特集論文

OSSをベースとしたデータベースパッケージ“H@DB”

鈴木和行*
Kazuyuki Suzuki
大山紗貴子*
Sakiko Oyama
原田雅史*
Masafumi Harada

Database Package "H@DB" Based on Open Source Software

要　旨

近年，高可用性や拡張性が求められる大規模な社会インフラシステムでも開発費用の低減，開発期間の短縮や運用を含めた維持費用の低減が要求されている。三菱電機は，これまで多くの社会インフラシステムの構築に携わってきているが，今後，社会インフラシステムビジネスを更に拡大していくためには，IT技術の進展に合わせてタイムリーかつ市場競争力のある製品開発が必要不可欠である。

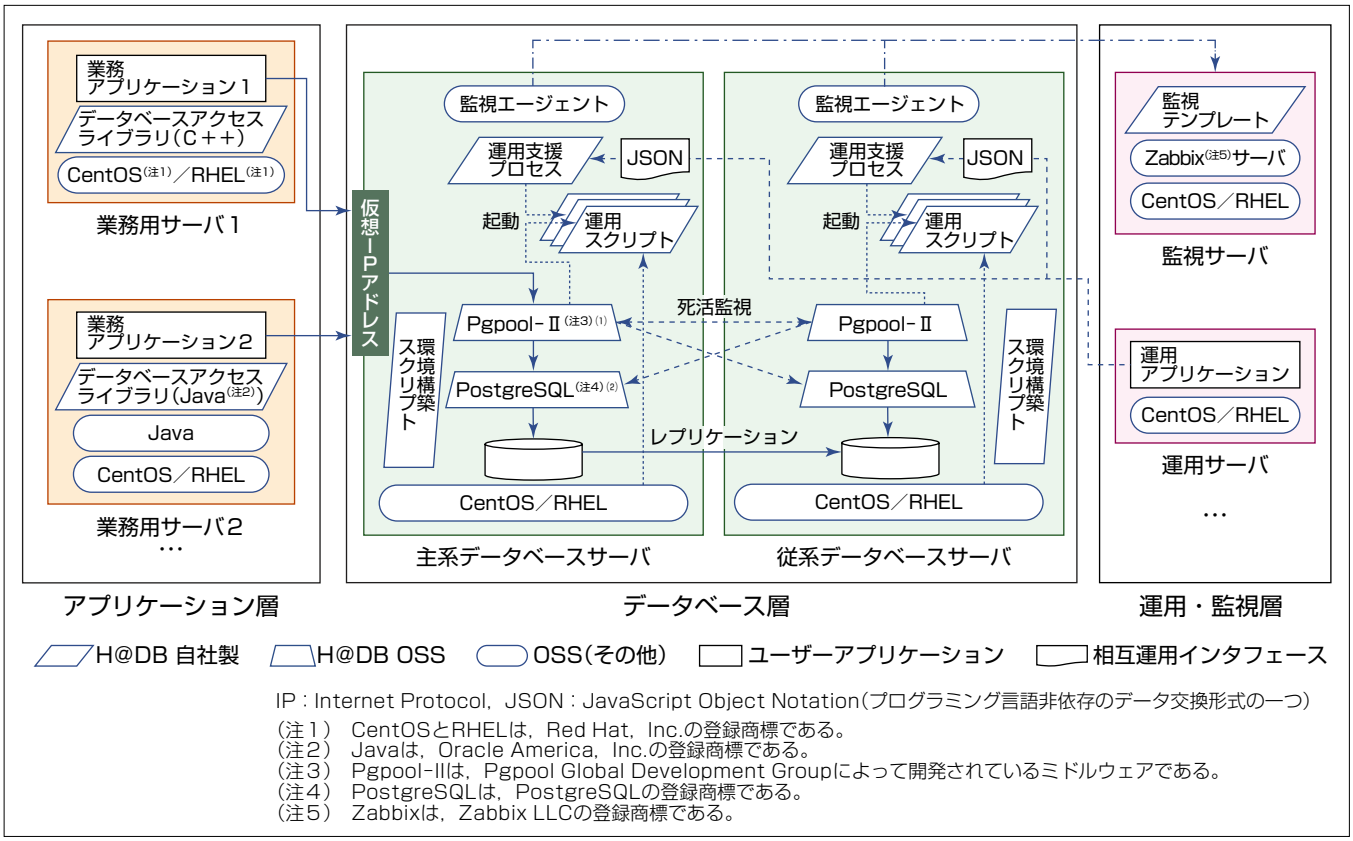
当社は，顧客からの要望が強いITシステムの総保有コストの低減を実現するため，OSS(Open Source Software)製品群と自社製ソフトウェアを組み合わせた次の特長を持つデータベースパッケージ“H@DB”を開発した。

(1) 商用データベースと比較してライセンス費用を削減するためにOSSデータベースを採用

(2) OSSデータベースに不足している環境構築の自動化，高可用性及び運用作業の平易化を実現する各種スクリプト，運用支援プロセス，監視機能用テンプレートを提供

(3) データベースの変更に伴う既存アプリケーションの移植性を向上させるデータベースアクセスライブラリの提供

現在，H@DBは，某社会インフラシステムでも採用されており，今後は，この開発で蓄積した各種OSS製品に関する技術や知見を利用した更なる製品開発の推進やOSS製品のバージョンアップへの対応を行い，幅広い分野への展開を図っていく予定である。



IP：Internet Protocol，JSON：JavaScript Object Notation(プログラミング言語非依存のデータ交換形式の一つ)

(注1) CentOSとRHELは，Red Hat，Inc.の登録商標である。
(注2) Javaは，Oracle America，Inc.の登録商標である。
(注3) Pgpool-IIは，Pgpool Global Development Groupによって開発されているミドルウェアである。
(注4) PostgreSQLは，PostgreSQLの登録商標である。
(注5) Zabbixは，Zabbix LLCの登録商標である。

データベースパッケージ“H@DB”を使用したシステム構成例

データベースパッケージH@DBは，ITシステムのアプリケーション層，データベース層及び運用・監視層をカバーしており，OSS製品によるライセンス費用の削減，各スクリプトによる環境構築の自動化及び運用作業の平易化，同期又は非同期レプリケーションによるデータベースの複製と主系-従系の自動系切換え，データベースアクセスライブラリによる既存アプリケーションの移植性の向上を実現している。

*インフォメーションシステム統括事業部

1. まえがき

当社は，航空，鉄道，自治体，電力などの社会インフラシステム構築に長年携わってきている。近年，ITシステムの中核になる商用データベース製品のライセンス費用に対するコスト削減のニーズが高まってきている。さらに短納期開発や平易なシステム運用作業も求められている。

当社は，これらの課題を解決して市場競争力を強化するため，当社のITプラットフォーム“DIAPLANET”の一構成要素になる，最新のOSS製品と自社製ソフトウェアを組み合わせたデータベースパッケージH@DBを開発した。

本稿では，H@DBの開発背景，特長及びデータベースアクセスライブラリ開発時に発生した技術課題とその解決策について述べる。

2. H@DBの開発背景

まず，特定プロジェクトを対象にして商用データベースをOSSデータベースに置き換え可能かどうか判断するため，**表1**のとおり分析を行った。

商用製品は，全方位サポートしている。一方，OSS製品は，基本項目以外は概して不十分である。そこで，不足項目を補う方式を検討した結果，それらを自社製にしてOSSデータベースに付加することで，商用データベースに代替可能な付加価値を持ったパッケージ製品にすることが可能と判断した。**3章**以降では，H@DBの基本的なアーキテクチャと洗い出した課題に対する取組みを述べる。

3. データベースパッケージH@DB

3.1 H@DBのアーキテクチャ

OSSデータベース製品は複数存在するが，国内での普及度合い，製品としての中立性，OSSサポートサービスなどを勘案してPostgreSQLを選定した。

表1．商用及びOSSデータベースとH@DBの比較

評価項目	商用製品	OSS製品	H@DB
機能	○	○	○
性能	○	○	○
可用性	○	△	○
運用性	○	×	○
移行性	○	×	○
導入容易性	○	△	○
保守性	○	×	○

○あり　△やや不十分　×不十分

次に，PostgreSQLで高可用性を実現するためにはOSSのプロキシミドルウェアであるPgpool-Ⅱを前段に配備し，PostgreSQLのストリーミングレプリケーションを使用する方法や，各種HA(High Availability)ミドルウェアを組み合わせる方法等がある。今回は組み合わせるOSSの種類を極力減らすことやクラウド上での展開も想定し，Pgpool-Ⅱを活用した構成を選択した。

さらに，OSS製品自体には実装されておらず付加価値になる環境構築自動化，運用作業の平易化，既存アプリケーションの移植性向上に資する各ソフトウェアを自社製にしてパッケージ化した。

3.2 運用スクリプトによる運用作業の平易化

システム運用作業は，通常運用作業(定期，不定期)と障害時運用作業から成る。H@DBは，両者の作業平易化をサポートしている。具体的には，OS，PostgreSQL，それに組み合わせるPgpool-Ⅱ及び運用支援プロセスに対して，次のカテゴリーに分類できる各種運用スクリプトを提供している。

(1) 障害

PostgreSQLのフェールオーバーや同期モードの切換えなどH@DBに障害が発生した場合，Pgpool-IIから呼び出され，障害復旧作業を自動化する。

(2) 通常運用

PostgreSQLのバックアップ・リストア(論理・物理)，PostgreSQLのデータベースクラスタの初期化，同期状態の確認など，代表的な通常運用局面で運用者によって呼び出され，運用作業を軽減する。

(3) 障害時運用

系切換え，H@DBの状態確認・停止，従系切離しなど，運用支援プロセスから呼び出され，運用作業を軽減する。

(4) 起動・停止

H@DB起動，運用支援プロセスの起動・停止，Pgpool-IIの起動・停止など，H@DBが稼働している物理サーバの起動・停止時に呼び出され，運用作業の一部自動化を行う。

運用作業平易化の一例として，障害からの復旧では，該当する物理サーバの電源をオンにするだけでデータベースクラスタに自動参加が可能である。PostgreSQLとPgpool-IIを組み合わせた一般的な冗長システムでは，構成要素に障害が発生した場合，サーバをまたがる通信ラインを構成する(**図1**(a))。そのため運用者は，障害から復旧させるために双方のサーバで作業を行う必要がある。一方，H@DBでは，常に同一サーバでPostgreSQLとPgpool-IIが稼働する構成を保つようにしており，操作は，切り離された単一のサーバに対して電源オンだけで完了する(**図1**(b))。

主従の系切換え作業や従系の切離し作業等に失敗した場

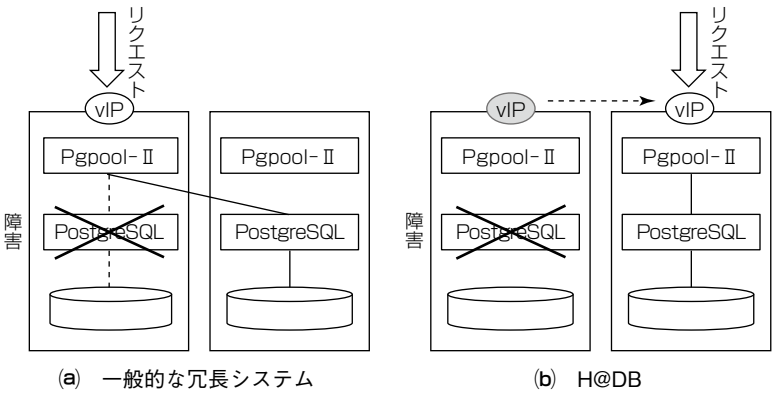


図１．障害発生時の物理サーバ構成の違い

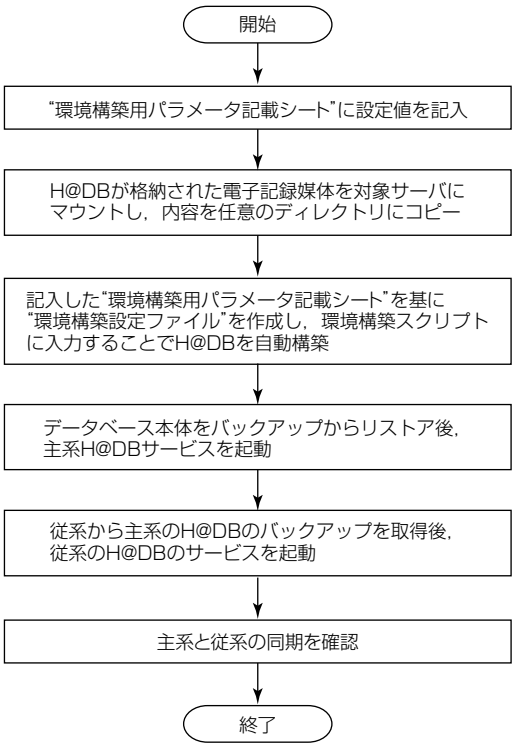


図２．冗長構成H@DBの構築手順

合，切戻し処理を行うことによって元の状態に復帰させる機能を実装している。さらに，運用スクリプトを構成する各処理ステップに対してタイムアウト処理を付加することで切換え時間，切離し時間の保証を実現している。

3.3 環境構築の自動化

商用製品には，ウィザード形式のインストーラが同梱（どうこん）されており，容易に環境構築が行えるようになっている。一方，OSS製品にはインストーラが提供されておらず，各操作を一手順ごとに実施しなければならない。

H@DBでは効率的かつ容易に環境が構築できるよう，次の特長を持つ環境構築スクリプト（インストーラ）を提供している。

⑴ 環境構築経験が少ない開発者も環境構築用パラメータ記載シートに記入することで環境が構築可能である。
⑵ システム特性に応じて単一構成（Pgpool-ⅡとPostgreSQLが各１台），冗長構成（Pgpool-ⅡとPostgreSQLが各２台）のどちらの構成も構築可能である。
⑶ 環境構築だけでなく環境の削除にも対応している。
⑷ H@DB用のPostgreSQL等の設定ファイル（パラメータ値）をあらかじめ調整済みである。
⑸ 例えば，冗長構成のH@DBは，**図２**に示すとおり６ステップだけで構築可能である。

3.4 運用・監視について

データベースは，システムをつかさどる重要な部位の一つであるため運用・監視は必須になる。H@DBの運用と監視の特長について述べる。運用については，運用支援プロセスを自社製にして対処した。運用支援プロセスは，H@DBを使用したシステムの運用作業を運用アプリケーションからの指示に基づいて遂行する。そのため運用支援プロセスは，運用アプリケーションと制約なしに連携し，運用スクリプトを呼び出して状態取得，系切換え，停止などを行える必要がある。H@DBでは，運用アプリケーションと運用支援プロセス間のインタフェースとして，プログラミング言語に依存しないJSON形式のファイルを送受信する方式を採用した。

監視については，近年，広く使用されているOSSのZabbixに対応しており，H@DBに対応した監視テンプレートを提供している。運用者は，監視テンプレートをZabbixサーバにインポートし，H@DBが稼働しているサーバを登録することで監視が可能になる。監視項目を**表２**に示す。

3.5 既存アプリケーションの移植性向上

データベースを変更する場合，データベース自体の移行のほかにアプリケーション（ソースコードとSQL（Structured Query Language）文）を改修する必要があり，改修量に応じた作業が発生する。H@DBでは，商用データベースのライブラリと互換性を持つ**図３**に示す２種類のデータベースアクセスライブラリを開発・提供することで，移行時のアプリケーション改修量を削減できた。

表２．監視項目

カテゴリー	監視項目
リソース監視	CPU／メモリ／ディスク使用率
プロセス監視	Pgpool-Ⅱ，PostgreSQL，運用支援プロセス
ポート監視	同上
ログ監視	同上，運用スクリプト

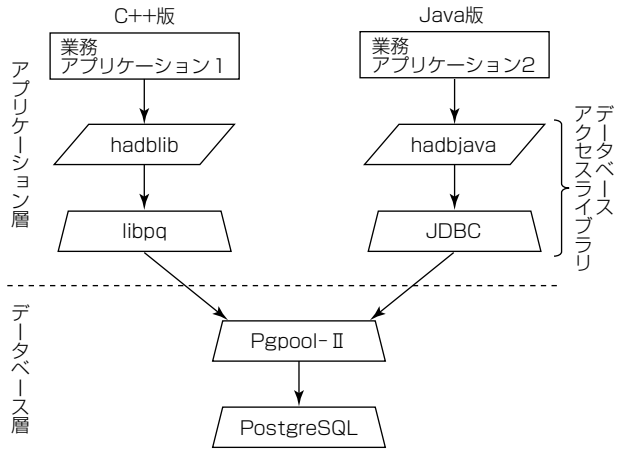


図3．データベースアクセスライブラリの最終形態

3.5.1　データベースアクセスライブラリ(C++，Java)の開発方針

H@DBでは，一般的に使用されていることが多い開発言語としてC++，Javaを想定している。しかし，PostgreSQLから提供されているC++版のデータベースアクセスライブラリは，現在では最新版が提供されていないため，最新版の提供が継続しているC言語版ライブラリ(libpq)を基に自社製のC++版ライブラリ(hadblib)を開発した。インタフェース設計の際，ユーザーアプリケーションで使用している商用データベースのライブラリの互換API(Application Programming Interface)を提供することで，API回りの修正が不要になりソースコードの修正量を大きく減らすことができた。

3.5.2　開発中に発生した課題と解決策

Java版ライブラリは，JDBC(Java DataBase Connectivity)に準拠したものがPostgreSQLから提供されているので採用した。これらのデータベースアクセスライブラリの試験を進めていく中で幾つかの課題が発生した。

(1)　タイムアウトが発生しない事象への対策

一つ目の課題は，タイムアウトが発生しない問題である。データベースの従系が主系に切り換わったタイミングで，そのときアクセスしていたアプリケーションが停止状態になる事象が発生した。

この事象は，C++版ライブラリ開発で基にしたC言語版ライブラリでは，タイムアウト時間がOSの設定値に依存する実装になっていたことが原因であった。そこで，OSの設定値に依存せずにタイムアウト時間をユーザーが設定可能にするよう，シグナルハンドラを利用したタイムアウト処理を新たにH@DBに実装することで対処した。

(2)　結果が正しく取得できない事象への対策

もう一つの課題が，アプリケーション側でトランザクションの成否を正しく取得できない事象の発生である。

データベースはトランザクションに対する不可分性を保証しているため，結果は成否のどちらかになる。しかし，データベースからの応答を受け取る前に通信エラーが発生した場合，アプリケーション側で正しいトランザクション結果を取得できない。例えば，データベース内部では成功したにもかかわらず，アプリケーション側では失敗と判断してしまう事象である。

トランザクション結果を正しく取得するためには，あらかじめトランザクション番号を取得しておき，エラーが発生した場合，トランザクション番号をキーにして結果の成否を再取得する手法がある。この手法は，アプリケーション側で対処するのが一般的であるが，H@DBではアプリケーション側への影響を少なくするため，H@DB側にその機能を実装する方式(hadblib及びhadbjava)とし，大きな性能劣化も引き起こすことなく対処できた。

このようにアプリケーション開発で利用するライブラリに関しても多くの検証と，発生した課題に対する対処を行い製品としての完成度を高めることができた。

4. むすび

今回開発したH@DBは，当社が担当している社会インフラシステムで商用データベースの代替品として使用され，当初の目的である商用データベースのライセンス費用削減(当初の1/3～1/10)と運用作業の平易化に貢献できた。

また，PostgreSQLに代表されるOSSデータベースは，従来は商用データベースだけが持っていたテーブルのパーティショニング，パラレルクエリ及びトランザクション番号の64bit化などをサポートしてきており，大規模で高信頼が要求されるシステムでも利用可能な状況にあることが確認できた。

H@DBの開発を行うことで得られたOSS製品に関する技術や知見は，今後の製品開発に有効に活用できるものと考えている。当社は，市場のニーズに応えられる製品開発を今後も継続していく。

近年，OSS製品の改版周期も短くなり，機能拡張や性能向上が頻繁に行われている。それらに追従するためには，H@DB自体も短周期での改版作業が必要であり，安定的に高品質なH@DBを提供していくために保守体制も整備していく予定である。

参考文献

(1)　The Pgpool Global Development Group：Pgpool-Ⅱ 4.0.5文書
https://www.pgpool.net/docs/40/ja/html/

(2)　日本PostgreSQLユーザ会：PostgreSQL 11.3付属ドキュメント(2019)
https://www.postgresql.jp/document/pg113doc/index.html