マルチメディア通信と分散処理　89－11
（１９９８．６．４）

高機能ネットワークカードを用いたWEBサーバの性能向上方式

前田 誠司　金井 達徳　鳥井 修　宮本 幸昌　浅野 滋博　増渕 美生

(株) 東芝 研究開発センター

インターネット/イントラネットでのWebの利用が広がると共に，Webサーバの性能向上が求められている．我々は，Webデータキャッシュ用の大容量メモリとプロトコル処理用のプロセッサを搭載した高機能ネットワークカードを用いて，Webサーバの性能を向上させる方式を開発した．本カードをソケットレベルでネットワークカードとして見せることで，既存のPCサーバ上のWebサーバソフトウェアに変更を加えることなく組み合わせて用いることが出来る．また，複数枚のカードを用いることによって負荷を分散させることも可能である．

Intelligent Network Adapter for High Performance Web Server

Seiji Maeda, Tatsunori Kanai, Osamu Torii,
Yukimasa Miyamoto, Shigehiro Asano and Yoshio Masubuchi

Research and Development Center
Toshiba Corporation

Today, web technology is widely used in the Internet and intranet, and demand for higher performance for web server is growing. We developed an intelligent network adapter for accelerating web server. The adapter has a large amount of DRAM for caching web data and a high performance microprocessor for handling HTTP and TCP/IP protocols. We can attach multiple adapters into a PC server to gain scalablility in performance. Any commercial web server software can be used with this adapter without modification.

1　はじめに

インターネットの普及とともに，世界中の様々な情報を電子的に引き出すことが可能になり，非常に効率の良い情報共有/情報提供の枠組が提供されるようになってきた．最近では，この仕組みをイントラネットに利用して企業内の情報共有システムを構築しようという動きが活発になっている．これらの中で中心的な役割を果たしているのが，World Wide Webの技術である．インターネット/イントラネットにおけるWebの利用が急激な勢いで増加するにつれ，そのインフラを支える主たる構成要素であるネットワークとサーバの能力が問題となることが多くなってきた．このうちインターネットでは，ネットワークの能力がそれほど高くないために，一般にはサーバよりもネットワークがボトルネックとなる傾向が強い．しかし，ユーザ数の爆発的な増加とともに，アクセス頻度の高いサイトではサーバの能力も不足しがちになり，常にサーバ増強の必要性に迫られているのが実状である．また，イントラネットでは，高速ネットワークの普及率が高くなっているため，イントラネット環境を利用した応用システムが増えるとともに，サーバに対する負荷が重くなるという傾向は，インターネットに比べて顕著である．さらに，画像をはじめとするデータ量の大きいマルチメディアコンテンツが次第に増加していることが，この傾向に拍車をかけている．

このように，いずれの環境においてもサーバのデータ供給能力が重要になる傾向は今後ますます強まるが，従来の一般的な共有メモリ型マルチプロセッサでは，メモリアクセス性能がボトルネックとなって，コストパフォーマンス良く大規模なデータサーバを構成することは困難である．一方，分散メモリ型マルチプロセッサ方式を用いると，並列化によってバランス良く性能を向上させることが可能であるが，実用面からは既存のサーバ/ネットワーク環境との相性が重要な要素となる．すなわち，システムのオープン性を損なわない実装が求められる．

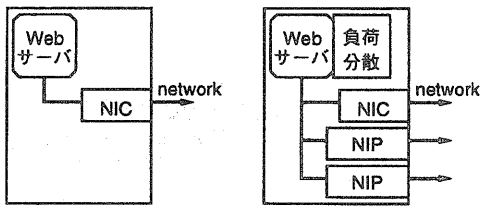
本報告では，データサーバ機能を持つネットワークカードを複数個並べることにより高性能化を図ると同時に，PC サーバという既存のオープンシステム環境との整合性を維持することを特長とした Web サーバの構成方法とその実装に関して述べる．

2 Web サーバの性能向上方式

2.1 システム構成

PC サーバ上で動作する Web サーバソフトウェア（例えば Netscape 社の Enterprise Server や Microsoft 社の Internet Information Server）は，図 1(A) に示すように NIC(Network Interface Card) を通してクライアントからの要求を受け取り，要求されたファイルを送り返す．

我々の開発した高機能ネットワークカード NIP (Network Interface Processor) は，図 1(B) に示すように PC サーバに複数枚搭載して用いることで，PC サーバの Web サービス能力を向上させる．Web サーバソフトウェアには，単に複数枚の NIC が存在するように見せることで，既存の Web サーバソフトウェアをそのまま使い続けることが可能である．NIP 上には，Web データのファイルをキャッシュするソフトウェアを実装している．クライアントが要求してきたファイルが NIP 上のキャッシュに存在すれば，そのファイルは PC サーバからではなく NIP から送り出すことで，PC サーバの負荷を小さくすることができる．複数枚の NIP 間の負荷分散は，Web サーバソフトウェアに組み込んだ専用の負荷分散モジュールによって行う．



(A) PCサーバ　　(B) PCサーバ+NIP

図 1: システム構成

2.2 動作原理

本システムでは，Web サーバの管理するファイルの持つ参照の局所性と，HTTP プロトコル [1] の持つリダイレクション機能を利用して，複数枚の NIP を動作させている．まず，ファイルの局所性に関して説明した後，全体の動作を説明する．

Web サーバで管理するデータファイルの単純化した例を図 2 に示す．この例は，home.html という名前の HTML ドキュメントの中で，logo.gif というイメージデータのファイルを参照し，さらに，products.html という HTML ドキュメントに対してリンクが張られている状況を示している．ここで，home.html 内では，logo.gif や products.html を指す URL が絶対 URL でなく相対 URL で書かれている．HTML の規約では，この部分に絶対 URL を指定しても相対 URL を指定しても構わないが，多くの場合，同じ Web サーバ内に存在するファイルに対する URL は相対 URL で書かれる．通常，1 つのページは複数のイメージ等のファイルから構成され，1 つの Web サイトは関連する複数のページから構成される．そのため，相対的 URL によ

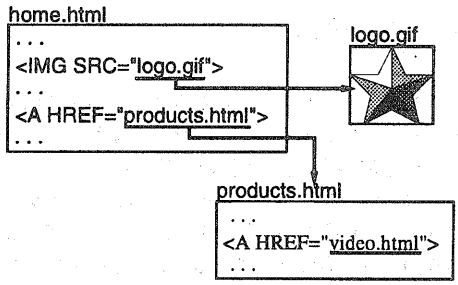


図 2: Web サーバ上のファイルの例

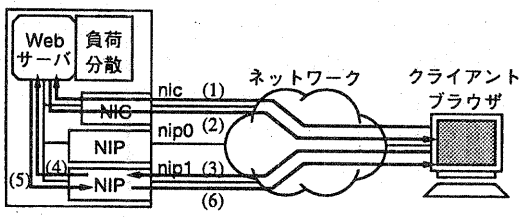


図 3: 最初のリクエストの処理

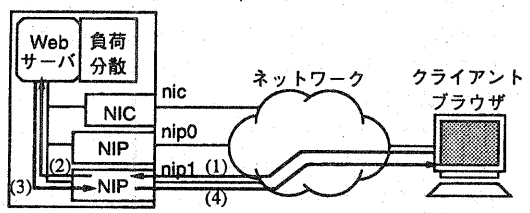


図 4: 2 回目以降のリクエストの処理

るローカルな参照の割合は大きい.

NIP を搭載した PC サーバにおいて図 2 のファイルをアクセスする場合の動作を説明したのが図 3 である. ここで, NIC と 2 枚の NIP はそれぞれ独立の IP アドレスを持ち, その名前を nic, nip0, nip1 とする. また, この PC サーバは nic という名前で公開されているものとする. PC サーバ上の Web サーバに対して, クライアントが初めて要求を出したときの処理の流れは次のようになる.

(1) クライアントが GET http://nic/home.html リクエストを発行
(2) Web サーバは http://nip1/home.html へリクエストをリダイレクション
(3) クライアントは GET http://nip1/home.html リクエストを再発行
(4) NIP は home.html がキャッシュに無ければホストにリクエスト
(5) Web サーバは home.html を送り返し, NIP はそれをキャッシュに記憶
(6) NIP は home.html をキャッシュから読み出してクライアントへ送り返す

このように, NIC に来たクライアントのリクエストは負荷分散モジュールによって一度リダイレクションされて適切な NIP に振られる. そのため, 通常の Web サーバよりもリダイレクション一回分のオーバヘッドを持つ. しかし, これ以降のリクエストは図 4 に示すように異なる手順で処理される. 例えば, home.html という HTML ドキュメントを取ってきたクライアントが, その中から相対 URL で参照されている logo.gif というファイルを取りに行く場合の処理の流れは次のようになる.

(1) クライアントは GET http://nip1/logo.gif リクエストを発行
(2) NIP は logo.gif がキャッシュに無ければホストにリクエスト
(3) Web サーバは logo.gif を送り返し, NIP はそれをキャッシュに記憶
(4) NIP は logo.gif をキャッシュから読み出してクライアントへ送り返す

通常 Web ブラウザは, 相対 URL で指定されたファイルが必要になると, その基準となる現在の文書の絶対 URL と必要なファイルの相対 URL から必要なファイルの絶対 URL を作り出し, それに基づいて Web サーバにリクエストを出す. ここで述べた例では, 最終的に home.html を nic1 から持ってきているので, logo.gif のリクエストは同じ nic1 に向けて発行する.

以降, リンクをたどって同じ Web サーバ内の相対 URL で指定されたファイルをアクセスする場合も, 同様にリダイレクションを起こさず, 直接 NIP にリクエストを発行することになるので, リダイレクションのオーバヘッドは相対的に小さくなる.

3 NIP のハードウェア構成

3.1 プロセッサシステム

NIP は図 5 に示すようにプロセッサ, メモリ, ネットワークインタフェース, PCI-PCI ブリッジを内部 PCI バスで接続した構成をしている. NIP に使用しているプロセッサは, 次の 2 点を考慮して PowerPC 603e を採用した.

- NIP は I/O カードとして動作するため十分低消費電力でなくてはならない.
- HTTP および TCP/IP のプロトコル処理のための計算能力が必要である.

プロセッサは Host-PCI ブリッジを介して内部 PCI バスおよび 64MB の DRAM に接続されている. 内部 PCI

バスには，ネットワークインタフェース（100BaseT Ethernet）と後述のPCI-PCIブリッジが繋がっている．

3.2 PCI-PCIブリッジ

PCI-PCIブリッジはNIPのプロセッサとホスト計算機であるPCサーバの間の通信を行う機能を担っている．通常のPCI-PCIブリッジとは異なり，ブリッジ自体にDMA転送の機能を持たせるため，市販のPCIブリッジチップ(V3社のV962PBC)を2個使用し，ローカルバス同士を接続して実装している．

ホストのPCサーバとNIPは，このPCI-PCIブリッジのDMA機能を利用して互いのメモリ間でデータを交換し，同じくPCI-PCIブリッジ内のMAILBOX機能を使用して割り込みベースのメッセージ交換を実現している．

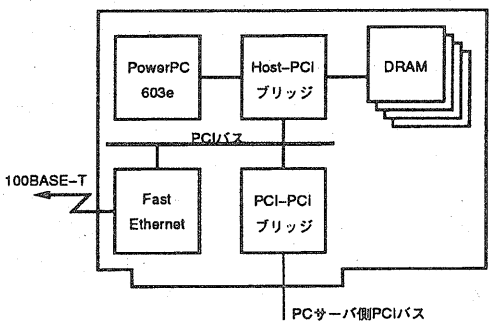


図 5: NIPのハードウェア構成

4 キャッシュ機能のファームウェア化

4.1 キャッシュ管理

NIP上ではファームウェア化したキャッシュモジュールが動作している．キャッシュモジュールの基本的な機能は，squid[2]等の一般的なキャッシュサーバと同様である．すなわち，ホストであるPCサーバ上のWebサーバとクライアントのブラウザの中間に位置し，頻繁にアクセスされるファイルを一時的にキャッシュする．しかし，ディスクなどの外部記憶装置を使用するキャッシュサーバと違って，キャッシュするファイルはNIP上の物理メモリ上に記憶するため，高速にアクセスすることができる．

キャッシュ中のファイルは，アクセスされた時刻をもとにLRUで管理している．新規にキャッシュするための物理メモリが不足すると，すでにキャッシュしているファイルの中から，LRUの順でもっとも古くアクセスされたファイルをキャッシュから消去する．

4.2 HTTPプロトコル処理

NIPのファームウェアは，ネットワーク経由で接続して来るクライアントのブラウザおよびPCサーバ上のWebサーバの双方と，HTTPプロトコルで通信しながら，ファイルのキャッシングや更新処理を行う．クライアントからのファイル要求を受け取ったNIPは，要求されたファイルがNIP上にキャッシュしていない場合には，PCサーバに対して要求を転送し，送られて来るファイルをキャッシュしながらクライアントに転送する．逆に，要求されたファイルをNIP上にキャッシュしている場合には次の手順で処理を行う．

1. NIP上にキャッシュしているファイルの更新時刻を用いて，それが最新かどうかPCサーバに対して更新確認を行う．
2. PCサーバ上にさらに新しいファイルがある場合には，PCサーバより送られる新しいファイルをキャッシングしながらクライアントに転送する．
3. PCサーバとNIPのファイルが同一である場合には，直接キャッシュよりクライアントに転送する．

この処理手順では，NIP上にキャッシュしているファイルへの要求が来た場合，毎回その内容が最新かどうかをPCサーバに確認することになる．しかし，一般にPCサーバ上のファイルの更新頻度は比較的少ないので，毎回確認する必要はない。そこで、一度PCサーバにファイルの更新確認を行なった後，一定期間は更新確認を行なわない設定をすることによって，オーバヘッドを軽減することも出来る．

Webサーバは，クライアントからの多数のコネクションを同時に効率良く実行する必要がある．そのため，NIPのファームウェアの開発に際しては，組み込みOSの提供するタスク1つに対して1つのコネクションを割り当てて実行するのではなく，1つのタスクで全てのコネクションを処理するような構成を採った．この方式により，タスクに割り当てるスタック容量の削減，およびタスクスイッチのオーバーヘッドを低減することができる．

4.3 ホスト計算機とのインターフェース

前節で説明したように，NIP はホスト計算機である PC サーバと HTTP プロトコルで通信する．HTTP の載るトランスポート層のプロトコルしては，通常のネットワークを介した通信のように TCP/IP プロトコルを用いるのではなく，より単純化したプロトコルを用いて通信を行なっている．これは，NIP と PC サーバを結合する物理層のインターフェースは PCI バスであり，通信データ紛失等の心配はなく，フロー制御の機能のみが必要なためである．このような単純化したトランスポートプロトコルを採用することで，独立したキャッシュサーバを用いた場合と比較して，ホストとなる Web サーバとの間のネットワークのオーバヘッドおよび TCP/IP プロトコル処理のオーバヘッドが低減できる．また，PC との通信プロトコルを BSD ソケットライクなインターフェースで実装することによって，NIP のプログラミングを容易に行なえるようにした．

5 Windows NT 環境への統合

5.1 NIP の仮想化

NIP は，ホストの PC サーバ上で動作している既存の Web サーバソフトウェアのアクセラレータとして，性能を向上させる．そのため，Web サーバからは，NIP は従来の NIC と同じように見える必要がある．そのような仮想化を実現するために，Windows NT のソケット API である Winsock2[3] のレベルで，NIP を仮想的な NIC としてアプリケーションプログラムに見せる方式を開発した．

Winsock2 は図 6(A) に示すように，アプリケーションプログラムから呼び出されるダイナミックリンクライブラリ (DLL) として実装されており，その中から TCP/IP 等のトランスポート層のドライバを呼び出す構成になっている．新しいトランスポートドライバや新しいサービス機能を拡張するために，Winsock2 には Layered Service Provider(LSP) と呼ぶ機能拡張のためのインタフェースが用意されている．本システムでは図 6(B) に示すように，NIP のドライバを Winsock2 の体系に統合するための新しい LSP(ここでは NIP LSP と呼ぶ) を開発した．次節で説明するように，NIP LSP が NIC と NIP の違いを吸収することで，Web サーバソフトウェアからはあたかも NIC のみが存在するよう見せることが可能になった。

5.2 NIP LSP の動作

NIC のみを持つ通常 PC サーバ上のアプリケーションプログラムがソケットを用いて通信を行う場合，図 6(A) のように Winsock2 DLL を呼び出して NIC を管理する TCP/IP ドライバ上にソケットを作成する．このソケットでクライアントからのコネクション確立要求が到着するのを待ち、要求が到着したらクライアントとの間にコネクションを確立し、データ配信を開始する。

これに対して本システムの場合，アプリケーションプログラムである Web サーバが Winsock2 DLL に対してソケットの作成を指示すると，Winsock2 DLL は図 6(B) のように NIP LSP にソケットの作成を指示する．NIP LSP は TCP/IP ドライバと NIP ドライバの双方に別々のソケットを作成するが，その 2 つのソケットのペアを仮想的な 1 つのソケットとして Winsock2 DLL に見せる．そのソケットに対して，Web サーバがクライアントからのコネクション確立要求を待つ関数 (listen) を呼び出すと，その呼び出しは NIP LSP に伝えられる．NIP LSP は NIC 上のソケットを要求待ち状態にする TCP/IP ドライバの手続きと，NIP 上のソケットを要求待ち状態にする NIP ドライバの手続きの両方を呼び出す．その後，NIC と NIP の双方のソケットでクライアントからのコネクション確立要求が到着するのを同時に待ち、いずれかのソケットにコネクション確率要求が来ると，要求が到着した側のドライバを使ってクライアントとの間にコネクションを確立し、データ配信を開始する．

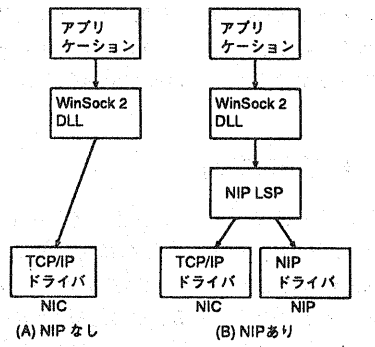


図 6: Winsock2 による NIP の仮想化

6 負荷分散モジュールの処理方式

6.1 負荷分散モジュールの実装法

複数枚のNIPボードをWebサーバのアクセラレータとして用いる場合に，複数のNIPに処理要求をうまく分担させてシステム全体の処理能力を向上させる事が必要になる．その処理を行うのが負荷分散モジュールである．

NIPのホストとなるWebサーバは，現在はNetscape Enterprise Serverを用いている．負荷分散モジュールは，Enterprise Serverの提供するNSAPI[4]と呼ぶAPIを用いて実装している．NSAPIでは，クライアントから送られてくるリクエストを処理する各段階に，フックとなるプログラムを組み込んで拡張することができる．この機能を利用し，リクエストされたファイルを送り返す直前に，そのままファイルを送り返すか，他のNIPにリクエストを転送するようにリダイレクション応答を返すかの判断をする負荷分散モジュールを組み込んでいる．

負荷分散モジュールは，送られて来たリクエストをどのNIPで処理するかを判断するために，各NIPの負荷状態等の情報を知る必要がある．そのために，NIPはWebサーバにファイルをリクエストする際に，以下のような情報を持つ特別なHTTPヘッダを挿入する．

- NIP番号
- NIPの負荷の状態
- 要求されているファイルがNIP上にキャッシュされているかどうかを示す情報

負荷分散モジュールはこの情報を参照することにより，リクエストがどのNIPから来たか，そのNIPの負荷はどれくらいか，要求しているファイルがキャッシュされているかどうか，といった情報を得る．

6.2 負荷分散戦略

負荷分散モジュールは，これまで各NIPに送り出したファイルの履歴を管理し，この情報を基に，新しいリクエストが来たときにどのNIPにファイルの送り出しを担当させるかを決定する．この履歴情報は，各NIP毎にこれまでに送り出したファイルのURLをLRUリストとして管理しているものである．NIP内ではファイルのキャッシュをLRUで管理しているので，負荷分散モジュールの持つ履歴情報は実際のNIPのキャッシュの内容の精度の高い予測になっている．実際にはNIP上にキャッシュされているファイルの数と，負荷分散モジュールで管理する履歴の数は正確には一致しないが，NIPから送られてくるヘッダを見て，NIPにキャッシュされているが履歴リストに無い場合や，逆にNIPに無いのに履歴リストにある場合を検出して履歴数を微調整する機能も実装している．

負荷分散モジュールがリクエストを処理するべきNIPを決定する戦略としては，以下のようなヒューリスティクスを用いている．

- そのファイルをキャッシュしているNIPを優先
- 負荷の低いNIPを優先
- リクエストが来たNIPと同じNIPを優先

このような戦略により，一度あるNIPに割り当てられると，同じサーバ上の関連するページをたどっている間は他のNIPに再リダイレクションされることが少なくなる．また，よくアクセスされるデータは複数のNIPにキャッシュされることもある．

7 おわりに

以上，既存のオープンサーバ機をベースとして、サーバ機能を持つ高機能I/Oカードを複数枚実装することにより高いデータ供給能力を実現したWebサーバの性能向上方式について述べた．現在，このWebサーバ試作機の開発を終え，評価を実施している段階である．今後の課題としては，大規模化を目指した並列度の高いデータサーバの構成方法の検討が挙げられる．

参考文献

[1] *Hypertext Transfer Protocol — HTTP/1.1*, RFC2068.

[2] *http://squid.nlanr.net/*

[3] *http://www.stardust.com/*

[4] *NSAPI Programmer's Guide*, Netscape Communications Corporation.