システム開発最前線
～ サーバーサイド技術におけるフレームワーク比較 ～

フレームワークの概要、導入による効果から今後の課題・問題点まで解説し、ソフトウエア／システム開発におけるフレームワーク活用の必要性を徹底検証します。J2EEにおける定番フレームワークStrutsなどのオープンソースからベンダー製品までその特徴と違いを紹介すると同時に他技術（ASP.NET、PHP等）における.NET FrameworkやPhrameなどとの比較も行います。

山田祥寛 (YAMADA, Yoshihiro)
CQW15204@nifty.com
http://www.wings.msn.to/

フレームワークの定義

- フレームワーク ＝ 枠組。骨組。体制。（広辞苑 第4版）
  - 戦略分析のためのフレームワーク
  - マーケティング立案のためのフレームワーク
  - ソフトウェア開発のためのフレームワーク

    → なにかしら物事を行うときにもととなる基本的な考え方・ルール

    → 問題解決のために、状況・対象（問題領域）に応じて、ある規則・概念に則って考案された枠組み
- アプリケーション・フレームワーク
  - アプリケーション設計のために考案された枠組み・ルール・思想
  - 設計を実装するために利用可能なテンプレート・ライブラリ・ツール

アプリケーション・フレームワークとは？

- アプリケーション・フレームワーク
  - 開発プロセス（開発ポリシー・概念）
  - 開発ドキュメント（標準化ドキュメント）
  - 開発ツール
  - コンポーネント
  - クラスライブラリ　etc,etc...

    → 形態・適用分野・利用目的も異なり、その意味は「曖昧」

アプリケーション・フレームワークの段階的分類

- **第1段階：標準化ルールを散文的にドキュメント化**
  - ○ コーディングが楽になる「かもしれない」
  - × あくまでコーディングは1から
  - × 人によって解釈が異なる可能性がある
  - × 標準ルールを強制しない（できない）
- **第2段階：コード・スニペット**
  - ○ コードを（部分的に）再利用可能
  - × 同じようなコードが複数作成される
  - × 修正時に影響は大
  - ×標準ルールを強制しない（できない）
- **第3段階：クラス・ライブラリ化**
  - ○ コードを再利用可能
  - ○ 標準ルールを強制する　→ 狭義の「フレームワーク」

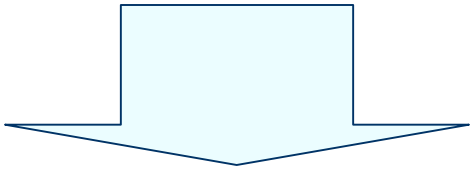
アプリケーションフレームワークの必要性

- ユーザー要件の高度化
  - 専門家に委ねる必要性
  - ミッションクリティカルなシステムへの適用
  - 少ない開発コスト・短い納期
- ビジネスニーズの流動化・多様化
  - 日常的な改定要求に対する仕組みづくりへの要請
  - 統合を前提としたシステムへ
- テクノロジーの多様化
  - 多様なフロントエンド（HTML、リッチクライアント、携帯端末 etc）

アプリケーションフレームワークの必要性 ～他競合技術と比較した場合～

- Javaと競合技術との比較

項目	Java	.NET	PHP
ターゲットとするシステム規模	大	大	小
標準APIのレベル	低水準	高水準	高水準
市場への普及度	◎	△	○



- Javaの課題

  市場ニーズと標準APIが提供する「格差」をどのように埋めるか？

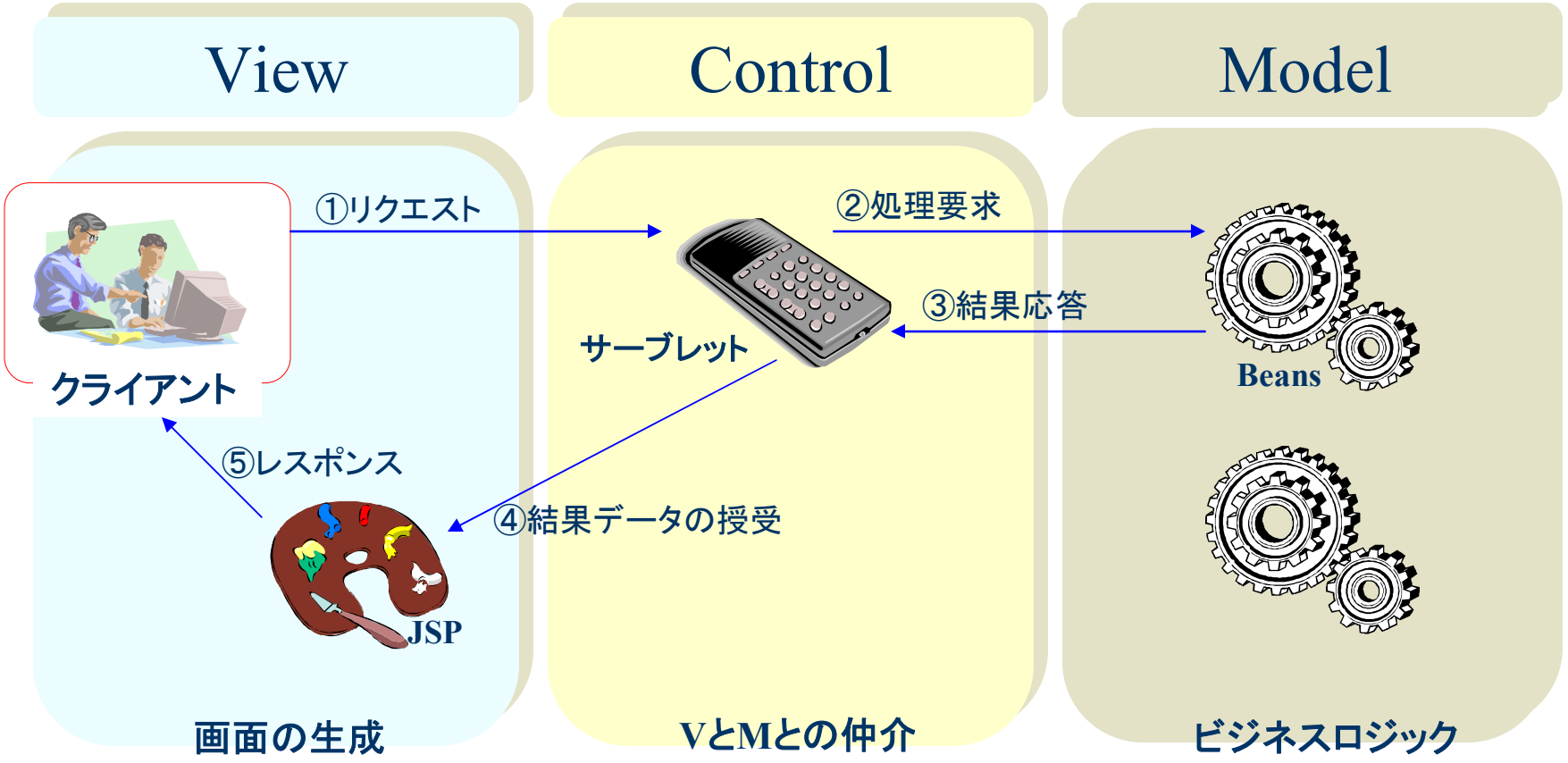
フレームワークの導入効果（1）

- 定型的な記述をフレームワークに委ねられる
  - 問題領域に対する「設計分析」のノウハウを再利用できる
  - ゼロベース開発と比較した場合にリスクが低い
  - コンカレントな開発でモジュールの品質を均質化できる
    （標準の強制。 “Don’t call us, we’ll call you.”）
  - サブシステム同士が構造的に均質であることが保証される
    （相互接続性）
  - 共通ルールに基づくので、IDEなどの導入も容易
  - 画面のメンテナンス性に優れる

フレームワークの導入効果（2）

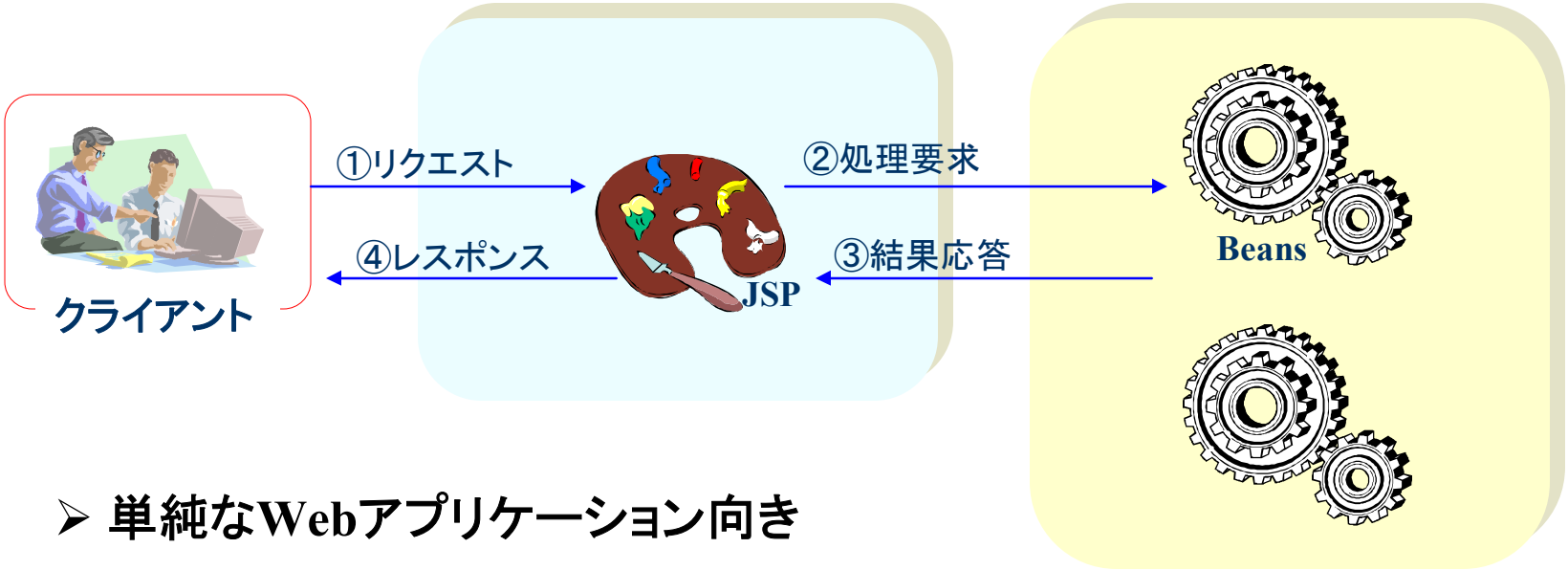
- アプリケーション固有のビジネスロジックに専念できる
  - 開発コード・工数の削減
    ※ある事例では、手書きのコード量を6割削減
  - J2EEに精通していなくても開発が可能
  - バックエンドシステムとの相互接続性を確保
  - 個々のモジュールが独立するため、協業・役割分担が可能
  - 追跡可能性、コードの可視性が向上
  - 問題特定の迅速化、修正時の影響箇所の局所化
- ビジネスロジックの再利用が可能となる
  - フロントエンドの拡張にも容易に対応可能
    ※ ただし、特定インフラ依存のコードを排除する必要

標準的なMVC（Model-View-Control）モデル2
～JSPモデル2～



（参考）JSPモデル1
～より単純なWebアプリケーションのために～

- JSPモデル1

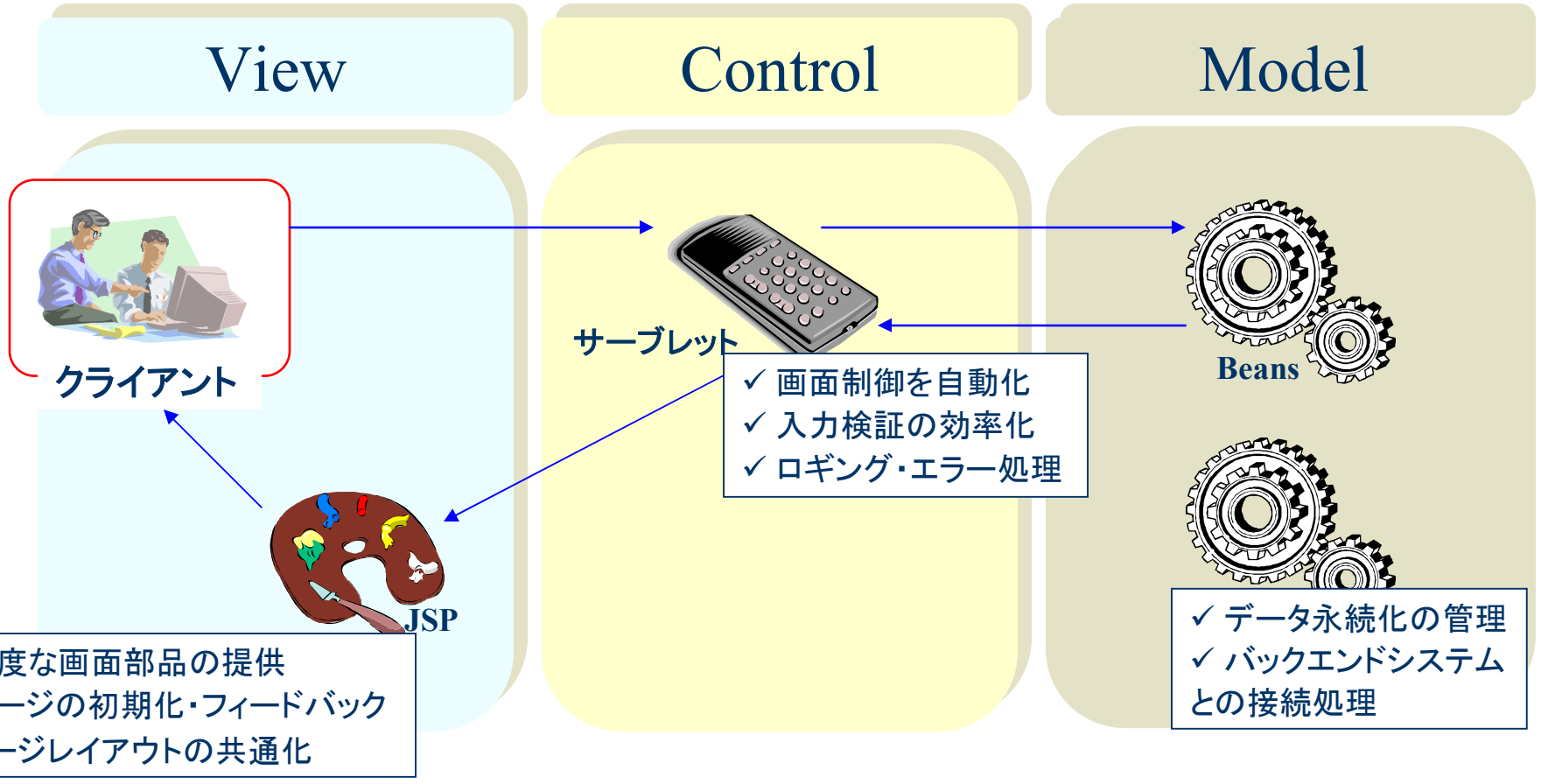


- 単純なWebアプリケーション向き
- サーブレットを介することで却って複雑になる場合に使用
- （当然）ひとつのアプリケーション内にモデル1と2とが混在することは好ましくない

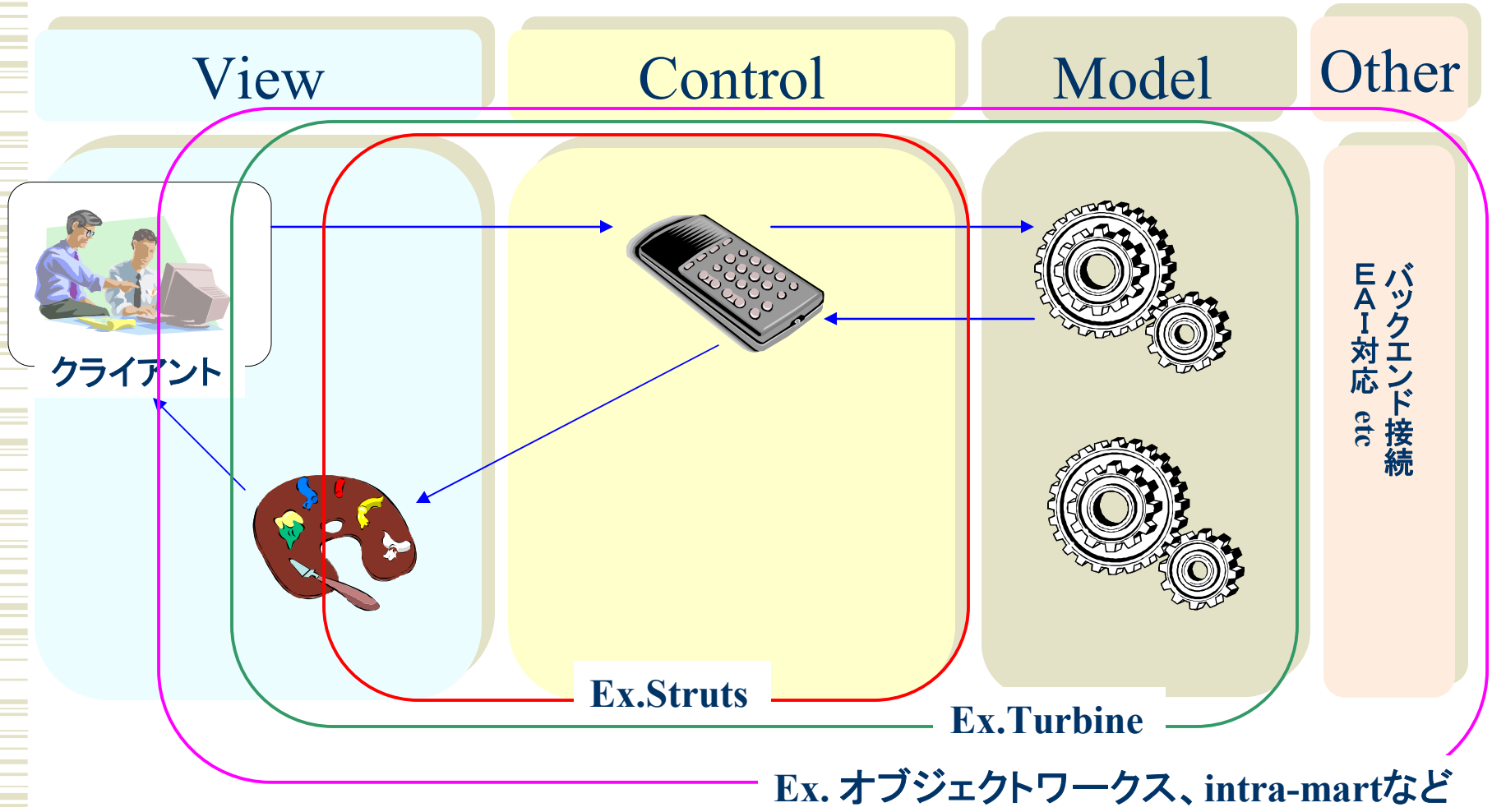
フレームワークを利用しない場合の問題点

- Control（サーブレット）はごく定型的な記述
  - 入力パラメータの解析・検証
    ※エラーのフィードバックだけでも困難
  - ビジネスロジックの呼び出し
  - 画面遷移の制御
  - 出力パラメータの生成
- View（JSP）の表現力は決して高くない
  - 共通的なテンプレートを表現する手段を持たない
    （JSP2.0からはJSPコンフィギュレーションの利用も可能だが...）
  - 貧弱なタグライブラリ
    （昨今ではJSTLやELも提供されているが...）

フレームワーク利用による問題の解決



アプリケーションフレームワークの守備範囲(1)



アプリケーションフレームワークの守備範囲（2）

- 汎用型フレームワーク（Ex. Struts）
  - ControlとViewの一部分のみを担当
- MVC対応フレームワーク（Ex. Turbine）
  - Model – View – Control を広範にサポート
    ※テンプレートエンジンVelocity、O/Rマッピングツール Torque、
    プロジェクト管理ツールMaven
- エンタープライズ フレームワーク（Ex. オブジェクトワークス、intra-mart）
  - フロントエンドからバックエンドまで全方位的なソリューションの提供
  - 多くの場合は商用

目的に応じて、フレームワークは使い分ける必要がある

商用製品とオープンソース製品

- 商用製品とオープンソース製品とのちがい
  - 必ずしも一概には言えないが...

項目	商用	オープンソース
製品価格	有償。多くのものは高額	基本的に無償
ターゲット	特定分野	汎用
プラットフォーム	クロスプラットフォーム	プラットフォームを限定
機能	エンタープライズ系の機能まで包括	非機能的な実装に限定
IDEとの連携	独自IDEが提供される ※EclipseなどとのPluginも	基本的にはなし（種類によっては、Eclipse対応Plugin有）
サポート	あり	原則、自己責任

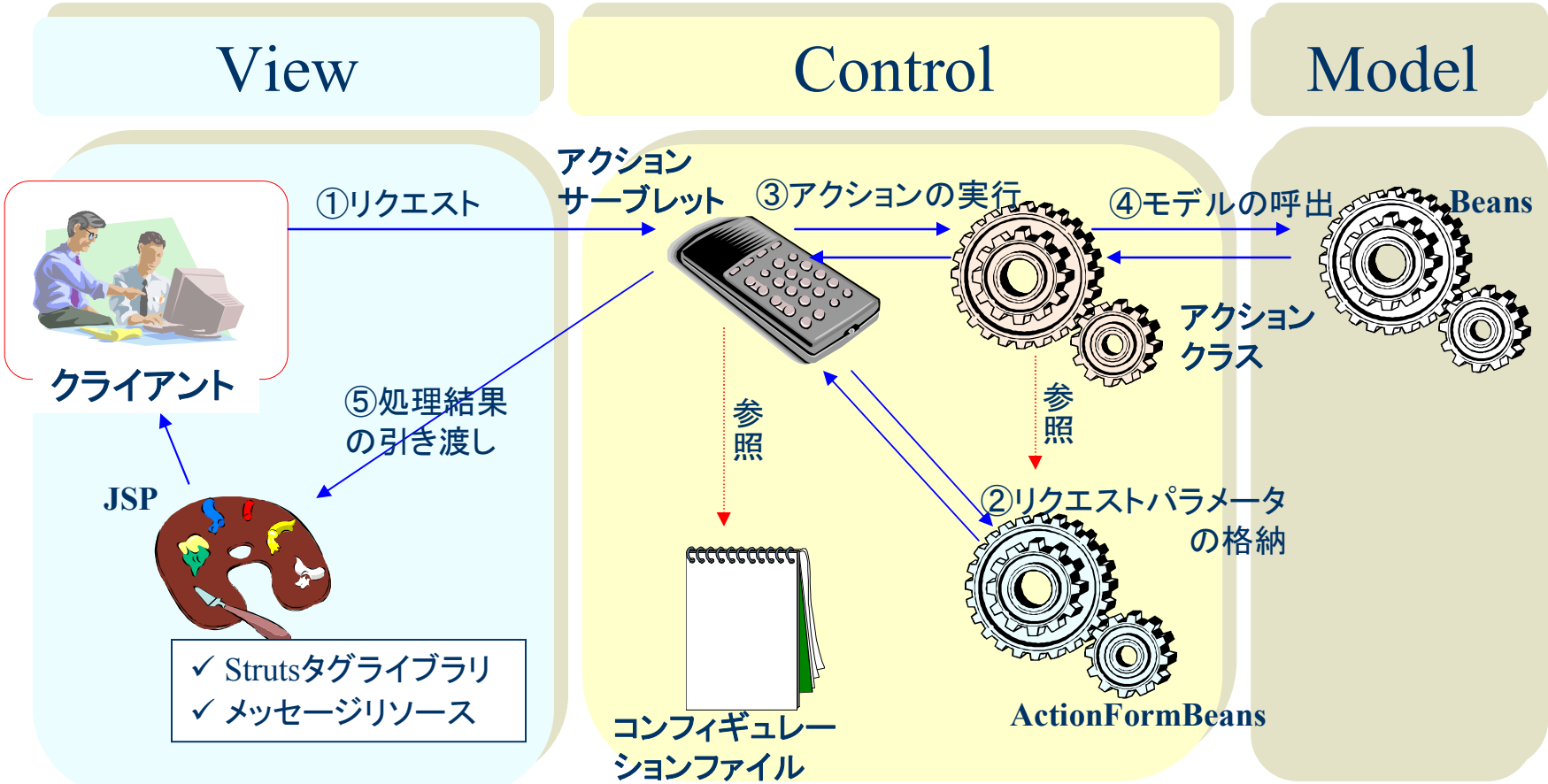
代表的なフレームワーク～Struts～

- Apache Software Foundationから提供されているフレームワーク
  - Jakartaプロジェクトから移動（2004/3/18）
  - Apacheソフトウェアライセンスのもとで提供
    ※再配布の制約が緩い
- 軽量でシンプルなフレームワーク
  - MVCモデル2のControl機能をサポート（一部、View機能を含む）
  - ビジネスオブジェクト（Model）には非対応
    ※EJBやTorqueなどの連携も自由（システム規模に関わらない適用が可）
  - Struts Application Project（http://struts.sourceforge.net/）などで拡張ライブラリやツールを提供
- 可搬性の保証
  - JSP1.1＆サーブレット2.2仕様に準拠

Strutsの機能構成

- アプリケーション制御部（Control）
  - アクションサーブレット（データ処理と画面遷移を制御）
    ※必要な情報はコンフィギュレーションファイルに集約
  - ActionFormBeans（リクエストパラメータの維持、検証）
  - Validatorなどの拡張プラグインに対応
  - ロギング、例外処理（システム例外・入力例外）の提供
- ユーザインターフェイス部（View）
  - Struts独自JSPタグライブラリの提供（但し、VelocityやXSLTも利用可能）
  - Tilesテンプレートエンジンに対応
- ビジネスロジック部（Model）
  - EJBやJavaBeansで記述（Torqueはじめ各種ライブラリとの連携も可）

Strutsにおけるアプリケーションの動作



（参考）コンフィギュレーションファイル

```
<?xml version="1.0" encoding="Shift_JIS" ?>
<!DOCTYPE struts-config PUBLIC
        "-//Apache Software Foundation//DTD Struts Configuration 1.1//EN"
        "http://jakarta.apache.org/struts/dtds/struts-config_1_1.dtd">
<struts-config>
  <form-beans>
    <form-bean name="BeginForm" type="struts.BeginForm" />
  </form-beans>
  <action-mappings>
    <action path="/BeginAction" type="struts.BeginProcess"
      name="BeginForm" scope="request">
    <forward name="success" path="/output.jsp" /></action>
  </action-mappings>
  <message-resources parameter="ApplicationResources" />
  <plug-in className="org.apache.struts.validator.ValidatorPlugIn">...</plug-in>
</struts-config>
```

ActionFormBeansの定義

URLとアクションとのマッピング

フレームワーク利用に伴う問題点

- 一般的な問題点
  - フレームワークを習得するための学習コスト
  - 問題（対象領域）の選定が不可欠
    ※いったん導入したフレームワークを移行するのは難しい
  - フレームワーク固有の制約に依存する
- Struts固有の問題点
  - View、Modelに採用すべきアーキテクチャの検討が必要
    → 業務上システムには付加機能の追加は必須
  - バージョンアップによる変動が激しい
  - 設定ファイルのメンテナンスが困難
    → アプリケーションの情報すべてを集約しているため、肥大化しやすい
    → Struts 1.1からはモジュール単位の分割を可能にしているが...

Strutsのロードマップ（Struts1.1）

- Struts 1.1（2004年5月時点の最新バージョン）
  - DynaActionForm → アクセサメソッドの記述省略が可能
  - PlugIn → Strutsの追加機能を実装（Validator、Tilesなども）
  - RequestProcessor → リクエストの前後処理など付随機能を追加
  - アクション単位のユーザロール設定
  - Struts-ELタグライブラリ（StrutsタグでELを利用するしくみ）

※ JSP2.0の登場によって、JSTLやELへの置換の必要性
（Logicタグライブラリのほぼすべて、Beanタグライブラリの一部はJSTLでの置換が推奨される）

✓ プロジェクト開発をより意識したつくりへ
✓ フレームワークとして、個別のモジュールの独立性を向上

（参考）DynaActionForm

```
public class BookUpdateForm extends ActionForm {
  private String isbn;
  private String title;
  private String author;
  public String getIsbn() {return isbn;}
  public String getTitle() {return title;}
  public String getAuthor() {return author;}
  public void setIsbn(String isbn) {this.isbn=isbn;}
  public void setTitle(String title) {this.title=title;}
  public void setAuthor(String author) {this.author=author;}
}
```

アクセサメソッドの羅列

```
<form-bean name="BookUpdateForm"
    type="org.apache.struts.action.DynaActionForm">
  <form-property name="isbn" type="java.lang.String" />
  <form-property name="title" type="java.lang.String" />
  <form-property name="author" type="java.lang.String" />
</form-bean>
```

（参考）Validation Plugin

```
<form name="BookUpdateForm">
  <field property="isbn" depends="required,mask">
    <arg0 key="ISBN番号" resource="false" />
    <var>
      <var-name>mask</var-name>
      <var-value>[0-9]{1}-[0-9]{3,5}-[0-9]{3,5}-[0-9A-Z]{1}</var-value>
    </var>
  </field>
  <field property="title" depends="required,maxlength">
    <arg0 key="タイトル" resource="false" />
    <arg1 name="maxlength" key="${var:maxlength}" resource="false" />
    <var>
      <var-name>maxlength</var-name>
      <var-value>100</var-value>
    </var>
  </field>
</form>
```

正規表現検証

最大長検証

（参考）Tiles Plugin（1）

```
<tiles-definitions>
  <definition name="site.base" path="/index.jsp">
    <put name="header" value="/HeaderDisp.do" />
    <put name="menu" value="/MenuDisp.do" />
    <put name="body"   value="/TopDisp.do" />
  </definition>
  <definition name="site.descript"
      extends="site.base">
    <put name="body"
      value="/SiteDiscription.do" />
  </definition>
</tiles-definitions>
```

ベーステンプレート

個別テンプレート

（参考）Tiles Plugin（2）

ベーステンプレート

```
<tiles:insert attribute="header" />
<table cellspacing="2" cellpadding="2" border="0">
  <tr>
  <td valign="top" bgcolor="#e3ebe2" colspan="1">
    <tiles:insert attribute="menu" />
  </td>
  <td valign="top" bgcolor="#ffffff">
    <tiles:insert attribute="body" />
  </td>
  </tr>
</table>
```

```
<action-mappings>
  <action path="/SiteIndex"
    type="org.apache.struts.tiles.actions.NoOpAction">
    <forward name="success" path="site.base" />
  </action>
  <action path="/SiteDescription"
    type="org.apache.struts.tiles.actions.NoOpAction">
    <forward name="success" path="site.descript" />
  </action>
</action-mappings>
```

コンフィギュレーションファイル

（参考）設定ファイルの分割

デプロイメント・ディスクリプタ

```
<servlet>
  <servlet-name>action</servlet-name>
  <servlet-class>～ActionServlet</servlet-class>
  <init-param>
    <param-name>config</param-name>
    <param-value>
      /WEB-INF/struts-config.xml </param-value>
  </init-param>
  <init-param>
    <param-name>config/sub</param-name>
    <param-value>
      /WEB-INF/struts-config-sub.xml </param-value>
  </init-param>
</servlet>
```

～/sub/配下のアプリケーション

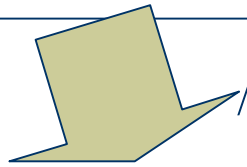
```
<init-param>
   <param-name>config</param-name>
   <param-value>
      /WEB-INF/struts-config.xml,
      /WEB-INF/struts-config-sub.xml </param-value>
</init-param>
```

Strutsのロードマップ（Struts1.2～2.0）

- Struts 1.2（次期バージョン）
  - Validator機能の強化（ValidWhen、ValidateUrl）
  - MappingDispatchActionの追加
  - ワイルドカードによるマッピングが可能に
  - DigestingPlugIn → 独自のXMLファイルをオブジェクト展開
  - Struts-Chain（実験段階）
  - GenericDataSourceの削除
  - Action#performメソッドの削除
- Struts 2.0
  - JSP2.0 & サーブレット 2.4をベース（JSTLへの対応）
  - JSF（JavaServer Faces）への対応

（参考）MappingDispatchAction（1）

```
<action path="/SelectAction" type="struts.SelectProcess"
    name="SelectForm" scope="request" parameter="key">
   <forward name="success1" path="/output1.jsp" />
   <forward name="success2" path="/output2.jsp" />
   <forward name="success3" path="/output3.jsp" />
</action>
```

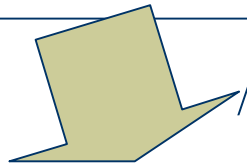


/SelectAction.do?**key=insert**

```
public class SelectProcess extends DispatchAction {
 public ActionForward insert(....) throws Exception(){...}
  public ActionForward update (....) throws Exception(){...}
  public ActionForward delete(....) throws Exception(){...}
}
```

（参考）MappingDispatchAction（2）

```
<action path="/InsertAction" type="struts.MappingProcess"
   parameter="insert">～</action>
<action path="/UpdateAction" type="struts.MappingProcess"
   parameter="update">～</action>
<action path="/DeleteAction" type="struts.MappingProcess"
   parameter="delete">～</action>
```

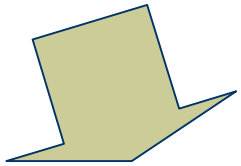


/InsertAction.do

```
public class SelectProcess extends MappingDispatchAction {
 public ActionForward insert(....) throws Exception(){...}
  public ActionForward update (....) throws Exception(){...}
  public ActionForward delete(....) throws Exception(){...}
}
```

（参考）ワイルドカードマッピング

```
<action path="/*Action" type="struts.{1}Process"
    name="{1}Form" scope="request">
    ～</action>
```



リクエストURI	アクションクラス	ActionFormBeans
/InsertAction.do	struts.InsertProcess	InsertForm
/UpdateAction.do	struts.UpdateProcess	UpdateForm
/DeleteAction.do	struts.DeleteProcess	DeleteForm

Strutsベースの独自フレームワーク（1）

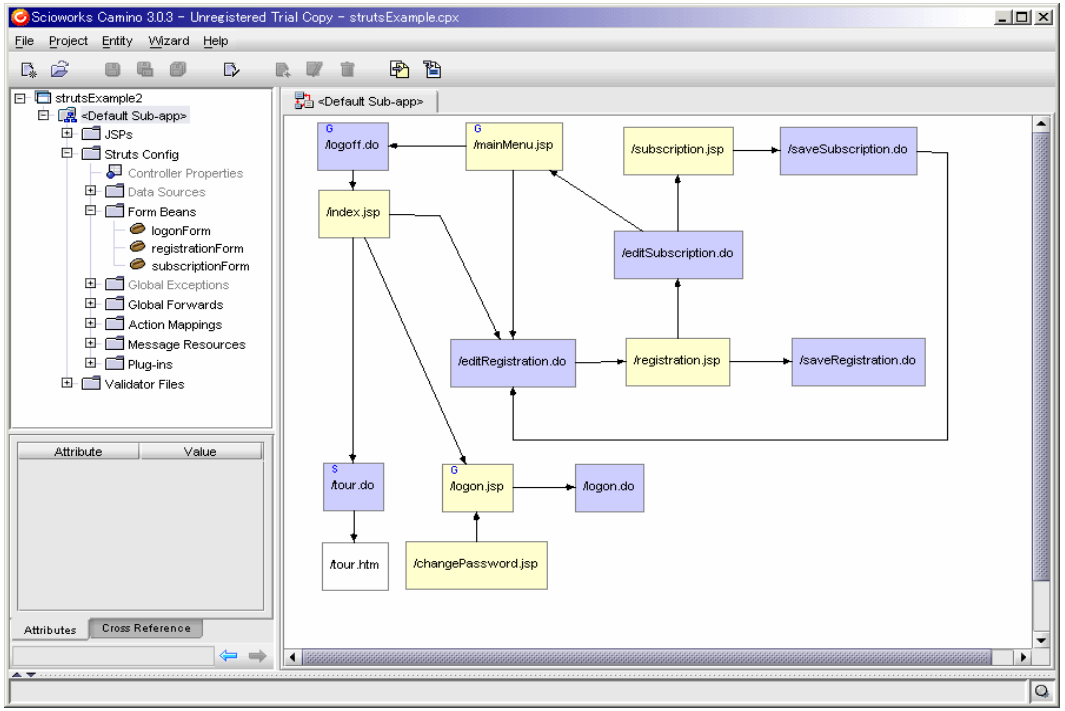
- T-Struts（http://t-struts.sourceforge.jp/）
  - データベースと連携した認証機能
  - 二重送信機能（セッション内で複数のトークンを保持可能）
  - Apache Mavenとの連携（開発環境構築・ビルドレポート）
  - キャッシュ、ロギング、キューなどをサービス化
  - ジャーナル、ロギング機能の強化
- Application Framework Suite（http://interstage.fujitsu.com/jp/v6/afs/）
  - フォーマット編集、スタイル装飾など追加タグライブラリを提供
  - 項目チェック仕様をExcelで定義（自動生成）
  - 電子フォーム対応

Strutsベースの独自フレームワーク（2）

- Extension for Struts
  （http://www-6.ibm.com/jp/software/websphere/developer/download/wasstx.html）
  - Pre/Post Invocation（事前事後処理）
  - Screen Order Control（画面遷移の強制）
  - ロギング、トレース機能の強化
- OpenMeisterEnterprise（http://www.openmeister.com/）
  - 業務構築のテンプレート・手順書を提供
  - コードの自動生成機能を対応
- その他
  - TERASOLUNA（http://www.nttdata.co.jp/services/s090159.html）
  - Arvicio（http://www.csk.co.jp/event/pdfs/2003ow_presen_07.pdf）

Struts開発をより効率化するツール（1）

- Scioworks Camino（http://www.10art-ni.co.jp/product/CoolVista/camino.html）
  - JSPコンバータ、コードジェネレータ（HTML→JSP、スケルトン生成）
  - ストーリーボード（コンフィギュレーションファイルのGUI編集）
  - コードへのクリックバック
  - バリデータファイルの GUI編集

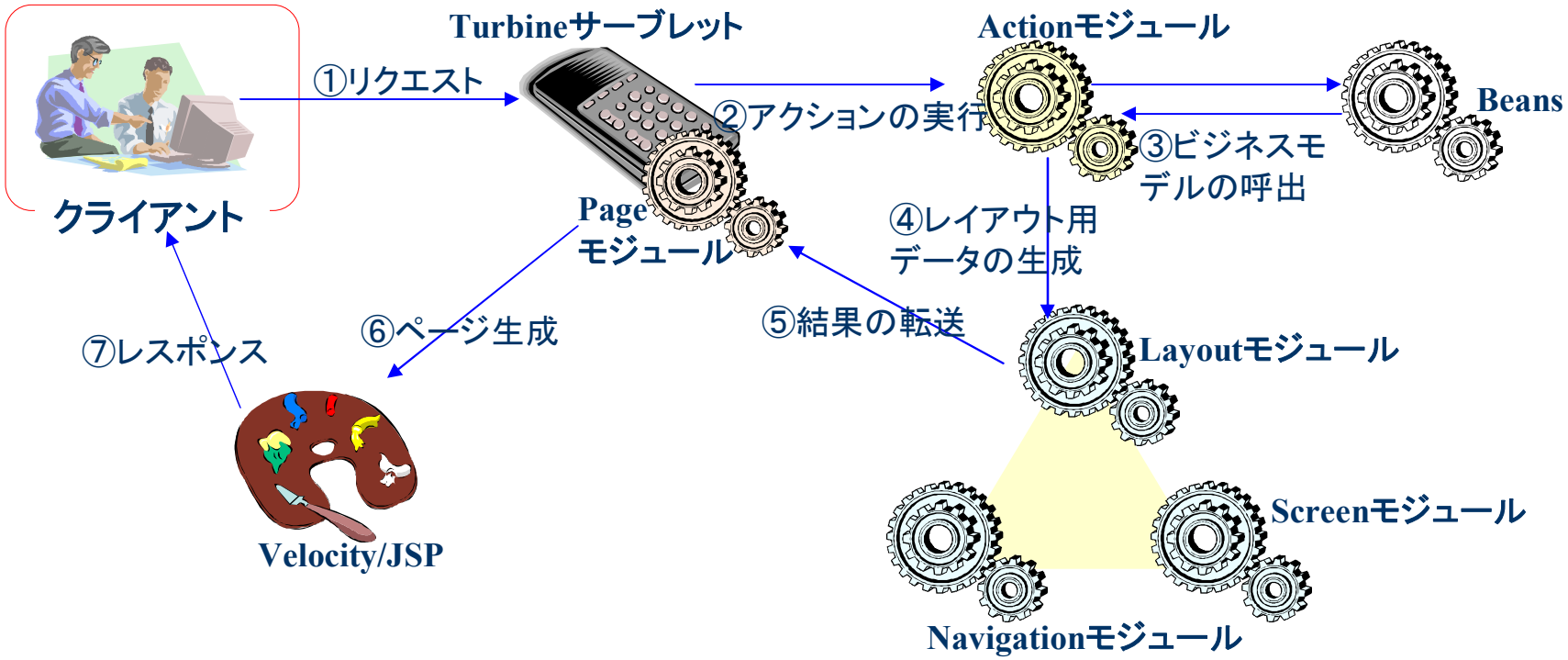


Struts開発をより効率化するツール（2）

- Easy Struts（http://easystruts.sourceforge.net/ ）
  - Eclipse対応プラグイン
  - コンフィギュレーションファイルのGUI編集
  - アクションクラス、ActionFormBeanクラスのスケルトン生成
- Improve Struts Configuration File Editor for Eclipse
  （http://www.improve-technologies.com/alpha/struts-config-editor/）
  - コンフィギュレーションファイルをグラフィカルに表示
  - 関係するクラス、JSPファイルを直接にロードするためのランチャ
  - Eclipse対応プラグイン

その他のアプリケーションフレームワーク(1)

- Turbine（http://jakarta.apache.org/turbine/）
  - Torque、Velocity、Mavenなどを輩出した高機能なフレームワーク



（補足）テンプレートエンジンVelocity

テンプレートファイル（VTL）

```
$title
<ul>
#foreach($i in $arch)
   <li> $i </li>
#end
</ul>
```

mergeTemplate

コントロール部

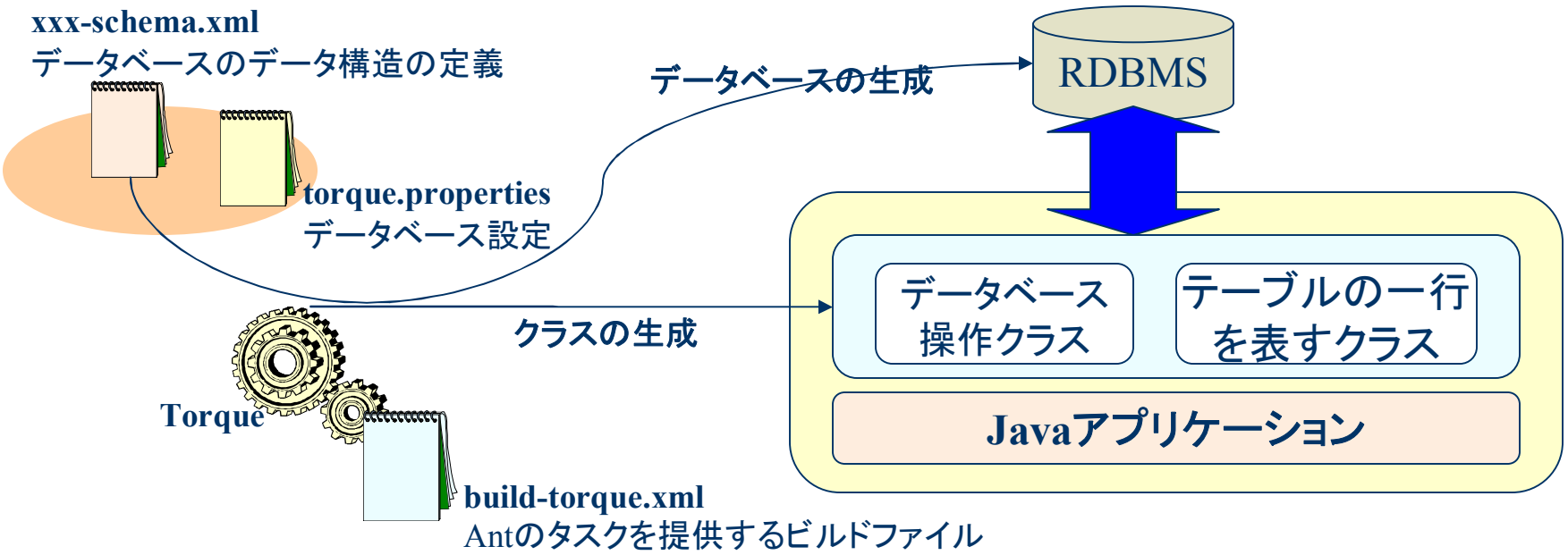
```
objAry=new ArrayList();
objAry.add(“JSP”);
objAry.add(“Servlet”);
objAry.add(“ASP.NET”);
context.put(“arch”,objAry);
context.put(“title”,”技術一覧”)
```

```
技術一覧
<ul>
  <li> JSP </li>
  <li> Servlet </li>
  <li> ASP.NET </li>
</ul>
```

変換結果

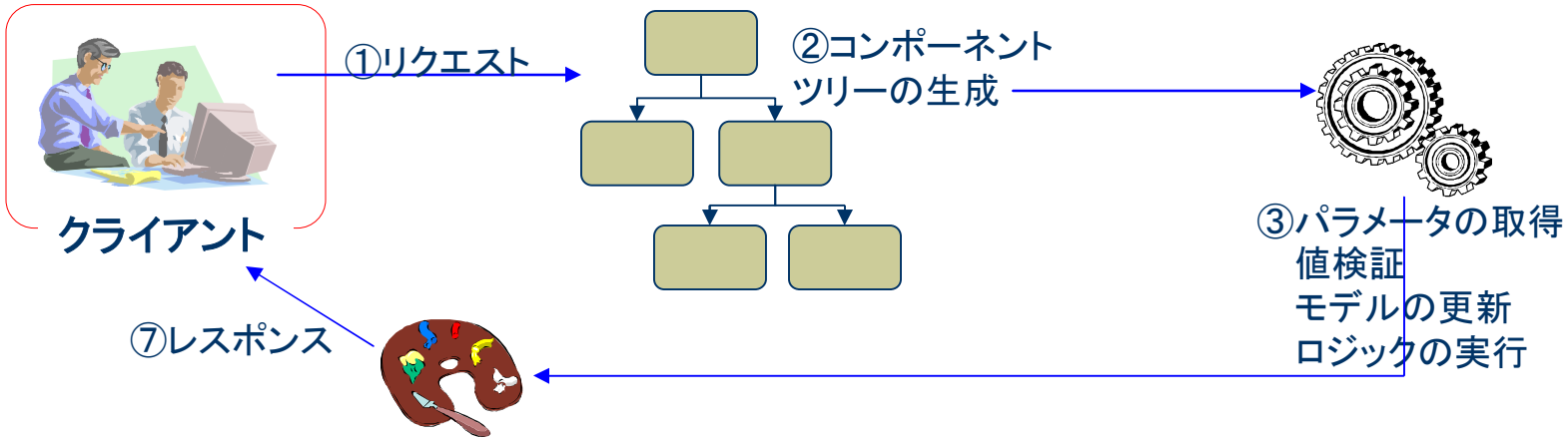
（補足）O/Rマッピングツール Torque

- Torque（http://db.apache.org/torque/）
  - Apache DB Project配下のO/Rマッピングツール
  - O/R（Object/Relational）マッピング
    ＝RDBMSのテーブルとJavaクラスとをマッピング



その他のアプリケーションフレームワーク(2)

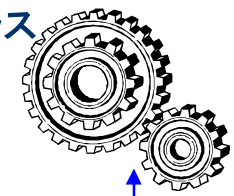
- Tapestory（http://jakarta.apache.org/tapestry/ ）
  - JWC（Java Web Component）がレンダリング、ロジック、表出力の制御
  - JWCとリスナクラスとを関連づけるイベントドリブン方式
- JSF（http://www.jcp.org/en/jsr/detail?id=127）
  - イベントドリブン型のアプリケーション開発
  - 抽象化されたウィンドウモデルを採用（広範な適用を想定）



（参考）Tapestoryのイベントドリブンモデル

アプリケーション定義ファイル（「.application」）

```
<application name="Sample"
    engineClass="～.BaseEngine">
    <property name="～"> ...</property>
</application>
```



アクションを実行

ページ情報ファイル（「.page」）

```
<page-specification class="~.SamplePage">
  <property-specification name="val"
      type="java.lang.String" ... />
</page-specification>
```

レイアウト情報（「.html」）

```
<form jwcid="@Form"
  listener="ognl:listners.sampleAction">
  名前 : <input jwcid="@TextField" value="ognl:val">
  <input type="submit" /></form>
<div jwcid="@Insert" value="ognl:val"></div>
```

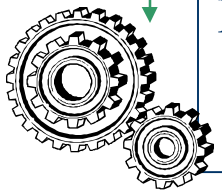
（参考）JSFのイベントドリブンモデル

レイアウト部分（「.jsp」ページ）

JavaBeansの定義

```
<jsp:useBean id="NumBean" class="jsf.NumBean" scope="session" />
<f:use_faces>
 <h:form id="fm" formName="calcuForm" >
  <h:input_number id="Num" valueRef="NumBean.num" /><br />
  <h:command_button id="sub" label="calcu" commandName="submit">
   <f:action_listener type="jsf.MyAction" /></h:command_button>
  <h:output_number id="out" valueRef="NumBean.result" /><br />
 </h:form>
</f:use_faces>
```

入出力フォームの定義



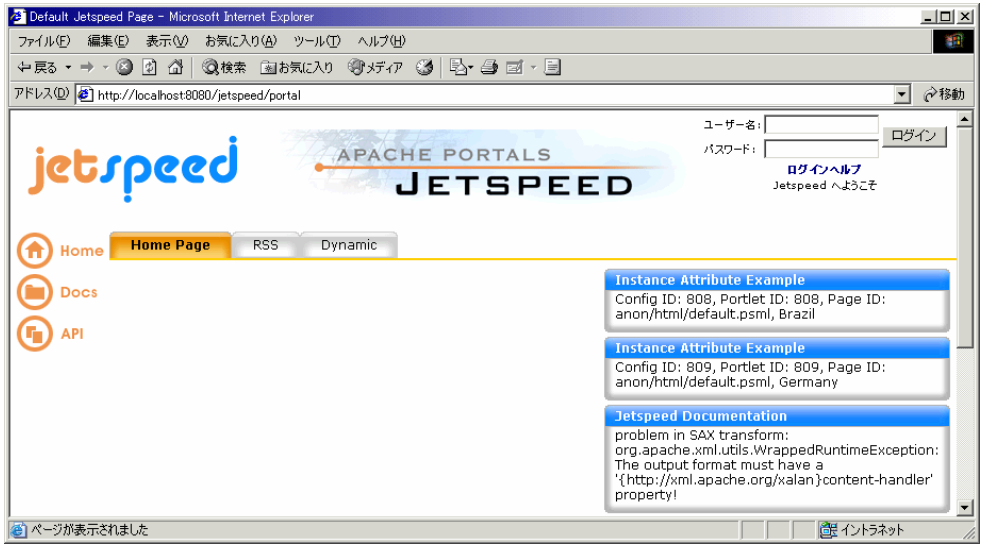
ActionListenerクラス

```
public class MyAction implements ActionListener {
  public void processAction(ActionEvent event) {...}
}
```

アクションの呼び出し

その他のアプリケーションフレームワーク(3)

- **Jetspeed（http://jakarta.apache.org/jetspeed/ ）**
  - **Portlet API（JSR168）をベースとしたポータル基盤（Turbineを利用）**
  - **ユーザ認証やパーソナライゼーション機能を標準で備える**
  - **HTML、JSP、FileServer、RSS、IFrame、Linkなどのポートレット**
    ※ PSML（Portal Structure Markup Language）によるカスタマイズ
    ※ Webサービスなどとの連携が今後期待される



（参考）JetspeedにおけるPSMLファイル

```
<portlet-entry name="HelloJSP" hidden="false" type="ref" parent="JSP“ application="false">
  <meta-info>
    <title>HelloJSP</title>
    <description>Simple JSP Portlet Example</description>
  </meta-info>
  <parameter name="template" value="hello.jsp" hidden="true"/>
  <media-type ref="html"/>
  <category>jsp.demo</category>
</portlet-entry>
<portlet-entry name="Apacheweek" hidden="false"
    type="ref“  parent="RSS“ application="false">
  <meta-info>
    <title>Apacheweek</title>
  </meta-info>
  <url>http://www.apacheweek.com/issues/apacheweek-headlines.xml</url>
  <category>news.software.opensource</category>
</portlet-entry>
```

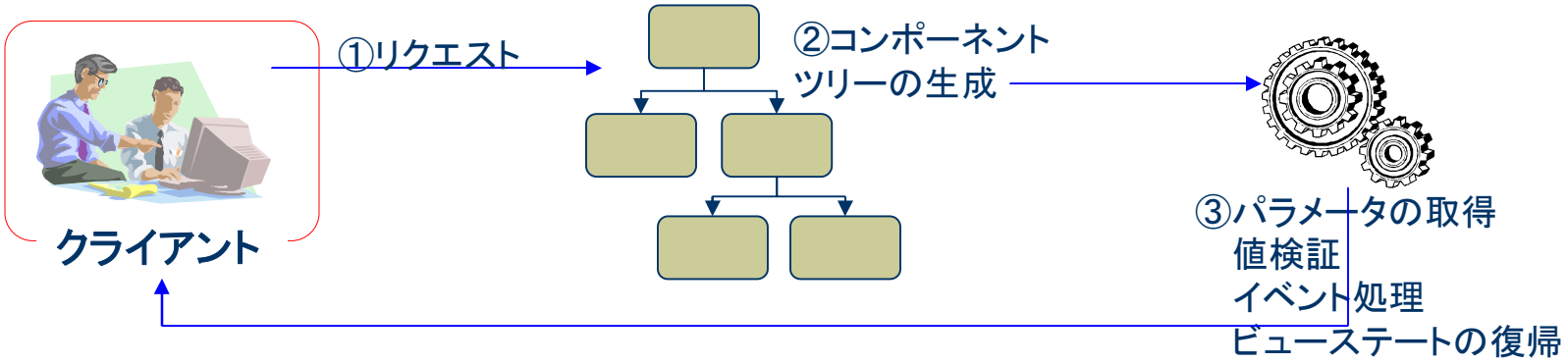
メタデータの設定

実行時の情報

ポータル上のカテゴリ

その他のアプリケーションフレームワーク（4）

- ASP.NET（http://www.microsoft.com/japan/msdn/netframework/）
  - サーバーコントロールとイベントドリブンモデルをベースとした開発環境
  - Visual Studio .NET、Web Matrixなどの開発環境



- Phrame（http://phrame.sourceforge.net/）
  - MVCベースのPHP対応フレームワーク（Strutsを擬した軽量FW）
  - Smarty・XSLT等のView、PEAR DB・ADODB等のModel
  - ArrayListやHashMap、Stack、ListIteratorなどのUtilsクラスを備える

（参考）ASP.NETによるイベントドリブンモデル

```
<script runat="Server">
Dim objDr As SqlDataReader
Sub Page_Load(sender As Object, e As EventArgs)
Dim objDb As New SqlConnection(“.... ")
  Dim objCom As New SqlCommand("SELECT * FROM books",objDb)
  objDb.Open()
  objDr=objCom.ExecuteReader()
  Page.DataBind()
  objDb.Close()
End Sub
</script>
```

抽出したデータのバインド

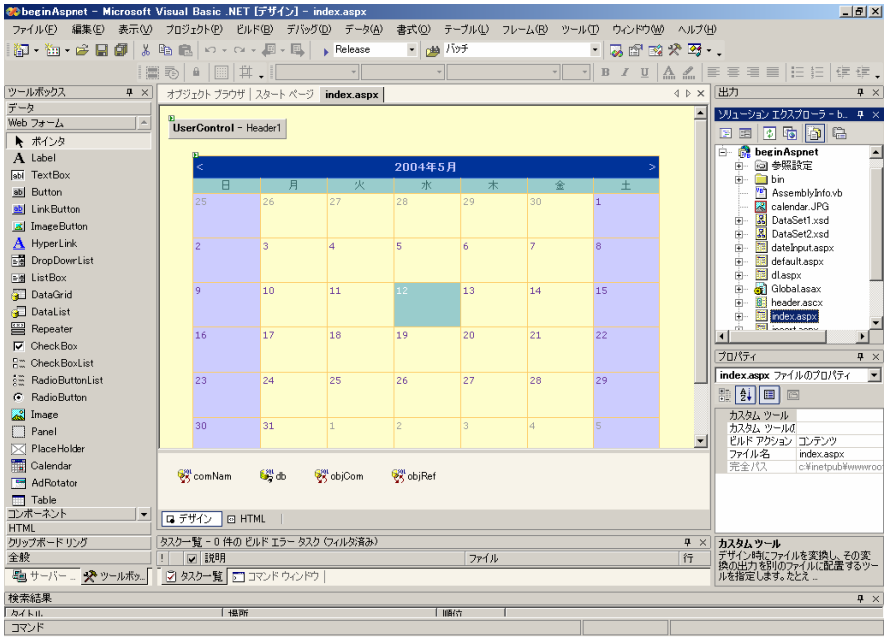
コードとレイアウトとの明確な分離

```
<form runat="server">
  <asp:DataGrid id="objGrd" DataSource="<%# objDr %>" runat="Server" />
</form>
```

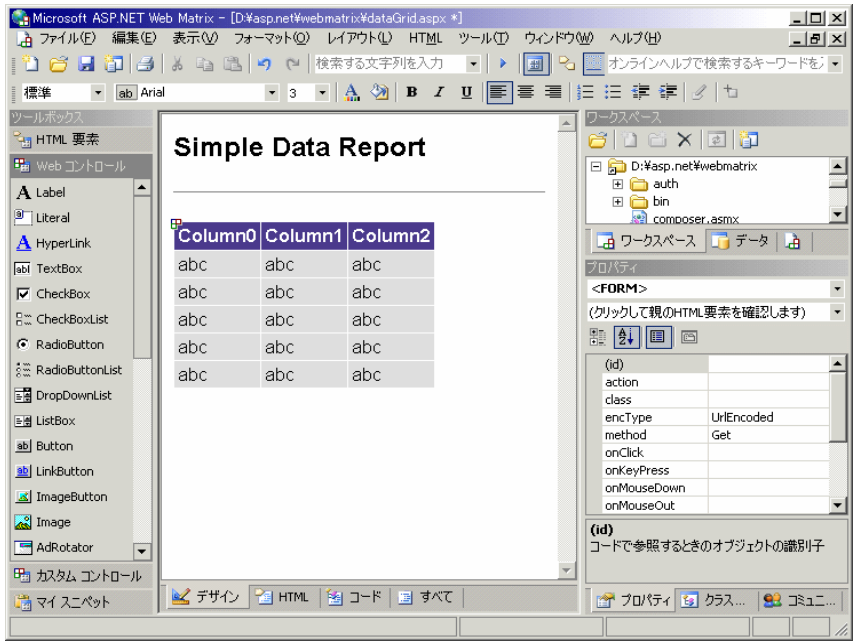
（参考）ASP.NETの統合開発環境
Visual Studio .NET ＆ Web Matrix

- Visual Studio .NET
  - 高機能な.NET対応開発環境
- Web Matrix
  - 無償のASP.NET開発環境。機能に制限、ただし独自機能も

Visual Studio .NET



Web Matrix



（参考）Phrameでマッピングを記述

```
$mappings = array(
  _ACTION_FORMS => array(
      'form' => array(_TYPE => 'HelloForm' )
  ),
  _ACTION_MAPPINGS => array(
    'sayHello' => array(_TYPE => 'HelloAction',
                  _NAME => 'form',
                  _INPUT => 'index.php?view=views/index.xsl',
                  _VALIDATE => 1,
                  _ACTION_FORWARDS => array(
                      'hello' => array(
                          _PATH => 'index.php?view=views/hello.xsl',
                          _REDIRECT => 0 ))))));
```

ActionFormの定義

アクションの定義

（参考）PHPのテンプレートエンジンSmarty

テンプレートファイル（TPL）

```
{$title}
<ul>
{foreach item=i
from=$arch}
  <li> $i </li>
{/foreach}
</ul>
```

コントロール部

```
$ary=array(“JSP”,
  “Servlet”, “ASP.NET”);
$smarty->asign(“arch”,$ary);
$smarty->asign(“title”,”技術一覧”);
```

display

技術一覧

```
<ul>
  <li> JSP </li>
  <li> Servlet </li>
  <li> ASP.NET </li>
</ul>
```

変換結果

（参考）データベースアクセスクラス PEAR DB

通常のMySQL関数

```
$db=mysql_connect("COMPUTER","root","root");
mysql_select_db("sample",$db);
$sql="INSERT INTO bbsmaster(title,nam,sdat,article,passwd)
  VALUES('".$title."','".$nam."','".date("Y/n/d G:i:s")."','".$article."','".$passwd."')";
mysql_query($sql,$db);
mysql_close($db);
```

PEAR DB

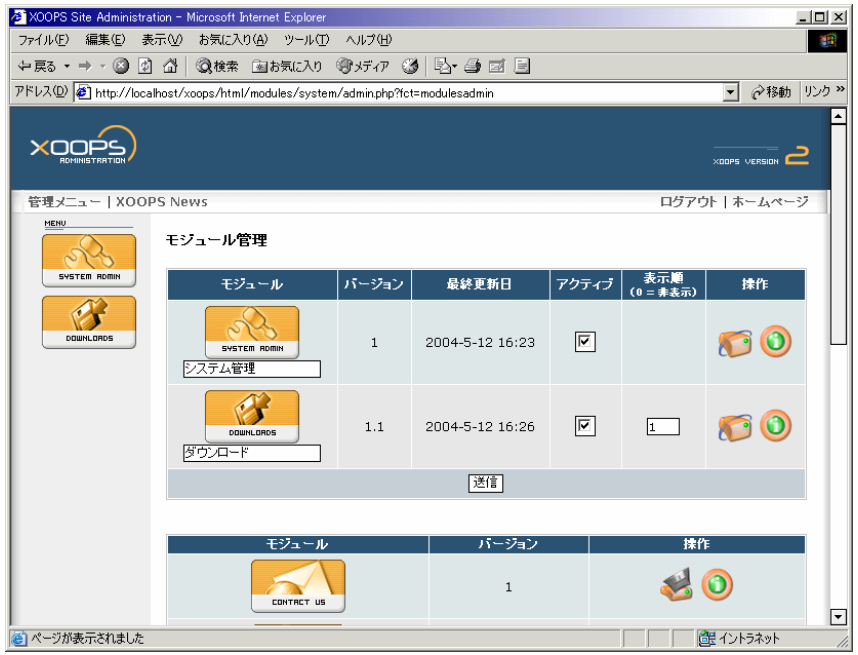
```
require_once("DB.php");
$dsn="mysql://root:root@COMPUTER/sample";
$db=DB::connect($dsn);
$sql="INSERT INTO bbsmaster(title,nam,sdat,article,passwd)
VALUES('".$title."','".$nam."','".date("Y/n/d G:i:s")."','".$article."','".$passwd."')";
$db->simpleQuery($sql);
$db->disconnect();
```

その他のアプリケーションフレームワーク(5)

- **Xoops（http://jp.xoops.org/）**
  - ➢ モジューラブルなコンテンツ管理システム（CMS）
    （インストールからモジュールの追加までがWeb画面上で完結）
  - ➢ 概観のカスタマイズ、通知機能、モジュールによる機能拡張
  - ➢ 独自のモジュール開発も可能

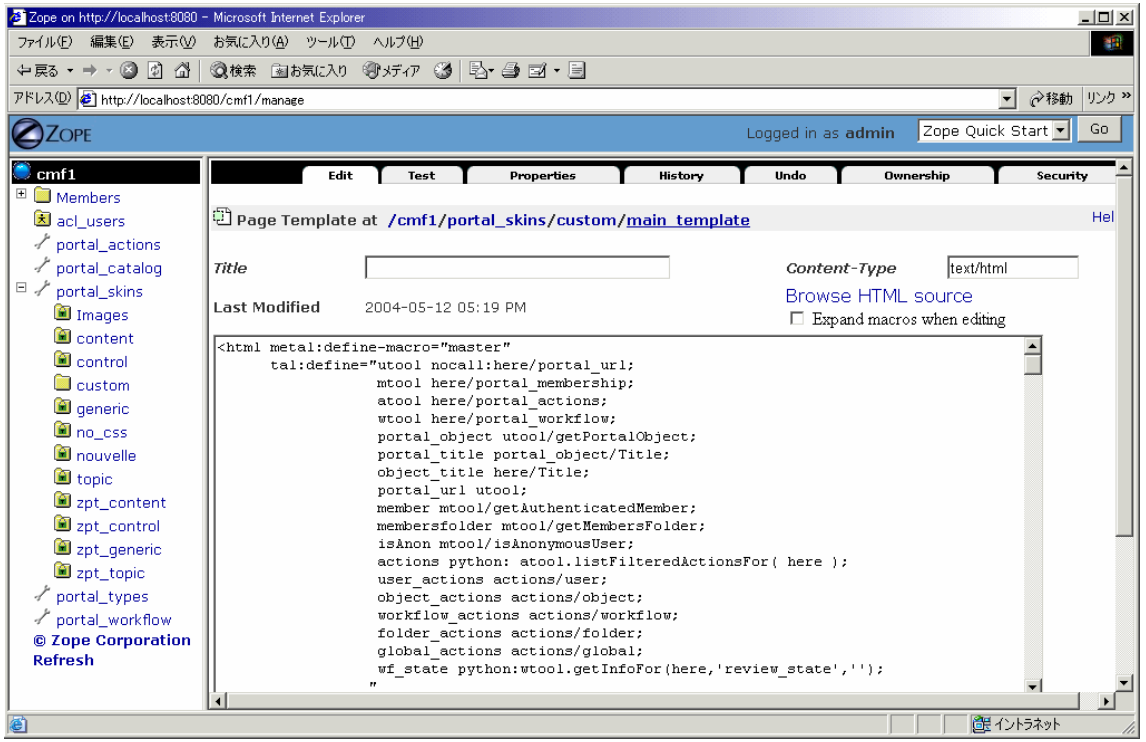
xoops_version.php

```
$modversion['name'] = _MI_CONTACT_NAME;
$modversion['version'] = 1.00;
$modversion['description'] = _MI_CONTACT_DESC;
$modversion['help'] = "top.html";
$modversion['license'] = "GPL see LICENSE";
$modversion['official'] = 1;
$modversion['image'] = "contact_slogo.png";
$modversion['dirname'] = "contact";
```



その他のアプリケーションフレームワーク(6)

- Zope（http://www.zope.org/ ）
  - PythonベースのWebアプリケーションサーバ 兼 コンテンツ管理システム
  - CMF（Content Management Framework）を利用可能
  - ブラウザ上で
    コンテンツ管理が可能



アプリケーションフレームワークの今後
～3つの標準化～

フレームワークそのものとしての機能充実は期待されるものの、むしろ...

- フレームワークとしての標準化
  - JSF（JavaServer Faces）への期待
  - フレームワーク同士の連携モデルが確立すること（Ex. JSR＋Struts）
  - 基盤技術（JSP［EL・JSTL］やEJB）との連携
- 開発方法論の標準化
  - 開発担当者の役割分担
  - 定型的な開発フローの確立
- 統合開発環境（IDE）としての標準化
  - 非プログラマも利用可能なオーサリングツール、IDEは不可欠
    ※ Sun JavaStudio Creator？

まとめ

さまざまなフレームワークが提供されているが、要は目的・状況に応じた使い分けが必要

- 自由な選択か、標準的な実装か
  - 満遍なく機能が実装されている（Ex. ASP.NET、Turbine）
    ○ 垂直立ち上げが可能　　× 固有の実装・環境に依存（適用の制約）
  - 必要最低限の実装（Ex. Struts）
    ○ 導入が容易　　×製品選択の難しさ
- オープンソースか、商用製品か
  - オープンソース
    ○ イニシャルコストの抑制　× 運用・保守コスト増大の可能性
  - 商用製品
    ○ サポートの充実・高機能性　　× 高価