データベースシステム 128－19
（２００２．７．１８）

異種情報源統合における XML 構造統一化手法

小西 一也　鈴木 源吾　堀口 恭太郎　林 孝志　芳西 崇

日本電信電話株式会社 NTT サイバースペース研究所
〒239-0847 神奈川県横須賀市光の丘 1-1
E-mail: {konishi.kazuya, suzuki.gengo, horiguchi.kyotaro, hayashi.takashi, honishi.takashi}@lab.ntt.co.jp

あらまし インターネット環境において，不特定多数の異種情報源に対して XML を共通データモデルとする構造を意識した統一的な検索を実行し，個々の情報源からの検索結果を統一化するニーズが高まっている．このような検索結果の統一変換においては，個々の情報源から抽出される部分木をリストとして扱い，各部分木に関して構造を変換して統一化することが必要である．本稿では，XML を共通データモデルとした不特定多数の異種情報源のためのメディエーションシステム MediPresto/XM における検索結果の統一のための変換：情報源のデータ構造を参照しない宣言 XML 構造変換定義と，基本的構造変換と要素併合による構造統一化について論じる．

キーワード XML，構造変換，異種情報源統合

XML Schema Unification in Heterogeneous Information Sources Integration

Kazuya KONISHI　Gengo SUZUKI　Kyotaro HORIGUCHI
Takashi HAYASHI　and　Takashi HONISHI

NTT Cyber Space Laboratories, NTT Corporation
1-1 Hikarinooka, Yokosuka-shi, Kanagawa, 239-0847 Japan
E-mail: {konishi.kazuya, suzuki.gengo, horiguchi.kyotaro, hayashi.takashi, honishi.takashi}@lab.ntt.co.jp

Abstract The unification of retrieval results from individual information source is expected at Internet, by the unified structural retrieval execution from many and unspecified heterogeneous sources, with XML common data model. This unification transformation is implemented by the subtrees extraction from individual source and the schema transformation of each subtrees. This paper investigates the transformation for unification of retrieve results in our mediation system: MediPresto/XM for many and unspecified heterogeneous sources with XML common data model. This transformation consists of basic schema transformation and elements grouping, and the schema transformation definition is declarative and does not include references to any source schemas.

Keyword XML, Schema Transformation, Heterogeneous Information Sources Integration

1. はじめに

XML は柔軟な表現能力を持ち，構造データおよび半構造データを表現することができる．この特性のために，XML を共通データモデルとする異種情報源統合に対するニーズが高まっている．我々は，メディエーション・アーキテクチャによる異種情報源統合を，XML モデルに基づき実現する技術について，研究開発を行っている[1].

開発中のシステム：MediPresto/XM は，不特定多数の情報源のスキーマやデータに関する異種性を動的に解消することを特徴とする．不特定情報源の統合を実現するためには，各情報源のデータ構造を参照しない構造変換定義に従って，各情報源の構造を統一化する必要がある．

情報源のデータ構造を参照しない構造変換定義は，変換結果構造の宣言的定義により実現される．宣言的定義に従う構造変換処理は，変換結果構造に対応する情報源部分構造の探索，探索部分構造に対応する部分木の抽出，抽出部分木の結果構造への変換という手順で実行する．ここで，部分木は情報源構造に依存して抽出されるため，構造統一化のためには異なる部分木内の要素同士の併合が必要になる．また，各情報源のデータ形式の違いを考慮すると，同一性以外の判断基準による要素併合が必要になる．

以上より，XML を共通データモデルとする不特定多数の異種情報源統合における XML 構造統一化の機能要件として，以下が挙げられる．

(1) 変換結果構造に対応する情報源部分木構造の探索
(2) 異なる抽出部分木内の要素同士の併合
(3) 同一性以外の判断基準による要素同士の併合

本稿では，これらの要件を満たす XML 構造変換を提案する．

2. 異種情報源統合における XML 構造変換の問題点

XML 構造変換定義が可能な標準言語として，XSLT[2]や XQuery[3]などが策定されている．また，半構造データモデルに基づく異種情報源統合システムとして TSIMMIS や MIX などがあり，これらのシステムに対する専用問い合わせ言語として MSL[4]や XMAS[5]などがある．しかしこれらの言語は，不特定多数の異種情報源統合における XML 構造変換定義には適していない．以下に，その理由を述べる．

(1) 変換結果構造に対応する情報源部分木構造の探索ができない．XSLT や XQuery の指定は，情報源のデータ構造の参照が必須である．MSL や XMAS の指定も，情報源やビューの参照が必要になる．不特定多数の異種情報源を統合するとき，統合対象となる情報源の構造は参照できないため，情報源構造を参照しない構造探索機能が必要になる．

(2) 異なる抽出部分木内の要素同士を併合できない．統一構造部分木のリストを統合検索結果とすること考えると，MSL や XMAS では，宣言的構造定義に従った部分木のリストを得ることができる．しかし，構造変換は情報源構造に依存して抽出される各部分木に適用され，異なる抽出部分木内の要素同士の併合はできない．これは不特定多数情報源の検索結果構造統一化の大きな制約となる．

(3) 同一性以外の判断基準による要素同士の併合ができない．MSL や XMAS では，スコーレム関数に指定要素値を適用して新規部分木を生成し，同一要素を持つ部分木同士を併合できる．しかし，異種情報源統合ではデータ形式の異なる要素同士の結合も要求され，利用者要求としても類似要素の併合などが予想される．異種情報源統合では上記要求に対する併合機能の拡張性が求められる．

3. XML に基づく異種情報源メディエーションシステム：MediPresto/XM

XML を共通データモデルとする，メディエーション・アーキテクチャの異種情報源統合システム：MediPresto/XM は，不特定多数の情報源のスキーマやデータに関する動的な異種性解消，およびビューの動的な生成・取得，という特徴を持つ．

動的な異種性解消は，問い合わせに応じた統合対象情報源の動的選定，および各情報源のスキーマやデータの異種性の動的解消を行う．システム管理者が最小限の統合ルールのみを事前登録しておき，利用者から問い合わせがあるたびに，ルールに従って問い合わせに合致もしくは類似する情報を探索し，スキーマやデータの異種性を解消する．

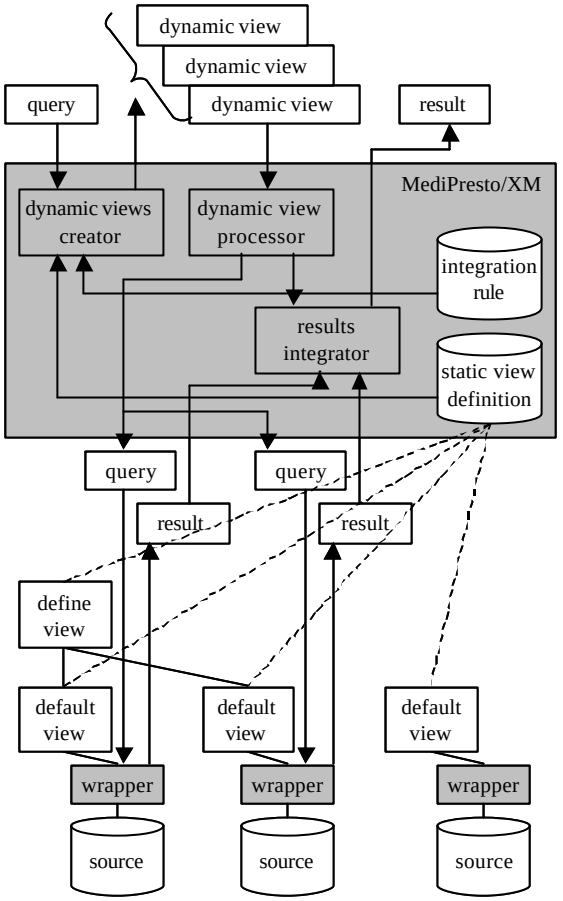

図 1. MediPresto/XM 概要

ビューの動的な生成・取得は，統合対象情報源のインタラクティブな決定を可能にする．利用者からの問い合わせに応じて静的ビュー(各情報源から XML 形式でデータを取得する問い合わせ，およびシステム管理者が定義する静的ビューに対する問い合わせ)から動的ビュー(利用者からの問い合わせと静的ビューの合成問い合わせ)を生成し，演算を行う．

MediPresto/XM ではあらゆる情報源を XML 形式で扱い，統合する．利用者は各情報源の XML 構造を参照せずに問い合わせを行う．MediPresto/XM は問い合わせに応じた対象情報源の探索，各情報源問い合わせの生成，そして各情報源からの実行結果の統合および構造変換処理を行う．図 1 に，MediPresto/XM の概要を示す．

4. MediPresto/XM における XML 構造変換

MediPresto/XM に対する問い合わせは，XMQL という宣言的 XML 構造変換定義言語で指定する．XMQL では各情報源の構造を参照しない．本章では，XMQL 処理における基本的構造変換機能と，要素併合機能を説明する．基本的構造変換は，(1)変換結果構造に対応する情報源部分木構造の探索，および各部分木の指定

```
<XMQL>
  <construct>
    <publisher-list>
      <publisher>$PUBLISHER</publisher>
      <author-list>
        <author>
          <name>$AUTHOR</name>
          <book-list>
            <book>$BOOK</book>
          </book-list>
        </author>
      </author-list>
    </publisher-list>
  </construct>
</XMQL>
```

図 2. XMQL 構造変換指定の例

構造への変換を実現する処理であり，要素併合は，(2)異なる抽出部分木内の要素同士の併合，および(3)同一性以外の判断基準による要素同士の併合を実現する処理である．

4.1. 基本的構造変換

MediPresto/XM は，まず利用者から XMQL 問い合わせを受け取ると，検索対象要素名と名前が同一である要素を含む静的ビューをすべて探索し，それぞれを XMQL と合成して動的ビューを生成する．次に利用者が選択した動的ビューを取得する．具体的には，動的ビューの検索対象要素名に対応する全要素を含む最小部分木構造を探索し，探索構造に対応する部分木を抽出し，各部分木を結果構造に変換する．この結果，情報源構造における部分木単位が維持された構造変換結果を得ることができる．

図 2 に，XMQL の例を示す．XMQL では，基本的に検索対象要素名と，結果階層構造を定義する．図 2 は，PUBLISHER，AUTHOR，BOOK という要素を検索し，construct 要素内に定義された部分木構造に変換する指定を定義している．部分木構造について，publisher-list は，PUBLISHER が割り当てられる publisher と，author-list という子要素を持つ．author-list は author という子要素を持ち，author は，AUTHOR が割り当てられる name と，book-list という子要素を持つ．book-list は，BOOK が割り当てられる book という子要素を持つ．図 3 に示す XML に対してこの XMQL で問い合わせた場合，図 4 に示す構造変換結果が得られる．構造変換結果は，構造変換元の XML インスタンスにおける PUBLISHER，AUTHOR，BOOK を含む最小部分木 BOOK-INFO ごとに構造変換が適用された部分木リストであることがわかる．

4.2. 要素併合

XMQL の基本的構造変換結果は対象情報源の XML 構造に依存するものであり，完全な構造統一化は実現されない．例えば，執筆文献情報リストを持つ著者情報

```
<ROOT>
  <BOOK-INFO>
    <BOOK>XML</BOOK>
    <PUBLISHER>pub-01</PUBLISHER>
    <AUTHOR-LIST>
      <AUTHOR>Konishi</AUTHOR>
      <AUTHOR>Suzuki</AUTHOR>
    </AUTHOR-LIST>
  </BOOK-INFO>
  <BOOK-INFO>
    <BOOK>RDB</BOOK>
    <PUBLISHER>pub-01</PUBLISHER>
    <AUTHOR-LIST>
      <AUTHOR>Suzuki</AUTHOR>
      <AUTHOR>Horiguchi</AUTHOR>
    </AUTHOR-LIST>
  </BOOK-INFO>
  <BOOK-INFO>
    <BOOK>Web Service</BOOK>
    <PUBLISHER>pub-02</PUBLISHER>
    <AUTHOR-LIST>
      <AUTHOR>Konishi</AUTHOR>
      <AUTHOR>Hayashi</AUTHOR>
    </AUTHOR-LIST>
  </BOOK-INFO>
</ROOT>
```

図 3. 情報源 XML の例

```
<xmql-results>
  <publisher-list>
    <publisher>pub-01</publisher>
    <author-list>
      <author>
        <name>Konishi</name>
        <name>Suzuki</name>
        <book-list>
          <book>XML</book>
        </book-list>
      </author>
    </author-list>
  </publisher-list>
  <publisher-list>
    <publisher>pub-01</publisher>
    <author-list>
      <author>
        <name>Suzuki</name>
        <name>Horiguchi</name>
        <book-list>
          <book>RDB</book>
        </book-list>
      </author>
    </author-list>
  </publisher-list>
  <publisher-list>
    <publisher>pub-02</publisher>
    <author-list>
      <author>
        <name>Konishi</name>
        <name>Hayashi</name>
        <book-list>
          <book>Web Service</book>
        </book-list>
      </author>
    </author-list>
  </publisher-list>
</xmql-results>
```

図 4. XMQL 構造変換結果の例

報リストを出版社ごとに得たいという要求で図 2 に示す XMQL が指定されることが考えられる．しかし，対象情報源が図 3 に示す XML である場合，図 4 に示す結果が得られ，異なる publisher-list に含まれる，同じ name の author 同士は併合されない．不特定多数の異

```
<XMQL>
  <construct>
    <publisher-list>
      <publisher>$PUBLISHER</publisher>
      <author-list>
        <author>
          <name>$AUTHOR</name>
          <book-list>
            <book>$BOOK</book>
          </book-list>
        </author>
      </author-list>
    </publisher-list>
  </construct>
  <grouping
    target = “publisher-list”,
    judgment = “publisher”,
    method = “equal” />
  <grouping
    target = “publisher-list/author-list”,
    judgment = “author/name”,
    method = “equal” />
  <grouping
    target = “publisher-list/author-list/author/book-list”,
    judgment = “book”,
    method = “true” />
</XMQL>
```

図 5. 部分木分割・併合指定を含む XMQL の例

種情報源統合では，構造統一化を実現するためにこのような併合を可能にしなければならない．

また，異種情報源統合ではデータ形式の異なる要素の併合が要求されることがある．例えば，名前が英語表記と日本語表記で表される同一著者要素の併合である．さらに，利用者が類似要素の併合を要求することも考えられる．例えば，タイトルにデータベースという文字列の含まれる文献要素の併合である．

XMQL では，①実施要素，②判定要素，③判定メソッド，の組の宣言で要素併合を指定する．実施要素は実際に併合する要素を指し，判定要素は併合するか否かの判断基準となる値を持つ要素を指す．各実施要素配下の判定要素を，判定メソッドを用いて比較し，併合するか否かを判断する．併合範囲は実施要素の親要素をルートとする部分木内とし，範囲内で実施要素をルートとする部分木を併合する．

情報源における併合要素を分解し，新たな併合要素を生成して構造統一化を実現するために，まず部分木分割処理を行い，その結果に対して併合処理を行う．

A) 範囲内において，実施要素をルートとする部分木を，判定要素が繰り返し要素とならないように分割(判定要素の要素番号を引数とするスコーレム処理．判定要素以外の要素はすべて同一となる部分木を，判定要素の数分だけ生成)．
B) 範囲内において，実施要素をルートとする部分木を，判定要素を判定メソッドに適用した結果に従って併合(判定要素の値を引数とするスコーレム処理．実施要素を併合し，新たな部分木を生成)．

図 5 に，要素併合指定を含む XMQL の例を示す．結果階層構造を示す construct 要素は，図 2 に示した XMQL と同様であり，grouping 要素が要素併合の指定を示す．grouping 要素内の target 属性が実施要素を，judgment 属性が判定要素を示す．それぞれ，construct 要素内のパスで表され，判定要素は実施要素からの相対パスで表される．method 属性は判定メソッドを示す．ここで, equal メソッドは判定要素の値が同一の場合に真値を返却するメソッド，true メソッドは判定要素の値に関わらず常に真値を返却するメソッドである．判定要素を判定メソッドで比較して真となる場合，実施要素を併合する．これを図 3 に示す XML に対して問い合わせた場合，図 4 に示す構造変換結果に対して要素併合処理が行われる．

まず分割処理を行う．publisher-list に対する併合指定について，各 publisher-list における publisher は繰り返し要素ではないために分割は実施されない．author-list に対する併合指定について，各 author-list における name が繰り返し要素であるため，name の数分だけ各 author-list が分割される．book-list に対する併合指定について，各 book-list における book が繰り返し要素ではないために分割は実施されない．この段階で得られる XML を図 6 に示す．

次に併合処理を行う．publisher-list に対する併合指定について，publisher が同一(pub-01)である 2 つの publisher-list が併合される．author-list に対する併合指定について，name が同一(Suzuki)である 2 つの author-list が併合される．book-list に対する併合指定について，book の値(XML と RDB)によらず兄弟関係にある 2 つの book-list が併合される．この段階で得られる XML を図 7 に示す．

なお, equal メソッドを使用する併合処理の結果，要素の重複が生じる．これは，判定要素の値が同一ではない場合の実施要素併合を実現するために，要素併合と重複排除を切り分けたためである．equal メソッドを使用した場合に生じる判定要素の重複を自動的に排除すると，図 8 に示す結果が得られる．

5. XML 構造変換処理の位置付け整理

MediPresto/XM の XML 構造変換と既存の構造変換を比較し，各変換処理の位置付けを整理する．

5.1. XML 構造変換の分類

XML 構造変換処理を, 大きく以下の 3 つに分類する．

(a) XML をノードツリー(ルートノードの下に子孫ノードが接続する形式のノード集合)として扱う構造変換．ツリーを辿りながら各ノードを抽出して必要処理を適用し，その結果を用いて新たなノードツリーを生成していく．XSLT 的な処理．

```
<xmql-results>
  <publisher-list>
    <publisher>pub-01</publisher>
    <author-list>
      <author>
        <name>Konishi</name>
        <book-list>
          <book>XML</book>
        </book-list>
      </author>
    </author-list>
    <author-list>
      <author>
        <name>Suzuki</name>
        <book-list>
          <book>XML</book>
        </book-list>
      </author>
    </author-list>
  </publisher-list>
  <publisher-list>
    <publisher>pub-01</publisher>
    <author-list>
      <author>
        <name>Horiguchi</name>
        <book-list>
          <book>RDB</book>
        </book-list>
      </author>
    </author-list>
    <author-list>
      <author>
        <name>Suzuki</name>
        <book-list>
          <book>RDB</book>
        </book-list>
      </author>
    </author-list>
  </publisher-list>
  <publisher-list>
    <publisher>pub-02</publisher>
    <author-list>
      <author>
        <name>Konishi</name>
        <book-list>
          <book>Web Service</book>
        </book-list>
      </author>
    </author-list>
    <author-list>
      <author>
        <name>Hayashi</name>
        <book-list>
          <book>Web Service</book>
        </book-list>
      </author>
    </author-list>
  </publisher-list>
</xmql-results>
```

図 6. 分割処理まで実行した結果

(b) XML をノードシーケンス(任意のノードツリーが順序付きで並んでいる形式のノード集合)として扱う構造変換．シーケンスを構成する各ノードを個別に抽出して必要処理を適用し，任意の順序で再びシーケンス化する．XQuery 的な処理.

(c) XML をノードリスト(同一パスで表されるノードツリーが繰り返されている形式のノード集合)として扱う構造変換．リストを構成する各ノードを個別に抽出して必要処理を適用する．ノード分割・グルーピングによる新たなノード生成も可能である．XMQL 的な処理.

```
<xmql-results>
  <publisher-list>
    <publisher>pub-01</publisher>
    <publisher>pub-01</publisher>
    <author-list>
      <author>
        <name>Konishi</name>
        <book-list>
          <book>XML</book>
        </book-list>
      </author>
    </author-list>
    <author-list>
      <author>
        <name>Suzuki</name>
        <name>Suzuki</name>
        <book-list>
          <book>XML</book>
          <book>RDB</book>
        </book-list>
      </author>
    </author-list>
    <author-list>
      <author>
        <name>Horiguchi</name>
        <book-list>
          <book>RDB</book>
        </book-list>
      </author>
    </author-list>
  </publisher-list>
  <publisher-list>
    <publisher>pub-02</publisher>
    <author-list>
      <author>
        <name>Konishi</name>
        <book-list>
          <book>Web Service</book>
        </book-list>
      </author>
    </author-list>
    <author-list>
      <author>
        <name>Hayashi</name>
        <book-list>
          <book>Web Service</book>
        </book-list>
      </author>
    </author-list>
  </publisher-list>
</xmql-results>
```

図 7. 併合処理まで実行した結果

XML をノードツリーとして扱う処理は，該当ノードに適用する処理の列挙で変換を指定する．結果構造は情報源の XML 構造に依存して変化する．

XML をノードシーケンスとして扱う処理は，情報源の XML 構造の影響を受けずに新たな XML 構造を生成でき，構造化度合いの低い文書的な XML を生成することもできる．指定は手続き的で複雑になる．

```
<xmql-results>
  <publisher-list>
    <publisher>pub-01</publisher>
    <author-list>
      <author>
        <name>Konishi</name>
        <book-list>
          <book>XML</book>
        </book-list>
      </author>
    </author-list>
    <author-list>
      <author>
        <name>Suzuki</name>
        <book-list>
          <book>XML</book>
          <book>RDB</book>
        </book-list>
      </author>
    </author-list>
    <author-list>
      <author>
        <name>Horiguchi</name>
        <book-list>
          <book>RDB</book>
        </book-list>
      </author>
    </author-list>
  </publisher-list>
  <publisher-list>
    <publisher>pub-02</publisher>
    <author-list>
      <author>
        <name>Konishi</name>
        <book-list>
          <book>Web Service</book>
        </book-list>
      </author>
    </author-list>
    <author-list>
      <author>
        <name>Hayashi</name>
        <book-list>
          <book>Web Service</book>
        </book-list>
      </author>
    </author-list>
  </publisher-list>
</xmql-results>
```

図 8. 重複排除まで実行した結果

XML をノードリストとして扱う処理は,構造化度合いの高いデータベース的な XML を生成することに適している．単純な宣言的変換指定が可能であるが，半構造の XML を生成することは難しい.

異種情報源統合は,複数の情報源を統合して単一のデータベースとして扱うことを可能にする.したがって,情報検索において必要となる機能は，構造的なデータ操作演算結果の取得であり，これには XML をノードリストとして扱う構造変換処理が適している.

5.2. ノードリストを扱うXML 構造変換

ノードリストを扱う XML 構造変換として，本稿で提案した XMQL 処理と，既存の MSL，XMAS 処理の違いを整理する.

まず，構造変換指定における，情報源の XML 構造に対する参照の有無が挙げられる．XMQL 指定では，静的ビューに含まれるノード名以外に参照する情報はない．これは構造の異なる不特定多数の情報源を統合する際に必要な条件であると考えられる.MSL や XMAS 指定では，静的ビューの構造を参照する必要があり，特定情報源の統合のみが可能である．この方法では統合処理速度を向上できる.

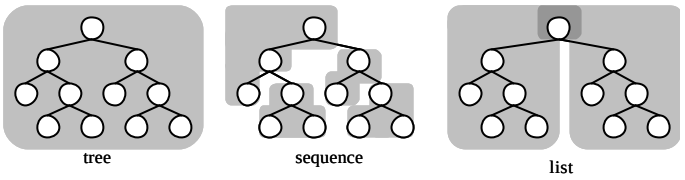

図 9. XML 形式のノードの捉え方

次に，XML 構造生成機能が挙げられる．前章で説明した通り，XMQL には要素併合機能があり，基本的構造変換結果を構成する各部分木について，異なる部分木内の要素同士を併合することが可能である．また，種々の判定メソッドによる併合実施の判断も可能であるが，特定情報源統合を考慮した MSL や XMAS にこれらの機能はない.

XMQL 処理は，情報源統合に適した宣言的 XML 構造変換定義言語に情報源構造を参照しないという特徴と，要素併合による構造統一化が可能であるという特徴を持ち,不特定多数の異種情報源統合に適している.

6. おわりに

本稿では，不特定多数の異種情報源統合を実現し，利用者の要求構造を生成可能な XML 構造変換定義，および変換手法を提案した．定義については情報源の構造を参照せず，宣言的な構造指定を可能にした．変換処理については，種々な条件による要素併合処理を導入することにより，情報源構造に依存しない部分木構造統一化を可能にした．以上より，不特定多数の異種情報源統合に必要な XML 構造統一化を実現した.

今後は，部分木探索方法の妥当性や，より実用的な要素同定方法について検証する.

文　献

[1] 鈴木, 小西, 林, 小林, 芳西. XML に基づく異種情報源メディエーションシステム：MediPresto/XM, データベースワークショップ 2001(DBWS2001), 2001.

[2] http://www.w3.org/Style/XSL/

[3] http://www.w3.org/TR/xquery/

[4] Y.Papakonstantinou, S.Abiteboul, and H.Garcia-Molina. Object Fusion in Mediator Systems, VLDB 1996.

[5] Y.Papakonstantinou and P.Velikhov. Enhancing Semistructured Data Mediators with Document Type Definitions, ICDE1999, 1999.