社団法人 情報処理学会　研究報告
IPSJ SIG Technical Report

2004－SE－144　(16)
2004／3／19

Webサービス連携のためのWebページラッピング技術の提案と評価

湯浅　亮祐[†]　　中所　武司[†]

概要

最近，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．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が，既存のWebアプリケーションとの連携が当面の大きな課題である．そこで，その解決方法として，Webページラッピング技術を開発した．例題として，学内の学生向けWebシステム（Oh-o!Meiji）が提供するHTML形式の個人別時間割表と学部事務室がら公表される試験時間割表をXML化したものからHTML形式の個人別試験時間割表を作成するシステムを試作した．ここで導入したHTMLからXMLへのラッピング処理と複数のXML文書のマージ処理とXMLからHTMLへの変換処理のアプリケーション依存部分のみをスクリプト言語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．

Web Page Wrapping Technology for Web Service Integration

Ryousuke Yuasa[‡]　and　Takeshi Chusho[‡]

Abstract

One problem for development of Web applications by Web service integration, is how to communicate with conventional Web applications. This paper proposes the Web page wrapping method as a solution. Technologies for HTML-to-XML wrapping, XML merging and XML-to-HTML transformation are developed in an application which produces a HTML-base individual examination schedule from a HTML-base individual class schedule and a XML-base whole examination schedule. While application-specific processes are described in script languages by end-users, 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

[†]明治大学大学院理工学研究科 ソフトウェア工学研究室
[‡]Software Engeneering Laboratory, Graduate School of Science and Technology, Meiji University.

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．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が，既存の Web アプリケーションとの連携が当面の大きな課題である [1][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，次にその解決策として Web ページラッピング技術を提案する [4]．そして，例題システムの開発を通して，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．

この場合のアプリケーションは，人間を相手にしている．つまり，データはクライアントと Web サーバ間で交換され，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．

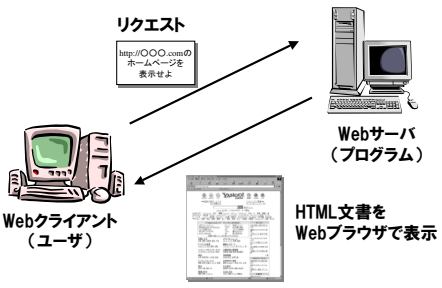


図 1: 従来の Web アプリケーション

そこで，図 2 のように HTML よりも機械処理が容易な XML を使ってアプリケーション間の連携を行うという方法が Web サービスである [5]．

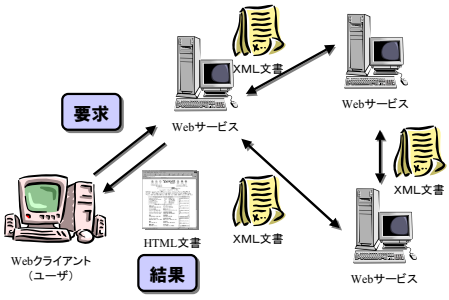


図 2: Web サービスによるアプリケーションの連携

2.2　Web サービス化の問題点

あるアプリケーションを Web サービスとして提供する場合について考える．新規に Web サービスを作成する場合は，Web サービスの仕様に合わせて作成すればよいが，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場合には，そのアプリケーションの仕様を Web サービス標準にする必要がある．具体的には，処理結果として HTML を出力している部分を，XML を出力させるように変更しなくてはならない．従来の手法では，アプリケーションに直接手を加えなければ，その変更が出来なかった．

その作業はエンドユーザにとって負担であり，その負担を軽減するためには，新たなアプリケーション連携技法が必要である．

3　Web ページラッピング技術の提案

Web ページラッピング技術とは，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技術である．

例えば，実際に Web ページラッピング技術を適用する場合の事例として，図 3 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いるとする．

現在はホテルの予約情報検索サービスと連携してお

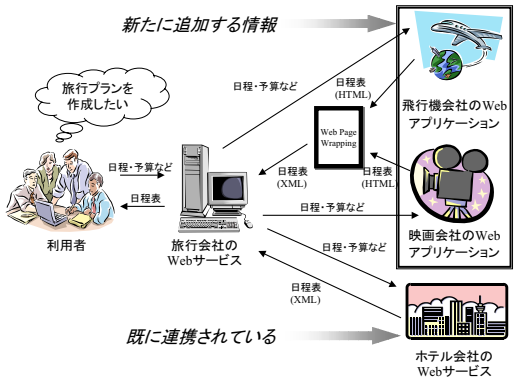


図 3: 旅行プラン作成サービス

り，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．

この旅行プランに，映画の上映情報と飛行機のフライト情報を含めたいという要望があったとする．しかも，映画会社と航空会社は各々の情報を Web ページ (HTML) としては表示しているが，Web サービスとして提供していないとする．その場合でも，Web ページラッピング技術を使えば，上映情報とフライト情報を XML 化して旅行プラン作成サービスと連携させることができ，ユーザには宿泊情報，フライト情報，上映情報を含む旅行プランを提示することができるようになる．

4　例題システムの開発

4.1　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 4 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紙媒体の時間割を掲示するといった形で提供されている．明治大学内で一般的に掲示されている試験時間割表には，各学部毎に，試験が実施されるすべての科目が表示されている．この膨大な量の科目の中から各学生は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．

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) で提供すれば，ユーザーである学生が履修している科目のみ表示されるような仕組みになっているので，これにより履修科目を探すといった手間を省くことができる．

この例題では，個人別時間割表をラッピング対象となる既存 Web アプリケーションの例として，また，試験時間割表を Web サービスの例として位置づけ，プログラムを試作した．

なお，このシステムの利用形態を以下のように想定する．

- 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

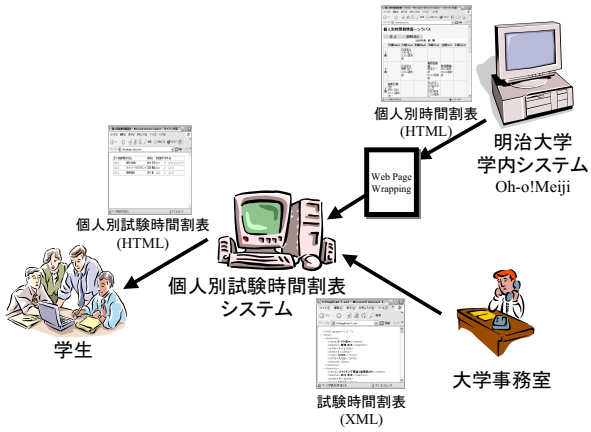


図 4: 個人別試験時間割表システム

4.2　システム構成案

まず，入出力に関して以下のように決定した．

個人別時間割表に関しては，既存の HTML 形式をそのまま利用することにして，Web ページラッピング技術を適用する．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．個人別試験時間割表に関しては，見易さという観点や，実際に学生が閲覧することを考慮して，HTML 形式を用いることにした．

- 入力
  - **個人別時間割表** (HTML)
    学生の履修している科目の情報がある HTML

形式の Web ページ．図 5 のような Oh-o!Meiji で実際に表示されるものを使用．

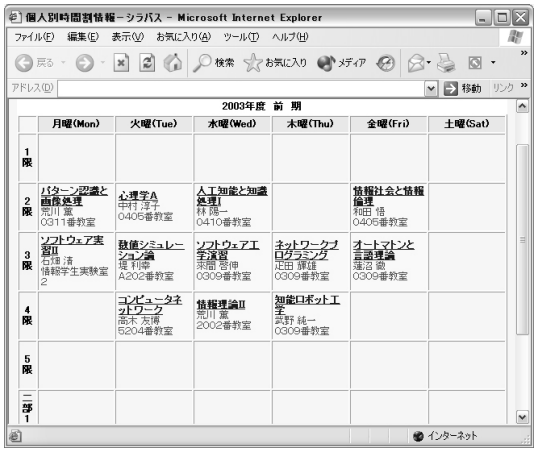


図 5: 個人別時間割表

– **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．紙媒体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 化したものを出力する．

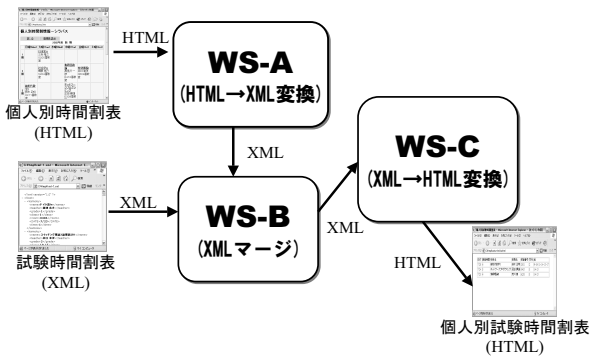


図 6: システムの構成

次に，システム本体は，図 6 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．

4.3　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の一手法として，システムの各構成要素の処理手順をスクリプト記述させる方法を開発した．それは，スクリプト言語はプログラミング言語と比べて命令の種類が少なく，また，難易度も低いことからエンドユーザにとって，習得が容易であるためである．

5　スクリプト化手法

5.1　WS-A:HTML → XML 変換処理

5.1.1　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>
```

5.1.2 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>
```

5.1.3 スクリプト化に関する考察

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．

5.1.4 スクリプト作成支援ツール

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る．そこで，HTML → XML 変換処理における XSLT スタイルシート自動生成ツールを考案した．

XSLT スタイルシートを記述する為には，対象 HTML に対して以下のような情報が必要になる．

- **(1) 何処に (Where)**
  データが存在する場所を示す XPath
- **(2) 何が (What)**
  (1) の場所が指し示すデータの項目名
- **(3) どのような構造 (How)**
  出力する XML の構造（タグ名など）

XSLT スタイルシート自動生成ツールでは，以上の 3 つの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ようにした．このツールの処理の流れを以下に示す．

- 対象となる HTML ファイル名の指定
- 出力させたい XML の構造を定義
- 抽出したいデータの選択
- 自動生成した XSLT スタイルシートの保存

XML の構造は図 7 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

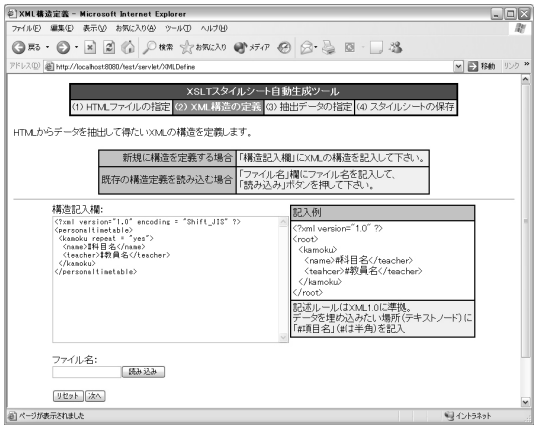


図 7: XML の構造を定義

データは図 8 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．

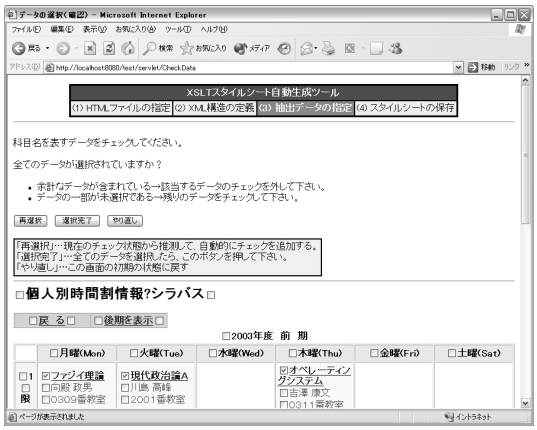


図 8: 抽出したいデータの選択

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．

5.2 WS-B:XML マージ処理

5.2.1 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>
```

5.2.2 独自のスクリプト言語を用いた方法

XML マージ処理をスクリプト化する場合にも，XSLT を用いた方法が考えられる．しかし，XSLT は 2 つの XML のマージ処理をする場合などは，変数を用いるなど，プログラミング言語ライクな記述方法になってしまうといった欠点がある．

そこで，プログラミング言語を意識させない記述方法として，独自のスクリプト言語 XML Merger Script を開発した．この言語は，2 つの XML ファイルを入力とし，マージ結果を 1 つの XML として出力させるものである．以下に，このスクリプト言語に含まれる情報を示す．

- **入出力ファイル名**
  - **xml-1.xml(入力 XML1 のファイル名)**
  - **xml-2.xml(入力 XML2 のファイル名)**
  - **xml-3.xml(出力 XML のファイル名)**
- **出力指定**
  - **マージをする際の条件式**
  - **条件が一致した場合の出力内容**

このアプリケーションで実際に使われる XML Merge Script を以下に示す．2 つの入力 XML 文書の中からマージする際に比較する要素 (この例では「科目名」と「教員名」) を指定して，両者が一致したら条件式が真となる．

```
<?xml version="1.0"?>
<xms:script xmlns:xms="http://www.se.cs.meiji.ac.jp/xms">
 <xms:input_xml>
  <xms:xml_1>c:\tmp\xml-1.xml</xms:xml_1>
  <xms:xml_2>c:\tmp\xml-2.xml</xms:xml_2>
 </xms:input_xml>
 <xms:output_xml>
  <xms:xml_3>c:\tmp\xml-3.xml</xms:xml_3>
 </xms:output_xml>
 <xms:context>
  <personalized-examine-timetable>
   <xms:select opeland="AND">
    <xms:nodelist>
     <xms:xml_1>//kamoku</xms:xml_1>
     <xms:xml_2>//kamoku</xms:xml_2>
    </xms:nodelist>
    <xms:if>
     <xms:xml_1>/name/text()</xms:xml_1>
     <xms:xml_2>/name/text()</xms:xml_2>
    </xms:if>
    <xms:if>
     <xms:xml_1>/teacher/text()</xms:xml_1>
     <xms:xml_2>/teacher/text()</xms:xml_2>
    </xms:if>
    <xms:output>
     <kamoku>
      <name>
       <xms:extract select="1">/name/text()</xms:extract>
      </name>
      <teacher>
       <xms:extract select="1">/teacher/text()</xms:extract>
      </teacher>
      <grade>
       <xms:extract select="1">/grade/text()</xms:extract>
      </grade>
      <class>
       <xms:extract select="1">/class/text()</xms:extract>
      </class>
      <room>
       <xms:extract select="1">/room/text()</xms:extract>
      </room>
      <date>
       <xms:extract select="1">/date/text()</xms:extract>
      </date>
      <time>
       <xms:extract select="1">/time/text()</xms:extract>
      </time>
     </kamoku>
    </xms:output>
   </xms:select>
  </personalized-examine-timetable>
 </xms:context>
</xms:script>
```

5.2.3 スクリプト化に関する考察

本スクリプトを用いた場合の XML マージ処理は以下のような流れになる．

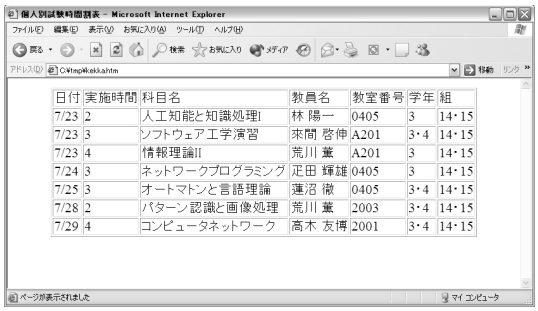
- XML Merge Script の準備
- XML Merge Script の解析

そ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 XML Merge Script 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，それ以外の解析処理部分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．

5.3 WS-C:XML → HTML 変換処理

5.3.1 変換処理の内容

ここでは，XML マージ処理の結果を HTML 形式に変換する処理を行う．その変換の結果，図 9 のような Web ページを表示させるための HTML 文書が作成されるものとした．



日付	実施時間	科目名	教員名	教室番号	学年	組
7/23	2	人工知能と知識処理I	林 陽一	0405	3	14・15
7/23	3	ソフトウェア工学演習	來間 啓伸	A201	3・4	14・15
7/23	4	情報理論II	荒川 薫	A201	3	14・15
7/24	3	ネットワークプログラミング	疋田 輝雄	0405	3	14・15
7/25	3	オートマトンと言語理論	蓮沼 徹	0405	3・4	14・15
7/28	2	パターン認識と画像処理	荒川 薫	2003	3・4	14・15
7/29	4	コンピュータネットワーク	高木 友博	2001	3・4	14・15

図 9: 個人別試験時間割表 (HTML)

5.3.2 XSLT を採用したスクリプト化方法

こちらも，HTML → XML 変換処理と同様に，XSLT によるスクリプト化方法を採用する．本来，XSLT は XML → HTML 変換の為に使われることが多いので，その変換に使う XSLT スタイルシートに関しては，作成ツールが既に多数存在している．

5.3.3 スクリプト化に関する考察

本スクリプトを用いた場合の XML → HTML 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
- XSLT スタイルシートの解析

そ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，それ以外の解析処理部分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．なお，この処理に用いる XSLT スタイルシートに関しては，既存のツールで作成するものとする．

6　考察

提案した Web ページラッピング技術を使うことにより，例題システムにおいて，大学の事務室がどの程度までの作業を行えるのか，考察する．なお，大学の事務室は，XML Merge Script や XSLT 自動生成ツールについては，それを使いこなすことができるものとし，プログラミング経験は全くないものと仮定する．

大学事務室が出来ることと出来ないことの例を以下に挙げる．

- **出来ることの例**
  - 個人別時間割表 HTML の出力形式に変更があった場合の対応
  - 個人別試験時間割表 HTML の出力形式の変更要請があった場合の対応
  - 試験時間割表 XML の形式及び内容の変更をする
- **出来ないことの例**
  - システムの構成を変更する場合の対応．例えば，個人別成績表も入力させるような構成にする場合など．
  - 個人別試験時間割表 HTML に,「試験時間割表の閲覧」以外の機能を持たす場合の対応．例えば、表示された科目の詳細を表示する機能を持つボタンを付け加える，など．

これらをまとめると，出力形式や入力形式の変更には対応できるが，アプリケーションの機能を変更する場合には，大学の事務室だけでは対応できないということになる．

なお，本研究で開発したプログラムの WS-A，WS-B，WS-C は，アプリケーション非依存なので，これらの組み合わせを変更するだけで対応できるものも多いと思われる．例えば図 10 のように，例題システムは 2 入力 1 出力だが，3 入力 1 出力に変更があった場合には，その組み合わせを記述するワークフロー言語を用いることにより，直接アプリケーションの内部処理を変更しなくても，外部記述による組み合わせの変更が可能となり，大学の事務室で対応することが出来る．従って，ワークフロー言語の採用が Web ページラッピング技術の今後の課題である．

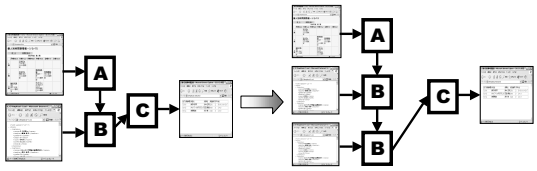


図 10: ワークフロー言語を用いた機能変更の例

7　おわりに

本稿では，エンドユーザによる，既存 Web アプリケーションを Web サービスとして連携させる手法として，Web ページラッピング技術を提案した．その際に，XSLT や XML Merge Script というスクリプト言語を用いることで，アプリケーション依存部分と非依存部分の切り分けることができ，後者を自動化できた．

よって，この Web ページラッピング技術がエンドユーザによる Web サービス連携を可能とする手法であることが確認できた．

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] 株式会社日本ユニテック：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/