Visual Basic 並びに Oracle を用いた XML 型情報検索ツール

Information Retrieval Tool of XML Type by Visual Basic and Oracle

宮崎 佳典, 曽 勝
Yoshinori MIYAZAKI, Shou SOU

インターネットの時代に莫大の情報をどう検索するか、通信するかが今日の課題となって久しい。この問題を解決するためにデータとデザインが分離されているメタ言語 XML が一つの方法として考えられている。そこで本研究では、データベースシステムと XML と VB を利用した検索モデルの試作を行った。データベースシステムには Oracle を用い、試作モデルでは VB によって作成された検索画面を媒介とし、XML の内容を XQL によってデータベースとして表に格納し、Oracle 内の SQL 命令を通して検索する仕組みになっている。本論の研究目的は、数学表現に対して柔軟に対応できない従来の検索エンジンに対し、文章構造をも考慮し、文章内容そのままだけではなく文章を格納するタグまで対象として検索することができるツールを開発することである。近い将来、MathML などの数学表現を取り扱うマークアップ言語が標準でブラウザに搭載されることが予想される中、このようなツール開発は急務とされると考えている。

第 1 章 XML の基礎知識と最新動向

1.1. XML とは

XML(eXtensible Markup Language) は HTML(HyperText Markup Language)と同じ SGML(Standard Generalized Markup Language)を基本にしてできた言語である。文章の情報を、任意のプラットフォームやマシン、ソフトウェア間でやりとりできるようにするという設計方針で開発されたものである。これを実現するため、文章を「データ(内容)」「データ構造」と「デザイン(体裁)」に分離したのが HTML と異なる XML の大きな特徴でもある。

具体的に XML が利用される目的は、「文章標準化」「Web ページ」「データ交換」の三つに分類できる。世間ではあたかも XML が HTML に代わる新たなマークアップ言語であるような触れられ方であるが、必ずしもそうではなく、他 2 つの目的でも需要が大きい。

まず、1 つ目の分野は「文章標準化」である。SGML の延長にあるもので、論文や報告書、雑誌や新聞の記事といった日常使われている文書を統一的なフォーマットに納めようというものである。XML という共通のフォーマットを使うことで、異なる組織間でも容易に文章情報を交換できるようにする。日本国内では、特に政府機関が出す各種官報や、役所

に届ける書類を電子化するにあたり、XML を利用する動きが活発になってきている。また医療のカルテについても XML を使った電子化が進んでいる。

2 番目の分野「Web ページ」は HTML の延長として Web ページの記述に XML を利用するという動きである。インターネットの普及によって、HTML で書かれた Web ページが数え切れないほど製作され、今も加速度的に増加している。ところが、これら多くの HTML ページは、データとデザインをごちゃまぜに記述しているのが現実である。これを XML 化することで、データとデザインを明確に分け、各種ソフトウェアが自動的にデータを部分だけを取り出して処理し、魅力的なデザインを動的に提供できるようになる。

3 番目の「データ交換」分野は実は次の 2 つの分野で注目されている。すなわち EC（電子商取引）や EAI（企業内アプリケーション統合：Enterprise Application Integration）の分野である。これまで企業間（B2B：Business to Business）ないし企業内でシステム同士が何らかの情報をやりとりする場合は EDI などの共通規格や、独自のプロトコルを採用する必要があったが、これも XML が間に立つことにより、従来の複雑なシステム化を軽減することに貢献している。

1.2. XML 周辺の動き

近年様々な分野で XML に対する激しい変化が起こっている。XML は世界有数のソフト会社が推奨する技術であり、多くのソフトウェアで XML フォーマットを扱うようになってきている。周知のように、今後は PC 市場に携わる全てのユーザーが関係する技術になると思われる。

現在入手可能な XML 関連ソフトウェアを分類すると、XML 文書を「作成」し「解析」し「変換または格納」する次の 6 つに大別される：

（1）XML パーサー
XML 文書のタグ付けや DTD を解析するソフトウェア。XSLT 処理機能を実装した製品もある。

（2）XML データ変更ツール
ワープロ文書や HTML 文書、SGML 文書などを XML 文書に変換するツール。DTD の自動変換機能もある。

（3）XML エディタ
XML 文書を効率よく作成するためのエディタ/ワープロ。DTD 作成を支援する機能も備える。

（4）XML 対応データベース
RDB(リレーショナ・データベース)または ODB（オブジェクト指向データベース）に、XML 文書を格納する。

（5）EAI/B2B サーバー
XML 文書をシステム間に繋ぐ「パイプ」として使い、異なる XML 文書の DTD 解析、タグ名称の自動変換など、大量の XML 文書を効率よく処理する。

（6）その他ユーティリティ
XML スキーマ設計ツール、DTD エディタ、Excel との連携ソフト、コンポーネントなど。

これら XML 対応製品のうち、現在特に注目を浴びているのは、本稿でも取り上げる XML 対応データベース技術である。データベースが XML を扱う場合、二つのパターンが考えられる。一つは、XML 文書の要素をテーブルの各フィールドにマッピング(変換)して、XML 文書をデータベースに格納する方法である。もう一つが、XML 文書を塊のまま一つのフィールドに格納する方法である。いずれにせよ、多くのデータベース・ベンダーが XML 対応を推進する理由は、データベース側から見ると XML は理想的なデータ・フォーマットだからである。異なるデータベース間あるいはシステム間でデータベース同士を連携させるには、タグを見るだけでそのデータの「意味」が分かる XML を使って連携させれば、データベース連携が容易になる。結果としてデータベースの利用がさらに進むと期待されている。

1.3. XML ファイル作成並びに Web ページ表示の例

この節では XML を用いた具体例を紹介する。特に XML が幅広く利用される理由の一つである自己定義できるタグについて示す。まずは著者の作成した中国の家族構成*についてタグ付けした XML ファイルの内容(ほんの一部)を見てみよう。図 1 が HTML 形式の場合、図 2 が XML 形式の場合である。

```
<body><table border=" 1" >
  <tr>
   <td>曽</td>
   <td>立群</td>
  </tr>
  <tr>
    <td>朱</td>
    <td>秋華</td>
  </tr>
  <tr>
    <td>曽</td>
    <td>勝</td>
        …
```

図 1 : HTML による記述例

```
<家族>
<父><名前>
  <苗字>曽</苗字>
  <名前>立群</名前>
</名前>
</父>
<母><名前>
  <苗字>朱</苗字>
  <名前>秋華</名前>
</名前>
<子供>
  …
```

図 2 : XML による記述例

図 1 と図 2 を比べると、HTML の方（図 1）では固定的なタグが使われているから、データの意味は分からない。逆に、XML の部分（図 2）ではタグを自分で定義できるから、データの内容の意味がタグで表されている。つまり XML ではタグがデータの意味を代表し、アプリケーション側ではタグに応じた処理を書けばよいわけである。逆に注目すべきは、Web ページなどでデータを表現する方法が構造や内容とは分離していることである。XML データは、まず構造と内容を定義してからスタイルシートを適用し、表現を定義するからであ

*日本と違い、中国の世帯では妻は改姓することなく、旧姓を保持することが多い。故、家族構成を書くときにもファイル構造が微妙に変化してくる。

る。XML データは、異なるスタイルシートを適用するだけで簡単にさまざまな方法で表現することができ、たとえば、ユーザー・プロファイル、ブラウザのタイプ、またはその他の基準にもとづいて異なる表現スタイルごとに別のスタイルシートを定義すれば適切に表示が可能となる。

実際にXMLで作ったサンプルを表示してみる。図3はWeb上で表示した結果である。図4はXMLのインスタンスである。そして図5はXMLをHTMLに変換してWebで表示するためのXSL(スタイルシートファイル)である。図4に示しているインスタンスからもタグの自己定義を有しているため、タグ名でデータが何なのかが人間からみても一目瞭然である。

Welcome to my homepage

名前:曽　勝

出身:中国

年齢:22歳

趣味:スポーツ

職業:大学生

メールをください f9821175@mail0.fujieda.ssu.ac.jp

図 3：XML サンプル表示結果

```
<?xml version="1.0" encoding="Shift_JIS"?>
<?xml-stylesheet type="text/xsl" href="123.xsl"?>
<全体>
  <題名>Welcome to my homepage</題名>
    <写真>123.gif</写真>
      <名前>名前：曽　勝</名前>
      <出身>出身：中国</出身>
      <年齢>年齢：22 歳</年齢>
      <趣味>趣味：スポーツ</趣味>
      <職業>職業：大学生</職業>
  <連絡><メール>メールをください</メール>
      <mailto>f9821175@mail0.fujieda.ssu.ac.jp</mailto>
  </連絡>
</全体>
```

図 4：図 3 のソース XML ファイル

```
<?xml version="1.0" encoding="Shift_JIS"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/TR/WD-xsl">
<xsl:template match="/">
<html><head><title>
    <xsl:value-of select="全体/題名"/>
    <style type="text/css">
      body{background-color:gold;}
    </style>
</title></head>
<body>
  <xsl:apply-templates select="全体"/
</body>
</html>
</xsl:template>
```

```
<xsl:template match="全体">
  <center><h1><xsl:value-of select=
   "題名"/></h1></center>
  <left> <img>
        <xsl:attribute name="src">
          <xsl:value-of select="写真"/>
        </xsl:attribute>
      </img>
  </left>
  <h3>
        <p><xsl:value-of select="名前"/></p>
        <p><xsl:value-of select="出身"/></p>
        <p><xsl:value-of select="年齢"/></p>
        <p><xsl:value-of select="趣味"/></p>
        <p><xsl:value-of select="職業"/></p>
    </h3>
  <left>
    <h2><xsl:value-of select="連絡"/></h2>
  </left>
</xsl:template>
</xsl:stylesheet>
```

図 5：XSL ファイル

```
<h3>
 <p><xsl:value-of select="名前"/></p>
 <p><xsl:value-of select="出身"/></p>
 <p><xsl:value-of select="年齢"/></p>
 <p><xsl:value-of select="趣味"/></p>
 <p><xsl:value-of select="職業"/></p>
</h3>
```

図 6：図 5 の一部

```
<h3><table border="1">
<tr>
<td><xsl:value-of select="名前"/></td>
<td><xsl:value-of select="出身"/></td>
<td><xsl:value-of select="年齢"/></td>
<td><xsl:value-of select="趣味"/></td>
<td><xsl:value-of select="職業"/></td>
</tr> </table></h3>
```

図 7：レイアウト変更

それでは、図 5 のスタイルシートの一部（図 6）を下の図 7 のように変化してみよう。
このとき、ブラウザ出力は図 8 のようになる。

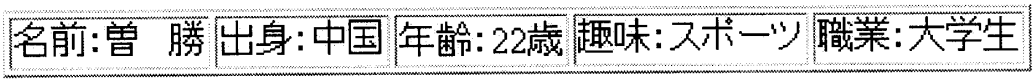


図 8：図 7 のブラウザ出力(IE)

XML 文章を Web ブラウザで読むのか、携帯電話で表示するのか、またはデジタル家電装置で見るのかは自由である。同じ内容の XML 文章はスタイルシートだけを変換すれば、さまざまな機械装置に対応し、異なるインタフェースで表示できる。データベースに格納されたデータは一切変更する必要がない点を考慮すれば、従来の HTML と比べた上での XML のひとつの大きな特長である。

第 2 章 XML 型情報検索

2.1. Oracle と VB による検索エンジン

データベース・ベンダーで XML に対して最も積極的なのは米 Oracle**だろう。日本オラクルが 2000 年 4 月から出荷しているデータベース Oracle8i　Release8.1.6 シリーズでは XML データをデータベース内に格納するための XML パーサーの実装が可能である。Oracle から提供されているツールを使うとブラウザ経由でデータベースにアクセスして処理結果を受け取ることができる。このように、Oracle を利用することで、容易にデータベースと XML を連携させたアプリケーションを作成することが可能となる。

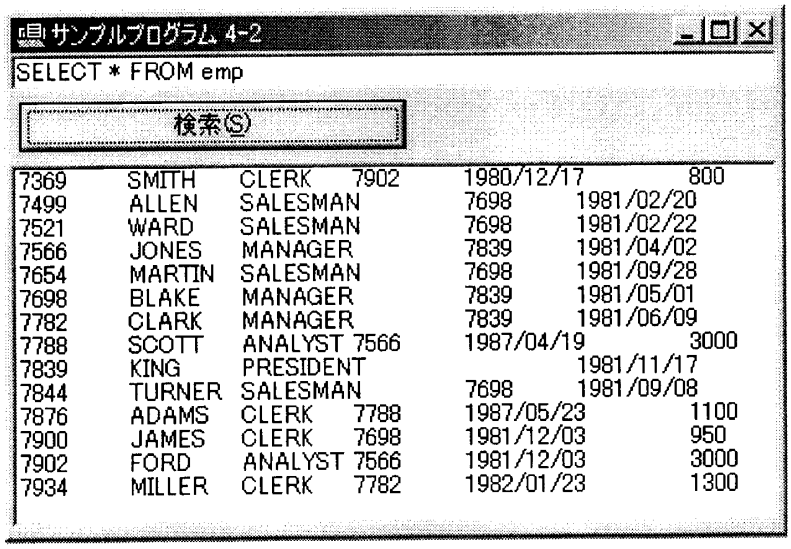


図 9：VB+Oracle の検索モデル（一例）

** 最近の調査によると、世界トップクラスの Web サイトのうち 69%が Oracle のテクノロジーを利用しているとの報告もある。

Oracle8i には、Java や XML など、インターネット標準のネイティブ・サポートが組み込まれている。実際、Oracle JServer、Oracle8i の組込み Java Virtual Machine を使用して、Java や XML で構築された Oracle XML コンポーネントとアプリケーションをデータベース自体で実行することができる。また、Oracle を使うための関数ライブラリである OCI を使いやすい形でクラス化することで VB に Oracle 入出力機能を追加することができる。その機能を使うことにより、VB で作成したプログラムが Oracle クライアントプログラムから閲覧可能となる。図 9 は Oracle と VB を組合して作った検索モデルの一例である。テキストボックスに SQL を入力し検索ボタンを押下すると、検索結果はリストボックスに表示される。

検索モデルの中身は次の様である ([4], 162p)。

```
Dim strSQL      As String
    Dim oraDs       As Object
    Dim strErrText  As String
    Dim strRow      As String
    Dim ilngCOl     As Long
On Error GoTo errClick:
    Me.MousePointer = vbHourglass
    Me.Refresh
      strSQL = Trim$(txtSQL.Text)
    Set oraDs = poraDb.DbCreateDynaset(strSQL, 4&)
     lstResult.Clear
    Do While Not oraDs.EOF
        strRow = ""
        For ilngCOl = 0 To oraDs.fields.Count - 1
            strRow = strRow & oraDs(ilngCOl).Value & vbTab
        Next
        lstResult.AddItem strRow
        oraDs.DbMoveNext
    Loop
    MsgBox "検索完了", vbOKOnly + vbExclamation, App.Title
exitClick:
    On Error Resume Next
    Set oraDs = Nothing
    Me.MousePointer = vbDefault
    Exit Sub
errClick:
    If poraSess.LastServerErr = 0 Then
        If poraDb.LastServerErr = 0 Then
            strErrText = Error$
```

```
        Else
            strErrText = poraDb.LastServerErrText
            poraDb.LastServerErrReset
        End If
    Else
        strErrText = poraSess.LastServerErrText
        poraSess.LastServerErrReset
    End If
    MsgBox "frmSamp402/cmdExecute_Click:" & strErrText, vbOKOnly + vbExclamation,
App.Title
    Resume exitClick:
    Resume Next
End Sub
```

2.2. XML 型情報検索

2.1で紹介したVBとOracleによる検索モデルは実際にXML文章に対してもその検索が可能である。しかし、Web上にあるXML文章はすべて表の形で存在するわけではないから、そのまま検索することは困難である。それで、XML文章を表に変換し、格納しなければならない。

検索の手順としては、まずは検索したいXML文章をOracle内に定義される表形式に変形し、格納する必要がある。これを実現するのがXML SQL Utility for Javaである。Oracle XML SQL Utility for Javaは、データベースへの問い合わせ結果をXML文書として生成するJavaのクラス群である（図10）。逆に文字列としてXML文書を出力することも、メモリ上にDOMの木構造として出力することもできる。さらに、問い合わせが参照している表を基にDTDを生成することも可能である。

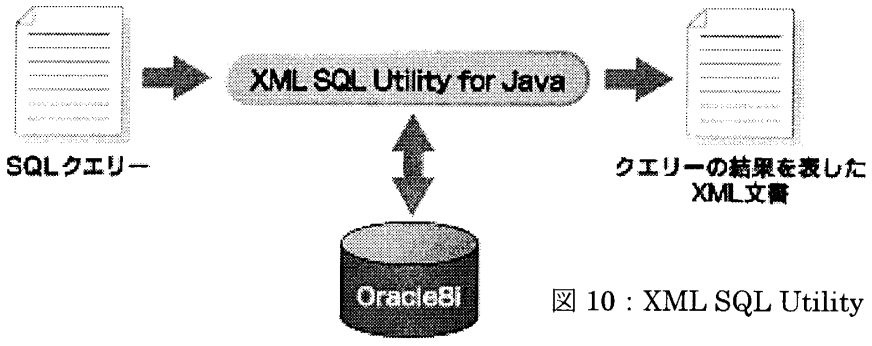


図 10：XML SQL Utility

Oracle XML SQL Utility for Java には、OracleXMLQueryというクラスがあり、このクラスがSQL クエリーの結果をXML に変換する処理をしてくれている。このOracle XML SQL Utility for Java にはOracleXMLSave というクラスも同様に用意されていて、XMLデータを

データベースに格納する。

例えば、前掲したXML文章のデータを格納したいとする。まず、この文章内の<名前>、<年齢>と<出身>タグの内容をデータベースの図11のようなDDLで作成された表に格納する。

```
CREATE TABLE 自己紹介(
名前 VARCHAR2(200),
年齢 VARCHAR2(200),
出身 VARCHAR2(200)
) ;
```

図 11：DDL の一例

上のXML 文書はそのままでは、表に格納できない。表に格納するためには、表にあったXMLに変換する必要がある。そこで、OracleXMLSave を用いて表に格納できるようにするための正しいXML の形を知るために、XSQL Page を用いる（図12）。

```
<?xml version="1.0"?>
<xsql:query connection="demo" xmlns:xsql="urn:oracle-xsql" max-rows="1">
SELECT *
FROM 自己紹介
</query>
```

図 12：XSQL Page

処理結果は以下図13のようになる。

```
<?xml version = '1.0'?>
<ROWSET><ROW num="1">
<名前>曽勝 </名前>
<年齢>22歳 </年齢>
<出身>中国 </出身>
</ROW></ROWSET>
```

図 13：図 12 の処理結果例

以上を考慮して、新たに検索モデルを試作した。画面上に三つの検索ボタンがあり、それぞれXML文章のタグ名に対応している。例えば、テキストボックスの中に“中国”と入力してから“出身”ボタンを押すと、結果は図14のように表示される。勿論、名前や年齢で検索する場合も同様の操作で行える（図15）。また、検索結果のリストには各人のURIをも出力し、自動的にホームページを立ち上げることができるようにも設計した。

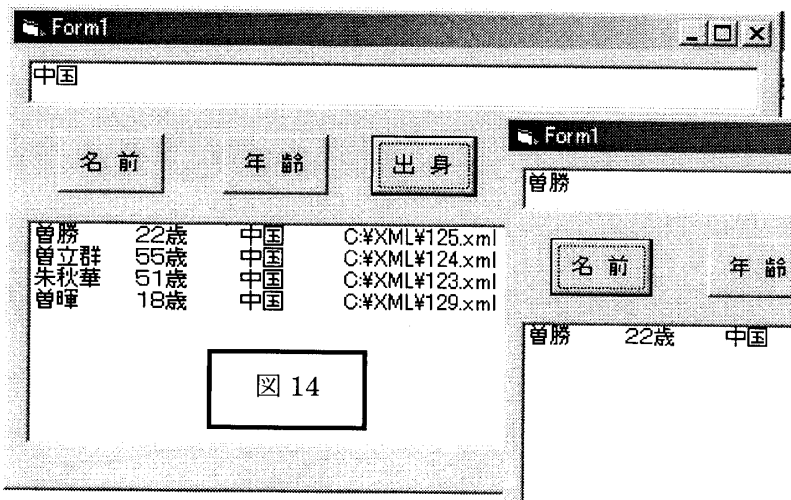


図 14

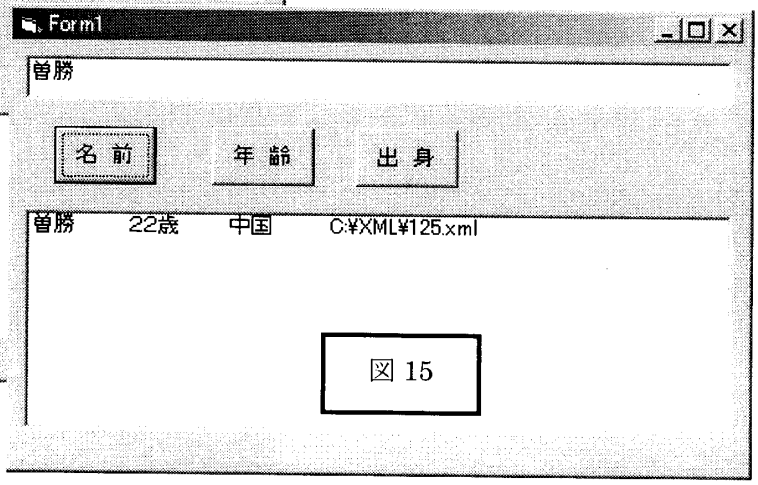


図 15

XML文章のデータ構造は木構造であるため、タグ間の親子関係を明確に規定している。複雑なXML文章情報を検索したい時、文章内容から、意味付けXMLタグを用いて検索した方が速いことが考えられる。例えば、ある商品の価格を検索したい時、「商品種類」から「商品名」、そして「価格」まで、タグの木構造を利用して根から枝、親から子のように検索していくと適切な情報を確実に探し出すことができるはずである（図16と17）。

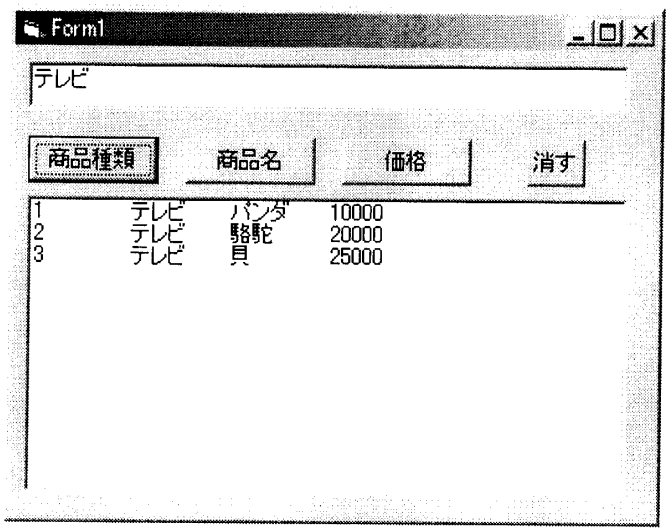


図 16

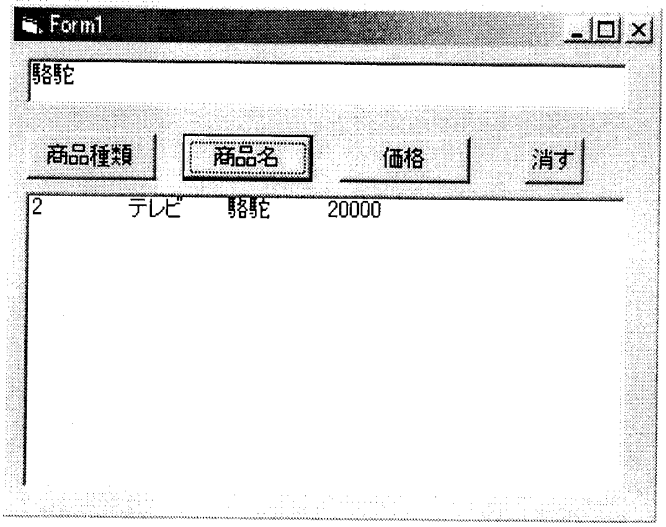


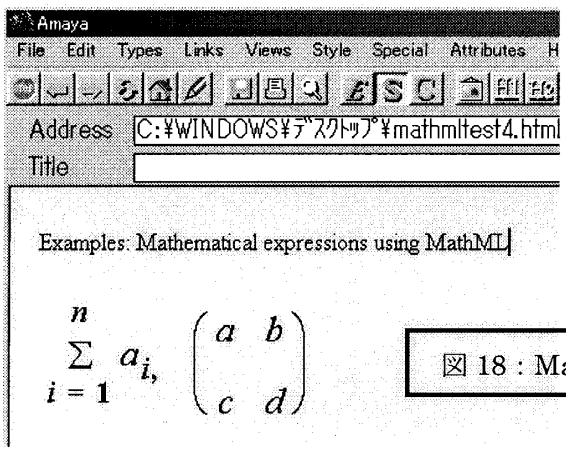
図 17

第 3 章 MathML 型情報検索

MathML (Mathematical Markup Language) [11]は W3C (WWW Consortium) [10]が開発している XML の１アプリケーションである。現在の所、Web 上で数学や科学などの特殊記号を表示するのは簡単ではない。しかし、MathML を使えば自由に作成する事ができる。現在

の所、二大ブラウザである IE (Internet Explorer) と NN (Netscape Navigator) が MathML をサポートしていないため、代わりに W3C が独自開発したフリーのブラウザである Amaya[12]を使えば表示できる。図 18 は Amaya による MathML 文章のサンプル表示である。

数式 $\int_1^2 \frac{dx}{x} x \mid t \mid x$ に対応するMathMLインスタンスは次のようになる：



$$\sum_{i=1}^{n} a_{i,} \begin{pmatrix} a & b \\ c & d \end{pmatrix}$$

図 18：MathML の出力例

```
  <?xml version="1.0"
encoding="iso-8859-1"?>
  <math
xmlns="http://www.w3.org/199
/Math/MathML">
  <mrow>
  <semantics>
    <mrow>
      <msubsup>
        <mo>&int;</mo>
        <mn>1</mn>
        <mi>t</mi>
      </msubsup>
      <mfrac>
        <mrow>
          <mo>&dd;</mo>
          <mi>x</mi>
        </mrow>
        <mi>x</mi>
      </mfrac>
    </mrow>
```

```
  <annotation-xml
encoding="MathML-Content">
      <apply>
        <int/>

<bvar><ci>x</ci></bvar>

<lowlimit><cn>|</cn></lowlimit>

<uplimit><ci>t</ci></uplimit>
        <apply>
          <divide/>
          <cn>|</cn>
          <ci>x</ci>
        </apply>
      </apply>
    </annotation-xml>
  </semantics>
  </mrow>
  </math>
```

図 19：図 18 ソース

MathMLはXMLの応用アプリケーションとして関数表示を意味付けタグを用いて表示しているため、簡単に検索することができる。実際、2.2に紹介した検索システムを数式の検索にも対応できるように作り替えた。図20と図21がその実行結果の一例である。数学記号が書かれている検索ボタンを押すとそれを含むファイルがリストボックスに表示される（図20)。さらに特定のリストをダブルクリックすれば（Amaya実行により）MathML形式のファイルがビジュアルな数式表示とともに表示されるようにプログラムを組んである（図21)。

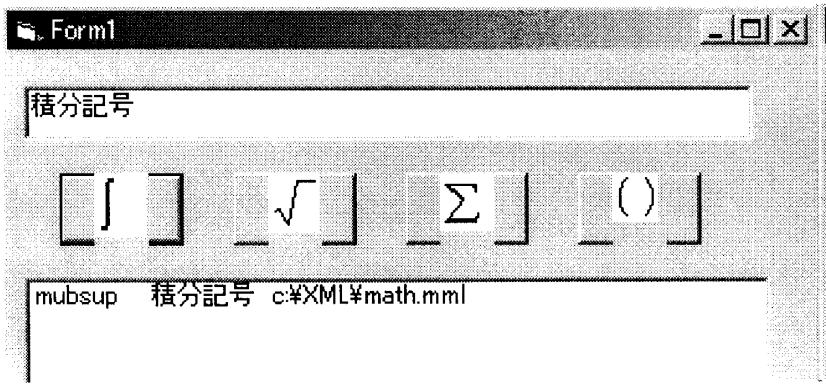


図 20：数式検索例

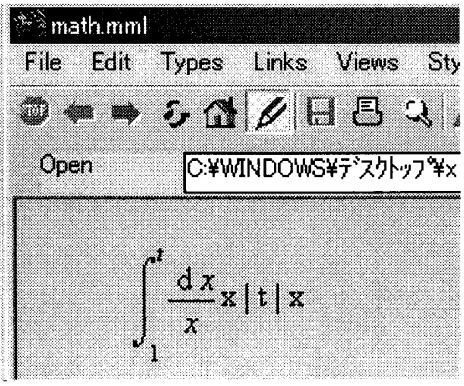


図 21：ヒット情報表示

これによって、将来MathMLを標準ブラウザがサポートした段階で各種数式の検索が可能となる。現在の数式表示は主に gif などの画像ファイルに頼っている為、データ量が無駄に多いだけでなく、数式の検索が実質上不可能である（HTMLの場合、Altタグに数式を書き込むなどの対処策はあるが根本的に問題は解決されていない)。本検索プログラムはそれに対して新たな検索対象を取り込む布石となる。さらに、MathMLの木構造(MathMLはXMLの一応用例であるから、MathMLとXML同様、木構造をもつ)を考えればより複雑な検索が可能となる。例えば、今まで「“積分記号”（$\int$)と“平方根”（$\sqrt{\ }$)をもつ数式」という検索(結果として $\sqrt{\int 2xdx}$ や $\int\sqrt{2xdx}$ などが入る)のみ可能であったのが「積分記号の中に平方根をもつ数式」とすると上述２つのうち、$\int\sqrt{2xdx}$ などを選り抜くことが可能となる。

第 4 章 結論と今後の課題

本研究では XML（ならびにその応用としての MathML）文章の構造化を考慮し、タグを指定して適切な内容検索のできる簡易検索モデルを試作した。XML で記述される Web 上文章データを検索できることがモデルを通して証明された。しかし、Web 上の既存 XML データをまず表に変換しなければならないので、検索時間に影響を与えることは必至である。現在の検索システムが直面している大きな問題に「WWW ページの指数関数的な増大に対して、現在のシステムは、いつまで耐えられるのか」を考えると、この問題も将来はクリアしな

ければならない。また、検索の過程と結果を重視すると同時に検索の結果をどのように(時・時間・場合に応じて) 表示させるかも考えなければならない。

今後の方向性としてはOracle XML SQL Utility for Java を利用して、検索と表示の両方が効率的かつ分かりやすいような検索エンジンシステムをさらに考案していく予定である。そして、XML で作成したホームページの情報だけではなく、MathML や VoiceML など幅広い範囲の XML 情報を検索することができる検索システム開発に臨みたいと考えている。

謝辞

本研究は静岡産業大学研究活動助成金制度 (2000 年度) のサポートを受けている。

参考文献

[1] 日経ソフトウェア, 日経オープンシステム : XMLガイド, 日系BP社, (1998).
[2] 林孝光, 中島謙二 : Oracle8i SQL完全ガイド, ソフト・リサーチ・センター, (2000).
[3] PHP, Perlで使うXML/XSL, 有限会社マークアップ, (2001).
[4] 初音玲 : Visual Basic+Oracleいちから始めるシステム構築, 翔泳社, (2000).
[5] XML/SGML サロン : 標準 XML 完全解説, 技術表論社, (2000).
[6] 河西朝雄 : Visual Basic 6.0, 技術評論社, (2000).
[7] オラクル・テクニカル・ホワイトペーパー http://otndnld.oracle.co.jp/
[8] 日本オラクル株式会社 http://www.atmarkit.co.jp/
[9] 電子技術総合研究所http://www.etl.go.jp/~yamana/Research/WWW/survey.html
[10] W3C(WWW Consortium) http://www.w3c.org/
[11] MathML http://www.w3c.org/Math/
[12] Amaya http://www.w3c.org/Amaya/