企業システムにおける XML・JSON とデータベース利用技術

日大生産工　○豊谷　純
日大生産工　渡邊　昭廣
日大生産工　亀井　光雄

1．はじめに

Web2.0 と呼ばれる技術が登場し，GoogleMap などでは，リアルタイムにブラウザ上の地図をマウスで移動出来たり，公開 API の利用によって Amazon や楽天の商品情報を参照したり，RSS(Really Simple syndication)によって自分の欲しい情報だけを収集することが可能になった。

これらは JavaScript による Ajax (Asynchronous JavaScript + XML)の登場や Flash の発展によるものであり，以前では使い勝手が問題となっていた Web アプリケーションのユーザインターフェースを格段に向上させている。

また Web サービスの登場によって，図１に示されるように，全く異なった開発言語で構築されたシステムが，お互いにインターフェースを公開し合う事により，相互にデータ交換を行い連動させる事が可能になった。

これによって企業間の商取引がお互いの既存システムを，継続して有効活用することが可能になり，業務の IT 化へ大きく貢献をしている。

さて，この Web アプリケーションであるが，世界中からアクセス可能であるため，セキュリティーポリシーにより，経理や営業関連のシステム等で利用されることは稀である。

しかしスケジュール管理や会議室や社内資産の予約管理システムなどは，課や部署の枠を超えて全社的に利用する必要があるため，Web アプリケーションとして利用されている。

ここで問題となっていたのが，Web アプリケーション特有の使い勝手の悪さであったが，先の技術によりこの問題は格段に改善され，今後益々業務システムが Web 化される事が予想される。

そしてここで用いられているデータ形式には，クライアントとサーバ間では XML や JSON 形式が採用される事が多いが，サーバ

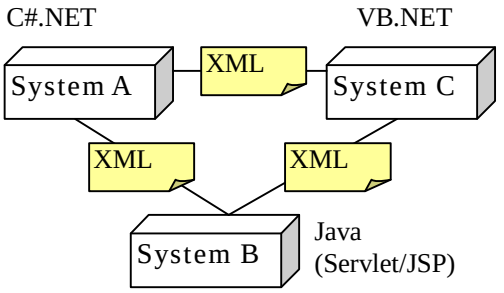

図１　Web サービス概要図

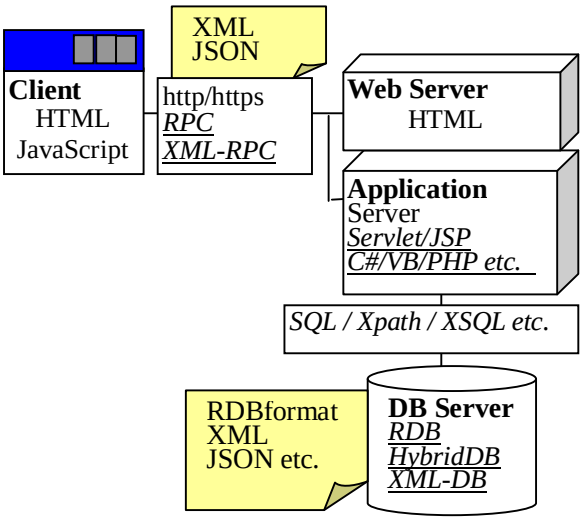

図２Web アプリケーションサーバ構成図

内でのデータ書式に関しては決まった仕様が無く，独自の書式で管理されている。

そして XML や JSON 形式のデータ管理で用いられるデータベースは一般的には Relational Data Base (以後 RDB)が利用されている。最近では Oracle や DB2，SQLServer など主要は製品は殆ど RDB を基本に XML 対応としたハイブリッド型データベースに発展させている。その他，XML 専用のデータベースも製品化されており，データの保存形式や管理方法も様々なものが考えられる。

従って本論では，経営面から見た情報の有

XML / JSON Applications in Corporate System

Jun TOYOTANI, Akihiro WATANABE and Mitsuo KAMEI

効活用性と，技術面から見たシステム構築の機能面を検証し，業務システムでは，データベースやデータ設計をどのようにすべきなのか，設計指針を明らかにする。

２．XML と JSON 形式

XML は技術の枠を超えて異なるシステム間で情報交換を行う際に用いられるメタ言語である。汎用言語であり，特にプログラミングやプラットホームを選ばない。

また JSON(JavaScript Object Notation)も XML と同様であるが，基本的には JavaScript での利用に限定されるが，XML よりも簡略化されておりデータ量が少なくてすむ。

例として商品コード a100000 のデータを XML と JSON で表記すると，次のようになる。

XML 形式で記載した場合：

```
<?xml version="1.0"?>
<item>
  <cd>a100000</cd>
</item>
```

JSON 形式で記載した場合：

```
{
 "item":[
   {"cd":"a100000"}
  ]
}
```

XML は各業界毎に標準化が図られており，またデータの見出しとなる要素名に意味の分かりやすい名称を付けるため，第三者でも何を意味するデータなのかが分かるようになっている。

一方 JSON はあくまでもクライアント側の JavaScript でもプログラミングが容易で，ソースコードが短くて済むという利点がある。

どちらも要素名が付いているため，これだけを見るとＪＳＯＮの方が，データ文字列が短くて済むことが分かる。

ただし，データをこのシステム内だけで利用するにはＪＳＯＮ形式でも良いが，将来的に他のシステムと連携する可能性があれば，標準仕様となっているＸＭＬ形式にしておく方が有利である。

３．XML のデータベース処理方法

XML をデータベースへ保存するには，いくつかの方法[1)]が考えられる。

マッピングと呼ばれる方法は，XML の１つの要素名を，RDB の項目名に置き換えて変換する方法である。これはデータの仕様変更により，データ項目が追加あるいは変更された際には，対応不可能となる。

また RDB に，XML をそのまま文字列として保存する方法も考えられるが，目的のデータを検索する際に，データベースのインデックス機能を活用することが出来ない。

XML をそのまま利用する方法として，ハイブリッド型を利用する方法と，XML 専用データベースを利用する方法がある。

RDB(Relational DataBase)

A: XML を１つの文字列として管理

1	<?xml version="1.0" encoding="UTF-8"?><item><cd>a100001</cd>・・・
2	<?xml version="1.0" encoding="UTF-8"?><item><cd>a100002</cd>・・・
	・・・

B: XML を１つの文字列として管理し，検索用データを別項目に付加

1	<?xml version="1.0" encoding="UTF-8"?><item><cd>a100001</cd>・・・	a100001
2	<?xml version="1.0" encoding="UTF-8"?><item><cd>a100002</cd>・・・	a100002
	・・・	

上記の B の方法は A に対して，索引のための情報を付加させて検索を高速化するものである。

RDB/XML Hybrid Type DataBase

C: XML をそのまま管理

1	<?xml version="1.0" encoding="UTF-8"?><item><cd>a100001</cd>・・・
2	<?xml version="1.0" encoding="UTF-8"?><item><cd>a100002</cd>・・・
	・・・

この方法は，XML をそのまま保存して，さらに各要素(項目)にデータベース用のインデックスを設定する事ができる。

ただし，Hybrid 型データベースの多くは，保存した XML の検索機能は高速であるが，新たに要素や属性を追加して XML を部分更新する事が出来ない製品が多い。本テストで用いた DB2 も部分更新が出来ず，更新は XML そのものを差し替える必要があった。

```
<?xml version="1.0" encoding="UTF-8"?>
<item>
    <cd>a 100001</cd>
</item>
```

```
<?xml version="1.0" encoding="UTF-8"?>
<item>
    <cd>a 100002</cd>
</item>
```

```
<?xml version="1.0" encoding="UTF-8"?>
                ・・・
```

図３　テスト用の XML データ

４．テスト方法

性能評価としてシステムからデータベースに対して，XML や JSON 形式のデータを，登録(追加)，更新(変更)，検索，削除処理させてその処理時間を計測した。データベースは IBM の DB2Ver9.1 を採用した。

RDB 操作言語は世界標準の SQL を採用しており，また Hybrid 型に関しては発展途上という事もあり，操作言語は数種類存在するが，一般的なものとして検索には XQuery を利用した。処理は Java プログラム[2)3)]で 250～1500 件をそれぞれ 10 回ほど実施して，平均値を採用した。

５．テスト結果

XML と JSON 形式による処理時間の比較は殆ど同じであった。これは単純に文字列として扱うため，文字数に応じて処理時間が変わることが確認できる。

RDB で扱う場合，その違いは文字数が JSON の方が少ないため，若干高速に処理が行われることが確認できる。

次に，図８～図１１に Hybrid 型データベースの XML 型を利用した場合と，RDB の処理時間が示されている。基本的には Hybrid 型は登録や更新や削除が非常に遅いことが分かる。データ量を増やしても，処理時間はほぼ線形に増加することが確認できる。

通常インデックスを設定すると，さらに処理が遅くなるが，登録処理で 1%，更新処理で 2%程度であった。

ただし検索処理に関しては，RDB は図９の十字マークで示されているが，XML を文字列として，あいまい検索するしかない。データ量に比例して処理時間が増大しているのが分かるが，Hybrid 型はインデックスが有効に活用されるため，データ数を増やしても処理速度は高速に保たれる事が確認できる。

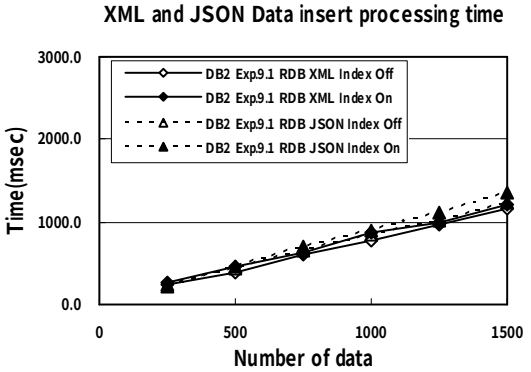

図４　XML と JSON データの追加時間

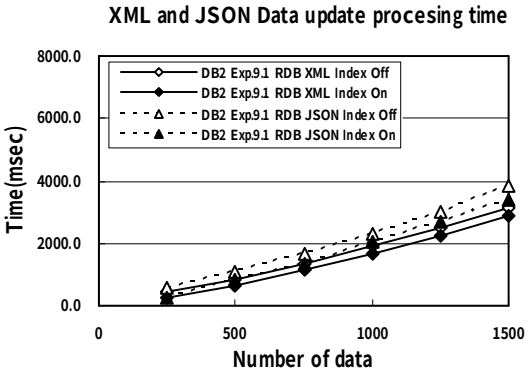

図６　XML と JSON データの更新時間

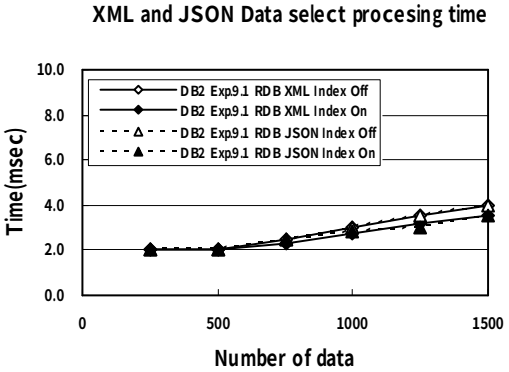

図５　XML と JSON データの検索時間

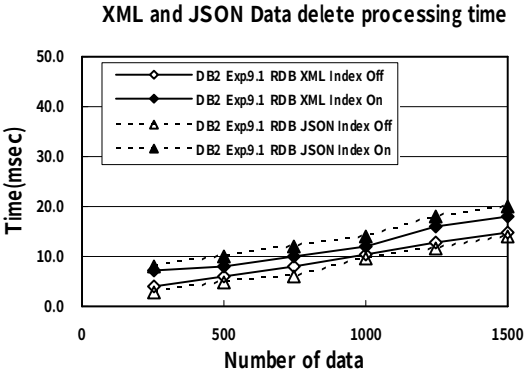

図７　XML と JSON データの削除時間

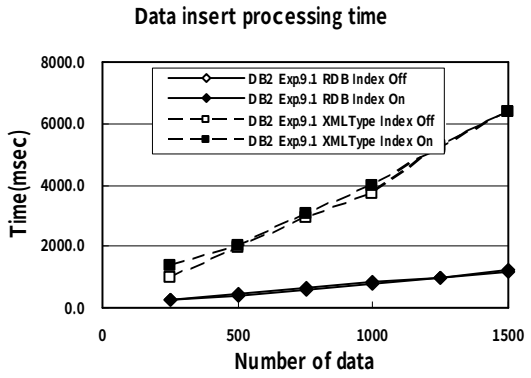

図８ XML 文字列と XML 型の追加時間

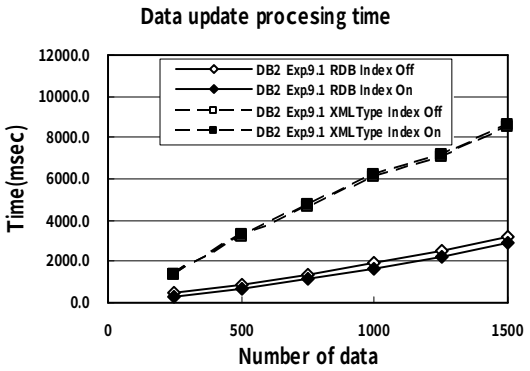

図１０ XML 文字列と XML 型の更新時間

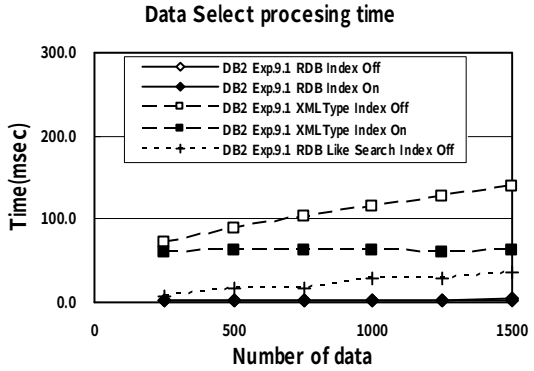

図９ XML 文字列と XML 型の検索時間

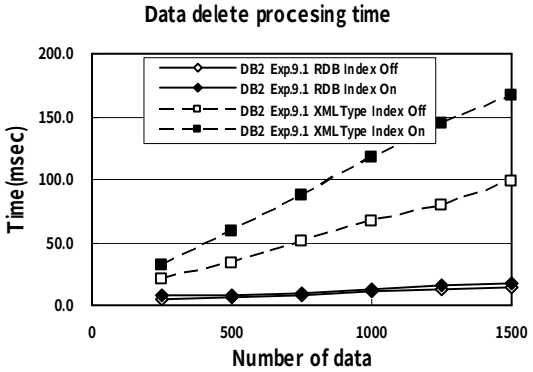

図１１ XML 文字列と XML 型の削除時間

６．おわりに

JSON や XML 形式のデータを企業システムで利用する際には，どのようなデータ設計や，データベース選定を行うべきかを検証した。

データが XML 形式の場合は，Hybrid 型や XML 専用の XML 型が利用でき，Xpath や XSQL といった XML に特化した機能を有効に活用できる。しかし JSON に関しては，このような技術仕様も無く，JSON 専用データベースというものは存在しないため，このような利用が不可能な現状である。

その結果として，RDB は高速ではあるが，JSON や XML 形式のデータを単純なる文字列としか扱えない。しかし Hybrid 型では特定の XML 要素にインデックス設定を行う事が可能であり，その処理性能も非常に高いことが分かった。

企業システムの中核となるのはサーバ側で処理が行われるバックエンド側のプログラムであり，これは JavaScript では作成が出来ず，Java や C#，VB などが用いられる。しかも企業によってはセキュリティーポリシーにおいて，JavaScript が使えない企業も多い。JSON は製造者のプログラマから見ると手軽であるが，汎用性に問題がある。これに対して，XML はすでに世界標準のデータ交換言語であり，他の多くの技術からも再利用可能である。

これらのことから，企業において業務用 Web アプリケーションを構築する際には，データモデリングは XML が有用であり，それを有効活用するためには，XML 対応のデータベースが必要と結論付ける。

参考文献

1) 豊谷，渡邊，角田，亀井，DOM/SAX による XML 処理と XML データベースの処理性能,日本大学生産工学部 第 37 回学術講演会 講演概要, pp.69～72, 2004 年
2) 豊谷，渡邊，若林，清水，XML-DB と RDB による XML データの処理特性，日本情報ディレクトリ学会誌, Vol.4, pp.23～26, 2006 年
3) 豊谷，若林，渡邊 他， XML データベースと DOM・SAX による処理特性， 日本ロジスティックスシステム学会誌，Vol.6, no.2, pp.63～68,2006 年
4) Toyotani, Watanabe, Wakabayashi, Osawa, et. al, Application method and processing characteristic of XML of physical distribution system of handicap terminal, Proc. of 3rd Int. Congress on Logistics and SCM Systems, pp.266～ 271, 2007