データベースシステム 128－11
（２００２．７．１７）

例示操作によるWebサイト構築支援ツールの実装

毛利 隆軌 † 森嶋 厚行 †† 北川 博之 †††
† 筑波大学 システム情報工学研究科 †† 芝浦工業大学 工学部情報工学科 ††† 筑波大学 電子・情報工学系

概要

本稿は，XML上にWebサイトを構築するためのツールAQUAの実装について説明する．AQUAの特徴は，ユーザがWebページの例をいくつか示すとシステムがWebサイト全体を推論し，自動的に構築することである．AQUAの実装における特徴は，特別な制約や条件無しに，汎用のXQueryエンジンと組み合わせて動作出来ることである．単純な実装では，パフォーマンスやAQUAとXQueryエンジン間のコミュニケーションにおける問題が生じる．本稿では，AQUAの概要の他，これらの問題を解決するための技法を説明する．

Implementation of An Example-based Web-site Construction Tool

Takanori Mouri †, Atsuyuki Morishima ††, Hiroyuki Kitagawa †††
† Doctoral Program in Sys. and Info. Eng. Univ. of Tsukuba
†† Dept. of Info. Sci. and Eng., Shibaura Inst. of Tech.
††† Institute of Info. Sci. and Elec., Univ. of Tsukuba

Abstract

This paper presents an implementation of AQUA, a novel tool to construct Web-sites on top of XML. AQUA is a tool that receives examples of Web pages from the user and infers the entire Web site. A feature in its implementation is that it can work with ordinary XQuery engines, without any special requirements. A naive implementation raises problems on performance and communications between AQUA and XQuery engines. The paper explains techniques to solve the problems.

1 はじめに

近年，多様な情報源中に格納されたデータに基づくWebサイト構築が重要なデータ操作のひとつとなっている．現在，情報源としてはリレーショナルデータベースが主流であるが，今後XMLがWebサイトの構築のための重要な情報源になる事が十分考えられる．XMLはデータ交換やアーカイブのためのデファクトフォーマットとなってきているからである．この領域に関してこれまでに多くの技術が提案されてきた．例えば，Strudel[3]のようなWebサイト管理システムは半構造データモデルに基づいて情報源上に様々なWebサイトを生成することができる．

Webサイト構築においては次の2つの処理が必要となる．(1) 情報源から必要なデータの抽出・再構築，(2) 抽出したデータを埋め込むためのWebサイトのレイアウトの設計．既存の技術や枠組みではそれぞれの操作を異なるツールを用いて行っているため，ユーザはWebサイトの構築に2つのツールの使用方法を覚えなくてはならない．典型的な例としては，データの抽出はテキストエディタを用いてSQL文を記述し，Webページの作成はGUIで行う等がある．

本稿ではWebサイト構築ツールAQUA(Amalgamation of QUery and Authoring)の実装について述べる．AQUAはXML文書上にWebサイトを構築するためのツールである．特徴としては，ユーザが結果として欲しいWebページの例を示すことで，システムがユーザの意図を推論してWebサイト全体を構築するということがある．一見，AQUAは通常のHTMLやSMIL [8]ページオーサリングツールと同じように見える．Webページ作成はユーザがオブジェクトをドラッグ&ドロップすることによって行う．一般的なオーサリングツールと異なる点は，ページ作成操作が，暗黙のうちにWebサイト全体の構築操作となっていることである．

操作対象のデータがリレーショナルデータベースの場合は，データが完全に規則的な構造をもち，データオブジェクトのドメインもあらかじめ固定されている．QBE [10] や他の QBE スタイルの問合せ言語はこの性質を利用して定義されている．しかし，XML データは半構造データ [1][2] であるため，データ構造が完全な規則性を持たない．我々のシステムではシステムとユーザのやりとりに応じてデータオブジェクトのドメインを動的に決定する仕組みを持つ．

我々の開発したプロトタイプシステム [6] を用いた実験では，AQUA を用いれば SQL 等の関連知識をまったく知らないユーザでも XML 上に Web サイトを構築できる事が確認されている．

AQUA の実装の特徴は汎用の XQuery [9] エンジンと組み合わせて利用できることである．XQuery は XML 問合せ言語として標準となりつつあり，効率の良い処理系が現れることが予想される．AQUA は特別な条件なしに，このような汎用 XQuery エンジンを用いた，XML 文書のための使いやすい Web サイト構築ツールとして利用できる．

本稿では，システムの概要と実装について述べる．特に実装に焦点を当てる．

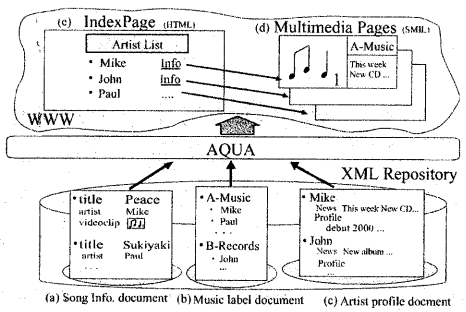


図 1: 利用例

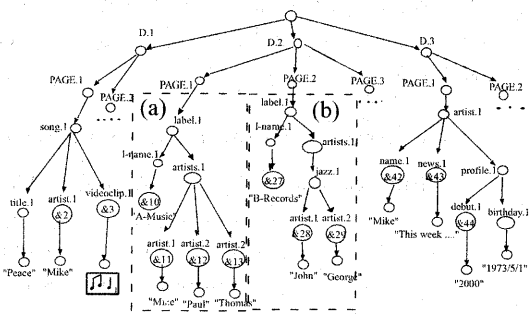


図 2: データモデル

2 利用例

本節では，AQUA の利用例について述べる．XML リポジトリに次の 3 種類の XML 文書があるとする．情報源として (1) 歌のタイトル，アーティスト名，ビデオクリップの情報を記述した XML 文書 (図 1(a))．(2) ミュージックレーベル毎にアーティスト名をリストした XML 文書 (図 1(b))．この文書の細部の構造はいくつかのバリエーションが存在する．図 2 を用いて説明すると点線で囲まれた部分 (a)(b) はそれぞれミュージックレーベルの情報を表している．しかしこれらの構造は異なる．一つは artist 要素がリスト表示しているもので，もう一つは artist 要素がジャンル毎にグループ化されてリスト表示しているものである．(3) アーティストのニュースやデビューした年などのプロフィール情報を記述した XML 文書 (図 1(c)) がある．

このとき「2000 年にデビューしたアーティストについて，各アーティストのビデオクリップをまとめたマルチメディアビュー (SMIL ページ) を作成し (図 1(d))，そのビューへのリンクを示すインデックスページを作る (図 1(e)).」という操作を行いたいとする．

3 AQUA の概要

AQUA で利用される基本概念は次の通りである．

DataBox:情報源中のデータ集合を表現する．2 節の利用例で用いた XML 文書を例として図 3 に示す．一つの DataBox にはユーザが示した要素を基にデータ集合を形成して表示する．本稿ではこれらを**ページ**と言う．(1) の `song` 要素，(2) の `label` 要素，(3) の `artist` 要素がそれぞれ一つのページとなる．また[▶]はビデオファイルの参照を示している．ユーザは「Next」と「Previous」のボタンをクリックすることで他のページをブラウジングすることができる．

Canvas:ユーザはここに DataBox からデータオブジェクトをドロップし，欲しい情報を描画する．

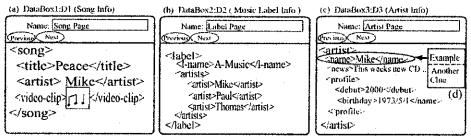


図 3: DataBox と ポップアップメニュー

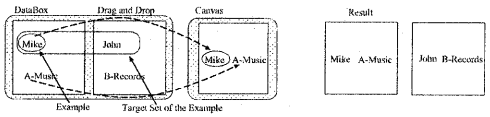


図 4: Example 指定と TargetSet

3.1 Example と TargetSet

AQUA の簡単な操作例を図 4 に示す．ユーザは DataBox 中のオブジェクトを Canvas に Drag&Drop(以下 D&D) し，配置するだけである．オブジェクトとは XML 文書の要素に該当する．

D&D する前にポップアップメニュー (図 3(d)) から“Example”と選択することにより，そのオブジェクトが Example として指定できる．図 4 において楕円で示したものが Example である．

ユーザが DataBox 中のオブジェクトを「Example である」と指定すると，そのオブジェクトの操作はそのオブジェクトが代表となるオブジェクト集合に対する操作であると解釈される．このときのオブジェクト集合を TargetSet という．TargetSet に含まれるオブジェクトは DataBox 中ではハイライト表示される．

DataBox 中の各ページにおいて，Example と同じ位置にあるオブジェクトの集合がデフォルトの TargetSet となる．例として，図 3 のアーティストのプロフィールのページにおいて‘**<name>Mike</name>**’を Example と指定するとき，各ページのアーティスト名の集合が TargetSet となる．“Another”と“Clue”をポップアップメニューから選ぶことで，TargetSet を修正することができる．

AQUA の操作手順を次に示す．

(Example 指定 (Another 指定 | Clue 指定)* | D&D)*

Example でないオブジェクトの D&D は，単にそのオブジェクトを Canvas に配置することを示す．Example と指定すると，デフォルトの TargetSet が設定される．Example オブジェクトに対する操作は TargetSet 中のオブジェクト群の操作を代表した操作と解釈される．

あるオブジェクトを Example と指定した後に，別のオブジェクトを Another と指定すると，Example オブジェクトと Another オブジェクトの関係を導出する．Example オブジェクトと導出された関係をもつオブジェクトすべてが含まれるように TargetSet が拡大される．

オブジェクトを Example と指定した後に，別のオブジェクトを Clue と指定すると，それを条件として TargetSet を狭めることができる．Clue と指定すると，そのオブジェクトが満たすべき条件を入力するためのウィンドウが現れる．例えば，‘**<name>Mike</name>**’を Example とした後に，**<debut>**の要素を Clue と指定する．条件として“=2000”と入れると，Mike が代表となる TargetSet は 2000 年にデビューしたアーティストのみが含まれるように縮小される．

3.2 Associations

異なる TargetSet 中のオブジェクト間の関係を示す．Associations の関係が成り立つときは，特定のオブジェクトの組合せに対してのみ操作が行われる．したがって，一種の結合演算として働く．AQUA では 2 種類の Association がある．

Structural Association(S-Assoc.) 2 つの Example 間の相対的な位置関係により生じる．最も単純な場合としては，同一のページで 2 つの異なるオブジェクトを Example と指定したときがある．図 5 において，DataBox 中のアーティスト名とミュージックレーベル名を並べてページの再構築を行うとする．このとき‘Mike’と‘A-Music’を Example とする．TargetSet A と B 間に S-Assoc. が発生して，ユーザの意図した結果が得られる．しかし‘John’を Example として TargetSet A を生成すると，TargetSet A と B 間に S-Assoc. が無いものと解釈される．よって結果は TargetSet A と B の直積となり，(Mike,A-Music),(Mike,B-Records),(John,A-Music),(John,B-Records) となる組合せが得られる．

Value Association(V-Assoc.) Example として指定されたオブジェクトの値が同一である場合に生じる．この Associaton は TargetSet の値により等結合することを意味する．図 6 に示した 2 つの DataBox を用いて説明する．DataBox からアーティスト名，ミュージックレーベル名，ニュースの組

図 5: TargetSet A と B 間の S-Association

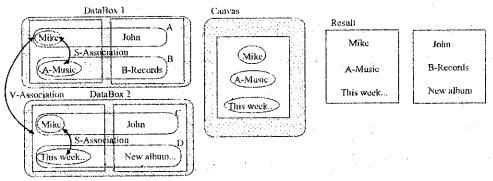


図 6: TargetSet A と C 間の V-Association

合せとなるページの構築を行うとする．このとき図 6 のように Example 指定すると，V-Assoc. があるものと解釈され，ユーザの意図した結果が得られる．しかし，'John' を Example として TargetSet A を生成すると，TargetSet A と C 間で V-Assoc. が無いものとされる．よって結果となる TargetSet の組合せは，(Mike,A-Music,This week...),(Mike,A-Music,New album...),(John,B-Records,This week...),(John,B-Records,New album...) となる．

4 AQUA による操作例

2 節に示された利用例は次の手順で実行できる．図 7 の番号は各操作に対応する．D1，D2，D3 はそれぞれ song 情報文書，ミュージックレーベル文書，アーティストプロフィール文書から生成される DataBox を表している．

(1) Canvas 中に HTML ページを作成するためのウィンドウを開く．

(2)D2 の '`<artist>Mike</artist>`' を Example と指定する (e_1)．このとき，図 2(a) と同じ構造を持つページにおいて Mike と同じ位置にあるオブジェクトの集合がデフォルトの TargetSet に含まれる．操作中の Example と，それが代表する TargetSet は，それぞれハイライト表示される．続いて D2 の '`<artist>Paul</artist>`' を Another と指定する (a_{11})．D2 の Next ボタンを押して図 2(b) と同じ構造を持つページに移動する．'`<artist>John</artist>`' を Another と指定する (a_{12})．すると，e_1 と a_{11} と a_{12} の関係を導出し，その関係を満たすように e_1 が拡大される．結果的に，e_1 は全アーティストの名前を含むことになる．

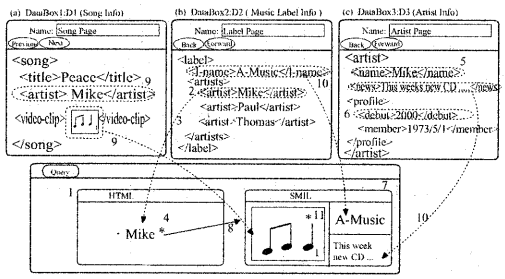


図 7: 操作手順

(3)D2 から e_1 である 'Mike' を Canvas に D&D する．

(4) Canvas 上の 'Mike' を repetition と指定 (*マークを追加) する．これにより，全アーティスト名がこのページに集められる．もし，この指定がなければ一つのページに一人のアーティスト名を含んだページが e_1 中のアーティスト分だけ作成される．

(5)D3 のオブジェクト '`<name>Mike</name>`' を Example と指定する (e_2)．すると，e_1 と e_2 のオブジェクト間に V-Assoc. が生じる．この Association は，これらの TargetSet を等結合することを意味する．

(6)D3 の`<debut>`要素を Clue とする．Clue の条件を "=2000" とする．すなわち，2000 年にデビューしたアーティストだけが e_2 の TargetSet に含まれる．

(7) Canvas 中に SMIL ページを作成するためのウィンドウを開く．

(8)HTML ページから SMIL ページへのハイパーリンクを作成する．

(9)D1 の '`<artist>Mile</artist>`' を Example として (e_3)，ビデオクリップも Example とする (e_4)．そして e_4 を Canvas へ D&D する．

(10)D2 の`<l-name>`要素を Example として (e_5)，D3 の`<news>`要素を Exmaple とする (e_6)．それぞれを Canvas へ D&D する．

(11) ドロップしたビデオクリップに repetition と指定する．これにより，各アーティストのビデオクリップが連続映像となる．指定がない場合はそれぞれのビデオクリップ毎にページが作成される．

5 セマンティクス

本節ではAQUAがユーザとのやりとりからWebサイトを構築する方法について述べる．図2で示したように，システムはデータをオブジェクト木としてモデル化する．すべてのノード(オブジェクト)はラベル付けされている．このラベルにはラベル名とラベルナンバーがある．ラベルはXML文書の要素と対応する．ラベルナンバーは同じノードを親とする同一のタグがあるとき，順序付けされる．各オブジェクトはOIDを持つ．説明に用いる際には&nと表記する．ルートからオブジェクトへのパスとオブジェクトの値をそれぞれpath(&n),value(&n)のように表記する．例えば，$path$(&11) =D.2→PAGE.1→label.1→artists.1→artist.1,value(&11)="Mike"である．

5.1 Candidate Sets

始めにシステムはExampleオブジェクトのCandidate Setsを計算する．Example e におけるCandidate Set(CS_e)は次のように示す．

$$CS_e = \{o | o \in O \wedge C\text{-}Pred_e(o)\}$$

Oはオブジェクト木の全てのオブジェクトの集合を示し，$C\text{-}Pred_e(o)$はワイルドカードを持ったパス式である．$C\text{-}Pred_e(o)$は次に示すように，Example指定とAnother指定から導出される．

例:4節の(2)の操作で得られる$CS_{\&11}$のCandidate Setを示す．

$CS_{\&11} = \{o|$ $o \in O \wedge$ D.2→PAGE.?⟨PAGE.1⟩ →label.1
→artists.1→*⟨ε⟩
→artist.?⟨artist.1⟩[o⟨<a>Mike⟩]}

"⟨"と"⟩"によって囲まれる部分はannotationである．この情報を除くパス式は，D.2→PAGE.?→label.1→artists.1→*→artist.?　となる．図2で示したデータを用いると，$CS_{\&11}$ = {"<a>Mike", "<a>Paul", "<a>Thomas",... "<a>John", "<a>George",...}となる．(ミュージックレーベルのページにあるすべてのアーティスト名が含まれる．)

annotationから得られる$path(e)$と$value(e)$の情報から，Predicateを導出する．一般的に$C\text{-}Pred_e(o)$を得るのは次のようである．

(1) オブジェクトeをExampleと指定する．デフォルトのCandidate-Predicateは$p[o\langle v\rangle]$のように定義される．ここで，pは$path(e)$のPAGE.iをPAGE.?⟨PAGE.i⟩で置き換えたものであり，vは$value(e)$である．例えば，4節の操作(2)において，ユーザがオブジェクト&11("<artist>Mike</artist>")をExampleと指定したとき，D.2→PAGE.?⟨PAGE.1⟩→label.1→ artists.1→ artist.1[o⟨<a>Mike⟩]がデフォルトのCandidate-Predicateとして導出される．このPredicateは，異なるページの同じ位置にあるオブジェクト全体を定義している．

(2)Candidate-PredicateはAnother指定が行われるたびに変更される．Another指定されたオブジェクトaがTargetSetに含まれるように変更される[5]．基本的なアイデアは，$path(e)$と$path(a)$が一致しない場所に，ワイルドカードを挿入するということである．

例えば，操作(2)においてユーザがオブジェクト&12("<artist>Paul</artist>")と&28("<artist>John</artist>")をAnotherと指定する．図2のオブジェクト木において，ルートから3つのオブジェクトまでの3つのパスに適合するPredicateとして上の$CS_{\&11}$が導出される．

5.2 Target RelationとWebサイトの構築

Example e におけるTargetSet TS_e はTarget Relationを基に定義されている．始めにTarget Relationを定義する．Target Relation TR は全てのTargetSetの結合演算によって得られ，次のように定義する．$TR = \sigma_{p_0}(CS_1 \bowtie_{p_1} \ldots \bowtie_{p_{n-1}} CS_n)$ ここで，p_0はClue操作での選択演算を示し，p_isは結合演算を示す．例えば，$value(e_1) = value(e_2)$とV-Associationを表すとき，オブジェクトe_1とe_2の値が同じであることを意味する．図9は図8を基にしたTRである．Example e のTargetSet TS_e は次のように定義される．

$$TS_e = \pi_{attr(CS_e)}(TR)$$

ここでの$attr(CS_e)$はTR中の属性eと一致するものを示す．

次に，Canvas上で操作を行ったrepetition指定(*)に基づいてNest演算とProjection演算を適応して[4]，Target Relationを変形させる．図7で示した例からは，図10に示す入れ子型リレーションが得られる．

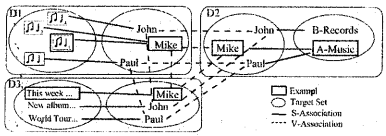


図 8: TargetSet と Association

Ex.Mike in D1	Ex. [image] in D1	Ex.Mike in D2	Ex. A-Music in D2	Ex.Mike in D3	Ex. This week... in D3
Mike	[image]	Mike	A-Music	Mike	This week...
Mike	[image]	Mike	A-Music	Mike	This week...
Paul	[image]	Paul	A-Music	Paul	World Tour...
John	[image]	John	B-Records	John	New album...

図 9: Target Relation 例

$$\nu_{V=([image])}(\pi_{Mike,[image],A-Music,Thisweek..}(TR))$$

最後に，その結果を Canvas で指定されたレイアウト従い Web ページ (図 1(d)(e)) として出力する．

6 実装

AQUA システムの構成について図 11 に示す．ユーザは DataBox と Canvas といった GUI コンポーネントを用いて操作を行う．ユーザは DataBox から Canvas にデータオブジェクトを D&D することで，Web ページの例をシステムに示す．ユーザが DataBox で行った操作から DataBox Manager がマッチングパターン (Candidate-Predicates) と選択・結合条件 (Association) を導出する．Canvas Manager は Canvas で指定されたグルーピング情報とサイトテンプレートを導出する．マッチングパターンと選択・結合条件とグルーピング情報を用いて Query Generator で XQuery を生成する．内部の XML エンジンでは TargetSet を計算する時に利用される (詳細は 6.2 節で述べる)．Site Constructor で XQuery エンジンからの問合せ結果をサイトテンプレートに埋め込み Web サイトを生成する．

システム (図 12) の実装には Java(Java2 SDK1.3) を用い，Drag&Drop API を利用した．コードサイズは約 21,000 行である．XQuery エンジンとしては QuiP [7] を使用した．

Ex.Mike in D1	V	Ex.A-Music in D2	Ex. This week... in D3
	Ex. [image] in D1		
Mike	[image]	A-Music	This week..
Paul	[image]	A-Music	World Tour..
John	[image]	B-Records	New album..

図 10: 入れ子型リレーション例

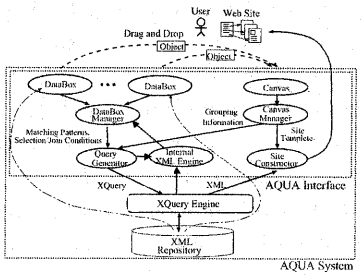


図 11: AQUA のアーキテクチャ

6.1 XQuery 問合せの生成

AQUA は 5 節で説明した TargetSet の計算や入れ子型リレーションを XML 形式で出力するために XQuery 問合せを生成する．XQuery はパス式，結合演算，Nest 演算を行うことができるため，XQuery を用いてこれらの処理を行うことは可能である．

ユーザが Example 指定，Another 指定，Clue 指定した時に AQUA は TargetSet の計算を行う必要がある．DataBox 中の TargetSet のハイライト表示を行うためである．TargetSet の計算には Target Relation を生成する XQuery 問合せを導出する必要がある．この問合せの結果として得られたオブジェクトを基にハイライト表示を行う．また Web サイト構築を行う時に，XML 形式の入れ子型リレーションを生成する XQuery 問合せを導出する．このときの XQuery 問合せのひとつとしては，次の 2 つの副問合せからなる問合せが挙げられる．(1) 入れ子になった for 節と where 節を用いて Target Relation を生成する．(2) この結果を元にして Nest 演算を行う．

図 10 に示した入れ子型リレーションを生成するための XQuery を図 13 に示す (document,data,text 関数は省略する)．S-Association はオブジェクト間での共通パスを変数に束縛することで実現している．例えば変数 $p2 は artist($o4) と l-name($o3) の正確な組合せを作るために使用されている．

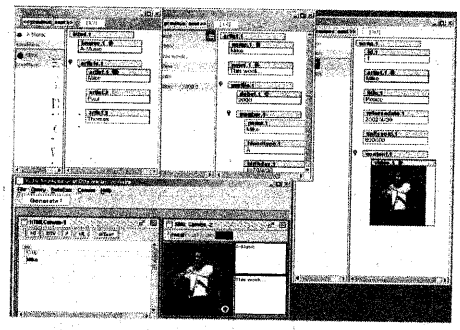


図 12: プロトタイプシステム

```
let $TR := <result>{
for $p1 in //D1/PAGE/song,
  $o1 in $p1/artist, $o2 in $p1/video-clip,
   $p2 in //D2/PAGE/label,
    $o3 in $p2/l-name, $o4 in $p2/artists//artist
     $p3 in //D3/PAGE/artist,
      $o5 in $p3/name, $o6 in $p3/news,
       $o7 in $p3/profile/debut
where $o1=$o3 and $o1=$o5 and $o7=2000
return <tr><o2>{$o2}</o2><c><o3>{$o3}</o3>
          <o4>{$o4}</o4><o6>{$o6}</o6></c></tr>
}</result>
let $result := <result>{
 for $ta in distinct-values($TR//tr//c)
 return <item>{$ta/o4}{$ta/o3}{$ta/o6}<rep>{
  for $tb in $TR//tr
  where $ta = $tb//c
  return <item>{$tb/o2}</item>
 }</rep></item>
}</result>
return $result
```

図 13: 生成される XQuery 問合せ

6.2 実装に関する問題とその解決

AQUA のアルゴリズムはデータのモデルレベルでは明確に定義されているが，実装はそれほど簡単ではない．3 節で述べたように，ユーザの操作により推論された TargetSet がユーザの意図するものかを判断させるために DataBox 上で TargetSet をハイライト表示させる必要がある．5.2 節で述べたように，TargetSet に含まれるオブジェクトを計算するたびに，システムは全てのデータに対して問合せを実行する必要がある．この仕組みを XQuery エンジンを用いて実装するには問題がある．それはパフォーマンスの問題と OID の問題である．次にこれらの問題の解決法について述べる．

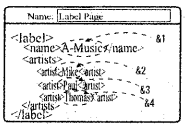
(A) パフォーマンスの問題: TargetSet の計算はデータ全体に対して行われる．システムは文書全体を取り込む必要があるため，結果として処理時間が長くなってしまう．ユーザインターフェースにとっては応答時間が長くなるのは問題である．

そこで解決法として次のようなことが挙げられる．ユーザが一度に見られるデータはある一つの DataBox 中にあるデータであり，全ての TargetSet に関して知る必要はない．我々はこの性質を利用する．例えば (1) ある TargetSet を XPath で表記したとき /labels/labal/l-name とする．(2) またこのときのページの単位は label 要素とする．(3) 現在 DataBox で開いているページは 3 番目とする．このとき，システムが問合せの XPath を /labels/labal[3]/l-name のように書き換えたとする．この条件を加えることで，一般的に結果は小さくなるので，XQuery エンジンでの処理効率が上がる可能性がある．

(B) OID の問題: AQUA のデータモデルではすべてのオブジェクトは OID を保持している．XQuery のセマンティクスでも OID は存在するが，OID を明示的に扱うことができない．そのためシステムは問合せ結果が DataBox に含まれているものかどうか判断できない．理由は，同じ DataBox 中に同じ値を持つオブジェクトが複数存在するかもしれないからである．しかし，AQUA は問合結果から DataBox 中のオブジェクトをハイライト表示させるのに OID を用いている．図 11 で示した内部の XML エンジンは，XQuery エンジンからの結果と DataBox 中のオブジェクトを結びつけるために使用される．これは次のように行う．Association を基に Candidate Set の結合を行い，Target Relation TR を得る ($TR = \sigma_{p_0}(CS_1 \bowtie_{p_1} \ldots \bowtie_{p_{n-1}} CS_n)$)．また DataBox 中の Example e_k の TargetSet のオブジェクト集合が $TS_k (TS_k = \pi_{attr(CS_k)}(TR))$ に含まれることが言えるので，次のように $TR_k^{s-assoc*}$ を定義することが出来る．

$$TR_k^{s-assoc*} = CS_{k_1} \bowtie_{p_{k_1}} \ldots \bowtie_{p_{k_{m-1}}} CS_{k_m}$$

ここでの CS_{k_i} は e_k と S-Association の関係をもつオブジェクトの Candidate Set である．例えば，図 8 において，D2 の 'Mike' を e_k とすると，CS_{k_i} は e_k 自身と 'A-Music' を含む．また TS_k は他の Association や選択演算を考慮しているので $TS_k \subseteq \pi_{attr(CS_k)}(TR_k^{s-assoc*})$ と言える．ここで内部のエンジンは，(1) 一時的に DataBox のオブジェクトにそれぞれ異なる OID を付ける (図 14(左))．(2) その OID をもった $TR^{s-assoc*}$ を計算する (図 14(右))．(3)XQuery エンジンで計算した TR の結果と結合演算を行う．結果として TS_k の OID を得ることが出来る (例では {&2,&3} が得られる)．結合演算は次の



Ex.Mike	Oid_1	Ex.A-Music	Oid_2
Mike	&2	A-Music	&1
Mike	&2	A-Music	&1
Paul	&3	A-Music	&1
Thomas	&4	A-Music	&1

図 14: オブジェクト識別機構

ようになる.

$$\pi_{oid_k}(\pi_{oid_k,c}(TR_k^{s-assoc*}) \bowtie_c \underline{\pi_c(TR)})$$

アンダーラインの部分は XQuery エンジンでの計算結果であり，c は $TR^{s-assoc*}$ と TR の結合条件となる属性である．$TR^{s-assoc*}$ と TR の結合には V-Association を用いる．パフォーマンスの所で説明した技法を使用できるので，一般に $TR_k^{s-assoc*}$ の計算コスト，$\pi_{oid_k,c}(TR_k^{s-assoc*})$ と $\pi_c(TR)$ の結果のサイズは比較的小さくなる．

7 おわりに

本稿では例示操作を用いた Web サイト構築ツールの概要とその実装について説明した．本ツールの特徴は，ユーザが作成したいくつかの Web ページの例から，ユーザが意図する Web サイト全体を構築できることである．半構造データである XML に対応するため，その仕組みは自明ではない．本稿では特に実装に焦点を当てて説明した．ポイントは汎用の XQuery エンジンと組み合わせて利用できることである．本稿ではシステムを実装する際に起きる問題と，我々の解について説明した．今後の課題としては，動的な Web サイトを生成する仕組みの開発，過去に行った操作の再利用を実現する機構の開発などが挙げられる．

謝辞

本研究の一部は，日本学術振興会科学研究費基盤研究 (B)(12480067) による．

参考文献

[1] S. Abiteboul. Querying semi-structured data. *Proc. 6th International Conference on Data Theory (ICDT'97)*, pp. 1-18, 1997.

[2] Peter Buneman. Semistructured data. *Proc. 16th ACM Symposium on Principles of Database Systems (PODS'97)*, pp. 117-121, 1998.

[3] M. Fernandez, D. Florescu, J. Kang, A. Levy, and D. Suciu. Catching the Boat with Strudel:Experiences with a Web-Site Management System. *Proc. ACM SIGMOD'98*, pp.414-425, 1998

[4] P. C. Fischer and S. J. Thomas. Operators for non-first-normal-form relations. *Proc. IEEE COMPSAC83*, pp. 464-475, 1983.

[5] A. Morishima, S. Koizumi and H. Kitagawa. Drag and Drop: Amalgamation of Authoring, Querying, and Restructuring for Multimedia View Construction. *Proc. 5th IFIP 2.6 Working Conference on Visual Database Systems(VDB5)*, pp. 257-276, 2000.

[6] A. Morishima, S. Koizumi, H. Kitagawa, and S. Takano. Enabling End-users to Construct Data-intensive Web-sites from XML Repositories: An Example-based Approach. *Proc. 27th VLDB Conf.*, pp. 703-704, 2001.

[7] Software AG. QuiP. http://www.softwareag.com/.

[8] W3C. Synchronized Multimedia Integration Language(SMIL). http://www.w3c.org/AudioVideo, 2002.

[9] W3C. XQuery 1.0: An XML Query Language. X3C Working Draft, http://www.w3.org/TR/xquery, 2002.

[10] M. M. Zloof. Query-by-Example: a Data Base Language. *IBM Systems Journal*, Vol. 16, No. 4, pp. 324-343, 1977.