セキュリティを考慮した RDB の XML 出版の提案

A Proposal of an XML Publishing Considering Security

品川 徳秀† 北川 博之‡
Norihide SHINAGAWA Hiroyuki KITAGAWA

近年，広く利用されつつある XML でのデータ統合において RDB を利用するために，XML ビューを提供する XML 出版技術が開発されてきた．一方，XML データ処理では，ネットワークを介したデータ交換により，複数のサービス提供者を通じて一連の処理が行なわれる事もあり，アクセス制御や部分暗号化，電子署名等によるセキュリティの確保が重要である．本研究では，セキュリティを考慮した効率の良い XML 出版機構の実現を目指す．XPERANTO 方式をベースとし，XQuery で定義された XML ビューに対して定義されたセキュリティポリシーを XML 出版過程において適用する方式を検討する．

XML-based data integration has been used widely. Schemes to publish XML data from RDBs are important to use RDBs in the data integration, and have been developed. In actual XML data processing, data are exchanged through networks, and processed by one or more service providers. Therefore, security mechanisms such as access control, data encryption, and desital signature, are also important. This paper proposes a scheme to enforce efficiently security policies in the cource of XML publishing. The basis of XML publishing is the XPERANTO approach.

1. はじめに

現在，XML は，ネットワークサービスやソフトウェアコンポーネント間のデータ表現や交換，統合処理において重要な技術の一つとなっている．一方で，多くの実データは，技術が高度に成熟され，多くの実績が培われてきた RDBMS で管理されているため，XML 技術を中心としたシステムにおいても RDB の利用機構が必要とされる．その一つとして RDB の XML ビューを提供する XML 出版と呼ばれる技術が有用であり，XPERANTO [1] や SilkRoute [2] をはじめとした様々な研究が行なわれてきた．

また，ネットワーク利用の一般化により，ネットワークを介して利用可能なシステムが構築され，様々なサービスが行なわれている．これに伴い，データの実利用の場面では，アクセス制御や暗号化，電子署名等のセキュリティ機構が重要である．ネットワークを介したオープンシステムでは，異なる組織が運営する複数のサブシステムを連携させて一連の処理が行なわれる．それゆえ，柔軟かつ標準化されたセキュリティ機構を利用する事が望ましく，XML に関しては標準化が行なわれている [3, 4, 5].

従来の XML 出版機構ではセキュリティ機構を統合していないため，より上位の層でセキュリティポリシーを適用する必要がある．始めに XML 出版層で完全な XML ビューを実体化し，上位ミドルウェアでアクセス制御を行なった結果に対して更に XML 暗号化や電子署名といった処理が適用される．このように段階的に XML の処理する方式では，XML のパースとシリアライズ，探索を繰り返し行なう必要があり，非効率的である．

本研究の目的は，XML 出版において，XML ビューに対して定義されたセキュリティポリシーを出版過程で効率良く適用する事にある．本稿では，XML 出版方式として XPERANTO を基本機構とする．セキュリティ機構としては，XACML のようなポリシーベースのアクセス制御方式と，XML Encryption 及び XML Signature による暗号化・電子署名を想定し，これらの適用について検討する．

†正会員, 科学技術振興機構 戦略的創造研究推進事業
筑波大学大学院システム情報工学研究科 siena@kde.cs.tsukuba.ac.jp
‡正会員, 筑波大学大学院システム情報工学研究科
筑波大学計算科学研究センター kitagawa@cs.tsukuba.ac.jp

department		
did	dname	address
D001	人事	addr-1
D002	経理	addr-2
D002	研究開発	addr-3
D003	営業	addr-4
...	...	...

employee				
pid	pname	did	jobcode	salarycode
P0001	Jack	D001	4 (manager)	M-13
P0002	Michael	D003	3 (vice manager)	V-10
P0003	Robert	D001	1 (regular)	R-11
P0004	Thomas	D002	1 (regular)	R-08
...	...	...	...	...

図 1 リレーション例
Fig. 1 Relational tables.

表 1 XQGM 演算子
XQGM operators.

演算子	機能	演算子	機能
table	表の指定	project	射影 (計算結果の形成)
select	タプルの選択	join	複数入力の結合
groupby	グルーピング	orderby	並び替え
union	複数入力の和集合	unnest	内容リストに展開
view	ビューの指定	function	XQuery 関数の指定

2. XPERANTO

2.1 概要及び例

XPERANTO は RDB の XML 出版ミドルウェアシステムである．RDB は，平坦かつ一様な構造のデフォルト XML ビューとして単純に表現され，これに対する XQuery 問合せによってユーザ XML ビューが定義される．ビューの実体化は，後述するように RDBMS における問合せ能力を有効に利用し，不足する機能をミドルウェア側で処理する事で実現される．

図 1 に示す部門と社員の RDB を例とし，そのデフォルト XML ビューを図 2 に示す．図 3 の XQuery 問合せは，これから部門別の構成員リストを再構成するユーザ XML ビュー定義である．導出される XML ビューの一部を図 4 に例示する．

2.2 問合せ処理

XPERANTO では，問合せを次のように処理する．その際，部分問合せを可能な限り RDBMS 側で処理し，RDBMS 側で処理できない残りの問合せを XPERANTO ミドルウェアで処理する．

1. **問合せ解析**：ビュー定義とユーザ問合せを表 1 の演算子を用いて **XQGM (XML Query Graph Model)** で表現
2. **問合せ変換**：XQGM グラフを再構成し，SQL 問合せとして RDBMS で処理可能な部分と，それ以外の部分に分離
3. **問合せ処理**：分離部分を RDBMS で処理し，更に残り部分をミドルウェアで処理した最終的な XML データを生成

図 3 のビュー問合せを表す XQGM グラフを図 5 に示す．各ノードは XQGM 演算であり，上部枠内は出力属性リストである．実線矢印はノード間の入出力関係を，点線矢印は結合の依存関係を表す．例えば，右側中央の `join(correlated)` とあるノードは，`employee` を `department` へ結合する事を意味する．

```
<db>
  <department>
    <row>
      <did>D001</did>
      <dname>
          Personnel</dname>
      <address>
          addr-1</address>
    </row> ...
  </department>
  <empleyee>
    <row>
      <pid>P0001</pid>
      <pname>Jorn</pname>
      <did>D001</did>
      <jobcode>4</jobcode>
      <salarycode>
          M-13</salarycode>
    </row> ...
  </empleyee>
</db>
```

図 2 デフォルト XML ビュー
Fig. 2 Default XML view.

```
create view employee_list as {
  <list>
    for $dep in view("default")/
                db/department/row
    return
      <department did="$dep/did">
        <dname>$dep/dname</dname>
        <address>$dep/address</address>
        <members>
          for $emp in view("default")/db/
              employee/row[did=$dep/did]
          return
            <member pid="$emp/pid">
              <pname>$emp/pname</pname>
              <jobcode>$emp/jobcode
                  </jobcode>
              <salarycode>$emp/salarycode
                  </salarycode>
            </member>
        </members>
      </department>
  </list>
}
```

図 3 ユーザ XML ビュー定義
Fig. 3 View definition.

```
<list>
  <department did="D001">
    <dname>
        Personnel</dname>
    <address>
        addr-1</address>
    <members>
      <member pid="P0001">
        <pname>Jorn</pname>
        <jobcode>4</jobcode>
        <salarycode>
            M-13</salarycode>
      </member>
      ...
    </members>
  </department>
  ...
</list>
```

図 4 ユーザ XML ビュー
Fig. 4 Defined XML view.

表 2 XML 関数
XML functions.

XML 関数	機能	出現可能演算子	SQL2003 対応関数
cr8Elem(Tag,Atts,Clist)	要素名 Tag，属性リスト Atts，内容 Clist の要素生成	project	xmlelement
cr8AttList($A_1,\cdots,A_n$)	属性群から属性リストを生成	project	xmlattriute
cr8Att(Name,Val)	属性名 Name，属性値 Val の属性生成	project	xmlattribute
cr8XMLFragList($C_1,\cdots,C_n$)	要素内容群から XML 断片リストを生成	project	xmlforest, xmlconcat
aggXMLFrags(C)	XML 断片リストを生成する集約関数の適用	groupby	xmlagg
getTagName(Elem)	要素名の取得	project, select	
getAttributes(Elem)	属性リストの取得	project, select	
getContents(Elem)	要素内容の XML 断片リストを取得	project, select	
getAttName(Att)	属性名の取得	project, select	
getAttValue(Att)	属性値の取得	project, select	
isElement(E)	要素か否かを判定	select	
isText(T)	テキストか否かを判定	select	
unnest(List)	ネストを展開	unnest	

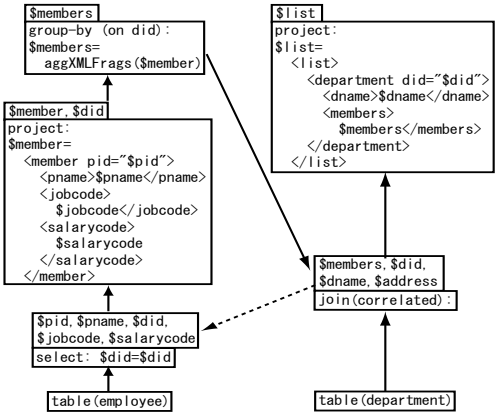


図 5 XGML グラフ
Fig. 5 XQGM graph.

演算子中では，表 2 に示す XML 関数が利用される．XPERANTO 発表時，一般にはこれらを RDBMS で処理できなかったが，SQL 2003 における XML 対応により，SQL 中で同様の関数が利用可能となった [6]．本稿では，SQL 2003 を前提とし，SQL 2003 に相当する関数が存在する XML 関数は，RDBMS 側で処理可能とする．XML 関数の冗長な組合せは，表 3 に挙げた合成規則により，合成結果と等価な XML 関数に置き換える事で除去される．

XQGM グラフを再構成し，RDBMS で処理可能な処理をプッシュダウンし，それ以外の処理をプルアップする事で，SQL で表現可能な部分問合せが分離される．これにより，RDBMS の能力を有効活用しつつ，XML 出版が実現される．

表 3 XML 関数合成規則
XML function composition rules.

Function	Composition Target	Result
getTagName	cr8Elem(Tag,Atts,Clist)	Tag
getAttributes	cr8Elem(Tag,Atts,Clist)	Atts
getContents	cr8Elem(Tag,Atts,Clist)	Clist
getAttName	cr8Att(Name,Val)	Name
getAttValue	cr8Att(Name,Val)	Val
isElement	cr8Elem(Tag,Atts,Clist)	True
isElement	except for cr8Elem	False
isText	PCDATA	True
isText	except for PCDATA	False
unnest	aggXMLFrags(C)	C
unnest	cr8XMLFragList($C_1,\cdots,C_n$)	$C_1 \cup \cdots \cup C_n$
unnest	cr8AttList($A_1,\cdots,A_n$)	$A_1 \cup \cdots \cup A_n$

3. セキュリティ機構

3.1 アクセス制御ポリシー

XML を交換形式としたネットワークを介したシステム連携では，異なる組織によって運営される複数のサービスを通じて一連の処理が行なわれたり，時としてその参加者を事前に限定できなかったり，アクセス制御の方針が複数の組織で決定・運用されたりする．XACML は，このような状況を想定したアクセス制御記述言語である．XACML のセキュリティモデルでは，アクセスの**主体**である利用者は認証システムで認証され，その主体の持つ属性に基づいて**対象**データに関する適切なアクセス制御ポリシー群が選択される．利用者には，これらが適用された結果が提供される．提供システムで果たされねばならない**責務**も指定可能である．

本稿では，XML データ中の対象を XPath で指定し，主体の属性に応じたアクセス可否と，責務としての暗号化・電子署名指示を与える．表 4 にビュー `employee_list` に対する例を挙げる．

表 4 アクセス制御ポリシーとセキュリティ問合せ (SQ)
Access control policies and their security queries.

Policy 1	
説明	人事部の者のみが従業員の給与を知る事ができる
対象	/list/department/members/member/salarycode
主体	人事部に所属しない
可否	拒否
SQ	$\mathtt{Apply}_{\mathtt{list:/list/department/members/member}}[\mathtt{Proj}_{\mathtt{not(salarycode)}}]$
Policy 2	
説明	従業員以外には役職を持つ者のみの名簿しか読めない
対象	/list/department/members/member[jobcode<2]
主体	従業員でない
可否	拒否
SQ	$\mathtt{Apply}_{\mathtt{list:/list/department/members}}[\mathtt{Mask}_{\mathtt{member[jobcode<2]}}]$
Policy 3	
説明	人事部の者は従業員の給与情報にアクセスできるが，それは暗号化されねばならない
対象	/list/department/members/member/salarycode
主体	人事部に所属
可否	許可
責務	encript(/list/department/members/member/salarycode)
SQ	$\mathtt{Apply}_{\mathtt{list:/list/department/members/member}}[\mathtt{Enc}_{\mathtt{salarycode}}]$

表 5 拡張 XQGM 演算子
Extended XQGM operators.

演算子	機能
apply	与えられた演算を指定 XML 断片に適用
mask	条件を満たさない時，指定属性を *null* で置換
enc	指定属性中の XML 断片の暗号化
sign	指定属性中の XML 断片の電子署名

3.2 セキュリティ問合せ

本研究では，XML ビューへ適用されるポリシーをビュー問合せおよびユーザ問合せと統一的に処理するため，適用されるポリシーを**セキュリティ問合せ**として表現する．XML ビューへのポリシーの適用結果を新たな XML ビューとすれば，ユーザ問合せの合成は XPERANTO での通常の問合せ合成と同様である．以下ではビュー問合せとセキュリティ問合せの合成のみに注目する．

セキュリティ問合せを表現するため，幾つかの演算子を導入する (表 5)．Apply は XML 型の属性に対して適用され，XPath で指定された XML 断片に [] 内の演算を適用する高階演算である．Proj 等の属性指定での not(属性) は，指定属性以外の全属性を意味する．Mask は [] 内の条件成立する時，指定属性を *null* とする．Enc，Sign は指定属性の XML 断片の暗号化，電子署名をする．ここでは Mask，Enc，Sign を演算子としたが，ユーザ定義関数や RDBMS の提供する暗号化機能等で実現できる場合，これらは RDBMS で処理可能として問合せ最適化を行なう．尚，暗号化や電子署名で使用される様々なパラメータは表記上省略する．

3.3 演算子のプッシュダウン

ビュー問合せとセキュリティ問合せを合成し，Apply 演算子をプッシュダウンする事で，より内部の小さな XML 断片が適用対象となる．完全にプッシュダウンした場合，Apply は消去され，[] 内の演算を直接適用する XQGM グラフになる．

Apply を完全にプッシュダウンできない時は，RDBMS で処理可能な部分グラフの直上に留め，ミドルウェアで適用対象をパースして，XPath で選択される XML 断片を処理する．この時も，Apply を可能な限りプッシュダウンする事で，適用対象が小さな多数の XML 断片になり，並行処理しやすくなると期待できる．また，Apply で適用される演算が一部のデータを除去する場合，XML 断片が小さくなる等，XML 断片生成コストの抑制に繋がる．

演算子交換規則を図 7 〜図 12 に示す．交換可能条件は，各図の (1), (2) として併記した．但し，条件成立時に必ず演算子を交換する事が望ましいわけではなく，問合せ内容の同一性を保証できる範囲で行なわねばならない．また，交換が処理コストを増加させる場合もあるため，その見積りを行なって適用する必要がある．

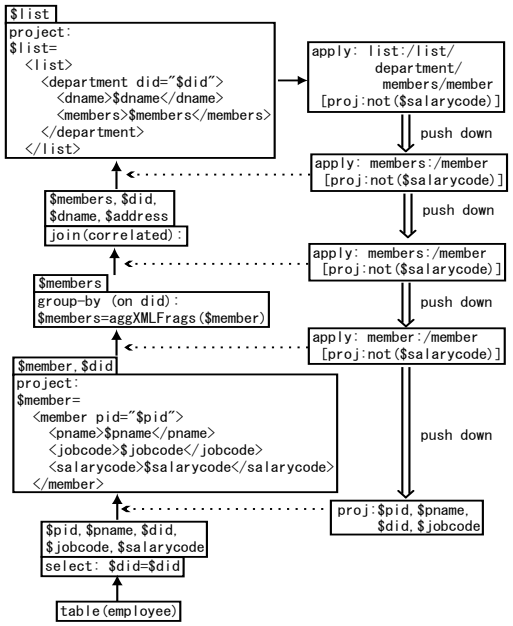


図 6 プッシュダウンの例
Fig. 6 Pushing down Apply.

図 7 の Apply と Proj の交換は，XPath の先頭部分と Proj によって構成される構造が一致する時に行なわれ，Apply の適用対象を XML 断片の内部へ限定する．

図 8 の Apply と Unnest の交換は，Apply 適用対象の断片を大きくしてしまうが，下位の演算子との交換で Apply の対象がより小さくなる場合や，Apply が除去される場合に有用である．

図 10 の Apply と Join の交換は条件で場合分けされる．

(1a) 結合条件が Apply の適用対象 $x に依存しない場合，常に交換可能である．実際には，処理コストを低くできる場合に限り，交換すべきである．タプル数を減少できる場合や，下位の演算子との交換で Apply の適用コストを減少できる場合等が該当する．

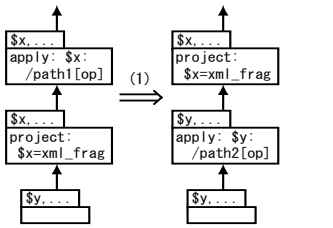
(1b) 結合条件で $x が参照され，Apply の適用演算が Enc か Sign である時，交換可能である．暗号化・電子署名は一般に高コストで，タプル数がコストに大きく影響する．この交換は，暗号化・電子署名は単一の XML 断片を入出力とする全単射関数で，適用前の値での比較と適用後の値での比較は等価な結果を導く事による．一方，XML 暗号化・電子署名の出力は一般に入力より大きく，適用後の値での比較はコストが高くなる事にも注意を要する．

(2) 演算子の交換は適用対象のタプル数を減少させるが，交換が比較コストを高くしたり，比較結果を保存しない演算を適用したりする場合を考える．比較前の値を保持しつつ演算を適用し，保存した値で結合した後に削除するという余分なコストを払っても，全体のコストが低くなるような場合に適用される．

Apply と Select の交換 (図 11) も，Apply と Join の交換と同様である．Union (図 9), GroupBy (図 12), OrderBy (GroupBy と同様) については割愛する．Enc, Sign 等は XML 断片を入力とし，XML 関数による断片生成が事前に必要であるが，SQL2003 の相当する関数を呼べる場合には RDBMS で処理できる．それゆえ，Enc, Sign を RDBMS で実現できる場合，入力の構成に必要なタグ付けの処理とともにプッシュダウンする事もできる．

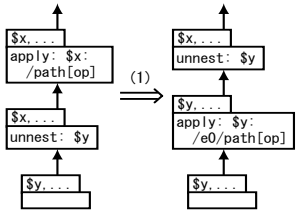
3.4 セキュリティ問合せ合成例

Policy 1 の合成過程を，関与ノードに限定して図 6 に例示する．アクセス制御を実現する Proj を直接適用し，除去対象属性が早期に削除される事が分かる．Policy 2 および Policy 3 の場合も，同様に処理を直接適用する XQGM グラフに再構成できる．セキュ



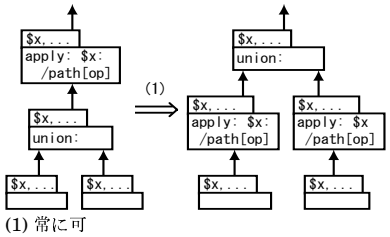
(1) path1 = e1/e2/.../ek/path2
∧ xml_frag = <e1><e2>...<ek>
$y</ek>...</e2></e1>

図 7 Apply-Proj の交換
Fig. 7 Exchange of Apply-Proj.



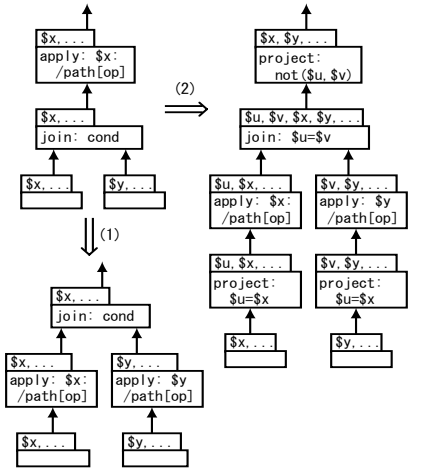
(1) $y = <e0>$x</e0>

図 8 Apply-Unnest の交換
Fig. 8 Exchange of Apply-Unnest.



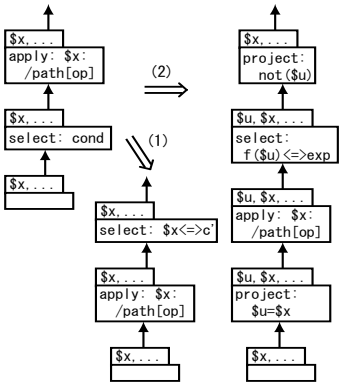
(1) 常に可

図 9 Apply-Union の交換
Fig. 9 Exchange of Apply-Union.



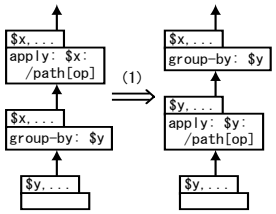
(1) cond: $x に依存しない
∨ cond: $x=$y ∧ op ∈ {enc, sign}
(2) cond: $x=$y ∧ op ∈ {proj, mask, enc, sign}

図 10 Apply-Join の交換
Fig. 10 Exchange of Apply-Join.



(1) cond: $x に依存しない
∨ cond: $x と定数式 c の比較
∧ op ∈ {enc, sign}
(書替え後は op 適用値 c′ と比較)
(2) cond: $x に関する比較
∧ op ∈ {proj, mask, enc, sign}
(書替え後は $u を用いて比較)

図 11 Apply-Select の交換
Fig. 11 Exchange of Apply-Select.



(1) $x が $y の処理結果でない
∨ $x が $y の処理結果
∧ op ∈ {enc, sign}
(2) (図中では割愛)
$x が $y の処理結果
∧ op ∈ {proj, mask, enc, sign}

図 12 Apply-GroupBy の交換
Fig. 12 Exchange of Apply-GroupBy.

リティポリシーの適用を問合せの合成に帰着させたため，複数のポリシーを適用する事も可能である．

4. まとめ

XMLデータ統合における XML 出版およびセキュリティ機構の重要性から，本稿では，セキュリティポリシーを XML 生成後ではなく XML 出版過程で適用し，処理コストを抑える手法を提案した．今後，最適化方式の高度化，コスト見積り手法の定式化，プロトタイプシステムの実装，効率の測定等を行なう予定である．

[謝辞]

本研究の一部は，CREST「自律連合型基盤システムの構築」，科学研究費補助金特定領域研究 (2) (#16016205)，基盤研究 (B) (#15300027) による．

[文献]

[1] M. J. Carey, D. Florescu, Z. G. Ives, Y. Lu, J. Shanmugasundaram, E. J. Shekita, S. N. Subramanian. XPERANTO: Publishing Object-Relational Data as XML, WebDB 2000, pp. 105–110, Dallas, May, 2000.
[2] M. F. Fernandez, W. C. Tan, D. Suciu, SilkRoute: trading between relations and XML, Computer Networks, Vol. 33, No. 1-6, pp. 723–745, 2000.
[3] O. S. Godik, E. T. Moses (eds.). Extensible Access Control Markup Language (XACML) v1.0, http://www.oasis-open.org/specs/index.php#xacmlv1.0, OASIS, Jan, 2003.
[4] D. Eastlake, J. Reagle (eds.). XML Encryption Syntax and Processing, http://www.w3.org/TR/2002/REC-xmlenc-core-20021210/, W3C, Dec, 2002.
[5] D. Eastlake, J. Reagle, D. Solo (eds.). XML-Signature Syntax and Processing, http://www.w3.org/TR/2002/REC-xmldsig-core-20020212/, W3C, Feb, 2002.
[6] A. Eisenberg, J. Milton, SQL/XML is Makinkg Good Progress, SIGMOD Record, Vol. 31, No. 2, Jun, 2002.

品川 徳秀 Norihide SHINAGAWA
科学技術振興機構 戦略的創造研究推進事業 研究員．2001 年筑波大学博士課程工学研究科修了．博士（工学）．構造化文書や異種データ統合，WWW の高度利用および，データセキュリティ等の研究に従事．ACM，日本データベース学会，情報処理学会，各会員．

北川 博之 Hiroyuki KITAGAWA
筑波大学大学院システム情報工学研究科および筑波大学計算科学研究センター教授．1980 年東京大学大学院理学系研究科修了．理学博士（東京大学）．異種情報源統合，文書データベース，WWW 高度利用等の研究に従事．著者「データベースシステム」（昭晃堂），「Unnormalized Relational Data Model」（共著，Springer-Verlag）等．ACM，IEEE-CS，日本データベース学会，情報処理学会，電子情報通信学会，日本ソフトウェア科学会，各会員．