データベースシステム 121－20
情 報 学 基 礎 58－20
（2000. 5. 26）

構造化文書の異種性解消のための「文書群」の導入と検索機能の実現

國島丈生
岡山県立大学情報工学部

鈴木美沙*
東芝アドバンストシステム (株)

宮川由香*
(株) シンフォーム

横田一正
岡山県立大学情報工学部

現在コンピュータ上で扱うことのできる文書には、フォーマット、バージョンに代表される変種、ファイル構成の違いなど、さまざまな異種性が存在する。我々は、このように異種性を持つ電子文書を**異種文書**と呼び、XML を用いてそれらを統合的に管理する方法について研究を行っている。今回我々は、異種文書管理の枠組のなかで、意味的に互いに関連する文書を扱うために**文書群**という概念を提案する。さらに、XML 文書および文書群に対する検索機能を統一的な方法で与え、DOM を用いて実装を行った。

Document Groups and their Retrieval Functions for Resolving Heterogeneities of Structured Documents

Takeo Kunishima
Okayama Prefectural Univ.

Misa Suzuki
Toshiba Advanced Systems Co., Ltd.

Yuka Miyagawa
Synform Co., Ltd.

Kazumasa Yokota
Okayama Prefectural Univ.

In electronic documents there are various heterogeneities such as file format, versions, and document structure design. We call those electronic documents with heterogeneity **heterogeneous documents**, and proposed an integrated management paradigm of heterogeneous documents based on XML. In this report, we propose the concept of **document groups** of heterogeneous documents to treat semantical relationships between the documents. We also propose integrated retrieval functions to both heterogeneous documents and document groups. These retrieval functions are implemented on DOM (Document Object Model).

*2000 年 3 月までの所属は岡山県立大学情報工学部

1 はじめに

現在コンピュータ上で扱うことのできる文書には、さまざまなフォーマットが存在し、これらは意味的に同一の文書と見なせるなどの関連によって互いに関連づけらている。また、同一の文書に関して変種が存在する場合がある。たとえば、著者らが進めている文学データベースの研究 [9,11] では、同一の物語であっても書かれた時代や著者が異なると表現が異なり、その差異を調べることが比較文学研究の重要な研究要素である。さらに、電子文書では、同一内容であってもファイル構成が異なるような場合がある (図 1)。

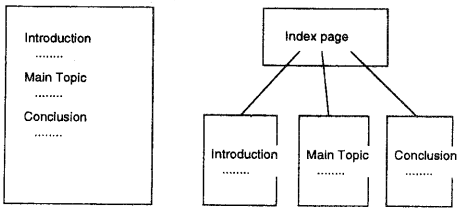


図 1: ファイル構成による電子文書の異種性

我々は、このようにファイル形式の異なる電子文書、あるいは同一の文書の変種を**異種文書**と呼ぶこととし、意味的に関連する異種文書を統合的に管理する方法について提案し、研究を行っている [10]。この異種文書管理方式では、すべての電子文書を相互変換可能な形で XML に変換し、その上に検索などの機能を実現する。異種文書管理の全体の概略を図 2 に示す。

今回我々は、異種文書管理の枠組のなかで、意味的に互いに関連する文書を扱うために**文書群**という概念、およびそれに対する検索機能を提案し、実装を行った。本稿ではこれについて報告する。

2 XML の木表現

XML 文書に含まれる文書要素[1]は木構造をなすことが一般に知られている。このあとの議論のため、XML の文書要素が構成する木の定義を示す。

[1]XML version 1.0 の規格では「要素」(element) と呼ばれている。集合の要素などと区別するため、本論文では文書要素という言葉を用いる。

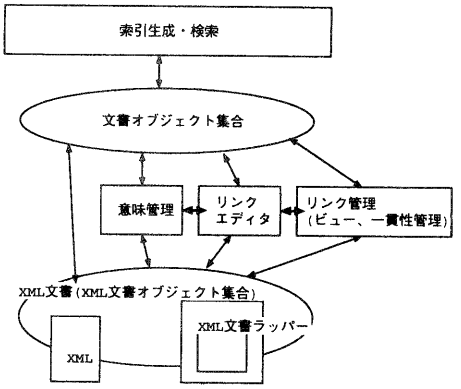


図 2: 異種文書管理システムの概要

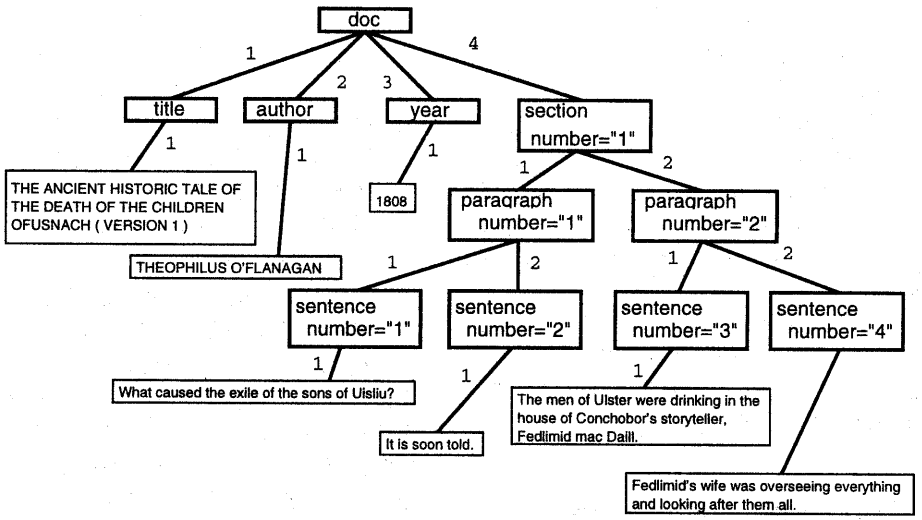
要素木 (element tree) を次のように定義する。

- 要素木のノードには要素ノード、PCDATA ノードの 2 種類がある。
- 要素ノードは XML の文書要素に対応しており、対応する文書要素名をラベル、その文書要素の属性を属性として持つ。要素ノードの属性は、元文書中での対応する属性が持っていた値を、値として持つ。PCDATA ノードは文字列 (PCDATA 値) に対応しており、対応する PCDATA 値を値として持つ。
- 要素ノード x は、1 個以上の PCDATA ノードと 0 個以上の要素ノードを子ノードとして持つ。子ノードはそれぞれ、x に対応する文書要素が包含している文字列または文書要素に対応している。子ノードへの枝には、対応する文字列または文書要素の XML 文書中での出現順を表すラベルが付加されている。このラベルにより、兄弟ノード間に順序が定義される。
- PCDATA ノードには子ノードは存在しない。

空文書要素に対応する要素ノードは、"EMPTY" という特殊な値を持つ PCDATA ノードを子ノードとして持つと考える。

XML 文書は、根となる文書要素をただひとつしか含まないため、XML 文書に対応する文

```
<?xml version="1.0"?>
<doc>
<title>
THE ANCIENT HISTORIC TALE OF THE DEATH OF THE CHILDREN OFUSNACH ( VERSION 1 )
</title>
<author>THEOPHILUS O'FLANAGAN</author>
<year>1808</year>
<section number="1">
<paragraph number="1">
<sentence number="1">
What caused the exile of the sons of Uisliu?
</sentence>
<sentence number="2">
It is soon told.
</sentence>
</paragraph>
<paragraph number="2">
<sentence number="3">
The men of Ulster were drinking in the house of Conchobor's storyteller,
Fedlimid mac Daill.
</sentence>
<sentence number="4">
Fedlimid's wife was overseeing everything and looking after them all.
</sentence>
</paragraph>
</section>
</doc>
```



(太枠のノードが要素ノード、細枠のノードが PCDATA ノードを表す。)

図 3: XML 文書と XML 文書木の例

書木はただひとつだけ存在する。これを**XML文書木** (XML document tree) と呼ぶ。図3にXML文書と対応するXML文書木の例を示す。

本稿では、エンティティ参照が展開されていることを仮定している。すなわち、対象とするXML文書はエンティティ参照を持たない(XML version 1.0 [2] によって規定された amp, lt, gt, apos, quot の5つのエンティティを除く)と仮定している。エンティティ参照によって一つのXML文書が複数のXML文書に分割されている場合の扱いについては4章で述べる。

また、XMLでは、文書要素の内容として、ほかの文書要素とPCDATA値が混在する「混在内容」というものが許されている。このため、要素ノードのほかにPCDATAノードを用意している。

XML文書木上での要素ノードのノードアドレスの表現方法はいくつか考えられる。本稿では、以下のようなものを用いる。要素ノード x について、文書木上での深さを n、根から x に至るパス上のノードのラベルを順に $l_1, l_2, \cdots, l_n$ とする。このとき、x のノードアドレスを次のように定義する。

$$[l_1(j_1), l_2(j_2), \cdots, l_n(j_n)]$$

ただし、$l_i(j_i)$ は、該当する要素ノードが、文書木の深さ i においてラベル l_i を持つ要素ノードのうち j_i 番目に出現することを表す。たとえば、図3のXML文書木において、属性numberの値が2であるようなsentence文書要素に対応する要素ノードのノードアドレスは $[doc(1), section(1), paragraph(1), sentence(2)]$ となる。

$l_n(j_n)$ だけで要素ノードは一意に定めることができるので、このノードアドレスは冗長である。しかし、3章で述べる検索処理において、この冗長性は有効に働く。

3 XML文書に対する検索

ここでいう「検索」とは、キーワードなどの検索条件、検索対象となるXML文書集合、出力したい文書要素名を入力として与え、出力として、条件を満たす文書要素に対応する要素ノードのノードアドレス集合が得られる、という処理である。

検索のための質問 Q は次の3つ組で与えられる。

$$Q = (D, e, C)$$

ここで D は質問の対象となるXML文書集合、e は質問結果となる文書要素名であるが、"document" という特殊なシンボルを指定することもできる。また C は検索条件であり、本稿では、次の定義によって構成される式とする。

1. l を文書要素、r を正則表現とするとき、$match(l, r)$ は検索条件である。また、$@a$ を属性、r を正則表現とするとき、$match(@a, r)$ は検索条件である。r には、文字列の先頭を表す "^"、文字列の末尾を表す "$" を用いることができる。
2. A, B が検索条件であるとき、$A \vee B$, $A \wedge B$, $\neg A$, (A) はいずれも検索条件である。

以降、1を基本条件、2を複合条件と呼び、まず基本条件に対する検索を、次に複合条件に対する検索を議論する。最後に検索結果の加工について議論する。

3.1 基本検索条件の意味

基本検索条件の意味は、次のように与えられる。

$match(l, r)$ l を根のラベルとして持つ要素木について、それに含まれるPCDATAノード[2]の値を深さ優先順に連結して得られる文字列が正則表現 r に含まれるときかつそのときに限り真。

$match(@a, r)$ 属性 a の値が正則表現 r に含まれるときかつそのときに限り真。

質問 $Q = (D, e, C)$ において C が基本検索条件である場合、C が真になる要素ノードを子孫に持ち、ラベルが e であるような最小の要素ノードのノードアドレスの集合が解となる。たとえば、図3のXML文書に対して、検索条件 $match(year, 1808)$ で検索をかけると $\{[doc(1), year(1)]\}$ が解として得られる。e としてシンボル "document" が与えられたときは、C が真になる要素ノードを持つ文書木に対応するXML文書の集合が解になる。

[2] PCDATAノードは子ノードを持たないため、PCDATAノードは必ず葉になる。

現在のところ、文書要素の内容に対する型指定の規格はまだ定まっていない[3]ため、本稿では、内容は文字列としてのみ扱う。

3.2 複合検索条件の意味

A, B を検索条件とするとき、要素ノード x について $A \vee B$ が真になるのは、x について A が真または B が真のとき、かつそのときに限る。また、x について $A \wedge B$ が真になるのは、x の子孫に A が真になる要素ノードと B が真になる要素ノードの両方が存在するとき、かつそのときに限る。

たとえば、図3の XML 文書 (franagan.xml) において、

$$A = match(paragraph, Uisliu)$$
$$B = match(sentence, Daill)$$

という検索条件を考える。このとき、A はノードアドレス $[doc(1), section(1), paragraph(1)]$ の要素ノードで真、B はノードアドレス $[doc(1), section(1), paragraph(2), sentence(3)]$ の要素ノードで真になる。したがって、質問 $(franagan.xml, section, A \wedge B)$ という検索条件で検索をかけると、解は $\{[doc(1), section(1)]\}$ となる。また、

$$(franagan.xml, paragraph, A \vee B)$$

という検索条件で検索をかけると、解は

$$\{[doc(1), section(1), paragraph(1)], [doc(1), section(1), paragraph(2)]\}$$

となる。

$\neg A$ の場合、単独では否定のノードアドレスの集合として保持され、他の条件と組合されたとき評価される。たとえば、上の検索条件 A, B について、$A \wedge \neg B$ は $[doc(1), section(1), paragraph(1)]$ で示される要素ノードで真、$\neg A \wedge B$ は $[doc(1), section(1), paragraph(2), sentence(3)]$ で示される要素ノードで真となる。

(A) は演算子の優先順序のための記法であり、意味は A と同じである。

[3]XML Schema Part 2 [3] で規格の制定が進められている。現在 Working Draft。

3.3 検索結果の加工

質問の結果得られたノードアドレス集合をさらに加工して出力させたいことがある。たとえば、図3のような構造を持つ文書の集合に対し、「authorが国島であるような文献の titleを求める」という質問の場合、(XML 文書集合, doc, $match(author, 国島)$) の結果得られるノードアドレス集合 (文書要素 doc に対応) から title 文書要素を求める必要がある。

XML-QL [7] では、この機能を実現するために CONSTRUCT/WHERE という構文を用意し、WHERE 節で条件を満たした文書要素の内容などを変数に束縛し、CONSTRUCT 節で変数の値を参照しながら任意の XML を出力できるようにしている。今回我々は、検索機能には必要最低限の加工機能のみ用意し、さらに結果を加工したい場合は XSLT [5] のような XML 変換のための規格を用いることとした。XML-QL のような独自の加工機能を用意するよりも実現できる範囲は狭いが、検索機能が単純化され理論的に扱いやすくなる、XSLT プロセッサなどの既存の実装を利用できる、などの利点があると考えている。

検索条件を評価して得られたノードアドレスの集合を A、加工のために指定された文書要素名を l、加工結果として新たに生成するノードアドレス集合を A' とすると、

- $a \in A$ で表される要素ノードの先祖に l をラベルに持つノード x がある場合、$x \in A'$。
- $a \in A$ で表される要素ノードの子孫に l をラベルに持つノード x がある場合、$x \in A'$。
- $a \in A$ で表される要素ノードと共通の親を持つ兄弟に l をラベルに持つノード x がある場合、$x \in A'$。

たとえば、検索条件 $match$(author, O'Flanagan) で検索をかけることを考える。このとき、文書要素 year を指定すると $\{[doc(1), year(1)]\}$ が、また文書要素 doc を指定すると $\{[doc(1)]\}$ がそれぞれ解として得られる。

4 文書群とその検索

4.1 文書群

文書群(Document Group)は、互いに関連づけられた異種文書の集合である。異種文書の関連には、たとえば、(1) 生成 (ある文書から別の文書が自動的に生成された)、(2) 異なるバージョン、(3) 編集 (ある文書の内容を人手で異なるフォーマットに直した)、(4) 一部共有 (文書の一部を共有している)、(5) 分割 (意味的には一つである文書を分割した)、(6) 付加 (メモなどの関連文書を新たに付加した)、などいろいろなものが考えられるが、本稿では関連の種類については議論しない。また、異種文書間の関連、文書群の定義 (どの文書から文書群が構成されるか) などはユーザが決定し、自動では決定しないことを想定している。

本稿の文書管理の枠組ではすべての異種文書は相互変換可能な形でXML形式に変換された後、管理される。したがって、文書群は、互いに関連づけられたXML文書の集合として扱われる。

複数のXML文書を関連づける方法として次の3つが考えられる。

1. エンティティ参照による文書の統合
2. xlink:href属性によるリンク指定 [6]
3. 外部リンクセット (external linkset) によるリンク指定 [6]

このうち、1は巨大なXML文書を分割する場合に広く使われる手段であるが、XMLパーザを通すと文書の分割に関する情報が保存されるとは限らない (単一の文書として扱われる可能性がある)、エンティティ参照された文書に制約がある (ex. 文書型定義 (DTD) を持つことができない)、など、異種文書管理の点から見ると問題がある。2は2つの文書要素間に対してリンクを張る手段であるが、関連する異種文書は一般に2つとは限らないので、リンクが複雑になる可能性がある。以上のような理由から、本稿では、複数のXML文書の関連づけは3で行うこととした。元の文書がエンティティ参照によって分割されている場合は、それと同等の定義を3によって行う。

4.2 文書群に対する検索

文書群の概念を導入する理由の一つに、意味的に関連する文書への検索機能の強化がある。たとえば、図1に示した二つの文書群はユーザから見ると意味的には同一のものであるから、キーワード "Main Topic" で検索した結果は同一になってほしい。このような検索は、ほかに、メモなどの関連文書を新たに付加して文書群を構成した場合や、バージョンのように関連する文書すべてに同じキーワードが含まれているとは限らない場合などに対して有用である。このような検索を実現するには、単なる全文検索では不充分であり、文書群に対する検索機能を考える必要がある。

文書群に対する検索のために、要素ノードのノードアドレスを文書名を含むように拡張する。すなわち、XML文書 d に対応するXML文書木について、拡張ノードアドレスは次のようになる。

$$[d, l_1(j_1), l_2(j_2), \cdots, l_n(j_n)]$$

文書群に対する質問 Q' を次のような3つ組で定義する。

$$Q' = (DG, e, C)$$

DG は対象となる文書群集合を、e は質問結果となる文書要素名を、C は検索条件を表す。e には、3章の質問と同様に "document" というシンボルのほか、"document_group" というシンボルも用いることができる。

C が基本検索条件のとき、Q' の解は、DG に含まれるすべての文書の文書木を対象にして、C が真になるような要素ノードを子孫に持ち、ラベルが e であるような要素ノードの拡張ノードアドレス集合となる。e としてシンボル "document" が指定された場合は、C が真になるような要素ノードを持つ文書木に対応する文書の集合が解になる。またシンボル "document_group" が指定された場合は、C が真になるような要素ノードを持つ文書木に対応する文書を含むような文書群集合が解になる。

C が複合検索条件のとき、次のように意味を与える。(DG, e_A, A) の解であるノードアドレス集合を N_A, (DG, e_B, B) の解であるノードアドレス集合を N_B とする。$e_A = e_B = e$

の場合、DGに属するすべての文書中の文書要素eに対応するノードアドレス集合をN_Uとすると、

- $(DG, e, A \vee B)$の解は$N_A \cup N_B$
- $(DG, e, A \wedge B)$の解は$N_A \cap N_B$
- $(DG, e, \neg A)$の解は$N_U - N_A$
- $(DG, e, (A))$の解はN_A

$e_A \neq e_B$の場合は3.2節と同様の方法で意味を与える。

5 実装

以上の議論を基に、XML文書および文書群に対する検索システムを実装した。その構成を図4に示す。XMLパーザとしてIBM社のXML Parser for Java 2.0.15を使用し、実装はJava言語(JDK 1.2.2)で行った。

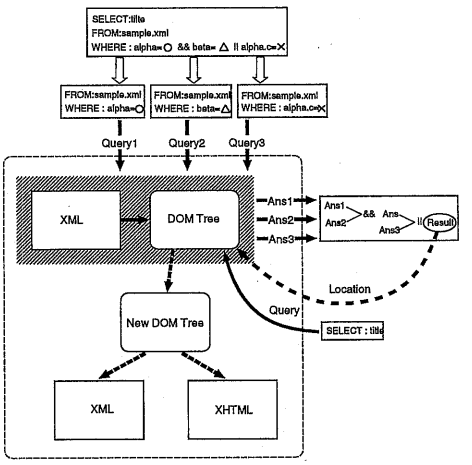


図4: システムの全体像

5.1 質問言語

これまで、半構造化文書やXML文書に対する質問言語はいくつも提案されてきている[1,4,7]。しかし、今回検索システムを実装するにあたり、我々は独自にSQL風の質問言語を定義し実装を行った。これは、実装の初期段階で利用を検討していたXML-QL [7]の処理系が文書要素の順序の扱いなどの点で我々の要求を満たさなかったこと[4]、現在W3Cにおいて XMLの質問言語の規格策定が進められており、今後の方針が不透明であることなどによる。

たとえば、質問

$$(\{book.xml\}, title, \\ match(year, 1999) \vee match(author, 鈴木))$$

を次のように記述する。

```
SELECT : title
FROM : book.xml
WHERE : year=1999 || author=鈴木
```

XMLに対する質問言語の規格が制定されれば、それを利用する方向でシステムを再実装することを考えている。

5.2 システムの構成

今回実装を行った検索システムは、5.1章で述べた質問言語で書かれた質問を処理し、検索結果をXML形式またはXHTML [8]形式に整形して出力する。システムは、質問構文解析部、基本検索条件処理部、検索結果合成部、出力部の4つから構成されている。

質問構文解析部では、与えられた質問を構文解析し、複合検索条件に関する構文木を構成して基本検索条件による質問に分解する。その後、個々の基本検索条件による質問を基本検索条件処理部に渡す。

基本検索条件処理部では、質問構文解析部から渡された質問を処理し、結果となるノードアドレス集合を検索結果合成部に渡す。本実装では、XMLの内部表現としてDOM(Document Object Model)を使用しており、ノードアドレスは実際にはDOM上でのノードの位置を表す内部表現になっている。具体的には、DOM木の根から対応するDOMノードまでのパス上の各ノードについて、同一の深さのDOMノード中で何番目に位置しているかという値を求め、それらを根から順に構成したリストになっている。

検索結果合成部では、質問構文解析時に作成された複合条件に関する構文木、および基本検

[4] [7]では、文書要素の順序はindex variableで扱うと書かれているが、現在提供されているサンプル実装ではこの機能は実現されていなかった。

索条件処理部から得られるノードアドレス集合を基に、最終的な解となるノードアドレス (の内部表現) 集合を求め、出力部に渡す。

出力部では、解となるノードアドレス集合と、基本検索条件処理部で構成したDOM木を基に、出力となるXMLまたはXHTMLのためのDOMを新たに構成し、結果を出力する。

現在、単一文書に対する検索機能については実装が完了している。文書群に対する検索機能、およびデータベース中のXML文書[5]に対する検索機能は実装中である。

6 おわりに

本稿では、複合条件検索における結果の統合において最小の共通ノードを基礎とした意味を与えたが、たとえば、文書要素が再帰的な構造を持っている場合は、ユーザの予期しない結果集合の縮退が起こってしまう可能性がある。このような文書要素の異種性に対する検討をより詳細に行いたいと考えている。

今回提案した文書群に対する検索手法は、分割された文書やメモなどが付加されたことによって生じた文書群などについては有用である。しかし、生成など、ほかの関連に基づく文書群にも有用であるかは明らかではない。

また、文書の集合を基礎にして文書群の定義を行っている。したがって、文書群自体を階層的に構成することが可能である。必要性を含め、定義や検索機能を検討していきたい。

実装面からは、今回のシステムは、これまでに異種文書管理の研究の一貫として実装してきたシステム [10] とは連携していない。これらの連携・統合が実装面での課題である。また、ユーザの望む検索結果と実際の結果がどの程度一致するのかという点からの評価、よりよい検索結果を得るための質問改善過程の支援も今後の課題である。

[5]データベースはeXcelon社のObjectStore PSE Proを使用することを予定している。データベース中にはDOM形式で格納する。

謝辞

さまざまな議論をいただく岡山県立大学横田研究室の皆様、および岡山理科大学総合情報学部の劉渤江助教授に感謝します。

参考文献

[1] S. Abiteboul, D. Quass, J. McHugh, J. Widom, and J. Wiener. The Lorel Query Language for Semistructured Data. *International Journal on Digital Libraries*, Vol. 1, No. 1, pp. 68–88, Apr. 1997.

[2] T. Bray, J. Paoli, and C. M. Sperberg-McQueen. Extensible Markup Language (XML) 1.0. W3C Recommendation REC-xml-19980210, Feb. 1998. available from http://www.w3.org/.

[3] P. V. Brion and A. Malhotra. XML Schema Part 2: Datatypes. http://www.w3.org/TR/xmlschema-2/, Apr. 2000.

[4] P. Buneman, S. Davidson, G. Hillebrand, and D. Suciu. A Query Language and Optimization Techniques for Unstructured Data. In *Proc. ACM SIGMOD Intl. Conf. Management of Data*, pp. 505–516, 1996.

[5] J. Clark. XSL Transformations Version 1.0. http://www.w3.org/TR/xslt, Nov. 1999.

[6] S. Derose, E. Maler, D. Orchard, and B. Trafford. XML Linking Language (XLink). http://www.w3.org/TR/xlink, Feb. 2000.

[7] A. Deutsch, M. Fernandez, D. Florescu, A. Levy, and D. Suciu. XML-QL: A Query Language for XML. http://www.w3.org/TR/1998/NOTE-xml-ql-19980819, Aug. 1998.

[8] S. Pemberton, et al. XHTML[tm] 1.0: The Extensible HyperText Markup Language – A Reformulation of HTML 4 in XML 1.0. http://www.w3.org/TR/xhtml1, Dec. 1999.

[9] 本行, 白木, 田槙, 国島, 横田. 文学データベースのためのプロトタイプシステムの実装. 第57回情報処理学会全国大会予稿集, Oct. 1998.

[10] 國島, 横田, 白木, 劉. XMLリンク機能による異種文書の統合方式. 情報処理学会データベース研究会研究会資料, No. DBS117-7, pp. 39–45, Jan. 1999.

[11] 田槙, 本行, 白木, 國島, 横田. 文学データベースのための索引・検索機能の拡張. 平成10年度電気・情報関連学会中国支部連合大会予稿集, pp. 113–114, Oct. 1998.