WWWアクセス非同期化方式の検討

近藤 毅† 高橋 泰弘† 松井 進†

モバイル環境においては，通信コスト低減や低回線品質の影響を受け難くするために，非同期アクセスが注目されている．我々は，既存の処理のデータ通信を非同期化する方法の検討を進め，特に既存 Web ブラウザや Web のサービスを変更することなく非同期アクセスを実現する Web 代理アクセス機能の実現方式に関して比較検討を行った．

本稿では，Web 代理アクセス機能のクライアント側の実現方式として，サーバ方式とプロキシベース方式の2方式について提案し，両方式の比較検討を行った．その結果，イントラネット内の特定サービスを非同期化する場合はサーバ方式が有利であり，インターネットで提供される一般的サービスを非同期化する場合は，プロキシベース方式が適しているとの結論を得た．

Asynchronous Access Method for WWW

Takeshi Kondou†, Yasuhiro Takahashi† and Matsui Susumu†

As mobile computing environment, Communication cost is high and communication line is unreliable. To avoid the disadvantage, there are some technique about asynchronous access method. Our research is the asynchronous access method for data communication about web access agent function without changing web browser and web service. In this paper we propose two implementation method named server method and proxy based method in the client computer, and discus the proposed method. As the discussion, Server method is suit for some service in the Intranet, and proxy based method is suit for some service in the Internet.

1. 背景と目的

計算機の製造技術の進歩によってPCの小型化／軽量化が進み，ノート型やサブノート型と呼ばれるコンパクトで持ち運び容易なPCが近年普及してきている．一方，我が国においても無線通信インフラが整備され，デジタル携帯電話やPHS等の無線通信サービスの利用者が急激な勢いで普及しつつある．

また，コンピュータ通信の世界においてはインターネットの急浮上により，従来の通信形態が革命的な変化を遂げつつある．特に企業向けの通信ネットワークにおいては，従来の専用線による通信ネットワークから，インターネットを使用した通信網構築への変化が進行中であり，このような企業内ネットワークをインラネットと呼びこのイントラネットを整備する企業が増えている．

前記背景により，無線網を使用して場所に制約されることなくコンピュータ通信を行うモバイルコンピューティングが現在注目されている。このようなモバイルコンピューティングにおいて，イントラネットと接続し，企業内の情報リソースにアクセスするとき，モバイル環境での通信インフラである無線通信網は，有線と比べると回線速度が制限されかつ通信費が高いという課題がある．

このようなモバイルコンピューティング環境で使用される小型携帯PCやクライアント端末において，動作する通信アプリケーションは種々多様である．これらアプリケーションソフトウェアを変更することなく前記課題を解決する非同期処理化の研究を行っている．

特に，本論文においては，Web 代理アクセス機能のクライアントにおける実現方式について述べる．

† 株式会社 日立製作所 システム開発研究所
Systems Development Laboratory, Hitachi,Ltd.

2. WWWにおける処理の非同期化

モバイル環境やインターネット環境において高まるネットワーク通信路のコスト低減と，ネットワークに接続していないときの処理をサポートすることが必要とされており，この目的のため，Webブラウザ APやWebのサービスを改造することなくTCP/IP通信ネットワークに流れる HTTP データを非同期通信化するミドルソフトウエアが望まれている．

一般的にこのような非同期化は，通信プロトコルによる制約とインタセプトの方法との2つの理由から困難であるため，非同期通信基盤上が提供する API 上に非同期用の新たなアプリケーションソフトを開発しなければならない．

しかし，Web ブラウザにおいては，アプリケーションの通信プロトコルである HTTP[1]がステートレスなプロトコルであるため，非同期化に必要な要求と応答との分離がプロトコル的に容易であるという特徴を備えている．

クライアントを改造することなく非同期化を実現するには通信データをインタセプトすることが必須となるが，Web ブラウザにおいては，要求をインタセプトする方法としてプロキシを使う方法がよく知られ，使われている[2]．これは，既存の Web ブラウザ AP がプロキシに対応しているため，クライアント内にプロキシ(ローカルプロキシと呼ぶ)を動作させ，ローカルプロキシで HTTP 要求・応答を取り込むことができる．

非同期化においては，前述のようにしてインタセプトした HTTP 要求を一旦ストックし，非同期通信基盤をつかって Web 代理アクセスサーバ側に要求を転送する．Web 代理アクセスサーバは，WWW サーバにアクセスし，応答を非同期通信基盤でブラウザに返す。

ブラウザは，要求を出した後，Web 代理アクセスサーバからのの応答が返ってくるまでずっと応答待ちの状態になる．

Web 代理アクセスサーバの機能としては，単なる HTTP の仲介機能だけでなく，通信回数の削減という観点から，以下の機能を備える．

・リンク先読み機能

サーバ側に要求を出した後，その要求に対する応答の HTML だけでなく，応答 HTML でリンクされているページファイルも一括してクライアント側へ送付する機能である．これにより，クライアントとサーバ間の通信回数を軽減する．

また，非同期通信基盤では，障害などにより通信回線が切断されたとき，これによる処理の中断をユーザからマスクし，通信回線が最接続されたとき，途中からデータの転送を行う機能を備える．この機能により，通信障害による HTTP 応答の受信エラーがなくなり，ユーザからは，時間がかかるが必ず応答が返ってくるようになる．Web ブラウザでのユーザ操作は，通常のアクセス操作と全く同一であり，モバイル環境を意識することはない．

このように従来の環境と全く変わらないユーザオペレーションでは，ブラウザの画面が応答が返るまで待ちの状態となるため，要求を出してその応答を受け取る前にブラウザ AP を終了したり，端末の電源をきることはできない．つまり，ブラウザ画面から要求を出して，これを一旦まとめて蓄積しておき，バッテリ消費を押さえるため，端末のパワースイッチを OFF にし，後で GW サーバと通信するという使い方はできない．

このように一旦ブラウザから要求を出した後は，アプリケーションを終了するだけでなく，端末の電源を切ることもできる非同期アクセス方法を完全非同期化と呼び，これの Web ブラウザ搭載クライアントにおける実現方式を検討した．

3. クライアント機能とその実現方式

3. 1 クライアント機能

前述した Web アクセスの完全非同期化において必須である機能をまず述べる．

(1)プレロード機能

モバイル環境において頻繁にアクセスするページ情報は，モバイル環境に移動する前に取得し，これをクライアント端末に蓄積しておき，オフラインの状態で参照できるようにする機能が必要である．

(2)要求蓄積応答機能

ユーザの submit 入力により Web サーバのプログラムが起動され応答ページが返るタイプの Web アクセスを完全非同期化するに必要な機能である．Web ブラウザから投入された submit 入力を，インタセプトし，これを一旦蓄積する．そして，ﾀﾞﾐｰの応答としてユーザには，要求が蓄積されたことを示す応答を Web ブラウザ上に返す．その後，ユーザ指示により蓄積された要求は Web 代理アクセスサーバ宛てに非同期通信基盤が提供する非同期通信プロトコルで送信される．蓄積された要求は，ファイルとして記憶するので，要求後に，ユーザは，ブラウザを終了するだけでなくクライアントの電源を切断してもよくなる．

非同期通信プロトコルで送られた要求は Web 代理アクセスサーバで処理され，この処理結果は，クライアント宛てに非同期的に返送される．

(3)応答回収機能

ユーザは，適切なタイミングで応答回収をブラウザを介して指示する。これにより，応答メッセージが受信され，応答に含まれた HTML ページがクライアントの所定の場所に格納され，要求との対応づけがなされる．この要求と応答との対応づけは，HTML のリンクとして記述され，ブラウザから参照できる．このようにして応答として受信したページは，ユーザが適時オフライン

状態においても，要求との対応を取りながら参照する事ができる．

3．2　実現方式

3．2．1 サーバ方式

（1）コンセプト

まず，本実現方式の基本的着想点を述べる．非同期アクセスにおいては，ブラウザからのユーザ入力を一旦蓄積し，これを後からリモートのサーバへ送る機能と，スタティックなページ情報を予めクライアント内に保持しておく機能が必要である．

ページ情報をクライアント内に保持する方法として，クライアント内でWWWサーバを動作させ，その管理下に置くことを考えた．このようにクライアント上で動作するWWWサーバは，各種提供されている．これを活用することで新規開発が少なくできるという開発上のメリットを想定した．

ブラウザから投入されるユーザ入力のインタセプトに関しては，通常ならリモートのWWWサーバ上のプログラムに渡される要求を，渡す先のWWWサーバのアドレスをクライアント内のローカルWWWサーバアドレス変更することによって実現することとした．このアドレス書き換え方式について次に述べる．

また，要求の送信や応答の回収は，メニューページでユーザに提供される．そして，送信と回収のボタンへのユーザ入力をトリガとして動作するCGIプログラムが処理を実行する。

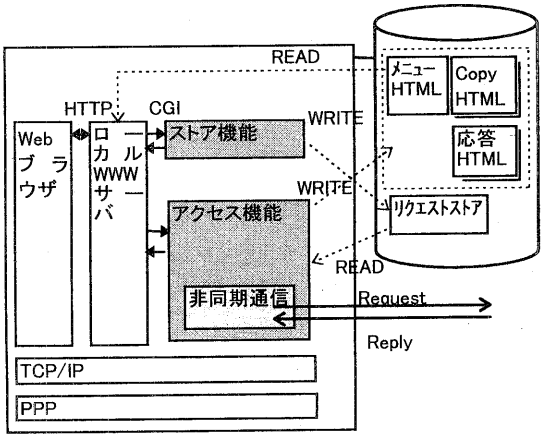

図1．サーバ方式

（2）書き換え方式

(a)要求ページの書き換え

ユーザ入力が渡されるWWWサーバのアドレス情報は，ユーザ入力を行うページのHTMLに記述されている．リモートへの要求をクライアント内のローカルWWWサーバ上で動作するCGIプログラムが受け取るためには，元々のHTMLにおいてWWWサーバアドレスをローカルWWWサーバのアドレスに書き換えればよい．このような，HTMLページのアドレス書き換えをおこなうことにより，ローカルWWWサーバ経由でクライアント内のCGIプログラムにユーザ要求が渡されるようになる．その後，CGIプログラムがリクエストストアと呼ぶディスク上のファイルに要求の書き込みを行う．

書き換え方法は，HTML中のFormタグ中のActionを見つけ，そこに記述されたリモートのCGIプログラムをクライアント内のストア機能を有したCGIプログラムの名称に書き換えておく事により実現する．具体的には，以下に示す処理を行う。

- 書き換え前のHTMLで，Formタグ中のActionステートメントで記述されたCGIプログラムの名称をユーザ入力からスクリプトを作成し，保管するプログラムの名称(上図ではStore.dll)に書き換える．
  これにより，ユーザがsubmitボタンを押したとき実行されるプログラムが，リモートCGIプログラムでなくクライアント内で動作するプログラムになる．

- タイプがhiddenのInputタグとして，リモートCGIプログラムのアドレス情報を追加する．
  これにより，サーバ側のWeb代理アクセスプログラムにアクセスするリモートCGIプログラムのアドレス情報を伝えることができる．なお，タイプがhiddenであるためブラウザには表示されない．

このようにして書き換えたページは，Copy HTMLとしてメニューページからリンクされて保管される．

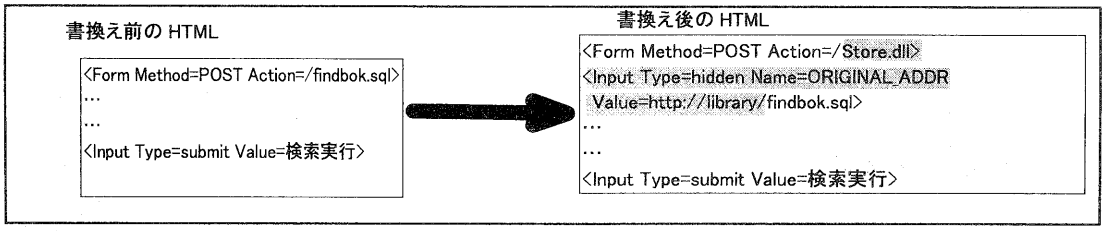

図2. 要求ページの書換え

(b) 応答 HTML のアドレス書き換え

応答の回収は，メニューページにおけるユーザの回収ボタンが押されることで動作し，回収した応答 HTML ページは，クライアントのローカル WWW サーバ配下のリソースとしてメニューにリンクして管理する．このため，ローカル WWW サーバの管理下におかれる応答 HTML ページからのリンクについては，予め，Web 代理アクセスサーバでそのアドレスの書き換えを行う必要がある．

例えば，応答 HTML ページ中に他の Web サーバからイメージリンクが記述されている場合，Web 代理アクセスサーバでは，当該イメージファイルを受信し，応答 HTML ページ中の他 Web サーバからのリンク(絶対パス)をローカル Web サーバ配下のパスに書き換え，当該イメージファイルと共にクライアント側に送ることをしなければならない．

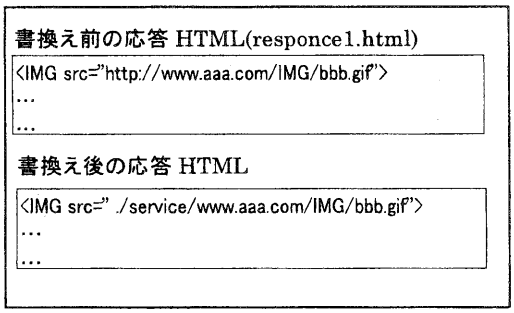

図3. 応答ページの書換え

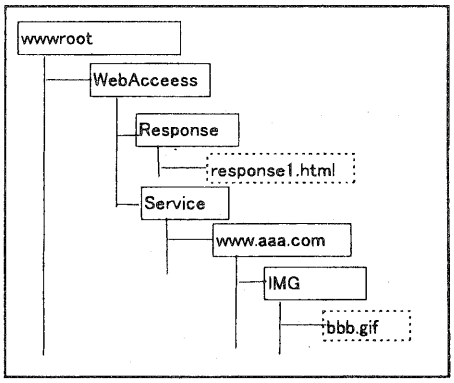

図4. クライアントのディレクトリ構成

上記のように，リモートの Web サーバ上のリソースに対しては，そのリンクのパスをローカル WWW サーバ上にマッピングし，当該リソースを応答 HTML と共に返送することで，クライアントがネットワークに接続していないディスコネクト状態でもブラウザで参照できる．

ここで，Web 代理アクセスサーバ側での書き換えルールについて補足説明しておく．

ルール1：同期的にアクセスする情報は，それが，相対パスでリンクされていたなら，絶対パスに書き換えておく．そして,その情報自体は,応答として返さない．

ルール2：非同期アクセスする情報は，それが絶対パスでリンクされていたなら，相対パスに書き換えておく．そして，その情報を Web サーバから取得し，応答として一緒に送る．また，情報のアドレスは,クライアント内の対応するローカルアドレスへの書き換える．

(3) 考察

要求の書き換えでは，HTML 自身がそのタグとしてブラウザに表示されず，かつ，CGI プログラムに渡す情報が記述可能なhiddenタグをサポートしているため，ブラウザ上の表現を変えることなく実現できた方法であり，HTML の規約に依存した方式である．

応答の書き換えでは，サーバにおける書き換えルールを定義しておくことが必要であり，クライアント内に保管する情報のローカルアドレスの付け方が重要である．

非同期アクセス用のページは，ローカルな URL であるため，ユーザの操作は，単純にメニューページに記載されたリンクをクリックするだけで済むので操作は単純明快である．

3. 2. 2 プロキシベース方式

(1) コンセプト

本方式の基本的着想は，インタセプトの方法として良く知られているプロキシの方法を採用することである．

プロキシで使用されてるキャッシュ技法により，サーバ方式のような面倒な書き換え処理を不要とすることを期待した方式である．しかし，蓄積された要求の送信と受信を行ったり，メニューページをアクセスするため機能をプロキシに付加することが必要となる．

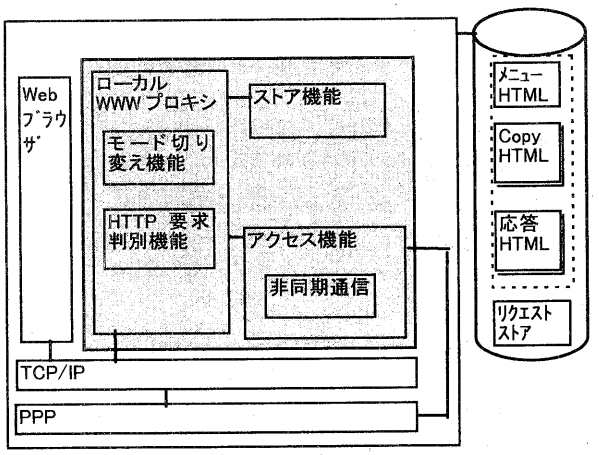

図5 プロキシベース方式

(2) 実現方式

(a) インタセプトの方法

ブラウザのプロキシ設定をローカル WWW プロキシのアドレスに設定させることで，ローカル WWW プロキシがブラウザからのHTTP要求を全てインタセプトできる．

(b) 要求蓄積方法

ローカル WWW プロキシには，ストアとスルーの2モード切り替え機能を設ける．そして，ユーザがストアモードを指定したときのみ，プロキシにおいて以降インタセプトしたHTTP要求をリクエストストアに蓄積する．ただし，メニューページ等のローカルリソースに対するHTTP要求は，ストア対象外とする(HTTP要求判別機能で振り分ける)．

なお，スルーモードが指定されたときは，通常のプロキシとして HTTP 要求の仲介を行う．

(c) 要求送受信方法

サーバ方式と同様にローカルリソースであるメニューのページから送信，受信の指示を行う．そして，リクエストストアに蓄積された HTTP 要求の送信や Web 代理アクセスサーバからの応答 HTML ページの受信を実行する．

また，受信時には，応答 HTML ページを格納し，これを参照するためのリンクをメニューページに作成することで管理する．

(d) 応答参照方法

応答HTMLは，受信処理の一連でメニューページにリンクが張られる．ユーザは，そのメニューページから応答HTMLを呼び出せる．このため，Web代理アクセスサーバへ送った要求 HTTP と受信した応答 HTML との対応を管理しておく．

応答 HTML からリンクされたイメージ情報は，サーバからその URL と共にファイルの実体が返送され，これを格納しておく．ローカル WWW プロキシでは，要求された URL に該当するファイルを先の格納場所から見つけこれをブラウザに返す．

このような管理を行うことで，サーバ方式とは異なり，応答HTMLの書き換えを必要のないものとしている．

(3) 考察

プロキシ部分を完全に作成しなくてはならないため，作成工数がかかる．

プロキシのキャッシュへの付加機能により，ユーザが希望するページの登録(Copy HTML)が簡単に行える．

非同期アクセスの場合は，ユーザがモードを切り替える必要があり，ユーザ操作が煩雑になる．反面，常にモードを意識することで，非同期と同期アクセスを自由に切り替えて要求を行うことができる．

3.3 比較検討

(1)各提案方式の特徴比較

表 1 方式特徴比較表

方式 / 比較項目	サーバ方式	プロキシベース方式
クライアントでの前提プログラム	WWW サーバ	特になし
ブラウザの設定	proxy 設定では，ローカル WWW サーバアドレスを適用外に設定する.	proxy 設定をローカル WWW プロキシアドレス(127.0.0.1)にする.
蓄積タイミング	ローカル web サーバ配下にコピーされた要求ページからCGI要求を出すとき	ストアモード'のときにリモートへの HTTP 要求を蓄積
蓄積対象	CGIプログラム起動要求	全てのHTTP要求
実装方法	CGI プログラム	プロキシの処理プログラム
要求ページの取得	URL を書き換えたものが必要	特別な処理は不要

(2)方式評価

以下の各項目で評価を行う.

(a)適応性

サーバ方式では，CGIでユーザ入力ページを動的に作るもので特に起動するCGIプログラムのアドレスを動的に指定するものには適用できない. これに対して，プロキシベース方式では，こういった制限がないため，適応性が広いといえる.

(b)ユーザ操作性

サーバ方式では，一旦ローカル WWW サーバ配下に置かれたユーザ入力ページからの要求が非同期化されるが，これに対してプロキシベース方式では，ストアとスルーの2つのモードをもち，ユーザがこのモードの切り替えを行うことにより，非同期アクセスか通常のアクセスかを区別して使用する. 従って，モード切り代えという，余分な操作が必要となり，プロキシベース方式では，ユーザ操作がより複雑になっている.

(c)機能拡張性

新たな機能を追加することを考えると，サーバ方式の実装は，CGIプログラムとして実装するため，追加や改変が比較的容易に行える. 一方，プロキシベース方式では，プロキシと一体化した実装であるため，機能追加等が厄介となりやすい. しかし，インタフェースに規定されないため，追加機能の自由度は大きい.

(d)開発容易性

サーバ方式では，WWW サーバを前提とするため，その分プロキシベース方式よりも開発量は少なくなる.

(e)運用・保守性

・システムユーザ

サーバ方式は，エンドユーザに提供するページの書き換えが必要であるので，運用に当たっては補助ツールが必要である. また，エンドユーザに提供したページが，更新されたかどうかの管理も行う必要があり，負担が多い. しかし，プロキシベース方式では，このような負担はない.

・エンドユーザ

サーバ方式では，書き換えを伴うため運用に当たってはシステムユーザのサポートが必要. プロキシベース方式では，システムユーザのサポートは，必ずしも必要ないが，運用に当たっては，モードの切り替えに習熟する必要がある. また，ページの更新管理はユーザが行うことになる。

(f) 用途に関する検討

・ユーザのタイプ

上述によりサーバ方式では，エンドユーザは，非同期アクセスのページを選択すれば，よいので操

作が単純であり，初心者や特定業務を専用にこなすユーザ向けである．プロキシベース方式では，ユーザのモード切り替えが必要であり，ストアモードを設定した以降のブラウザアクセスが非同期化するのでユーザが意識して使用することが必要となる．また，非同期アクセス用のページを事前にブラウザでアクセスし，ファイルとして保存しておくことで自由に非同期アクセス用ページをユーザ毎にカスタム化して保存することが容易である．したがって，非定型的なサービスを望むユーザやパワーユーザ向けであるといえる．

・サービスタイプ

サービスを考えると，インタネットで提供されているページをユーザの必要に応じて非同期化してアクセスする場合は，プロキシ方式が向いている．

逆に，特定企業内のイントラネット内で提供される定型業務用のサービスは，システムユーザがメンテナンスを行うサーバ方式が適しているといえる．

表2 方式評価結果

項目＼方式		サーバ方式	プロキシベース方式
一般的評価	適応性	△	○
	ユーザ操作性	◎	○
	拡張性	○	◎
	開発工数	○	△
	保守性	○	○
用途	ユーザタイプ	定型業務ユーザ	パワーユーザ，非定型業務ユーザ
	サービスタイプ	イントラネットサービス用	インターネットサービス用

4. あとがき

既存ブラウザを使用して Web アクセスを完全非同期アクセス化する実現方式に関して，クライアント内の WWW サーバを利用するサーバ方式とクライアント内に WWW プロキシを動作させるプロキシベース方式との2つの方式を提案し，比較を行った．

作成の容易性やユーザ操作性等の一般的事項を総合すると一長一短であり，用途により向き不向きがあることが分かった．検討の結果，サーバ方式は，イントラネット内の特定のサービスをシステム管理者が手を加えてモバイル用に提供する場合に適しており，特に定型的な処理を実行するユーザに向いている．また，プロキシベース方式は，インターネットで提供しているサービスをユーザ自身が選択して非同期化する場合に適しており，特に，非定型的な処理を実行するパワーユーザ向きの方式であるとの結論を得た．

参考文献

1)J.Mogul, H.Frystic, T.Berners-Lee, "Hypertext Transfer Protocol-HTTP/1.1", RFC2068,January 1997

2)Henry Chang, CarlTait, NormanCohen, Moshe Shapiro, Steve Mastrianni, Rick Floyd, Barron Housel, David Lindquist, "Web Browsing in a Wireless Environment: Disconnected and Asynchonous Operation in ARTour Web Express" Proc. MobiCom '97,ACM, Sept. 1997