分散 XML ストリーム統合に関する提案

横山 昌平† 太田 学† 片山 薫† 石川 博†

†東京都立大学大学院工学研究科 〒192-0397 東京都八王子市南大沢 1-1

E-mail: †{shohei,ohta,katayama,ishikawa}@hikendbs.eei.metro-u.ac.jp

あらまし 最近, XML 形式のコンテンツを提供する, Web サービスと呼ばれるシステムが数多く登場している. また, これら分散した Web サービスや XML データを統合し一つのアプリケーションとして利用する手法が注目されている. XML の規格は非常に自由度が高く, 高度な記述ができる反面, その自由度のために膨大な種類のスキーマが存在している. これら種々雑多な XML データの統合は, それぞれのスキーマを理解するプログラムを事例毎に作成する必要があり, 開発コストが高い. 本論文では, XML ストリーム内にデータ統合の手続きを記述する手法を提案する. XML データを, 低レベルの概念である XML ストリームとして扱うことによって, Web サービスの統合が可能となり, 利用者はそれぞれのスキーマを意識せず, 統合された XML を得ることができる.

キーワード XML, 半構造データ, 分散 DB, XML ストリーム

A Method of XML Stream Integration

Shohei YOKOYAMA† Manabu OHTA† Kaoru KATAYAMA† and Hiroshi ISHIKAWA†

†Tokyo Metropolitan University 1-1 Minami-Osawa, Hachioji-shi Tokyo, Japan 192-0397

E-mail: †{shohei,ohta,katayama,ishikawa}@hikendbs.eei.metro-u.ac.jp

Abstract Recently, there is growing interest in Web service. A Web service is a software system, whose public interfaces and bindings are defined and described using XML. Because of the flexibility of XML, various schemas of Web services and XML resources exist. Therefore, it is essential to integrate various types of XML data. It is, however, practically infeasible to make a huge number of programs for every case. Furthermore, standardized methods for handling various schemas of XML data have not been proposed yet. This paper attempts to standardise XML-based web service integration. We propose an extended XML stream allows for XML integration. This enables flexible web service integration by having access to distributed XML resources without knowledge of each schema.

Keyword XML, Semi-structured Data, Distributed Database, XML Stream

1. はじめに

インターネットの登場によりコンピュータ間の接続が容易になった. そしてそれら接続されたコンピュータ間の共通データ形式として XML[1]は広く利用されている.

XML の利用機会は非常に多く, 例としてはベクトル画像を記述するための SVG, 数式を記述するための MathML, Web ページを記述する XHTML 等があげられる. また公に規格化されたものの他に, 企業間での情報交換に利用するための規格や, 個人が自分の住所録を管理するためだけに決めたスキーマといった局所的な規格の存在も XML は許容している.

これら XML データはファイルとして存在する以外に, 関係データベースの View や Web サービスの出力など, 動的に変化するコンテンツの表現にも多く用いられている. 以下, インターネット上に存在する, 静的なファイルとしての XML データ, および動的な XML データを合わせて XML 情報源と呼ぶ.

XML の普及により, XML を文法に持つ膨大なデータがインターネット上に存在している. それら無数に存在する情報源を統合あるいは連携させる技術を確立する事は急務とされている.

XML を処理する手法は大きく分けて二つある. 一つは XML の構造を木とみなし, メモリ上に展開する手法で, もう一つは, XML ファイルを先頭から読み込み, XML の構成要素や文字列をイベントストリームとして返す手法である. 前者は書き換えや資源の再利用など高度な処理を目的としており, 後者は高速・低消費メモリを特徴としている. それぞれの特徴はトレードオフの関係にある.

本論文では後者の手法を用いて, XML 情報源の統合を行っている. 提案する手法は XML ストリームを拡張し, 他リソースへの問い合わせをストリームの構成要素として記述可能にしている. また, その問い合わせは本手法のフレームワーク内で自動的に処理されるため, ユーザはスキーマ等に関する知識無く, 複数の

XML 情報源を画一的に利用できる．

また，最近注目が高まっている Web サービスへ適用についても議論を行う．

Web サービスは XML を入出力に持つソフトウェアコンポーネントで，インターネットを介して利用される．実際のシステムではこれら Web サービスと呼ばれるコンポーネントを複数統合し，それをアプリケーションとして，あるいは新たな Web サービスとして提供する事が多いと考えられる．しかしながら，Web サービスの統合手法はあまり議論されていない．提案手法はこの Web サービス統合に応用する事ができる．

1.1. 提案手法の概要

プログラムで XML を扱う場合，DOM や SAX といった API を利用する．DOM は XML を読み込みその木構造をメモリ上に展開する．SAX は XML ファイルを先頭から読み込み XML の構成要素を発見すると逐次イベントという形でプログラムに伝える．本稿ではこのイベント列を XML ストリームと呼んでいる．SAX は DOM に比べ低レベルでの処理と言う事ができ，高速・低消費メモリ量を特徴としている．本手法はこの SAX のイベントに小さな拡張を提案する事で，分散した XML データの連携を可能にしている．

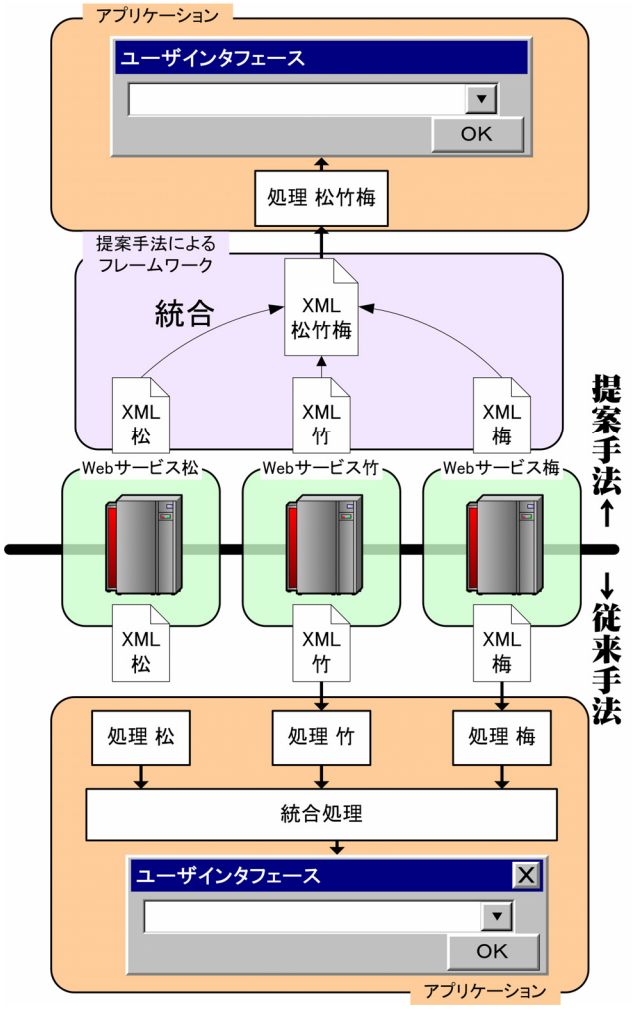

図1　システム全体像

Web サービスの統合を例にとり，従来手法と提案手法の全体像を図 1に示す．従来手法との違いは，データ(XML ストリーム)内に，統合の手続きを記述することである．そのため，Web サービスの統合処理はアプリケーションから独立し，統合された Web サービスはアプリケーションから一つの XML データとしてアクセスできる．

また，XML ストリーム統合処理を独立させることにより，統合の手続きは Web サービス提供側が管理する事も可能になる．この仕組みにより複数の Web サービスを束ねた新たな Web サービスの構築が容易になる．

1.2. 本稿の構成

続く第 2 章では提案手法と関連のある研究に関して記述する．第 3 章では提案手法について説明する．第 4 章で提案方式の応用として我々が実装中のシステムについて述べ，第 5 章で Web アプリケーションや P2P ネットワークへの適用可能性に関して議論を行う．最後に第 6 章で本稿の結論と今後の課題を述べる．

2. 関連研究

提案手法は二つの側面を持つ．一つは XML ストリームの処理手法としての研究，もう一つは複数の XML 情報源を統合する研究である．

XML ストリームに関連する研究は幾つかある．そのほとんどは問い合わせ手法の最適化に関する研究[2][3][4]である．それらに対し我々の研究の主眼はインターネット上に分散する XML データの画一的な処理手法にあり，本稿では問い合わせに関してはあまり留意していない．これらは補完しあう技術であると考えられる．

XML 情報源の統合に関する研究も行われている．代表的なものに W3C の XInclude[7]がある．この手法では，XML 文書中に XInclude で規定されたタグを挿入する事により，他の XML 文書の部分文書を挿入する事ができる．挿入する部分文書は XPointer[6]で示され，動的に解釈されるため，参照先の XML 文書が更新された場合，即座に変更が反映される．XInclude 自体が XML で表されており，DOM 等の XML を木とみなして利用する手法と整合性が高い．提案手法では，この XInclude と同様の機能を持つ“Query イベント”を定義しているが，これは，XML ストリームとして表現されるため，SAX の様なストリーム指向のツールからの利用を目的としている．つまり XInclude と提案手法は，XML を木として扱うか，ストリームとして扱うかという点で異なっている．

これらデータ中に他の XML リソースへの呼び出し

を記述する手法は，JOIN 演算を表現する際，要素数に比例した記述量が必要となる点が問題である．提案手法では，個々の呼び出し(Query イベント)を記述する代わりに，フックとしてあらかじめ JOIN の条件を記述する事ができる．個々の Query イベントはフックの条件に基づき自動生成される．

先行研究として我々は関係データベース上に XML ストリームを保存する XML データベース Saxophone の開発を行っている[7][8]．提案手法は Saxophone との連携も考慮している．我々は本研究を分散 XML データベース実現の基礎研究と位置づけている．

3. 提案手法

本章ではまず研究の前提となっている技術について説明を行う．そして提案手法である XML ストリーム拡張に関して詳述する．

3.1. 前提事項

XML ストリームは SAX のイベントを基に構成される．本稿では以下の四つの SAX イベントを利用する．

- 要素開始
- 属性出現
- 文字列
- 要素終了

複数の XML 情報源の統合を考えるとき，呼び出し側で，利用する情報源の種類や数，そしてそれぞれ対応したスキーマを把握した上で，提供するアプリケーションに沿った処理を行わなくてはいけない．前述したように，XML を利用した規格は非常に多くあり，それらを利用するアプリケーション毎に，処理を記述するのは効率が悪い．提案手法では XML ファイルよりさらに低レベルの概念である XML ストリームに着目し，そのレベルで統合手法を提案する事により，スキーマやアプリケーションによらない，画一的なフレームワークを構築している．

3.2. XML ストリーム統合

本節では提案する XML ストリームを構成する要素と，その統合技術について記す．

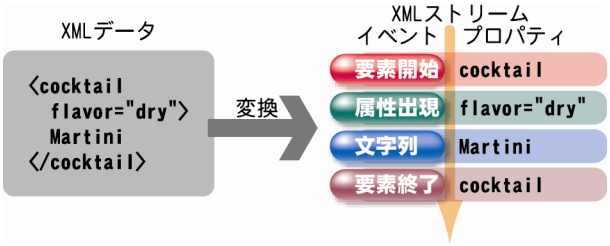

図2　XML ストリームへの変換

3.2.1. SAX イベント

提案方式では XML ファイルをストリームとして扱う．XML からストリームへの簡単な変換を図 2に示す．さらに SAX イベントには文法エラーなどを表すイベントも存在するが，説明の簡便のため本稿では考慮しないこととする．

3.2.2. Query イベント

Query イベントは提案手法で拡張されたイベントで，他のリソースへの問い合わせをプロパティとする．問い合わせは XPath[9]を利用し，URL の Query-String で表現される．問い合わせの書式とその例を図 3に記す．この例では http://a.b.com/e-j.xml 文書内に存在 /EngJpn/item/@ENG の値が Martini をとるノードを探し，そこに属する/EngJpn/item の文字列を XML ストリームとして返している．問い合わせ結果が得られた場合 Query イベントはその Query の結果として返ってきた XML ストリームで置き換えられる．

Query イベントによる問い合わせは提案手法によるフレームワーク内で処理されるため，アプリケーションに渡される XML ストリームは SAX で定義されている四種類のイベントのみで構成される．そのため，アプリケーションからの利用は SAX を用いる．

ある XML 文書全体を呼び出したい時も，この Query イベントで表現する．例えば，

http://a.b.com/e-j.xml?select=/

が指定された場合，a.b.com にある/e-j.xml のデータ全体が XML ストリームとして返される．

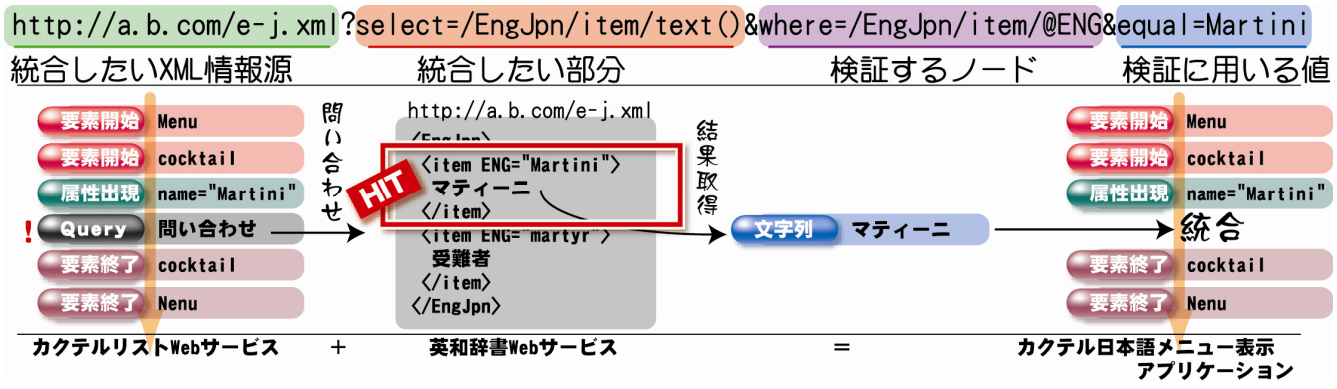

図3　Query イベントの例

図4　Hook イベントの例

3.2.3. Hook イベント

Query イベントは問い合わせ毎にイベントを発生させる必要がある．大きな XML データを扱う場合，Query イベントを大量に発生させる必要がある．Hook イベントを利用することにより，個々の Query イベントを一つのイベントにまとめることができる．

Hook イベントのプロパティも URL の形式をとり，Query-String に問い合わせ情報などが含まれる．Query-String は{select|where|equal|hook}のフィールドから成る．Query イベントと異なるのは hook フィールドの存在である．hook フィールドは以下の書式を持つ．

hook ::= (‘1st|’before’|’after’|’ final’) ’(“‘ *XPath* ‘”)’

例えば hook フィールドに final (“/Menu/cocktail”)と記述された場合，cocktail の要素終了イベントの直前に hook イベントで与えられた{select|where|equal}フィールドを持つ Query イベントが動的に発生する．ただし hook イベントの equal フィールドに XPath が指定されている場合，動的に発生する Query イベントの equal フィールドはその XPath が指し示す実際の値をとる．

図 3の問い合わせを Hook イベントで表すと次のようになる．

http://a.b.com/e-j.xml?select=/EngJpn/item/text()&where=/EngJpn/item/@ENG&equal=./@name&hook=final(“/Menu/cocktail”)

この Hook イベント発生以後，cocktail の要素終了イベント直前に Query イベントが動的に発生する．その際，equal フィールドは XPath が選択する実際の値”Martini”に変更される．

同様に 1st(“*XPath*”)は XPath で指し示す要素の開始イベントの直後に Query イベントが挿入される．before(“*XPath*”)と after(“*XPath*”)はそれぞれ XPath が指すノードの直前・直後に Query イベントが挿入される．

図 4は Hook イベントが動的に Query イベントを発生させる例を図示している．提案手法の利用者は Hook イベントのプロパティを記述する事により，複数の

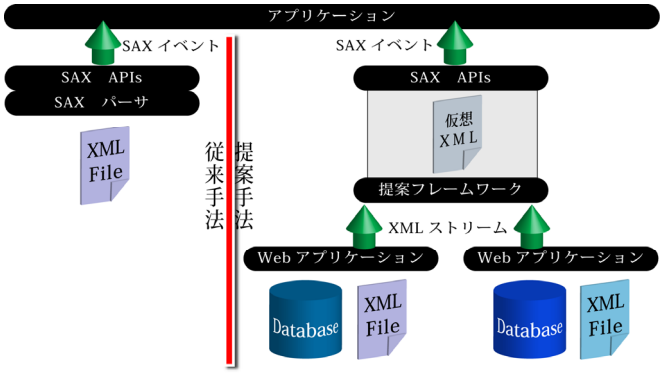

図5　提案手法の透過性

XML 情報源を統合し，一つの XML データにする事ができる．

4. フレームワークの実装

本章では提案手法を利用したサービスの要求者と提供者が提案手法を利用するためのフレームワークについて述べる．続く4.1節では統合された XML 情報源の利用手法を説明し，4.2節では XML ストリームをインターネット上で交換するためのプロトコルについて述べる．最後に4.3節で，XML 情報源統合を担うサービス仲介者の役割を例示し，システム全体像を描く．

4.1. サービス要求と XML ストリーム受信

提案手法は SAX を介して，ネットワークや XML 情報源統合に関する知識が無くとも，利用することが出来る．XML 情報源の統合やそれに付随する XML ストリームの送受信は提案手法が提供する SAX パーサ内に隠蔽される．この仕組みにより，SAX パーサからローカルホストにある XML ファイルを呼び出す手順と同じくして，ネットワーク上に分散された XML 情報源を統合し，画一的に呼び出すことができる．

クライアントアプリケーションは統合されたサービスの所在を URL で指定し，その結果を一つ XML データとして読み込み，それに対する処理を行う．

図 5で提案システムの透過性を図示する．提案システムをでは複数の Web サービスの統合処理を隠蔽し，一つのサービス(XML 文書)として公開できるため，アプリケーションはローカルホスト上の XML ファイルを SAX で読み込む手順で，統合された Web サービスを利用する事ができる．

4.2. XML ストリームの通信プロトコル

提案手法では XML ストリームの送受信に SOAP[10]を利用している．本節ではまず SOAP について簡単に説明を行う．

SOAP は Simple Object Access Protocol の略で，HTTP や SMTP 等トランスポート層のプロトコル上で XML

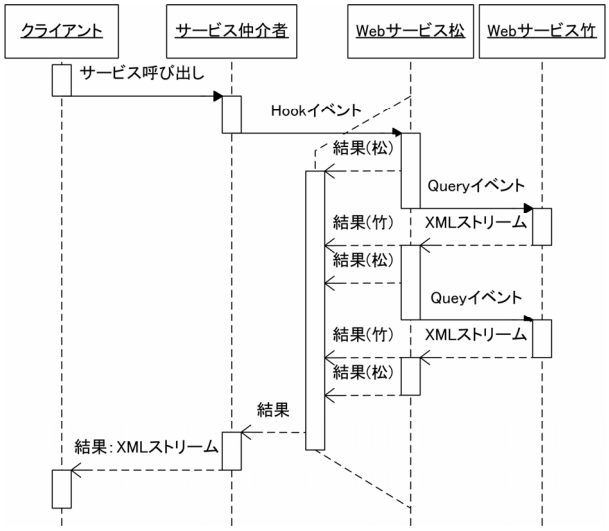

図6　メッセージの伝達

データを交換するためのプロトコルである．SOAP は基本的には一方向通信であるが，SOAP メッセージ内に用意されている Header 領域にメッセージ ID を埋め込むことにより，要求と応答や転送等を行う事ができる．

提案手法では XML ストリームを SOAP エンベロープに記述し，HTTP を介して通信を行っている．例えば他の Web サービスへの問い合わせは，Query イベントで構成される XML ストリームを問い合わせ先に SOAP 形式で送信する事によって行われる．問い合わせ結果もまた SOAP 形式の XML ストリームで表現される．

図 6は SOAP によるメッセージの伝達を UML のシーケンス図を用いて表している．図中サービス仲介者とあるのは，統合のための Hook イベントを定義する役割のことで，クライアント側・Web サービス提供側のどちらかが担当してもかまわない．Query イベントは提案手法によって自動で生成されるため，サービスの要求者・仲介者・提供者は個々の XML 要素の統合を記述する必要はない．また，サービスの要求者が行う処理は，結果 XML データを処理する SAX イベントハンドラの作成と URL によるサービスの呼び出しである．

4.3. 提案するフレームワークの全体像

提案手法を利用することにより，XML 情報源の統合をクライアントアプリケーションから独立させ，実際の処理を提案するフレームワーク内に隠蔽できる事はすでに述べた．本節ではサービス仲介者が複数の XML 情報源を統合するための Query イベントあるいは Hook イベントを Web 上で公開し，利用する過程を例に基づいて説明する．

説明に利用する XML 情報源を図 7に示す．これらの

カクテルレシピ辞書
http://a.com/recipe.xml

```
<Cocktail>
 <Recipe name="martini">
  <Alco name="gin" ratio="2"/>
  <Alco name="vermouth" ratio="1"/>
 </Recipe>
 <Recipe name="margarita ">
  <Alco name="tequila " ratio="3"/>
  <Alco name="triple sec" ratio="1"/>
  <Juice name="lime " ratio="2"/>
  <Salt/>
 </Recipe>
 <Recipe name="malibu and soda">
  <Alco name="malibu rum " ratio="1"/>
  <Alco name="Cola" ratio="1"/>
  <Salt/>
 </Recipe>
</Cocktail>
```

カクテルメニュー
http://b.com/menu.xml

```
<Menu>
 <Cocktail>martini</Cocktail>
 <Cocktail>margarita<Cocktail>
</Menu>
```

図7　XML データ例

XML 情報源に対し，サービス仲介者がレシピ付きメニューを提供する例を考える．メニューに記載されているカクテル毎に，辞書にレシピを問い合わせ，結果を挿入する必要がある．仲介者はカクテル毎に Query イベントを発生させる必要がある．しかし，カクテルメニューの XML ストリーム内に Query イベントを記述するのは，カクテルメニューが更新される度に，新しいレシピ問い合わせの Query イベントも追加する必要がありコストがかかる．また，カクテルメニューがサービス仲介者の管理下になくてはならず，制約が大きい．Hook イベントを用いれば，問い合わせをカクテルメニューデータ中に記述する必要がなくなる．

カクテルメニューの/Menu/Cocktail/text()の値とレシピ辞書の/Cocktail/Recipe/@name 値が等しいものを結合するための Hook イベントは以下のようになる．

http://a.com/recipe.xml?select=/Cocktail/Recipe&where=/Cocktail/Recipe/@name&equal=./text()&hook=final("/Menu/cocktail")

この Hook イベントをカクテルメニューの XML スト

リーム読み出し以前に宣言すれば，フレームワーク内で自動的に Query イベントが生成され，カクテル毎にレシピを問い合わせることが出来る．

カクテルメニューの XML ストリームを呼び出すには次のプロパティを持つ Query イベントを利用する．

http://b.com/menu.xml?select=/

このように XML 情報源の統合が，個々の XML の情報源の改変を行わずに，Hook イベントと Query イベントのみで表現する事ができた．

提案手法では Hook イベント，Query イベント共に XML ストリームの一部として扱っている．この例のように Query イベントと Hook イベントのみから構成される場合も，これを XML ストリームとみなし，URL でアクセス可能な場所に保存する．

http://c.com/toranomaki.xml

XML ストリームは Saxophone データベース[7][8]上に配置する事ができる．

この拡張された XML ストリームが提案するフレームワーク上で処理される過程を図 8に示す．

クライアントアプリケーションはサービス仲介者の URL を呼び出すだけで，統合された Web 情報源を利用することができる．また，Hook イベントはストリーム内の任意の場所で Query イベントを発生させる．Query イベントはプロパティに基づき問い合わせ行い，結果の XML ストリームで置き換えられる．その結果，クライアントアプリケーションは SAX イベントからなる XML ストリームを受け取る．

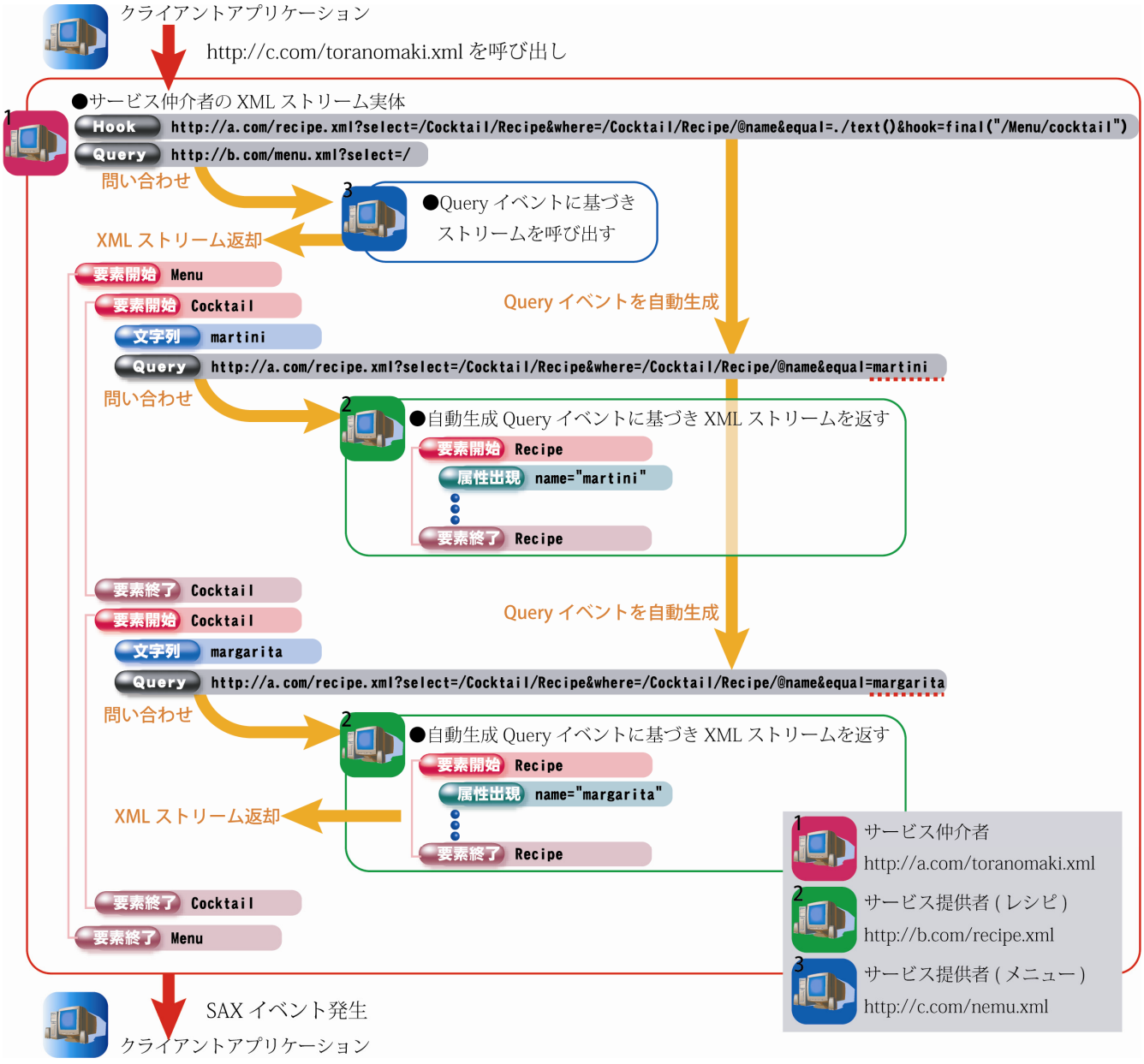

図8　XML 情報源の統合例

5. P2P ネットワークへの応用

SOAP を利用したメッセージングの特徴は，要求も結果も SOAP によって記述される事である．すなわち，メッセージ送信側・受信側双方が送信・受信両方の機能を持っているということである．これは Peer-to-Peer (P2P)の関係である．図 6の **Web サービス松**と**竹**の関係がその P2P の関係に相当する．

本論文では明確なサービス仲介者による，XML 情報源の統合手法について主に述べてきた．本節では P2P ネットワークのような，接続する全てのノードがサーバでありクライアントとなる環境で提案手法の有効性を議論する．

音楽ファイル共有サービスに代表される P2P ネットワークは，サーバが不要，あるいは比較的小さなインデクスサーバのみ設置するという手軽さで現在よく利用されている．提案手法を用いれば，P2P ネットワークに参加する多数のノードから得られる XML ストリームを統合し，一つの大きな XML データとして扱うことが出来る．一般的に P2P ソフトウェアは検索の結果として得られた各ノードからの情報を個別に扱っている．それらをどうユーザに画一的に見せるかは，クライアントソフトウェアの実装にかかっている．例えば P2P の検索結果として以下のようなデータ例が考えられる．

ノード A 結果

<Cocktail name="Martini">King of Cocktail</Cocktail >

ノード B 結果

<Cocktail name="Margarita">Sweet,Strong</Cocktail >

クライアントアプリケーションは，これら別々のノードから個別にやってくる XML データを処理しなければならない．

提案手法を用いれば，各ノードからの情報は XML ストリームとして受信され，フレームワーク内で一つの大きな XML ファイルに統合される(図 9)．クライアントアプリケーションはこの統合された XML ファイルを得ることが出来る．統合された結果は次の例が考えられる

<comment>.

<Cocktail name="Martini">King of Cocktail<Cocktail/>

<Cocktail name="Margarita">Sweet,Strong<Cocktail/>

</comment>

現在，提案手法の実装系として，Web ページのブックマークに関する P2P ネットワークを検討している．このシステムでは P2P ネットワークに参加する各個人が個別に管理するブックマークを，P2P の技術と提案手法を用いて仮想的に統合し，ひとつの非常に大きなブックマークを作り出す計画である．このような P2P ネットワークへ応用させることは今後の課題である．

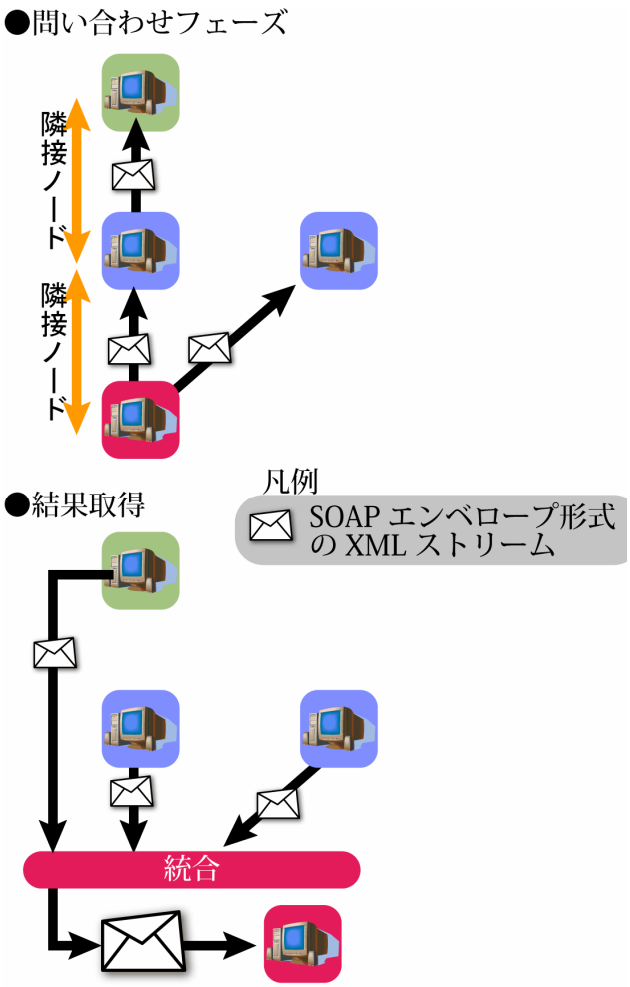

図9　P2P ネットワークへの適用例

6. まとめ

本稿では分散した XML 情報源の統合を行う手法を提案した．提案するシステムを利用した Web サービス提供者やサービスの仲介者は Hook イベントのプロパティにサービス統合のための問い合わせを記述することにより，クライアントに統合された動的な XML データを提供する事ができる．クライアントは Web サービスの組み合わせや統合手法を気にする事なく，SAX を用いて XML データを読み込むことが可能である．

また XML ストリームのネットワーク送受信を SOAP で行う手法を述べ，実際のデータの動きを例を挙げて示した．

今後の課題として，前節で述べた P2P ネットワークへの応用のほか，問い合わせ機能の強化および問い合わせ最適化の研究との統合を考えている．

文　献

[1] W3C, Extensible Markup Language (XML), http://www.w3.org/XML/

[2] B. Ludäscher, P. Mukhopadhayn and Y. Papakonstantinou, “A transducer-based XML query processor,” Proceedings of 28th International Conference on Very Large Data Bases, Hong Kong, Aug. 2002.

[3] T. Green, M. Onizuka and D. Suciu, “Processing XML Streams with Deterministic Automata and Stream Indexes,” The 9th International Conference on Database Theory, Jan.2003.

[4] F. Peng and S. S. Chawathe, “XSQ: Streaming XPath Queries,” Proceedings of the 2003 ACM SIGMOD international conference on Management of data, pp.431-442, June 2003.

[5] W3C, “XML Inclusions (XInclude) Version 1.0,” http://www.w3.org/TR/xinclude/

[6] W3C, “XML Pointer Language (XPointer) Version 1.0,” http://www.w3.org/TR/WD-xptr

[7] 横山昌平, 太田学, 片山薫, 石川博, “XML パーサを考慮した応用向き関係データベース構成法,” 第 13 回データ工学ワークショップ(DEWS2002)論文集, A5-3, Mar. 2002.

[8] 横山昌平, 太田学, 片山薫, 石川博, “XML 文書管理におけるブランチ機能を有するバージョン系列のための関係データベース構成法,” 信学論(D1), Feb. 2004.

[9] W3C, XML Path Language (XPath) Version 1.0, http://www.w3.org/TR/xpath

[10] W3C, SOAP Version 1.2 Part 1: Messaging Framework, http://www.w3.org/TR/SOAP/