2003年日本オペレーションズ・リサーチ学会
春季研究発表会

Legacy連携型分散オブジェクトシステムのモデル化と性能評価

01206780　(株) NTTデータ 技術開発本部　　*坂田　洋幸　SAKATA Hiroyuki
01605320　筑波大学大学院 ビジネス科学研究科 牧本　直樹　MAKIMOTO Naoki

1. はじめに

近年、企業内基幹系システムや大規模Webサイト実現におけるサーバ構築基盤として、J2EE(Java2 Platform,Enterprise Edition)[1]に代表される分散オブジェクト技術が注目されており、標準・実用化が進められている。

その背景には、同技術の本質的一側面である「ネットワーク透過なオブジェクト相互接続・実行環境」を導入することでソフトウェアの部品化・再利用を促進し、開発工期短縮や運用時のメンテナンス性を確保するねらいがある。

しかしながら上述のメリット実現のみならず、性能面をはじめとするシステム品質確保のためには、システム分析・設計といった開発上流工程の段階から考慮すべき課題が多い。

特に性能面に着目した場合、分散オブジェクト環境上でのアプリケーション処理はネットワーク上に分散配置されているサブシステム上のオブジェクトの相互作用により構成されるため、分析・設計工程でのオブジェクト抽出・定義作業がシステム稼動後のネットワークトラフィックやシステム全体の性能に大きく影響する。

一方で最近では、勘定系など従来より開発・メンテナンスされてきたシステム(Legacy)の機能を新規開発システムの一部として取り込み、より高度なサービスとして提供する「統合サービス」への要求が高まっている。その実現のためには分散オブジェクト環境とLegacy間での相互接続・機能連携が必然となるが、両者ではデータ表現形式や通信プロトコル、また応答時間分布などのトラフィック特性が大きく異なるため上流工程段階からのボトルネック特定・性能見積といった品質確保のアプローチがより不可欠となる。

これまで、同様の問題意識にて主にHTTPサーバを対象とした性能評価モデルの研究報告がなされている[2]。

本研究ではこれら既存研究を参考としながら、分散オブジェクトとしての特質ならびにボトルネック要因を孕んだLegacy連携に焦点をあてたモデルベースの性能評価アプローチ確立を目標とする。

2. 研究のアプローチ

以下のアプローチにてモデル構築と精度向上を進める。

1. 論理モデルの構築
   標準仕様に基づき、分散オブジェクトシステムのサブシステム構成ならびに処理実行時のオブジェクト間相互作用に着目した論理モデルを構築する。

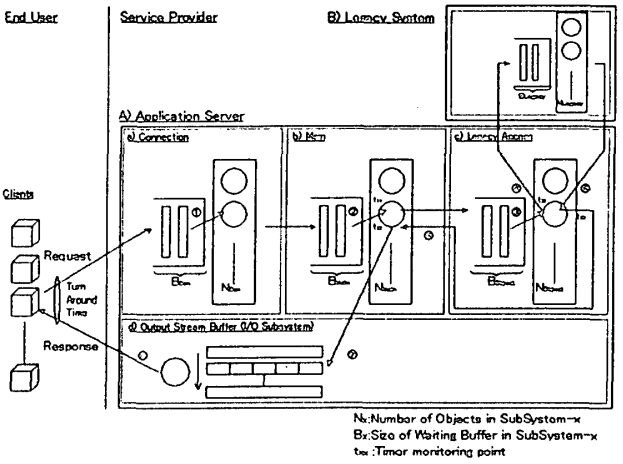

図. 1.　論理モデル

2. 実機上でのAP実装と実測によるモデル検証
   実機上に評価用アプリケーションを実装し、複数の処理要求をサーバに対して同時に発行する「多重アクセス試験」と一連の処理要求をPoisson過程に従って発生させる「連続アクセス試験」を実施する。多重アクセス結果から得られる応答特性に基づき論理モデルの妥当性を検証し、分布に従った測定結果からは各サブシステム内処理遅延の分布を特定し、以後シミュレーションモデル構築時の入力とする。

3. シミュレーションによるシステム内挙動解析
   これまでで確定した論理モデルに基づきシミュレーションモデルを構築する。同モデルの実行を通し、実機上では観測しきれない各サブシステム内の「オブジェクト活性化状況」や「処理待ち部における待ち行列長分布」、「各オブジェクトの稼動率」などの確率的特性を把握する。

4. 解析モデルへの展開
   上記を通してモデルの静的・動的な特性を明らかにした後に、待ち行列解析アプローチを中心とした理論モデルに展開する。

3. 論理モデル概要

3.1. 論理モデル

構築した分散オブジェクトシステムの論理モデルを図1

に、モデル構成要素の概要を以降の小節に示す。

3.2. サブシステム構成

分散オブジェクト環境上の各オブジェクトは、主にその処理種別やライフサイクル毎に、異なるサーバ内サブシステムによって管理される。各サブシステムは独自の規律に従って管理対象オブジェクトへの処理割当てを制御する。

システム案件により導入するサブシステムの種類・構成は様々であるが、今日のサーバ形態の主流である"HTTP ベースの Client/Server"を実現する上では、下記のサブシステムが最低限の基本構成要素としてあげられる。

a) Connection 制御
エンドユーザ側に位置する Client との下位レイヤ（一般には TCP/IP）レベルでのコネクション確立を制御する。

b) Main
システムの前方に位置し、コネクション確立後に Client から送信されてくる Request(処理要求）に対する一元的な窓口となる。受信した Request は複数の Method（サブタスク）に分割され、各 Method の処理は後方に位置する複数のサブシステムに適宜委譲される。全ての Method の処理終了後、Main 処理部は各処理結果を集約して Client への Response（処理応答）を形成し、I/O サブシステムに送出する。冒頭で述べた J2EE をベースとする場合、本サブシステムは Servlet 技術により実現される。

c) Legacy 連携部
システムサービスの中核となる業務ロジックと同処理中から適宜呼び出される Legacy システム連携機構を提供する。特に Legacy システム連携部は一連の Legacy 接続処理を内包するため、同処理を実装している Method 自体の挙動が、後述する Legacy システムとしての特性を示す事になる。本サブシステムは、J2EE では主に EJB(Enterprise Java Beans) 技術により実現される。

d) I/O
ネットワーク出力や DiskI/O 等の入出力機構を制御する。

3.3. Legacy システム

独自インタフェースやアクセス方式など、Legacy システムとしての特質・側面は様々な観点で存在するが、本研究では特に性能品質への影響に着目し、下記の特性を示すサブシステムを Legacy の本質として捉える。

- 相互接続時にプロトコル変換などのオーバヘッドが不可避であるため、他オブジェクトの Method 実行に比べて、その処理応答時間が大きい。
- 既に独自のポリシーによってサービスを公開している性質上、一般に Legacy 以後は異なるサービス体系（課金形態等）となり、アクセスそのものに制約が存在する。本研究では同時アクセス数制限として捉える。

3.4. Method 実行形態

一般に、ある Method(A) の処理過程で更に別の Method(B) を起動する場合、Method(B) の処理が終了し、その応答を受け取るまで呼出元の Method(A) の処理をブロックする「同期型」が一般的な処理形態となる。この原則は分散オブジェクト間での Method 呼出時にもあてはまるため、各 Method の依存関係と各々の応答特性により、サブシステムの状態が大きく変わる。

4. 実機評価

4.1. 評価項目・目的

SingleProfile 評価
システムに対して単一 Client からアクセスした場合の基本処理性能を取得し、以後の評価結果との比較対象とする。

多重アクセス評価
システムに対して複数 Client から同時にアクセスし、その応答特性から内部の処理待ち状況を推測することで、前述したサブシステム構成の妥当性を確認する。Client 多重度の設定は、システムパラメータとして設定する各サブシステム内オブジェクト数の上限値に基づく（図 1 中の N_X）。

Poisson アクセス評価
システムに対するアクセスを率 λ の Poisson 過程に従って断続的に発生させ、システム内部の確率的挙動を把握する。

なお、評価実施時の実機上の設定として、各サブシステム内処理待ち部（図 1 中の B_X）は十分に確保し、呼損は発生させないものとする。また、今回の評価における Legacy 部としては「アクセス制限を有し、処理要求を受けてから一定時間経過後に固定の応答を折り返す」擬似的な機構を導入した。

4.2. 評価指標

Turn Around Time
Client 側の計測項目であり、Client 側からシステムに対して Request 送出をはじめてから Response を完全に受け取るまでの時間をあらわす。

Method 処理時間
Server 側の計測項目。Client 側からの Request をうけて起動する、各サブシステム上のオブジェクトの Method 実行時間をあらわす。図 1 中の t_{XX} のポイントにプログラム的に計時機構を埋め込む事で取得する。

4.3. 評価結果と考察

評価結果詳細とその解釈ならびにシミュレーションへの展開については発表当日に述べる事とするが、基礎データとして SingleProfile 評価からの「Turn Around Time」、「Method 処理時間」取得結果を以下に示す。（単一 Client から一定間隔で 3000 回システムにアクセスして取得した各データを、バッチサイズ 100 とするバッチ平均法にて処理した）

	総平均	95 %信頼区間
Turn Around Time	2013.994	1.263
Main($t_{12} - t_{11}$)	2008.609	0.352
Legacy($t_{22} - t_{21}$)	2003.045	0.201

表. 1. SingleProfile 結果（単位：[msec]）

参考文献

[1] Sun Microsystems: "Java2 Platform,Enterprise Edition", http://java.sun.com/j2ee/overview.html.

[2] Van der Mei et al.: "Web Server Performance Modeling", Telecommunications Systems,16,pp.361-378(2001).