社団法人 情報処理学会 研究報告
IPSJ SIG Technical Reports

2007－ＥＩＰ－36（１）
2007／5／25

ログ検査を目的としたAP/DBログ関連付け機能の実装と評価

山中 真和† 榎本 俊文† 谷口 展郎† 山室 雅司†

† NTTサイバースペース研究所

E-mail: †{yamanaka.masakazu,enomoto.toshifumi,taniguchi.noburou,yamamuro.masashi}@lab.ntt.co.jp

あらまし SOx法等に対応する内部統制の確立に向けては，複数の業務アプリケーションとそれらがアクセスするデータベースに対し統合的なログ監査を行うことで，業務プロセスの正当性を検証できる．しかし，現在のWeb/AP/DB三階層型システムにおいては，AP/DB間においてコネクションプーリングが行われるため，AP/DBに跨るログの関連付けは困難である．本論文では，AP/DBログ関連付け機能を提案し，その実装・評価結果を示す．性能評価では，ログ関連付け機能を実行時オーバヘッド5%程度に抑えつつ実現することが出来た．

Implementation of Facilities to Integrate AP/DB Logs for Deep Inspection

Masakazu YAMANAKA†, Toshifumi ENOMOTO†, Noburou TANIGUCHI†, and Masashi YAMAMURO†

† NTT Cyber Space Laboratories

E-mail: †{yamanaka.masakazu,enomoto.toshifumi,taniguchi.noburou,yamamuro.masashi}@lab.ntt.co.jp

Abstract The deep inspection of AP and DB logs in multiple business systems enables to validate the process in the system, and is helpful to realize internal control. DBMS doesn't support the method to combine the AP and DB logs with utilization. This paper proposes facilities to integrate AP/DB logs for the deep inspection efficiently. In the performance evaluation, our log integration facilities show only 5% runtime overheads against programs without the integration facilities.

1. はじめに

2008年施行予定の金融商品取引法（通称日本版SOx法）により，企業における内部統制の強化・対策の必要性が高まっている[1]〜[3]．これに伴い，企業内で利用される業務アプリケーション（以下，業務ＡＰ）においても，各種ログの記録や監視などによるシステム挙動の動作保証が必須となっている．

[2],[3]では，内部統制の6つの基本的要素として，(1)統制環境，(2)リスクの評価と対応，(3)統制活動，(4)情報と伝達，(5)モニタリング，(6)ITへの対応が挙げられ，これらは内部統制確立の基礎である．これらの中でも特に，"モニタリング"と"リスクの評価と対応"を実現するための方法として，業務ＡＰの動作保証を目的とした各種ログの記録や監視は重要である．

まず，"モニタリング"は，内部統制の有効性を継続的に監視及び評価するプロセスを意味し，業務での各プロセスを証跡として残すことが重要である．ITシステムおいては，システムログの記録がこれにあたる．

次に，"リスクの評価と対応"は，組織の目標の達成に影響を与えるすべてのリスクを識別，分析及び評価することによって，当該リスクへの対応を行う一連のプロセスを言う．ITシステムおいては，システムが意図した通りに動作していることを継続的に監視することが求められる．ログ検査により，業務ＡＰの挙動を監視することで，意図しない誤操作や不正な操作を検出することができる．

業務ＡＰでは，一般的に良くWebサーバ，アプリケーションサーバ（以下APサーバ），DBサーバの構成が利用される（図1）．業務ＡＰの挙動を詳細に検査するためには，これらWeb/AP/DBサーバに分散したログを統合して検査する必要がある．しかし，現状では，APログとDBログを関連付けることは困難である．

我々は，PostgreSQLを対象に，AP/DBログを関連付けるための機能を考案し，実装及び評価を行った．本論文では，それら手法による性能面での影響を紹介する．加えて，PostgreSQL

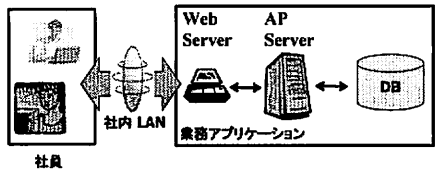

図 1　業務ＡＰの構成例

のログ機能を利用した場合の性能特性を紹介する．

論文構成は，以下の通りである．まず，2. 章で，内部統制の確立を意図したログ検査の事例を紹介する．その後，3. 章にて，AP/DB ログの関連付け手法について述べ，4. 章にて，性能評価を行う．5. 章にてまとめる．

2.　ログ検査の事例

本章では，内部統制の確立に有効となるログ検査の事例を紹介する．

図 1 に，業務ＡＰとして一般的に良く利用される Web/AP/DB 構成を示す．図 1 のような構成では，その検査対象が Web/AP/DB サーバと複数に渡る．そのため，良く実施される Web サーバ上のログ検査にとどまらず，AP や DB サーバ上のログを含めた全アクセスに対する検査が重要である．特に，データベースには，個人情報や金銭に関わるデータなどの重要な情報が格納されることが多いため，DB 上のログを含めた全体のシステム挙動を詳細に検査することは重要である ([3], [4])．実際，各種ログを保管・処理する商用システムが存在する ([5]〜[9])．

Web サーバで記録される情報 (例えば，ユーザの IP アドレス) は，一般的に AP サーバ上で取得することが可能であり，AP サーバ上で Web サーバでのログ内容を記録することは可能である．そのため，特に AP ログと DB ログを統合し，検査に利用することが重要でなる．具体的な検査としては，DB ログを検査する際に，AP ログの情報を利用することで，AP レベルのユーザ単位で DB アクセスを検査し，権限のないユーザがアクセスすべきでないデータへアクセスしているといった異常を発見することが挙げられる．

3.　AP/DB ログの関連付け

3.1　課　　題

検査を実施するために，分散したログを統合化する場合，AP ログと DB ログに跨ったデータを関連付けることが必須となる．例えば，図 2 のような Web/AP/DB に跨ったログデータを統合し，業務ＡＰ全体としての挙動について検査を行うためには，関係するひとまとまりのログデータを関連付けることが必要になる．

AP サーバと DB サーバに分散したログデータを統合化するため，関係するひとまとまりのログデータに対して関連付ける必要があるが，DBMS が提供している通常の機能を利用してもログ統合を実現することは容易ではない．例えば，DBMS は DB アクセスを行っているユーザを示すために，ログインユーザを指定しなければならないが，ログインユーザ名をキーの代わりとして利用することはできない．ログインユーザの指定をキーとして利用するためには，ユーザからの要求毎に DB 側に対してログインユーザを指定する必要がある．但し，ログインユーザの指定は，AP と DB のネットワークコネクション確立時に限られるため，ログインユーザを指定するためには，AP へ要求がある度にコネクションを切断・確立を行う必要がある．AP から DB へのコネクションの確立はコストの高い処理であるため，通常，頻繁なコネクションの確立・切断を避けるため，一度確立したコネクションを切断せずに次回の AP から DB へのアクセス時に再利用するというネットワークコネクションのプーリングが行われる．そのため，ログインユーザの指定をキーとして利用することはできない．

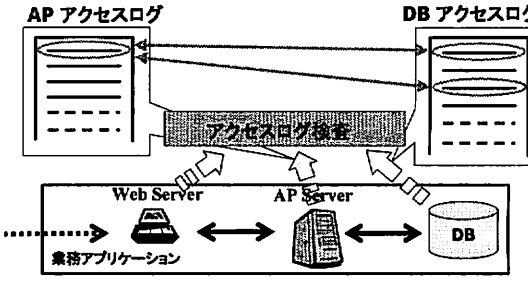

図 2　AP/DB ログ検査

3.2　AP/DB ログ関連付け機能の実装方針

AP/DB ログを関連付ける方式を提案する．AP/DB 間でのログを関連付けるための基本方針として，あるキーを発行し，AP/DB 間で共有／利用するという方法を採用する．キーの発行方式としては，AP サーバ側でキーを管理する方式と PostgreSQL 側でキー管理を行う方式の 2 種類を提案する．加えて，PostgreSQL の標準ログへのキーの記録も考慮している．

AP サーバ側でのキー管理

AP サーバ側でキーを管理する場合，どのようにしてキーを DB 側に伝えるかが問題となる．本方式では，PL/pgSQL [10] を使用した図 3 のような空のストアドプロシージャをキーの受け渡しのために利用する．このストアドプロシージャの引数としてキーを与え，実行することで AP 側から DB 側にキーを渡し，DB 側で受け取ったキーをログとして記録する．

AP サーバ側では，一連の DB アクセスの実行前 (もしくは，後) に図 3 のストアドプロシージャを呼び出し，合わせて，管理するキーを AP ログにも記録しておく．図 4 は，このストアドプロシージャを利用した JAVA プログラムである．これにより，DB 側では，図 5 (右下) のようにログが記録されるため，記録したキーを元に，関連する DB アクセスと AP アクセスのログを抜き出すことができる．

AP 側でキーを管理した場合，アプリケーション開発者にとって負担もあるが，一方で，PostgreSQL に対する改造の必要はないという利点もある．

PostgreSQL によるキー管理

```
CREATE LANGUAGE plpgsql;
CREATE FUNCTION ApDbKey (TEXT) RETURNS VOID AS'
  DECLARE
  BEGIN
  END;
' LANGUAGE 'plpgsql';
```

図 3　ストアドプロシージャの例

```
...
String str = req.nextReq(); // ユーザからのリクエストを受け取る
String key = keyGen.getNextKey();
outputApLog(str, key);
execJDBC("SELECT ApDbKey(" + key + ")");
execJDBC("SELECT tab0.f0 WHERE f1 = " + str + ");
...
```

図 4　AP 側での呼び出しの例 (AP 側でのキー管理)

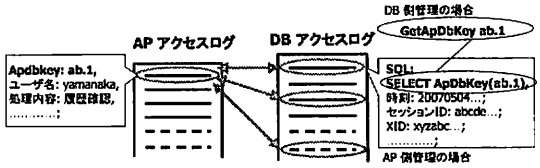

図 5　ログ記録のイメージ

```
...
String str = req.nextReq(); // ユーザからのリクエストを受け取る
String key = execJDBC("GetApDbKey;"); // 独自コマンド呼び出し
outputApLog(str, key);
execJDBC("SELECT tab0.f0 WHERE f1 = " + str + ");
...
```

図 6　AP 側での呼び出しの例 (DB 側でのキー管理)

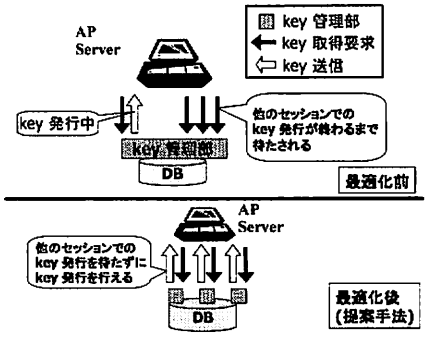

図 7　PostgreSQL による関連付けキーの発行

PostgreSQL 内部でキー管理を行う場合，発行したキーを AP サーバ側に渡すために，PostgreSQL の改造による独自コマンドを作成した．AP 側でのキー管理の場合と同様，AP 側で，このコマンドを一連の DB アクセスの実行前 (もしくは，後) に呼び出せば良い．このコマンドを呼び出すと，PostgreSQL 内部でキーを生成し，DB 側のログとして記録すると同時に，AP 側にそのキーを返却する．図 4 は，この独自コマンド (GetApDbKey コマンド) を利用した JAVA プログラムである．これにより，DB 側では，図 5(右下) の "SELECT ..." の部分が図 5(右上) のように変更されたログが記録され，記録したキーを元に，関連する DB アクセスと AP アクセスのログを抜き出すことができる．

一意なキーを AP サーバ側に発行する場合，単純に実装してしまうとボトルネックを招く可能性もある．例えば，図 7 上のように，AP からキーの取得要求があった場合，全体で一意なキーを作成する方法が考えられる．しかし，他のキー取得要求を一切受け付けず，あるキーの発行が済んでから次のキー取得要求に応じることになるため，データベースに対して集中アクセスがあった場合，キー取得を要求しても待ちが発生し，ボトルネックの原因となる．

そのため，本方式では，PostgreSQL 内部で，各コネクション毎に管理されているセッション ID を利用してボトルネック発生を回避している．セッション毎に AP/DB 共有用のキーを管理することで，複数のセッションからキー取得要求があった場合でも，それらを並行に実行することができ，待ちが発生しない (図 7 下)．

PostgreSQL によるキー管理では，AP 側でキーの管理を意識する必要がないため，利用は容易である．

表 1　評価環境

CPU	Pentium4 Xeon 2.8 GHz x2 (Hyper Threading により x4)
メモリ	2 Gbytes
ディスク	1 disk
OS	Red Hat Enterprise Linux 4
DBMS	PostgreSQL 8.1.4
	SUN jdk1.5.0+JDBC3.0

4. 評　　価

AP/DB 関連付け機能について性能評価する．評価環境を表 1 に示す．

測定内容としては，クライアントが各アクセスを要求してから，要求を完了するまでの時間をレスポンスとして測り，加えて，スループットを計測する．測定対象のレスポンスを測定した値の平均値をレスポンスとして示す．

TPC-C ベンチマークを利用して，性能評価する．測定内容として，PostgreSQL ログ機能を利用しつつ，AP/DB 関連付けを行った場合の性能を計測する．

また，合わせて PostgreSQL ログ機能の性能評価も併記する．

比較対象は，

- **NoLog**:ログ機能を一切使用せずに，単純に TPC-C を実行した場合
- **Logging**:ログ機能を利用してディスク内にログを保存した場合

Logging のログ保存に加えて，AP/DB 関連付け方式を適用した場合それぞれ，

- **APDBKey0**:クライアント側でのキー管理方式

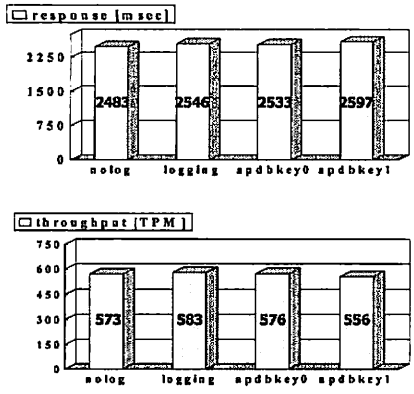

図 8 性能結果 1

- **APDBKey1**: PostgreSQL 側でのキー管理方式

の 4 種類を比較する．**APDBKey1** のみ PostgreSQL に対して改造を行っている．

クライアントに利用するマシンとして評価環境と同じ性能のマシンを用意した．TPC-C における warehouse 数を 10 に設定しており，データベースサイズは，1 Gbytes 程度と比較的小さいサイズであるものの，ディスク数が 1 であり，vmstat コマンドによると I/O ボトルネックとなっていた．

クライアント数を，16 として，15 時間連続で TPC-C を実行した場合のレスポンスとスループットの測定結果を図 8 に示す．

まず，PostgreSQL ログ機能について述べる．**NoLog** と **Logging** を比べると，レスポンスについては，ログを記録しているためか **Logging** が 2.5%程度遅くなっている．一方のスループットについては，逆に，**NoLog** に対して 1.7% の速度向上を果たしている．これについては，速度低下が測定誤差に対して非常に小さいためと考えられる．実際，同条件にて複数回同一プログラムを測定した場合に，5 %程度の測定誤差が見られた．つまり，**NoLog** と **Logging** の測定結果から，レスポンス，スループットともに数% 以下程度のオーバヘッドで PostgreSQL ログ機能を用いることができ，性能面では，十分実用的であると言える．

次に，3. 章で述べた AP/DB アクセス関連付けを行う **APDBKey0**, **APDBKey1** について述べる．まず，**Logging** と **APDBKey0** とを比較すると，レスポンスが，0.5% 早く処理を完了している．ログの記録をしていない **NoLog** と比較しても，2% の速度低下に留まる．スループットについては，1.1% 低下している．**NoLog** と比較しても，0.5% 程度の速度向上を果たしている．AP/DB アクセス関連付け処理を行った **APDBKey0** が関連付けを行わない場合と比較して，速度向上を見せることもあったのは，測定誤差のためと思われる．次に，**APDBKey1** については，**Logging** に対して，レスポンスで 2%，スループットで，4% のオーバヘッドが見られた．

提案する **APDBKey0**,**APDBKey1** の各手法では，レスポンス，スループット共に速度低下 (一部速度向上) しているものの，5%以下と小さいため，性能面において，本手法は実用に耐え得るといえる．

5. ま と め

内部統制支援を目的とした Web/DB/AP サーバに跨るログを統合・検査の実現を意図し，本論文では，AP/DB ログ関連付け機能について紹介した．PostgreSQL にプロトタイプを実装し，性能評価では，ログ関連付けを実施しない場合と比べても，実行時オーバヘッドが 5% 以下に抑えられた．AP/DB の関連付け手法は，DBMS 全般に適用可能であり，広く利用できると考える．

謝辞 本研究を進めるにあたり，NTT 情報流通プラットフォーム研究所の間形文彦研究主任より貴重なアドバイスをいただきました．心より感謝申し上げます．

文 献

[1] トレッドウェイ委員会組織委員会: 内部統制の統合的枠組み (2006).
[2] 金融庁企業会計審議会内部統制部会: 財務報告に係る内部統制の評価及び監査に関する実施基準 - 公開草案 - (2006).
[3] 間形文彦, 濱田貴広, 岡崎聖人, 塩野入理, 金井敦: DBMS における業務処理統制機能の要件と課題に関する考察, 第 35 回電子化知的財産・社会基盤研究会 (2007).
[4] データベース・セキュリティ・コンソーシアム (DBSC) http://www.db-security.org/: データベースセキュリティガイドライン (1.0 版) (2006).
[5] *SQL Guard*. http://www.air.co.jp/products/sql_guard/sql.html.
[6] 情報漏洩監視システム PISO. http://www.insight-tec.com/products/service_piso.html.
[7] *IPLocks*. http://www.iplocks.co.jp/products/.
[8] *AppScan*. http://www.techmatrix.co.jp/products/security/watchfire/.
[9] *WebProbe*. http://www.softek.co.jp/Sec/WebProbe/.
[10] PL/pgSQL - SQL 手続き言語. http://www.postgresql.jp/document/current/html/plpgsql.html.