オーディオビジュアル複合情報処理　38－5
(2002. 11. 14)

アプリケーションサーバ連携型 Web プロキシの実装と評価

竹島 由晃† 中原 雅彦† 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

E-mail: †{takesima, m-nakaha, noda}@sdl.hitachi.co.jp

あらまし Web 通信システムを利用したサービスが多種多様化している．今後は，Web コンテンツの提供者と，そのコンテンツを中継するキャリア(通信事業者)とが連携して，付加価値の高い新サービスをユーザに提供する「連携型サービス」が重要になると考える．そこで，本論文では，このような新サービスの実現を目的として，アプリケーションサーバ連携型 Web プロキシ方式を提案する．本方式では，Web サーバとクライアント間の通信を，Web プロキシがアプリケーションサーバに転送し，付加的処理したデータをクライアントに送信する．性能評価により，コンテンツレベルでの付加価値サービス提供を実現し，性能劣化がほとんどないことを確認した．

キーワード WWW，Web プロキシ，付加価値サービス

An Implementation and Evaluation of Service-Enhancing Web Proxy Server

Yoshiteru TAKESHIMA† Masahiko NAKAHARA† and Fumio NODA†

† Systems Development Lab. Hitachi, Ltd. 292 Yoshida-cho, Totsuka-ku, Yokohama-shi, Kanagawa, 244-0817 Japan

E-mail: †{takesima, m-nakaha, noda}@sdl.hitachi.co.jp

Abstract Value added services can now be offered to users thanks to the cooperation between of the Web proxies and application servers. In our proposed approach, transactions between and client and a Web server go through the Web proxy which forwards the response from the Web server to the application server. This application server appends value added data to the contents and sends them back to the proxy. Our evaluation of the implementation of this function at the Web proxy shows that value added services at the content level can be offered almost without any degradation of the performance.

Keyword WWW, Web Proxy Server, Value Added Services

1. はじめに

近年，Web ブラウザ搭載型携帯電話の急激な普及により，キャリア(通信事業者)による Web 通信サービスを利用したサービスの利用者が急増してきている．携帯電話による Web 通信サービスでは，キャリア側で様々なサービスを提供している．これらのサービスは，Web アクセス中継機能やポータルメニューの提供，情報料回収代行サービスなどであり，コンテンツ提供者が提供するコンテンツをキャリアが単に中継するだけの Web インフラストラクチャ提供サービスがメインである．しかし今後は，Web でのサービスの多種多様化に伴い，コンテンツ提供者，サービス代行業者といったサービス提供者とキャリアが連携することによって，ユーザに対し新しいサービスを提供する，といった「連携型サービス」形態への要求が出てくるものと考えられる．このような背景の下，キャリアとサービス提供者による連携型 Web 付加価値サービスの実現を目的とし，Web 付加価値サービスネットワーク基盤の検討を行った．以下本稿では，上記連携型 Web 付加価値サービスの方式の設計及び評価結果について述べる．第 2 章では，本研究の全体アーキテクチャについて説明する．第 3 章では，提案方式について説明し，第 4 章では，その性能評価の結果について説明する．最後に第 5 章では，本論文のまとめを行う．

2. Web 付加価値サービスシステム

2.1. 従来型システム

キャリアの従来型の Web 通信サービスでは，ポータルメニュー提供サービスの他に，図 1に示すように，ユーザとコンテンツ提供者間に Web プロキシを配置して，Web アクセス中継サービスや有料コンテンツの課金回収代行サービスなどを提供している．すなわち，キャリア側でのサービスは，基本的には Web 中

継インフラストラクチャの提供サービスがメインである．しかし，このような枠組みでは，ユーザに対するサービスは，コンテンツ提供者とユーザ間での End-to-End サービスがベースとなり，キャリアによるユーザへの提供サービスの幅は，非常に限られてしまう．

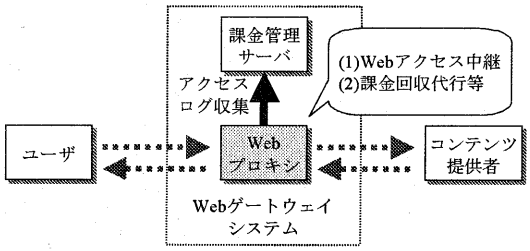


図 1： 従来型 Web 通信サービス

2.2. 連携型 Web 付加価値サービスシステム

今後の Web での付加価値サービスでは，サービスの多種多様化に伴い，コンテンツ提供者，サービス代行業者といったサービス提供者と，キャリアが連携することによって，ユーザに対し柔軟なサービスを提供する，といった連携型サービス形態への要求が出てくるものと考えられる．ここで，Web 付加価値サービスとは，コンテンツの内容を個々のユーザ向けにカスタマイズしたり，フィルタリングしたりするようなサービスのことを言う．たとえば，HTML への広告バナー挿入や実行プログラムのウィルススキャン，言語翻訳，コンテンツフィルタリングなどである．

連携型サービスを実現するためには，キャリアは，ネットワーク中にクライアントと Web サーバ間の Web アクセスを中継する装置上で，Web データを加工する仕組みが必要となる．しかし，キャリアの持つネットワークは常に非常に多くのユーザに利用されており，上記のような仕組みを実現するためには，2.3節で述べる要件を満たすシステムを構築する必要がある．

2.3. システム必要要件

キャリア側で Web 付加価値サービスを提供するためには，以下のような必要要件がある．

(1) 機能拡張性
　様々なサービス要求があるため，それらの要求に幅広く対応する必要がある．そのため，付加価値サービス機能の追加・変更・削除は容易に行える必要がある．

(2) スケーラビリティ(規模拡張性)
　キャリアのコアネットワークはトラフィックが集中するため，高トラフィック時でもシステムダウンさせずに，また応答性能を極力維持する必要がある．

(3) サービス無停止性
　キャリアのサービスには，常にサービスを停止してはならないという高信頼性が必要とされる．サービスの追加・変更・削除時でも，その他のサービス提供を停止しない．

(1)の機能拡張性を提供する方法として，最も一般的なものに，Web プロキシ自体に付加価値サービス機能を機能拡張として実装する方法がある．この方法は，プロキシのソフトウェアに必要なコードを追加すれば良いだけであるので，付加価値サービスの実装の容易さという面では最も優れている．しかし，性能面から見ると，付加価値サービス機能の処理負荷や処理時間が無視できないため，Web プロキシ自体のパフォーマンスに影響を及ぼしてしまう．また単一ソフトウェアに多種多様な機能を拡張する設計にしてしまうと，そのソースコード量が莫大となり，ソフトウェアの保守や更なる機能拡張が次第に困難になってしまうという欠点を持つ．

また，Web プロキシがローカルマシン内で外部のプログラムを実行させて，サービス対象のリクエストやレスポンスデータに処理を行わせる方法もあるが，CPU などローカルのリソースを消費する面ではプロキシ内実装の方法と同じである．また，サービスを実現するプログラムを全てローカルマシンに用意して，実行させておく必要がある．そのため，サービスが増加するほど運用が困難になり，性能は低下していく．

3. アプリケーションサーバ連携サービス制御方式

2.3節の必要要件を満たすため，IETF の OPES(Open Pluggable Edge Services)ワーキンググループでは，外部呼び出しサーバ(remote callout server)と Web プロキシが連携して付加価値サービスを提供する方式を提案している[4]．提案されている Web プロキシは，何らかのルールに従ってメッセージを転送し，外部呼び出しサーバ側でサービス処理を行うことにより，Web プロキシの負荷の上昇を回避し，パフォーマンスの極端な低下を防ぐ．更に，付加価値サービス機能の追加・変更・削除を，該当する外部呼び出しサーバの置き換えなどにより，Web プロキシでの処理や他の付加価値サービス機能に影響を与えることなく行うことが可能になる．

そのため，本研究では，Web クライアントと Web サーバ間に，Web 通信を中継する Web プロキシと，付加価値処理を提供するアプリケーションサーバを配置して，2.3節の必要要件を満たす付加価値サービスを提供する Web システムを実現することを目的とし

た．そのため，Web プロキシには，サービス機能を外部のアプリケーションサーバ(外部呼び出しサーバ)に分離するための機能を実装した．また，アプリケーションサーバは，Web プロキシからネットワーク経由で送信されたデータに対して付加価値処理を加え，Web プロキシに処理結果を応答する，Web サーバの一種として実装した．上記のシステムを用いることにより，アクセス先の URL やファイルの拡張子等をキーワードとして，キャリアとサービス連携者間でのサービス連携が可能となる．

図 2に本方式の概念図を示す．

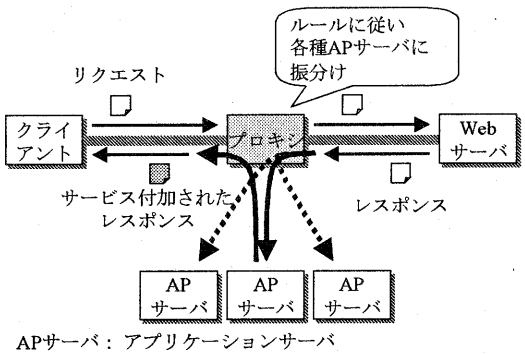


図 2: アーキテクチャ概要

3.1. アプリケーションサーバ連携 Web プロキシ

本研究の Web プロキシは，Web クライアントと Web サーバ間でデータを中継する際に，中継データに付加価値処理を加える．ただし，実際には Web プロキシ自体は処理を行わない．本システムは，幾つかのパラメータがサービスの発動条件にマッチした場合に，Web プロキシがコンテンツデータをアプリケーションサーバに転送し，付加処理を加えられたデータをクライアントもしくは Web サーバに送信することで，付加価値処理を実現する．サービス発動条件のトリガとなるパラメータは，リクエスト先コンテンツの URL やファイル拡張子などがある．

本方式を実現するため，Web プロキシに以下の二つの機能を実装する．

(1) サービス制御機能

(2) アプリケーションサーバ呼び出し機能

(1)のサービス制御機能は，HTTP 中継処理中のセッションに対して，サービスの検索と発動を制御する機能である．また，(2)のアプリケーションサーバ呼び出し機能は，あるプロトコルに従ってアプリケーションサーバと通信する機能である．

図 3に，本 Web プロキシの機能概要を図示する．

以下では，これらの機能の詳細について述べる．

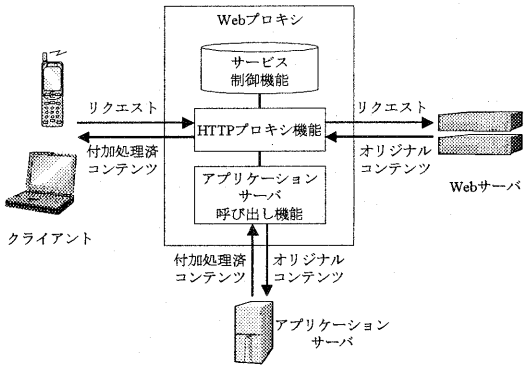


図 3： Web プロキシ機能概要

3.1.1. サービス制御機能

サービス制御機能は，リクエストやレスポンスデータに対して，どのサービスの適用を行うかを決定する機能である．本機能は，サービス制御ルールテーブルにより，サービス制御を行う．このテーブルは，以下のような情報を管理する．

(a) サービス発動条件

(b) (サービス発動時の)アプリケーションサーバアクセス情報

(c) サービス名

(d) サービス発動タイミング

サービス制御テーブルの例を，表 1 に示す．

(a)のサービス発動条件は，サービスの発動を行うかどうかのトリガ情報である．今回の実装では，リクエスト先の URL やファイル拡張子がテーブルに記載の文字列に一致しているかどうかがサービス発動の条件となる．

(b)のアプリケーションサーバアクセス情報は，アプリケーションサーバにアクセスするための情報である．今回の実装では，アプリケーションサーバアクセス情報を，『アプリケーションサーバ URL』として URL 形式で設定する．

表 1： サービス制御テーブル

ID	発動条件	アプリケーションサーバ URL	サービス名	発動タイミング
1	URL= "http://www2.sdl.hitachi.co.jp /index.html"	http://apsvr3:8081/cm_insert	cm_insert	コンテンツ受信後
2	拡張子 = ".exe"	http://apsvr1/virus_scan.cgi	virus_scan	コンテンツ受信後
3	拡張子 = ".jar"	http://apsvr2/function_check	function_check	コンテンツ受信後

(c)のサービス名は，サービス内容を識別するための文字列である．表 1 の例では，"cm_insert"や"virus_scan"などが該当する．

(d)のサービス発動タイミングは，本 Web プロキシが，どの時点でアプリケーションサーバへのメッセージ転送を行うかが設定される．タイミングとしては，クライアントからの『リクエスト受信後』か Web サーバからの『レスポンス受信後』の二つがあるが，今回の実装では，『レスポンス受信後』のみとした．

3.1.2. アプリケーションサーバ呼び出し機能

アプリケーションサーバ呼び出し機能は，3.1.1節のサービス制御機能でサービス発動する際に，リクエストメッセージやレスポンスデータを，Web プロキシとアプリケーションサーバ間で取り決められた通信プロトコルに従うフォーマットに直し，3.1.1節(b)で述べたアクセス先のアプリケーションサーバに対して接続する機能である．本機能により，本 Web プロキシは，アプリケーションサーバに対してクライアントとして振舞う．Web プロキシとアプリケーションサーバ間の通信プロトコルについては，3.3節で述べる．

3.2. アプリケーションサーバ

アプリケーションサーバは，Web プロキシとアプリケーションサーバで取り決められた通信プロトコルを解釈し，また本プロトコルで定められたフォーマットに従ってメッセージを構築する．通信プロトコルについては，3.3節で述べる．一般的な実装の形態としては，独自サーバを構築することにより実現するケースと，Web サーバ上で動作する CGI 等で実現するケースが考えられる．

3.3. アプリケーションサーバ呼び出しプロトコル

Web プロキシとアプリケーションサーバ間の通信プロトコルとして一般的なものには，従来の HTTP の他に，ICAP(Internet Content Adaptation Protocol)と呼ばれるプロトコル仕様がある[1][3]．HTTP はインターネットで標準的に使用されているプロトコルであり，数多くの実装がある．また，CGI などのプログラムを Web サーバに設置することで，容易に Web アプリケーションを構築することができる．一方，ICAP は，Web プロキシ(ICAP クライアント)と，付加価値サービスを提供する専用サーバ(ICAP サーバ)間で RPC を実現するプロトコルであり，現在 iCAP Forum や IETF で議論が行われている．ICAP のプロトコル仕様は，HTTP に似通ってはいるが，HTTP とは異なるプロトコルであり，HTTP よりも高機能である．しかし，ICAP サーバを構築するためには，サーバを ICAP 独自仕様で実装する必要があり，既に広まっている HTTP ベースで CGI などを用いて構築するよりも，Web プロキシ及びアプリケーションサーバの開発コストは大きくなるものと考えられる．本研究では，Web プロキシとアプリケーションサーバの実装の容易さを重視して，HTTP ベースの通信プロトコルを採用した．

本 Web プロキシは，アプリケーションサーバに対し，HTTP/1.1 での POST 形式のリクエストを送信する．具体的には，POST 形式のリクエストメッセージのボディ部中に，クライアントからのリクエストヘッダ，Web サーバからのレスポンスのヘッダ，及びボディのデータを，図 4のようにカプセル化してアプリケーションサーバに送信する．また，本プロキシは，このメッセージを，chunk 形式で転送エンコード(Transfer-Encoding)して送信する．本リクエストメッセージの一例を，図 5に示す．

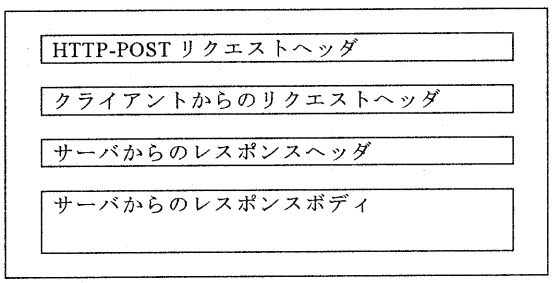


図 4： アプリケーションサーバ
呼び出しプロトコルフォーマット

```
POST http://apserver:8081/SampleAP HTTP/1.1
Host: apserver:8081
Transfer-Encoding: chunked
Connection: close

60
GET http://websvr:80/test.html
Host: websvr
Proxy-Connection: close


8d
HTTP/1.1 200 OK
Server: webserver
Cache-Control: no-cache
Connection: close
Content-Length: 1500


5dc
************************************(中略)
0
(メッセージ終了)
```

図 5： プロトコル例

4. 性能評価および考察

本実装の Web プロキシの性能を評価するため，クライアントからのリクエスト送信レートを変動させて，性能測定を行った．なお，ここでは，クライアントからのリクエスト及びサーバからのレスポンスは，全てアプリケーションサーバに転送されるように，Web プロキシを設定している．

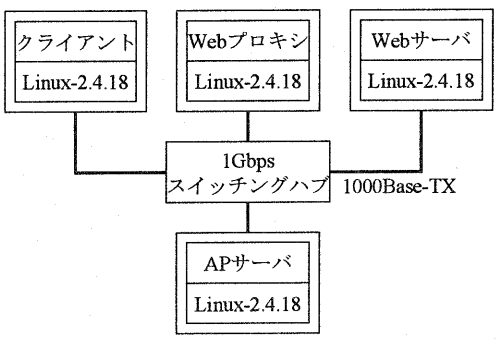
4.1. 実験環境

性能測定に用いた計算機環境を表 2に，ネットワーク構成を図 6に示す．

また，各計算機の OS は Linux†(kernel-2.4.2)をベースとしており，数種のパラメータについて，表 3のようなチューニングを行っている．

表 2： 計算機環境

	CPU	メモリ	OS
Web サーバ	1GHz	1GB	Linux-2.4.18
Web プロキシ	1.2GHz×2	2GB	Linux-2.4.18
クライアント	1GHz	512MB	Linux-2.4.18
アプリケーションサーバ	1.2GHz×2	2GB	Linux-2.4.18



APサーバ：アプリケーションサーバ

図 6： ネットワーク構成図

表 3： カーネルパラメータチューニング

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

本研究の Web プロキシは独自実装であり，表 4の環境で実装及び設定した．本来はメモリベースでキャッシング を行うが，本実験ではキャッシングは行わないように設定した．Web サーバとクライアント間で，通常の Web プロキシとして動作させた場合のスループットは，Web サーバのコンテンツサイズが 1.5KB のとき，約 500～1,000[req/sec]程度であった．

表 4：Web プロキシ実装環境及び設定

OS：	Linux-2.4.18
実装言語：	C 言語
キャッシュ：	なし
persistent 接続：	Web サーバ側，クライアント側共になし
スループット：	平均コンテンツサイズ 1.5KB 時で，約 1,000[req/sec].

Web サーバも独自実装であり，以下の環境で実装及び設定した．実際の Web プロキシと Web サーバ間はインターネットを介しているものと考えられるため，遅延のある実際のサーバ，ネットワーク環境を模擬するため，ダミーの遅延処理(10msec)を実装した．

表 5： Web サーバ実装環境及び設定

OS：	Linux-2.4.18
実装言語：	C 言語
処理：	リクエスト受信後，ダミーの遅延処理(10msec)を行い，固定メッセージ(1.6KB)を応答.
スループット：	約 100[req/sec].

アプリケーションサーバは，以下の環境で実装を行った．実際のネットワーク構築では，アプリケーションサーバは Web プロキシと同一 LAN 上に設置されるものと考えられ，高速に応答できるので，遅延処理は実装していない．

表 6： アプリケーションサーバ実装環境及び設定

OS：	Linux-2.4.18
実装言語：	C 言語
Web プロキシ間プロトコル：	HTTP-POST 形式
処理：	リクエストメッセージ受信完了まで待機後，固定メッセージ送信(1.8KB).
スループット：	700[req/sec]

また，クライアントソフトウェアには，httperf-0.8 を利用した[2]．httperf のリクエスト送信レートを変動させ，Web プロキシ経由で Web サーバにアクセスした場合について評価を行った．

4.2. リクエスト送信レートに対する応答性能

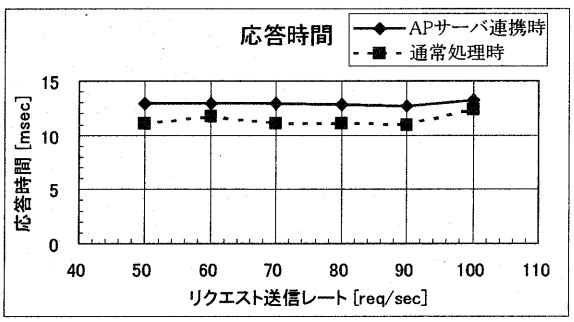
httperf のリクエスト送信レートは，50～100[req/sec]

† 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．

として，クライアント側(httperf)で平均応答時間を測定した．Web サーバのスループット仕様が 100[req/sec]であるため，100[req/sec]を超える送信レートでは，Web プロキシから Web サーバへのコネクションがタイムアウトしてしまい，それ以降の測定が不可能になる．そのため，測定範囲を 100[req/sec]以下とした．比較のため，アプリケーションサーバと連携せずに，Web サーバとクライアント間で通常のプロキシとして動作させた場合の平均応答時間も測定した．測定結果を図 7に示す．

図 7より，付加価値処理をアプリケーションサーバで処理した場合は，リクエスト送信レートが 50～100[req/sec]時点で平均応答時間が 12.7～13.3[msec]，通常プロキシ処理の場合は 11.0～12.4[msec]と，アプリケーションサーバを経由させたことによる遅延の増加は約 1msec，10%程度であった．これより，通常プロキシ処理時と比較して，アプリケーションサーバ連携時での処理性能の劣化は非常に小さいと言える．

図 7: Web プロキシの応答性能



5. まとめ

本論文では，キャリアとサービス提供者による連携型付加価値サービス基盤を構築するため，Web プロキシとアプリケーションサーバとの連携による付加価値サービス提供方式の実装及び評価を行った．Web プロキシとアプリケーションサーバ間の通信プロトコルは，アプリケーションサーバの実装の容易さを考慮して，HTTP-POST ベースで設計した．実装した Web プロキシを用いて応答性能を評価した結果，アプリケーションサーバを経由させることによる性能劣化は，実環境を模擬したネットワーク構成において，約 10%程度と小さいことを確認した．本方式により，コンテンツレベルでの付加価値サービス提供を，高速に実現できることが分かった．今後の課題としては，今回実装した Web プロキシを用いたサービスアプリケーションの開発があげられる．

謝辞 本研究のプロトタイプ実装にあたり，御尽力頂いた日立 INS ソフトウェア株式会社 山田義之氏に感謝いたします．

文 献

[1] J. Elson, 他, “ICAP the Internet Content Adaptation Protocol”, IETF Internet-Draft, available at http://www.ietf.org/internet-drafts/draft-elson-icap-01.txt

[2] 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

[3] Andre Beck and Markus Hofmann, "Enabling the Internet to Deliver Content-Oriented Services", 6th International Workshop on Web Caching and Content Distribution, Boston, Massachusetts, USA, June 20-22, 2001.

[4] IETF OPES(Open Pluggable Edge Services} working group, available at http://www.ietf-opes.org/