XML データベース
NeoCore XMS ベンチマークレポート

スキーマレスデータの検索・格納処理における、RDB の XML 機能に対する優位点

2010 年 12 月

株式会社サイバーテック

目次

1. はじめに

本文書はベンチマークプロジェクト(本プロジェクト)の実施内容を記述したものである。
本プロジェクトでは XML データベース「NeoCoreXMS(以下、NeoCore)」と「A 社製品(RDB)の XML 機能」について、CRUD 処理(挿入、検索、更新、削除)のパフォーマンスについて比較/検証を実施した結果について記述したものである。

2. NeoCoreXMS について

2 – 1. NeoCoreXMS の用途

XML データベース「NeoCoreXMS」は、構造が一定ではない XML データの格納に最適な"やわらかい"データベース・エンジンです。国内出荷ライセンス数は 500 ライセンスを超え、国内 No.1 シェア（富士キメラ総研調べ）の XML DB/XML データベースです。

「NeoCoreXMS」は、XML をハンドリングすることに特化、RDB では実現不可能な水準のパフォーマンスを発揮する事ができる、柔軟性と拡張性に優れた XML データベース製品です。製造業の製品データやドキュメントに付随するメタデータなど、多様で変化しやすいデータをスキーマレスの XML としてそのままデータベースに格納する事ができるため、システム運用中のデータベースの変更コストを最小限に抑える事が可能です。

「NeoCoreXMS」の特長は、データ量やデータ構造に依らない「超高速の検索性能」（速い）、すべてのタグに対して自動的にインデックスをはる「フルオートインデックス機能」（カンタン）、XML 形式のデータを柔軟に吸収できる「スキーマレス機能」（やわらかい）です。これらの特長は、ビジネスサイドからの要求によって発生する仕様変更に対して、システム側で即座に対応する事ができます。

「NeoCoreXMS」は、企業情報システムの中で仕様変更が頻繁に発生する情報系システムの開発・運用フェーズにかかるシステムエンジニアコストと時間を大幅に削減する事を可能にしました。また、「NeoCoreXMS」は、アプリケーション開発者向けの情報やノウハウが多数公開されているだけでなく、初級者向けのトレーニングメニューを取り揃えているため、技術者の入門用またはプロトタイプ構築用に最適な XML データベースです。

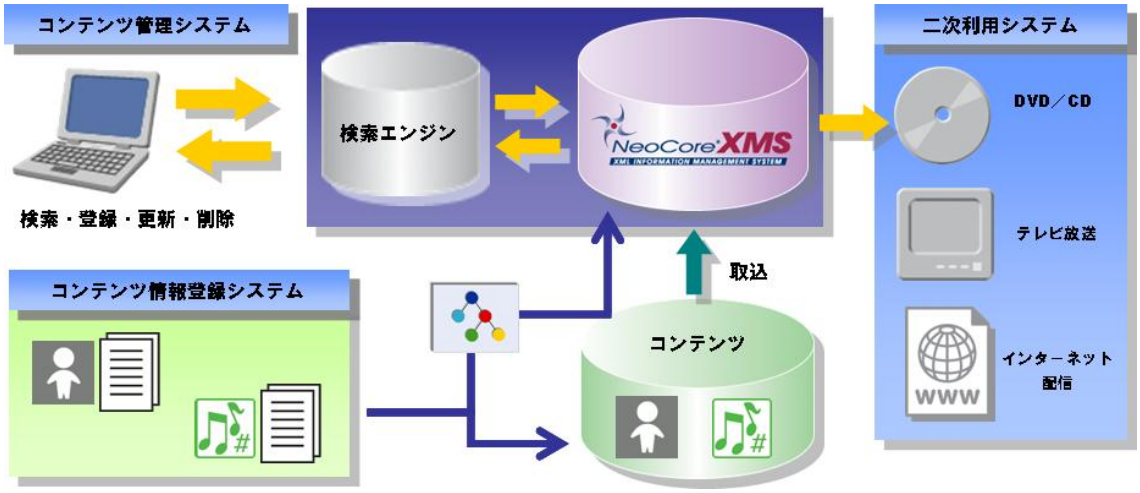
■「NeoCoreXMS」の主な用途

＜ドキュメント管理に強い！＞

「NeoCoreXMS」は、XMLドキュメントを「そのまま」格納できるため、マニュアル・約款・規定集や教材などのドキュメント管理データベースに最適で、多くの導入実績があります。「内部統制による業務の明確化」や、「紙の電子化による印刷･配布･保管コストの削減」、「DTP などの制作現場の業務改善」など、ドキュメント管理に関する課題を解決します。

＜メタ情報の一元管理・検索に強い！＞

デジタルコンテンツの社内管理とユーザーへの提供サービス向上の実現での利用ケースがこれにあたります。デジタルコンテンツの音楽、写真に作曲者、著作権者、収録日などのメタ情報を付加することで、社内用途ではコンテンツ管理として利用でき、デジタルコンテンツを再利用したいユーザーにとっては、付加されたメタ情報が検索キーワードとなり、効率良く目的のコンテンツにたどり着くことができるのです。Web カタログや製造業の製品情報管理、電子カルテなどその応用範囲は多岐にわたります。

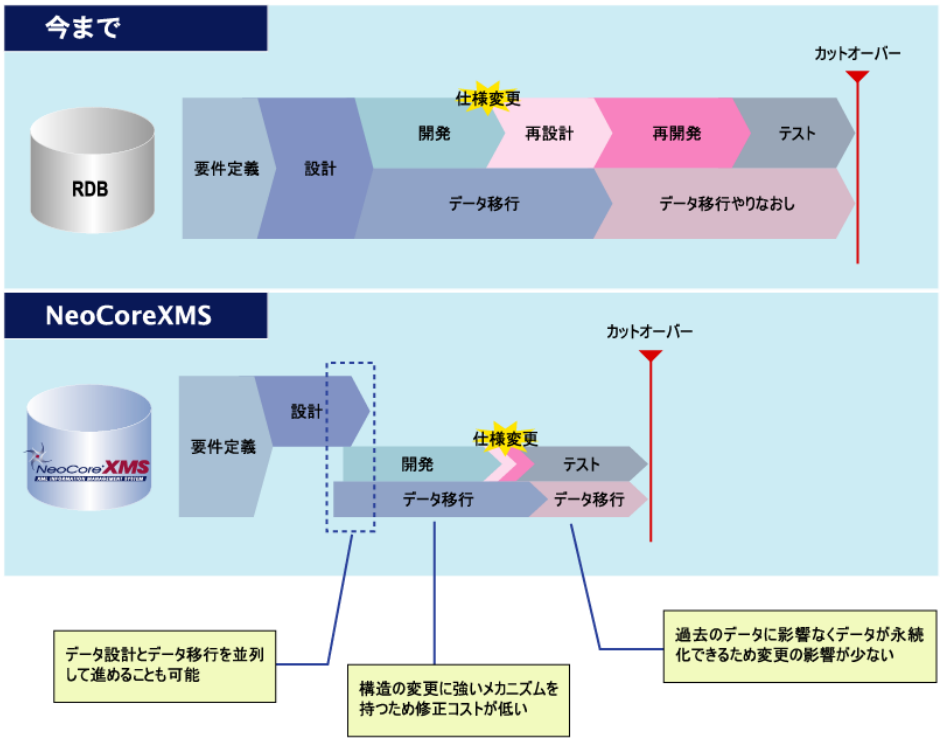


■XML を活用した新しい開発スタイル

＜アプリケーションの設計変更コストを削減する「NeoCoreXMS」＞

ビジネス環境がスピード化・多様化する中で、情報系システムの開発・運用フェーズで「仕様変更が頻繁に発生する」事は、もはや避けられない状況です。ビジネスが変われば当然管理すべきデータも変わり、さらにデータを支えるアプリケーションにも影響が及びます。従来の RDB は、厳密なスキーマ情報を前提とするデータベースであるため、「仕様変更＝スキーマ変更」となり、さらに適正なパフォーマンスを確保するために「インデックス再設計」が必要でした。

「NeoCoreXMS」は、厳密なスキーマ設計を必要としないため、ビジネスサイドからの要求によって発生するスキーマ設計とインデックス再設計にかかるシステムエンジニアコストと時間を大幅に削減する事を可能にしました。



2 – 2. NeoCoreXMS の特長

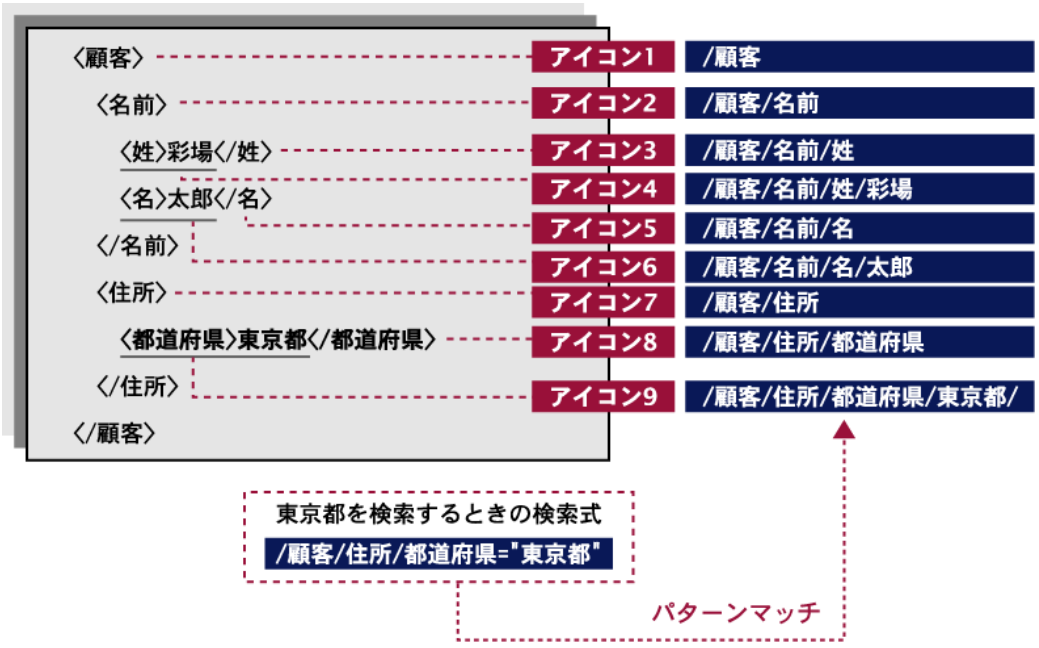
「NeoCoreXMS」は、まったく新しいタイプの XML データベースエンジンです。
スキーマ定義は一切不要、XML 形式のデータであれば、すべて柔軟に吸収。フルオートインデックス機能により、すべてのタグに対して自動的にインデックスを作成します。システム開発者の悩みどころであった、インデックス設計を不要にし、システム開発の効率を飛躍的に向上させることができます。

XML の超高速検索を実現する、DPP（Digitatl Pattern Processing）を搭載。元データを見ることなく、フラットな構造のアイコンをパターンマッチさせる事でデータ量やデータ構造の複雑さに依存しない安定した検索パフォーマンスを実現します。XML 形式のデータをスキーマレスで格納、XML の柔軟性を活かしたデータ管理が可能です。

■XML 超高速検索機能

リレーショナルデータベース（RDB）のテーブル構造に XML データをマッピングしたり、XML データ型の領域に格納した場合に、著しく検索性能が低下してしまう場合があります。NeoCoreXMS は、独自の特許技術である DPP（Digital Pattern Processing）により、XML のデータ量や XML の階層構造の深さに依存しない安定した検索が可能です。

NeoCoreXMS は、XML データを格納する際、Parsing を通過した XML データを、Flattener と呼ばれるモジュールにより、パスとデータに分解します。DPP は、この分解した 1 つ 1 つのパスとデータを、64bit の固定長のデータに変換し、全てのタグに対してユニークな「アイコン」として生成します。
NeoCoreXMS では、データベースに格納された XML データを検索する際に、元データを見ることなく、フラットな構造の「アイコン」をパターンマッチさせる方式を採用。この「アイコン」が XML データを検索する際のインデックスとして機能し、実データはこの「アイコン」が指し示す場所に格納されます。



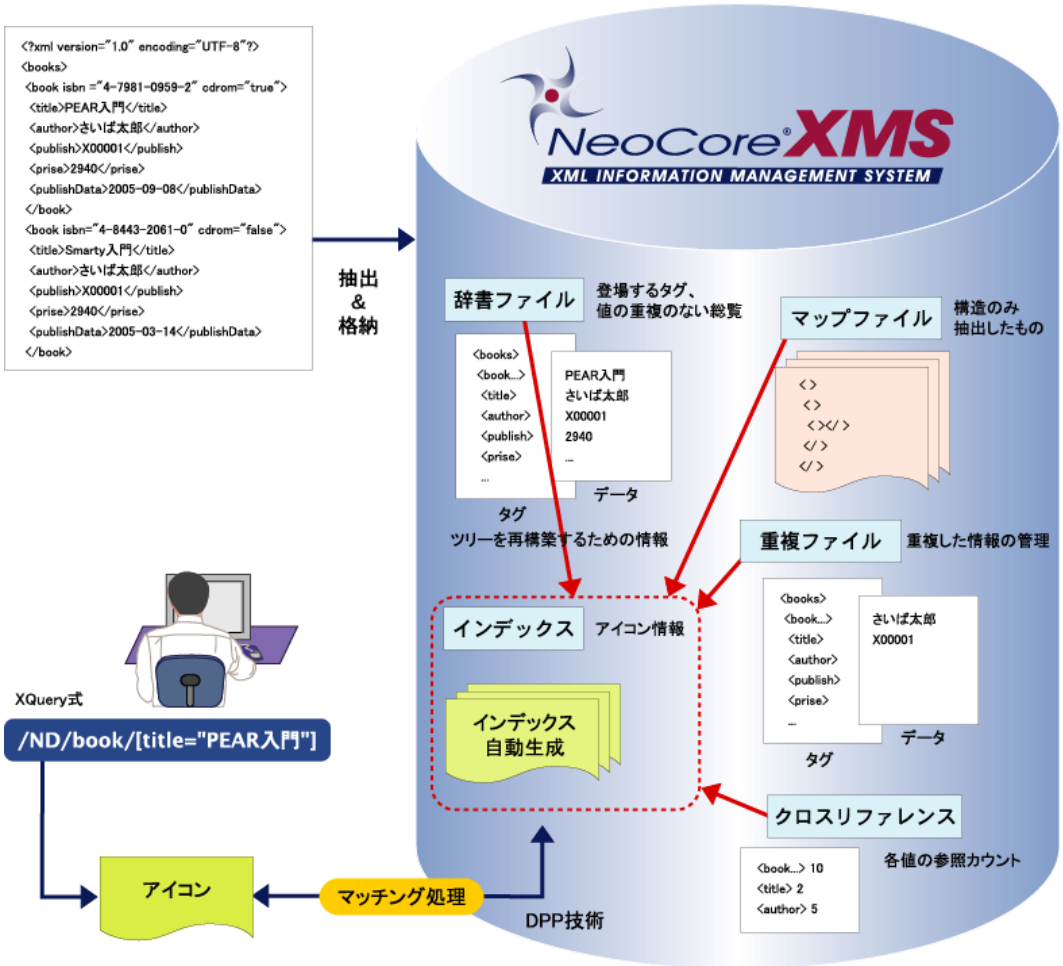
■フルオートインデックス機能

NeoCoreXMS は、XML データをデータベースに格納する際、DPP（Digital Pattern Processing）により、すべてのタグに対して自動的にインデックスを生成します。

フルオートインデックス機能は、リレーショナルデータベース（RDB）で必要とされるインデックス設計や生成のための作業を不要とします。つまり、開発者はどの項目にインデックスを設定すれば良いかを考える必要はなく、インデックスを上手に使用するような検索方法を考えるだけで済むため、システム開発時の時間とコストを飛躍的に向上する事が可能です。NeoCoreXMS は、格納時や更新時に DPP により分解された XML の構造を、独自の方式でバイナリデータとして種類別に分類して格納します。

NeoCoreXMS の内部は、下記の複数ファイルで構成・管理されます。

- タグ部及びデータ部の実体を格納する「辞書ファイル」
- タグ部及びデータ部のインデックス情報を格納する「インデックスファイル」
- タグ部及びデータ部の重複インデックス情報を格納する「重複ファイル」
- 辞書ファイルとインデックスファイルを相互参照する「クロスリファレンス」
- XML データの物理的な構造情報を格納する「マップファイル」
- データベースファイルのロケーション情報や、格納されている文書数などの管理者情報を格納する「Admin ファイル」

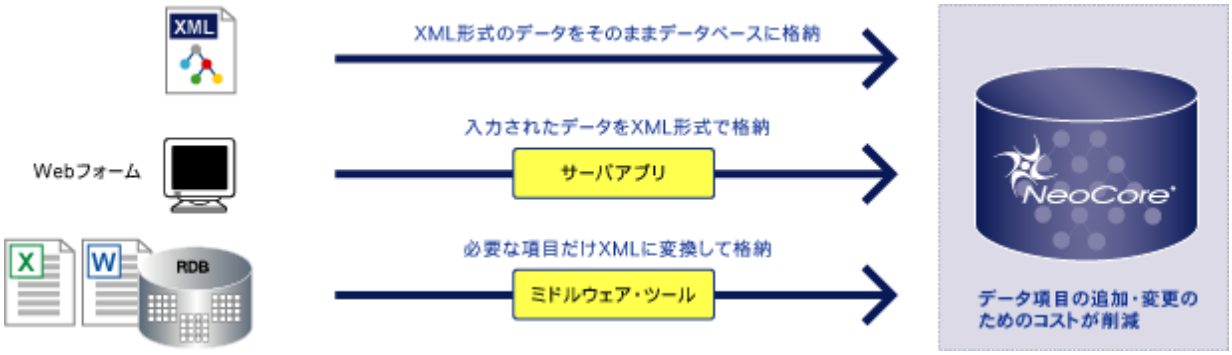


■スキーマ定義不要

NeoCoreXMS は、XML 形式のデータを格納する際、スキーマを定義する必要がありません。格納、更新等の操作の際に入力する XML については、妥当性検証（Validation チェック）を行わず、整形式であるかどうかのチェック（Parsing）のみを行います。従って、DTD や XMLSchema とは関係なく、あらゆる XML 形式のデータを格納し、またエレメントの追加や削除を自由に行うことができます。

リレーショナルデータベース（RDB）は、格納するデータ全てのデータについて、厳密なスキーマ定義を必要とするため、仕様変更のたびにスキーマを再定義する必要がありました。これに対して、NeoCoreXMS を用いたデータベース設計では、ノードや属性を Tree 型に配置した構造を設計し、パフォーマンスを考慮した Tree 構造の変更を行います。リレーショナルデータベース（RDB）の場合はインプリメントの前に構造設計が完結している必要がありますが、NeoCoreXMS の場合は、基礎が決まっていれば、ある程度の変更はアプリケーションで吸収できてしまいます。

これにより、ビジネスサイドからの要求によって発生するスキーマ設計とインデックス再設計にかかるシステムエンジニアコストと時間を大幅に削減する事が可能となり、システム開発の初期の段階でデータ構造を厳密に定義できない場合や、システム運用フェーズでのデータ項目の追加変更の際には、大きなメリットとなります。



2 – 3. 機能詳細

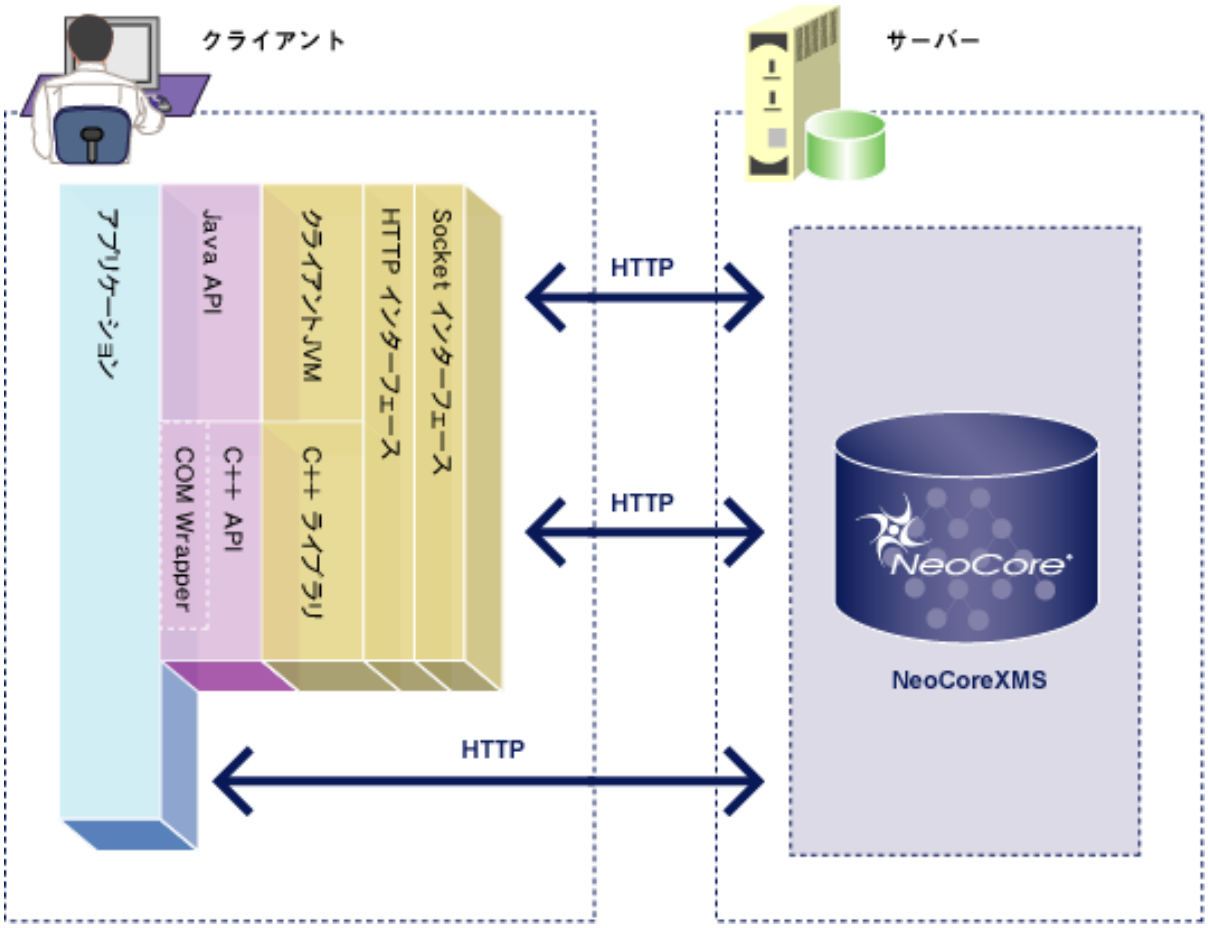
■HTTP によるアプリケーション通信（プロセス・アーキテクチャ）

NeoCoreXMS のサーバープロセスは、各コンポーネントが実行を制御します。各コンポーネント内部ではマルチスレッドで処理され、処理効率を向上させています。クライアントアプリケーションは JavaAPI、C++API、または C++API をラップした COM ラッパーから、HTTP クライアントのソケットを実装した各ライブラリを介してサーバープロセスへ接続します。Perl や PHP などのスクリプト言語からは HTTP を介して接続します。

アプリケーション開発者は、NeoCoreXMS を利用する場合、使用する言語を Java または C++から選択できます。これらの言語からは、 NeoCoreXMS より提供される API ライブラリを使用する形で開発が可能です。さらに、HTTP I/F を用いることで、C#、PHP、Perl、Ruby、JavaScript や Ajax、Flash などの RIA

をはじめとする、ソケット通信が可能なあらゆる言語を使ったアプリケーション開発が可能です。

API ライブラリおよび HTTP I/F いずれの手段を使っても NeoCoreXMS の持つ機能全体を使用できるため、アプリケーション開発者は最も慣れている言語、または開発タスクに最も適した言語を選択できます。また、NeoCoreXMS は、HTTP 通信のオーバーヘッド自体が非常に小さいため、パフォーマンスへの影響が出にくいのも特長です。



■XPath／XQuery によるデータアクセス

NeoCoreXMS へのデータ格納および格納されたデータに対するアクセスを行う場合には、SQL ではなくXPath／XQuery を使います。これを使うことで、複数存在する要素のうちアクセスしたい任意の要素を指定して検索結果を取得することができます。

また、引数として XPath 式を与えることで、XML データの一部を更新したり削除・追加をすることができます。XPath はパス形式の記述言語なので、複数の文書にまたがって存在する同名の要素に対し、一回のコマンドで横断的にアクセスすることもできます。XPath/XQuery は共に W3C の正式勧告を受けた世界標準規格なので、長く安心してお使い頂く事ができます。

■トランザクション・データロック機能

＜(1)トランザクション機能＞

NeoCoreXMS は、リレーショナルデータベース（RDB）同等の ACID レベルのトランザクションをサポートしています。全てのトランザクションは、トランザクションログに記録され、電源障害などの予測できない障害が発生した場合におけるデータベースの復元・復旧を可能とします。明示的なトランザクションの開始、コミットおよびロールバックも可能で、分離レベルの設定も可能です。

(2)データロック機能

一般的なリレーショナルデータベース（RDB）と同様に、NeoCoreXMS においてもデータロックを使用したデータへの同時アクセス制御が可能です。情報の更新時には、更新がコミットされるまで、その情報はロックされ、発行またはコミットされるまで、ロックされている情報は誰も変更できません。これにより、システムのデータ整合性が保証されます。

■ブラウザ及びコマンドラインによる各種ユーティリティ

NeoCoreXMS では、インスタンスの起動・停止や、データベース管理のために、各種ユーティリティプログラムを提供しています。

＜(a)NeoXMLUtils＞

NeoXMLUtils は、データベースの管理を行うユーティリティプログラムで、以下の機能を備えています。

- データベース生成・削除関連
  データベース作成・空のデータベース作成・データベースクリア・データベース削除が可能です。
- データベースの保存・復元関連
  データベースのバックアップ・リストア・インポート・エクスポート等が可能です。
- インデックス再構築・ファイル移動関連
  インデックスの再構築や、データベースファイルの移動が可能です。

＜(b)管理コンソール＞

管理コンソールは、データベース管理者のための、ブラウザベースの管理ツールです。

- サーバ管理

  NeoCoreXMS の起動、停止、バージョンの確認が可能です。

- データベースの状態表示・サーバログとアクセスログの表示

  NeoCoreXMS の内部で管理されている各ファイル(マップファイル等)の物理サイズや使用率等を確認することが可能です。

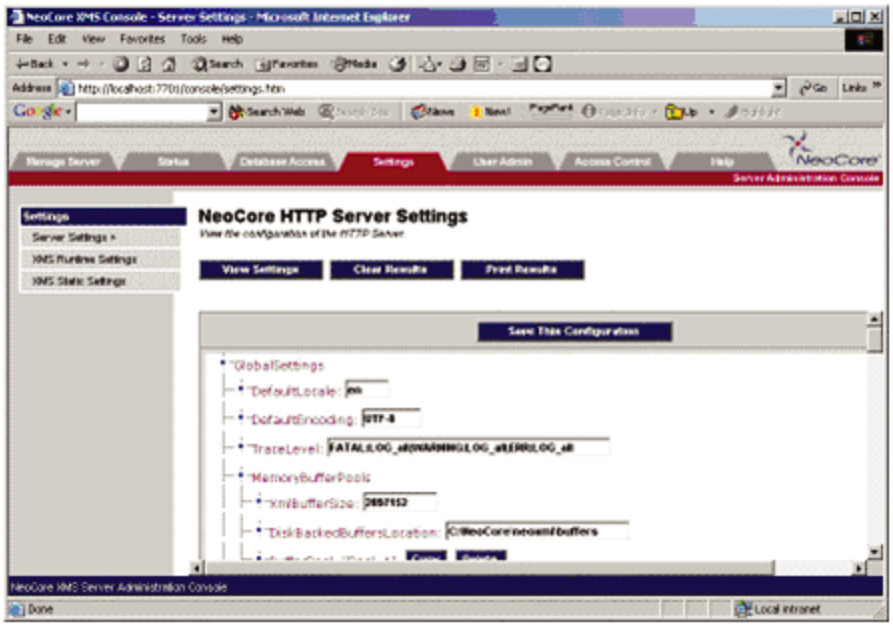
- XML 文書の操作とデータベースアクセス

  (Copy/Query/Insert/Modify/Delete)

  データベースへ XML データの登録、XPath／XQuery を使ったデータの参照などが可能です。

- サーバコンポーネントの設定表示・設定変更・チューニング

  メモリバッファサイズ、デフォルトエンコード値、JAVA 仮想マシンのヒープサイズ、HTTP アクセスのポート番号、セッションタイムアウト時間、メモリの自動拡張設定等、設定の確認と変更をきめ細かく行うことが可能です。



■キーファイルによるライセンス管理

NeoCoreXMS は、ライセンスキーファイル(license.xml)でライセンス情報の管理を行います。購入前に無償で評価頂くための「30 日評価版ライセンス」からデモやプロトタイプ用途に利用頂くための「Developer Edition」、さらには、それぞれのご利用シーンに応じた形でのご選択が可能な「Workgroup Edition」「Limited Edition」「Standard Edition」へのステップアップは、このライセンスキーファイルの入れ替えのみで可能です。

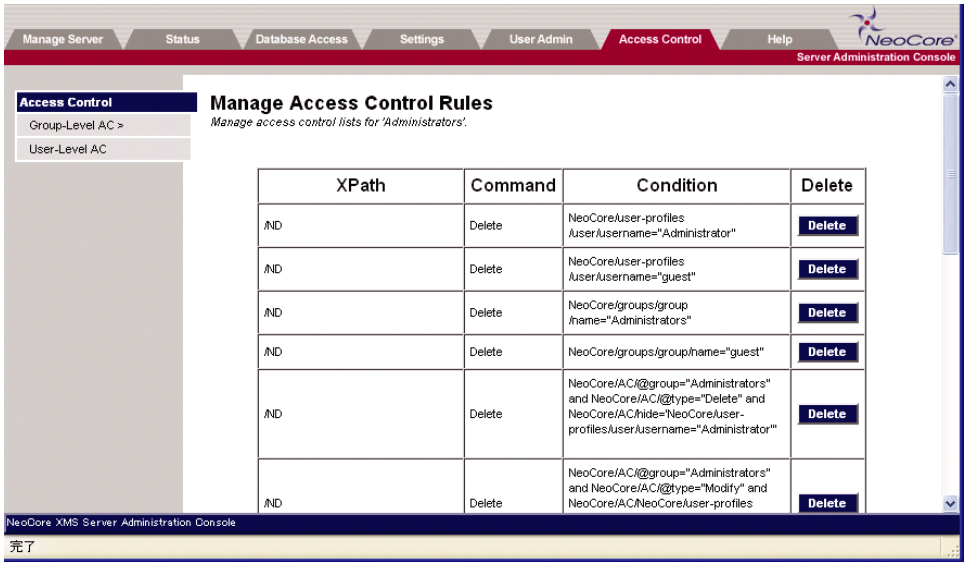
さらにこの仕組みを活用することで、容易にパッケージソフトへの組み込みが可能となります。
組み込むパッケージソフトの利用形態や運用形態に応じて、利用期間の制限、データベース容量の上限値の設定、リモートサーバからのアクセス制御などをライセンスキーファイルのみで設定・管理するこ

とができます。

■セキュリティ機能（アクセスコントロール）

NeoCoreXMS は、データベースアクセスを制御するためのセキュリティ機能を標準で提供しています。このアクセスコントロールを使用することで、XML データへのアクセスをノードレベルで自由自在に制御する事が可能となります。

アクセスコントロールの設定には、ブラウザベースの管理コンソールを使い、制御ルールを作成します。すべてのノードを対象とする単純なルールから、該当文書内の情報に基づく条件に合致するノードだけを対象とするといった複雑なルールにも対応する事が可能です。



■XML メタデータの格納

NeoCoreXMS は、XML 文書データの他に、その格納された XML 文書データのメタ情報を、MetaData セクション配下に格納します。これらメタデータは、規定集や約款など、文書の履歴管理や版管理が必要とされるアプリケーションから利用することが可能です。

- 変更時刻 ＜ModifyTime＞
  XML 文書が更新されたときのシステム時刻（秒単位）。XML 文書を初めて格納したときには、TimeStamp タグと同じ時刻を設定します。
- 格納時刻 ＜TimeStamp＞
  XML 文書が格納されたときのシステム時刻（秒単位）。XML 文書をコピーした場合、コピー先には新しい時刻を設定します。
- ソースファイル ＜SourceFile＞
  データの元となるファイルの名前。格納時に設定したファイル名を設定します。
- 文書 ID ＜DocID＞
  格納順に XML 文書に割り当てられるユニークな値です。
- 文書コピーの番号 ＜CopyNumber＞

XML 文書のコピー時に割り当てられる管理番号です。

- スキーマファイル ＜SchemaFile＞
  文書と一緒にスキーマを格納した場合に、その文書スキーマが入っているファイルの URI を設定します。
- プレフィックスファイル ＜PrefixFile＞
  プレフィックスファイルの名前（prefix.xml など）を設定します。

3. テスト方法

事前に用意したテストプログラム（Java）の実行により、NeoCore XMS、A 社製品（RDB）の各オペレーションにおける処理速度、通信量(検索時)を計測。

また、以下を考慮してテストを実施：

- データベースのキャッシュ機能が使用されないよう、オペレーションごとにデータベースサーバを再起動する。
- 計測は(ストアを除いて)10 回行い、その最大値と最小値を除いた値の平均値を使用する。

3－1. テストデータ

テストデータとして Twitter の公開ステータスをリアルタイムに取得できる「Streaming API」を使用して投稿された発言(ステータス)を収集したものを使用（データ数：100 万件）

【データサンプル】

```
<status>
  <created_at>Fri Apr 16 08:10:35 +0000 2010</created_at>
  <id>12271540700</id>
  <text>ビックカメラ派</text>
  <source>&lt;a href="http://www.nibirutech.com" rel="nofollow">TwitBird iPhone&lt;/a></source>
  <truncated>false</truncated>
  <in_reply_to_status_id/>
  <in_reply_to_user_id/>
  <favorited>false</favorited>
  <in_reply_to_screen_name/>
  <user>
    <id>103350540</id>
    <name>go</name>
    <screen_name>gk_steppers</screen_name>
    <location>Tokyo Japan</location>
    <description>DJ STEPPERS RECORDS</description>
    <profile_image_url>http://a3.twimg.com/profile_images/692431697/_____3_normal.png</profile_image_url>
    <url>http://ameblo.jp/sr-tokyo/</url>
    <protected>false</protected>
    <followers_count>130</followers_count>
    <profile_background_color>709397</profile_background_color>
    <profile_text_color>333333</profile_text_color>
    <profile_link_color>FF3300</profile_link_color>
    <profile_sidebar_fill_color>A0C5C7</profile_sidebar_fill_color>
    <profile_sidebar_border_color>86A4A6</profile_sidebar_border_color>
    <friends_count>124</friends_count>
    <created_at>Sat Jan 09 19:09:58 +0000 2010</created_at>
    <favourites_count>11</favourites_count>
    <utc_offset>32400</utc_offset>
    <time_zone>Tokyo</time_zone>
    <profile_background_image_url>http://a1.twimg.com/profile_background_images/75049632/andy_warhol.jpg</profile_background_i
mage_url>
    <profile_background_tile>true</profile_background_tile>
    <notifications/>
    <geo_enabled>false</geo_enabled>
    <verified>false</verified>
    <following/>
    <statuses_count>177</statuses_count>
    <lang>ja</lang>
    <contributors_enabled>false</contributors_enabled>
  </user>
  <geo/>
  <coordinates/>
  <place/>
  <contributors/>
</status>
```

3 – 2. 使用クエリ

NeoCore

NeoCore 特有の Xpath （/ND で開始）を使用し、テストデータ内の<location>タグ値の検索、挿入、更新、削除テストを実施。

A 社製品

A 社製品の SQL 文を使用し、XML 機能により Xpath を条件として設定し、<location>タグ値の検索、挿入、更新、削除テストを実施。

※1)検索条件

いずれの製品も、検索条件はヒットする検索結果件数が異なるような条件をいくつか指定した。（条件として”Cambodia”(検索結果 35 件)、”Planet Earth”（検索結果 262 件)、”Bandung”（検索結果 577 件)、”Australia”（検索結果 1008 件)、”Jakarta”（検索結果 2221 件）を使用した）

※2)挿入・更新・削除方法

いずれの製品も、以下に記載した方法で実装＆計測を行った。

挿入方法： 指定の条件(id=’4’で、location=’東京’などの条件)に適合する XML の特定要素の配下に特定要素（<add_data>挿入データ</add_data> のようなタグと値のセット）を挿入した。ここで指定の条件とは、※1 の検索条件と同一とする。（以下更新、削除も同様）

更新方法： 指定の条件に適合する XML の特定要素(<location>東京</location>など)を別の値(<location>Jakarta </location>など)に更新した。

削除方法： 指定の条件に適合する XML の特定要素を削除した。

3 – 3. テスト環境

テストでは、同じスペックのマシン 2 台（NeoCore、A 社製品用）を用意し、テストを実施。

各マシンスペックは、以下の通り。

NeoCore 用マシン

プロセッサ: Pentium(R) Dual-Core CPU @ 2.60GHz(2 CPUs)

メモリ: 2038MB RAM

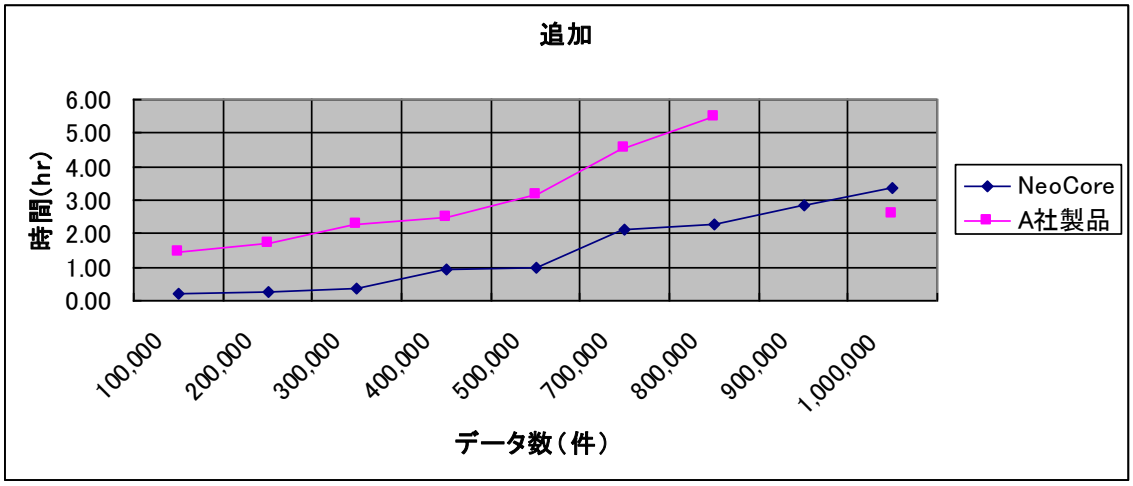
A 社製品用マシン

プロセッサ: Pentium(R) Dual-Core CPU @ 2.60GHz(2 CPUs)

メモリ: 2038MB RAM

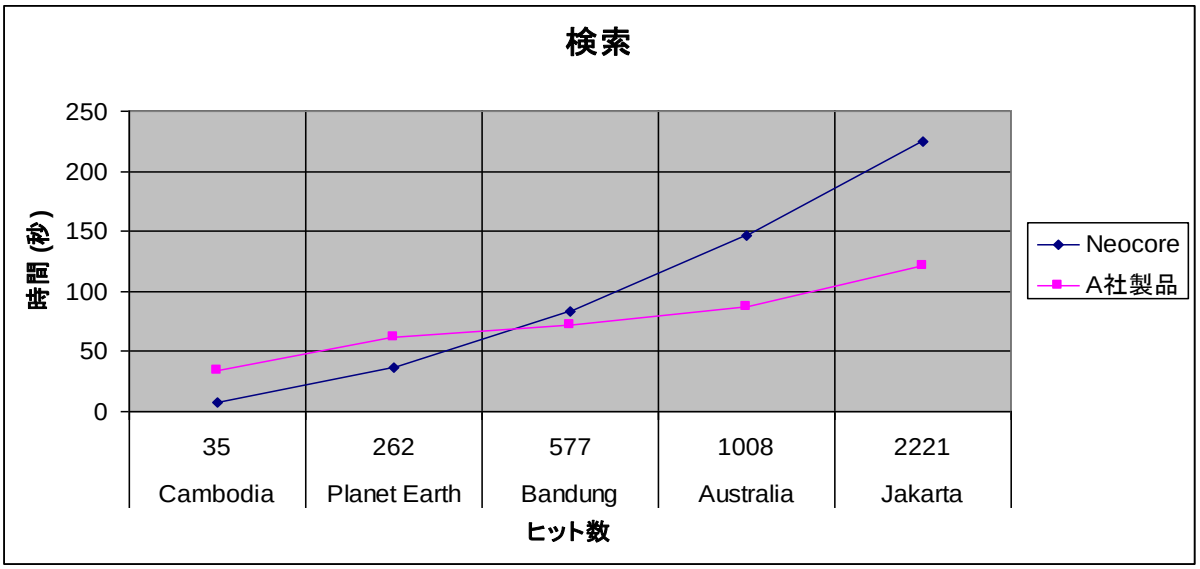
4. 各処理におけるテスト結果、検証

各処理におけるテスト結果は、以下の通り。



- 100 万件データ追加時の処理時間比較

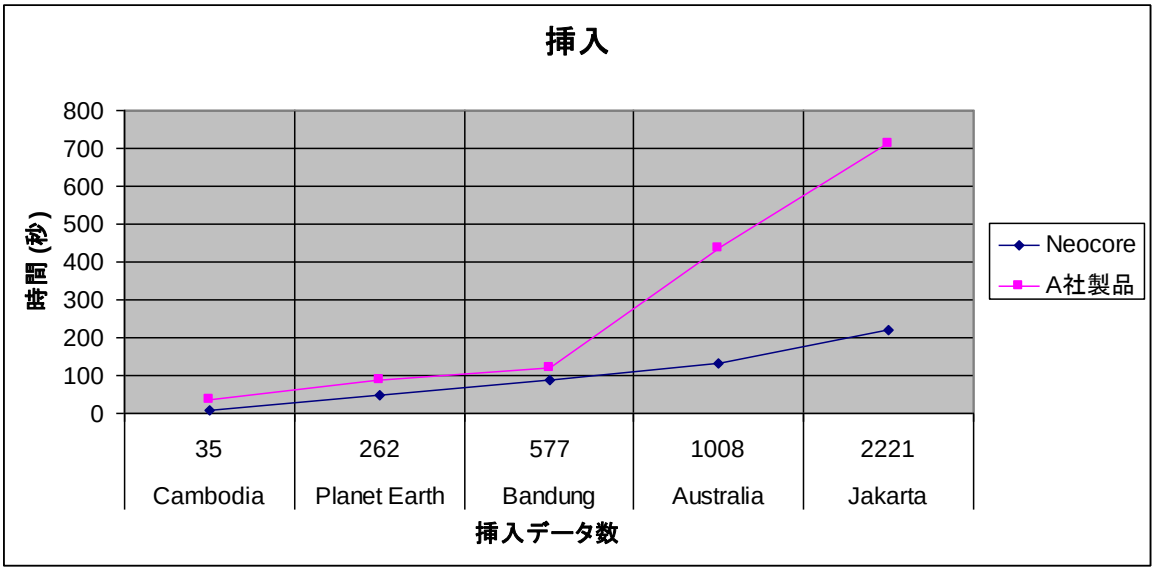
追加処理において NeoCore はデータ数の増加に対して、ほぼ均等に追加時間が増えていくのに対し、A 社製品では、データ数の増加と共に、処理時間が倍増する結果となった。(80 万件以降のデータ追加時に、エラーが発生したため、これ以降は時間計測せず、一回に投入するデータ件数を減らしてトータル 100 万件のデータを格納した)



検索	Cambodia	Planet Earth	Bandung	Australia	Jakarta
NeoCore	7.94	36.46	83.55	132.97	224.77
A 社製品	34.34	61.41	72.51	87.26	121.77

- 特定の文字列の検索時の処理時間比較

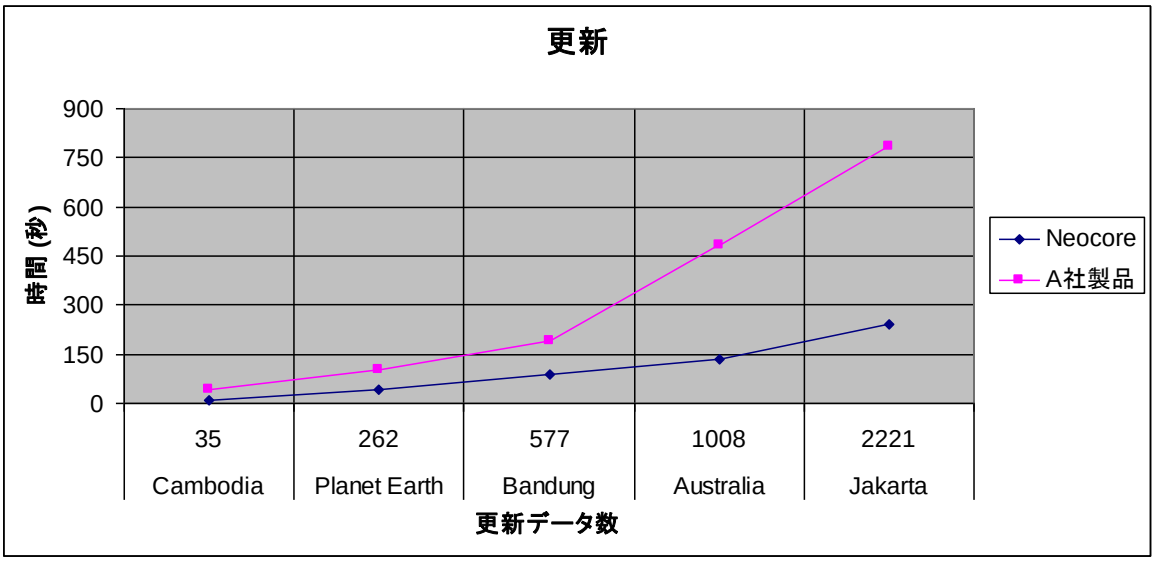
データの検索処理では、ヒットするデータの数が少ない場合は、NeoCore の検索速度の方が速いのに対し、A 社製品では、データの増加量に関係なく、均等に処理速度が低下する結果となった。(グラフの処理時間の上がり幅が一定である)



挿入	Cambodia	Planet Earth	Bandung	Australia	Jakarta
NeoCore	9.96	48.17	88.74	132.97	221.68
A 社製品	36.71	86.26	118.61	436.06	710.94

- 特定の文字列の挿入時の処理時間比較

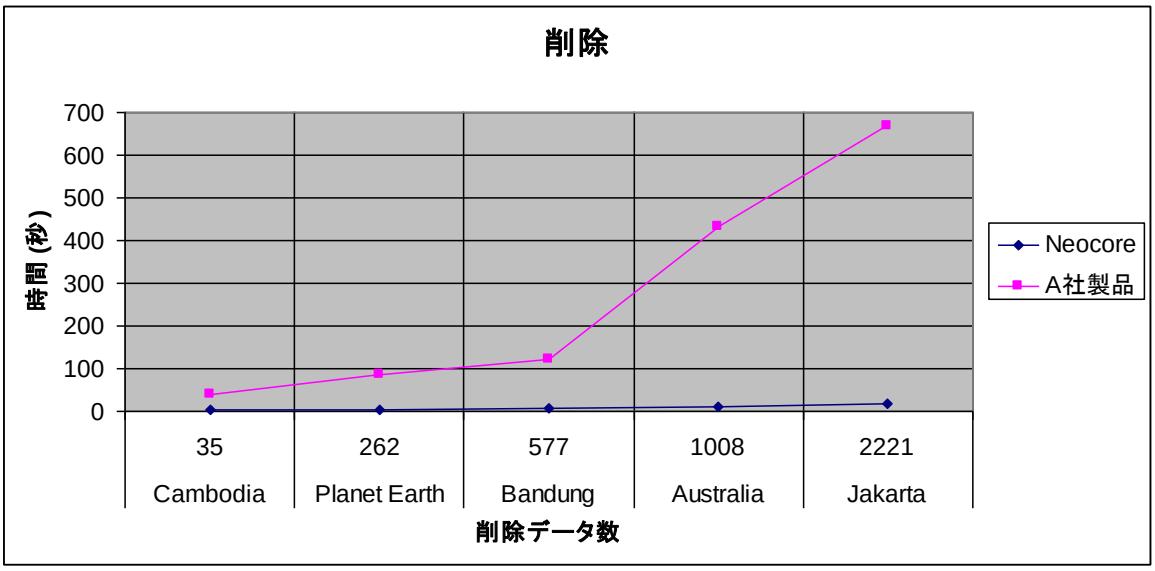
データの挿入処理では、データの量に関わらず、NeoCore の処理速度の方が速い結果となった。特にデータ量の増加により、A 社製品の挿入処理速度が急激に低下するなど、明らかな差が見られた。



更新	Cambodia	Planet Earth	Bandung	Australia	Jakarta
NeoCore	8.78	42.45	90.14	132.97	239.56
A 社製品	39.97	100.04	188.66	481.86	784.15

- 特定の文字列の更新時の処理時間比較

データの更新処理も、挿入処理と同様に、データの量に関わらず、NeoCore の処理速度の方が速い結果となった。特にデータ量の増加により、A 社製品の更新処理速度が急激に低下するなど、明らかな差が見られた。



削除	Cambodia	Planet Earth	Bandung	Australia	Jakarta
NeoCore	1.97	4.59	8.47	11.03	17.06
A 社製品	39.64	87.19	121.12	431.37	666.65

- **特定の文字列の削除時の処理時間比較**

削除処理においては、NeoCore の処理速度が全体的に A 社製品より速い結果となった。NeoCore の処理速度は、データ量に関係なくほぼ一定を保っているのに対し、A 社製品では、挿入、更新と同様、データの増加に伴い、処理速度が急激に低下する傾向が見られた。

5. 考察

- 大量の XML データを扱う必要がある場合、A 社製品の XML 機能と比較して NeoCore は更新系処理で約3倍以上高速に動作する事が判った。また、検索系処理についても、2～4倍程度高速に動作するという結果が得られた。

- 検索結果として大量にデータを返す場合に限定されるが、A 社製品の XML 機能が、NeoCore より処理速度が速いケースが存在する。しかし、この問題は、以下の対策を行う事で回避可能である。
  - 構造が異なるスキーマレスデータを NeoCore に格納する際、同一構造を持つXMLドキュメントは、同じカテゴリ属性を持たせるようにする。そのデータをアプリケーションから検索する際は、カテゴリで絞り込みを行ってから、その配下の XML ドキュメントを検索する方法にする事で、検索結果を最小限に抑え、安定したパフォーマンスを得ることができる。
  - 全文検索エンジンオプション製品である、「QuickSolution for NeoCore」を併用する。

- 上記の通り、XML データベース「NeoCoreXMS」は、構造の異なるスキーマレスの XML データの処理に関しては、RDB の XML 機能よりも高速かつ安定したパフォーマンスが得られる事が実証された。しかし、NeoCoreXMS のパフォーマンスを生かすためには、XML データを設計する段階で考慮しなければいけない点がいくつか存在する事も事実である。