LC-001

Java アプリケーション多重実行における データベースアクセス管理方式

Database Access Management for Redundant Java Application on 3-Tier Service Model

藤山 健一郎　Ken-ichiro FUJIYAMA
中村 暢達　Nobutatsu NAKAMURA
平池 龍一　Ryuichi HIRAIKE

1. はじめに

IT システムには 24 時間常時運用を実現する耐障害機能が求められている。それを実現する手法の一つとして、アプリケーション（AP）を多重に実行し、一次（現用）系 AP で障害が発生しても、二次（バックアップ）系 AP が処理を引き継ぐようにすることでサービスを継続する手法[1][2][3]がある。

しかし、多くの AP は、外部リソースと連携することで処理を行うため、単純に AP を多重実行するだけでは耐障害機能を実現できない。例えば三層モデルに基づく WEB-AP の場合、AP はデータベース（DB）と連携しサービスを提供する。しかし、図 1に示すように、多重実行された AP が 1 つの DB を共用する場合、複数の AP が DB を重複して更新してしまうという問題がある。DB を系毎に用意するという回避策もあるが、DB を複数用意することは、コストの面から実用的ではない。

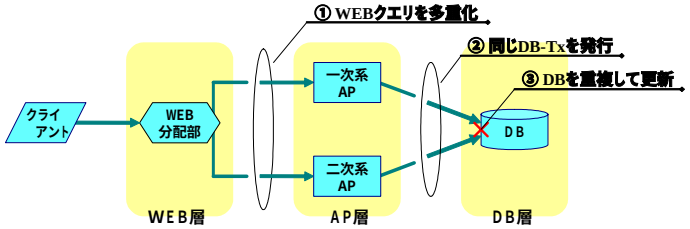


図 1：データベースの重複更新

本稿では、このような問題を解決するために、AP 多重実行環境において、共用する単一の DB への矛盾の無いアクセスを実現する方式について述べる。

2. データベースアクセス管理

2.1. 提案方式

DB へのアクセス、つまり DB トランザクション（Tx）を、 図 2に示すように制御する。

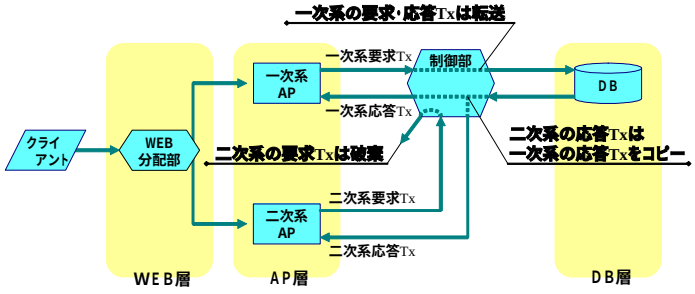


図 2：トランザクション制御

- 一次系 AP から DB への要求トランザクションは、そのまま DB に送信し、処理する。処理結果である応答トランザクションも、そのまま一次系 AP に返送する。その際、応答トランザクションをトランザクション毎に識別して記録する。
- 二次系 AP から DB への要求トランザクションは廃棄する。処理結果としては、一次系で記録された各トランザクションに対応する応答トランザクションをコピーして二次系 AP に返送する。なお、二次系 AP の方が先に動作し、対応する応答トランザクションがまだ記録されていない場合は、一次系で記録されるまで待機する。

以上のように、二次系 AP は実際には DB アクセスを行わないが、あたかも DB アクセスを行ったかのように応答トランザクションを受け取るようにすることで、DB を重複して更新することなく、一次系と同様の処理を行うことができる。

また、一次系 AP において障害もしくはその予兆を検出した場合、異常動作の可能性のある一次系 AP からの DB トランザクションを破棄し、二次系 AP からの DB トランザクションを DB に転送するようにする。一次系 AP と二次系 AP は同じ処理状態にあるので、このようにアクセス経路を切り替えるだけで、初期化処理等を行うことなく、二次系 AP が高速に処理を引き継ぐことができる。

2.2. 実装

一般的な三層モデルにおける AP 層と DB 層間の DB トランザクションの経路を図 3に示す。

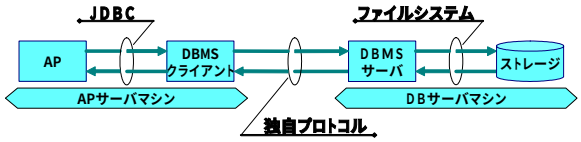


図 3：DB トランザクション経路

まず、AP からの要求トランザクションは、DBMS（Data Base Management System）クライアントに送られる。DBMS クライアントはネットワークを介して、DBMS サーバにその要求トランザクションを送信する。DBMS サーバはストレージに対し、要求トランザクションに応じた処理を行い、その結果である応答トランザクションを DBMS クライアントに返信する。最後に、DBMS クライアントが、その応答トランザクションを AP に返す。このように DB トランザクションは、大別して 3 つのインタフェースを経由する。以下に、3 つのインタフェースにおける DB トランザクション制御のための機能拡張性について述べる。

- **AP－DBMS クライアント間インタフェース**
  WEB-AP 等でよく用いられる J2EE プラットフォームにおいては、JDBC[4]が一般に用いられる。JDBC はオープン仕様の API のため、機能拡張が容易である。
- **DBMS クライアント－DBMS サーバ間インタフェース**
  DBMS 内の通信であるため、DBMS ベンダ独自のプロトコルに基づくインタフェースが用いられており、機能拡張は困難である。

NEC インターネットシステム研究所
Internet Systems Research Laboratories, NEC Corp.

- **DBMS サーバーストレージ間インタフェース**
  DB サーバ上の OS のファイルシステム等が相当する。この部分は DBMS の性能に大きく影響するため、高度なパラメータ調整が行われており、性能を維持したまま機能拡張を行うことは困難である。

以上の特徴を鑑みて、AP－DBMS クライアント間インタフェース、つまり JDBC を拡張して、DB トランザクションの制御機能を実装した。なお、JDBC を用いることで、それ以外の構成要素の差異は JDBC によって吸収されるため、DBMS や OS 等を限定しないという利点もある。

3. 評価実験

提案した DB アクセス管理方式を実装し、実用性を検証するために、実運用されている ASP 製品に適用して、評価実験を行った。

3.1. 実験環境

図 4に示すように、一次系 AP サーバマシン(AP1)、二次系 AP サーバマシン(AP2)、DB サーバマシン(DB)、そしてクライアントマシン(CLIENT)の４台のマシンを 100Base-T イーサネットで接続した。AP1、AP2 には、AP サーバとして BEA WebLogic Server 8.1、Java 実行環境として Sun JDK 1.41、WEB-AP として住宅業向け工事発注 ASP サービス[5]をそれぞれ使用した。DB には、DBMS として Oracle9i Database Standard Edition を使用した。CLIENT には、WEB クエリ発生ツールとして Microsoft Web Application Stress Tool 1.1[6]を使用した。

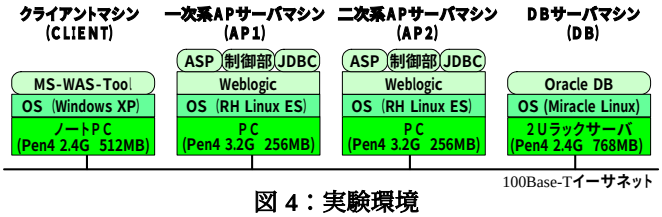


図 4：実験環境

3.2. 実験方法

DB アクセス管理方式のオーバーヘッドを評価するため、管理機能の無い通常状態と、管理機能を適用した状態とで、WEB-AP に対し、以下の動作を行う WEB クエリを、それぞれ 30 秒間連続で投入し、その処理数を計測した。

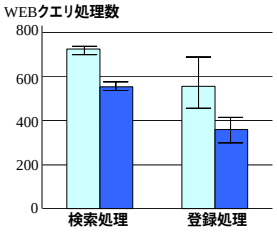
- 検索：1000 件の発注データから特定の発注データを検索
- 登録：発注データを新規で登録

なお、WEB クエリには、WEB-AP への処理要求だけでなく、イメージの取得要求など、DB トランザクション発行に至らないものも含まれる。また、DB トランザクション発行に至る WEB クエリも、その処理内容によって発行する数は異なる。そこで、オーバーヘッドの詳細を検討するために、DB トランザクション発行に至る WEB クエリについて、個々の処理時間と、発行される DB トランザクション数も計測した。

3.3. 実験結果

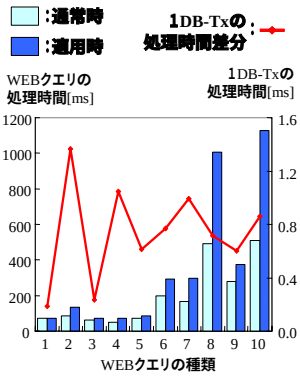
WEB クエリの処理数を計測した結果を図 5に示す。DB アクセス管理機能を適用することにより、全体で 25～35%程度のオーバーヘッドが発生している。

さらに、実際に DB トランザクションを発行する WEB クエリ 10 種の処理時間と、発行される DB トランザクションの数を計測した結果を図 6に示す。なお、時間の単位はミリ秒である。通常時と適用時との WEB クエリの処理時間の差分は、発行される DB トランザクションの数にほぼ比例しており、１ DB トランザクションあたり平均 0.8 ミリ秒のオーバーヘッドが発生している。



処理	WEBクエリ処理数		比率
	通常時	適用時	
検索	723.7	555.0	76.7%
登録	552.7	359.0	65.0%

図 5：WEB クエリの処理数



WEB クエリ 種類	発行 DB-Tx 数	WEBクエリ 処理時間			1DB-Tx 処理時間
		通常時	適用時	差分	差分
1	22	70.0	74.1	4.1	0.2
2	36	85.1	134.2	49.1	1.4
3	22	64.9	70.1	5.2	0.2
4	22	47.7	70.7	23.0	1.0
5	22	72.0	85.5	13.5	0.6
6	127	197.2	294.8	97.6	0.8
7	132	166.6	297.6	131.0	1.0
8	721	490.0	1005.5	515.5	0.7
9	154	279.9	372.8	92.9	0.6
10	721	507.8	1126.7	618.9	0.9
合計	1979	1981.2	3532	1550.8	平均:0.8

図 6：WEB クエリ・DB トランザクションの処理時間

3.4. 考察

DB アクセス管理方式を適用することによりオーバーヘッドが発生するが、その値は、データベース処理や人間の操作による待ち時間に比べれば、十分に小さいものである。よって、このような処理や操作が発生する WEB-AP 等においては、本方式は十分実用的であると考える。

4. おわりに

本稿では、多重実行された複数の AP が、共用する単一の DB に矛盾無くアクセスを行うための DB アクセス管理方式について述べた。本方式は、JDBC の拡張であるため、OS や DBMS を限定せずに適用可能である。さらに、実運用されている ASP 製品に適用して評価実験を行い、WEB-AP 等においては十分実用的であることを確認した。

今後は、オーバーヘッドの軽減や、マルチスレッド等により多重実行時に一次系 AP と二次系 AP とで動作が異なる場合への対応などを行う予定である。

参考文献

[1] J. Napper, L. Alvisi, H. Vin, ” A Fault-Tolerant Java Virtual Machine”, DSN2003.
[2] R. Koch, S. Hortikar,, et al., “Transparent TCP Connection Failover”, DSN2003.
[3] 中村 他, “耐障害に向けたサービスアプリケーション多重実行方式の提案”, 情処 66 全大.
[4] JDBC API：http://java.sun.com/j2ee/ja/jdbc/
[5] 住宅業向け工事発注ＡＳＰサービス プレスリリース：http://www.nec.co.jp/press/ja/0212/2401.html
[6] MS- WAS Tool：http://www.microsoft.com/technet/itsolutions/intranet/downloads/webstres.mspx