システムソフトウェアと
オペレーティング・システム 92－17
（２００３．２．２８）

携帯端末向け拡張可能なブラウザアーキテクチャ

才田 好則† 稗田 諭士† 千嶋 博† 佐藤 直樹† 中本 幸一†

近年，Web ブラウザや Java 実行環境を備えた携帯電話が増加し，これらを利用した様々なサービスが提供されている．本研究では，Java アプリケーションからのコントロール及び機能拡張を可能とするようなブラウザアーキテクチャを提案する．このアーキテクチャは，Java アプリケーションに対して，ブラウザコントロールのためのインタフェース，ブラウザで発生する各種イベントの受信手段，及びドキュメントへのアクセスインタフェースを提供するものである．これによって，ブラウザ機能を利用または拡張したサービスの容易な構築及び提供が可能になる．

An Extensible Browser Architecture for Mobile Terminals

YOSHINORI SAIDA, † SATOSHI HIEDA, † HIROSHI CHISHIMA, †
NAOKI SATO† and YUKIKAZU NAKAMOTO†

The number of mobile terminals with a Web Browser or java execution environment has increased in recent years, and many diverse mobile terminal services using the Internet have evolved. However, there are still limitations that it is difficult to change or replace browser functionalities in mobile terminals. We proposed an extensible browser architecture (EBA) to extend the functionalities. The EBA consists of two layers; browser foundation functionalities and browser applications. The EBA provides the core functions of the browser as browser components that are available from the Java runtime environment. The browser applications, which are written in the Java language and downloadable, utilize the browser foundation functionalities to provide varieties of browser applications.

１．はじめに

近年，ブラウザや Java 実行環境を備えた携帯電話が増加し，これらを利用した様々なサービスが提供されている．また，携帯電話本体が備える機能も，カメラ機能，GPS 機能，赤外線インタフェース機能等，多様化している．その中でブラウザは，コンテンツの表示機能，ダウンロード機能，フォーム部品からなる UI(User Interface)機能を備えているため，単なるコンテンツのビューワとしてだけではなく，その上でサービスを提供するための主要アプリケーションとしての役割も担っている．例えば携帯電話においては，メーラ，ユーザオプション設定，着信メロディーや Java アプリケーションのダウンロード及び管理アプリケーション等で，ブラウザの機能が利用されている場合がある．これら以外にも，今後携帯電話に追加される機能の設定や処理結果の表示時に，ブラウザの機能が利用されることが考えられる．

また，ブラウザ本来の機能であるコンテンツ表示機能については，ブラウザが表示及び再生対象とするものとして各種マークアップ言語，画像（静止画，動画）及び音声，スクリプト言語等がある．これらの言語及び画像の仕様は年々バーションアップされ機能拡張されており，キャリアやベンダ，サービスプロバイダがさらに独自拡張を行うケースもある．しかしながら現在の携帯電話は，搭載 OS の制限により PC のように実行ファイルやライブラリをダウンロー

†日本電気(株) NEC ネットワークス
ネットワークス開発研究所
Development Laboratories, NEC Networks,
NEC Corporation

ドしてインストールする機能を備えていないため，製品発売後のアプリケーション更新や機能拡張は不可能である．同様の理由で，PC 上のブラウザに対してダウンロードにより機能追加が可能となるプラグインモジュールに関しても，携帯端末上ではダウンロードしてのインストールはできない．また上述のブラウザを利用したアプリケーションにおいても，発売後にその機能拡張や更新を行うことはできない．従って例えばサービスプロバイダがコンテンツ言語の仕様を拡張して新サービスを提供しようとした場合，PC の場合はそれに対応した新規のブラウザをダウンロードして対応するということが可能であるが，携帯電話の場合は既に発売済みの製品に対してはそのサービスに対応させることができない．このような機能拡張を行う際は，新機種発売の際に予めその機能を実装したアプリケーションを組み込んで出荷するしかなく，特にサービスプロバイダの観点からは自由なサービス提供を難しくする要因の一つになっている．

ここで，Java アプリケーションはダウンロードによる実行が可能だが，現状の日本における携帯 Java サービスでは数十 KB 以内という Java アプリケーションのサイズ制限があり，大規模なアプリケーションの構築は無理である．

このような状況において本稿では，サービスプロバイダが新しいサービス提供のためにブラウザ機能を製品発売後に独自拡張できるようなアーキテクチャの提案を行い，その設計，実装，適応例について述べる．次章で今回提案する拡張可能なブラウザアーキテクチャの構成を述べ，３章ではこれを利用したサービス例について述べる．４章ではこのアーキテクチャの実装について述べ，５章で考察及び課題について記す．

2．拡張可能ブラウザアーキテクチャ

前章で説明した現状を踏まえ，本稿で解決しようとしている課題は，

- 携帯端末上のブラウザに対して，ダウンロードによる機能追加，更新及びプラグインモジュールの追加ができない，

ということである．

我々が本稿で提案する携帯端末向け拡張可能ブラウザアーキテクチャ EBA (Extensible Browser Architecture)は，上述の問題を解決するためのものである．EBA では，ブラウザ機能の Java 部品化及び DOM (Document Object Model)[1] インタフェースの提供を行うことで，Java の持つ自由度と動的拡張性を，ブラウザ機能を利用したサービスにもたらすことを目的としている．具体的には，サービスに必要な機能拡張部分を Java アプリケーションで記述し，これを携帯電話上にダウンロードし，ブラウザと連携して動作することで，新サービスに対応したブラウザアプリケーションを実現できる．ブラウザの基本機能に関しては，携帯電話に予め組み込まれているブラウザモジュールを利用するので，ダウンロードする Java アプリケーションのサイズも抑えられる．これによりサービスプロバイダは，ブラウザを利用した新サービスを独自に定義し，提供することができる．

このアーキテクチャは次の特徴を備えている．

(１)ブラウザ機能の Java 部品化

(２)DOM インタフェースを介したドキュメント情報の共有

(３)ブラウザイベントの Java アプリケーションへの通知

(１)は，予め携帯電話に組み込まれたブラウザの機能を，Java アプリケーションから利用可能なブラウザ部品として提供する．Java アプリケーションとブラウザが連携して動作することで，既存のブラウザに対して新機能を追加することが可能となる．(２)及び(３)は，これまでブラウザ内部に閉じていた表示ドキュメントに関するデータと，ブラウザ内部で発生するイベ

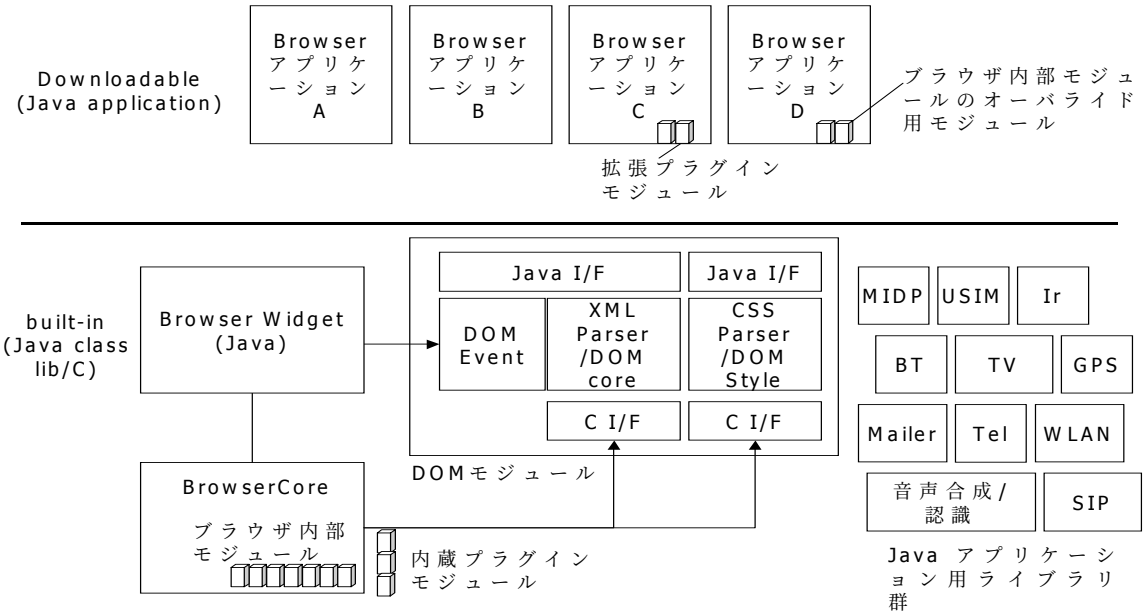

図1 EBA の構成図

ント情報を，DOM Core，DOM Style 及び DOM Event インタフェースを介して Java アプリケーションと共有するためのものである．DOM インタフェースは，World Wide Web Consortium(W3C)で標準化されている，アプリケーションから構造化ドキュメントへアクセスするためのインタフェース仕様である．このうち DOM Core インタフェースは，XML[2)]形式のドキュメントへのアクセス手段を提供するものである．一方 DOM Style インタフェースは，Style Sheet ドキュメントへのアクセス手段を提供する．DOM Event インタフェースは，ブラウザで発生した各種イベントを，アプリケーションに通知するためのものである．DOM Event インタフェースで定義されているイベントとしては，ブラウザ上のフォーカス移動イベント，キー入力イベント等がある．

図１に，EBA のソフトウェア構成を示す．図中の各モジュールの説明は次の通りである．

● DOM モジュール

DOM Core，DOM Event 及び DOM Style インタフェースを実装したモジュール．また，XML ドキュメントをパースする XML Parser，Style Sheet ドキュメントをパースする Style Sheet Parser を備える．DOM Core インタフェースは，XML Parser が XML ドキュメントのパース処理を行った後に生成した解析済みデータに対して，参照及び変更手段を提供する．DOM Style インタフェースも同様に，Style Sheet Parser が Style Sheet ドキュメントのパース処理を行った後に生成した解析済みデータに対して，参照及び変更手段を提供する．このモジュールは，後述する Browser アプリケーションと，Browser Core の両者から利用される．

● Browser Core

コンテンツの取得，レンダリング，表示及び UI 操作等，ブラウザの一般的な基本機能を提供するモジュール．Browser アプリケーションによるプラグインモジュール拡張や，内部モジュールのオーバーライドを想定した構造を備えることにより，Browser アプリケーションでの内部モジュール置き換えや，Browser アプリケーションをブラウザのプラグインとして利用することも

可能となる．次章で具体例を説明する．

また本モジュールでコンテンツの表示処理を行う際には,DOM モジュール内の Parser 及び DOM インタフェースを用いる．これにより，Browser Core が現在表示しているコンテンツのドキュメント情報を，Browser アプリケーションからも DOM インタフェースを介して参照及び変更可能となる．また Browser Core で発生したイベントは，DOM Event インタフェースを使用して Browser アプリケーションに伝達される．

● Browser Widget

Browser アプリケーションから GUI 部品の一つとして利用可能な，Java で記述されたブラウザ部品モジュール．Browser アプリケーションは，Browser Widget を使用することで，ブラウザの画面を携帯端末のディスプレイ上に表示することができる．コンテンツの表示，ヒストリの遷移といった Browser Core モジュールが提供する機能は，このモジュールを介して Browser アプリケーションに提供される．

● Java アプリケーション用ライブラリ群

J2ME の MIDP（Mobile Information Device Profile）[3]環境等で提供されている，Java アプリケーションから利用可能な機能モジュール群．カメラ機能，GPS 機能等を始めとして，今後様々な機能が携帯電話に搭載される見込みであるが，それらの機能をコントロールするためのライブラリ群である．Browser アプリケーションは，ここで提供される機能を Browser Core のブラウザ機能と組み合わせて利用することができる．

● Browser アプリケーション

予め携帯端末に組み込まれた Browser Widget，DOM，Java アプリケーション用ライブラリ群の各モジュールを使用して，機能拡張されたブラウザサービスを実現する，Java アプリケーション．ネットワーク経由で適宜ダウンロードされ，複数の Browser アプリケーションが携帯端末内に格納可能である．それぞれの Browser アプリケーションがそれぞれ独自のブラウザサービスに対応する．

3．サービスの例

本章では，EBA を利用したブラウザの機能拡張及びブラウザを利用したサービスの例を示す．

・コンテンツ記述言語拡張対応

携帯電話上のコンテンツサービスでは，HTML[4]や XHTML[5]言語に対してキャリアやベンダが独自拡張を行うケースがあり，同様にサービスプロバイダが仕様拡張を行うことも考えられる．また，Web サービスや電子 Commerce の分野では，XML 言語をベースにした様々なデータが取り扱われることが想定される．このような Browser Core が対応していない形式のドキュメントに関しては，その処理を DOM Core インタフェースを通して Browser アプリケーション側で行うことができる．例えばタグを拡張してコンテンツ中の一部分を音声合成機能を使って読み上げさせるサービスを行う場合，拡張されたタグの処理及び音声合成機能の呼び出しを Browser アプリケーション側で行い，従来どおりのコンテンツ部分の表示は Browser Core 側で行うことで実現できる．ここで音声合成機能は，Java アプリケーション用ライブラリ群として予め携帯電話に組み込まれているとする．

・HTTP ヘッダ拡張対応

Cookie[6]や CC/PP（Composite Capabilities / Preference Profile）[7]といった技術では，HTTP（HyperText Transfer Protocol）[8]のヘッダ部分を利用して情報のやり取りを行っている．Browser Core がこれらの機能に対応し

ていない場合も，Browser アプリケーション側で HTTP ヘッダ処理を行うことで，これらの機能に対応することができる．具体的には，Browser Core が HTTP 通信を行う際に発生するイベントを Browser アプリケーション側で受信し， HTTP ヘッダの解析処理及び，Browser Core が送信する HTTP ヘッダの内容を追加修正するという処理を Browser アプリケーション側で行うことで対応できる．

・Browser Core 内部モジュールの機能更新

Browser Core で発生したイベントは，DOM Event インタフェースを通して， Browser アプリケーションへと伝達されるが，その際 DOM Event インタフェースの仕様を利用し，該当イベントに対する Browser Core 側の処理をキャンセルさせ，Browser アプリケーション側で該当イベントに対応する処理を実行することができる．この仕組みを利用して，Browser Core 内部モジュールの機能更新を Browser アプリケーションで行うことが可能である．

例えば Browser Core でヒストリ情報の変更イベントが発生した場合，通常は Browser Core は自身のヒストリ管理モジュールで管理しているヒストリ情報を更新する．しかし Browser アプリケーションがヒストリ情報変更イベントを受け取った際に，Browser Core 側での対応処理のキャンセルを指示することで，Browser Core がヒストリ情報の更新処理を行わないように指定できる．その場合，Browser アプリケーション側でヒストリ情報を管理し，Browser Core 側においてヒストリのフォワード方向またはバック方向移動の操作が起こった際に，Browser アプリケーションが管理しているヒストリ情報に合わせて適切な URL を Browser Core に渡して表示を指示することで，Browser Core 側のヒストリ管理モジュールを Browser アプリケーションがオーバーライドすることが可能となる．Browser アプリケーションにおけるヒストリ管理方法を機能拡張することにより，ブラウザのヒストリ機能の拡張が実現できる．

ブラウザで発生するイベントの種類を適切に定義し，そのイベントに対応する処理を Browser アプリケーション側に任せ，本来のブラウザ側の処理をキャンセル可能とするような構造を Browser Core に持たせることで，様々な Browser Core の内部機能を Browser アプリケーション側でオーバーライドできるようになる．

・プラグイン

現在，携帯電話用のブラウザでは，PC 用ブラウザのようなダウンロードによるプラグインモジュールの追加はできない．しかし本稿で示した仕組みにより，Browser アプリケーション側で DOM インタフェースを使用してコンテンツのドキュメント情報を参照し，ドキュメント内のプラグイン呼び出し情報を取得して指定されているプラグイン処理を行うことで，プラグイン機能の追加を Browser Core に対して行うことができる．

4. EBA の実装

今回，図１で示した EBA の各モジュールを，図２に示す PC 環境上に実装した．OS は WindowsNT2000 で，この上に携帯電話の OS，各種システムコール，UI をエミュレーションする携帯電話エミュレータを実装した．また，CLDC（Connected Limited Device Configuration）[9] に対応した Java VM 及び MIDP 対応の Java ライブラリをこの環境上に準備した．この Java 実行環境上で，EBA の Java で記述されたモジュールは実行される．以下，EBA で定義した各モジュールは，次のように実装した．

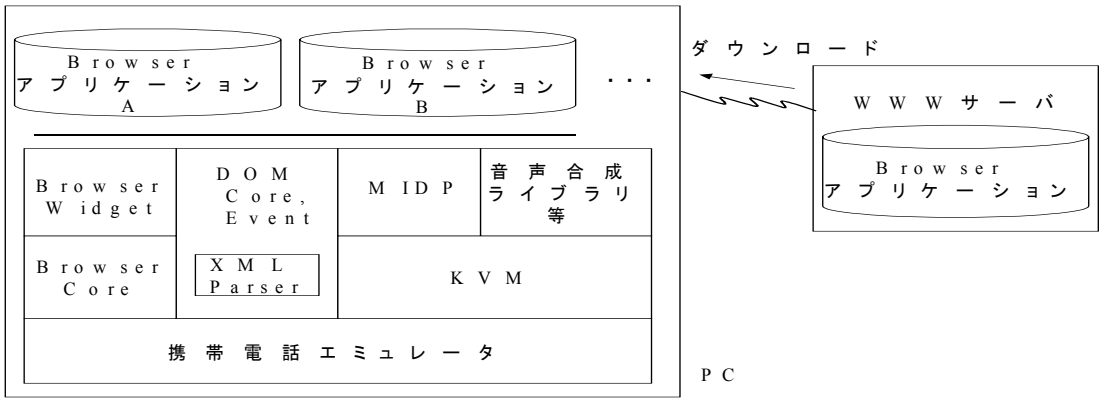

図2 EBA の実装例

● DOM モジュール

DOM モジュールのうち，DOM Core モジュールは DOM Core Level２仕様に，DOM Event モジュールは DOM Event Level３仕様に対応した．DOM Event で Level３仕様を採用したのは，キー入力イベントを取り扱う DOM Text Event への対応が Level３以降であるためである．Browser Core モジュールは Style Sheet に未対応のため，今回 DOM Style モジュールは実装しなかった．XML Parser モジュールは，XML1.0 に対応している．この DOM モジュールは，Java アプリケーションである Browser アプリケーションと，C 言語で記述されている Browser Core の両者から利用可能なように実装した．

● Browser Core

今回実装した Browser Core は，C 言語で記述した．この Browser Core は，XHTML Basic1.0 に対応し，Style Sheet には対応していない．また，XHTML コンテンツを表示する際は，DOM モジュールの XML パーサ及び DOM Core モジュールの C 言語用インタフェースを使用して処理を行っている．また今回以下に示すブラウザイベントを，Browser コアで発生させるようにした．

・DOM UI Event，DOM Text Event，HTML Event

いずれも DOM Event インタフェースで定義されたイベント．DOM UI Event はフォーカスの移動，フォーカス対象の選択動作時に，DOM Text Event はキー入力，文字列入力時に発生する．HTML Event はコンテンツ表示，スクロール等，ブラウザ全般の動作時に発生する．

・Browser Event

今回独自に定義したイベント．特に，上記 HTML イベントでカバーされていないブラウザの動作に関連したものとなっており， サブコンテンツ（HTML コンテンツ中のイメージデータ等）ダウンロードイベント，ヒストリ情報変更イベント，ユーザ認証イベント，アイコン表示変更イベント等からなる．

● Browser Widget

表１は，今回実装した Browser Widget モジュールの Java クラスの主なものである．また BrowserService クラスが提供する，Browser Core の機能を利用するためのメソッドとしては，以下のようなものを準備した．

- URL を指定してのコンテンツ表示指示
- コンテンツデータを指定しての表示指示
- ヒストリ移動，コンテンツ情報，ヒストリ情報取得，等．

● Java アプリケーション用ライブラリ群

Java アプリケーション用ライブラリとして，音声合成ライブラリを準備した．音声合成ルーチンは,Microsoft の SpeechSDK5.1 を使用した．

表１　主なクラス一覧

クラス名	概要
BrowserWidget	ブラウザ画面を表示するための GUI 部品クラス
BrowserService	Browser Core の機能を利用するためのインタフェースを提供するクラス
ContentInfo	コンテンツ情報を保持するクラス
HistoryList	ブラウザのヒストリ情報を保持するクラス
FocusInfo	ブラウザのフォーカス情報を保持するクラス
ScrollInfo	ブラウザのスクロール状態を保持するクラス

● Browser アプリケーション

今回アプリケーション例として，フォーカスが設定された個所のテキストを音声で読み上げる，音声読み上げブラウザを作成した．このアプリケーションでは，フォーカスが移動した際に Browser Core が発生するフォーカスイベントを DOM Event インタフェースを使用して受信し，その際にフォーカス設定部位のテキストを DOM Core インタフェースを使用して取得，このテキストを音声合成ライブラリを用いて読み上げさせるという処理を行っている．

また，擬似 Push サービスを実現するアプリケーションも作成した．このアプリケーションでは，Browser アプリケーション側でサーバのコンテンツを監視し，コンテンツが更新されたらサーバからこれをダウンロードし Browser Core に渡し，コンテンツの表示処理を指示するという処理を行い，擬似的な Push サービスを実現している．

5．考察及び課題

本章では，EBA を利用した際の，

- セキュリティ
- アプリケーションのサイズ

の２点に関して考察する．

まずセキュリティに関してであるが，現在の携帯電話における Java アプリケーションのダウンロードサービスでは，いくつかの機能に関して制限が設けられており，これによってセキュリティが確保されている．どういった機能が制限されているかに関しては，Java アプリケーションサービスを提供しているキャリア，ベンダによって異なるが，例えば通信機能に関して，ある特定のサーバとの通信のみしか許可されないといったような制限が設けられているケースがある．EBA でのブラウザサービスも，携帯電話上の Java アプリケーションとして実装される場合はこれに従う必要がある．

Browser Core が実現するブラウザ機能は，個人データにアクセスする機能などは持たず，内部的には HTTP 通信，コンテンツの表示処理，UI 操作を行うのみなので，Java で記述された Browser アプリケーションが携帯電話用 Java アプリケーションの規則に従えば，特にセキュリティ上の問題はないと言える．ただし，携帯電話用 Java アプリケーションの制限として通信先サーバの制限が指定されている場合は，Browser Core は通常のブラウザのように URL で指定されたサーバと直接通信してコンテンツを取得することはできない．この場合は，通信が許可されたサーバ上でプロキシサービスを動かした上で，Browser Core が直接 HTTP 通信を行ってデータを取得するのではなく，Java アプリケーションである Browser アプリケーションが，通信が許可されたサーバ上のプロキシサービスを経由して指定 URL のサーバ上のコンテンツを取得し，こ

れを Browser Core に渡すといったような手段をとる必要がある．

現在， Java 実行環境側でのアクセス制限，アプリケーション認証機能等の検討が行われているが，EBA での Browser Core モジュールもこれに準じた対応を行っていき，Browser アプリケーション及び Browser Core 全体として，Java で規定されたセキュリティ要件を満たすようにする必要がある．

次にアプリケーションのサイズについてであるが，４章の実装例で作成した各 Browser アプリケーションは，コードサイズがそれぞれ 30 行程度で実装できた．これは，ブラウザ本体の機能及び音声合成機能等は予め携帯電話側に組み込まれているためだが，日本における携帯電話用の Java アプリケーションのコンテンツサイズは現状数十 KB 程度に制限されていることを考えると，非常に大きな利点となる．

Browser Core モジュール本体も Java で記述してダウンロードするというやり方も考えられるが，全機能を備えたブラウザを全て Java で実装すると，ダウンロード可能な Java アプリケーションの制限サイズのほとんどをブラウザ本体の実装で占めてしまうことになってしまう．また独自に実装したブラウザを使用すると，あらかじめ携帯電話に組み込まれているブラウザとの実装の違いによって，コンテンツ表示に差異が発生する可能性がある．EBA のように携帯電話が備えているブラウザ機能を利用するようにすれば，このような問題は発生しない．

6．まとめ

本稿では，サービスプロバイダが，新しいサービス提供のためにブラウザ機能をダウンロードによって独自拡張できる方法として，携帯電話向けの拡張可能ブラウザアーキテクチャを提案し，この概要と実装について述べた．この方式により，ブラウザ機能を利用及び拡張することができる携帯電話用 Java アプリケーションの実行環境ができた．

残された課題としては，まずセキュリティ対策として，EBA 環境から保護されるべき各種機能及び個人情報へのアクセス制限機構の検討が挙げられる．また，携帯電話実機上に今回実装したモジュールを移植して，性能評価をすることも必要である

参考文献

1) Arnaud Le Hors, et al, “Document Object Model (DOM) Technical Reports”, W3C, http://www.w3.org/DOM/DOMTR/ (2002).

2) Tim Bray, et al, “Extensible Markup Language (XML) 1.0”, W3C, http://www.w3.org/TR/1998/REC-xml-19980210 (1998).

3) R. Riggs, et al, “Programming Wireless Devices with the Java2 Platform Micro Edition”, Addison-Wesley (2001).

4) Dave Raggett, et al, “HTML 4.01 Specification”, W3C, http://www.w3c.org/TR/html401 (1999).

5) Steven Pemberton, et al, “XHTML 1.0: The Extensible HyperText Markup Language”, W3C, http://www.w3c.org/TR/2000/REC-xhtml1-20000126/ (2000).

6) Netscape Communications, “Persistent Client State HTTP Cookies”, http://wp.netscape.com/newsref/std/cookie_spec.html (1999).

7) M. Nilsson, et al, “Composite Capabilities / Preference Profiles: Requirements and Architecture”, W3C, http://www.w3c.org/TR/CCPPra/ (2000).

8) R. Fielding, et al, “Hypertext Transfer Protocol – HTTP/1.1”, IEFT RFC2616, http://www.ietf.org/rfc/rfc2616.txt (1999).

9) Sun Microsystems, “Connected, Limited Device Configuration(JSR-30)”, Java Community Process, Java2 Platform Micro Edition (2000).