1K−1

Webアプリケーション自動生成のためのアーキテクチャの一提案*

和田 信平　小林 要†
金沢工業大学‡

1　はじめに

Webサイトの構築方法には，静的にHTMLファイルをあらかじめサーバに配置する方法と，動的にHTMLテキストを生成する方法とがある．

近年，サーバのパフォーマンスの向上やアプリケーションサーバの普及などにより，サーバ内での動的なコンテンツ生成が盛んに行われるようになってきた．Apache[1]によるCocoon[1]など，XMLを利用したWebサイト構築技術が注目されている．

本稿では，ユーザが定義したコンテンツデータと処理内容情報をおよび画面構成データ元に，Webアプリケーションを自動生成するためのアーキテクチャを提案する．

2　Webサイト構築の問題点と解決方法

従来型Webサイトの構築作業は，クライアント側でのHTMLファイルの作成から始まる．HTMLファイルをテキストエディタで作成し，ファイル名を決めて保存する．ファイル間の構成を見ながらのリンクの付加，ファイル転送など多くの作業が必要とされる．

HTMLにおけるリンク機能はURIで識別されるため，Webサイト作成者は，自ら作成したファイル名の管理を厳密に行う必要があるため負担が大きい．参照元ファイル名の変更に伴うリンク先の変更は1対多の関係にあるため，リンクの参照先変更は膨大な作業量を要求する．HTMLエディタなどのツールを使うと，リンクの参照先が存在しないケース，どこからも参照されていないファイルがあるケース，などの論理的なエラーには対処できるが，ユーザによるリンクの自己管理という煩雑な作業は避けられない．

また，Webコンテンツには，ユーザの便宜を図るため，繰り返し同じ表現の構成が使われることが多い．たとえば，他のコンテンツへのナビゲーションの役割をするリンクは，表示するページが変わっても繰り返し画面上に現れるように意図的にHTMLに記述してある．本稿では，コンテンツを分割して定義し，実行時に合成する方法と，リンク構造を論理名指定で扱うことにより解決することができた．

3　本システムの概要

本システムはユーザが定義した仕様書に従って，Java Servletを生成するためのアーキテクチャである．本システムではServletを生成するメカニズムはブラックボックス化されており，ユーザはServletのソースを理解する必要はなく，ユーザは自ら定義するデータのみを管理することに専念すればよい．ユーザは希望する画面に対応した定義ファイルを記述し，コンテンツ群を定義することによって，Webアプリケーションを作成することができる．

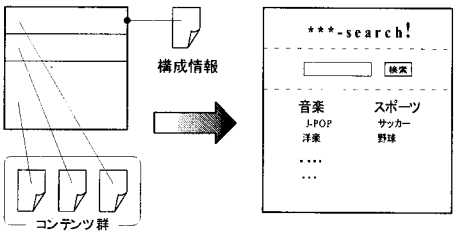

図1: ファイル構成

図1はファイル構成を示した図である．アプリケーション自動生成のために必要なファイルは，構成情報記述ファイルと，コンテンツを定義するXMLファイル群である．

構成情報ファイルは出力するHTMLテキストに含まれるコンテンツ群を定義するためのファイルで，コンテンツ群をどのような形でWebページ上に配置するかのレイアウト情報が含まれる．つまり，コンテンツ群を配

*A proposal of the architecture for automatic Web application generation
†Shinpei Wada, Kaname Kobayashi
‡Kanazawa Institute of Technology, Ishikawa Japan
[1]The Apache Software Foundation

置するための骨格情報と参照されるコンテンツ名を記述する．

コンテンツ群は普通の HTML のような記述方法の静的なコンテンツと，サーバ上で処理を行う必要がある動的なコンテンツにわけられる．動的コンテンツは，処理対象となる XML ファイル等を解析して，どのように処理するかを定義したファイルを作成することにより実現できる．動的なコンテンツは，サーバ上で動作可能なコンポーネントとして生成され，対応する Servlet から必要に応じて呼び出される．

4 アーキテクチャ

本アーキテクチャでは，出力する HTML ページ 1 つにつき，対応した Servlet が 1 つ存在する．Web アプリケーションの設計段階における，ページ単位の構成概念をそのまま反映できるアーキテクチャである．

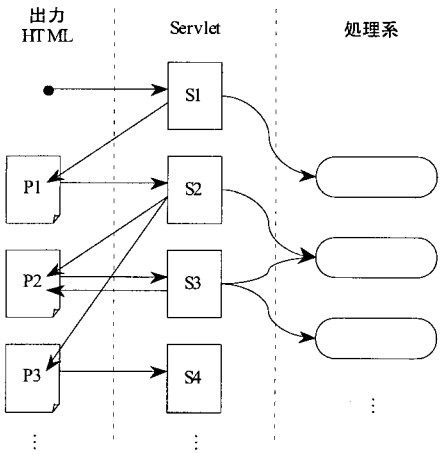

図 2: システム概要図

図 2 は，Web アプリケーションのページ推移を示す一例である．S0 はアプリケーションの開始点となる Servlet で，S0 で生成された HTML テキストの中には次の Servlet に対するリンクが存在する．S0 から HTML テキスト P1 が生成され，P1 から呼び出される Servlet は S1 となる．S1 は P1 専用の Servlet で P1 以外からのリクエストは行なわない．同様に P2 に対応する Servlet は S2 となる．S2 は P1 からのリクエストを受け取り，P2，P3，の HTML テキストを生成している．Servlet の機能は，Web ページからのリクエストを分析して，処理に必要なコンポーネントへ作業を依頼する働きをする．出力する HTML ページの情報は，構成情報ファイルで制御する．

1 つの Servlet は Web ページ 1 画面に対応しており，1 入力 n 出力という関係を基本とする．S2 は P1 からのリクエストを受け取り，P2，P3，の HTML テキストを生成している．

本アーキテクチャを利用すると，Web ページの中のリンクは，1 つ serlvet へのリンクのみとなるため，リンクの対応関係がわかりやすくなる．リンクの管理は物理的な名称（ファイル名）ではなく，論理的な名称で表現する方法を採用した．論理的な名称は次に呼び出す Servlet の URL に依存することがなく，論理名と物理名を対応づける表によって管理されている．リンクを作成する場合，論理名で指定することができるため，ページ同士が互いを知らなくてよく，ページ間の依存関係が少なくなる．

5 おわりに

本稿は，Web アプリケーション自動生成のためのアーキテクチャを提案した．すべての Web ページは Servlet を利用した動的なコンテンツ生成を行っているため，静的に HTML を配置した場合に比べて，動作速度の面では不利になるが，動的なコンテンツ生成を行うことで，リンクを論理名で扱えるというメリットが生まれた．

コンテンツの分割化と合成の基本的な考え方は，JSP[2] の実行時の動的生成の枠組みに似ている．本アーキテクチャでは XML を利用している点で異なるが，コンテンツの動的な生成とコンテンツ群の合成が行える．ユーザはコンテンツデータを入れ替えることで Web アプリケーションの動作の変更が即時に可能となる．

また，同種の Web アプリケーションを作るとき，蓄積したコンポーネントを再利用することも可能で，テスト済みのコンポーネントの組み合わせで信頼性の高いアプリケーションの作成ができる．

参考文献

[1] 中道義之, Coocon, XML PRESS, Vol.1, pp.146-158, 技術評論社, 2000

[2] Duane K. Fields, Mark A. Kolb, 荒井美千子 監訳, JSP による Web 開発 pp.214-224, 翔泳社, 2001