

INTERSYSTEMS  
WHITE PAPER

## INTERSYSTEMS CACHE

インメモリデータベースに代わる  
ソリューション

InterSystems  
CACHE<sup>®</sup>

David Kaaret  
InterSystems Corporation

## INTERSYSTEMS CACHÉ

### インメモリー データベースに代わるソリューション

#### はじめに

単独のコンピュータで稼動しているアプリケーションから、相互接続された大規模グリッドまで、従来のリレーショナル データベース アプリケーションのパフォーマンス上の制限の克服には、インメモリー データベースを使用してデータ アクセスを高速化することが一般的です。インメモリー データベースとキャッシュ製品を使用することで確かにスループットは上がりますが、一方で大量のデータの取扱い、過大なハードウェア要件、スケーラビリティの面での制約など、数多くの制限が生じます。

InterSystems Caché® は、インメモリー データベースを一般的に使用しているアプリケーションに最適な、独自のアーキテクチャをもつ高性能オブジェクト データベースです。Caché は、インメモリー データベースに匹敵するパフォーマンスを発揮するだけでなく、次のような特長を持っています。

- コンピュータの電源が落ちた場合やクラッシュした場合でもデータ損失がない永続性
- 大規模なデータ セットに対する高速アクセス
- 数百台のコンピュータ、数万人のユーザーに対応できるスケーラビリティ
- SQL と、Java、C++、.NET などのオブジェクトからの同時データ アクセス

このホワイト ペーパーでは、大量のデータに高速にアクセスを必要としている企業にとって、Caché がインメモリー データベースに代わる有力なソリューションである理由について説明します。

#### 独自のデータ エンジンで永続性と高性能を実現

Caché は永続性を備えたデータベースです。つまり、RAM に保持されているデータをバックグラウンド プロセスでディスクに書き込んでいます。一方、インメモリー データベースでは、データを定期的に何らかのデータ 記憶装置に書き込んでいるにすぎません。では、なぜ Caché が、このようなインメモリー データベースに匹敵するパフォーマンスを発揮できるのでしょうか。

この理由の一つは、Caché が持つ独自のアーキテクチャにあります。従来のデータベースに見られる表形式によるデータ構造に代わり、Caché ではオブジェクト定義に基づくデータ構造を持つ多次元配列を採用しています。このアーキテクチャでは、アーキテクトが設計した方法でデータを保存し、かつ、インメモリー キャッシュと同じデータ構造をディスク上で使用します。また、まとめて保存する必要があるデータは、まとめて保存します。この結果、Caché はディスク上のデータにきわめて高速にアクセスできるようになっています。

多くの分散キャッシュ製品では、データを更新する際に、複数のインメモリー キャッシュの同期を必要としますが、これがパフォーマンス低下の原因にもなっています。Caché では、

データの更新処理と、キャッシュへのデータ配信処理が論理的に独立しています。これにより、ワークフローをはるかに簡素化でき、高いパフォーマンスが実現します。

Caché では、C++ および Java 向けに "インプロセス バインディング" という機能も備えています。この機能により、C++ または Java で記述したアプリケーション、Caché の内部データ構造に直接実装することができます。

## 永続性の利点

Caché は、インメモリー データベースと同等のパフォーマンスを発揮し、さらにディスク上のデータへのアクセスに関しては、インメモリー データベースをはるかにしのぐ利点があります。何よりも明らかな利点は、永続的なデータ 記憶装置を別途必要としないことです。Caché 自体が永続記憶装置であり、常に最新のデータを保持しています。コンピュータの電源が落ちた場合やクラッシュした場合でもデータの損失が発生しません。

Caché のその他の利点として、利用できる RAM の大きさとデータ セットのサイズの制限がないことが挙げられます。データがローカル キャッシュに存在しない場合は、リモートキャッシュまたはディスクからシームレスにデータを取得できます。Caché ベースのシステムでは、RAM の制限を受けないことから、ペタバイト クラスのデータ サイズを扱うことができます。これは、インメモリー データベースでは不可能なことです。

処理容量を上げるためにシステムに RAM を増設するには、ディスク ストレージ増設よりもコストがかかります (テラバイト クラスのディスク ストレージは、同クラスのサイズの RAM よりも安価です)。さらに、多くのインメモリー システムでは、コンピュータのクラッシュによる障害の対策として、データの冗長コピーを別のコンピュータに保持する必要があります。Caché のような永続データベースによる分散キャッシュ システムを運用することで、多くの場合はハードウェア コストの削減が実現します。

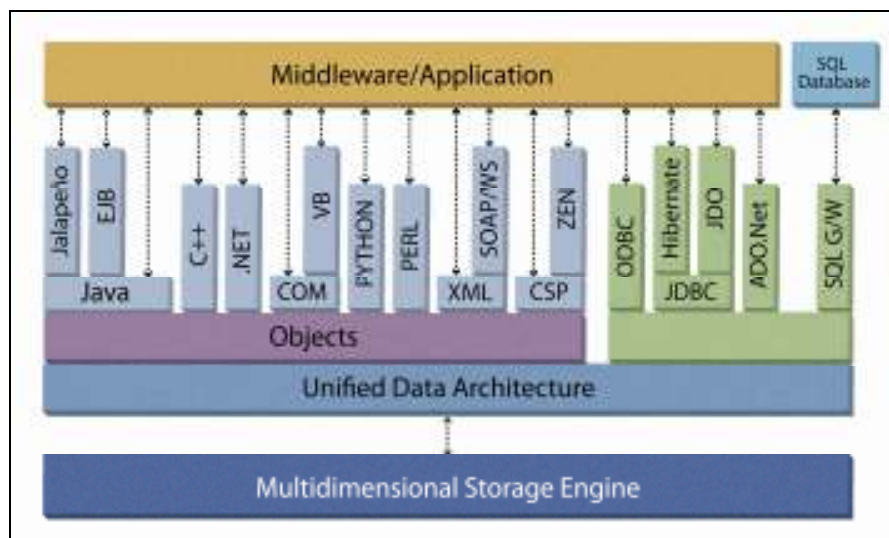
## SQL とオブジェクトによるシームレスなデータ アクセス

ほとんどのインメモリー データベースは、高速処理に最適化したデータ構造を採用していることから、共通の問題として、多くの場合、SQL によるデータ アクセスが困難であることが挙げられます。大半の分析ツールやレポート ツールでデータを使用できるようにするには、まずデータをリレーショナル テーブルに "マッピング" する必要があります。このマッピングは通常、インメモリー データベースから永続データ 記憶装置にデータを転送するときに行われ、一般的には、ETL (抽出、変換、ロード) プロセスを必要とします。このマッピングには、処理オーバーヘッドと余分な時間が発生し、これが、きわめて高速な分散アプリケーションにとって、リレーショナル データベースの処理速度が不十分あり、代わりにインメモリー データベースが多用される理由にもなっています。

リレーショナル モデルに基づいて構成され、SQL によるデータ アクセスを可能にしているインメモリー データベースはわずかです。しかも、このようなシステムは、アプリケーション開発で広く利用されているオブジェクト指向テクノロジーによるデータ アクセスが困難であるという、逆の問題を抱えています。さらに、大半のリレーショナル インメモリー データベースでは、複数台のコンピュータによる構成が考慮されていません。1 台のコンピュータでのみ実行可能であり、その結果 RAM による制限が発生します。

Caché はこの点が大きく異なります。Caché が採用している多次元配列は、リレーショナルテーブルとしての利用とオブジェクトとしての利用が同時に可能です。Caché の 統一データアーキテクチャでは、データオブジェクト とリレーショナル テーブルの両方で同時に参照することがききます。当然のことながら、ビュー同士のマッピングは不要です。

Fig. 1: 複数の方法によるデータ アクセスを 統一データアーキテクチャで実現



Caché のSQLは、ODBC と JDBC をサポートしています。オブジェクトとしての扱いでは、Java、.NET、C++ など、ほとんどのオブジェクト指向言語とのバインディング機能を提供しています。Caché では、全てのオブジェクト表現を提供し、継承、多態性、カプセル化などのオブジェクト指向概念がサポートされています。

## ECP(Enterprise Cache Protocol)

マルチ コンピュータ アプリケーションで使用する Caché では、ECP(Enterprise Cache Protocol)を使用して、自動的にキャッシュを保持します。

ECP を使用すると、Caché のインスタンスをデータ サーバーとアプリケーション サーバーの両方、あるいはどちらかに構成することができます。各データはすべてデータ サーバーで保有し、アプリケーション サーバーではデータが存在する場所を把握して、最近使用したデータをローカル キャッシュで保持します。このローカル キャッシュにあるデータだけでは要求を満たせない場合、アプリケーション サーバーは、必要なデータをリモート データ サーバーに要求します。キャッシュの整合性は ECP によって自動的に管理されています。

ECP の利用にあたってアプリケーションの変更は不要です。アプリケーションでは、データベース全体をあたかもローカル データベースであるかのように扱うだけです。分散キャッシュ システムには、クエリを投げる前に、データを必要とするクライアントそれぞれが、目的のデータのサブセットを指定することが必要なものがあり、この点が Caché と分散キャッシュシステムとの大きな違いとなっています。

## 1 台のコンピュータに 1 つのキャッシュ

Caché と他の分散キャッシュ製品との大きな相違点として、これまで紹介したもののほかに、ほとんど分散キャッシュ製品では、コンピュータ上で実行しているプロセスごとに別々のキャッシュを保持していることが挙げられます。たとえば、ある 1 台のコンピュータに 8 つのクライアントが存在すると、その 1 台のコンピュータは 8 つの独立したキャッシュを保持することになります。

一方、Caché では共有メモリーにキャッシュを保持し、それぞれ専用のメモリー アドレスで実行されているプロセスからキャッシュのデータにアクセスするバインドを提供します。データにアクセスする手段として、JDBC のような TCP ベースのプロトコルや言語バインディングがあります。また、アプリケーションからキャッシュを直接操作できるようにするバインドもあり、この方法ではきわめて高いパフォーマンスが得られます。これらの手段により、データに同時にアクセスできます。

1 つのキャッシュを複数のクライアント間で共有可能にすることにより、多くの利点が得られます。その 1 つとして、共有キャッシュ システムでは、必要とするメモリー量が少なくて済むという点が挙げられます。複数のクライアントから同じデータに対して重複してアクセスが発生することは十分に考えられますが、Caché 以外の分散キャッシュ製品では、データのコピーを複数保持してこのような状況に対応しています。一方、Caché では、データのコピーをコンピュータごとに 1 つだけ保持するだけです。

コンピュータごとにキャッシュを 1 つにすることで、ネットワーク I/O の減少という結果をもたらします。高性能なアプリケーションでは、キャッシュの保持に伴うネットワーク トラフィックが大きな問題になることがあります。それでも、コンピュータが保持するキャッシュが 1 つであることから、元のデータに変更が発生しても、更新が必要になるキャッシュは 1 つのみで済むという利点があります。一方、キャッシュが複数存在する場合は、同じ更新処理を重複して実行する必要があります。

マルチ コア プロセッサのコンピュータであっても、Caché ベースのシステムで 1 台のコンピュータは、使用する共有キャッシュは 1 つだけです。これにより、他の分散キャッシュ製品に比べ、優れたスケーラビリティが得られます。たとえば、8 コアのコンピュータ 250 台で構成した Caché ベースのシステムの場合、整合性を維持するには、250 個のキャッシュどうしでのみ通信すれば良いことになります。一方、コアごとに別々のキャッシュを必要とするシステムでは、2,000 個ものキャッシュを調整する必要があります。最新型のコンピュータではコアの数が 8 個から 16 個、さらにそれ以上になることがあるので、スケーラビリティの面で Caché が持つ利点がますます重要になります。

Fig. 2a: インターシステムのエンタープライズ キャッシュ プロトコルを使用しない場合のキャッシュの整合

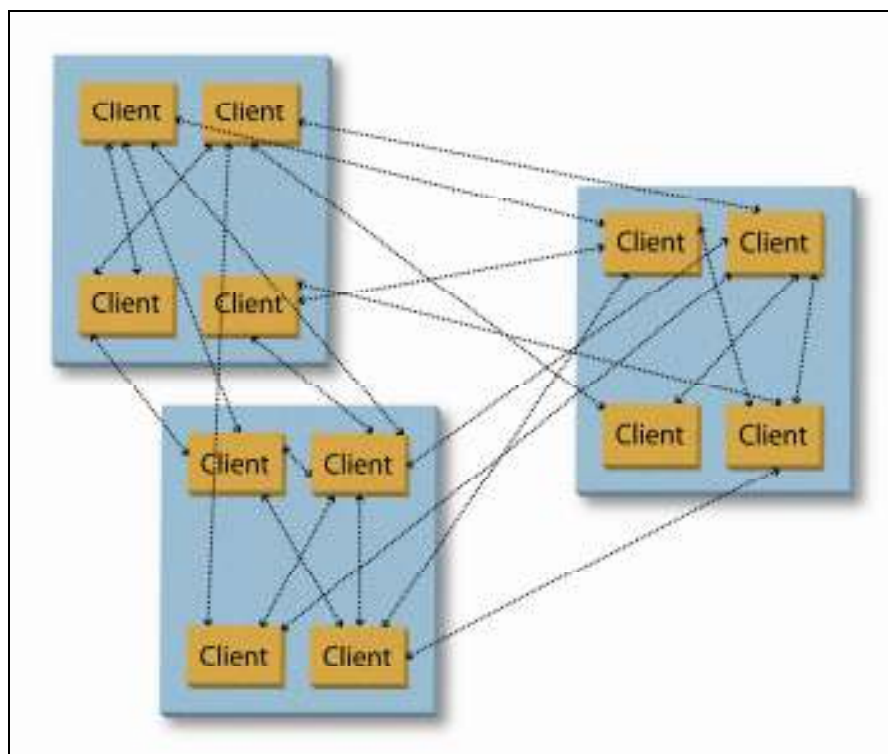
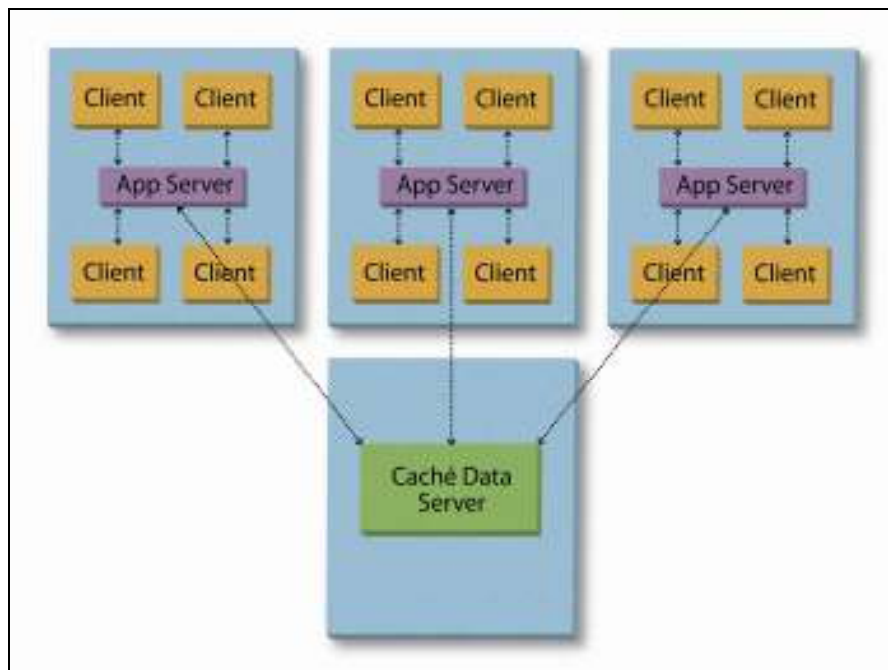


Fig. 2b: Caché ベースのシステムでのキャッシュの整合性



## キャッシュの生成

多くの分散キャッシュ アプリケーションでは、キャッシュのプリロードが長時間を要するプロセスになることがあります。この原因として考えられるのは、データの絶対量が多いこと、またはアプリケーションで使用しているオブジェクト指向構造にリレーショナル記憶装置からデータを "マッピング" する処理に時間がかかることです (この両方が原因になっていることもあります)。データを集中的に処理するアプリケーションの中には、インメモリー キャッシュにあるデータで実際に計算を実行する時間よりも、インメモリー キャッシュにデータを取り込む処理に時間がかかるものもあります。

Caché ではこのようなことはありません。Caché の非常に優れた SQL 機能により、基本的なリレーショナル データ ソースから容易にデータを取得できます。当然のことながら、永続的なデータベースとして、Caché を基本的なデータ ソースとすることもできます。この場合、キャッシュをプリロードしておく必要はまったくありません。クエリが実行されると、ローカルキャッシュでは必要とするデータを自動的にロードします。

別の考慮事項として、キャッシュにデータを取り込む作業に何台のコンピュータが関与するのかという点があります。Caché の場合、分散グリッド環境の中ではわずかな数のコンピュータが、データの基本的な所有権を保有しています。このような環境でデータをキャッシュするには、ECP データ サーバーへアクセスすれば十分です。データサーバーにバックグラウンドでロードしている間に、他のコンピュータでは別の作業を実行できます。アプリケーション サーバーがオンラインになると、データの要求があったときに、そのキャッシュにデータが自動的に再度取り込まれます。

一方、ほとんどのインメモリー製品では、データがロードされると、データが分割されて分散キャッシュに行き渡り、それにより、少なくとも 1 台のコンピュータのメモリーにすべてのデータ (または仮想的にすべてのデータ) が存在するようになります。その結果、わずかな台数のコンピュータにデータをロードしながら、別のコンピュータが必要に応じてオンラインになることは、多くの場合実現不可能です。

## まとめ

インメモリー データベースを使用する最大の理由は処理速度です。インメモリー データベースは、確かに処理速度は速いですが、スケーラビリティの乏しさ、SQL サポートの欠如、過大なハードウェア要件、予期しない停止によるデータ喪失のリスクといった問題が伴います。

Caché は、インメモリー データベースに匹敵するパフォーマンスを実現できる唯一の永続データベースです。きわめて大規模なデータ セットをサポートし、SQL とオブジェクト双方からのシームレスなデータ アクセスを可能にします。また、数百台のコンピュータで構成する分散システムを実現し、高度な信頼性を提供します。

ここまで説明したすべての特長の通り、大規模なデータをきわめて高速で処理する必要のあるアプリケーションにとって、Caché はインメモリー データベースに代わる魅力的なソリューションとなります。

## インターシステムズについて

インターシステムズ社は、ソフトウェアテクノロジーの世界的リーダです。米国マサチューセッツ州ケンブリッジに本社を置き、世界 23 カ国 に拠点を有します。インターシステムズ社は、エンタープライズクラスのアプリケーションの迅速な開発展開と、素早い統合を可能にする先進的な製品を提供しています。InterSystems ENSEMBLE®(アンサンブル)は、迅速なインテグレーションプラットフォームで、新たな機能や連携能力を加えて、アプリケーションを高機能にすることができます。InterSystems CACHÉ®(キャッシュ)は、高性能オブジェクトデータベースで、アプリケーションをより高速にし、拡張性を高めます。InterSystems HealthShare™(ヘルスシェア)は、既存の医療アプリケーションを最大限に利用して、地域や全国の電子医療記録を迅速に構築するプラットフォームです。InterSystems DeepSee™(ディーブシー)は、トランザクショナルシステムにリアルタイムのビジネスインテリジェンスを付加して、業務上でよりよい意思決定を可能にするソフトウェアです。

詳細は、[InterSystems.co.jp](http://InterSystems.co.jp) を参照してください。

インターシステムズジャパン株式会社

〒 160-0023  
東京都新宿区西新宿 6-10-1  
日土地西新宿ビル 17F  
Tel: 03-5321-6200(代)  
Fax: 03-5321-6209

[InterSystems.co.jp](http://InterSystems.co.jp)

INTERSYSTEMS