モバイルコンピューティングと
ワ イ ヤ レ ス 通 信 24－8
（２００３．３．６）

シンクライアント方式による Java 携帯電話応用システムの実現

北村 操代　　轟木 伸俊　　小島 泰三

概要

Java 携帯電話利用の産業応用システムを構築するための一実現方式を報告する。本方式の特徴は、(1) 携帯電話上の全ての表示内容や処理内容のサーバでの動的生成、(2) 端末非依存の画面記述方式による端末毎の API の隠蔽、(3) サーバとの通信なしで図形を扱うインタラクション機能の提供、である。携帯電話応用システム構築では端末にプログラムサイズ等の制約があるが、本方式により、図面を含む多種類の画面を表示でき、一つの画面記述で利用 API の異なる複数端末対応を実現できる。試作の結果、クライアント内アクションの利用により図形操作等の即時対話も可能であった。また、画面表示の応答速度は約 9 秒という実用に耐える速度が得られた。

Java Phone Applied Systems Using Thin Client Architecture

Misayo KITAMURA　　Nobutoshi TODOROKI　　Taizo KOJIMA

ABSTRACT

This paper describes an application framework for Java phones applied to industrial systems. Our approach has the following features: (i) thin client architecture where a server provides all contents including views and controls of a mobile phone client, (ii) device-independent content description which hides the differences of API sets among phones, and (iii) local interaction which enables users to handle figures on a client screen without connecting to a server. By using this architecture, many drawings can be displayed in spite of program-size limitation, application programs with a single screen specification run on various kinds of mobile phones.

1　はじめに

近年のインターネット技術の普及に伴い、PC やネットワーク型携帯電話を用いた Web 上の情報の取得が可能となった。PC では、HTML 形式を用いた、文書と画像で構成される Web 型情報をブラウザで閲覧できる。そして、携帯電話や携帯端末でこれらの情報を流用し、閲覧可能とする手法が既に多数提案され、実用化されている。例えば、XML と XSLT の組み合わせにより、コンテンツを各端末が解釈可能なフォーマットに変換できる [1]。また、その変換を動的な Web ページ生成用のスクリプト技術と XML 技術の組み合わせで行う方法もある [2]。

産業分野でも携帯電話応用システムの実用化が進められている。産業応用システムの中には、多数の図面を取り扱うこと、および、図面中の図形やその表示位置に意味があり、その図形を用いて簡単なインタラクションを行

三菱電機株式会社 先端技術総合研究所
Advanced Technology R&D Center,
Mitsubishi Electric Corporation

うことが要求されるものが多い。例えば、監視制御システムや保守支援システムでは、端末上に表示された設備図面上でいくつかの設備を指定することで、当該設備の監視値のトレンドグラフを表示する等の機能が求められる。しかし、Web 型情報が提供する、文書と画像で構成される情報では、この要求を満たすことができない。また、図面や図形を携帯電話上に表示する方法として、Mobile SVG[3] が提案されている。この Tiny プロファイルは携帯電話を対象としているが、スクリプト機能は省かれており、図形を対象としたインタラクションの記述は不可能である。インタラクション機能を実現するには、インタラクションを実行するプログラムが別途必要となる。

本報告では、Java[4] 搭載携帯電話上で図形を用いたインタラクションを実現する方式について提案する。この方式の特徴は、クライアントが軽量であること、端末依存の API を隠蔽したこと、ローカルインタラクション機能をクライアントに持たせたことである。

現在の携帯電話の Java 環境を利用する上で、二つの問題が存在する。一つは、CPU 性能、プログラムサイズ、通信速度といった性能面の制約である。CPU 性能は PC よりも著しく劣っている。プログラムサイズは 30〜100kbyte 以内に制限されており、複雑な処理を行うことはできない。また、現行の通信速度は 9.6〜144kbps であるが、産業分野では 28.8kbps の携帯電話でも動作することが求められている。

もう一つは、携帯電話会社と電話機メーカが多数存在することによる多様性の問題である。携帯電話会社ごとに Java アプリケーションで利用できるプロファイル (API 集合) が異なり、現在公開されているプロファイルには DoJa プロファイル [5] と MIDP プロファイル [6] がある。また、携帯電話会社や電話機メーカによるプロファイルの独自拡張や、携帯電話の高性能化に伴うプロファイルのバージョンアップにより、プロファイル数が増加する。様々な携帯アプリケーションを、これらの多数の携帯電話プロファイル用に製作することが、ソフトウエア管理上、改修時の修正洩れ等の点で問題となっている。

本報告の提案方式は次の三つの特徴を持つ。

(a) 軽量クライアント

携帯電話上の表示内容や処理内容を全てサーバ側で用意するアーキテクチャとする。サーバは表示内容や処理内容を動的に生成する。携帯電話上のクライアントソフトウエアは、サーバ出力通りの表示と簡単な対話処理だけを行う軽量なものとする。これにより、CPU 性能とプログラムサイズの制約下で、実現可能とする。

(b) 端末依存 API の隠蔽

プロファイルに依存しない画面内容記述方式を設計し、全ての種類の端末毎にそれぞれ実行可能なクライアントソフトウエアを一つ作成することで、端末依存の API をクライアント内に隠蔽する。これにより、多様性の問題を解決してシステムを構築できる。

(c) ローカルインタラクション機能

図形を対象にした対話操作機能や、対話操作に連動した無効化 (見え消し) 機能等を、個々のクライアント内に実現し、サーバ側から機能を指定することでインタラクション可能とする。この機能により、通信性能の制約下で、図形操作の結果をすぐに反映可能とする。

本方式を (a) に因んでシンクライアント方式と呼ぶ。

本報告の構成は次の通りである。まず第 2 節では、軽量クライアントの概要について述べる。第 3 節で、機種依存 API の隠蔽のための画面内容記述方式について説明する。第 4 節では、ローカルインタラクションを実現するためのアクション定義と、様々なローカルインタラクションについて説明する。第 5 節で、サーバの実現と試作システムについて紹介した上で、評価について述べる。

2　軽量クライアント

本方式では、軽量の携帯電話クライアントと、携帯電話上の表示内容や処理内容を動的に生成するサーバでシステムを構成する。

サーバは、表示状態の管理と最終図形への変換の機能を持つ。表示状態の管理機能は、携帯電話の表示画面を把握し、ユーザの操作から次に表示する画面を決定する。最終図形への変換機能は、決定された画面から、長方形や文字列等の描画情報や、文字列入力エリアなどのグラフィカルな要素部品に変換し、画面内容を生成する。もし表示に必要なデータがあれば、DB 等から取得した上で、描画情報やグラフィカルな要素部品に変換する。

一方、クライアントは、インターネット経由でのサーバアクセス、サーバ出力の読み込み、出力通りの表示、および対話機能を持つ。対話機能として、スクロール表示、キー押下時の次画面要求等を行う。

表示の拡大縮小やグラフの折れ線表示の座標計算処理は、サーバで行う。例えば、折れ線グラフ表示を行うには、サーバで、x 軸、y 軸、軸の目盛、折れ線等を図形の集合に変換し、クライアント上の座標値を算出する。

携帯電話 Java 環境では通信に HTTP(とHTTPS) しか使えないため、サーバ機能は Web サーバ上に実現する。表示内容や処理内容の生成には Web サーバのページ生成機能を用い、そのページ記述がアプリケーション記述となる。

以上のようにシステムを構成することで、携帯電話のプログラムサイズや CPU 性能の制約を受けずに応用システムを実現することが可能となる。

3　画面内容の記述

携帯電話の端末依存のプロファイルをクライアント内に隠蔽するため、サーバとクライアントの間で通信する画面内容を、プロファイルに依存しないものとする。すなわち、サーバ出力は全てのクライアントで同一形式とする。本節では、この画面内容の記述について説明する。

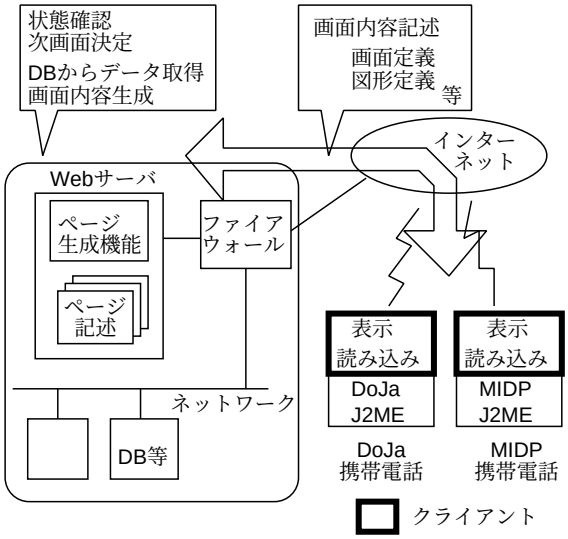
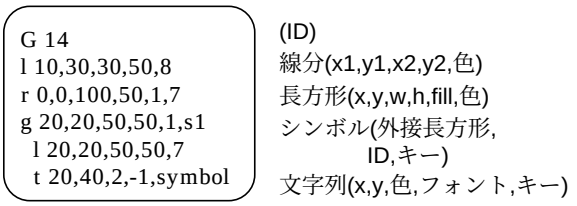


図 1: システムアーキテクチャ

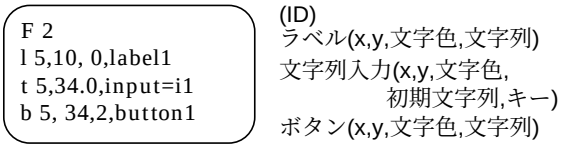
本方式では、各プロファイル中の同じ目的の機能を持つオブジェクトを共通部品として扱う。例えば、DoJa プロファイルの TextBox と MIDP プロファイルの TextField は、文字列を入力するという同じ目的を持つが、クラス名、クラス階層、当該クラスの API は、全く異なる。これらを共通の文字列入力部品として取り扱う。

クライアントは、一つのプロファイルに対して一つのミドルウエアとして用意され、共通部品による記述内容を、指定プロファイルでの処理に変換して画面表示等を行う。例えば、共通の文字列入力部品が指定されると、指定プロファイルの文字列入力オブジェクトを生成し、プロファイル提供の API を用いて初期文字列を設定する。これにより、プロファイルごとの差異の部分をクライアントプログラム内に隠蔽する。システムアーキテクチャを図 1に示す。

画面記述は、画面中に表示される共通部品定義の集合と、画面定義で構成される。共通部品には、線分、長方形、文字列等の図形部



(a) グラフィック画面



(b) フォーム画面

図 2: 画面内容記述の例

品と、文字列入力、ボタン、ラベル、リスト等の要素部品がある。共通部品定義には、個々の部品の種類に、位置、色、フォント等の属性を併記する。一方、画面定義には画面の種類とその属性が指定される。画面の種類には、要素部品を配置して表示する画面 (フォーム型画面と呼ぶ)、任意の図形を描画する画面 (グラフィック画面と呼ぶ)、および、ユーザに警告等の文字情報を通知する画面 (ダイアログ画面と呼ぶ) の 3 種類がある。画面毎に記述できる部品は限定され、フォーム型部品には要素部品を、グラフィック画面には図形部品を記述できる。画面の属性には、画面の仮想サイズ、自動更新周期を記述できる。また、グラフィック画面には、複数の図形を一括りにしたもの (シンボルと呼ぶ) を登録できる。シンボルは複数の図形部品から構成され、外接長方形、シンボル ID 等の属性を持つ。

図 2に画面内容記述の例を示す。これは実際の画面記述を説明のため単純化したものである。図中、(a) はグラフィック画面の記述、(b) はフォーム画面の記述である。(a) の記述では、画面最初の図形は線分であり、その後に線分の始点、終点、色が指定されている。この記述形式は、CPU 処理能力と通信速度を考慮し、データ量を圧縮し、かつ、部品属性とその記述順序を固定した形式とする。

このような画面内容記述を定義し、これを解釈して実行する各プロファイル用のクライアントを作成することにより、個々のプロファイルをクライアントに隠蔽できる。また、画面内容記述は同一のものを使えるため、クライアントのプロファイル数に関係なく、サーバのページ記述を一種類だけ用意すればよい。

4 アクション定義

前節で述べた画面内容記述には、画面に表示される部品の定義に加え、各部品を対象にしたユーザ操作時の処理手順を記述できる。これをアクション定義と呼ぶ。本節では、アクション定義と、その中で、サーバに問い合わせずにクライアント内で完了するアクション (ローカルインタラクションと呼ぶ) について説明する。

4.1 アクション定義の記述

アクションはイベント発生時に起動される処理である。アクションには、画面更新、次画面要求、入力データ送出等の種類がある。アクションの記述では、アクションの種類と、処理実行に必要な引数を、文字列として記述する。クライアントには、イベント発生時にアクション定義で指定された処理を行う機能を実装する。

アクション起動イベントには、キー押下、テキスト入力確定、リスト項目選択がある。またグラフィック画面では、シンボルのフォーカス中の選択キー押下や、自動更新用タイマもアクション起動イベントである。

アクション定義は、画面、要素部品、シンボルに対して記述する。画面全体の場合、自動更新時、特殊キー押下等、発生イベントが複数ある。このため、画面の属性記述内にイベントの種類を特定してアクション定義を記述する。一方、要素部品とシンボルでは、部品の種類に応じてアクション起動イベントを一つ特定できる。そこで、当該部品の属性記

述内にそのイベント発生時のアクション定義を記述する。

4.2 ローカルインタラクション

本方式では、下記のアクションをローカルインタラクションとして用意する。これにより、原則としてサーバが全ての表示内容を生成するが、一部の処理をクライアント内で完了させることにより、ユーザとの対話を円滑に行える。

画面スクロールとフォーカス移動: 画面スクロールおよびフォーカス移動は、グラフィック画面の上下左右キーのイベントに対するアクションである。シンボル記述中に上下左右の次シンボルを指定する。フォーカスがあるシンボルにその指定があれば、その先のシンボルにフォーカスを移し、それ以外の場合は画面指定のスクロール幅でスクロールする。

部品選択状態の設定: シンボルには選択/非選択いずれかの状態があり、このアクションはフォーカスされたシンボルの選択状態を交互に変更する。次画面要求アクションが呼ばれた時には、要求の引数に選択状態の部品のリストが添付される。

画面再利用表示: 一度表示した画面を、2回目からは通信せずに表示するアクションである。再利用する画面に ID を割り当てておき、本アクションの引数に、その画面の ID と URL を指定する。アクションが呼ばれると、指定の画面が未表示の場合には、指定の URL を用いて画面内容を入手し、登録する。表示済であれば、登録された画面を表示する。

部品表示属性の変更: アクション発生時に、同一画面上の表示部品の、表示値、無効化 (見え消し)、リスト (オプションメニュー、ラジオボックス等の候補文字列) の項目を変更する。なお、これらのうちプロファイルで機能が提供されないものは、クライアントが無視する。

複合アクション: 上で述べたアクションを組み合わせて実行する、逐次実行アクションと条件分岐アクションを用意する。逐次実行アクションは複数のアクションを順に実行する。条件分岐アクションは、条件により二者択一でアクションを実行する。判定条件には、値の文字列の一致、あるいは、リストの選択値の一致だけを許す。

5 試作と評価

筆者らは本方式を用いて、携帯電話応用のプラント監視制御システムを試作した。従来の監視制御システムの端末上には、監視対象の機器の稼働状態やセンサの計測値を図面上に色や文字列で表示したり、それらデータの時間経過に従った推移をトレンドグラフで表示する。本試作では、これに準じる機能を作成した。

この試作システムを用いて、画面内容記述によるプロファイルの隠蔽、ローカルインタラクションの操作性の検証、および応答速度の点で評価を行った。クライアントには DoJa 2.0 用と MIDP 1.0 用を用意した。

本節では、まずサーバを実現する上で実装したサーバ用ライブラリを紹介し、次に、試作システムを用いた評価について述べる。

5.1 サーバの実現

サーバの実現のため、画面内容記述の生成機能に JSP(JavaServer Pages) を採用し、JSP のページ記述作成用の Java ライブラリを用意した。

まず、ページ記述作業の容易化のため、表示ライブラリを備えた。各画面、画面上に配置される図形部品、要素部品、シンボルに対応するクラスを、それぞれ表示部品として用意した。各表示部品はシリアライズ機能を備え、画面内容記述を文字列として出力する。これにより、画面内容記述をデータ量を圧縮した視認性の悪い形式としたことに起因する、記述作成時の誤記や障害発見の困難さの問題を回避する。

```
MSpec sp = MSpec.getSpec("D504i");
MCanvas c = new MCanvas(sp);
int h = sp.getFont().getHeight();
MRect r = new MRect(5,5,50,h+3);
r.setColor(MColor.RED);
c.add(r);
c.print();
```

図 3: 表示ライブラリの使用例

また、端末の仕様情報を提供する仕様部品を表示ライブラリに追加した。画面サイズは機種によって異なり、各種端末の仕様にあわせた、ページ記述内でのレイアウトの微調整を許すためである。仕様部品は、各機種のフォントサイズ、画面サイズ等の仕様を、別途作成した定義ファイルから読み込み、提供する。

図 3に表示ライブラリを用いた例を示す。高さがフォントサイズより 3 ピクセル高い長方形を、グラフィック画面上に描画している。

また、試作にあたり、監視制御分野向けのデータ収集機能を用意した。データ収集要求の接続先は、DiaSynapse[7] 等の監視データサーバであり、接続先とその接続方法の差異をこの機能内に隠蔽している。

5.2 評価

本方式で試作したアプリケーションの DoJa 2.0 プロファイルでの画面例を図 4に示す。図中、矢印に添えられた括弧内の数字は動作順を示す。また、太線の画面遷移はサーバへの要求が必要であることを、細線のものはローカルインタラクションを示す。(C) と (D) の画面は一つの画面内容記述から表示され、(C) の画面と (D) の画面 4 枚から構成される。また、(F) の画面は、246x192(ピクセル) の大きさの画像 (表示面積の約 2.5 倍の大きさ) とその上に置かれたシンボルから構成されている。

端末プロファイルの隠蔽

図 5は図 4中の画面 (B)、(C) と同じ画面内容記述から生成される MIDP 1.0 プロファイルの画面である。MIDP 1.0 プロファイルではボタン部品はソフトキーのコマンドに対応し、二つめ以降のボタン部品は右下の “menu” から起動できる。これらの画面は同一の画面内容記述から構築できた。

ローカルインタラクション

本アプリケーションでのローカルインタラクションの動作を、図 4で説明する。

図中、(C) と (D) の画面、および (D) の 4 画面は、上下左右キーの押下によって即時切替えられる (操作 (4)(5)(6))。

操作 (7)、(8)、(14)、(15) では、画面再利用表示アクションが実行される。表示画面が既に一度表示されているため、Web サーバとの通信を行わずに即時表示される。この後、再度操作 (2) を行った際にも (B) を再利用できる。

画面 (E) では、上のプルダウンメニューに、項目選択時に下のプルダウンメニューに表示項目リストを設定する部品表示属性設定アクションが設定されている (操作 (10))。上のプルダウンメニューで大分類を選び、下のプルダウンメニューで最終的な項目選択を行う。これにより、30 項目程度の選択肢の中からの選択操作を、サーバから画面を取得せず、かつ、少ない上下スクロール操作回数で行える。

また、図中 (F) の画面には、シンボルの選択状態設定アクションが指定されている。いくつかのシンボルを選択状態にした (操作 (12)) 後にキー操作を行う (操作 (13)) と、それら選択項目の詳細データが表示される (画面 (G))。

図 4全体の操作中、サーバからの画面入手は、ローカルインタラクション機能無しでは 20 回必要だったが、6 回だけであった。これらのことから、本方式によれば図面上の図形を取り扱うアプリケーションを実現でき、ローカルインタラクションも、サーバからの画面要求の待ち時間を減らす上で有効であることが確認できた。

応答時間

サーバへの画面内容問い合わせ時の応答時間を測定した。図 4中、操作 (2)、(3)、(11) の場合を測定対象とした。

測定に用いたシステムの構成を図 6に示す。

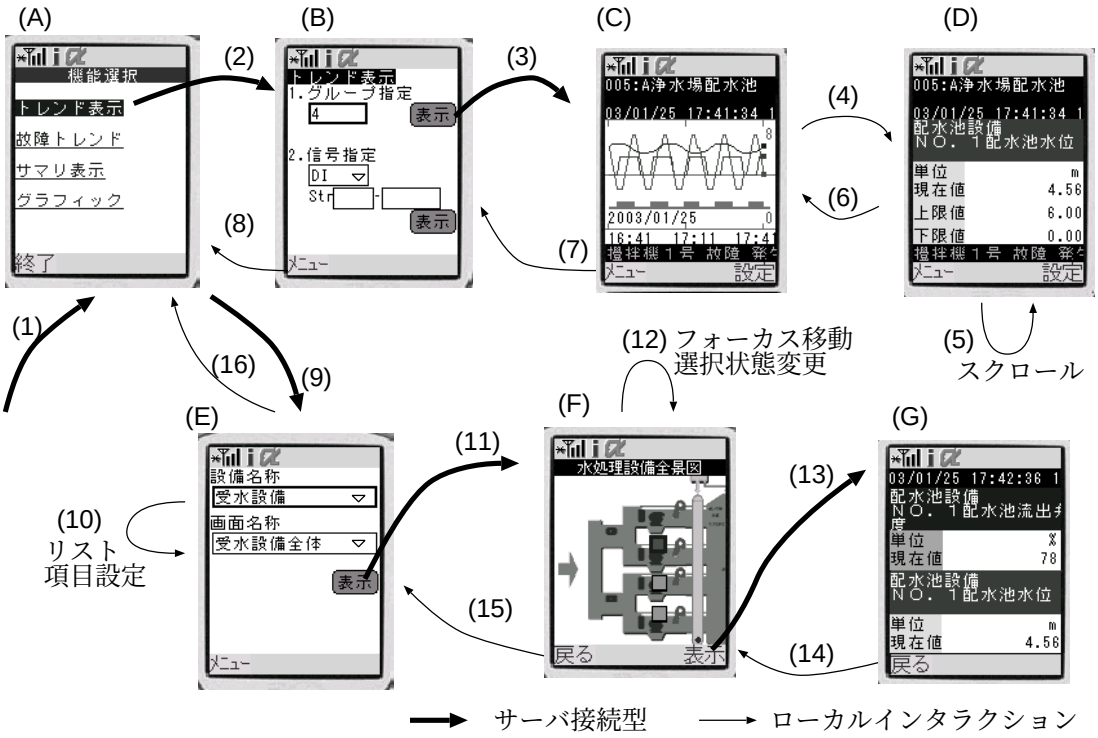
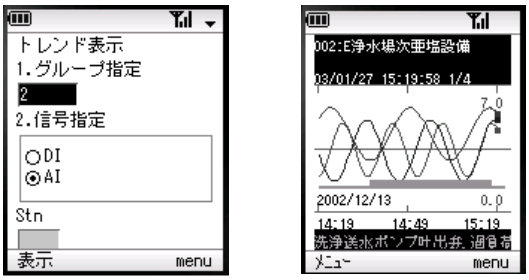


図 4: アプリケーション画面例 (DoJa 2.0)



(a) フォーム画面　　(b) グラフィック画面

図 5: 画面例 (MIDP 1.0)

サーバ機は PC であり、Web サーバとプラントデータ取得用模擬環境が搭載されている。携帯電話には D504i(以下、実機) を用い、HTTP の場合と HTTPS の場合の応答時間を測定した。比較実験として、携帯電話に PC 上のエミュレータ [8] を用いた場合も測定した。エミュレータ環境のモデムの通信速度は、予備実験で実機の実効通信速度が 11kbps であったため、その速度に近い 14.4kbps とした。この

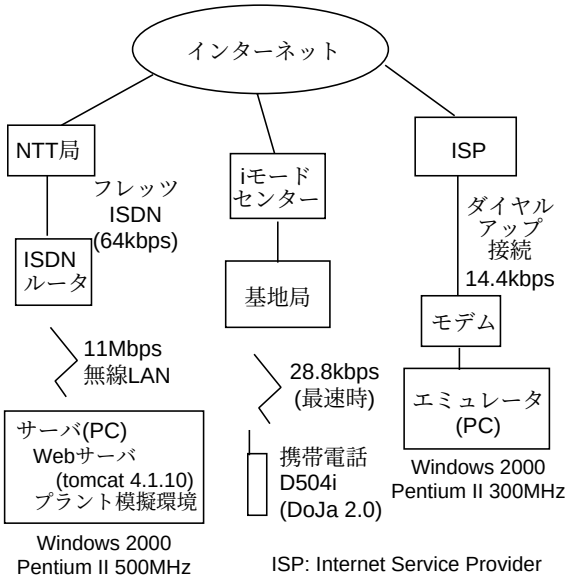


図 6: 性能測定用システムのシステム構成

環境でストップウォッチで応答時間を 20 回測定し、平均を求めた。その測定結果を表 1に

操作	部品数 通信量		平均応答時間 (秒) HTTP	 HTTPS
(2)	12	D504i	3.41	9.82
	469	PC	1.11	1.48
(3)	91	D504i	7.87	12.54
	3258	PC	2.75	3.92
(11)	25	D504i	8.36	19.45
	912	PC	4.85	5.05
(+画像 3287)				

表 1: 各画面表示に要した通信量と応答時間。部品数はサーバが出力した部品数 (単位：個)、通信量はサーバが出力した文字列のバイト数 (単位：バイト)。PC はエミュレータを示す。

示す。

表 1より、実機で HTTP の場合には応答時間は概ね 9 秒以内に収まっている。この応答速度は、サーバの情報を表示する携帯電話応用システムの端末としては、実用に耐える速度である。

一方、HTTPS では実機の最悪の場合約 20 秒の応答速度であり、これは実システムとしての許容範囲を超えている。別途サーバ上で TCP パケットモニタリングを行ったところ、再接続認証通信のための所要時間が約 4 秒増加している。操作 (11) の HTTPS での表示で所要時間が特に長いのは、まず画面記述を入手し、次に Web サーバに再接続して画像を取得するため、認証が 2 回行われることによる。

6 おわりに

本報告では、携帯電話上の表示内容や処理内容を全てサーバ側で用意するシンクライアント方式を提案した。本方式の特徴は、(1) 携帯電話上の全ての表示内容や処理内容のサーバでの動的生成、(2) 端末非依存の画面記述方式によるプロファイルの隠蔽、(3) サーバとの通信なしに即時対話可能なローカルインタラクションの提供、である。

また本方式を適用したシステムを試作し、評価を行った。その結果、同一の画面内容記述から異なるプロファイル上のシステムを構築することができた。また、図面上の図形を取り扱うアプリケーションを実現でき、ローカルインタラクションの有効性も確認できた。応答速度の調査の結果、HTTP では概ね 9 秒以内であり、サーバの情報を表示するシステムの端末としては、実用に耐える速度が得られた。

本報告では携帯電話のプロファイルのみを取り扱ったが、今後、PDA の特徴を生かした機能への対応について検討を進める予定である。

参考文献

[1] W3C: “XSL Transformations(XSLT) Version 1.0”, http://www.w3c.org/TR/xslt (1999)

[2] 田口雅人、小林宏至、藤岡秀樹: “携帯電話向けコンテンツ変換システムの開発”, 情報処理学会第 64 回全国大会, Vol. 3, pp. 3–543 – 544, 2002

[3] W3C: “Mobile SVG Profiles: SVG Tiny and SVG Basic”, http://www.w3c.org/TR/SVGMobile/ (2003)

[4] http://java.sun.com/

[5] NTT ドコモ: “i アプリコンテンツの作成について”, http://www.nttdocomo.co.jp/p_s/imode/java/index.html

[6] Sun Microsystems: “ Mobile Information Device Profile (MIDP) 1.0 Specification”, http://jcp.org/aboutJava/communityprocess/final/jsr037/ (2000)

[7] 石原鑑、大崎雅代、高田秀志: “インターネット応用監視制御フレームワーク DiaSynapse / JAXSON”, 三菱電機技報, Vol.76 No.9, pp.44 – 48, 2002

[8] NTT ドコモ: “504i 向け i アプリ開発ツール”, http://www.nttdocomo.co.jp/p_s/imode/java/tool_504i.html