Ｗｅｂサイト・パフォーマンスの実態と改善について

富士通エフ・アイ・ピー（株）

■ 執筆者Ｐｒｏｆｉｌｅ ■

宮　原　憲　仁

1982 年　富士通エフ・アイ・ピー（株）入社.
1990 年　シグマ計画で開発・拡販従事.
1995 年　某放送会社気象システム従事.
2001 年　現行の Web パフォーマンスマネージメントサービス業務に従事し現在に至る.

■ 論文要旨 ■

近年の厳しい経営情報において，ビジネスの差異化手段としてＩＴ化が推進されている.

企業の基幹業務システムがオープン化技術を採用した Web システムとして構築されてきている（B2B や B2C）. しかしながら，インタネット・ベースのシステムはオープン技術の組み合わせで容易に構築できる一方，そのシステムの複雑さのために至るところにパフォーマンス上のボトルネックが潜在的に内在している. ほとんどのシステムで設計値通りのパフォーマンスを実現できていない.

我々は，市販の負荷テストツールを採用して，Ｗｅｂシステムの特徴を生かし，リモートから対象のＷｅｂサイトに負荷をかけることにより，Ｗｅｂサイトが設計値までの負荷に耐えられるかどうかのテスト，およびその限界状況でのボトルネックはどこにあるのかの分析を行い，特定したボトルネックの改善に再度の負荷テストを数回繰り返すことにより数倍のパフォーマンス改善を達成した. なお，本テストは弊社の負荷テスト・サービス ActiveTest として商品化している.

■ 論文目次 ■

■ 図表一覧 ■

１．はじめに

我々は 2001 年 7 月より「Web パフォーマンスマネージメントサービス」として，Mercury Interactive 社製 LoadRunner を用いて負荷テストを行うサービス（以後，ActiveTest とする）を実施している．約１年を経過し，多くの企業に対して実施した結果から初期設計時の目標値を達成している企業の少なさに驚ろかされる．ハードウェア，ソフトウェアおよびネットワークに関して豊富な資源がありながらその資源を有効に使いきることなく（現行資源内でどうにか稼動し）運用されているのが一般的な実情のようである．ここでは ActiveTest を実施した現状の実態についてご紹介し，最後に今後の取り組みについて記載する．

２．テストの種類

一般的なアプリケーション・テストの目標として Robert W. Buchanan, Jr が 1996 年に示している．LoadRunner では各目標に対し計測することが可能である．

- **エンド・ユーザの応答時間測定**
- **最適なハードウェアの定義**
- **信頼性の検査**
- **ハードウェアおよびソフトウェアのアップグレード検査**
- **新製品の評価**
- **システムの許容性の測定**
- **ボトルネックの測定**

ActiveTest では，「エンド・ユーザの応答時間測定」，「信頼性の検査」，「システムの許容性の測定」および「ボトルネックの測定」を主眼においている．その環境は様々で**図１**に示すように多岐に渡っている．

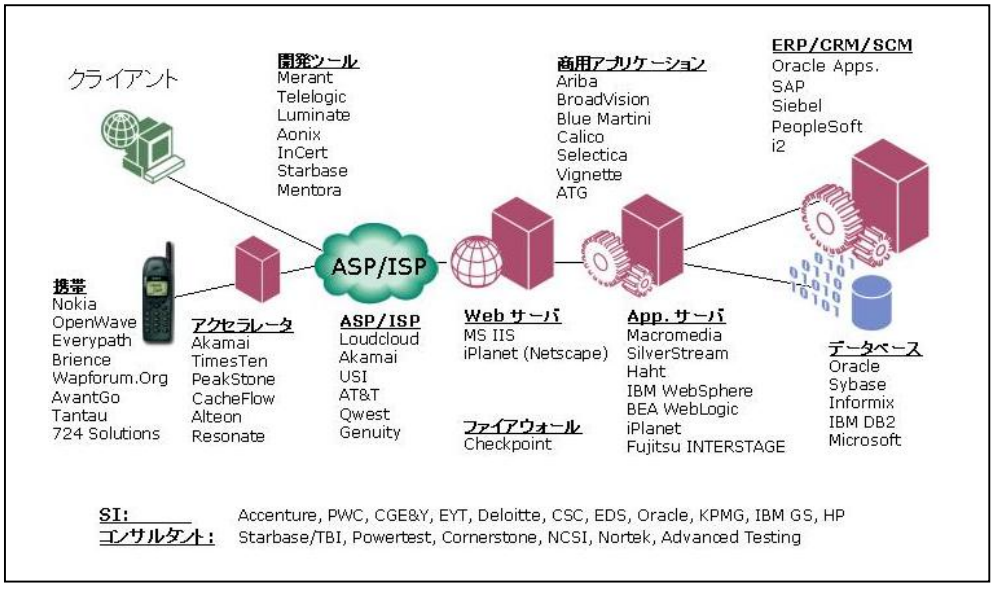

図１　Web を取り巻く環境

２．１　エンド・ユーザの応答時間測定

ビジネス・プロセスが完了するまでの時間を測定し確認する.

エンド・ユーザがビジネス・プロセスを実行し，サーバからの応答を得るまでの時間を調査する．例えばシステムが通常の負荷条件で動作しているときに，エンド・ユーザは 20 秒以内に全ての要求に対する応答を受け取ることを確認できる.

２．２　最適なハードウェアの定義

どのようなハードウェアによって最高のパフォーマンスが得られるか.

様々なシステム構成（メモリ，ＣＰＵ速度，キャッシュ，アダプタ，モデム）がパフォーマンスにどのように影響するかを検査する．システム・アーキテクチャを理解し，アプリケーション応答時間をテストしたら，様々なシステム構成でのアプリケーション応答を測定して，どの設定が望ましいパフォーマンス・レベルを提供するかを判断する.

２．３　信頼性の検査

エラーや失敗なしでシステムがどれだけ高い負荷の下で，どれだけ長い時間動くか.

高い負荷または連続的な作業負荷のもとでシステムの安定度を判断できる．LoadRunner を使って，システムにストレスをかけ，システムに短い時間で広範な動作を実行させることにより，システムが通常は数週間または数ヶ月の期間に経験するような動作状況をシミュレーションすることができる.

２．４　ハードウェアおよびソフトウェアのアップグレード検査

アップグレードすることによりパフォーマンスや信頼性にどのような影響があるか.

ハードウェア又はソフトウェアの新しいリリースと以前のリリースを比較するには，回帰テストを実行する．アップグレードが応答時間にどのように影響しているのか（ベンチマーク），または信頼性にどのように影響しているのか検査できる．アプリケーション回帰テストは，アップグレードの新しい機能を検査はしない．新しいリリースが以前のリリースと同じ程度の効率性と信頼性を持っているかどうかを検査する.

２．５　新製品の評価

どのサーバ（ハードウェア）またはソフトウェアを選ぶべきか.

製品のライフ・サイクルの規格および設計段階で個々の製品やサブシステムを評価するテストを実行することが出来る．例えば，評価テストを基にして，サーバ・マシンのハードウェアを選択や，データベース・パッケージを選択できる.

２．６　システムの許容性の測定

パフォーマンスを大きく低下させることなくシステムがどれだけの負荷に耐えられるかを測定する.

システムのパフォーマンスを低下させずにどれだけの処理が可能か，その余裕度を測定できる．許容範囲の測定には，既存システムでパフォーマンス対負荷を比較し，応答時間の大幅な低下が始まる箇所を特定できる.

２．７　ボトルネックの測定

どの構成要素が応答時間を遅らせているか.

どの構成要素がファイル・ロッキング，リソースの融合，ネットワーク過負荷などのパフォーマンスの低下を引き起こしているか判断するために，システムのボトルネックを特定する. 新しいネットワークおよびマシンの監視ツールと共に ActiveTest を使って，システムの様々な場所で負荷を生成し，パフォーマンスを測定する（**図２**）.

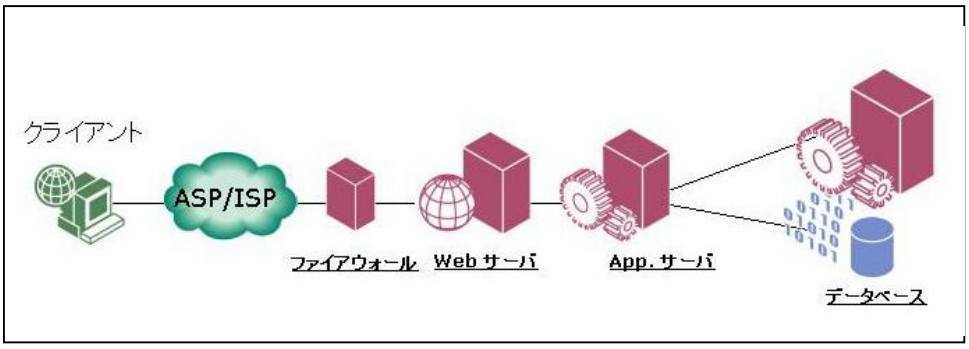

図２　パフォーマンス測定の流れ

ボトルネックの要因として Mercury Interactive 社がまとめた分析結果を**図３**に示す.

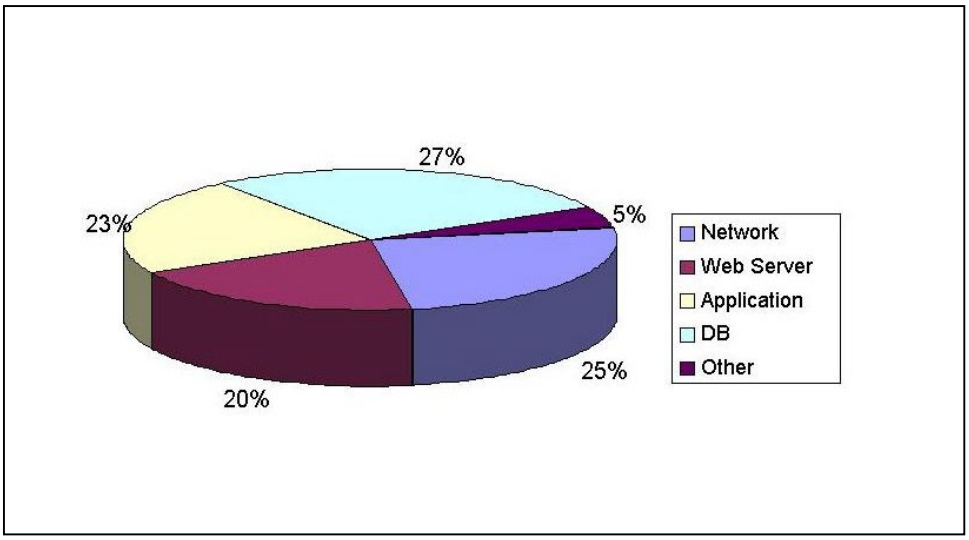

図３　ボトルネックの分析結果

３．現状の実態

ActiveTest は Web アプリケーションシステムに対し性能負荷テストを実施し，2.1 項「エンド・ユーザの応答時間の測定」，2.3 項「信頼性の検査」，2.6 項「システムの許容量の測定」，2.7 項「ボトルネックの特定」について測定する．その結果から分析，改善が行なわれる．

３．１　試験前の性能目標

元来，Ｗｅｂシステムに限らず性能目標はシステム構築時にお客様の要望を聞いて目標が決定され，搭載ハードウェア，ソフトウェア，ネットワークが決定される．

決定される項目は，以下の通り．

【システム要件】

・利用者数
・ピーク時のページビュー数（コンテンツ・アクセス回数）
・ピーク時の許容応答時間
・ピーク時の同時ユーザ数（顧客多重度）
・ピーク時の同時ユーザセッション数（システム多重度）

など

【ハードウェア】

・システム搭載機器（SUN,PRIMERGY...）
・冗長性有り／無し（Cluster）
・負荷分散有り／無し(ロードバランサスキーム)

など

【ソフトウェア】

・稼動 OS(Solaris, Windows2000...)
・ＷｅｂサーバＰＰ(apache,IIS...)
・アプリケーションＰＰ(InterStage,ColdFusion...)
・データベースＰＰ(Oracle,SQL...)

など

【ネットワーク】

・ピーク時の許容帯域幅
・エンド・ユーザ帯域幅（モデム，ＩＳＤＮ，ＤＳＬ...）

これらのハードウェア，ソフトウェアおよびネットワークが個々にどの程度までの許容範囲を持っているかは，個々のベンチマークテストなどで出てはいるが，それぞれがシステムを介して合体したときにどうなるのかは，ある程度の想像は出来てもデジタルな数値では持てないのが現状である．お客様および担当ＳＥ間での現在までの経験的な判断で了解を得てシステム構築に携わっているのが現状である．

ＡｃｔｉｖｅＴｅｓｔを実施するにあたり，システム要件，ハードウェア，ソフトウェアおよびネットワークについて聞き取り調査を行い，テスト目標を決定していく．

よくお客様より「同時仮想ユーザ数とはどのように算出するものなのか」との質問を受けることがある．同時仮想ユーザ数とは，負荷を上げていった時にシステムが本当に重なりあうユーザ数を意味し，ビジネスプロセスの処理時間やコンテンツの構成要素によってかなり違ってくる．以下に例として同時仮想ユーザと帯域の計算例を示す．

【例】同時仮想ユーザ数と帯域の計算例

【テスト要件】

(1) 1 ビジネスプロセス処理時間 ： 3 分

(2) MAX 時間当たりの依頼件数 ： 3.6 万件

(3) MAX 時間当たりの PageView ： 18 万 PV (PV：PageView)

【仮想ユーザおよび使用帯域計算式】

- 1 業務フロー当たりの PV 数 ： (3) / (2) ≒ 5 (PV/件)(4)
- 1 秒当たりの MAX 処理 PV 数 ： (3) / 3,600 ≒ 50 (PV/秒)(5)
- 1 秒当たりの MAX 処理件数 ： (2) / 3,600 ≒ 10 (件/秒)(6)
- 1 秒当たりの**仮想ユーザ数** ： (6) ＊((1) ＊ 60) ≒ 3000
- 1 秒当たりのヒット数 ： (5) ＊ n =50*n (n：1ＰＶ当たりサーバ処理数)
- **帯域**：(5) ＊ m = 50*m*8(m： 1PV 当たりのコンテンツサイズ，1byte=8bit)

３．２ 事例（A 社：認証処理）

ここで事例をご紹介する． A 社ではサービスイン前の最終システム確認を迎えて認証処理のレスポンスに問題認識を持っていた．目標とする同時仮想ユーザ数は１００と想定し算出していたが，どの程度のレスポンスが返されるのかは不明な部分があった．今回 3 回の ActiveTest を実施し問題となるサーバのコンフィギュレーションのみを変更することにより大きな成果を上げた．ActiveTest を実施し成果を挙げた事例としてご紹介する．

(1) ActiveTest 実施に向けた目標値

・同時仮想ユーザ数：１００（１０００～１０,０００エンド・ユーザ想定）

・ログイン許容時間：６０秒以内

・ログイン以外の許容時間：３０秒以内

・サーバ・リソースの利用状況確認（結果にて機器増強有無判断）

(2) ActiveTest 実施前のサービス状況

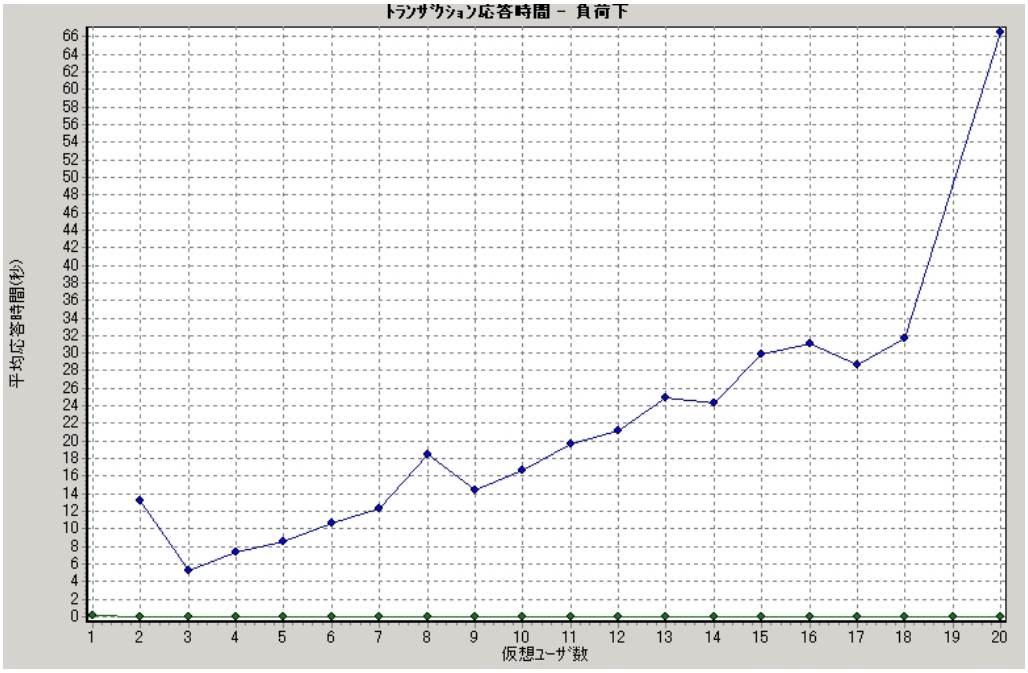

図４　ActiveTest 実施前のサービス状況

仮想ユーザ２０でパフォーマンスが６０秒を超えたため，負荷テストを中断した（**図４**）．これ以上の測定はユーザ条件を超えているため．計測時に並行して測定していたアプリケーション･サーバから DB サーバへのセッション確立する制限値に問題があることが判明．そのための処理が高くなると待ち行列が発生しレスポンスが悪化したことがわかった．（サーバやアプリケーションなどのリソース情報と合わせて検証することにより原因究明が迅速に行える）
セッション数は DB 内のコンフィギュレーション･ファイル内の定義されている値が小さかったため**図４**のようになることが判明した．（定義情報の確認により，原因を特定）
定義情報の変更による他ソフトウェアへの影響（デグレート）が発生しないよう検討し，実機への環境変更を行った．

ActiveTest を実施することにより，レスポンス（時間，失敗/成功），Web リソース（帯域幅，サーバ・ヒット率など），Web サーバリソース，システムリソース，ネットワーク遅延，Web アプリケーションリソース，データベースリソースなどをグラフ化し仮想ユーザ数の推移(徐々に仮想ユーザを増やしながら目標仮想ユーザ数まで到達する)と合わせて検証することが可能となり，原因を特定するのに有効に働くのである．

(3) ActiveTest 実施後のサービス状況

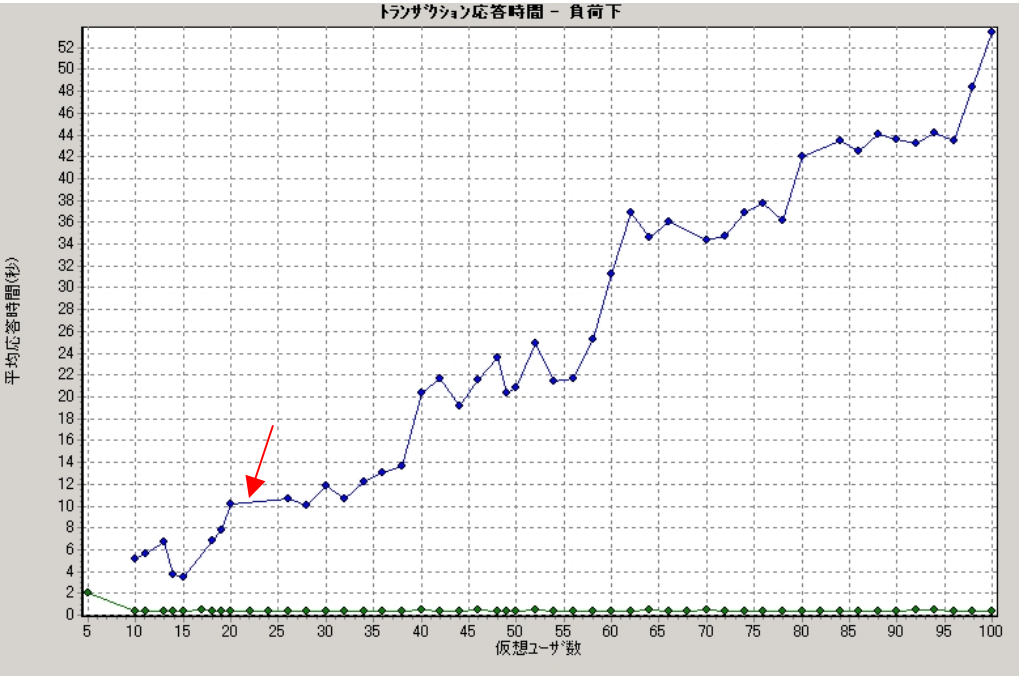

図５　ActiveTest 実施後のサービス状況

矢印の箇所までが ActiveTest 実施前の実績値です．今回の改善処置を施し，再度 ActiveTest を実施したところ目標とする仮想ユーザ１００を達成し，許容レスポンスの１分以内での処理を実現することができた．改善前の仮想ユーザ２０ではレスポンスが１０秒となり６分の１まで短縮し成果を挙げた．

今回の６分の１に短縮されるまでの期間は，環境設定の変更を試行錯誤しながら変更し約５日程度を要して行われた．ユーザ・アプリケーションへの修正は最低限に留めこの期間を実現できた．原因が特定されている場合の環境・システム変更は原因不明で悩んでいるよりも目標があって迅速なものである．
もし ActiveTest を実施していなかった場合，仮想ユーザ２０という爆弾を背負ってのサービスインとなったかもしれない．その時，お客様からのクレームに対し原因究明に追われる日々を迎えることはいうまでもない．

３．３　ＡｃｔｉｖｅＴｅｓｔ実施結果

開発者はシステムの目標応答時間およびエンドユーザの同時アクセス数などを聞き取り調査などで得て，ハードウェアやソフトウェアを決定し，開発者のスキルに依存した部分となり，そのシステムがどの程度の許容範囲を持っているのかは現実的に掴めていないのが現状である．本 ActiveTest では，目標となる応答時間や同時アクセス数およびハードウェア・ソフトウェアの資源がどの程度存在しているのか，実行結果としてデジタルな数値として提示することができる．結果として設計通りになっていない場合などが表面化してみえてくる．

我々が行った実態調査（イントラ内でのクローズテストおよびサイビスイン前のオープ

ンテスト双方を含む）でも以下の情報を得た.

・初回テストで問題が発見されたユーザの割合：９８％
・その時の負荷は設計値の割合：１５％
・6 回程度テストを繰り返し行った場合の効果：5 倍
・チューニングだけで改善されたサイト：７０％
・FireWall の外からのテストだからこそ発見された問題点の割合：３５％

３．４　改善のポイント

ActiveTest を一回の負荷テストに限り実施したのでは，現状把握で終わってしまう．ハードウェア，ソフトウェアおよびネットワークを含むアプリケーションの弱点がどこにあるのか，何回かの負荷テストにより得られたテスト結果をレビューし，個々のコンフィギュレーションの改善を行うことによって仕上げていくのが重要である．アプリケーションはハード・ソフトウェアおよびネットワークといった環境下で動くため，それぞれに最適なコンフィギュレーション環境が存在していることを認識しなければならない．そのためには，同一条件での負荷テストの実施を繰り返しながら最適な環境に導いていくことが重要である.

４．考察

テスト実施を行って，同一のパッケージを使用している顧客がある場合，想定される問題点（改善項目）について事前に指摘できる環境になってきた．当然新しいものにはノウハウが無いため明確にならない部分もあるため今後のノウハウの蓄積課題となる．ハード・ソフトウェア・ネットワークで共通に扱える部分が多々あることから負荷テストを実施する前段階での事前コンサルティングが可能な環境を整備し，顧客への負荷テストの有効性を高めたい.

５．むすび

今回ご紹介したActiveTestは負荷テストを行いWebサイトの弱点を早期に発見し改善することにより信頼性の高いものに築きあげていく有用なサービスと自負している.

また「Web パフォーマンスマネージメントサービス」には性能監視を行うサービス（ActiveWatch）も実施している．Web サイトレスポンスの状況を１５分おきにエンド・ユーザに代わって操作を行いレスポンスを計測し，Web で状況や改竄を確認することが可能である.

「運営している Web サイトの性能が出てないが検証方法はないのか」，または「Web サイトの性能に不安を抱いている」というシステムであればシステムの向上のために，この負荷テストを実施し，サービスイン後は監視サービスを行うアプローチを推薦する.

以　上