解説

Webプログラミング・フレームワーク

下村 隆夫　simomura@is.tokushima-u.ac.jp
徳島大学工学部知能情報工学科

高橋 宗雄　takahasi@cc.toin.ac.jp
桐蔭横浜大学

池田 建司　ikeda@is.tokushima-u.ac.jp
徳島大学工学部知能情報工学科

最上 義夫　moga@is.tokushima-u.ac.jp
徳島大学工学部知能情報工学科

インターネットを利用したビジネスの利便性から，多くのWebアプリケーションが開発され，サービスに供されている．Webアプリケーションの開発には，さまざまな統合開発環境が用いられる．現在，世界には70,000を超えるオープンソースソフトウェアを開発するプロジェクトがあり，Webアプリケーション開発支援についても多くのプロジェクトが進められている．Webプログラミング・フレームワークは，品質の高いWebアプリケーションを容易に開発するための枠組みであり，Webプログラミング手順とそれをサポートする開発･実行環境からなる．本稿では，Webアプリケーションの開発，保守を容易にするWebプログラミング・フレームワークについて解説する．

はじめに

インターネットを利用したビジネスの利便性から，多くのWebアプリケーションが開発され，サービスに供されている．Webアプリケーションの開発には，さまざまな統合開発環境が用いられる．現在，世界には70,000を超えるオープンソースソフトウェアを開発するプロジェクトがある．その中でも，特に，Apache Software Foundation[1]は，Webアプリケーションの開発を支援するための多くのプロジェクトを進めている．

通常のプログラムとは異なり，一般に，Webアプリケーションでは，Webブラウザからのフォームデータの受け取り，値の検証，エラーメッセージの表示，セッションの管理，データベースの操作，ビジネスロジック(業務処理)，ページの遷移，Webブラウザに返すページの生成等の各種の処理が必要となる．これらの処理の記述を容易にし，かつ，サーバソフトウェアの保守を容易にすることが大変重要となる．Webプログラミング・フレームワークは，品質の高いWebアプリケーションを容易に開発するための枠組みであり，Webプログラミング手順とそれをサポートする開発・実行環境からなる．

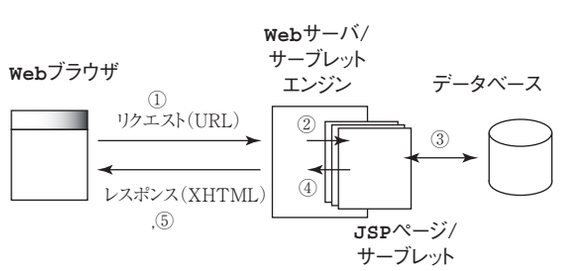


図-1　Webアプリケーションの構成

Webアプリケーション

ここでは，Webアプリケーションの構成について説明し，簡単なオンラインショップWebアプリケーションの作り方について述べる．

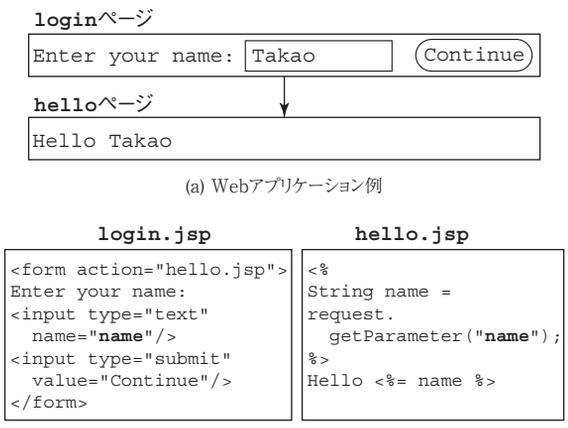
Webアプリケーションの構成

Webアプリケーションは，Webブラウザによりアクセスされるサーバ上のソフトウェアである．**図-1**に示すように，①サーバ上のJSPページ，サーブレット等のプログラムを指定して，Webブラウザがリクエストをサーバに送信すると，②サーバ上で，そのプログラムが実行され，③データベース等の検索，更新を行い，④処理結果を表示するためのXHTML文書を生成し，⑤それをレスポンスとしてWebブラウザに送り返す．

たとえば，Webブラウザ上でユーザが次のURLを指定すると，

http://server/webap/login.jsp

Webブラウザは，次のようなリクエストをWebサーバ



(a) Webアプリケーション例

login.jsp

```
<form action="hello.jsp">
Enter your name:
<input type="text"
  name="name"/>
<input type="submit"
  value="Continue"/>
</form>
```

hello.jsp

```
<%
String name =
request.
  getParameter("name");
%>
Hello <%= name %>
```

(b) JSPページによる実現

図-2　JSPによるWebプログラミング

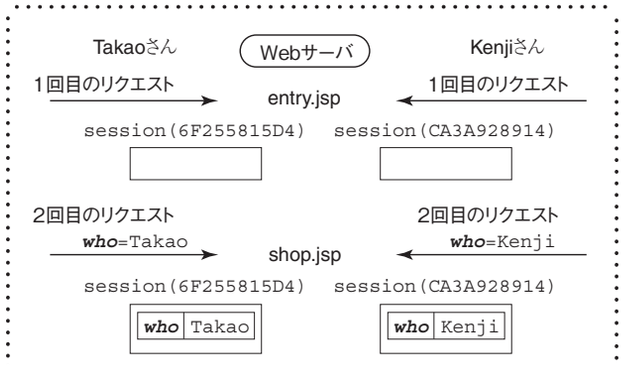


図-3　セッション管理の仕組み

(serverマシンのポート番号80) に対して送信する.

GET /webap/login.jsp HTTP/1.1

host: server

このリクエストは，リクエスト行とヘッダ行からなっており, GETリクエストで, Webサーバの管理するリソースのパス名(/webap/login.jsp)を指定している．Webサーバは，このリクエストを受け取ると，指定されたサーバ上のJSPページ (login.jsp) を実行し，JSPページによって生成されたXHTML文書を，次のようなレスポンスとしてWebブラウザに送り返す.

HTTP/1.1 200 OK

Content-Type: text/html

<html>...</html>

レスポンスは，ステータス行，(複数の)ヘッダ行，および，メッセージ本体からなる．Webブラウザは，このレスポンスを受け取り，ヘッダ行からコンテンツ種別を認識し，メッセージ本体として送られたXHTML文書を解釈してウィンドウに表示する.

サーバ上に配置されたソフトウェアには，サーバからダウンロードされてWebブラウザ上で動作するスクリプトやアプレット等のプログラムを含む場合もある.

◘JSPページを用いたWebプログラミング

図-2(a)に示す簡単なWebアプリケーションを記述してみる．最初のloginページに名前を入力し,「Continue」ボタンをクリックすると，次のhelloページに，入力された名前を"Hello"に続いて表示する．JSPページにより実現した例を**図-2**(b)に示す．JSPページは，XHTML文書の中に<% … %>タグで囲んでJava言語のソースコードを記述できるようにしたものである．また，<%= … %>タグで囲むと，式の値を出力することができる．XHTML文書では，常に同じ内容のWebページが表示されるが，JSPページを用いると，動的に変わる内容を表示することができる.

login.jspページでは，XHTML言語を用いて，フォーム (<form>タグ) を記述し，その中に名前を入力するためのテキストフィールド (<input>タグ) を配置する．テキストフィールドに入力されたデータはnameというパラメータが付けられ，formタグのaction属性で指定されたサーバ上のJSPページhello.jspに送信される．hello.jspページでは，パラメータnameの値を受け取り，"Hello"に続いてその値を表示している.

◘セッション管理

オンラインショップ，予約システム等のWebアプリケーションでは，ユーザがWebサイトにアクセスすると，リンクされたページを次々と遷移しながら，購入，予約等の処理を行う．Webアプリケーション側では，どのユーザからのアクセスであるかを認識し，これらの一連の処理 (セッション) を管理する必要がある[6].

図-3に示すように，最初にentry.jspページにアクセスし，次に，そのページからリンクされているshop.jspページにアクセスする場合を考える．最初にentry.jspページにアクセスしたときに，固有のセッションIDを持つsessionオブジェクトが生成される．たとえば，Takaoさんがentry.jspページにアクセスしたときには，"6F255815D4"というIDを持つsessionオブジェクトが生成される．名前を入力して，次に，shop.jspページに遷移した場合には，Webアプリケーション側では，このユーザ固有のsessionオブジェクトを識別し，その中にユーザの名前を格納することができる．sessionオブジェクトの識別が可能となるのは，ユーザがアクセスするごとに，WebブラウザがWebサーバに対してユーザに固有のセッションIDを送ってくるからである.

図-4に示すように，最初にentry.jspページにアクセスしたときに，Webサーバは，Set-Cookieというヘッダにセッション IDを指定して，レスポンスを返す．こ

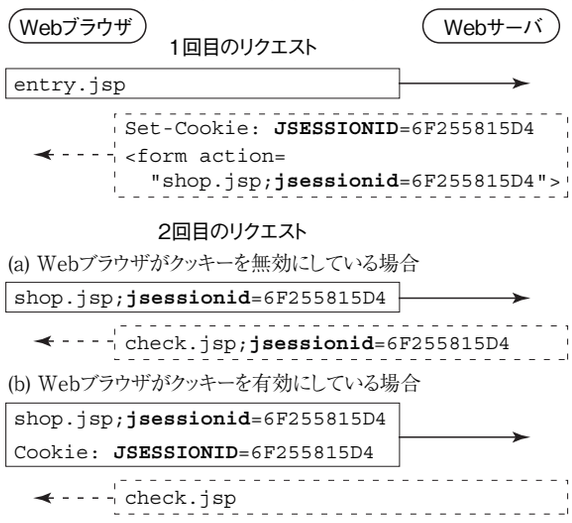


図-4　セッションIDの送信

れは，Webブラウザに対する指示であり，次に同じサイトにアクセスする場合には，Cookieヘッダにこのセッション ID を指定してリクエストを送信してもらうことを示す．次の shop.jsp ページへのリンクにも，セッション ID を付加する．ユーザが，次に shop.jsp ページへアクセスすると，Web ブラウザは，リクエストとともに，このセッション ID を Web サーバへ送信する．セッション ID を Web サーバへ送信する方法は 2 つある．(a) Web ブラウザによってはクッキー機能を無効にしている場合もあるので，この場合には，shop.jsp ページへのリンクに付加されたセッション ID を送信する．(b) クッキー機能が有効になっている場合には，Cookie ヘッダにセッション ID を指定してリクエストを送信する．その場合には，Web サーバから送り返される次のページ（check.jsp）へのリンクには，セッション ID は付加されない．

◘Web アプリケーション例

簡単なオンラインショップ Web アプリケーションを JSP ページを用いて作成してみよう．**図-5** に示すように，①ユーザが entry ページで名前を入力すると，②shop ページで，フルーツの一覧を表示し，ユーザがその中の 1 つを選択すると，③check ページで，選択されたフルーツと，最初に入力された名前を表示する．

オンラインショップの 3 つの JSP ページを**図-6** に示す．entry.jsp ページの response.encodeURL("shop.jsp") は，セッション ID を付加した URL を返す．shop.jsp ページの session.setAttribute("who", who) は，送信されてきた名前を session オブジェクトに格納する．また，"select name, image, price from fruit" という SQL 文を実行して，DB テーブル fruit からフルーツの名前，画像ファイル名，価格を取り出し，<table> タグを用いて，それらを表示

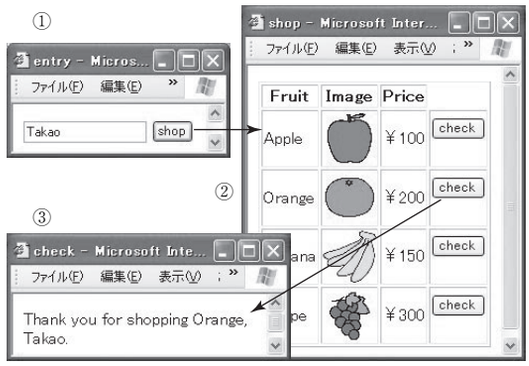


図-5　簡単なオンラインショップ例

entry.jsp

```
<form action=
  "<%= response.encodeURL("shop.jsp") %>">
<input type="text" name="who">
<input type="submit" value="shop"></form>
```

shop.jsp

```
<%
String who = request.getParameter("who");
session.setAttribute("who", who);
%>
<table border="1"><tr>
<th>Fruit</th><th>Image</th><th>Price</th></tr>
<%
ResultSet result = stm.executeQuery(
"select name, image, price from fruit");
while (result.next()) {
  String fruit = result.getString("name");
  String image = result.getString("image");
  int price = result.getInt("price");
%>
<tr><td><%= fruit %></td>
<td><img src="<%= image %>" width="50"></td>
<td><%= fmt.format(price) %></td>
<td><form action=
  "<%= response.encodeURL("check.jsp") %>">
<input type="hidden"
  name="fruit" value="<%= fruit %>">
<input type="submit" value="check">
</form></td></tr>
<%}%></table>
```

check.jsp

```
<%
String fruit = request.getParameter("fruit");
String who = (String) session.getAttribute("who");
%>
Thank you for shopping <%= fruit %>, <%= who %>.
```

図-6　オンラインショップの JSP ページ

している．

◘Web アプリケーション開発上の問題点

オンラインショップ Web アプリケーションの例は，小さなプログラムであるため，一見すると，簡潔に記述されているように見える．しかし，大規模な Web アプリケーションを開発する場合には，このプログラミング方法には，プログラムの保守を困難にする，次に示すような問題点がある．

- 1 つの JSP ページの中に，Web ページのデザイン

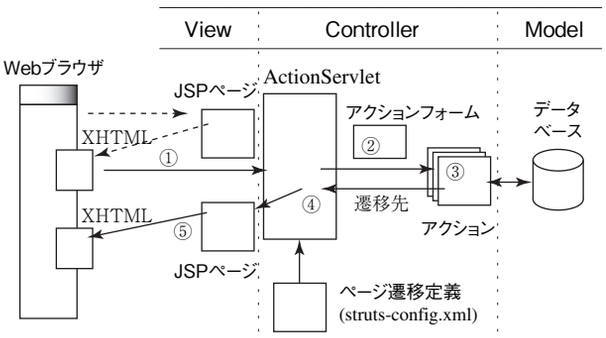


図-7　Strutsフレームワーク

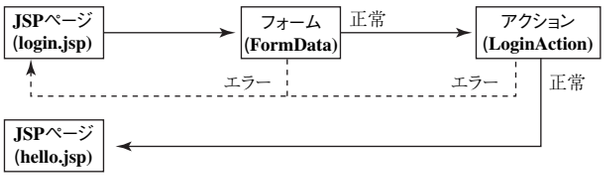


図-8　エラー時のページ遷移

（XHTML記述の部分）とビジネスロジック（Javaコード記述の部分）が混在しているため，読解性が悪い．

- 「はじめに」で述べた，Webアプリケーションに必要な各種の処理（フォームデータの受け取り，値の検証，エラーメッセージの表示，ページの遷移等）を記述するためのプログラミング方法が統一化されておらず，個々の処理を変更したり，拡張したりすることが容易ではない．

これらの問題を解決するために，いろいろなフレームワークが開発されており，本稿では，その中のStrutsとJWIGについて紹介する．

MVCフレームワーク

MVC（Model-View-Controller）アーキテクチャは，ソフトウェアをモデル（データの操作），ビュー（データの表現），コントローラ（ユーザ入力制御）に分離して開発するフレームワークである．3つのコンポーネントの独立性が高くなり，ソフトウェアの保守が容易になる．

Struts

Struts[2]は，リクエストを一元的に管理する独自のコントローラを提供し，MVCアーキテクチャに基づくWebアプリケーション開発を支援する．Strutsは次のような特徴を持つ．

(1) Webページのデザインとプログラムロジックを分離して記述できる．
(2) 入力データの検証が容易である．
(3) 遷移するページ名をプログラムのロジックから切り離して記述できる．
(4) セッション管理が自動化されており，セッションIDの付加を意識する必要がない．

Strutsでは，**図-7**に示すように，Webアプリケーションは，JSPページ，アクションフォーム，アクション，ページ遷移定義等から構成される．①Webブラウザがフォームデータを送信すると，②Strutsは，フォームデータを格納したアクションフォームを生成し，③アクションを起動する．アクションでは，アクションフォームを受け取り，データベースにアクセスし，ビジネスロジックを実行する．④そして，遷移先となるページにリクエストを転送すると，Strutsは，ページ遷移定義から実際に遷移するJSPページを探し，⑤そのJSPページの出力結果であるXHTML文書をWebブラウザに送り返す．

Strutsプログラミング

Strutsを用いて，図-2(a)のアプリケーションを記述してみる．Strutsでは，フォームデータの検証処理やリクエストの転送先をビジネスロジックから切り離して記述することができる．**図-8**に示すように，フォームデータの検証時，あるいは，アクションの実行時にエラーがあれば，最初のloginページを表示することとする．

このアプリケーションは，**図-9**に示すように，loginページを表示するJSPページ（login.jsp），フォームデータを格納するJavaBean（FormData.java），ビジネスロジックを記述するアクション（LoginAction.java），結果を表示するJSPページ（hello.jsp），および，ページ遷移を定義するXMLファイル（struts-config.xml）からなる．パラメータnameの値は，プロパティnameを定義したFormDataオブジェクトに自動的に格納される．validate()メソッドを記述することにより，このフォームデータを検証することができる．ここでは，名前の入力がなかったときにエラーを返している．アクションでは，パラメータnameの値をFormDataオブジェクトから取り出し，"anonymous"が入力された場合には"failure"ページへリクエストを転送し，それ以外の場合には"success"ページへ転送する．実際には，ページ遷移定義に記述されているように，各々，login.jsp，hello.jspへ遷移する．struts-config.xmlファイルの表す処理フローをグラフィカルに表示・編集し，Strutsアプリケーションの開発を支援するEclipseプラグインに，Easy Struts，Struts Studioがある．

JSPページ(login.jsp)

```
<html:errors/>
<html:form action="/Login">
Enter your name: <html:text property="name"/>
<html:submit value="Continue"/>
</html:form>
```

login.jspから生成されWebブラウザに送られるXHTML文書

```
<form name="formData" method="post"
  action="/myStruts/Login.do">
Enter your name:
<input type="text" name="name" value="">
<input type="submit" value="Continue">
</form>
```

アクションフォーム(FormData.java)

```
public class FormData extends ActionForm {
  private String name;
 public String getName() { return name; }
 public void setName(String name) {
    this.name = name; }
 public ActionErrors validate(...) {
    ActionErrors errors = new ActionErrors();
    if (name.length() == 0) {errors.add("name",
      new ActionError("name.noInput")); }
   return errors; }
}
```

アクション(LoginAction.java)

```
public class LoginAction extends Action {
  public ActionForward execute(...) ... {
   String name = ((FormData) form).getName();
    if (name.equals("anonymous")) {
       ActionErrors errors = new ActionErrors();
      errors.add("name",
         new ActionError("name.anonymous"));
      saveErrors(request, errors);
      return map.findForward("failure"); }
   request.setAttribute("name", name);
   return map.findForward("success"); }
}
```

JSPページ(hello.jsp)

```
<%
String name = (String) request.getAttribute("name");
%>
Hello <%= name %>
```

ページ遷移定義(struts-config.xml)

```
<action path="/Login" type="LoginAction"
  name="formData" input="/login.jsp">
  <forward name="success" path="/hello.jsp"/>
  <forward name="failure" path="/login.jsp"/>
</action>
```

図-9　StrutsによるWebプログラミング

シングルスレッドフレームワーク

ここでは，クライアントマシン上の1つのスタンドアロンプログラムのように，Webアプリケーションを開発することができるシングルスレッドフレームワークについて紹介する．

JWIG

Webアプリケーションでは，次々とWebページを遷移しながら，処理が進められる．サーバプログラムは，図-1に示したように，フォームから送信されたデータを受け取って処理を行い，その結果を表すXHTML文書を生成して，Webブラウザに送り返す．Webアプリケーションには複数のユーザが同時にアクセスするため，サーバプログラムは，複数のスレッドで動作し，個々のユーザを識別するためにセッションを管理している．しかし，Webブラウザを操作している1人のユーザから見れば，Webアプリケーションも，クライアントマシン上の1つのスタンドアロンのプログラムと見かけは変わらない．JWIG[3]は，Webアプリケーションを1つの連続したスレッドと見なしてプログラミングすることを可能にしたJavaの拡張言語である．

JWIGプログラミング

図-2(a)に示したWebアプリケーションを少し拡張し，JWIGを用いてプログラミングしてみる．このWebアプリケーションでは，①ユーザが名前を入力すると，②次のページに，入力した名前とユーザ番号を表示し，③最後のページに，"Goodbye"を表示する（**図-10**(a)）．ユーザ番号は，このWebアプリケーションにユーザがアクセスするごとに1ずつ増えていく．

プログラミングした例を**図-10**(b)に示す．main()メソッドからWebアプリケーションが開始される．①まず，wrapperにフォームを埋め込み，そのフォームに"Enter your name:"と，テキストフィールドnameを埋め込んだXHTML文書をWebブラウザに送り返す．②次に，テキストフィールドnameから名前を受け取り，名前とユーザ番号を埋め込んだフォームを送り返す．③最後に，"Goodbye"と名前を送り返す．

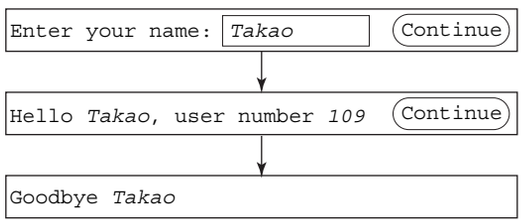
セッションを意識しないで済むため，サーブレットやJSPページを用いるよりも，プログラムの構造が簡単になる．また，JWIGでは，XHTML文書の各部分を，階層構造を持つテンプレートとして定義し，XHTML文書の生成時に具体的な内容を埋め込む方式をとっている．このため，

(1) フォームデータの受信時に，実行時エラーが起こらない．
(2) XHTML文書を動的に生成するときに，実行時エラーが起こらない．
(3) 生成されるXHTML文書の構造が正しいことを，JWIGプログラムのコンパイル時に検証できる．

という特徴がある．

その他のフレームーワーク

サーブレット，JSPページ等を用いたWebアプリケーションの統合開発環境として，Sun Java Studio，Eclipse，Spring，Torque等がある．GUIコンポーネントやウィザードを利用したビジュアルな統合開発環境としては，Visual Studio .NET，Web Matrix，Macromedia



(a) Webアプリケーション例

```
int counter = 0;
synchronized int next() {
  return ++counter;
}
XML wrapper = [[<html><head>...</head>
  <body><[body]></body></html>]];
XML form = [[<form><[contents]>
  <input type="submit" value="Continue"/>
  </form>]];
XML login =
  [[Enter your name:
  <input type="text" name="name"/>]];
XML hello =
  [[Hello  <[who]>, user number <[count]>]];
XML goodbye = [[Goodbye <[who]>]];
public void main() throws IOException {
  XML x = wrapper<[body=form];
  show x<[contents=login];
  String name = receive name;
  show x<[contents=hello
    <[who=name,count=next()]];
  exit wapper<[body=goodbye<[who=name]];
}
```

(b) JWIG プログラム

図-10　JWIGによるWebプログラミング

	テンプレート メソッド方式	イベント ドリブン方式	その他
テキスト指向	**Struts**	JSF Struts-Faces	Servlet **JSP** Torque **JWIG** Tiles Tapestry Spring
↕	Easy Struts Struts Studio	JSF Studio	Spindle
ビジュアル指向	Iron Speed	Web Matrix Visual Studio .NET WebSphere Studio Sun Java Studio Creator	BioPro Macromedia Dreamweaver Flex

図-11　フレームワークの分類

Dreamweaver MX，Iron Speed Designer，Flex，BioPro等がある．これらの中には，データベーステーブルをナビゲーションするための多彩なコントロールや，テーブル一覧表示から詳細表示に遷移する1つのアプリケーションに近いコンポーネントが提供されているものもある．BioPro[4]では，ページの遷移や，テーブル，フィールド，フレーム等の参照関係が矢印で表示されるため，アプリケーションを構成する各コンポーネントの間の関連や，処理の流れが分かりやすい．

JavaServer Faces[5]は，レンダリング機能とイベント処理機能を持つコンポーネントモデルを提供するWebアプリケーションフレームワークである．コンポーネントを用いてWebページを設計し，送信ボタン等のクリック時の処理はイベントハンドラとして記述する．StrutsとJavaServer Facesを連携したStrutsFacesや，JavaServer Facesをサポートしたビジュアルな統合開発環境としてSun Java Studio Creator，WebSphere Studio，EclipseプラグインJSF Studio等がある．

このほかに，Tilesは，Webページのレイアウト定義とコンテンツを分離したWebページ生成フレームワークであり，Strutsの中でリクエストの転送先となるJSPページの作成に用いる．Tapestryは，Webページのデザインとロジックを分離するWebアプリケーションフレームワークである．ページテンプレートで，動的表示部分を含むXHTML文書を定義し，ページコンポーネントで，動的表示内容を定義するJavaクラスを記述する．

おわりに

各種のフレームワーク，開発環境を，イベントハンドラによりアクションを記述するイベントドリブン方式，および，処理の流れの各部分をカスタマイズするテンプレートメソッド方式に分類すると，**図-11**のようになる．今後は，柔軟性・保守性の高さと，ビジュアルプログラミング技術が焦点になっていくと考えられる．また，Webページに表示されるコンテンツとしては，XHTMLだけではなく，アプレットやFlash等の動画像の導入もますます広がっていくと思われる．

参考文献
1）The Apache Software Foundation, http://www.apache.org/ (2004).
2）The Apache Software Foundation: Struts, http://jakarta.apache.org/struts/ (2004).
3）Christensen, A.S., Moller, A. and Schwartzbach, M.I.: Extending Java for High-Level Web Service Construction, ACM TOPLAS, Vol.25, No.6, pp.814-875 (2003).
4）Shimomura, T., Takahashi, M., Ikeda, K. and Mogami, Y.: Visual Design for Server-Side Programs and Program Generation, IPSJ Journal, Vol.45, No.7, pp.1737-1744 (July 2004).
5）Sun Microsystems, Inc.: JavaServer Faces, http://java.sun.com/j2ee/javaserverfaces/ (2004).
6）下村隆夫: Javaによるインターネットプログラミング，近代科学社，pp.165-170 (2002).

（平成16年7月16日受付）