オーディオビジュアル複合情報処理　38－4
(2002. 11. 14)

Webプロキシサーバにおける Webトラフィック制御方式の提案

永見 明久† 中原 雅彦† 西門 隆‡ 野田 文雄†

†株式会社日立製作所 システム開発研究所

〒244-0817 神奈川県横浜市戸塚区吉田町 292 番地

‡株式会社グレープシステム NB 開発センター

〒220-6108 神奈川県横浜市西区みなとみらい 2-3-3 クイーンズタワーB 8F

E-mail: †{nagami, m-nakaha, noda}@sdl.hitachi.co.jp, ‡nisikado@yokohama.grape.co.jp

あらまし 近年，電子商取引やインターネットショッピングといったWebでのサービスが一般的になってきている．これらのサービスの多くは処理の分散化が困難であり，特定のWebサーバにアクセスが集中する場合がある．このような過度なアクセス集中は，Webサーバの処理能力の急激な低下や，サービスダウンを招くことが知られている．そこで，本稿では過度なアクセス集中による処理性能の低下やサービスダウンを回避するため，Webプロキシサーバによるトラフィック制御を提案する．提案方式では，クライアントからのWebアクセス要求をキューにいれて管理する．これにより，Webサーバへの過剰な要求転送を抑制し，Webサーバの安定稼動と高スループットの維持を実現した．また，クライアントからの要求を優先度に応じて異なるキューを用いて管理することで，HTTPレベルでの優先制御を実現した．

キーワード Webプロキシサーバ，トラフィック制御，優先制御，HTTP

An Approach of Web Traffic Control by Web Proxy Server

Akihisa NAGAMI† Masahiko NAKAHARA† Takashi Nishikado‡ and Fumio NODA†

†Systems development Laboratory, Hitachi, Ltd.

292 Yoshida-cho, Totsuka-ku, Yokohama-shi, Kanagawa, 244-0817 Japan

‡NB Development Center, Grape Systems Inc.

Queen's Tower B 8F, 2-3-3 Minatomirai, Nishi-ku, Yokohama-shi, Kanagawa, 220-6108 Japan

E-mail: †{nagami, m-nakaha, noda}@sdl.hitachi.co.jp, ‡nisikado@yokohama.grape.co.jp

Abstract Electronic commerce and electronic shopping have become common services. As a consequence, some web servers have to face access concentration from their customers in a very short time because these quick responses and database reference required services are not easily provided by distributed systems. And it is well-known that access concentration lowers the performances of web servers and can even cause the service down at web servers. In this paper we propose the control of the traffic by the web proxy to avoid these common shortcomings. In our proposed approach, web access requests from clients are managed in queues in order to suppress the excessive web accesses to the web server. Furthermore, the web access requests from clients are managed in different queues according to their priority, thus, a priority control is realized at the HTTP layer.

Keyword Web Proxy Server, Traffic Control, Priority Control, HTTP

1. まえがき

近年，電子商取引やインターネットショッピングといったWebでのサービスが一般的になってきており，従来，営業店窓口で行われていたサービスの多くが，個人消費者も利用可能なWebでのサービスとして提供されるようになってきている．この中でも，チケット予約や銀行・証券取引等のサービスは，時間によるサービス要求の集中が激しく，また，そのサービスにはデータベースアクセスが必要なため，処理の分散化が難しい．そのため，これらのサービスを行う特定のWebサーバにアクセスが集中する場合がある．Webサーバの一般的な特性として，アクセスが集中し負荷が急激に高まると，その処理性能は低負荷時に比べ極端に低下することが報告されている[1-3]．その結果，サ

ービス要求の滞留をさらに拡大させ,最悪の状況では,サービスダウンにまで繋がるという状況が起こっていた.

アクセスが集中した状況では,すべてのサービス要求が満足に処理できず,重要な利用者のサービス要求も,一般の利用者と同様に区別なく応答を待つことになる.しかも,後からサービスを要求しても運良く先に応答が返ることもあれば,長い間待ってもなかなか応答が返らないこともある.特に後者の場合,利用者は何度も再実行を繰り返すため,よりアクセスが集中し負荷を高める結果となっていた.

本稿では,上記のような背景をふまえ,HTTP[4]で行われるWeb要求のトラフィックについて,その制御方式を提案する.提案方式を適用することにより,Webサーバへの過剰な要求の到達を防ぎ,Webサーバの高スループットの維持やWebサーバのダウンの防止を実現する.また,多くの要求が発生している場合においても,重要な利用者からの要求は一般の利用者と区別し,優先したサービスが可能となる優先制御を実現する.

2. 従来のWebアクセスサービスの問題点

従来のWebアクセスサービスでは,クライアントからの要求が増大した場合,次のような方法を適用し対応していた.

(1) 負荷分散装置を導入し,Webサーバを増設することで性能を増強する方法
(2) リバースプロキシサーバを導入し,Webサーバの負荷を削減する方法

ここでは,それぞれの方法の特徴と問題点について検討する.

2.1. Webサーバ増設による性能増強

負荷分散装置を導入することにより,クライアントからの要求を複数(n台)のWebサーバに振り分けることが可能になる.1台当たりのWebサーバの負荷を1/nにすることができるため,Webサーバが1台の場合と比較するとn倍の要求に応答することが可能になる.

必要なサーバ性能を検討する際,基本的にはクライアントからの要求が最も多い場合に合わせてWebサーバの台数を決定する必要がある.さらに,DBアクセスなどを必要とする場合,それらの周辺サーバ群も合わせて増強する必要があるため,コストがかかる方法となる.ところが,クライアントからの要求パターンは,時間,タイミングにより異なり,ピーク性能を要求される時間帯は限られている.そのため,ほとんどの時間帯でサーバ群は遊休状態となってしまう.

負荷分散装置には,単位時間当たりのリクエスト数や同時接続数で制限値を設け,クライアントからの要求を拒否する製品もある.しかし,URL毎に接続数を管理・制限する機能がないため,特定の負荷の高いコンテンツに対して,想定を超えて多くの要求があった場合には,スループットの低下やサービスダウンの可能性があるという問題がある.また,特定のコンテンツに対するアクセス集中によりWebサーバがダウンすると,そのサービスだけではなく他のサービスも提供できなくなるという問題もある.

2.2. リバースプロキシサーバによる負荷削減

当初,Webプロキシサーバはクライアント側ネットワークとインターネットとの間に設置されることが多かったが,近年,Webサーバ側ネットワークとインターネットとの間に設置するリバースプロキシサーバという利用法が増加している.

リバースプロキシサーバは,Webコンテンツのキャッシュやユーザ認証,httpsの暗号解凍など,Webサーバの処理の一部を代行することが目的である.特に,キャッシュサーバとしての利用は,Webサーバへのアクセス数やトラフィックを削減することができるため,静的なコンテンツが中心の場合には,非常に効果的である.

しかし,近年増加しているチケット予約や銀行・証券取引などのサービスでは,動的なコンテンツや,DBアクセスを必要とするコンテンツが多い.これらのコンテンツではキャッシュを用いることができないため,Webサーバの負荷軽減には効果がない.特に,発売時間直後や決算日などは利用が急増することが多く,バースト的に増加したクライアントからの要求は,結局Webサーバに集中するため,Webサーバの負荷を急激に高めるという問題があった.

2.3. 従来法の問題点

上述の従来法では,Webサーバの性能の増強や負荷の削減が可能だが,その性能限界を超える要求が発生した場合には急激な性能低下や要求拒否などが問題となる.これは,Webサーバが高負荷時に次のような原因でWebサーバの性能が低下するからである.

(1) DBアクセスや動的なコンテンツに対する同時要求数の増加により物理メモリが不足するために発生するページングやスワップ
(2) 同時実行プロセスが増加することによるプロセススケジューリングのオーバーヘッド

従来の方法では，これらの問題に対する対策にはならない．そのため，クライアントからの要求がバースト的に増加した場合に，それが Web サーバに影響し，Web サーバの性能低下やサービスダウンの危険性があった．

3. Web トラフィック制御に関する提案方式

本稿では，前記の問題に対する対策として，クライアントから Web サーバへの要求を，Web プロキシサーバがキューを用いて管理し，Web サーバとの接続数を監視しつつ転送タイミングを制御するという Web トラフィック制御方式を提案する．これにより，下記の効果が得られる．

(1) Web サーバへの過度な要求の到達を回避することができるため，サービスダウンの可能性を低下させるとともに，高いスループットを維持することが可能になる．

(2) Web サーバの能力を超えるような要求がバースト的に増加しても，Web プロキシサーバがそれらの要求を受け付けるため，クライアントに対して要求を拒否する可能性が低くなる．

(3) クライアント（利用者）としては，Web サーバの高いスループットが維持されることにより，高速な応答を得ることができる．

(4) 重要な利用者からの要求や重要なコンテンツへの要求を，優先してサービスすることができる．

提案する Web トラフィック制御方式は，図 1 のように Web プロキシサーバが Web サーバとの同時接続数を制御する方式である．クライアントからの要求は，Web プロキシサーバにて受け付けた後，キューに入れて待ち状態にする．Web プロキシサーバは，Web サーバや特定の URL との同時接続数を管理し，それらが設定された最大同時接続数を超えないように転送のタイミングを制御しつつ要求を転送する．

負荷分散装置では，バースト的に増加した要求は拒否することにより Web サーバに過剰な要求が到達することを防いでいたのに対し，提案方式では，要求は Web プロキシサーバにより受け付けられるため，クライアントに対して要求を拒否する可能性は低くなる．

Web サーバとの同時接続数を制限することにより，クライアントからの要求がバースト的に増加してもその影響が Web サーバに到達することもなく，安定稼動が可能である．これにより，Web サーバの高いスループットを維持することが可能であるため，利用者は，要求の転送タイミングを制御しなかった場合と比較すると，高速な応答を得ることができる．

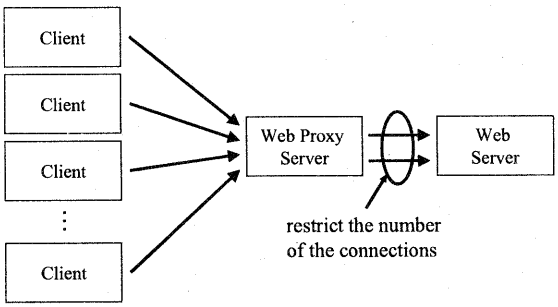

図 1 Web プロキシサーバによる
Web サーバへの接続数制限

さらに，優先度に応じて要求の転送タイミングを変えることにより，HTTP レベルでの優先制御が可能となる．

3.1. キューによる制御

従来の Web プロキシサーバでは，クライアントからの要求は，転送タイミングの制御は行わずに Web サーバに転送を行うが，提案する Web トラフィック制御方式では，Web プロキシサーバはキューを用いて要求を管理し，転送のタイミングを制御する．

図 2 は，Web プロキシサーバから Web サーバへの要求の転送タイミングを表すシーケンス図である．Web プロキシサーバは Web サーバとの同時接続数を管理しており，Web サーバに対して同時に一定以上の要求を転送しないように接続数の管理を行う．Web サーバとの同時接続数の上限を n 本と設定した場合，n+1 番目以降の要求は Web プロキシサーバのキューに入れて待ち状態にする．そして，1 番目の要求に対して Web サーバから応答があると，n+1 番目の要求を Web サーバに対して転送する．このようにクライアントからの同時接続数が n 本以上でも，Web サーバには n 本の要求しか到達せず，Web サーバにバースト的に増加した要求が到達することを防ぐ．

Web プロキシサーバの性能限界以上にクライアントとの接続を行わないために，キューの最大長を設定しアクセス規制を行う．キューの最大長以上に要求が到達した場合には，クライアントに HTTP エラーメッセージを応答し，即座に接続を切断することでアクセス規制を実現する．

以上のように，Web プロキシサーバと Web サーバとの間においては，Web サーバに対する同時接続数と，Web サーバのあるコンテンツに対する同時接続数とを制限することが可能となる．

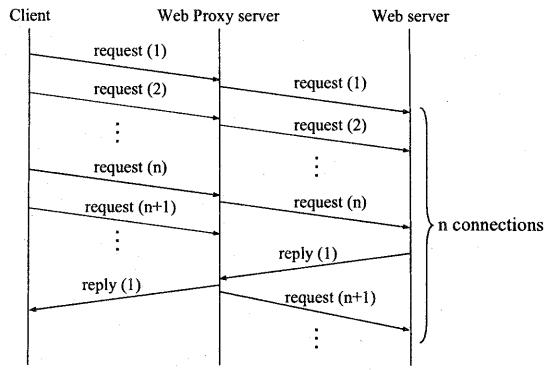

図 2 Web プロキシサーバから Web サーバへの
要求の転送タイミングのシーケンス図

3.2. 優先キューイング

本方式では，優先制御を実現するためにトラフィック制御のためのキューを優先度別に複数持ち，要求を優先度に応じて異なるキューを用いて管理する．以降，本方式を優先キューイングと呼ぶこととする．

この優先キューイングにより，Diffserv[5]などのプロトコルを実装したQoS対応ルータを用いることなく，HTTP レベルでの優先制御を実現することができる．

クライアントからの要求には，下記の条件などにより優先度を付加する．

(1) 要求元アドレス
(2) ユーザ ID
(3) 宛先ホストアドレス
(4) 宛先 URL

これらの情報により優先度が異なる要求を，その優先度に応じて異なるキューを用いて管理する．そして，優先度の高いキューを用いて管理している要求から順に転送することにより，要求が多く発生している場合にも，優先度の高い要求に関しては迅速なサービスを提供することが可能となる．

3.2.1. 優先キューと一般キュー

優先キューイングでは，図 3 のように優先度つきの要求を管理する n 個の優先キューと，すべての要求を管理する一般キューとを持つ．n 個の優先キューにより，n+1 段階の優先制御が可能となる．

要求は Web プロキシサーバに到達した順に一般キューに格納する．さらに，優先度つきの要求は，優先度に応じた優先キューに格納する．ここで，Web サーバとの接続数がその上限値に達していない場合には，即座に要求を転送する．しかし，接続数が上限値に達している場合には転送を行わず，要求を待ち状態にする．

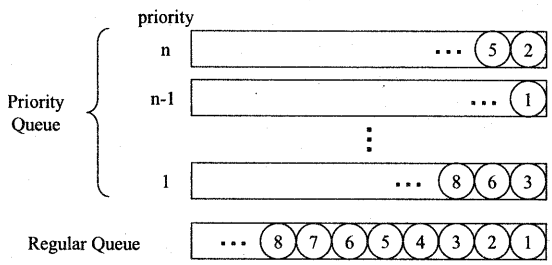

図 3 優先キューと一般キュー

3.2.2. 転送する要求の検索方式

ここでは，Web サーバから応答が返され要求の転送が可能になった場合に，キューに格納された要求の中から次に転送する要求を検索する方法について述べる．

まず，要求が格納されている一番優先度の高いキューを検索し，そのキューの先頭に格納されている要求を転送する．この要求の宛先の URL に対する同時接続数が設定されている場合には，その上限値を超えないことを確認する．もし，超えている場合には，そのキューの先頭から 2 番目の要求に関して同様に宛先の URL の接続数を調べる．要求の格納されている一番優先度の高いキューに，転送できる要求がない場合には，次に優先度の高いキューを同様に検索する．このように，優先度の高いキューから順に転送できる要求があるか検索し，転送可能な要求があれば，それをキューから取り出し転送する．

優先キューから取り出すのと同時に，一般キューからも同一の要求を取り出す．それが一般キューの先頭ではなかった場合，先頭の要求が追い越された回数を増やす．この値は後述の沈み込み防止方式にて利用する．

3.2.3. 沈み込み防止方式

前述の方法では優先度の高い要求についてのみ転送が行われ，優先度の低い要求に対してサービスが提供されずに，その要求が沈み込んでしまう可能性がある．このような要求の沈み込みを防止するため，一般キューの先頭にある要求の追い越された回数が一定回数以上になるか，もしくは一般キューの先頭になってから経過した時間が一定時間以上になったら，一般キューの先頭にある要求を転送する．

本方式により優先度の高い要求と優先度の低い要求が混在する状況においても，優先度の低い要求に対してサービス提供が可能となる．

また，キューに入れてから一定時間以上転送を待たせている場合，タイムアウト処理を行うことにより利用者が必要以上に待つことを防ぐことができる．

4. 性能評価

本提案のトラフィック制御方式を実装し，性能評価を行った．本章では性能測定の方法や環境について述べ，測定結果について評価・考察を行う．

4.1. 測定環境

性能測定に用いた計算機の性能を表1に，ネットワーク構成を図4に示す．OSはLinux†（kernel-2.4.2）を基に，処理可能な最大ソケット数とソケット待ち受けキューの最大長を増やすため，表2に示すパラメータについてチューニングを行った．

Webサーバのコンテンツには，DBアクセスを模擬するCGIコンテンツを用いた．

4.2. 評価内容

クライアントからの要求のリクエストレートと同時接続数を変えて，応答時間とスループットの測定を行った．Webプロキシサーバによる Webサーバへの同時接続数は30[req]に設定した．キューの最大長の設定によるアクセス規制は行わなかった．

4.2.1. リクエストレートによる評価

クライアントのソフトウェアに，httperf-0.8[6]を用い，リクエストレートを変えて，Webプロキシサーバを経由して Webサーバにアクセスした場合と Webサーバに直接アクセスした場合とでスループットと応答時間を測定した．

4.2.2. 同時接続数による評価

独自に開発したクライアントソフトウェアを用い，同時接続数を変えて，Webプロキシサーバを経由してWebサーバにアクセスした場合とWebサーバに直接アクセスした場合とでスループットを測定した．

表1 計算機の性能

	CPU	Memory
Webサーバ	1GHz	1GByte
Webプロキシサーバ	1GHz	1GByte
クライアント	1GHz	1GByte

表2 カーネルパラメータ

パラメータ	チューニング前の値	チューニング後の値
INR_OPEN	1024	10240
NR_OPEN	1024	10240
__FD_SETSIZE	1024	10240
SOMAXCONN	128	2048

† Linuxは，Linus Torvalds氏の米国およびその他の国における登録商標または商標である．

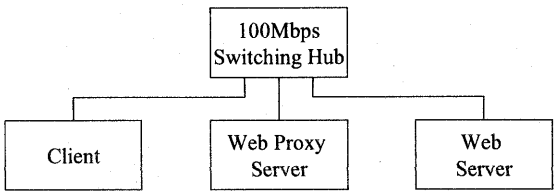

図4 ネットワーク構成図

4.2.3. 同時接続数による優先制御の評価

4.2.2 項で説明した性能測定用のソフトウェアを用い，優先制御の測定を行った．同時接続数を変えて，Webプロキシサーバを経由して Webサーバにアクセスした場合と Webサーバに直接アクセスした場合とで応答時間を測定した．Webプロキシサーバを経由して Webサーバにアクセスした場合には，優先要求と非優先要求とでそれぞれ応答時間を測定した．

4.3. 結果・考察

図5，図6にリクエストレートとスループット，応答時間の関係を示す．リクエストレートが80[req/sec]までは，スループット，応答時間に違いは見られないが，Webサーバに直接アクセスした場合には，100[req/sec]を超えるとスループット，応答時間共に劣化した．応答時間の増加率はリクエストレートの増加に伴い高くなり，150[req/sec]を超えると30秒以内に応答が得られなかった．一方，Webプロキシサーバを経由した場合には，スループットを60[rep/sec]に維持することができた．リクエストレートが150[req/sec]の場合で，Webサーバに直接アクセスした場合と比較すると，応答時間を25%程度に改善することができた．また，リクエストレートの増加に対する応答時間の増加率を一定に抑えることができた．

図7，図8に同時接続数とスループット，応答時間の関係を示す．同時接続数が50[req]までは，応答時間，スループットに違いは見られないが，Webサーバに直接アクセスした場合には，50[req]を超えると応答時間，スループット共に劣化した．一方，Webプロキシサーバを経由した場合には，スループットを60[rep/sec]に維持することができた．同時接続数が60[req]の場合で，Webサーバに直接アクセスした場合と比較すると，応答時間を10%以下に改善することができた．

図8のWebプロキシサーバを経由した場合について，優先要求と非優先要求の応答時間を比較すると，優先要求は，非優先要求よりも高速な応答を得られることがわかる．優先要求の応答時間は，同時接続数が増加しても劣化が見られないため，効果的な優先制御であるといえる．

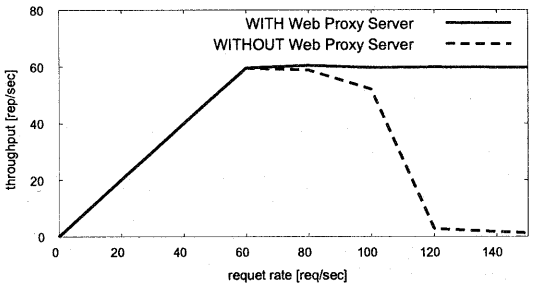

図 5 リクエストレートとスループットの関係

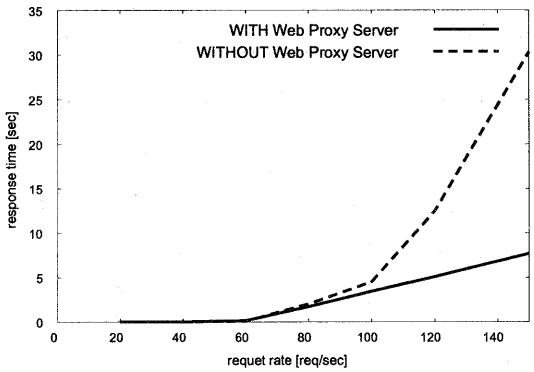

図 6 リクエストレートと応答時間の関係

5. むすび

本稿では，Web アクセス要求をキューに入れて管理する Web トラフィック制御方式を提案した．

提案方式により，Web サーバへの過剰な要求の到達を防止し，Web サーバの高スループットの維持や，Web サーバダウンの回避が可能になる．

また，優先キューイングという優先度別のキューを用いて要求を管理する手法により，重要な利用者からの要求や重要なコンテンツへの要求を一般の要求と区別して，HTTP レベルでの優先制御を行うことが可能になる．

実験を通じて，提案方式を適用することにより，リクエストレートが 150[req/sec]の場合に応答時間を 25%程度に改善できることを確認し，同時接続数が 50 以上の場合に応答時間を 10%以下に改善できることを確認した．また，高負荷時にも高スループットを維持できることを確認した．さらに，優先キューイングにより，優先制御が行われることを確認した．

謝辞 本研究の実装にあたり御尽力頂いた日立ＩＮＳソフトウェア株式会社栗城信明氏に感謝いたします．

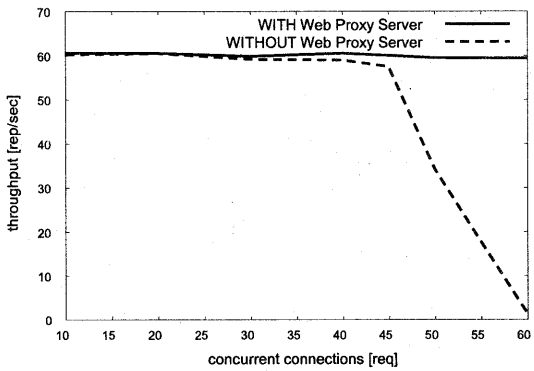

図 7 同時接続数とスループットの関係

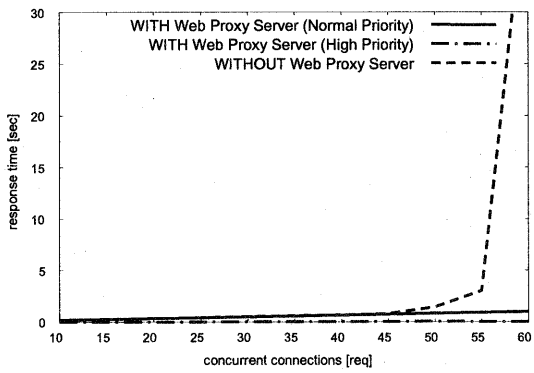

図 8 同時接続数と応答時間の関係

文　献

[1] Mark S. Squillante, David D. Yao, and Li Zhang, “Web Traffic Modeling and Web Server Performance Analysis”, Proceedings of the 38th Conference on Decision and Control, pp.4432-4439, Phoenix, Arizona USA, December 1999.

[2] 藤田靖征，村田正幸，宮原秀夫，“プロキシサーバにおけるキャッシュ機能を考慮した Web システムのモデル化と性能評価”，電子情報通信学会論文誌 B，Vol. J82-B, No.8, pp.1449-1461, 1999 年 8 月

[3] 菊池慎司，松本晋一，佐藤義治，青柳好織，安達基光，勝山恒男，“異なるリソース間の相互作用を考慮した Web サーバシステムのモデル化”，電子情報通信学会技術研究報告，信学技報 CQ2001-65, pp.1-6, 2001 年 9 月

[4] R. Fielding, UC Irvine, J. Gettys, Compaq/W3C, J. Mogul, Compaq, H. Frystyk, W3C/MIT, Xerox, P. Leach, Microsoft, T. Berners-Lee, W3C/MIT, “Hypertext Transfer Protocol – HTTP/1.1”, RFC2616, June 1999

[5] D. Grossman, Motorola, Inc., “New Terminology and Clarifications for Diffserv”, RFC3260, April 2002

[6] David Mosberger and Tai Jin, “httperf --- A Tool for measuring Web Server Performance”, available at http://www.hpl.hp.com/personal/David_Mosberger/httperf.html