エンドユーザ主導開発を志向したWebサービス連携方式の実験と評価

西田　晋平　　湯浅　亮祐‡　　中所　武司

明治大学 理工学部 情報科学科

概要

新たなビジネスモデルを実現するアプリケーションをサービスの組み合わせで短期開発し，稼働後のビジネスモデルの変更に対してもタイムリーに保守を行っていくためには，エンドユーザ主導開発が適している．そのための技術課題として，HTML 形式のインターフェイスを持つ既存の Web アプリケーションとの連携，および XML 形式のインターフェイスを持つ Web サービス間のマージ処理があげられる．そこで，例題として，学内の学生向け個人別試験時間割表作成システムを試作し，HTML から XML へのラッピング処理と複数の XML 文書のマージ処理を XSLT で記述する方式により，アプリケーション非依存の汎用性のある Web サービス連携システムを実現した．さらに，ビジュアルツールの導入により，エンドユーザ主導の開発方式を確立した．

Enduser-Initiative Web Service Integration

Shinpei Nishida,　Ryousuke Yuasa¶　and　Takeshi Chusho

Department of Computer Science, Meiji University.

Abstract

For enduser-initiative development of Web applications by Web service integration, technologies for HTML-to-XML wrapping and XML merging are developed. While application-specific processes are described in a script language with a visual tool, application-independent processes are generated automatically.

1　はじめに

近年，インターネットやイントラネットに接続されたパソコンの普及と共に，オフィス業務の効率化という観点から，業務の専門家が自ら情報システムを構築する必要性が高まっている．たとえば，基幹業務システムのように投資に対する効果が明確なものは，従来のように情報処理の専門家が開発すればよいが，小さな部門あるいは個人の業務を対象とするものや頻繁に機能変更が発生するものは従来の開発方法はなじまない．

特に，電子商取引に代表される Web アプリケーションの中には頻繁な機能変更を伴うものが増加しており，このような絶えざる変化に対応することが求められるアプリケーションは，業務の専門家が自ら開発し，自ら保守を行なうのが望ましい．

この新しい動向に対応して，短期間でタイムリーにアプリケーションを開発するために 90 年代半ばから

‡今年４月より（株）日立情報システムズに勤務

¶He joined Hitachi Information Systems, Ltd. in April.

コンポーネントやビジュアルプログラミングに代表されるような新しい開発ツールが実用化され，コンポーネントの組み合わせ（モデリング）というボトムアップ技法の開発方法がとられるようになっている．

さらに，最近では，既存の Web サービスを組み合わせて，迅速にアプリケーションを構築する技法が注目されているが，このようなソフトウェアのサービス化の傾向は今後加速されると思われる．エンドユーザが考案したビジネスモデルを実現するアプリケーションをサービスの組み合わせで短期開発し，稼働後のビジネスモデルの変更に対してもタイムリーに保守を行っていくという方法は，エンドユーザ主導開発 [1] に適している．そのための技術課題として，HTML 形式のインターフェイスを持つ既存の Web アプリケーションとの連携，および XML 形式のインターフェイスを持つ Web サービス間のマージ処理があげられる [2][3].

本稿では，最初に Web サービスにおける技術課題について述べ，次に，例題システムの開発を通して，HTML から XML へのラッピング処理 [4] と複数の XML 文書のマージ処理を XSLT で記述する方式について述べ，その評価を行う．

2 Web サービスの技術課題

2.1 従来の Web アプリケーション

従来の Web アプリケーションは，図 1 のようにブラウザからの入力データを受け取り，ブラウザに処理結果を表示する，といった形式をとることが一般的であった．この場合，結果は HTML 形式でクライアントに返されるため，データをさらに処理・加工するという処理が難しい．これは HTML が Web ブラウザに表示するためのドキュメント記述言語であって，機械処理に向かないからである．

そこで，Web サービスでは，HTML の代わりに機械処理が容易な XML を使ってアプリケーション間の連携を行う [5].

2.2 Web ページのラッピング方式

既存の Web アプリケーションを Web サービスとして提供するには 2 通りの方法がある．その 1 つは，新規に Web サービスの仕様に合わせてフロントエンド部分を再構築する方法である．しかしながら，Web サービスが十分に普及するまでの過渡期においては，既存の Web アプリケーションの多くがこの方法をとるとは考えにくい．そこで，もう 1 つの方法は，既存の Web アプリケーションの内部処理を変更しないで，その出力だけを XML 化して，Web サービスとして連携させるという Web ページのラッピング方式である．

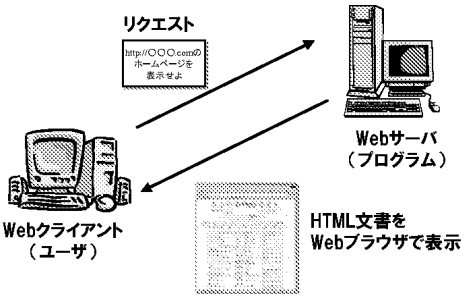

図 1: 従来の Web アプリケーション

例えば，実際に Web ページラッピング方式を適用する事例として，図 2 のような，ある旅行会社の例を考えてみる．この旅行会社は，ユーザの日程や予算に応じた旅行プランを作成するという旅行プラン作成サービスを提供しており，現在はホテルの空室情報検索サービスと連携して，ユーザの予算に応じた部屋が指定日に空室であるか調べて，通知するようになっているとする．

この旅行プランに，現在は映画会社と航空会社が Web ページ (HTML) として提供している劇場の空席情報と飛行機の空席情報を含めたいという要望があったとする．このような場合に，Web ページラッピング技術を使って，空席情報を XML 化し，旅行プラン作成サービスと連携させる必要がある．

2.3 Web サービスのマージ方式

Web サービス連携では，複数の Web サービスから得られた情報をマージして適切な情報にまとめる必要がある場合が多い．たとえば，上記の旅行プラン作成サービスにおいて，ある指定した期間内において，劇場の空席と飛行機の空席とホテルの空室がすべてありの場合の日程をすべて表示するような場合である．

このような Web サービス連携を実現するためには，XML 形式で提供される情報をマージする方式が必要

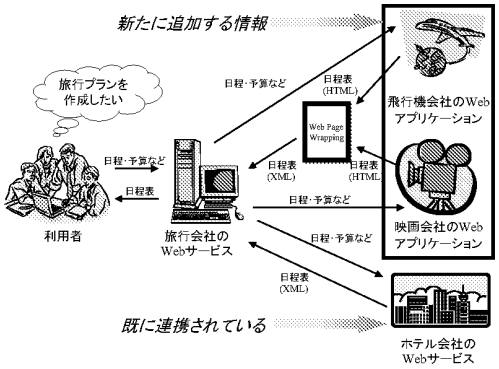

図 2: 旅行プラン作成サービス

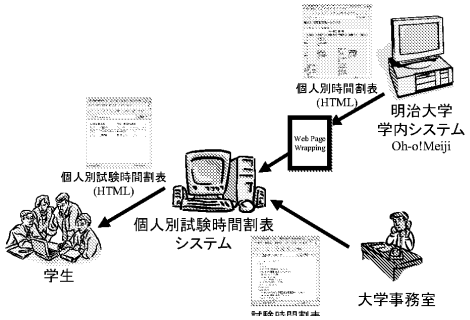

図 3: 個人別試験時間割表システム

である．

3　例題システムの開発

3.1　例題システムの概要

本稿では，Web ページラッピング技術の例題として，図 3 のような個人別時間割表と試験時間割表から個人別試験時間割表を作成するシステムをとりあげる．

現状では，個人別時間割表は，明治大学の学内システム Oh-o!Meiji[6] から HTML 形式で提供されている．また，試験時間割表については，学部内のすべての試験科目を掲載した印刷物を掲示する方法で提供されている．そのため，学生はこの膨大な量の科目の中から自分が履修している科目を探し出し，その科目の試験実施日を確認しなければならない．

そこで，個人別時間割表から学生の履修情報を調べて，試験実施スケジュールを各学生に特化した形 (以降，個人別試験時間割表と呼ぶ) で提供するサービスを例題としてとりあげ，プログラムを試作した．個人別時間割表は，ラッピング対象となる既存 Web アプリケーションの例である．また，試験時間割表は，Web サービスの例として位置づけ，XML 形式で提供されているものと仮定した．さらに，このシステムの利用形態を次のように想定する．

- システム利用者は学生とする
- システム管理者は大学の事務室とする

3.2　システム構成

２つの入力のうち，個人別時間割表は，既存の HTML 形式をそのまま利用し，Web ページラッピング技術を適用するとともに，試験時間割表は，既存の形式は無いので，XML 形式を設定し，Web サービス用のインターフェースとした．出力となる個人別試験時間割表は，学生が閲覧するので HTML 形式とする．

- **入力１：個人別時間割表** (HTML)
  学生の履修科目の情報を含む HTML 形式の Web ページ．図 4 のような Oh-o!Meiji で実際に表示されるものを使用．

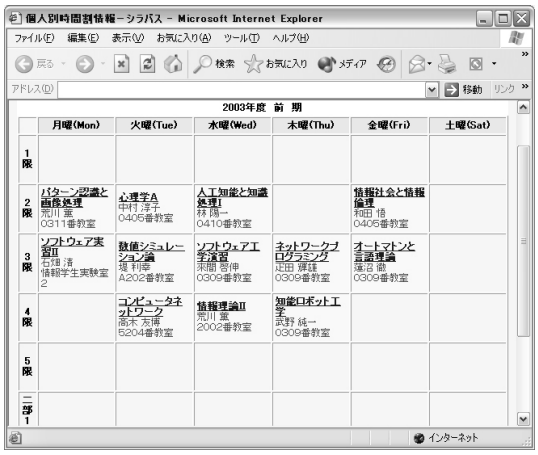

図 4: 個人別時間割表

- **入力２：試験時間割表** (XML)
  試験実施スケジュールの情報（科目名，日付，曜日，実施時間，教室など）を XML 化したもの．印

刷物の試験時間割表を手作業で XML 化する．

- **出力**：**個人別試験時間割表** (HTML)
  その学生が履修している科目の試験実施スケジュールの情報（科目名，日付，曜日，実施時間，教室など）を HTML 化したもの．

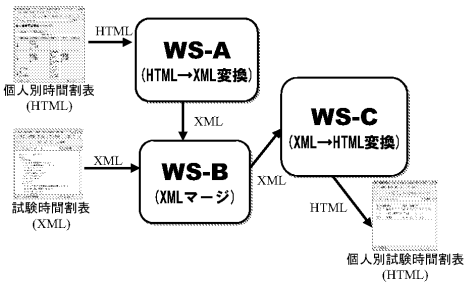

図 5: システムの構成

次に，システム本体は，図 5 のような 3 要素から成る構成とした．

- **WS-A：HTML → XML 変換処理**
  個人別時間割表 (HTML) から履修情報 (XML) を作成する．この部分が Web ページラッピング技術の主要部分である．
- **WS-B：XML マージ処理**
  履修情報 (XML) と試験時間割表 (XML) を照らし合わせて，両者に共通する科目情報（その学生の履修科目でかつ試験実施科目）を XML 化する．
- **WS-C：XML → HTML 変換処理**
  XML マージ処理の結果が XML 形式なので，それを視覚的に見やすい HTML 形式に変換する．

3.3 エンドユーザ支援方法

このシステムの管理者は，情報処理の専門家ではなく，エンドユーザ (大学の事務室) である．例えば，個人別時間割表や試験時間割表のフォーマットが変わっても，大学の事務室がシステム変更に対応できる必要がある．なお，大学の事務室はプログラミング経験の無いものとする．

そこで，エンドユーザがプログラミング言語を意識することなく，システム変更を行えるようにするために，基本的には処理手順をスクリプト記述させる方法とした．そして，できるだけビジュアルツールを導入して，習得の容易化を心がけた．

4 スクリプト化手法

4.1 WS-A:HTML → XML 変換処理

4.1.1 ラッピング処理の内容

ここでは，個人別時間割表 (HTML) から履修情報 (XML) を作成する処理を行う．

与えられた個人別時間割表 (HTML) は，以下のような形式を採用しており，科目名等を表す部分が履修状態によって適宜変更されるが，それ以外の表示形式に関しては対象とする学生が違っても統一されている．

```
<TR>
  <TH align=middle bgColor=#fff0f5 height=60>2<BR>限</TH>
  <TD bgColor=#f0fff0><A href="http://oh-o.meiji.ac.jp/zy_page.php?
   lcode=15573400&amp;tid=890085&amp;ttype=1&amp;tconf=1" target=_blank>
   <B>パターン認識と画像処理</B></A><BR><FONT color=#008000>荒川 薫<BR>
    ０３１１番教室<BR></FONT><!-- JIWARNO = 7508--><!-- KAMOKUCD = 15573400-->
  </TD>
```

その HTML 文書からデータを抽出して，作成した XML 文書は以下のような形式とする．

```
<?xml version="1.0" encoding="UTF-8"?>
<personaltimetable>
 <kamoku>
  <name>パターン認識と画像処理</name>
  <teacher>荒川 薫</teacher>
 </kamoku>
</personaltimetable>
```

4.1.2 XSLT を採用したスクリプト化の方法

HTML から XML を作成する手順を以下に示す．

手順 1:HTML の読み込み

手順 2:必要なデータの抽出

手順 3:XML の作成

この処理をスクリプト化する方法として，XSLT を採用する．XSLT は，手順 2 と手順 3 をサポートするスクリプト言語である．手順 1 に関しては，XSLT は HTML を入力として扱えないので，HTML → XHTML 変換を行うことにより，XSLT で HTML を扱えるようにする．このアプリケーションにおける HTML → XML 変換処理を記述した XSLT スタイルシートは以下のようになる．

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
 <xsl:output encoding="UTF-8" method="xml"/>
 <xsl:template match="/">
  <personaltimetable>
   <xsl:apply-templates
    select="/html[1]/body[1]/table[3]/tbody[1]/tr/td/a[1]/b[1]"/>
  </personaltimetable>
 </xsl:template>
 <xsl:template match="/html[1]/body[1]/table[3]/tbody[1]/tr/td/a[1]/b[1]">
  <kamoku>
   <name>
    <xsl:value-of select="/text()"/>
   </name>
   <teacher>
    <xsl:value-of select="../../font[1]/text()"/>
   </teacher>
  </kamoku>
 </xsl:template>
</xsl:stylesheet>
```

4.1.3 アプリケーション依存性に関する考察

本スクリプトを用いた場合の HTML → XML 変換処理は以下のような流れになる．

- XSLT スタイルシートの準備
- HTML を XHTML に変換
- XSLT スタイルシートの解析

これらの処理をアプリケーション依存性という観点から分類すると，スクリプトである XSLT スタイルシートを準備する部分だけがアプリケーションに依存しており，エンドユーザが記述する必要がある．それ以外はアプリケーションに依存しない処理として自動化できた．

4.1.4 スクリプト化に関する考察

エンドユーザ主導開発のための課題

XSLT スタイルシートは，プログラミング言語より習得が容易であり，記述方法も簡単であるが，抽出したいデータの位置を XPath で指し示す必要があったり，テンプレートの概念など，エンドユーザにもある程度のスキルが要求され，記述は容易でないと思われる．

ビジュアルツールの開発

この問題点に対し，HTML → XML 変換処理における XSLT スタイルシートを自動生成するビジュアルツールを開発した．これは HTML → XML 変換処理における XSLT スタイルシートを記述する為に必要な情報を，エンドユーザに選択または記述してもらい，その内容から XSLT スタイルシートを自動生成するツールである．このツールの処理の流れを以下に示す．

1. 対象となる HTML ファイル名の指定
2. 出力すべき XML の構造を定義
3. 抽出したいデータの選択
4. 自動生成した XSLT スタイルシートの保存

XML の構造は図 6 のような画面を用いて，ユーザに直接記述してもらう．HTML から抽出したデータを埋め込みたい位置なども，ここで指定する．抽出したい

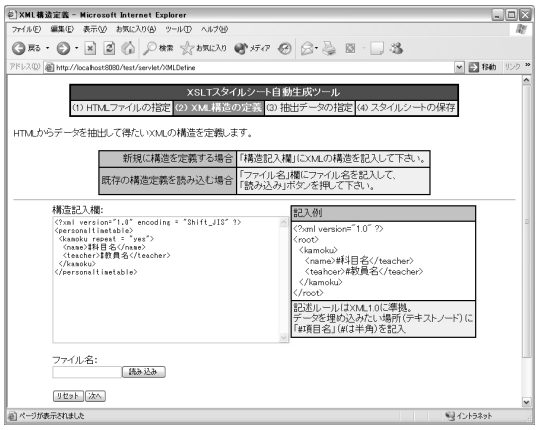

図 6: XML の構造を定義

データは図 7 のような画面で，選択させる．ページ下部に対象となる HTML の Web ページが表示され，各テキストの直前に CheckBox が付加されている．ユーザは，この CheckBox を用いて抽出したいデータを選択する．

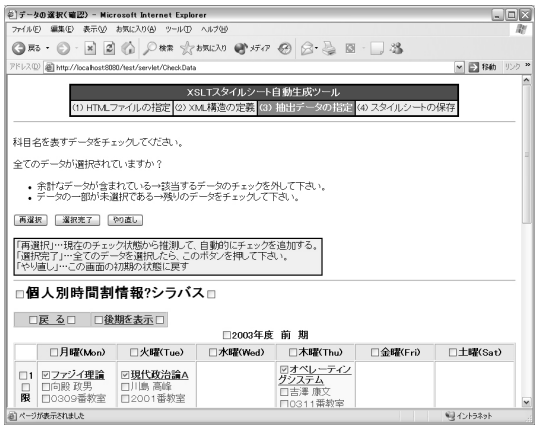

図 7: 抽出したいデータの選択

以上のユーザからの入力によって，XPath を算出し，テンプレートを自動生成する．

4.2 WS-B:XML マージ処理

4.2.1 マージ処理の内容

ここでは，履修情報 (XML) と試験時間割表 (XML) を照らし合わせて個人別試験時間割表 (XML) を作成する処理を行う．このアプリケーションで用いる試験時間割表は以下の構造となっている．

```
<?xml version="1.0">
<examine>
 <kamoku>
  <name>ファジィ理論</name>
  <teacher>向殿 政男</teacher>
  <grade>3・4</grade>
  <class>14・15</class>
  <room>A201</room>
  <date>7/28</date>
  <time>1</time>
 </kamoku>
 <kamoku>
  <name>ネットワークプログラミング</name>
  <teacher>疋田 輝雄</teacher>
  <grade>3</grade>
  <class>14・15</class>
  <room>0405</room>
  <date>7/24</date>
  <time>3</time>
 </kamoku>
</examine>
```

前述の履修情報と照らし合わせて，両者に共通する科目を抽出し，以下のような個人別試験時間割表を XML として出力させる．

```
<?xml version="1.0">
<personalized-examine-timetable>
 <kamoku>
  <name>ファジィ理論</name>
  <teacher>向殿 政男</teacher>
  <grade>3・4</grade>
  <class>14・15</class>
  <room>A201</room>
  <date>7/28</date>
  <time>1</time>
 </kamoku>
</personalized-examine-timetable>
```

4.2.2 XSLT を採用したスクリプト化の方法

XML マージ処理も，HTML → XML 変換処理と同様に，XSLT によるスクリプト化の方式をとる．このアプリケーションの XML マージ処理は以下のような処理手順で XSLT スタイルシートに記述されている．

1. 入力となる 2 つの XML ファイルのノード集合を変数へ格納する．
2. 2 つの入力から比較対象となる要素 (この例では科目要素) を一つずつ取り出す．この際，取り出した要素が先頭から数えて何番目であるかを変数に記憶する．
3. その要素内の比較したい要素 (この例では科目名と教員名) の持つデータが同じであるかを調べる．
4. 同じであった場合は，指定した形式に従ってデータを出力する．

```
<?xml version="1.0" encoding="UTF-8" ?>
<xsl:stylesheet version="1.0"
                xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:output method="xml" version="1.0" encoding="UTF-16" indent="yes" />
 <xsl:variable name="first" select="document('xml-1.xml')" />
 <xsl:variable name="second" select="document('xml-2.xml')" />
 <xsl:template match="/">
  <personalized-examine-timetable>
   <xsl:for-each select="$first/examine[1]/kamoku">
    <xsl:variable name="firstposition" select="position()" />
    <xsl:for-each select="$second/personaltimetable[1]/kamoku">
     <xsl:variable name="secondposition" select="position()" />
      <xsl:if test="$first/examine[1]/kamoku[$firstposition]/name[1]
                = $second/personaltimetable[1]/kamoku[$secondposition]/name[1]" >
      <xsl:if test="$first/examine[1]/kamoku[$firstposition]/teacher[1]
                = $second/personaltimetable[1]/kamoku[$secondposition]/teacher[1]">
       <kamoku>
        <name>
         <xsl:value-of select="$first/examine[1]/kamoku[$firstposition]/name[1]"/>
        </name>
        <teacher>
         <xsl:value-of select="$first/examine[1]/kamoku[$firstposition]/teacher[1]"/>
        </teacher>
        <grade>
         <xsl:value-of select="$first/examine[1]/kamoku[$firstposition]/grade[1]"/>
        </grade>
        <class>
         <xsl:value-of select="$first/examine[1]/kamoku[$firstposition]/class[1]"/>
        </class>
        <room>
         <xsl:value-of select="$first/examine[1]/kamoku[$firstposition]/room[1]"/>
        </room>
        <date>
         <xsl:value-of select="$first/examine[1]/kamoku[$firstposition]/date[1]" />
        </date>
        <time>
         <xsl:value-of select="$first/examine[1]/kamoku[$firstposition]/time[1]" />
        </time>
       </kamoku>
      </xsl:if>
     </xsl:if>
    </xsl:for-each>
   </xsl:for-each>
  </personalized-examine-timetable>
 </xsl:template>
</xsl:stylesheet>
```

XSLT は XML ドキュメントを別の XML ドキュメントに変換するための言語であるが，XPath の機能を用いることで複数の XML 文書を取り扱うことや，変換の際にデータを加工することが可能である．この例では以下のような XPath の関数を用いている．

document 関数 外部の XML 文書へのアクセスを可能にする関数．この例のスタイルシートでは，5 行目で xml-1.xml を変数 `first` に，6 行目で xml-2.xml のを変数 `second` に割り当てている．

position 関数 ある親要素の直下の子である同じ要素の中で，現在取り扱っているノードが何番目かを返す関数．この例のスタイルシートでは，10 行目で入力 xml-1.xml に含まれている科目要素内の現在位置を変数 `firstposition` に，12 行目で入力 xml-2.xml に含まれている科目要素内の現在位置を変数 `secondposition` に記憶している．

なお，実際の科目名と教員名のデータの比較は，14〜17 行目でこれらの変数を用いて行っている．

4.2.3 アプリケーション依存性に関する考察

今回の XML マージ処理では，XSLT スタイルシートの定義内容そのものはアプリケーションに依存する

ので，エンドユーザが記述する必要があるが，その処理系はアプリケーション非依存である．

4.2.4　スクリプト化に関する考察

エンドユーザ主導開発のための課題

本研究では，エンドユーザ (大学の事務室) は，プログラミング経験はないと想定している．そこで，本スクリプトのように，変数 (xsl:variable) や繰り返し処理 (xsl:for-each) といった概念を用いて，マージ処理を記述した XSLT スタイルシートを記述することは，エンドユーザにとって難しいと思われる．

ビジュアルツールの開発

この問題点に対し，XML マージ処理における XSLT スタイルシートを自動生成するビジュアルツールを開発した．これは，マージ処理用の XSLT スタイルシートを記述するために必要な情報を，次のような手順でエンドユーザに選択または入力してもらい，その内容から XSLT スタイルシートを自動生成するツールである．

1. 入力 XML ファイルの指定
2. 比較対象要素の選択
3. 比較条件の入力
4. マージ処理の出力となる XML 文書の構造定義
5. 出力 XML 文書に埋め込むデータの選択
6. 生成する XSLT スタイルシートの保存先の指定

基本的には，HTML → XML 変換処理に用いたツールと同様の操作方法とした．まず，手順 1 で入力 XML ファイルを指定した後，手順 2 では，その入力 XML の中から比較対象要素を選択する．具体的には，図 7 と同じように，入力 XML を HTML に変換して表示し，比較対象の候補となる要素の直前に付加したチェックボックスを選択するようにした．

手順 3 では，手順 2 で選択された比較対象要素を用いて，比較の条件式を入力する．図 8 はその様子を示したものである．図の上部には，4 つの選択された比較対象要素の XPath を以下のように表示している．

最初の 2 つは，1 番目の入力 XML(試験時間割表) から選択された科目名と教員名である．あとの 2 つは，2

図 8: 比較条件の入力

省略記号	XPath
F0	/examine[1]/kamoku[n]/name[1]
F1	/examine[1]/kamoku[n]/teacher[1]
S0	/personaltimetable[1]/kamoku[n]/name[1]
S1	/personaltimetable[1]/kamoku[n]/teacher[1]

番目の入力 XML(個人別時間割表) から選択された科目名と教員名である．XPath の左側の省略記号は，条件式を簡潔に記述するために導入したものである．この例題では，2 つの入力 XML の科目名の一致という条件を $F0 = S0$ という式で入力できる．図では，この条件式は入力済みの式として，以下のような XPath を用いた式に変換されて，図の中ほどに表示されている．

```
$first/examine[1]/kamoku[$firstposition]/name[1]
= $second/personaltimetable[1]/kamoku[$secondposition]/name[1]
```

そのすぐ下のテキスト入力エリアには，最初の条件に続く AND 条件として，2 つの入力 XML の教員名の一致を意味する $F1 = S1$ という式が入力されている．

次に，図 6 と同じように，手順 4 でマージ処理の出力となる XML 文書の構造を定義し，手順 5 で，その出力に埋め込むデータを選択する．最後に，手順 6 では，手順 1 から手順 5 までの情報に基づいて自動生成した XSLT スタイルシートの保存先を指定する．

4.2.5　汎用性に関する考察

XSLT スタイルシートを用いた XML マージ処理方式の汎用性について評価するため，ここで旅行プランの問い合わせに応じるシステムを例題として考え，以下のような利用シナリオを想定する．

「歌手Kの秋季コンサートに１０月中に行きたいので，C社で扱うそのコンサートチケットの予約が可能で，当日のF社の飛行機に空席があり，Hホテルに空室がある日を調べたい.」

この場合の Web サービス連携システムの概要は以下のようになる.

「C社とF社とHホテルは，それぞれ個別に空席あるいは空室の情報を XML 形式で提供しているとして，３種類の情報をマージして，上記の問い合わせに応じる.」

このシステムの XML とそれに含まれる情報は以下のように想定できる.

xml-1.xml 利用者の希望日程 (日付)

xml-2.xml C社のコンサートチケット予約状況 (日付)

xml-3.xml F社の飛行機の空席状況 (日付，出発時刻)

xml-4.xml Hホテルの空室状況 (日付，部屋のランク，料金)

シナリオから，利用者の第一の目的は 10 月中にコンサートに行くことなので，以下のようなマージ処理をすればよい.

1. **xml-1.xml と xml-2.xml のマージ**
   利用者の希望日程の XML(xml-1.xml) とコンサートチケットの予約状況の XML(xml-2.xml) を入力として，コンサートが予約可能である日程の一覧を XML(xml-5,xml) に出力するマージ処理.
2. **xml-5.xml と xml-3.xml のマージ**
   コンサートが予約可能な日程一覧の XML(xml-5.xml) とホテルの空室状況の XML(xml-3.xml) を入力として，コンサートとホテルが予約可能である日程の一覧を XML(xml-6.xml) に出力するマージ処理.
3. **xml-6.xml と xml-4.xml のマージ**
   コンサートとホテルが予約可能な日程一覧の XML(xml-6.xml) と飛行機の空席状況の XML(xml-4.xml) を入力として，利用者の要求を満たす日程の一覧を XML(xml-7.xml) に出力するマージ処理.

以上のマージ処理は，個人別試験時間割表作成に用いたマージ処理と同一レベルなので，旅行プランの問い合わせに応じるシステムについても XSLT スタイルシートを用いたマージ処理で処理可能であることから，今回の方式の汎用性が確認できた.

5 おわりに

本稿では，エンドユーザ主導によるサービス連携の実現方式として，個人別時間割表と試験時間割表から個人別試験時間割表を作成するシステムを試作し，既存の Web アプリケーションを Web サービスとして連携させるための Web ページラッピング方式，および複数の Web サービスのマージ方式を開発した．そして，アプリケーション依存部分と非依存部分を明確に分離し，後者を自動化するとともに，前者については，スクリプトによる記述とそのビジュアルツール化により，エンドユーザ主導の Web サービス連携が可能であることを確認した.

参考文献

[1] Takeshi CHUSHO, Katsuya FUJIWARA, Hisashi ISHIGURE and Kei SHIMADA : A Form-based Approach for Web Services by Enduser-Initiative Application Development, SAINT2002 Workshop (Web Service Engineering), IEEE Computer Society, pp.196-203（Feb. 2002).

[2] (株) 日本ユニテック : Web サービス技術-基礎と実践-, 技術評論社 (2002).

[3] 米持幸寿 : Web サービス完全解説, 翔泳社 (2002).

[4] 湯浅 亮祐:Web サービス連携における Web ページラッピング技術の提案と評価, 明治大学 2003 年度修士論文.

[5] 丸山　宏, 田村健人, 浦本　直彦 : XML と Java による Web アプリケーション開発-第 2 版-, ピアソン・エデュケーション (2002).

[6] 明治大学 : Oh-o!Meiji
http://www.oh-o.meiji.ac.jp/