社団法人 情報処理学会 研究報告
IPSJ SIG Technical Report

2003－DBS－131 (5)
2003／7／16

XMLの意味情報の作成支援とその応用

古川夏子† 森嶋厚行††

XMLはインターネットにおけるデータ交換のためのデファクトスタンダードとしての地位を既に確立している．現実に作成・管理されるXMLデータの量が劇的に増加するにしたがい，情報統合やデータ再構成の問題の重要性が増大するのは明らかである．我々はリバースエンジニアリング的な発想で，既存のXMLデータの統合利用や再構成の支援を行う問題に焦点を当てる．具体的には，既存のXMLスキーマやインスタンスから高水準の意味情報を作成する手法を提案する．この意味情報はXMLデータの統合利用や再構成に利用可能である．本稿では提案手法およびその応用について説明する．

A System for Extracting Semantic Information from XML Data and Its Applications

Natsuko Furukawa† and Atsuyuki Morishima ††

XML has become the de facto standard for data interchange on the Internet. As the amount of XML data grows, there are increased needs for information integration and restructuring. We address the problem of supporting such processes by "reverse engineering" XML data: We propose a framework for extracting high-level semantic information from XML schemas and instances. The extracted semantic information can be effectively used for integration and/or restructuring of XML data. This paper explains the framework and its applications.

1. はじめに

XMLはインターネットにおけるデータ交換のためのデファクトスタンダードとしての地位を既に確立している．現実に作成・管理されるXMLデータの量が劇的に増加するにしたがい，次のような問題の重要性が増すことは明らかである．

(1) 情報統合の問題: 異なる組織が類似の種類のXMLデータを個別に持っているが，それぞれスキーマ(DTDやXML Schemaなど)が異なる．これらを統合したい．

(2) データ再構成の問題: 情報システムの移行や拡張にともない，XMLのスキーマを変更したい．また新旧スキーマ間のマッピングとそれらの間でデータの移行を行いたい．

以上の問題はリレーショナルデータベースやオブジェクト指向データベースなどの文脈でも認識され，研究が行われてきた[6]．しかし，XMLデータの場合には一般にこれらより問題が複雑になる．その理由は，(1) 正規化の理論やスキーマの設計論が確立していないこと (2) 要素属性や参照など多様な構成子を持つこと，などの要因によって同種の情報を表すスキーマのバリエーションがより多くなるからである．

例えば，図1のUMLクラス図で表される情報をXMLで表現する場合を考える．このとき，例えば図2(a)(b)のようなスキーマで表現することが可能である．これらは同じ情報をかなり異なる方法で表現している．たとえば，(a)は教員と学生を異なる型の要素で区別しているのに対し，(b)では属性を用いて区別している．

我々はリバースエンジニアリング的な発想で，既存のXMLデータの統合や再構成の支援を行うフレームワークの問題に焦点を当てる(図3)．本フレームワークでは，既存のXMLスキーマやインスタンスから図1のような意味情報を作成する．情報統合ではこのような意味情報が有効に利用できることが知られている[2)4)7)]．我々は，このようなフレームワークの鍵となる技術としてCX図(Class-XML mapping Diagram)を提案する．CX図は，XMLの意味情報がXMLインスタンスにどのようにマッピングされているかを表

† 芝浦工業大学大学院工学研究科
Grad. Sch. of Eng., Shibaura Inst. of Tech.
†† 筑波大学知的コミュニティ基盤研究センター
RCKC, Univ. of Tsukuba

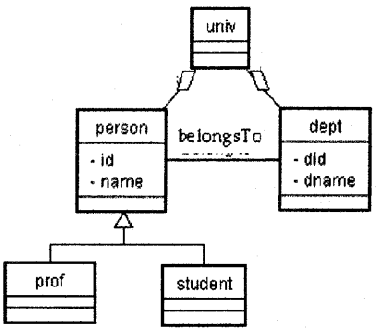


図 1 UML クラス図の例
Fig. 1 A UML class diagram

```
univ=(dept*)
dept=(did, dname, people)
people=(prof*,student*)
prof=(pid,name)
student=(sid, name)
```
(a)
```
univ=(dept*,person*)
dept=(@ID:did, dname)
person=(pid, @type, name, @IDREF:did)
```
(b)

図 2 図 1 のクラス図のための 2 種類のスキーマ
Fig. 2 Two Different Schemas for the Class Diagram in Fig. 1

す表現である．また，XML とのマッピング情報を維持したまま CX 図の操作を行うオペレータが定義されている．本稿では，CX 図を用いた XML の意味情報作成支援システムおよびその応用について説明する．

2. Class-XML mapping Diagram

CX 図の構成要素は UML のクラス図のサブセットであり，クラス，関係 (関連，汎化)，属性から構成される．集約は名前 has を持つ関連として表現される．これらの各構成要素には，XSBE (XML-Semantics Binding Expression) が付加されている．これは，各構成要素と XML インスタンスの関係を表現する式である．図 4 は CX 図の例である．

図 4(a)，(b) はそれぞれ図 2(a)，(b) のスキーマで規定される XML データと意味情報を結びつける CX 図である．XSBE は各クラス，関連，属性に付加される☆．本稿では，クラスの XSBE はクラスの脇に記述する．関連の XSBE は，関連の横に並べて記述している．属性の XSBE は，各属性の横に記述している．UML のクラス図では汎化においては上位クラスからの継承される属性を省略するが，CX 図では全て記述する．

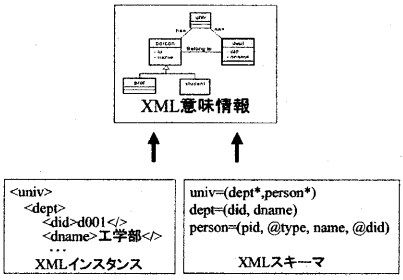


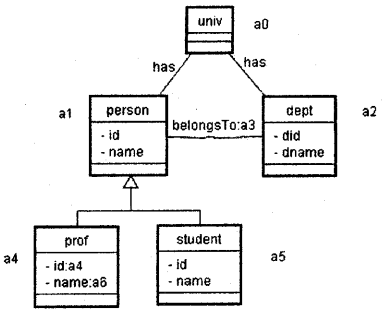
図 3 XML のリバースエンジニアリング
Fig. 3 Reverse engineering of XML data

XSBE は図 5 の構文を持ち，かつ付加対象の種類 (クラス，関連，属性など) によって，記述に制約がある．各 XSBE の結果はリレーションとみなせる．例えば，図 4(a) の関連 belongsTo の XSBE a_3 が表すリレーション ($R(a_3)$ と表記) は $\$k_1, \$k_2$ に対応する属性を持つリレーションである☆☆．XSBE の union 演算は各 XSBE に対応するリレーションが和両立で無ければならない．

以下，それぞれの制約について説明する．

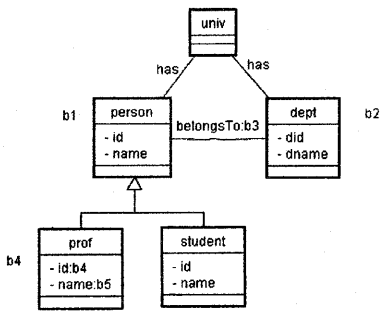
クラスの XSBE. (制約 1) 変数として，ID 変数 ($\$\mathtt{k}_j$ もしくは $\$\mathtt{i}_j$) を唯一つ持つ．ここで $\$\mathtt{k}_j$ はこのクラスのインスタンスのキー値に束縛される変数である．図 4(a) において，クラス prof の XSBE は ID 変数 $\$k_2$ を持つ．また $\$\mathtt{i}_j$ は，このクラスのインスタンスの**アイデンティティ要素**に束縛される変数である．ここでアイデンティティ要素とは，クラスの各インスタンスと一対一対応する要素である．例えば，図 4 において，a0 の $\$i_1$ はクラス univ のアイデンティティ要素 univ に束縛される．(制約 2) リレーション R のリレーション属性のうち，ID 変数に対応するものを $ID(R)$ と表記する．このとき，あるクラスの XSBE b から求められるリレーション $\pi_{ID(R(b))}(R(b))$ は，そのクラスの

☆ 図では一部の構成要素に対してのみ XSBE を記述している．
☆☆ XSBE に union 演算がある場合は最初のオペランドの変数名がリレーションの属性名となる．



a0: $\$i_1$ in /univ
a1: $a4$ union $a5$
a2: $\$k_1$ in /univ/dept/did
a3: $a7$ union $a8$
a4: $\$k_2$ in /univ/dept/people/prof/pid
a5: $\$k_3$ in /univ/dept/people/student/sid
a6: $\$x_1$ in /univ/dept/people/prof,$\$v_1$ in $\$x_1$/name, $\$k_2$ in $\$x_1$/pid
a7: $\$x_2$ in /univ/dept, $\$k_1$ in $\$x_2$/did, $\$k_2$ in $\$x_2$/people/prof/pid
a8: $\$x_3$ in /univ/dept, $\$k_1$ in $\$x_3$/did, $\$k_3$ in $\$x_3$/people/student/sid

(a)



b1: $\$k_1$ in /univ/person/pid
b2: $\$k_2$ in /univ/dept/did
b3: $\$x_1$ in /univ/person, $\$k_1$ in $\$x_1$/pid, $\$k_2$ in univ/dept/@did where $\$x_1$ /@did = $\$k_2$
b4: $\$x_2$ in /univ/person, $\$k_3$ in $\$x_2$/pid, where $\$x_2$/@type = prof
b5: $\$x_3$ in /univ/person, $\$v_1$ in $\$x_3$/name, $\$k_3$ in $\$x_3$/pid, where $\$x_3$/@type = prof

(b)

図 4 CX 図の例
Fig. 4 Examples of CX diagrams

インスタンスすべての ID 値 (キー値もしくはアイデンティティ要素) の集合となる．例えば，図 4(a) においてクラス prof のリレーション $\pi_{\$k_2}(R(a4))$ は，pid の集合となる．

関連の XSBE．(制約 1) 変数として，接続されたクラス全ての ID 変数を持つ．(制約 2) ある関連の XSBE b から求められるリレーション $\pi_{ID(R(b))}(R(b))$ の各

```
<xsbe>::=  (<xsbe>) union (<xsbe>)
        |  <expr> { , <expr> } [ <whereExpr> ]
<expr>::= <var> in [ <var> ] <locationPath>
<whereExpr>::= where <predicates>
```

図 5 XSBE の構文
Fig. 5 XSBE's syntax

タプルは，接続されたクラス間の関連のインスタンスである．例えば，図 4(a) の関連 belongsTo のリレーションは，$\pi_{\$k_2,\$k_1}(R(a3))$ となる．このリレーションにタプル (k,l) が含まれる時，k で表される person が l で表される dept に所属することを表す．

属性の XSBE．(制約 1) 変数として，属性が所属するクラスの ID 変数と，値変数 ($\mathbf{\$v}_j$) を一つ持つ．(制約 2) リレーション R のリレーション属性のうち，値変数に対応するものを $V(R)$ と表記する．このとき，クラス c の属性 a の XSBE b から求められるリレーション $\pi_{ID(R(b)),V(R(b))}(R(b))$ は，(c の ID 値，a の値) の集合である．例えば，図 4(a) のクラス prof の属性 name は，ID 変数$\$k_2$ と値変数$\$v_1$ を持ち，XSBE a6 のリレーションは $\pi_{\$k_2,\$v_1}(R(a6))$ となる．このリレーションにタプル (m,n) が含まれてる時，m で表される prof の name が n である事を表す．キー属性の場合は，$ID(R(b)) = V(R(b))$ となるため，単項リレーションに対応するリレーションは $\pi_{ID(R(b))}(R(b))$ とする．

CX 図は次のように定義される．

定義．CX 図は 4 つ組 $(N, R, name, b)$ である．ここで $N = C \cup A$，C はクラス集合，A は属性集合，$R : N \times N$ は関係集合である．$name : (N \cup R) \to String$ はクラス，属性，および関係の名前をあらわす関数である．また，$b : (N \cup R) \to BE$ は各ノードおよび関連に付随した XSBE を求める関数である．BE は図 5 で規定された XSBE の集合である．

3. オペレータ

本節では，CX 図のオペレータについて説明する (図 6)．これらを適用すると，CX 図とそれに付随する XSBE が変更される．

以下の説明では，$cxd = (N, R, b, name), N = C \cup A$ であると仮定する．

3.1 クラスの操作

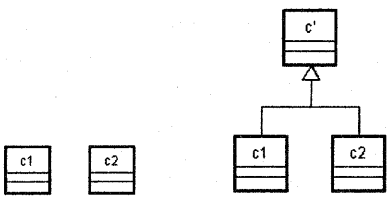
SupC 演算．与えられた cxd に関して，$c_1, c_2 \in C$ はクラスであるとする．$\mathbf{SupC}_{(c_1,c_2)\to c'}(cxd)$ は，cxd に $\mathbf{R}(c') = \mathbf{R}(c_1) \cup \mathbf{R}(c_2)$ となるようなクラスを追加する演算である (図 7)．形式的には，$(N', R', b', name') =$

主な目的	演算子 (*は複合演算子)
クラスの操作	SubC, SupC, RC
関連の操作	CR, UR, NR, RR
属性の操作	CA, RA
その他	Key, RN, A2R, C2A*

図 6 オペレーター覧
Fig. 6 Operators

$\mathbf{SupC}_{c_1,c_2}(cxd)$ と定義する．ただし，

$$
\begin{aligned}
N' &= C' \cup A' \\
C' &= C \cup \{c'\} \\
A' &= A \\
R' &= R \cup \{(c_1, c'), (c_2, c')\} \\
name'(x) &= \begin{cases} \epsilon & (x = c') \\ \texttt{isa} & (x = (c_1, c') \text{ or } x = (c_2, c')) \\ name(x) & otherwise \end{cases} \\
b'(x) &= \begin{cases} b(c_1)\ \texttt{union}\ b(c_2) & (x = c') \\ b(x) & otherwise \end{cases}
\end{aligned}
$$



(a) オペレータ適用前　　(b) オペレータ適用後

図 7 SupC 演算例
Fig. 7 SupC operation

SubC 演算．与えられた cxd に関して $c \in C$ はクラスであるとする．また，p を条件式とする．$\mathbf{SubC}_{(c,p)\to c'}(cxd)$ は，cxd に $\mathbf{R}(c') = \sigma_p(\mathbf{R}(c))$ となるようなクラスを追加する演算である．

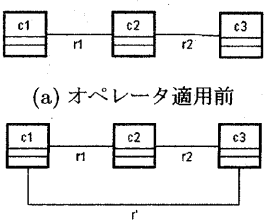
RC 演算．与えられた cxd に関して $c \in C$ はクラスであるとする．$\mathbf{RC}_c(cxd)$ は，cxd から c および c に接続された関連および属性を除去する演算である．

3.2 関連の操作

CR 演算．与えられた cxd に関して，$r_1 = (c_1, c_2), r_2 = (c_2, c_3) \in R$ は関連であるとする．$\mathbf{CR}_{(r_1,r_2)\to r'}(cxd)$ は，cxd に $\mathbf{R}(r') = \pi_{\neg common(\mathbf{R}(r_1),\mathbf{R}(r_2))}(\mathbf{R}(r_1) \bowtie \mathbf{R}(r_2))$ となるような関連を追加する演算である (図 8)．ここで，$\neg common(R, S)$ はリレーション R, S の属性のうち，R, S の双方には現れないものである．形式的には，$(N', R', b', name') = \mathbf{CR}_{(r_1,r_2)\to r'}(cxd)$ と定義される．ただし，

$$
\begin{aligned}
N' &= C \cup A \\
R' &= R \cup \{(c_1, c_3)\} \\
name'(x) &= \begin{cases} \epsilon & (x = (c_1, c_3)) \\ name(x) & otherwise \end{cases} \\
b'(x) &= \begin{cases} Join(b(c_1), b(c_2), p) & (x = (c_1, c_3)) \\ b(x) & otherwise \end{cases}
\end{aligned}
$$

ここで，$Join(b_1, b_2, p)$ は XSBE b_1, b_2 を条件 p で結合する式である．p は，$R(c_1)$ と $R(c_2)$ に共通の ID 変数 k_i に関する $K_i = K_i'$ の並びである．



(a) オペレータ適用前

(b) オペレータ適用後

図 8 CR 演算例
Fig. 8 CR operation

UR 演算．与えられた cxd に関して $c_1, c_2 \in C$ はクラスであり，s は関連の名前を表す文字列とする．$\mathbf{UR}_{s\to(c_1,c_2,r')}(cxd)$ は，cxd に $\mathbf{R}(r') = \cup_{r_i \in R_s} \mathbf{R}(r_i)$ となるような関連 r' を追加する演算である．ここで，R_s は，名前に s を持つ関連 $r_s = (c_k, c_l)$ の集合である．c_k は，c_1 もしくはそのサブクラスであり，c_l は c_2 もしくはそのサブクラスである．

NR 演算．与えられた cxd に関して $c \in C$ はクラスであるとする．$\mathbf{NR}_{c\to r'}(cxd)$ は，c から n 項関連 r' を作成し cxd に追加する (図 9)．ここで，$\mathbf{R}(r') = \pi_{\neg common(\mathbf{R}(r_i))}(\bowtie_{r_i \in R_c} \mathbf{R}(r_i))$ となる．R_c は c に接続する関連の集合である．

RR 演算．与えられた cxd に関して $r \in R$ とする．$\mathbf{RR}_r(cxd)$ は，cxd から r を除去する演算である．

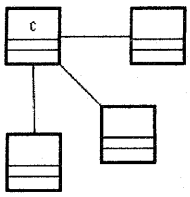
3.3 属性の操作

CA 演算．与えられた cxd に関して $c_1 \in C$ はクラス，$a \in A$ は属性であるとする．a が属するクラス c_2 は，c_1 と 1 対 1 関連 $r \in R$ で接続されていなければならない．この時，$\mathbf{CA}_{a\to(c_1,a')}(cxd)$ は，a を c_1 の属性 a' としてコピーする演算である (図 10)．ここで，$\mathbf{R}(a') = \pi_{K(c_1),V(a)}(\mathbf{R}(r) \bowtie \mathbf{R}(a))$ となる．

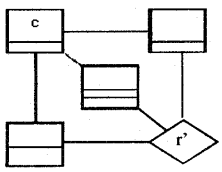
RA 演算．与えられた cxd に関して $a \in A$ とする．$\mathbf{RA}_a(cxd)$ は，cxd から a を除去する演算である．

3.4 そ の 他

Key 演算．与えられた cxd に関して，$c \in C$ はクラス，$a \in A$ は c の属性であるとする．$\mathbf{Key}_{c,a}(cxd)$ は，

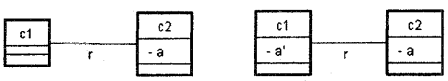


(a) オペレータ適用前



(b) オペレータ適用後

図 9 NR 演算例
Fig. 9 NR operation



(a) オペレータ適用前　(b) オペレータ適用後

図 10 CA 演算例
Fig. 10 CA operation

c の ID 変数を$\$i_j$ から$\$k_j$ に変更する．ここで$\$k_j$ は a の属性値 v に束縛される．この演算はクラス図の構造には影響を与えないが，c および c に直接接続された全ての関連・属性の XSBE に影響を及ぼすため，やや複雑な操作となる．演算結果において，元の a に対応するものを a'，c に対応するものを c'，c に接続された属性 a_i に対応するものを a'_i，c に接続された関連 r_i に対応するものを r'_i とする．このとき，結果の XSBE が計算するリレーションはそれぞれ次のようになる．

- $\mathbf{R}(a') = \pi_{V(a)}(\mathbf{R}(a))$
- $\mathbf{R}(c') = \mathbf{R}(\mathbf{a}')$
- $\mathbf{R}(a'_i) = \pi_{\neg common(\mathbf{R}(a),\mathbf{R}(a'_i))}(\mathbf{R}(a) \bowtie \mathbf{R}(a_i))$
- $\mathbf{R}(r'_i) = \pi_{\neg common(\mathbf{R}(a),\mathbf{R}(r'_i))}(\mathbf{R}(a) \bowtie \mathbf{R}(r_i))$

RN 演算．与えられた cxd に関して，$x \in N \cup R$ はクラス，属性，および関連のいずれかであるとする．また s は文字列であるとする．この時，$\mathbf{RN}_{x,s}(cxd)$ は x の名前を s に変更する．

A2R 演算．属性から関連を作る演算である (図 11)．与えられた cxd に関して，$c_1, c_2 \in C$ はクラス，$a_1, a_2 \in A$ はそれぞれ c_1, c_2 の属性であるとする．(すなわち，$(c_1, a_1), (c_2, a_2) \in R$ である．) この時，$\mathbf{A2R}_{(c_1,a_1),(c_2,a_2)\to r'}(cxd)$ は新たな関連 r' を cxd に追加する．ただし，$\mathbf{R}(r') = \pi_{K(a_1),K(a_2)}(\mathbf{R}(a_1) \bowtie_{V(a_1)=V(a_2)} \mathbf{R}(a_2))$ となる．

(a) オペレータ適用前　(b) オペレータ適用後

図 11 A2R の演算例
Fig. 11 A2R operation

C2A 演算．クラス c をクラス c' の属性 a' に変換する複合演算である．$c, c' \in C$，$a \in A$，$(c, a), (c, c') \in R$，$name((c, c')) \neq \mathtt{isa}$，$name(c) = s$ とする．このとき，$\mathbf{C2A}_{c\to(c',a')}(cxd) = \mathbf{RC}_c(\mathbf{CA}_{a\to(c',a')}(\mathbf{RN}_{a,s}(cxd)))$ と定義する．

4. 意味情報の作成支援

本章では，本システムを用いた意味情報の作成について説明する．意味情報の作成は次の 3 ステップから構成される．(ステップ 1) システムが XML のスキーマを基に単純な規則を用いてデフォルトの CX 図を作成する．(ステップ 2) システムがヒューリスティクスと 3 章で説明したオペレータを用いてステップ 1 で作成したデフォルトの CX 図を変更する．(ステップ 3) 利用者がオペレータを用いてステップ 2 の結果を変更する．

4.1 ステップ 1

このステップで図 2(b) から CX 図を作成した結果を図 12 に示す．図からわかるように，XML の要素，要素属性がそれぞれ CX 図のクラス，クラス属性に対応する．以下で，生成規則の詳細を説明する．

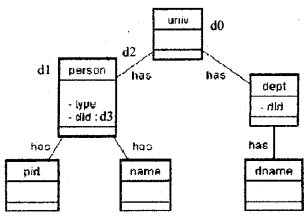
クラス生成規則. スキーマに現れる各要素毎に，一つのクラスを作成する．クラスの XSBE は，$\$i_j$ `in` p となる．ここで，p はスキーマのルート要素からそのクラスに対応する要素までのパスに対応する XPath のロケーションパス式である．

例えば，図 12 のクラス person は図 2(b) の要素 person に対応するので，クラス person の XSBE は $\$i_1$ `in /univ/person` となる．

属性生成規則. スキーマの要素属性毎にクラス属性を生成する．例えば，図 2(b) の要素 person の要素属性 did は，図 12 のクラス person の属性 did に対応する．要素属性名を $attr$ とすると，対応するクラス属性の XSBE は，そのクラス属性が所属するクラス

の XBSE $\$i_j$ in p に，$\$v_j$ in $\$i_j$/@$attr$ を追加したものである．例えば，図 12 のクラス person の属性 did の XBSE は $\$i_1$ in /univ/person, $\$v_1$ in $\$i_1$/@did となる．

関連生成規則. 要素間に親子関係があるとき，それらの要素に対応したクラス間を二項関連で結び，関連名を has とする．その関連の XSBE は，親に相当するクラスの XBSE $\$i_j$ in p に$\$i_k$ in $\$i_j$/$cname$ を追加したものである．ここで，$cname$ は子に相当するクラスの名前であり，i_k はそのクラスの XSBE の ID 変数である．例えば，図 2(b) の要素 univ と person 間は親子関係であるため，図 12 のクラス univ と person 間の関連の XSBE は，$\$i_0$ in /univ,$\$i_1$ in $\$i_0$/person となる．



d_0:$\$i_0$ in /univ
d_1:$\$i_1$ in /univ/person
d_2:$\$i_0$ in /univ,$\$i_1$ in $\$i_0$/person
d_3:$\$i_1$ in /univ/person,$\$v_1$ in $\$i_1$/@did

図 12 ステップ 1 の出力
Fig. 12 An output from Step1

4.2 ステップ 2

このステップでは，ステップ 1 の結果をヒューリスティクスを利用して変更する．変更には 3 章で定義したオペレータを利用する．ヒューリスティクスの一部を次に示す．

- 同じ名前を持つ ID 属性と IDREF 属性は A2R オペレータを用いて関連に変更する．
- テキストノードしか持たないクラスは C2A オペレータを用いて属性に変換する．
- あるクラス c に has で結ばれた全ての子クラスが，いずれもクラス属性をもたずかつ関連を唯一つ持つとき，c を NR オペレータを用いて n 項関連に変換する．
- ID 属性を持つクラス c は，Key オペレータを用いて c の ID 変数をキー変数と変更する．

ステップ 2 では図 12 の $cxd1$ に対して次の変換を行う．

$$cxd2 = (Key_{c_2,a_6}(RA_{a_5}(A2R_{(c_1,a_5),(c_2,a_6)\rightarrow r'}(C2A_{c_5\rightarrow(c_2,a_3)}(C2A_{c_4\rightarrow(c_1,a_2)}(C2A_{c_3\rightarrow(c_1,a_1)}(cxd1)))))))$$

ただし，c_1 は person クラス，c_2 は dept クラス，c_3 は pid クラス，c_4 は name クラス，c_5 は dname クラス，a_5 は c_1 の did 属性，a_6 は c_2 の did 属性である．

4.3 ステップ 3

このステップでは，ステップ 2 の結果に対して利用者がオペレータを適用し，最終的な CX 図を得る．ステップ 2 の結果から図 4(b) を求めるためには，次の演算を行う必要がある．

$$cxd3 = (RN_{r',"belongsTo"}(RN_{c_7,"student"}(RN_{c_6,"prof"}(SubC_{c_1\rightarrow(c_7,p_2)}(SubC_{c_1\rightarrow(c_6,p_1)}(RN_{a_1,"id"}(Key_{c_1,a_1}(RA_{a_4}(cxd2)))))))))$$

ただし，p_1 は，$\$k_1$/../@type="prof" p_2 は，$\$k_1$/../@type = "student"，$a_4$ は c_1 の type 属性である．

5. 応 用 例

1 章で述べたように，XML の意味情報は様々な用途に利用可能である．本章では，CX 図を用いた具体的な応用例として，異なるスキーマ間で XML のマッピングを行うための XQuery 問合せの作成支援システム[8] の概要を示す．

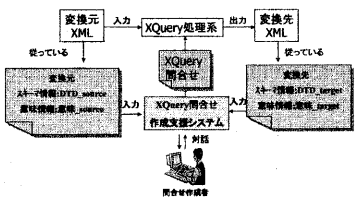


図 13 システム概要
Fig. 13 System's overview

図 13 は，本システムを用いた問合せ作成支援の概要を示す．変換元のスキーマ情報及び意味情報をそれぞれ DTD_source 及び意味_source で表し，変換先のスキーマ情報及び意味情報を DTD_target 及び意味_target で表す．ここで，意味_x は CX 図を用いて記述されているとする．本システムは DTD_source 及び意味_source に従った XML データを入力とし，DTD_target 及び意味_target に従った XML データを出力する問合せ作成の支援を行う．

本システム利用の流れは次の通りである．(1) スキー

マ情報および意味情報を入力する．(2) 利用者は CX 図のオペレータを用いて意味_source と意味_taget の対応付けを行う．(3) システムが XQuery 問合せの自動作成を行う．自動的に作成できない部分は空白としたまま残す．(4) 利用者はシステムが作成した XQuery 問合せの空白部分を埋める．

5.1 システムによる XQuery 問合せ作成処理

ここでは例を用いて，本システムにおける XQuery 問合せ自動作成処理の流れ (上記 (3)) を説明する．図 14 は，図 2(a) に従った XML インスタンスから，(b) に従った XML インスタンスを出力する問合せの作成処理の流れを示したものである．

本例において，DTD_target の要素 did を出力するような問合せを作成する流れは次のようになる．(1) 要素 did に対応する意味_target 中の構成要素 a を求める．(2)(Step 2) で利用者が指定したマッピング情報を用いて，a に対応する意味_source b を求める．(3)a の XSBE を XQuery 問合せの for 節に出力し，return 節に，その XSBE が持つ ID 変数を出力する．(4) 作成された XQuery 問合せが意味_target 全体の制約を満たすよう，(3) で作成した XQuery 問合せを変更する．

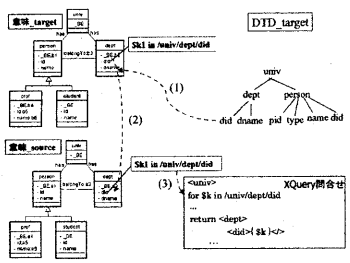


図 14 問合せ作成例
Fig. 14 Query generation example

6. 関連研究

DTD や XML Schema などの XML スキーマ言語から，意味情報を生成する研究はいくつか存在する．Mello らの研究[3)4)] では DTD を Conceptual Schema に変換するための規則を定義している．また，XML のスキーマ言語の一つである XGrammer[5)] では，EER ダイアグラムとの相互変換方法が与えられている．これらが本研究と異なる点は次の通りである．(1) 意味情報の構成要素と XML インスタンス間との関係の記述を明示的に保持しない．(2) 意味情報のオペレータが定義されていない．(3) 作成される意味情報が，与えられたスキーマとほぼ 1 対 1 対応する．

C-Web[1)] では，XSBE と同様，意味情報の構成要素と XML インスタンス間とのマッピングを表すための言語を提供している．しかし言語の表現力が XSBE より小さく，XSBE でしか記述不可能なマッピングが存在する．また，C-Web では意味情報の自動生成については議論していない．

7. おわりに

本稿では，XML データの意味情報を表す表記法，意味情報の作成支援システム，およびその応用について説明した．今後の課題としては，ヒューリスティクスの適用手順の明確化，システムの実装などがある．

謝　辞

ゼミなどでご議論いただきました芝浦工業大学工学部情報工学科古宮誠一教授に感謝いたします．本研究の一部は文部科学省科学研究費補助金 若手研究 (B)(課題番号 15700108) による．

参考文献

1) B. Amann, I. Fundulaki, M. Scholl, C. Beeri, Anne-Marie Vercoustre: Mapping XML Fragments to Community Web Ontologies. WebDB 2001: 97-102
2) S. Bergamaschi, S. Castano, M. Vincini: Semantic Integration of Semistructured and Structured Data Sources. SIGMOD Record 28(1): 54-59 (1999)
3) R. S. Mello, C. A. Heuser: A Rule-Based Conversion of a DTD to a Conceptual Schema. ER 2001: 133-148
4) R. S. Mello, C. A. Heuser: A Bottom-Up Approach for Integration of XML Sources. Workshop on Information Integration on the Web 2001: 118-124
5) M. Mani, D. Lee, R. R. Muntz: Semantic Data Modeling Using XML Schemas. ER 2001: 149-163
6) E. Pitoura, O. A. Bukhres, A. K. Elmagarmid: Object Orientation in Multidatabase Systems. ACM Computing Surveys 27(2): 141-195 (1995)
7) P. F. Patel-Schneider, J. Simeon: The Yin/Yang web: XML syntax and RDF semantics. WWW 2002: 443-453
8) 古川夏子，森嶋厚行「意味情報を用いた XQuery 問合せ作成支援システムの開発」情報処理学会第 65 会全国大会講演論文集 (3),2003 年 3 月．