自律分散 L3 キャッシュの性能優位性を基にした適応アプリケーションへの差別化展開

高橋　宏尚（DTS（株），東京工業大学大学院情報理工学研究科計算工学専攻）　森　欣司（東京工業大学大学院情報理工学研究科計算工学）

概要　近年 Web サービスシステムの形態はインターネットを利用したサービスモデルに移行しており，リアルタイム性のある RTSOAA（Real-Time Service-Oriented Architecture and Applications）型サービスを求められることが多い．RTSOAA はユーザーによる Web サイトからの情報検索とユーザからの Web サイトへの情報発信を，双方向により最小限の遅延時間で実行できる Web サービスの方式である．こうしたサービス方式を実現するための既存技術としては，スケールアウト型の分散ファイルシステムがあるが，その書き込み処理は，リアルタイム性を持たない．このためリアルタイム性が重視されるアプリケーションを既存技術を用いて実現すると，リアルタイム性を確保するためにアプリケーションごとのチューニングが必要となり，結果として特異性が生じて特定顧客，特定システムのみの狭い運用ソフトウェアとなり，汎用性がないものとなってしまう．これに対し，Web サービスにおいて双方向通信とローレーテンシー通信を実現する方式として，自律分散 L3 キャッシュソフトウェアを汎用モジュールとして利用する方法を考えることができる．汎用化された自律分散 L3 キャッシュソフトウェアを Web サービス構築に用いることにより， Web サービスのリアルタイム性を運用アプリケーションの特異性や地域性を生ずることなく実現することができる．

本論文では，L3 キャッシュの概要及び従来の自律分散ノードと比較した性能を示すとともに，アプリケーション適用における性能優位性による差別化を意識した世界市場への展開方法について，具体的な事例を紹介する．

1. はじめに

1977 年に日本で開発された自律分散システム[1] [2] は，350 以上の特許が出願され国内での実用化も多い．代表的な例としては世界最大の情報制御システムである JR 東日本の東京圏輸送管理システム ATOS (Autonomous Decentralized Transport Operation System)，川崎製鉄における鉄鋼プラント・システム，ブリヂストンの生産管理システム FOA(Flow Oriented Approach)，朝日新聞社の新聞生産工程管理システムなどがある．一方，海外を中心に近年のサービスシステムの形態もインターネットを利用した Web サービス型に移行され，特にリアルタイム性が常時求められる RTSOAA (Real-Time Service-Oriented Architecture and Applications)[3] 型サービスの需要が急激に高まっている．RTSOAA 型の Web サービスの代表例としてはヘルスケアでの Web サービスや，各種のサイバーフィジカルシステムなどをあげることができる．Web サービスのリアルタイム性に関しては，例えば，Amazon.com の場合，レスポンス時間が 1/10 秒余計にかかると 1% の売り上げを逃すと言われており，Google でも 1/2 秒の遅延増加がトラフィックを 1/5 も低下するといわれている．さらに，Twitter, Blog など，ユーザーからの応答性を要求されるサービスも急激に増大している．また，マシンコムの様な制御データをインターネット網を介して転送される運用アプリケーションも多数出現し，制御システムの制御情報データがインターネット網で転送されることによるデータ転送の時間コミットの必要性も急速に高まっている．この Web サービスモデルを実現するシステム要件としては，多様化への対応，オンライン実行，応答時間のリアルタイム性などがあるが，それらの要求を，同時に満たす技術の整備は進んでいない．現在の Web システムは，スケールアウト型モデルとした分散ファイルシステムで拡張性を実現するが，そのデータ更新情報は，ブリュワーの CAP 定理[4]が示す様に，書き込み時間のコミットが難しい．また自律分散キャッシュシステムとして FIF(Faded Information Field)[5]も提案されているが，この技術は読み出しの頻度によって情報の質をクラス化する提案であり，応答時間はコミットしていない．そのため，現在の Web サービスでは，リアルタイム性要件をアプリケーションプログラムで定義し，特定のシステムハードウェアに合わせたソフトウェア処理を施している．この手法は，要件定義を満たす一つの解ではあるが，システム要件が変わるたびにソフトウェアプログラム変更やパラメータ変更を余儀なくされる．一方で自律分散システムは本来，

個々のノードの集合が全体システムとなるアーキテクチャーを持つが，システムを構成する個々のノードに対し，アプリケーションプログラムに応じた，応答時間への個別対応が難しいといった問題を抱えている．

こうした応答時間がコミットできない双方向通信処理の低遅延問題を，ストレージデバイスレベルまで考慮した，高応答通信可能なブロックキャッシュ空間を構築する汎用的な自律分散 L3 キャッシュソフトウェア[6] [7]を用意し，解決する方法を考案した．L3 キャッシュは，CPU からノースブリッジを経由して接続されるローカルメモリ上に構築される I/O 専用のブロックキャッシュ空間で，書き込みイベントに対し Write-Back ポリシーを持つことで，更新処理の応答性をを高めることができる．このため自律分散 L3 キャッシュは性能面で従来方式に比べ優位であり，この優位性が大きな差別化のポイントとなる．この自律分散 L3 キャッシュを世界市場に提案，浸透させていくに当たっては，この性能面での優位性を前面に出して，実運用アプリケーション例を増やしていき，これによって，この分野における必須ソフトウェアとして訴求していく戦略が有効と考えられる．以下，本論文では２章で国際化戦略について紹介し，３章では性能優位性からの差別化の実践例を示す．また，４章では開発手順と品質管理を，５章に L3 キャッシュソフトウェアの基本アイデアを紹介し，６章にまとめを述べる．

2. 国際化戦略

2.1 国際化のための標準規格

自律分散システムは，1994 年に Ethernet ベースのオープン自律分散システム（およびネットワーク）が製品化され，1996 年にはネットワーク仕様が公開された．その後，国内の FA 標準化推進と共に，公開された仕様を用いて，MSTC（（財）製造科学技術センター）／JOP（FA オープン推進協議会））にて，大学・ユーザー・ベンダが参加して標準化および実証プロジェクトが進められた．JOP 仕様として，Ethernet ベース自律分散ネットワーク（ADS-net）の仕様が制定され，1999 年にそのプロトコル仕様書 [8]（和文版・英文版共）が公開された．この仕様に基づいて，世界中の誰もが自由にロイヤリティフリーで，ADS-net を開発・販売・使用できる様になり，オープンな分散型システム向けコミュニケーションプラットフォームとして，普及している．さらに，国際標準化については，国際標準化機構(ISO)において，ADS-net を含む Ethernet ベースの制御ネットワークのフレームワーク標準化提案が NWIP（New Work Item PropOSal）として採択され，2003 年には，ADS-net が国際標準（ISO15745）となった．この ADS-Net は日本発の国際標準ネットワークとしての地位を確立したが，この例に見るように，技術やソフトウェアの標準化は，海外進出を進める上での重要な戦略の一つであると考えられる．一方，Web サービスでは性能要件が重視されるが，その中でも応答時間は極めて重要なファクターである．この応答時間は通常，アプリケーションソフトウェアが実行されるハードウェアの性能に大きく左右されるが，その対処法として，個々のシステムに搭載されるアプリケーションソフトウェアで対応する方法も考えられる．しかしこの方法では，システムの特異性やインフラの違いによる地域性が生じるため，そのまま海外展開するのは難しく，たとえ同じ運用であっても，使用するシステムのハードウェアが変れば，遅延時間も変ってしまう．そこで，こうしたハードフェアの影響や個別のアプリケーションソフトウェアの影響を極力排除し，必要な応答性能を実現する方法として，OS 配下のドライバーとして高応答を実現する自律分散 L3 キャッシュという方式を考えることができる．この方法では，ハードウェアやアプリソフトの影響を排除するという特色ゆえに，汎用性を確保しやすく，結果的に市場展開もしやすいと考えられる．

図 1 は，自律分散システムの環境である OS，ミドルウェア及びアプリケーションプログラムと，L3 キャッシュの関係を示す．

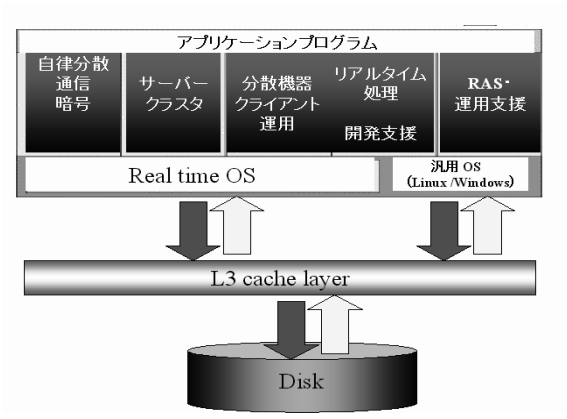

図 1. 自律分散システム環境と L3 キャッシュの関係

L3 キャッシュは，性能要件を満たすためにカーネルプログラムとし，OS 環境に依存する形で実装されるため，I/O デバイスドライバーには依存せず，ブロックデバイスドライバーで使用できるインタフェースを持つ．これによって L3 キャッシュ自身のソースコードを開示せずに市場展開することができる．

2.2 実運用アプリでの採用主義

自律分散 L3 キャッシュは当初，コンピュータシステムの I/O の効率化を行うためのキャッシュソフトウェアとして，データベース，高速データ通信や科学技術計算用として開発されたが，その特性を考慮すると，リアルタイム性のある RTSOAA 型サービスの Web の双方向通信要求を実現するための技術としての応用が有用であると考えられる．国内では，通信事業者，データセンター，各種研究所での実績がある．しかし，運用アプリケーションに特殊性がある為，出荷数量が限定的となり，ビジネス的には成功に至っていない．その経験を踏まえ，新たに検討し立案されたのが“実運用アプリでの採用主義”である．“実運用アプリでの採用主義”とは，L3 キャッシュを実装したほうが“性能的”または，“機能的”に優位で，逆に採用無しでは運用に支障をきたす実アプリケーション分野に的を絞ってエンドユーザーと共に実地検証し，リファレンスを国内外で得るローラ作戦を実施する考え方である．そして，その結果として，この性能優位性の差別化を基にして，それらの実運用アプリケーションにおいての，実質的なデファクトスタンダードとする技術標準化へのアプローチ方法である．標準化のメリットは，品質，安全性，互換性の確保であるが，ISO などのデジュールの国際規準にするためには調整に大きな時間を費やすというデメリットもある．一方，デファクト化は，市場が取得選択して淘汰されたもので常に競争が前提となるが，市場主導型で即効性というメリットがある．その為，性能や機能で差別化できる L3 キャッシュでは，実運用アプリでの採用主義に基づくデファクト化を狙う戦略を採用した．

3. 性能優位性からの差別化の実践

本章では実際のアプリケーションに L3 キャッシュを搭載することで性能優位性を実現し，システム差別化に成功した事例２つを紹介し，これらの例から L3 キャッシュのデファクトスタンダード化による市場展開の戦略について述べる．

3.1 事例１：自律分散 L3 キャッシュを持つ Web アプリケーションファイヤウォール

Web サービスが年々増大するなかで，Web サイトの脆弱性をついたアプリケーションレイヤアタックが増大しており，クロスサイトスクリプティングや，SQL インジェクション攻撃の被害が大きな社会問題になっている．これらを防ぐには，Web サイトプログラムレベルで White list を作成する必要があるが，Web アプリケーションファイヤウォール（WAF）が登場し，それを利用するケースも多くなった．しかし，WAF は，HTTP や HTTPS のアクセスのプロトコルを逐次解析し，良否の判断を行うため処理時間が相当発生し遅延が発生する．WAF に関しては，この遅延時間をどのようにして減らし，Web のサービスレベル・アグリーメント(SLA)を如何にして維持していくかが，これまで課題となっていた．この課題に対し，我々は自律分散ノードの L3 キャッシュをこの WAF 構築に用いる方式を考案した(図 2)．この方式では比較解析するデータを L3 キャッシュ上に保持できるため HTTP や HTTPS アクセスの解析時間を大幅に短縮することが出来，従来方式に比べ性能面での差別化を図ることができる．

図２は，L3 キャッシュを構成した自律分散 WAF の構築例である．WAF は Web サーバーの前段階に接続される Reverse Proxy であるため，その解析性能は Web サイトのサービス時間に直結し，大きな影響を持つ．L3 キャッシュは，設定されたブロックデバイスのみをキャッシュする特性を持つため，専用に設定すればアクセスプロトコルの検索と解析が高速に行える．国内外の RTSOAA の運用ユーザーや e コマースを行う Web サービス企業もしくはインターネットデータセンターなどに対し，我々は上記の方式を実現提案することで，実質的なデファクトスタンダード化を推進する計画を立てた．この計画では，WAF メーカーへのアプローチはあえて取らず，我々の提案に賛同して提案方式を利用していただいた利用者から L3 キャッシュの利便性，利用価値を訴求していただくというアプローチを採用した．学会を通じた研究発表はもちろんであるが，実ユーザーでの運用試験に立会い，市場への浸透を第一プライオリティにおいた．この戦略は，地味ではあるが実運用アプリケーションでのリファレンスは，単に市場への展開だけでなく，結果的には，デファクト化を目指す標準化へ向けた近道となると考えられる．実際の市場への展開については，Web サービスが多く存在する米国以外に，韓国などアジアで Web サービス需要の高い地域への顧客コンタクトから実施している．アクセス解析の性能が高められることはそのままユーザーの Web サービスレベルの向上と，レスポンス性能維持につながるため一度導入が進めばそのまま継続採用に至ることになる．この様に，Web システムにおける高いセキュリティはその解析に時間が掛かるという概念があったが，L3 キャッシュの性能優位性を基に

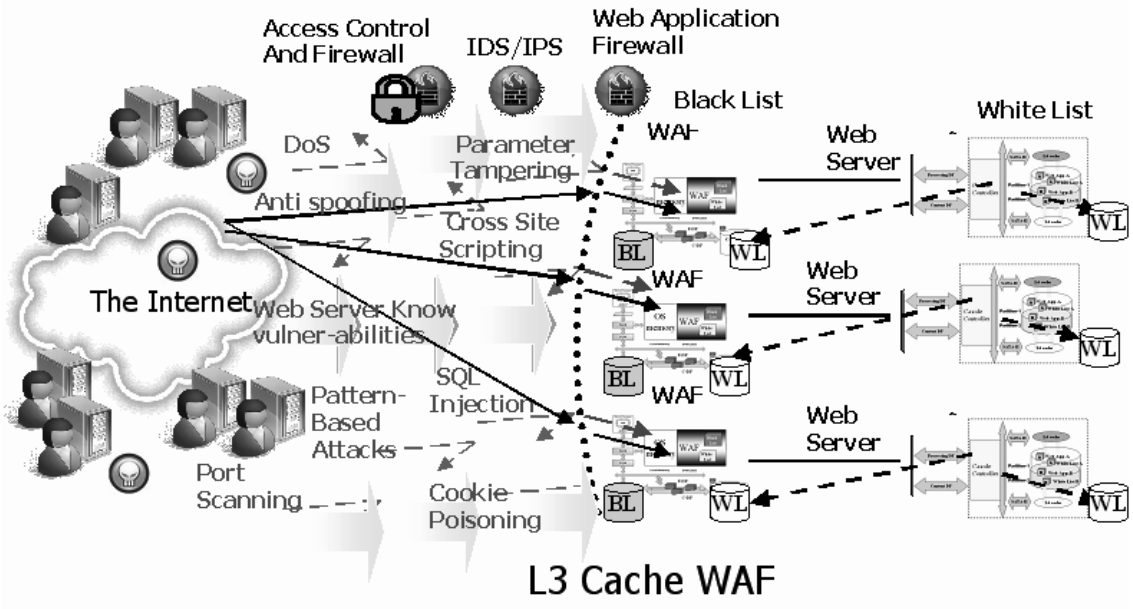

図２. L3 キャッシュ WAF の構築例

解析時間を短縮することで差別化し，従来トレードオフの関係であった高いセキュリティと，高レスポンスを同時に満たす解決策を提供することで，市場でのデファクト化による標準化を目指すことが可能となった．

3.2 自律分散 L3 キャッシュノードのローレーテンシー映像配信サービス

一般に，映像収録・配信サービスでは，画像フレーム落ちを防ぐため，書き込み・読み出しの遅延時間が最も大きな課題となる．

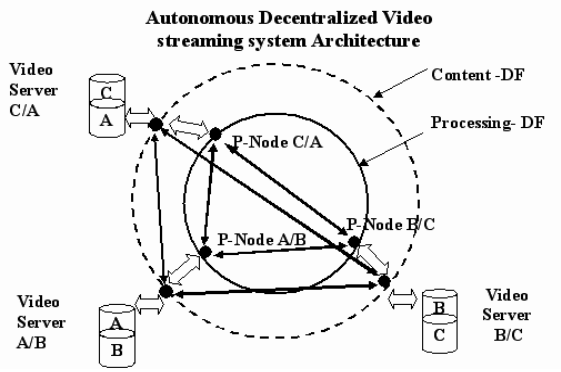

図３. 自律分散 L3 キャッシュノードのローレーテンシー映像配信サービスシステムの構成例

こうした課題を解決するために，図３に示すように，L3 キャッシュを用いた自律分散ローレーテンシー映像配信サービスシステムを構成する方法を考えることができる．

この方式では，L3 キャッシュを用いた自律分散プロセッシングノード（P-Node）がローレーテンシーで I/O 処理を行うため，映像収録・再生をスムースに行うことが可能となる．また，映像データをコンテントノード（C-Node）のサーバーが所有するので，同じデータの配信がデマンドに応じて L3 キャッシュの P-Node のみをオンラインで増設して対応できるアシュアランス性も確保する．L3 キャッシュノードの I/O 性能が高いだけでなく平均転送性能を高める事ができるために映像データの連続性を確保できる“映像品質の確保”を実現することができる．この“映像品質の確保”はシステムを構成する上で，その機能に関しての性能面における優位性を訴求する必要不可欠な特性である．映像配信は Web 技術やインターネットの普及に伴い必須のサービスとして市場が拡大しており，この分野で「映像品質の確保」を実現する方式は，需要が極めて高いと考えられる．

この事例のように映像配信システムに自律分散 L3 キャッシュを用いると，こうした市場からの要求を満たしながら，L3 キャッシュの有用性を主張でき，L3 キャッシュの市場でのデファクト化を進める上での適切な事例としてアピールすることができる．

4. L3 キャッシュソフトウェアの開発手順と品質管理

4.1 開発手順

L3 キャッシュの基本設計は，次の手順で開発を進めるが，まずその前提となるのが，要件定義のまとめである．要件定義は，マーケットの要求をインタビューし，その

最低要件と，最大要件を求める．次に平均値を算出するが，市場ニーズに最適化する必要があるので，その閾値は，インタビュー件数に合わせて加重配分を考慮する．

$$X_i(n+1) = f\left[\sum_{i=1}^{n} W_{ij} X_i(n) - \theta_i\right] \cdots\cdots\cdots (1)$$

（式１）

X_i は，インタビューから得た要件定義値，W_{ij} は，加重，θ_i は，閾値である．

L3 キャッシュの基本設計については，以下の３つのステップで進めていく．

Step-1:要求事項の整理分析

まず，最初のステップでは，L3 キャッシュに求められる要求事項を整理分析しておく．具体的には，マーケットにおける 運用アプリケーションと必要性能条件，そのデータのファイルサイズ，アクセスアドレスのローカリティの有無をインタビューし，その最低性能と，可能最大理論値を求めておく．

Step-2:アクセス頻度の平均値算出

次にアクセス頻度の平均値を算出する．アクセス頻度は市場ニーズに合わせて最適化する必要があるため，その閾値は式１に示すような形で，インタビュー件数にあわせて加重配分を考慮し必要な要件定義を明確にした後，プロセスフローを作成する．

Step-3:プログラムフローの策定

市場のニーズから得られた数値を基に，算出された基準値から要件定義を決定して，プログラムフローを起こしていく．

L3 キャッシュは，OS によってパラメータが異なるため，その数値と性能期待値を要件定義から想定して決定する．１）キャッシュポリシー，２）キャッシュオーバーヘッドの最大値，３）実行転送スピード，４）キャッシュ・リロケーショナルペナルティ，５）リアルタイム実行のためのレーテンシータイム，６）IOPS の最大値と最小値，７）アクセスユーザー数の最大値と最小値．

これらの数値を基にプログラム開発を進めてゆく．

4.2 品質管理

L3 キャッシュの開発においては，その品質の均一性をどのようにして担保するかという事もきわめて重要である．自律分散 L3 キャッシュの場合，主任技術者がメインプロセスフローを担当し，コーディングはプログラミングチームが担当するため，デバック後のモジュールレベルで様々な環境条件での品質チェックが重要となる．L3 キャッシュの品質チェックについては，複数の関係機関や，日本と海外の大学，研究所及びパートナーを通じた連携評価を実施している．L3 キャッシュモジュールは，OS 言語には直接関与しないが，Windows 版モジュールなどは一部 GUI も存在するため，その確認も地域毎に実施する．L3 キャッシュのモジュール評価テストは，市場にある一般的なツールを使ったものと，アプリケーションプログラムを走らせた場合の 2 通りで行う．我々のラボと関係のある海外の開発チームでは，English 版 OS を中心にテストを実施するが，日本語対応は日本国内でテストを実施した．これは，各国での言語基準が違う為であり，最終的なチェックで，統一出荷基準に到達した時点でリリースを許可する様にしている．

これによって，出荷地域に関わらず同一基準での品質確保を得ることが可能となる．

5. L3 キャッシュソフトウェア

5.1 L3 キャッシュソフトウェア概要

L3 キャッシュは，高応答性を求める運用プログラムのみをキャッシュする事で特定なサービス実行時間のレーテンシーを確保 [9] するもので，次の 3 つの機能を実現している．

機能１：ローカルメモリの一部を I/O ブロックデバイスに変換，これをターゲットのブロックデバイスにマウントしキャッシュ空間とする．

機能２： L3 キャッシュ上に要求されたブロックアドレスを検索する高速サーチを実装している．

機能３：　書き込みイベントの高応答性を実現するために Write-Back ポリシーを採用している．

以下，上記３つの機能について，その実現方法のポイントを紹介する．

機能１の実現のために２つのモジュールを用意している．一つ目のモジュールは，ローカルメモリをブロックデバイス用のキャッシュ空間とする役割を担う，RAM ディスクドライバー [10] である．このモジュールは OS メモリ空間と切り離した専用の RAM ディスクブロックデバイス空間を作成する．2 つ目のモジュールは L3 キャッシュドライバーで，RAM ディスク空間をキャッシュのテーブルとしてマウントし，メモリとデバイスの階層

制御を担っている．

機能２の L3 キャッシュの高速データ検索については，次のハッシュ型検索方法を用いている．

```
IRP Offset =0;
IRP length = 2048;
NrBlocks = ((offset + NrSectors - 1)>> *(targetDisk->SectorPerBlock2)) +1;
NrBlocks = ((  0 + 4  - 1)>> 3  ) +1;
Index = (ULONG) ((blockSector.QuadPart) % (targetDisk->NrBlocks));
Index = 1 % 100 = 1;
RowIndex = (Index / (targetDisk->index Columns));
ColumnIndex = 0%32= 0;
Block = _SearchBlock (targetDisk, blockSector, RowIndex, Column Index);
Block = _SearchBlock (targetDisk, 0, 0, 0);
```

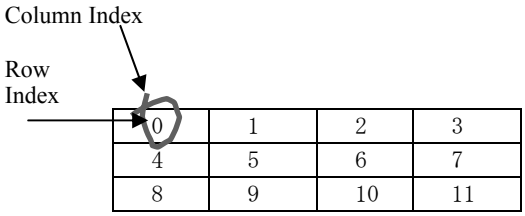

Table 1 L3 キャッシュ ハッシュベース検索

Table 1 は L3 キャッシュ上で要求されるブロックアドレスを効果的に検索する方法についてのコード記述の例を示している．この方法では，最初に格納されるキャッシュデータを Flush する場合，キャッシュ検索を必要としない事で，検索効率を高めている．各ブロックはこのインデックス上で結びついているリスト中で検索され，L3 キャッシュに存在するか否かを判断する．一般にハッシュ関数での検索は連続データでは有効性が高いが，この方法では小さなデータでも効果が期待できる．

機能３の書き込みイベントの高応答性を実現については，Write-Back ポリシーキャッシュの考え方を取り入れている．Write-Back ポリシーキャッシュは分散ノードの相互方向でのアクセス性能を上げる目的で，書き込み性能を向上させて結果的に読み出し・書き込みの双方向の I/O 性能を高めることができる．Dirty Data の Flush 値の設定は，主に読み出し可能なキャッシュデータ領域設定であり，運用に応じて RAM ディスクサイズの 50%～80%に設定する．L3 キャッシュでは，一度書き込まれたデータの再読出しのみに Read キャッシュ効果がある Read-Through モードに設定している．これは，OS 配下の管理メモリのバッファキャッシュ効果を利用するためで，これによって，読み出しキャッシュデータの L3 キャッシュと OS 管理メモリ空間でのデータの重複が防げると共に L3 キャッシュが完全に Write-Back イベントに専任できる様にもなる．DRAM で構成されたローカルメモリを用いた専用ブロック I/O キャッシュは小規模なパケットデータでも高い I/O トランザクション性能を示し，ローカルメモリと等しい帯域幅でのデータ送受信が実現できる．また，Write-Back, Read-Through モードとし，常にレーテンシータイムを確保できるキャッシュポリシーとしている．

5.2 L3 キャッシュソフトウェアの実行環境とキャッシュゲイン

L3 キャッシュは OS に依存するカーネルプログラム上で動作し，自律分散ノードとしての高い応答性を実現している．その実行可能な OS 環境は，Linux 2.4Kernel, 2.6Kernel, WindowsXP, Windows7 である．L3 キャッシュに使用するメモリは分散ノードのメインメモリを使用するため，OS メモリ以外に必要なメモリ容量を搭載するのが条件となる．また，高応答するためのプログラムを決定し，その実行するプログラムが常駐するストレージ・パーティションに L3 キャッシュの RAM ディスクをマウントして実際の構成を構築する．

式（２）は，キャッシュの効果をそのヒット率と L3 キャッシュを構成するソフトウェアの負荷を表したもので，Amdahl's Law に従って，キャッシュゲインが求められる．

キャッシュのレベル"i" は，その算出理論値である．

$$G_i = \frac{1}{1 - C_i + \frac{C_i}{X_i}} \prod_{j=1}^{m} L_{i,j} \quad \text{.....} \quad \text{.....} \quad (2)$$

Where

G_i = Gain due to cache at level i
C_i = Cache size ratio (hit rate) at level i
$L_{i,j}$= overhead factor j, at cache level i,
X_i= Cache speed ratio of lower level storage

例えば，接続されるデバイスの性能が HDD の場合，T_{d4}=9ms, 一方，キャッシュメモリの性能は T_{d3}=0.045ms なので，性能差は，X_3=200 となる．

キャッシュの性能は，そのヒット率が最も大きなインパクトを持ち，その性能維持には，如何にキャッシュヒット率を落とさないかという工夫に最終的には集約される．キャッシュアルゴリズムは現在 20 種を超えるほど提案されているが，ヒットさせるべきキャッシュ空間をタイムリーに確保できるかが，性能維持の重要要件である．さらに，多層化キャッシュでは，各層でのキャッシュゲインと，多重に存在するキャッシュソフトウェアによるプログラムオーバーヘッドを考慮して算出する必要

がある．

L3 キャッシュは，ユーザーが使用する OS 環境にあったモジュールを選択するだけで，接続されるブロックデバイスには影響されず，ハードウェアデバイスとは完全に独立的に使用できる．しかし，L3 キャッシュ空間を効果的に構成するためには，ハードウェア性能の高いローカルメモリをキャッシュ空間に設定する必要がある．従って，適切なメモリ容量を準備，確保し効果的運用を可能とする．このため L3 キャッシュを用いるためには，予め適切なメモリ容量を準備，確保しておくことで，その潜在的な性能優位性を生かした効果的運用が可能となる．

5.3 L3 キャッシュのネットワーク特性

従来のネットワーク転送では，通信時に必要なデータであるオペランドの存在する場所までは，考慮されていないため，遅延時間を保証する事ができなかった．小規模なパケットの自律分散ネットワーク通信の応答時間で，最も遅延の発生する箇所は送信するデータの読み出しと受信時のデータ格納である．ISO モデルの 7 レイヤのうち，送信時の 4 層以前のカプセル化が始まる工程と，受信時の 4 層までの非カプセル化工程以降のオペランドの収集と格納時の遅延を如何に少なくしてトータルのネットワーク性能を高めるかが通信性能の大きな性能要因となっている．従って，第 7 層のアプリケーションレイヤにおけるストレージ性能の向上は，小規模パケットの自律分散ネットワーク通信処理において時間コミットを可能にするのに有効な実装手段である．

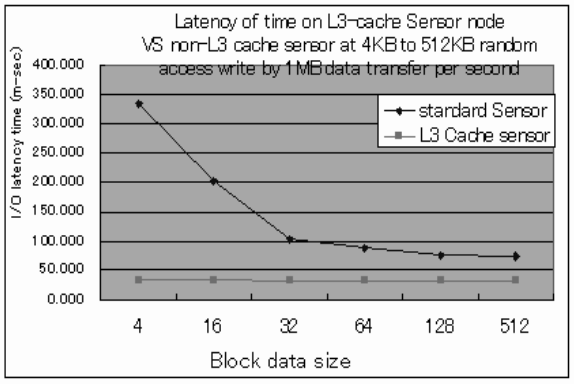

図 4 . L3 キャッシュのネットワーク特性

図 4 は，L3 キャッシュノードと通常のノードのレーテンシータイムを比較したネットワーク特性である．非 L3 ノードは 4KB では，333.8msec，8KB で 202.5msec，16KB で 104.1msec であるのに対し，L3 キャッシュノードは，4KB で，33.7msec，8KB で 32.6msec，16KB で 32.4msec とパケットサイズに関わらずレーテンシータイムをほぼ一定に保持できる優位性が確認できる．

6. おわりに

リアルタイム性を要求される新しい Web サービスの RTSOAA は，国内外を問わず大きな需要が見込まれる．RTSOAA 型の Web サービスでは，双方向通信でのリアルタイム性が重要であるが，従来の分散ノードでは困難である．またレーテンシータイムを確保する手段としてアプリケーションレベルでも対応可能だが，特定の用途，特定の機器での局所的かつ限定的な対応になる．従ってこのタイプのソフトウェアでは，標準化がしづらく，国際市場の展開が難しい．この問題を解決する手段として，通信時のカプセル化と非カプセル化をストレージデバイスまで考慮した自律分散 L3 キャッシュソフトウェアを開発し利用することにより，対象とする運用アプリケーションに左右されない方法を考案した．本論文では L3 キャッシュの市場への展開方法として，その実運用アプリケーションでの性能や，機能面で優位性を提示し，それらを訴求ポイントとしてデファクトスタンダード化を狙った世界展開を進める方法を提案した．また，その実例として，リアルタイムな解析が求められる WAF において，ローレーテンシーにより Web サービスレベルを高めることで，市場の支持を得る考え方を紹介した．また映像分野では，画像品質を確保する事で市場の機能面の要求に応え，普及させる考え方を示した．さらに，L3 キャッシュの品質評価については，４カ国での同時評価を行うことにより，日本国内のニーズに縛られず，海外ニーズも取り込んだ形での評価とする工夫を紹介した．これにより言語環境の影響を排除し，国際的に通用しうるソフトウェアとしての展開を加速することができる．

今後の展開は，性能を高めるための各種パラメータの最適化の研究と，性能有利性の働く実運用アプリケーションの発掘を進め，多様な分野でのデファクト化を進めていきたい．

参考文献

1) I.-L. Yen, R.Paul and K. Mori. “Towards integrated methods for high-assurance systems” IEEE Computer, 31(4):32-34, April 1998.
2) K. Mori. “Autonomous decentralized systems: concept, data field architecture and future trends” In Proc. of ISADS, IEEE, pages 28-34, 1993
3) Jen-Yao Chung, Jun-Jang (JJ) Jeng, and JOSef Schiefer “The 1st International Workshop on Real-Time Service-Oriented Architecture and Applications (RTSOAA 2008)” 30 June 2008, USA
4) ブリュワーの定理　2012 年 8 月 12 日閲覧
http://www.julianbrowne.com/article/viewer/brewers-cap-theorem

5) Xiaodong Lu, Kinji Mori ”Automous Information Services Integration Architecture for Service Assurance in Multi-Agent System” The 9th International SympOSium on Autonomous Decentralized Systems (ISADS 2009), Athens, Greece, March 23-25, 2009, pp 395-401
6) Hironao Takahashi, Hafiz Farooq Ahmad, Kinji Mori, “Layered Memory Architecture for High IO Intensive Information Services to Achieve Timeliness” 11th IEEE High Assurance Systems Engineering SympOSium Nanjing, China, December 3 - 5, 2008
7) Hironao Takahashi, Hafiz Farooq Ahmad, Kinji Mori, "Balanced Memory Architecture for High I/O Intensive Information Services for Autonomous Decentralized System", The 9th International SympOSium on Autonomous Decentralized Systems (ISADS 2009), Athens, Greece, March 23-25, 2009, pp 93-99
8) ADS protocol specification R3.0 , MSTC/JOP 1101-19999/09/3K.
9) 高橋宏尚，森欣司，”自律分散システムの高応答 I/O ノードの挑戦” 情報処理学会 Vol.51 No5May 2010 P529
10) RAMDISK browsed at sep 20th 2009
http://www.vanemery.com/Linux/Ramdisk/ramdisk.html

高橋 宏尚（正会員）

E-mail: hiro@dts-1.com

東京工業大学大学院情報理工学研究科計算工学専攻，博士課程在籍， 自律分散システムの情報システムでの応用と高速化の研究に従事．また，データトランスミッション技術の構造を提案，その基本特許を持つ．現在 DTS 株式会社にてその応用製品の開発を担当， IEEE, IEICE, IEE, IPSJ 会員

森 欣司 （正会員）

E-mail: mori@cs.titech.ac.jp

1969 年，1971 年および 1974 年に早稲田大学から電気工学の学士，修士および博士号を受ける．1974 年～1997 年，システム開発(System Development) 研究所，日立株式会社にて研究に従事．1997 年，東京工業大学大学院の教授に就任．研究分野は分散コンピュータ，フォールトトレラントおよびモバイルエージェント．IEEE，IEICE のフェロー，および IPSJ，SICE 会員

投稿受付：2010 年 10 月 16 日
採録決定：2011 年 2 月 16 日
編集担当：平山 雅之（東芝）