

分散オブジェクト基盤への広域キャッシュ機構の適用*

5 T-4

柏木 伸一郎† 犬束 敏信† 浜田 雅樹†

NTT ソフトウェア研究所‡

1 はじめに

広域ネットワーク環境には、高価な通信コストや負荷の集中など、様々な問題が存在する。これまでは、WWW におけるキャッシュのように、アプリケーション毎に独自の問題解決を図っていた。

一方、これらの問題の多くを、分散オブジェクト技術を利用することにより、アプリケーションやプロトコルに依存しない形で統一的に解決できる可能性がある。しかし、現在の分散オブジェクト通信基盤では、広域環境において必要となるキャッシュや複製の管理機構を提供しておらず、広域ネットワークアプリケーション開発にはあまり有効でなかった。

本稿では、広域ネットワークアプリケーションに共通して求められるキャッシュ・複製管理機構を、分散オブジェクト基盤に組み込むことを提案する。

2 分散オブジェクトの広域利用

多くのネットワークアプリケーションが、分散オブジェクト技術を用いて構築されつつある。分散オブジェクトを用いることにより、クライアントやサーバの種類や位置に依存しないソフトウェアを開発できる。

しかし、これらのアプリケーションは、イントラネットなど、比較的均一なネットワーク環境で稼働するものが多い。広域ネットワークで不特定多数のユーザにサービスを提供するアプリケーション（以下、広域アプリケーションと呼ぶ）を、分散オブジェクトを用いて構築することはまだあまり行われていない。その原因として、広域ネットワーク環境の特性が考えられる。

分散オブジェクトは、位置独立な通信手段を提供することによりネットワークの存在を隠蔽する。LAN など、比較的均一なネットワーク環境では、ネットワークポロジを考慮せずにアプリケーションを設計しても大きな支障は生じない。

しかし、広域ネットワークでは、高価な通信コストや、スケーラビリティの問題などがあり、設計段階からネットワークポロジを考慮に入れて最適化を図ることが必要となる。そのため、分散オブジェクトによる位置独立性のメリットを十分に生かすことができない。

ネットワーク特性に対応するための最適化手法をどのように実装するかは、依然としてプログラマに任されている。

最適化手法の一例として、キャッシュを考える。同一資源へ頻繁なアクセスが生じる場合、キャッシュによりアクセスコストを低減させることができ、既存の広域アプリケーション（WWW など）では広く使われている。しかし、これらの広域キャッシュ機構はアプリケーション毎に独自に構築されており、広域ネットワーク環境に共通する多くの問題を個別に解決しているのが現状である（図1）。複数の転送プロトコルをサポートする広域キャッシュシステムも存在するが[1]、新たなアプリケーションへの対応は容易ではない。

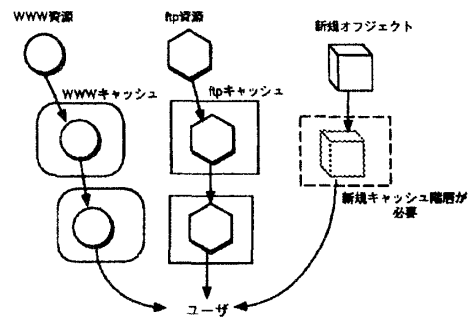


図1: アプリケーション毎の広域キャッシュ機構

分散オブジェクトで構成されるソフトウェアでも、物理的に分散されることにより、アプリケーションの一部の構成要素間で頻繁な通信が生じることが予想される。キャッシュや複製の設置など、広域ネットワーク特有の最適化手法を分散オブジェクト基盤において実現できれば、より位置透過性の高い広域アプリケーションの構築が可能になる。

3 広域キャッシュ機構

本稿では、分散オブジェクト基盤の中に広域キャッシュ機構を組み込み、アプリケーションからオブジェクトのキャッシュ・複製管理を分離することを提案する（図2）。

* "Global Object Cache in Distributed Object Platforms"

† Shun-ichiro KASHIWAGI, Toshinobu INUZUKA, Masaki HAMADA

‡ NTT Software Laboratories

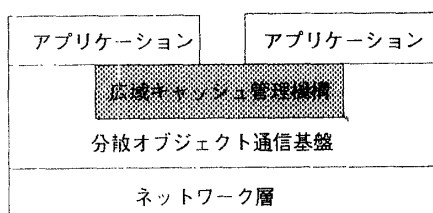


図 2: 分散オブジェクト基盤上のキャッシュ機構

これまで、アプリケーションを構成する複数のオブジェクトやプロセスは、多くの場合、手でそれらの複製が物理的に配布されていた。WWW キャッシュサーバや電子メールサーバなどがその一例である。また、アプリケーションが扱うデータのキャッシュも独自に行っていた。

今回提案する広域キャッシュ機構は、分散オブジェクト基盤において共通の機能として提供されるため、アプリケーション毎のサーバ分散やキャッシュ管理等は不要になる。同じ機能を持つ複製オブジェクトは、アプリケーションから見た場合は論理的に1つのオブジェクトとなる。実体は各ホストに必要に応じて転送され、従来と同様にこれらの複製が実際のサービスを提供する。

論理実体を1つとして、物理実体を透過的に分散させることにより、(a) キャッシュ管理をアプリケーションの論理記述から分離できる、(b) アプリケーションが動作する実際のネットワーク環境へ適応しやすいソフトウェアを作成できる、(c) アプリケーションの配布や更新が容易になる、などの利点が生じる。

4 実現に必要な機能

我々は現在 Java 言語上の分散オブジェクト環境を実装の対象としている。

Java を選択した理由は、実行環境が広く普及しており広域利用に適していることや、プログラムの可搬性が高く、オブジェクトの移動や複製を実現しやすい環境であることなどである。

広域キャッシュ機構を実現するために必要な機能として、以下のものが考えられる。

- オブジェクトの転送
- キャッシュ管理
- キャッシュへの透過的なアクセス機構
- 並行性制御

オブジェクトの転送 オブジェクトのキャッシュを行うための転送機構が必要となる。Java 上の分散オブジェクト通信基盤である Java RMI [2] や HORB [3] では、インスタンスの転送機能を提供しているが、クラスの転送は提供されていない。

オブジェクトのキャッシュを持つサーバ間でクラスを共有するために、何らかのクラスレポジトリ機構を用いる。転送したいオブジェクトのクラスが手元に存在しない場合は、クラスファイルも自動的に転送する。Java 仮想機械はクラスの動的読み込み機能を持っているので、オブジェクトの実行時転送が可能になる。

キャッシュ管理 キャッシュされたオブジェクトの一貫性管理を行う。アプリケーションが必要とするキャッシュポリシーは様々であり、キャッシュ方式もオブジェクトによって異なると考えられる。これらを細かく指定することを可能にする、キャッシュ管理機構インターフェースが必要である。

キャッシュへの透過的なアクセス機構 キャッシュされたオブジェクトは、分散オブジェクト基盤から見れば異なる実体であるが、アプリケーションからは同一に見える必要がある。そのため、透過的なアクセス機構が必要となる。HORB [3] ではローカルとリモートのオブジェクト参照間での高い透過性を実現しているが、複製オブジェクトへの透過的な参照はまだ提供されていない。

並行性制御 キャッシュされたオブジェクトは、各ホストにて独自のスレッドを持つことになるので、複製の数だけ潜在的な並行性が増すことになる。アプリケーションのセマンティクスを保つため、同期や排他制御など、並行性を制御する機構が必要である。

5 おわりに

本稿で提案した広域オブジェクトキャッシュ機構により、通信コストの高い広域ネットワーク上で、効率的に動作するアプリケーションを分散オブジェクトを用いて構築することが容易になると考えている。

現在、Java 分散オブジェクト基盤上に本システムの実装を進めている。今後は、既存の広域アプリケーションを本システム上で動作させ、定量的な実験を通じて有効性を検証していく方針である。

参考文献

- [1] Squid Internet Object Cache, <http://squid.nlanr.net/Squid/>
- [2] Java RMI - Remote Method Invocation, <http://chatsubo.javasoft.com/current/>
- [3] Satoshi Hirano, "HORB: Distributed Execution of Java Programs", Proceedings of the *International Conference on Worldwide Computing and Its Application (WWCA '97)*, Tsukuba JAPAN, March 10-11, 1997.