ヒューマンインタフェース　86－1
モーバイルコンピューティング　11－1
（1999．11．25）

CC/PPを利用した多種端末向けWebアプリケーションの検討

松村洋一[†]　山田宏之[††]

従来からのPCに加え，携帯電話，PDA，カーナビ，ゲーム機，ネットワークコンピュータなどからインターネットにアクセスできるようになってきている．現在のWebアプリケーションはPCでの表示を前提としたWebコンテンツを生成するため，これらの情報端末の仕様や性能の影響により正確に表示されないという問題がある．

本稿では，多種多様な情報端末向けWebアプリケーションを構築する上で重要となる端末属性通知方式を提案し，属性情報を基に端末に適したWebコンテンツを動的に生成するWebアプリケーション構築を検討する．

Web Application Development using CC/PP for Infomation Appliances

YOICHI MATSUMURA[†] and HIROYUKI YAMADA[††]

Users will be accessing the Internet from information appliances such as smart phones, PDAs, car navigation systems, game consoles and network computers, in additon to usual PCs. These devices have proper features of specifications and capabilities respectively. Thus, Web contents must be modified for displaying on those devices.

In this paper We propose mechanism informing device capability information, and describe a design of development system of Web applications for many varieties of infomation appliances.

1. はじめに

World Wide Web(以下，Web)が普及し，Webブラウザを使用して世界中の様々な情報にアクセスすることが可能となっている現在，Web技術を利用したアプリケーションが非常に多くなっている．

また，近年の特徴として携帯電話，PDA，カーナビ，ゲーム機，ネットワークコンピュータなどの多種多様な情報端末(以下，端末)が登場し，それぞれによりインターネットへのアクセスが可能になってきている．顕著な例として，Webブラウザ専用端末としてのBrowser Board[1)]，Sega Dreamcast[2)]，携帯電話からのインターネットアクセスとしてNTT DOCOMOのiモード[1)]やWAP(Wireless Application Protocol)がある．特にiモードに至っては2月の開始から10月までの加入者が200万人を数え，急速に普及している．

しかし，これらの端末が普及するにつれ，表示装置(液晶のサイズ・解像度)，入力装置(キーボード，ペン，マウス)，通信機能(モデム，ISDN，PDC，PHS)，ソフトウェア(OS，Webブラウザ)等がそれぞれ異なる仕様で開発されているため，従来のWeb技術では端末に応じた適切なサービスを提供することができないという問題が出てきた．

本稿では，端末に最適化された形での情報交換が可能なWebアプリケーションの構築法を検討する．まず，2章で従来のWebアプリケーションの問題点を整理する．3章では，問題点を解決するために必要な端末の情報を取得する端末属性通知方式，4章では端末に合ったWebコンテンツの生成方法について述べる．5章では，現在実装中のプロトタイプシステムについて報告する．最後に，6章で本論文のまとめを述べる．

2. 従来のWebアプリケーションでの問題点

様々な端末の登場は，インターネット利用者のすそ野が確実に広がっていることを象徴している．しかしながら，端末毎に画面サイズや入力方法等が異なるため，従来のPC上のWebブラウザを想定して構築したWebアプリケーションでは画面に正しい結果が表

† 愛媛大学大学院理工学研究科
Graduate School of Science and Engineering, Ehime University
†† 愛媛大学工学部
Faculty of Engineering, Ehime University

示されないという問題が明らかになってきた．その具体例を以下に示す．

- 大きな画像サイズのコンテンツ
  画面解像度が 320x240 でモノクロ 16 階調のモバイル端末に，解像度 800x600 で 1677 万色の画像に表示させようとした場合，端末には画像の一部しか表示しない，あるいは全く表示できない場合が考えられる．また，画像の一部しか閲覧できないため，重要な情報の場所を見つけることが困難である．さらに，ファイルサイズが大きいためにダウンロードに時間がかかるという問題もある．
- 大容量メモリや速い CPU が必要なコンテンツ
  モバイル端末はコストの観点から少容量メモリ，比較的低速な CPU を使用している．そのためファイルサイズが大きく，CPU の速度を重視する動画や音声等のプラグインを利用した Web コンテンツを扱うのは難しく，Web ブラウザではそのようなコンテンツの閲覧をサポートしない場合が多い．長い文章が書かれた HTML ファイルを表示することもメモリの余分な使用，大幅な読み込み時間等より，適切な Web コンテンツとはいえない．

これらの具体例から PC を前提とした Web コンテンツと情報端末向けの Web コンテンツには明確な相違が見出される．

各端末にはそれぞれにあった利用形態があり，その仕組みを有効に活用するコンテンツでなければならない．このような問題点の解決策として現在，2 つの手法が取られている．

(1) 端末に対応した数の Web アプリケーションの構築
アクセスされる可能性のある全ての端末に対し，それぞれに特化した Web アプリケーションを構築することが考えられる．i モードの場合，画面サイズとして全角文字で最大 10 文字 × 10 行 (100 文字) を表示する専用携帯電話を用いる．また，表示できる画像ファイルは GIF 形式・モノクロ 2 値のみであり，Web コンテンツ記述言語として HTML のサブセットである Compact HTML[5] を採用している．開発者はこれらの仕様を元に i モード専用の Web アプリケーションを構築している．また，WAP サポート端末に対しても独自の Wireless Markup Language(WML) により構築する必要がある．しかし，コンテンツの変化が少ない場合は問題がないが，頻繁に更新する場合には無駄が多くなる．その上，現在の端末の他にこれから登場してくる端末すべてについて考慮することはできない．

(2) Proxy Server での変換
サーバとクライアント (端末) との間に Proxy サーバを置き，個々のコンテンツを変換するというものがある[4]．Proxy サーバは具体的には以下のような処理を行って端末へ転送している．
  - 画像の縮小・減色
  - イメージファイルのフォーマット変換 (JPEG から GIF へ)
  - HTML タグの代用，HTML タグの無視 HTML タグ属性の無視
  - イメージをテキストリンクに変換
  - Web ページの目次化 (リスト化)

  Proxy サーバを使うことによって画像に関してはファイルサイズがかなり減少されることからダウンロードの高速化などの効果が生じる．一方，文書のリスト化や HTML タグの無視は Web コンテンツの作り方によっては必要な情報が削除される可能性がある．これは HTML タグを本来の使い方で利用していない場合が非常に多いからである (<H1>や<TABLE>タグなど)．これは企業データなどを取り扱う場合には正確な情報が求められるため，問題となってくる．

2 つの解決策を利用した場合でもそれぞれに問題点があり，有効な手段とはなりえていない．以上のことから新たな解決策として端末属性情報 (以下，プロファイル情報) を取得し，端末に適した Web コンテンツをサーバ側で正確に生成する Web アプリケーションの構築という考えが登場してきた．次章から新たな Web アプリケーションについて端末属性通知方式，Web コンテンツ生成を述べる．

3. 端末属性通知方式

3.1 Composite Capability/Preference Profiles (CC/PP)

端末のプロファイル情報を記述する方法として，現在，W3C(World Wide Web Consortium) で Composite Capability/Preference Profiles (CC/PP)[7] が議論されている．これは Resource Description Framework(RDF)[8] と呼ばれるリソース記述フレームワークを使って定義したものであり，端末のプロファイル情報 (解像度や CPU などのハードウェア情報，サポートしているコンテンツ記述言語のバージョ

```
<?xml version="1.0"?>
<rdf:RDF
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:prf="http://www.w3.org/TR/WD-profile-vocabulary#">
<rdf:Description about="HardwarePlatform">
<prf:Defaults>
Vendor="Abc"
Model="2160"
Type="PDA"
ScreenSize="640x240x8"
CPU="MIPS"
Keyboard="Yes"
Memory="16mB"
Bluetooth="YES"
Speaker="Yes" />
</rdf:Description>
<rdf:Description about="SoftwarePlatform">
<prf:Defaults>
OS="XXX1.0"
HTMLVersion="4.01"
WML Version="1.0" />
<prf:Modifications
Sound="off"
Images="Yes" />
</rdf:Description>
</rdf:RDF>
```

図 1: ある端末の CC/P フォーマット

ンや画像が表示できるかなどのソフトウェア情報) の記述フォーマットを解説している. 図 1 はある端末のハードウェア・ソフトウェアプロファイル情報を記述したものである.

この例では Abc 社製の PDA, CPU は MIPS でメモリは 16MB, スクリーンは 640x240 で 8bit 対応, キーボードで入力する. ネットワークとして Bluetooth をサポート, ソフトウェアとして HTML4.01, WML1.0(WAP) 対応, 変化する機能として Sound は off, 画像表示可となっている.

CC/PP を用いる利点として RDF, XML を元にしていることから, プログラムが理解可能なため, 自動処理が可能となることが挙げられる.

3.2 端末属性通知方式

前節の W3C の CC/PP 案ではプロファイル情報の記述フォーマットのみを規定しており, 実際にそれをどのようにサーバへ転送するかについては記述されていない. そこで, 本稿では HTTP 環境変数を用いることを提案する.

HTTP 環境変数とは, クライアントが URL でサーバにリクエストを送る際に発信する情報である (図 2). HTTP 環境変数の中には User-Agent というフィールドがあり, クライアントプログラム (Web ブラウザ) の名称が記述されている. 現在, この HTTP 環境変数は, ブラウザの名称や言語 (日本語, 英語等) の情報を取得し, 目的の HTML ファイルへ移動する CGI プログラムで利用されている (図 3). この場合は標準の User-Agent 情報のみを使用していたことから, 単純な移動しかできず, User-Agent 情報から詳細な情報を取得することには限界があった. しかし, User-Agent

```
GET /cgi-bin/echoserver HTTP/1.0
Referer: http://yebisu.cs.ehime-u.ac.jp/~index.html
Connection: Keep-Alive
User-Agent: Mozilla/4.7 [en] (WinNT; I)
Host: yebisu.cs.ehime-u.ac.jp:8080
Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg, image/png, */*
Accept-Encoding: gzip
Accept-Language: ja,en,en-US,en-GB,zh,zh-TW,zh-CN,ko
Accept-Charset: iso-8859-1,*,utf-8
```

図 2: HTTP ヘッダ情報

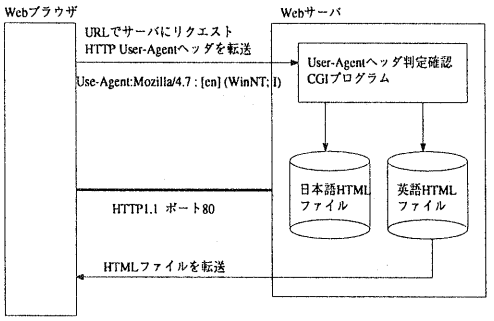

図 3: CGI サンプルプログラム (英語環境に対して)

```
Use-Agent:Mozilla/1.9N : Mabba/2.0
(                                    CC/PP部分
<?xml version="1.0"?>
<rdf:RDF
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:prf="http://www.w3.org/TR/WD-profile-vocabulary#">
<rdf:Description about="HardwarePlatform">
<prf:Defaults>
Vendor="Abc"
Model="2160"
Type="PDA"
ScreenSize="640x240x8"
CPU="MIPS"
Keyboard="Yes"
Memory="16mB"
Bluetooth="YES"
Speaker="Yes" />
</rdf:Description>
<rdf:Description about="SoftwarePlatform">
<prf:Defaults>
OS="XXX1.0"
HTMLVersion="4.01"
WML Version="1.1" />
<prf:Modifications
Sound="off"
Images="Yes" />
</rdf:Description>
</rdf:RDF>
)
```

図 4: User-Agent に埋め込んだ CC/PP の例

では'(' と')' で囲まれた部分にコメントを記述することになっており, その場所に CC/PP を追加することが可能である. 具体的には図 4 のようにして埋め込むことによりサーバへプロファイル情報を転送することが可能となる.

3.3 端末属性処理手順

ここでは端末がどのように CC/PP を処理するかということを議論する. 本稿では, Web ブラウザを中心としているため, Web ブラウザが定期的にプロファ

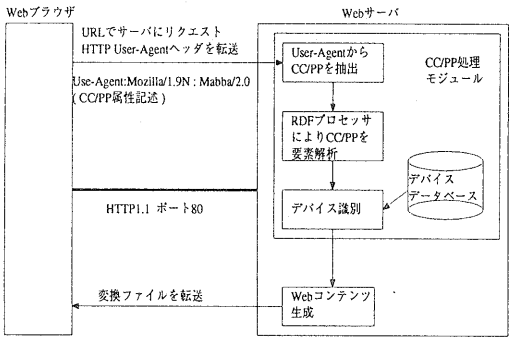

図 5: デバイス認識までの処理手順

```
triple('http://www.w3.org/TR/WD-profile-vocabulary#Defaults',
      'online#HardwarePlatform',
      'Vendor="Abc"
Model="2160"
Type="PDA"
ScreenSize="640x240x8"
CPU="MIPS"
Keyboard="Yes"
Memory="16mB"
Bluetooth="YES"
Speaker="Yes"').
triple('http://www.w3.org/TR/WD-profile-vocabulary#Defaults',
      'online#SoftwarePlatform',
      'OS="XXX1.0"
HTMLVersion="4.01"
WML Version="1.1"').
triple('http://www.w3.org/TR/WD-profile-vocabulary#Sound',
      'online#genid4',
      'off').
triple('http://www.w3.org/TR/WD-profile-vocabulary#Images',
      'online#genid4',
      'Yes').
triple('http://www.w3.org/TR/WD-profile-vocabulary#Modifications',
      'online#SoftwarePlatform',
      'online#genid5')
```

図 6: CC/PP 解析結果

イル情報を OS から取得し，ファイル.ccpp ファイルとして保存しておく．そして，サーバにアクセスするタイミングで HTTP User-Agent の中に.ccpp の内容を埋め込む形態をとる．

デバイスの認識までの処理をまとめると図 5 のようになり，次のステップを順に実行していく．

(1) 端末から User-Agent ヘッダが転送
(2) User-Agent ヘッダから CC/PP 記述部分を抽出
(3) RDF プロセッサにより CC/PP を要素解析
(4) デバイスデータベースと照合してデバイスの状態を識別

実際に図 1 の CC/PP に対して 3 番目の CC/PP の要素解析を行った結果の一部を図 6 に示す．このように端末のプロファイル情報を取得することが可能になる．

4. Web コンテンツ生成

現在の HTML では基本的に文書内にレイアウト情報を含むケースが多い．変更することが許されないような重要な文書や自分に変更する権限がない文書 (他人の Web コンテンツ等) に対してレイアウトを変更することは不可能である．

しかし，文書の論理構造とレイアウト等のプレゼンテーションとを分離させると上記の問題を解決できる．論理構造とレイアウトを明確に分離させることにより，元の文書に変更を加えずにレイアウト情報が記述されたスタイルシートを変更するだけで見栄えを変更することが可能となる．

多種多様な端末向けの Web コンテンツを作成する場合では，モノクロ表示しかできない端末のスタイルシートや，携帯電話のスタイルシートを作成すれば 1 つの文書をいくようにも見せることができ，あらゆる場面で利用できるようになる．

文書とレイアウトの分離に対しては Extensible Markup Language(XML)[6] と XSL Transformation(XSLT)[10] を用いることを提案する．

```
<?xml version="1.0" encoding="shift_jis"?>
<presentation>
<title>XMLサポートブラウザについて</title>
<section>
<title>特長1</title>
<paragraph>
XML Parserを標準装備しXMLをフルサポート．
XSLTをサポートしたことによりブラウザ上
での表示が可能となった。
</paragraph>
</section>
<section>
<title>特長2</title>
<paragraph>
Windows環境でのみ動作する．
</paragraph>
</section>
</presentation>
```

図 7: XML 文書サンプル

XML は文書の用途に従って独自のタグを定義することができる言語である．例えば，図 7 の XML によるレポートでは，<section>で項目を 2 つに分け，<title>で章のタイトル，<paragraph>に具体的な特長を記述した詳細内容を記述している．

したがって，XML を使用することによりタグが要素の内容を表しているため，タグ中のデータを識別し，文書の中の情報を抽出することが可能になる．

一方，XSLT は XML 文書を他のフォーマットの文書に変換するための仕組みである．XSLT は XML 文書を読み込み，XSL ファイルの指示に従って文書を別のフォーマットの文書への変換処理を行う．例えば，独自に定義したタグを用いて作成した XML ドキュメントでは，専用タグを表示できるブラウザを作らない限り望む表示はできない．しかし，XSLT を利用するとこの XML 文書を HTML や WML へと変換するこ

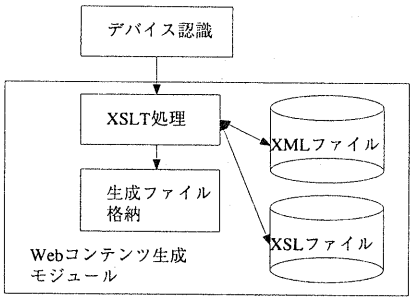

図 8: Web コンテンツ作成手順

とが可能になる．

図 4の XML 文書を携帯電話向けに表示させる場合，<section>中に含まれる<title>の内容をリストアップして目次を作成し，それをクリックすることにより詳細項目を表示するのが有効な手段である．それには<title>部分をリスト構造に変換させる XSL ファイルを作成し，XSLT プロセッサを用いて変換することが必要である (図 8)．

5. プロトタイプシステム

現在，これまでに検討した端末属性通知，Web コンテンツ作成手法のコンセプトの確認や実用上の問題点を洗い出すためにプロトタイプを実装中である．

プロトタイプの構成を図 10 に示す．

想定端末

クライアント端末として Compact HTML サポート携帯電話，WAP サポート端末，Microsoft Windows CE Handheld Edition 搭載端末の 3 種類について検証していく．

クライアント

クライアントでは専用のクライアントプログラムを Java 言語を用いて Windows NT 4.0 Workstation 上で作成する．これはサーバへ CC/PP の情報が格納された User-Agent を HTTP に基づいて送信し，最終的にサーバからプロファイル情報に沿った適切なコンテンツ (ファイル) を取得するものである．3 種類の端末のプロファイル情報から CC/PP ファイルをあらかじめ作成し，切り替えを行い，ネットワークへアクセスする．今回は動的なプロファイル情報取得は想定しないものとする．

サーバ

Web サーバとして Windows NT4.0 Server の Internet Infomation Server を利用し，その上にサーバサイドの Java プログラムである Servlet を実行する環境 JRun[11] を整備する．(1)～(5) までの処理を全て

```
<?xml version="1.0" encoding="shift_jis"?>
<!DOCTYPE WML PUBLIC "-//WAPFORUM//DTD WML 1.0//EN"
    "http://www.wapforum.org/DTD/wml.xml">
<WML>
<HEAD>
<META USER-AGENT="VND.UP.MARKABLE" CONTENT="TRUE"/>
</HEAD>
<CARD TITLE="XMLサポートブラウザについて">
XMLサポートブラウザについて
<SELECT>
<OPTION ONCLICK="#特長1">特長1</OPTION>
<OPTION ONCLICK="#特長2">特長2</OPTION>
</SELECT>
</CARD>
<CARD NAME="特長1">
<P>
 XML Parserを標準装備しXMLをフルサポート．
XSLTをサポートしたことによりブラウザ上
での表示が可能となった．
</P>
</CARD>
<CARD NAME="特長2">
<P>
Windows環境でのみ動作する．
</P>
</CARD>
</WML>
```

図 9: WML 文書

Java アプリケーションとして作成する．(1)～(3) を CC/PP の処理，(4)～(5) を Web コンテンツ生成として 2 つの Servlet プログラムを作成し，内部アプリケーションの管理を行う．これらはサーブレット・チェーンとして働くものである．XML 文書は図 7 のようなある程度形式の決まった文書をサンプルとし，1 つの文書に対して端末毎に XSL ファイルを 3 つ用意する．

最終的に WAP サポート端末に対して全処理を行った場合，図 7 の XML 文書は以下の WML 変換結果 (図 9) と表示予想 (図 11) が得られると考えられる．

6. おわりに

本稿では W3C で議論されている CC/PP の転送方式を提案し，実装方法を明確にした．この方式により，多種端末向けに Web コンテンツを作成する Web アプリケーションを構築することが可能となる．

また，W3C では端末属性通知方式として，HTTP1.1 の拡張とした HTTP Extension Framework[12] 上に CC/PP 交換プロトコルを定義することを提案している[13]．Web コンテンツ生成に関しても HTML タグのモジュール化[14] により，端末に合ったタグセットのみを送信することも検討されている．

今後はこの拡張プロトコルを用いた方式や HTML のモジュール化についても検討したい．

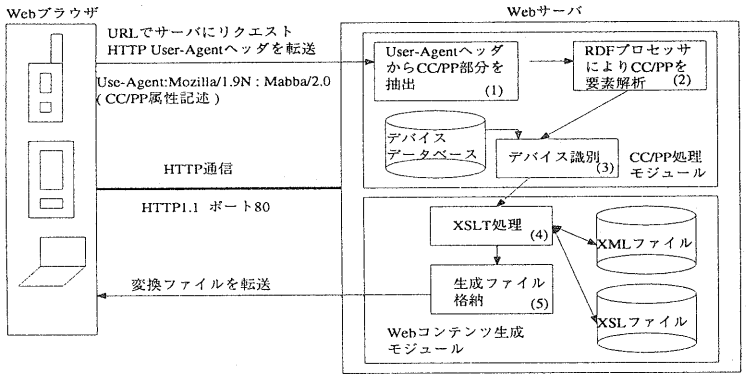

図 10: プロトタイプシステム

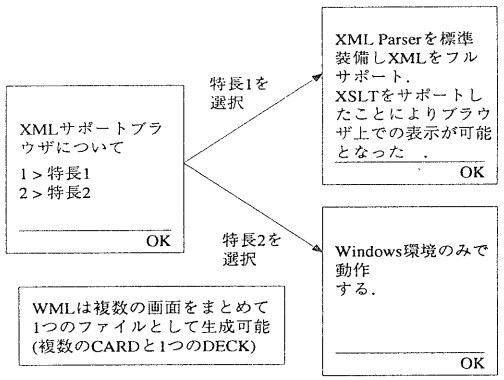

図 11: WML 文書の端末表示予想

参 考 文 献

1) NTT 移動通信網株式会社: http://www.nttdocomo.co.jp/
2) 株式会社セガ・エンタープライゼス: http://www.sega.co.jp/
3) WAP Forum: http://www.wapforum.org/
4) Spyglass: http://www.spyglass.com
5) Tomihisa Kamada:Compact HTML, W3C, 1998, available at http://www.w3.org/Submission/1998/04/
6) Tim Bray, Jean Paoli, C. M. Sperberg-McQueen:Extensible Markup Language(XML) 1.0, W3C, 1998, available at http://www.w3.org/TR/REC-xml/
7) Franklin Reynolds, Johan Hjelm, Spencer Dawkins, Sandeep Singhal:Composite Capability/Preference Profiles (CC/PP), W3C, 1999, available at http://www.w3.org/TR/NOTE-CCPP/
8) Ora Lassila, Ralph R. Swick:Resource Description Framework(RDF) Model and Syntax Specification, W3C, 1999, available at http://www.w3.org/TR/REC-rdf-syntax/
9) Tomihisa Kamada, Tomohiko Miyazaki: Client-Specific Web Services by Using User Agent Attributes, W3C, 1997, available at http://www.w3.org/TR/NOTE-agent-attributes/
10) James Clark:XSL Transformation(XSLT), W3C, 1999, available at http://www.w3.org/TR/xslt/
11) JRun: http://www2.allaire.com/Products/Jrun
12) P. Leach:HTTP Extension Framework, IETF, 1999, available at http://www.w3.org/Protocols/HTTP/ietf-http-ext/
13) Hidetaka Ohto, Johan Hjelm:CC/PP exchange protocol based on HTTP Extension Framework, W3C, 1999, available at http://www.w3.org/TR/NOTE-CCPPexchange/
14) Murray Altheim, Frank Boumphrey, Sam Dooley, Shane McCarron, Ted Wugofski: Modularization of XHTML, W3C, 1999, available at http://www.w3.org/TR/xhtml-modularization/
15) 北山文彦, 広瀬紳一, 久世和資：多種端末向けビジネスオブジェクト Web アプリケーション構築システムのプロトタイプ実装, 情報処理学会オブジェクト指向 '98 シンポジウム (1998)
16) 村田 真ほか:XML 入門, 日本経済新聞社 (1997)