# WebTSによるレガシーアプリケーションのWebアプリケーション化検証

## Verification of Transition from Series 2200 Legacy Application to e-Business Application

末永敏昭

**要　約**　Web技術を中心としたEビジネスシステムが増えてきている．従来から使用されているシステムで持っている大量のソフトウェアおよびデータを有効利用し，Eビジネス対応をおこなう必要がある．

本稿では，従来のシステムをEビジネス対応するための課題を整理し，検証モデルにより実証評価した結果をもとに，Eビジネスへの適用について論じる．

**Abstract**　The Web technology-based e-business applications have been rapidly increased. It is important to migrate the legacy applications to the e-business ones bringing numerous software and data assets held on legacy applications into productive use.

This paper examines problems inherent in migration from an operational legacy application to an advanced e-business one using WebTS (Web Transaction Server), and discusses the applicability of WebTS to e-business applications, depending on results from verification and evaluation performed by the proposed verification model.

## 1. はじめに

シリーズ2200/HMP IXシリーズ/CS 7802（以下シリーズ2200と略す）上に存在する大量のソフトウェアやデータを有効利用し，Eビジネス対応を行う一つの方法として，当社が提供するソフトウェアWebTS（Web Transaction Server）を利用する方法がある．

従来のビジネスとEビジネスは表1のように比較されている[1]．

**表1　従来のビジネスとEビジネスとの比較**

| | 従来のビジネス | Eビジネス |
|---|---|---|
| ビジネスモデル | 従来のビジネスモデルをシステム化 | 従来とは異なるビジネスモデルが必要 |
| ビジネスモデルの変化 | 静的で使い続けるモデル | 動的で変化するモデル |
| 顧客特性 | 固定的，継続的顧客との安定取り引きが主体 | 全世界的，新規顧客の獲得に重点 |
| 業務特性 | 「基幹業務」とそれ以外の業務 | 商品企画，設計，購買，製造，配送，マーケティング，販売，顧客サポートのすべての業務が「基幹業務」になる可能性 |
| サービス時間 | 営業時間内 | 24時間365日 |

Eビジネス対応をおこなうとは，従来のビジネス環境で使用されていた大量のソフ

トウェアやデータを E ビジネス環境でも有効に活用できるようにすることである．現行アプリケーションを WebTS を利用する Web アプリケーションに改修することが，その一つの方法である．

本稿では，WebTS を利用した Web アプリケーションを E ビジネス・アプリケーションに対応させるための問題点の洗い出し，その対応策および結果の検証について記述する．

## 2. WebTS の概要

図 1 は，WebTS を取り巻くソフトウェア構成を表わしたものである．WebTS はシリーズ 2200 上で稼働する Web サーバであり，Web ブラウザと HTTP プロトコルによりデータ送受信を行う機能と，ユーザ・プログラムとの間でデータを送受信する機能を持ったソフトウェアである．ユーザ・プログラムは，WebTS が提供する API を使用することにより，容易に Web アプリケーションを構築できる．

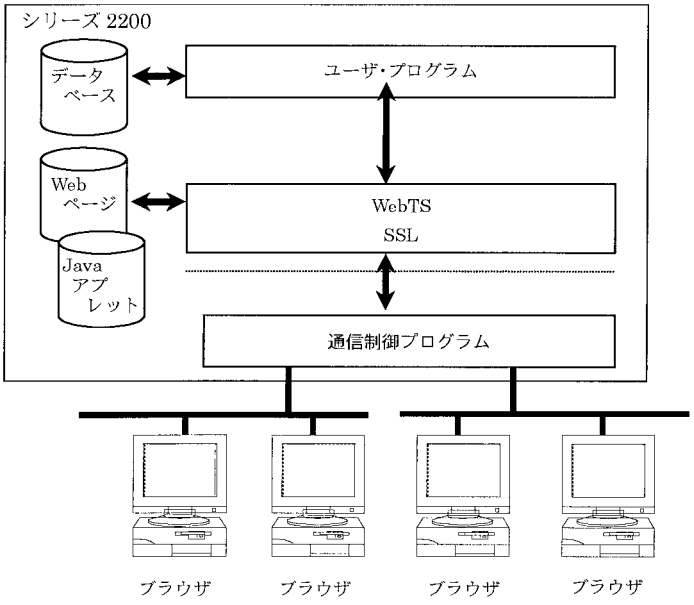



**図 1　ソフトウェア構成図**

WebTS の利用形態には，Web ブラウザに対して HTML 文書を直接入出力する方法と，入出力に必要なデータのみを WebTS との間でやり取りし，WebTS が提供する Java アプレットがユーザ・プログラムから渡されたデータを HTML 文書に埋め込んで入出力する方法の二種類がある．後者の場合，ユーザ・プログラムは HTML 文書を直接操作しないため，既存ユーザ・プログラムの入出力部分を WebTS の API を呼ぶように改修することで Web 対応を図ることができる．HTML 文書とユーザ・プログラムを分離することができるため，ユーザ・プログラムは HTML 文書を意識しないプログラミングができ，また HTML 文書のレイアウト変更はプログラムを修正することなくおこなえる．

WebTS は，ブラウザへ HTML 文書を渡すための専用 Java アプレットを提供している．この Java アプレットがシリーズ 2200 上からブラウザ上にダウンロードされ，実際の出力処理をおこなう．

## 3. E ビジネス対応への課題と対策

WebTS を利用した Web アプリケーションを，E ビジネス環境に照らし合わせた場合，次の五つの課題があり，それぞれの対策を以下に示す．

### 3.1 Java アプレットを実行できないブラウザへの出力

WebTS を利用し，ユーザ・データのみを WebTS との間でやり取りする形態を使用するプログラムの場合，出力用の雛形 HTML 文書にユーザ・データを埋め込みブラウザ上に表示するために，Java アプレットが使用される．Java アプレットは 6 種類あり，雛形 HTML 文書の記述方法に対応した Java アプレットがブラウザにダウンロードされ実行される．

図 2 は，WebTS とブラウザとの間のやり取りを簡単に表わしたものである．

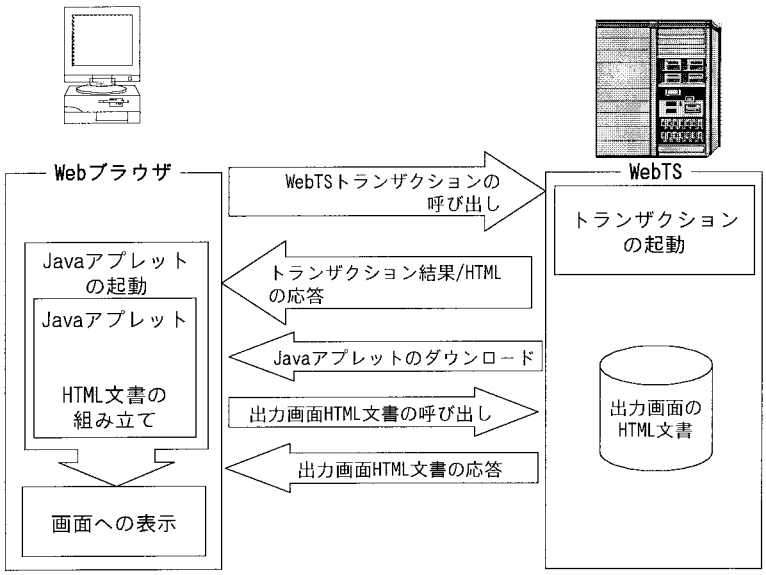



**図 2　WebTS とブラウザとのやり取り**

しかし Java アプレットを実行できないブラウザが存在し，WebTS からの HTML 出力を直接処理することができないものもある．

最近では i モード端末でも KVM（K virtual machine）が実装され，Java アプレットが稼働できる環境が整いつつあるが，その大きさは 80 KB 以下と限られており現時点では実行させるにはまだ充分とはいえない．

このようなブラウザに対しては，Java アプレットを使用せず最終的な形の HTML 文書を作り出し，ブラウザに送る処理が必要である．

対策として 3 階層構成を構築し，WebTS からの出力を一旦中間層で受信し，HTML 文書にユーザ・データを埋め込む処理を中間層でおこない，最終的な HTML 文書を

ブラウザ側に送信する方法が考えられる．この方法を採ることにより，付帯効果として 6 種類の Java アプレットのダウンロードが必要なくなる．

WebTS からの出力において，ブラウザの種類を判別し，種類に対応した HTML 文書を作り出す方法も考えられるが, 現在 WebTS が提供している機能では, ユーザ・プログラムが直接 HTML 文書を出力する方法しかない．現行のアプリケーションに改造を加えるとすると，入出力部分の改修以外に HTML 文章を作り出す処理を追加する必要があり，HTML 文書の作り出し部分が増加となる．直接 HTML 文書を入出力する方法は，新規アプリケーションを開発する場合に選択することを薦める．

このように，現行アプリケーションを使用し，Java アプレットを使用できないブラウザに対して出力をおこなう場合は，3 階層構成にすることが有効な手段である．

## 3.2 セキュリティ

全世界が顧客としての対象となる E ビジネスでは，インターネットへの接続が必須となる．

WebTS を利用したアプリケーションは，シリーズ 2200 上に構築し，シリーズ 2200 が Web サーバとしての役目を担う．この形態を直接インターネットに接続した場合，ハッキング行為などによりシリーズ 2200 の資源が破壊される可能性がある．FireWall などをシリーズ 2200 フロントに置いたり SSL を使用することによりある程度防ぐことはできるが，完全に阻止することはできない．

ハッキング行為から重要なサーバを守るためのセキュリティ対策として，インターネットとイントラネットとの間に二台の FireWall を設置し，緩衝地帯となる FireWall 間の DMZ（De−Militarized Zone）上に Web サーバを置く 3 階層構成が一般に提唱されている[6]．

このようなネットワーク形態では，インターネットからアクセスできるのは DMZ 上の Web サーバだけで，イントラネットへのアクセスはその Web サーバがゲートウェイとなり中継しておこなわれる．これによりインターネットから直接イントラネットへアクセスすることは難しくなり，ハッキング行為といった脅威は低減されて安全性の向上が図れる．

DMZ 上の Web サーバとして，シリーズ 2200 を置く方法と WindowsNT などを搭載したサーバ機を置く方法が考えられる．WebTS を使用したアプリケーションが存在するシリーズ 2200 を DMZ 上に置き，そのアプリケーションとイントラネット内のシリーズ 2200 データ・ベース・サーバ間を個別のプロトコルで接続する方法では．Web アプリケーション・サーバとなる DMZ 上のシリーズ 2200 には，WebTS を使用したアプリケーションしか乗せることができないなど，シリーズ 2200 が持っている汎用機としての能力を有効に利用することができない．図 3 のように WebTS を使用したアプリケーションが存在するシリーズ 2200 をイントラネット内に置き，DMZ 上には WindowsNT などを搭載したサーバを置く方法では，DMZ 上のサーバには Web アプリケーション・サーバとしての機能と，WebTS を利用するアプリケーションとの間でデータをやり取りするための機能が要求されるが，WebLogic Server など Windows 上で動かすことができる汎用アプリケーションを有効に利用することができるので，この方法を推奨する．

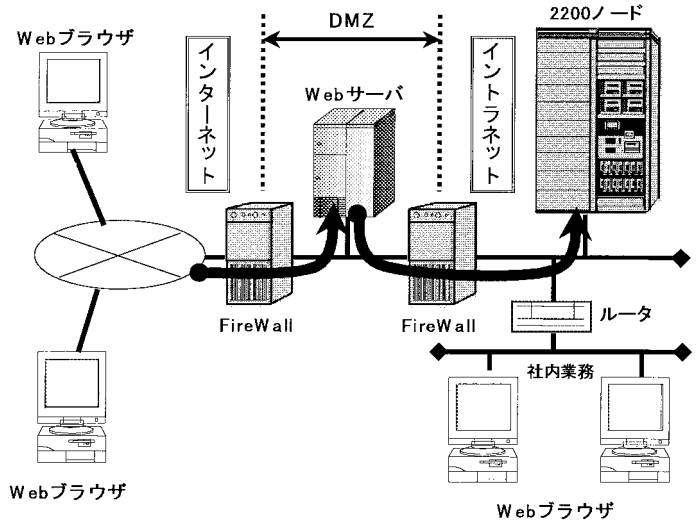



**図 3　緩衝地帯となる DMZ 上に Web サーバを置くモデル**

### 3.3　HTML のすべてのシンタックスをサポートしていないブラウザとのやり取り

すべての業務が基幹業務となり得る E ビジネスでは，従来イントラネット内だけに公開していた情報をインターネットに提供するなど，イントラネットとインターネットでの情報共有や，端末の種類を問わずにデータを送信できることが求められる．

最近では携帯電話端末などのように HTML の一部分のみをサポートし，処理部分を簡略化しているブラウザが出現してきた．このような簡略化したブラウザでは，出力できるデータ量の制限も存在する．このようなブラウザに，現行の HTML 文書をそのままの形で送信することはできない．表 2 は各社携帯電話端末の仕様を表わしている．

**表 2　各社携帯電話端末の仕様**

| 種　類<br>[会社名] | i モード端末[2]<br>[NTT ドコモ] | EZ サービス端末[3]<br>[KDDI] | J-スカイ端末[4]<br>[J-フォン·グループ] |
|---|---|---|---|
| ページ記述言語 | i モード用 HTML | HDML | MML |
| ページ許容量 | 5KB まで | 1.2KB まで | 6KB まで |
| 画像形式 | GIF | BMP | PNG |
| 表示色 | カラー256 色 | 白黒 4 階調 | カラー256 色 |

現行の WebTS は HTML を変換する機能を備えていない．そのため対象となるブラウザは HTML をフル・サポートしているもので，かつデータ量に制限がないものに限られる．

携帯電話端末のようにページを記述する言語が各社で異なるような場合，各社の言語に対応した入出力ページを用意し，WebTS 側のアプリケーションでブラウザ毎に振り分ける方法が考えられる．しかしこの方法では，ページの更新はそれぞれのページごとにおこなわなければならないなど，保守性に劣る．また WebTS 側のアプリケ

ーションでHTMLをブラウザに対応して動的に変換する方法が考えられるが，WebTS側の個々のアプリケーションでブラウザを判断しそれぞれに対応した出力を作成することは，シリーズ2200側の処理が複雑になるため避けるべきである．従って，対策としては，3階層構成にし，WebTSからのHTML出力を一旦中間層で受信し，各ブラウザに対応したページに変換する方法が有効な手段である．

### 3.4 アクセス数の増大

Webページで要求されるレスポンス・タイムには，よく言われる「8秒ルール」がある．これは米国Zona Research社が1999年6月末に公表した報告書[5]の結論で述べられているもので，Webページの表示に8秒以上かかるサイトでは，3分の1以上がアクセスを止めてしまうため顧客を逃がす公算は大きいというものである．

Eビジネスへの対応によりイントラネット，インターネットを問わずWebブラウザからの処理要求がシリーズ2200に直接入力されることになるため，シリーズ2200の負荷増大が予想される．そのため，上記の顧客離れや業務遅延などを生じさせる可能性がある．

インターネットへの接続により接続相手が不特定多数となるため，アクセス数を予測することは極めて難しくなる．その予想できないアクセス数の負荷に耐えるシステムを構築するには，負荷分散装置の導入と3階層構成への移行が効果的である[7]．

3階層構成とすることで，ビジネス・ロジックは中間層で処理し，シリーズ2200はデータ・ベース・サーバとしての処理に徹することができる．このように，アクセス数の増大に対する対策としては，負荷分散装置の導入と3階層構成にすることである．しかしながら，ビジネス・ロジックの作り方によっては，アプリケーションの処理に時間がかかることが懸念される．ビジネス・ロジックの作成にあたっては，この8秒以内のレスポンスを保障することが必要である．

### 3.5 24時間365日のサポート

Eビジネスでは1日24時間，365日のサービスが必要である．シリーズ2200上にWebサーバを構築し，直接Webブラウザからアクセスする場合，シリーズ2200を24時間365日稼働させる必要がある．シリーズ2200が本来受け持つ定型業務の影響を受けずに，24時間に近いサービスを提供するためには，WebTSの多重サーバ機能を使用してシリーズ2200上に複数のWebサーバを稼働させることで回避できる．しかしメンテナンスやブート時などのサービス停止への対応には，シリーズ2200を複数台設置しXTPA構成などにより二重化の措置が必要である．また，今まで述べてきたような3階層構成でも同様に二重化の措置が必要である．このように，24時間365日のサポートに対する対策としては，機器の二重化が必要となる．

## 4. 3階層モデルでの検証

3章の結果から，WebTSを利用するWebアプリケーションをEビジネス環境に当てはめた場合，3階層構成が有効な手段であると判断できる．そのためシリーズ2200のWebアプリケーション・サーバとブラウザとの間にWindowsNTを置き，そのWindowsNT上で稼働するプロトタイプを作成し結果の検証をおこなった．

ブラウザからアクセスするのは常に中間層として，中間層を経由してシリーズ2200

の Web アプリケーションを呼び，その結果を中間層で受信してブラウザに返す構成とした．

### 4.1 中間層の要件

中間層の要件には，次の 5 項目を設定した．

① ブラウザに HTML のみを返すこと．
② HTML 変換が可能であること．
③ シリーズ 2200 の負荷が増大しないこと．
④ ビジネス・ロジックが容易に組み込めること．
⑤ パフォーマンスに問題がないこと．

### 4.2 中間層の構成

中間層を DMZ に置くことを考慮すると，ブラウザと中間層との間および中間層からシリーズ 2200 との間はそれぞれ FireWall を通過することになる．そのためセキュリティの観点から，むやみに FireWall の通過帯を設定することは避け，できる限り通過できるポートの数は押さえる必要がある．ブラウザと中間層との間は要件①にもあるように，「ブラウザに HTML のみを返す」ことから単にブラウザと Web サーバの関係となるため，通過帯はポート番号 80 の HTTP(Hyper-Text Transfer Protocol) のみとすることができる．一方で中間層からシリーズ 2200 の間については，WebTS を利用するアプリケーションとのやり取りとなるため選択肢は多数考えられた．しかし E ビジネスにおける拡張性や要件④の 「ビジネス・ロジックが容易に組み込める」から，中間層での中継が容易でかつ他社製品とも簡単に組み合わせて使用できる HTTP がここでは有利と判断し，この部分もポート番号 80 で接続することにした．

検証は中間層の処理に図 4 のような ASP（Active Server Pages）を使用した場合と，図 5 のような WebLogic Server を使用した場合の二種類について実施した．

#### 4.2.1 中間層アプリケーションの概要

中間層で動作するアプリケーションは，汎用性がある Java 言語を使用して新たに作成した．このアプリケーションの概要は次のとおりである．

WebTSConnecter と名付けたこの Java アプリケーションは，HTTP を使用してシ

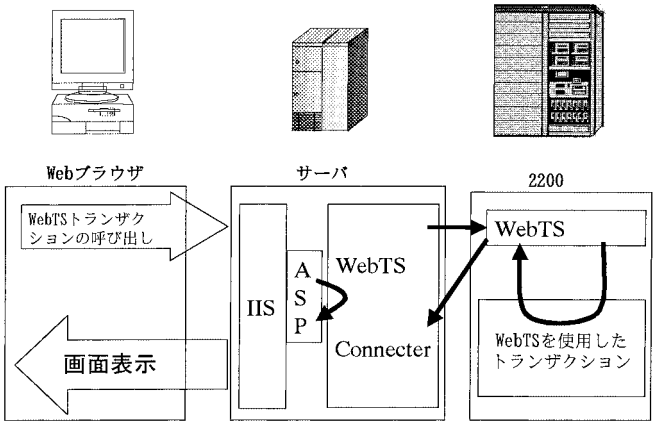



**図 4　ASP を使用した WebTSConnecter 構成図**

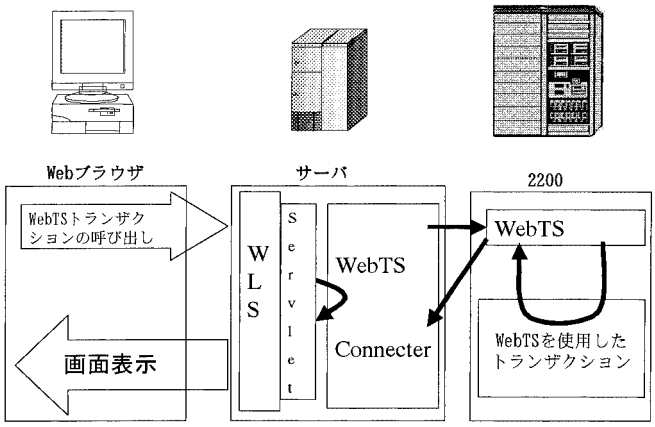



**図 5　WebLogic Server を使用した WebTSConnecter 構成図**

リーズ 2200 上のアプリケーションとやり取りをおこない，ブラウザに HTML を応答する機能を持っている．加えて WebTSConnecter は，ブラウザへの応答時にオプションで次の処理をおこなう．

1）3 階層構成にすることにより，シリーズ 2200 上に存在する Web ページや画像といった静的コンテンツをブラウザにダウンロードすると，中間層を経由してダウンロードされるため無駄な転送がおこなわれる．静的コンテンツは頻繁に変更されるものではないので，WebTSConnecter が存在する中間層に置き，中間層から Web ブラウザに直接応答するようにする．これにより，シリーズ 2200 において静的コンテンツのダウンロードのための負荷を軽減できる．

WebTSConnecter では静的コンテンツを中間層から直接ブラウザにダウンロードさせるため，HTML 文書に記述されているシリーズ 2200 上の Web ページや画像などへのリンクを，次のように中間層へのリンクに変換する．変更前の例で，172.19.105.51 および 83 は，それぞれ WebTS が稼働しているシリーズ 2200 の IP アドレスとポート番号である．その後ろの SAMPLE は，HTML 文書が存在するファイル名である．このように http により，シリーズ 2200 上に存在する HTML 文書を参照しようとしている．変更後の例では，直接，HTML 文書が記述されており，WebTSConnecter が存在するサーバ上の HTML 文書が参照される．

【変換前】

```
<a href="http://172.19.105.51:83/SAMPLE/input.html">入力頁</a>
```

【変換後】

```
<a href="input.html">入力頁</a>
```

この機能を使用すると，静的なコンテンツは常に中間層に置かれたものが参照されることになる． そのため HTML 文書から参照される Web ページや画像は，

あらかじめ WebTSConnecter が存在する中間層のサーバにコピーしておく必要がある．

2) WebTSConnecter を使用した場合，HTML 文書に記述されているシリーズ 2200 上のトランザクションへのアクセスは，中間層を経由しておこなわれる．そのため HTML 文書に記述されているリンク部分は，シリーズ 2200 上のトランザクションを起動する形式ではなく，WebTSConnecter が置かれている中間層のサーバへアクセスする形に書き替える必要がある．HTML 文書内にある以下のようなシリーズ 2200 上のトランザクションへのリンクを（<a　href＝”http : //172.19.105.51 : 83/TXN/TRAN 1. . .），次のように WebTSConnecter が存在する中間層へのリンク（<a href＝”ASPname. . .）に変換する．

【変換前】

```
<a href="http://172.19.105.51:83/TXN/TRAN1?IN-1=NEXT_PAGE">次頁</a>
```

【変換後】

```
<a href="ASPname?TXN=TRAN1&IN-1=NEXT_PAGE">次頁</a>
```

上記の例は中間層の処理に ASP を使用した場合のもので，変換後のリンク部分は中間層にある ASP を起動し，ASP にシリーズ 2200 上のトランザクション名（TXN＝TRAN 1）およびデータ（IN-1＝NEXT_PAGE）を渡すようになる．その後，ASP 経由で起動された WebTSConnecter の内部から実際のシリーズ 2200 上のトランザクションを呼び出す形になる．

3) HTML 文書の JavaScript 内に記述されているリンク部分を，WebTSConnecter が存在する中間層へのリンクに変換する．

HTML 文書中にはタグのほかに，JavaScript などのスクリプトが記述されている場合もある．そのため WebTSConnecter では，JavaScript の document.write（）メソッドなどで指定されている文字列中にあるシリーズ 2200 上へのリンクも同様に，WebTSConncter が存在する中間層へのリンクに変換する機能を設けている．これにより JavaScript で HTML を動的に生成している場合でも，その中のリンクは中間層のものを参照するように変更される．

上記 1）から 3）に共通して，WebTSConnecter により変換されるリンクは表 3 に示すタグのパラメタに指定されたものである．

**表 3　URL 変換の対象となるタグとパラメタ**

| タグ | パラメタ |
|---|---|
| BODY | BACKGROUND |
| FRAME | SRC |
| BASE | HREF |
| LINK | HREF |
| IMG | SRC |
| A | HREF |
| BGSOUND | SRC |

| タグ | パラメタ |
|---|---|
| FIG | SRC |
| AREA | HREF |
| ISINDEX | ACTION |
| FORM | ACTION |
| INPUT | SRC |
| LAYER | SRC |

4） ブラウザおよびシリーズ 2200 から送信される Cookie 情報を中継する．

WebTS の PID（Position IDentifier）保持機能を使用すると，トランザクションをスケジュールする際に割り当てた PID 番号などが収められた Cookie 情報が，WebTS から送信される．Cookie 情報を受信したブラウザからは，それ以降 WebTS への送信データにその Cookie 情報が付加される．WebTS が次にトランザクションを起動する際には，この Cookie 情報に収められた PID 番号が使用されてスケジュールがおこなわれるため，これによりセッションの管理と一連の連携したトランザクションの実行が可能となる．

WebTSConnecter は，ブラウザおよびシリーズ 2200 から送信される Cookie 情報を中継する機能を持たせた．これには WebTS が送信する Cookie 情報の他に，ユーザが設定した Cookie 情報も含まれる．この機能を使用することで，中間層を意識すること無く WebTS における PID 保持機能を使用したり，Cookie のやり取りをおこなうことができる．

5） シリーズ 2200 とのやり取りでエラーを検出した場合に，次のようなエラー・メッセージの HTML を応答する．

```
<html>
<head>
<title>Error</title>
<body>
<p>WebTS 連携でエラーが発生しました．
<p>ステータス：エラーコード
<p>エラーメッセージ
</body>
<html>
```

この機能を設けることで，シリーズ 2200 における障害や，通信上のエラーなどが発生した場合に対応が可能となる．

WebTSConnecter では上記のメッセージの他に，ユーザ独自のエラー・メッセージを設定することも可能にした．

6） トランザクションの出力結果を HTML 文書にマージする

WebTS を利用した場合，トランザクションの出力は次のような過程を経て Web ブラウザに表示される．

① トランザクションの出力結果が付加された Java アプレットを起動するための HTML 文書が，Web ブラウザに読み込まれる．

② 上記の HTML 文書が読み込まれると，6 種類の Java アプレットが Web ブラウザ上にダウンロードされる．

③ Web ブラウザで Java アプレットが動作し，Java アプレットはシリーズ 2200 上にある出力画面の HTML 文書をプログラム内に読み込む．

④ Java アプレットは読み込んだ出力画面の HTML 文書に，トランザクションの出力結果を埋め込んで Web ブラウザの画面上に表示する．

これにより，出力結果が Web ブラウザに表示されるまでには，シリーズ 2200 上の WebTS に対して合計で 8 回のアクセスがおこなわれている．この場合も中

間層に位置する WebTSConnnecter でマージ作業をおこなうことで，6 種類の Java アプレットを Web ブラウザへダウンロードする部分のアクセスを無くすことができる．

そのため WebTSConnecter では，WebTS の Java アプレットでおこなっているユーザデータの HTML 文書への埋め込み作業と同様の機能を装備した．

### 4.2.2 中間層の処理に ASP を使用した場合の検証

ここでは，次のような WebTSConnecter を呼び出す ASP を用意して検証をおこなった．

```
<%
'       トランザクションの URL 設定
    url = http://172.19.105.51:83/TXN/CODE
'
'       メソッドの取得
    method = Request.ServerVariables("REQUEST_METHOD")
'
'       ASP 名の取得
    aspName = Request.ServerVariables("SCRIPT_NAME")
'
'       Cookie 情報の取得
    cookie = Request.ServerVariables("HTTP_Cookie")
'
'       クエリー情報の取得
    questr = Request.ServerVariables("QUERY_STRING")
'
'       パラメタの組み立て
    For Each x In Request.Form
```

```
        Temp = x & "=" & Request.Form(x) & "&"
        Parm = parm & temp
    Next
    Parm  = parm & questr
'
'       WebTSConnecter オブジェクトの生成
    Set obj = Server.CreateObject("WebTSConnecter.TSC")
'
'       WebTS トランザクションの呼び出し
    status  = obj.CallTS(url, parm, method,cookie)
'
    If status ="ok" OR status > "700" Then
'
'       出力画面の HTML 文書取得
        outhtml = obj.GetDoc(aspName)
    else
'
'       エラー用 HTML 文書の作成
        outhtml = obj.SetErrorHTML(Error.html)
    End If
'
'       Cookie 情報の取得
    get_cookie = obj.GetCookie()
'
'       WebTSConnecter オブジェクトの消去
    Set obj = Nothing
'
'       Cookie 情報の設定
    IF get_cookie <> vbNullstring Then
         Response.AddHeader "Set-Cookie", get_cookie
    End If
'
'       HTML 文書の表示
    Response.Write outhtml
%>
```

GET または POST メソッドによる Web ブラウザからの入力はこの ASP で受信され，Java アプリケーションである WebTSConnecter が起動される．WebTSConnecter はシリーズ 2200 上のアプリケーションとやり取りをおこない Web ブラウザに HTML を応答する．

通常の WebTS とブラウザとの間のやり取りは，図 2 で示したようになっているが，ASP を使用した場合は図 6 のようになる．ここでは出力画面の HTML 文書を中間層に置き，トランザクションの出力と HTML 文書とのマージ作業を中間層にある WebTSConnecter でおこなうことで，6 種類の Java アプレットのダウンロードと，出力画面の HTML 文書の読み込みが無くなった．また，ASP の作りにより，中間層のデータベースを参照する方法や，別サーバへの参照へ切り替えるなどの処理を追加することも可能である．Web ブラウザからの参照は常に中間層へのアクセスであるため，Web ブラウザでの考慮は必要ない．加えて今回は検証をおこなっていないが，要件②の「HTML 変換」については WebTSConnecter に変換テーブルを持たせることにより可能となる．

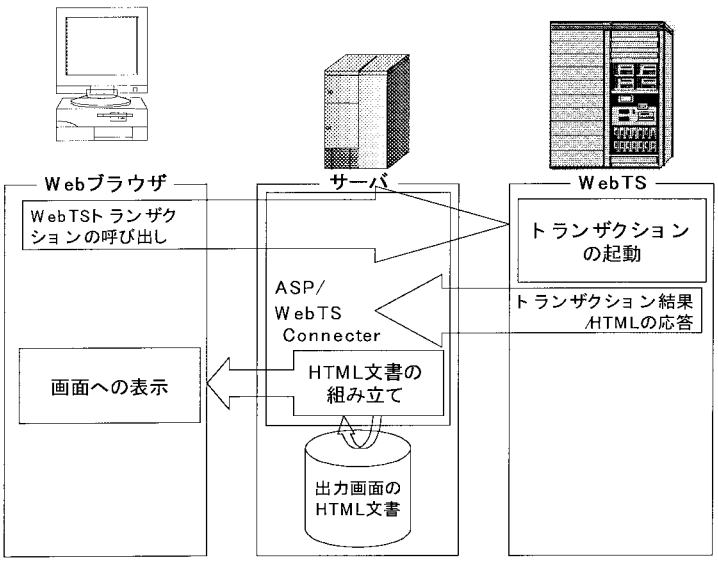



**図 6　中間層の処理に ASP を使用した場合の WebTS とブラウザとの間のやり取り**

### 4.2.3　中間層の処理に WebLogic Server を使用した場合の検証

WebLogic Server は分散 Java アプリケーションの開発，運用，管理を目的として開発された高い拡張性を持つアプリケーション・サーバ・ソフトウェアである．

要件③の「シリーズ 2200 の負荷が増大しないこと」という点では ASP を使用した場合と同様に，シリーズ 2200 上へのアクセスはトランザクションとの入出力部分の 1 回である．また，ブラウザから要求がある毎にプロセスが実行される ASP の場合とは異なり，WebLogic Server では WebTSConnecter がプロセス中のスレッドとして動作する Servlet 形式であるため，アクセス数が増加してもプロセスの数は変わらない．そのため，サーバにおける CPU 使用率を比較した場合には，「パフォーマン

スに問題がない」という⑤の要件でも WebLogic　Server の方が優位であることが確認された.

また WebLogic　Server を使用すると，多階層モデルが容易に構成できる．今回の検証では中間層のサーバに WebTSConnecter や静的なコンテンツを置いているが，この中間層のサーバが DMZ 上にあると，万が一このサーバに対してハッキング行為がおこなわれた場合にはそれらの資産が破壊される可能性がある.

図 7 は WebLogic　Server を使用して多階層モデルとした場合の例である．中間層は HMP　IX の NT ノードとして，ここに WebTSConnecter を置いている．この場合 DMZ 上のサーバは，イントラネット内の NT ノードに置かれた中間層との中継のみをおこなう形となるため，たとえハッキング行為がおこなわれても資産は守られ，セキュリティは更に向上する.

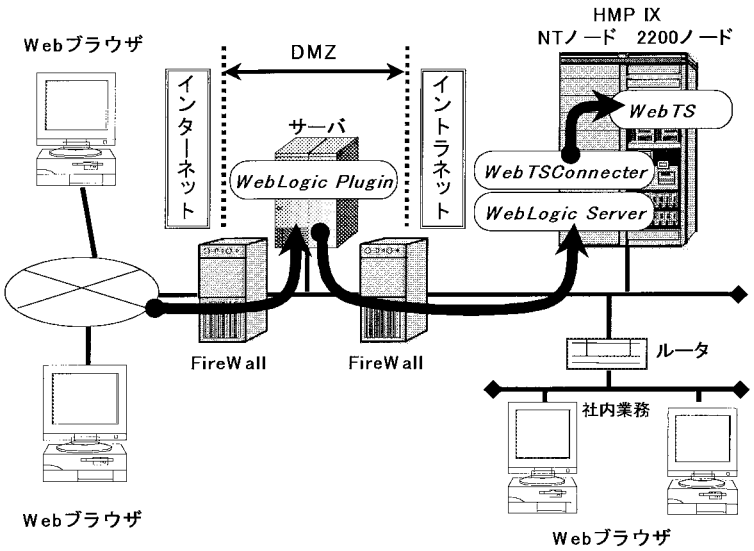



**図 7　WebLogic Server を使用した多階層モデル**

## 5.　検証モデルの評価

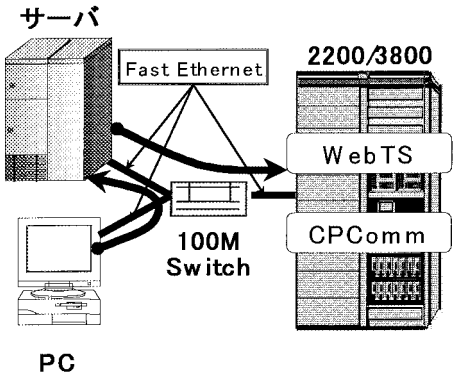
検証モデルの評価は，機能面，効率面について実施した．機能面では，3 階層モデルにおいて 3 章の課題が解消されていることを確認することである.

まず，「Java アプレットを実行できないブラウザ対応」では，中間層の WebTSConnector で，ユーザデータを HTML 文書へ埋め込む処理で対応できた.

「セキュリティ対策」では，3 階層構成にし，中間層の WebTSConnector において，ブラウザとのやり取りとシリーズ 2200 とのやり取りをおこなう処理で対応できた.

「HTML のすべてをサポートしていないブラウザ対応」に関しては，今回のプロトタイプには含めなかったため，検証はできていない.「アクセス数増大に対する対策」および 24 時間 365 日」対応については，ハードウェア構成に関わることなので十分な検証はできなかった．また 3 階層モデルにおけるパフォーマンス評価は，図 8 のような構成で，中間層（サーバ）の処理に ASP を使用した場合と WebLogic Server を

使用した場合の両方について実施した．起動は WebStone[8] を使用して 2,500 バイトの HTML 文書を出力するプロセスを複数のクライアントから連続的にアクセスする方法でおこなった．




| PC | |
|---|---|
| CPU: PentiumIII 500MHz<br>Mem:128M | |
| WindowsNT | 4.0 |
| WebStone | 2.5 |

| サーバ | |
|---|---|
| CPU: PentiumIII 500MHz<br>Mem: 576M | |
| WindowsNT | 4.0 |
| IIS | 4.0 |
| WebLogic | 4.52 |

| 2200/3800 | |
|---|---|
| Model:3800-42<br>IP:IP900×1<br>MSU:268,438K | |
| EXEC | 46R2 |
| WebTS | 3R2A |
| CPComm | 2R1 |
| MCB1100 | 7R3 |

**図 8　パフォーマンス測定における構成と諸元**

クライアント数を変化させた時の処理件数とサーバにおける CPU 使用率は表 4 に示すとおりで，いずれの場合も，クライアント数の増加に応じて平均処理件数および CPU 使用率は滑らかに推移しており，WebTSConnecter を使用した 3 階層モデルの構造には問題がないと言える．

## 6. XML への対応

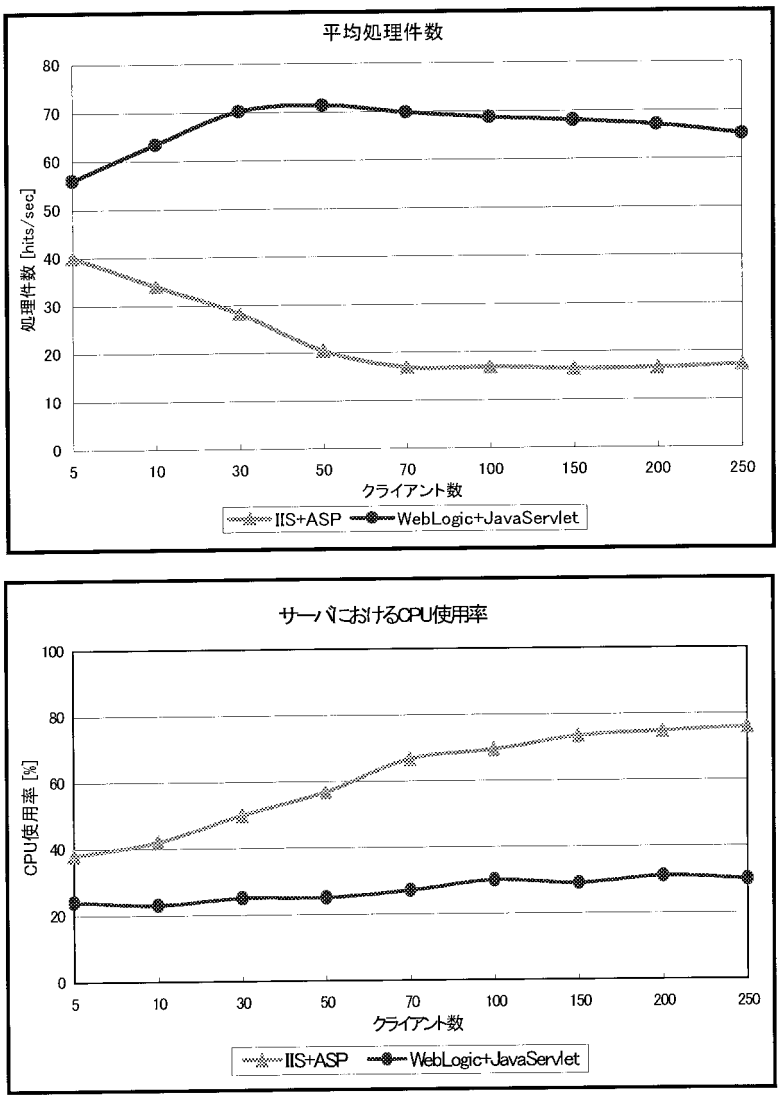
現在，XML を企業間システムのデータ連携に利用することで，汎用的なシステム統合の手段を提供する動きが業界団体などで推進されている．

XML をブラウザに表示するためには，ブラウザが XML 対応していなければならない．Internet　Explorer は 4.0 から XML 対応しているが，Netscape は最新版 6 で初めて XML 対応した．しかし，Netscape 6 の XML 対応は完全なものではなく，XML パーサによる解析やテキスト・データの表示は可能であるが，XSLT プロセッサ機能がないため，XSLT スタイル・シートによる HTML への変換などができない．

シリーズ 2200 を Web サーバとした場合，XML を利用した通信が現状どの程度可能かを若干記述する．シリーズ 2200 上の Web サーバ・ソフトウェア WebTS を使用した場合，現状では XML を扱うことはできない．ただし，WebTS からの出力に関しては，以下の方法が考えられる．

1）　静的ページとして，XML 文書を出力する．この場合 WebTS による加工が入らないためそのままブラウザに表示することが可能である．

**表 4　WebConnecter を使用した 3 階層モデルのパフォーマンス**




2)　CPYGEN により作成された COBOL 登録集を使用するトランザクションでは，データ送信用の HTML 文書とフレーム用の HTML 文書を，WebTS が提供する Java アプレットがマージしてブラウザ上に表示する．この処理を XML を使用した形態に変更するには，フレーム出力用の HTML 文書およびデータ出力用の HTML 文書を XML 文書に変換する必要がある．フレーム出力用の HTML 文書は使用者が作成しページ・ファイルに保存するため，HTML 文書を XML 文書に変換して保存することができる．しかし，データ出力用の HTML 文書は，WebTS がダイナミックに作成するため，この方法を実現するには WebTS の改造が必要になる．

3)　WebTSConnector を使用した場合，HTML 文書のマージ処理は WebTSConnector 自身が行う．XML 文書を使用する場合，WebTSConnector に XML 変換

機能を組み込む必要がある．WebTSConnectorがXML文書を認識することにより，データ出力用のHTMLと，ユーザが登録した出力用のXML文書を変換することができる．

このようにXML文書を扱うには，WebTSConnectorが必須になる．今後WebTSConnectorの拡張機能として実現される予定である．WebTSConnecterを使用することによりXML文書の出力はある程度可能であることが分かるが，入力に関してはまだまだ不十分である．HTMLとしての入力は可能であるが，XMLパーサ機能などがシリーズ2200上にないため，XML本来の使用には課題が多い．

## 7. おわりに

今回検証した3階層モデルを使用することにより，従来のアプリケーションをEビジネスに対応したアプリケーションとすることができることが確認できた．サンプルとして中間層に作成したWebTSConnecterは，機能面で十分とはいえないが新たに中間層のプログラムを作成する場合の指針になると思う．Eビジネス環境ではスケーラビリティ，アベイラビリティ，メンテナビリティが求められる．実際のEビジネス・アプリケーションにおいては，アクセス数の予測やビジネス・ロジックの作り方を十分検討する必要がある．加えて，使用する機器の選択も重要な作業である．

**参考文献**
［1］「ECとその関連技術」, CSK技術通信No.32,（株）CSK, 2000年1月
［2］「NTT DoCoMo Net」, http：//www.nttdocomo.co.jp/,（株）NTTドコモ
［3］「KDDI」, http：//www.kddi.com/,（株）DDI
［4］「J-PHONE-TOKYO」, http：//www.j-phone-tokyo.com/, Jフォン東京（株）
［5］「Estimated $4.35 Billion in Ecommerce Sales at Risk Each Year」, News Release, http：//www.zonaresearch.com/info/press/99-jun 30.htm, Zona Research, Inc., 30 June 1999
［6］「セキュリティ対策のキーポイントサーバーの配置を考える」, 日経インターネットテクノロジーNo.035, pp.194-203, 日経BP社, 2000年6月
［7］「EC時代のWebシステム構築」, 日経インターネットテクノロジーNo.035, pp.136-155, 日経BP社, 2000年6月
［8］「Mindcraft-WebStone Benchmark Information」, http：//www.mindcraft.com/webstone/, Mindcraft, Inc.

**執筆者紹介** 末永敏昭（Toshiaki Suenaga）
1976年下関工業高校電子科卒業．同年日本ユニシス（株）入社．以来，シリーズ2200通信関連プロダクトの開発・保守に従事．現在ネットワークサービス部ネットワークソフトウェア室に所属．