表現

複雑なアーキテクチャモデルの概念ブロックを直接オブジェクトに割り当てると、オブジェクト構成が複雑になり、システムの進化の妨げになりやすい。この問題に対し、オブジェクトを階層的にまとめ、抽象化したオブジェクトとして扱うことで複雑性を隠蔽できる。さらに、階層間の独立性を高めるために階層間通信プロトコルを設定する。このプロトコルに従い階層間通信を行うオブジェクトは自由に置換えられ、その再利用性が高まる。

2.2　複雑な内部状態を持つオブジェクトの表現

アーキテクチャモデルには複雑な内部状態を持つ概念ブロックが多い。それをオブジェクトに直接埋め込むと内部が複雑になり、保守性が低下しやすい。そこで、オブジェクトに内部の状態遷移を管理する抽象オブジェクトを導入し、状態遷移をスマートに表現する手法を提案する。

状態遷移管理オブジェクトは、図 2(a) のようにターゲットのオブジェクトに埋め込まれ、常に目的

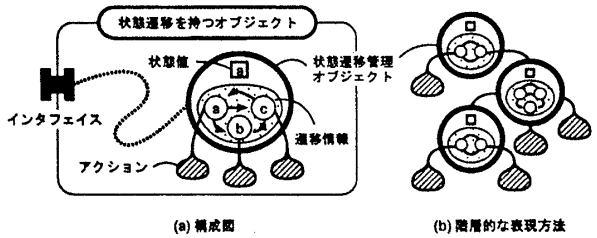

図 2: 状態遷移管理オブジェクトによる表現

のアクション (= 状態に対応する機能) を指し示す役割を果たす。状態遷移管理オブジェクトは遷移情報をデータ構造として内蔵し、図 2(b) で示されるように、複雑な状態遷移を階層的で単純な状態遷移の組合せとして表現できる。これにより、全体像を把握しやすくなり、保守性が向上する。

2.3 オブジェクトの並列性の表現

ハードウェア上で並列に動作する概念ブロックを直接アーキテクチャオブジェクトにマッピングし、同等の動作を実現するためには、オブジェクトを並列に動作させる必要がある。本シミュレータでは、逐次処理型プログラミングの上でオブジェクトを並列に動作させるために、コントロールオブジェクトと呼ぶ抽象オブジェクトを導入した。

コントロールオブジェクトは、図 3に示したランクと呼ぶ配列構造によってクロックサイクル以下の微小時間を表現し、適切なランクにある被制御オブジェクトを起動、微小動作させた後、次の起動ランクへ遷移させる。このような微小動作を繰り返し、疑似的にオブジェクトの並列動作を実現する。

また、図 1に示したように、コントロールオブジェクトとアーキテクチャオブジェクトは独立している。従って、並列性を伴うオブジェクトの動作とオブジェクトの構造は、独立して進化できる。

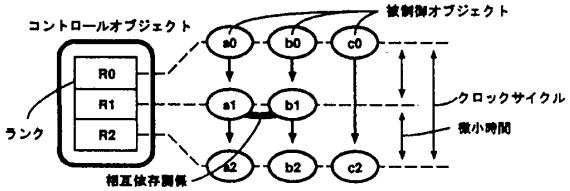

図 3: コントロールオブジェクトによる並列性の表現

3 PA シミュレータの実現

ここでは、前節で検討した手法を適用して構築した本シミュレータについて、具体的部分を述べる。

3.1 命令パイプラインの表現方法

本シミュレータでは、命令パイプラインの動作を図 4(a) のような複数のオブジェクトの組合せによって実現した。ステージオブジェクトは、パイプラインステージの資源と機能を抽象化したオブジェクトである。各ステージオブジェクトは、クロックサイクル以下に起こる相互依存関係を正しく表現するために、コントロールオブジェクトのランク制御を受けて並列動作する。命令パイプラインオブジェクトは、その動作結果を監視し、パイプライン上の命令流の制御を行う。このような二重の制御によって、パイプライン動作が表現されている。

3.2 メモリ操作オブジェクトの階層的表現方法

本シミュレータにおけるメモリ操作は、階層構造を利用して抽象化され、上位階層の命令パイプラインオブジェクトと下位階層のメモリ操作代行オブジェクトと呼ぶ抽象オブジェクトとの簡単な通信として実現されている。その概念図を図 4(b) に示す。

命令パイプラインオブジェクトは、メモリ操作プロトコルを用いてメモリ操作代行オブジェクトに目的の操作を要求し、メモリ操作代行オブジェクトは要求に従い、適切にキャッシュメモリやメモリバスを制御し応答する。メモリ操作代行オブジェクトの複雑な内部制御には状態遷移管理オブジェクトを用いて、状態遷移を階層的に表現している。

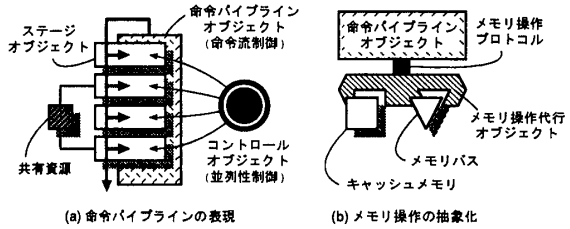

図 4: シミュレータの実現方法

4 おわりに

概念ブロックの組合せとして捉えたハードウェアは、オブジェクト指向設計手法を適用しやすい対象と考えられる。ところが、ハードウェアの表現をオブジェクトに直接埋め込むと、ハードウェアの特性によってオブジェクト内部が複雑化し、組織的なオブジェクト構成が困難になりやすい。

前述のいくつかの手法は、オブジェクトによるモデリングの自由度を広げ、オブジェクト構造の複雑化を回避するのに効果的である。これらの手法をシミュレータ設計に導入することにより、ハードウェアの特性とオブジェクト指向の親和性を高め、双方の特徴をうまく調和することが可能になった。この結果、本シミュレータは、アーキテクチャの変更 / 詳細化に応じて、柔軟かつ効率的にシステムを進化させることが可能となり、実際にこれらの手法が有効であることを確認することができた。

参考文献

[1] 本村ほか, "順序付きマルチスレッド実行モデルの提案とその評価", JSPP95, pp.99-106, May 1995.