平成１４年度　卒業研究

Servlet を用いた携帯端末用画像情報提供システム

島根大学総合理工学部数理情報システム学科

ｓ９９４７８－ｋ　堀　隆志

目次

１．序文

現在携帯電話の加入台数は７０００万台ともいわれている。その用途は「電子メール」が突出しており「Web サイトのコンテンツ閲覧」「PIM（スケジュールや住所録などの個人情報管理）」、がそれに続いている、携帯電話＝通話、PDA＝PIM という枠は崩れ、モバイル機器は情報システムのクライアント端末やソフトウェア開発のプラットフォームとして、その存在意義を拡大している。

本研究ではモバイルシステムでの用途のトップに上げられる「コンテンツ・情報配信」システムを開発し、現在のモバイルシステムの技術的状況を検証する。

２．Java2 Platform,Micro Edition とは

２.１　Java2、Micro Edition

（「IBM　Java　technology」より引用）

１９９０年６月に Sun Microsystems によって発表されて以来、Java2、Micro Edition（以下Ｊ２ME） は、Java 開発者のさまざまなニーズに応えるための幅広い製品のひとつとして提供されてきた。Sun は、Java 2 Platform における Java テクノロジー・アーキテクチャーを再構成し、それを 3 つのエディションに分類した。Standard Edition (J2SE) は、デスクトップ開発およびローエンドのビジネス・アプリケーションのための実用的なソリューションを提供し、また、Enterprise Edition (J2EE) は、エンタープライズ環境のアプリケーション開発者を対象としたもの。そして、Micro Edition は、（＊1）コンシューマ機器および組み込み機器を扱う開発者に適している。この J2ME の普及は、サーバーからデスクトップ、また携帯機器に至るまで、さまざまな機器をひとつの言語およびテクノロジーによって包括する。これらの機器のためのアプリケーションはさまざまに異なるが、Java テクノロジーはその相違を埋める役割を果たし、ひとつの分野を専門とする開発者が、いろいろな機器やアプリケーションにその技術を生かすことができるようになる。また J2ME の特徴は「1 つの」仕様ではなく、関係する仕様の集まりであり、そのそれぞれの仕様によって、リソースに制限のある機器に対する Java テクノロジーが定義されるという点である。

（＊1）コンシューマ機器：中間生産者向けの機器ではなく、末端の、そして大多数の末端消費者をターゲットとする機器のこと

２.２　携帯端末と Java

（「IBM　Java　technology」より引用）

J2ME を使用することができる機器には、PDA、携帯電話、ポケットベル、その他の組み込み機器など、さまざまなものがあります。これらのすべての機器に対する最適な (最適に近い) テクノロジーを 1 つに限定することは明らかに困難である。それは、これらの機器のプロセッサー・パワー、メモリー、持続ストレージ、ユーザー・インターフェースの違いがきわめて大きいためである。

この問題を解決するために、Sun は J2ME に適した機器を分類し、それをさらに細かなセクションに分割した。最初の分類において Sun は、その機器の用途には関係なく、プロセッサー・パワー、メモリー、ストレージ機能を基準とした 2 つの大きなカテゴリーに分類した。次に同社は、必要最小限の Java 言語機能を提供することを目的とし、各々のカテゴリーに属する機器の制約に対応できる Java 言語バージョンを作成した。さらに、これら 2 つのカテゴリーに属する機器に対してその役割ごとのクラス分けを行った。例えば、「携帯電話」に関しては、その製造元にかかわらずすべて 1 つのクラスに分類した。また、Sun は、（＊2）Java Community Process（JCP）のパートナーの支援によって、それぞれの縦方向の分類に対して機能面での細目を追加した。

（＊2）Java Community Process：Java プラットフォームの発展を目的としたオープンな組織。JCP は、世界中の Java 開発者およびライセンシによって構成され、Java テクノロジ仕様、リファレンス実装、およびテクノロジ互換性キットを開発および改訂する特権を持ちます。Java テクノロジおよび JCP は、最初、Sun Microsystems によって開発され、創設された。JCP は、1995 年に Sun が非公式のプロセスとして採用して以来、発展を遂げ、今では Java コミュニティに参加するさまざまな組織の代表が監督する公式のプロセスとなっている。

第 1 段階の分類では、現在の J2ME のコンフィギュレーションである Connected Device Configuration (以下 CDC) と Connected, Limited Device Configuration (以下 CLDC) の 2 つのコンフィギュレーションに分類が行われた。各コンフィギュレーションは、Java 仮想マシン (JVM) および特定の機器のためのランタイム環境を提供するために最小限必要とされるクラス・ライブラリーおよび API によって構成され、そのコンフィギュレーションが対象とする機器のリソースの制約に合った、最小限必要な Java 言語の共通サブセット

を持つ。

1 つのコンフィギュレーション内でも、ユーザー・インターフェース、機能、使用法は多岐に渡るため、通常のコンフィギュレーションでは、ユーザー・インターフェースのツールキットや持続ストレージ API のような重要な部分は定義されていない。このような機能の定義は、「プロファイル」によって定義される。

J2ME プロファイルは、ポケットベルや携帯電話などといった特定の機器クラスに対応する業界グループによって指定された Java API によって構成される。各プロファイルは、コンフィギュレーションにおける最小限必要な Java 言語の共通サブセットに追加して作成され、そのコンフィギュレーションを補完します。現在、CDC のための Foundation プロファイルと CLDC のための Mobile Information Device Profile (MIDP) の 2 つのプロファイルがある。

J2ME の仕様は 1 つではなく、いくつかの仕様の集合であり、それぞれの仕様が特定の要件に対応している。

図 1 は、J2ME のコンフィギュレーション / プロファイルと J2SE、J2EE との関係を示している。

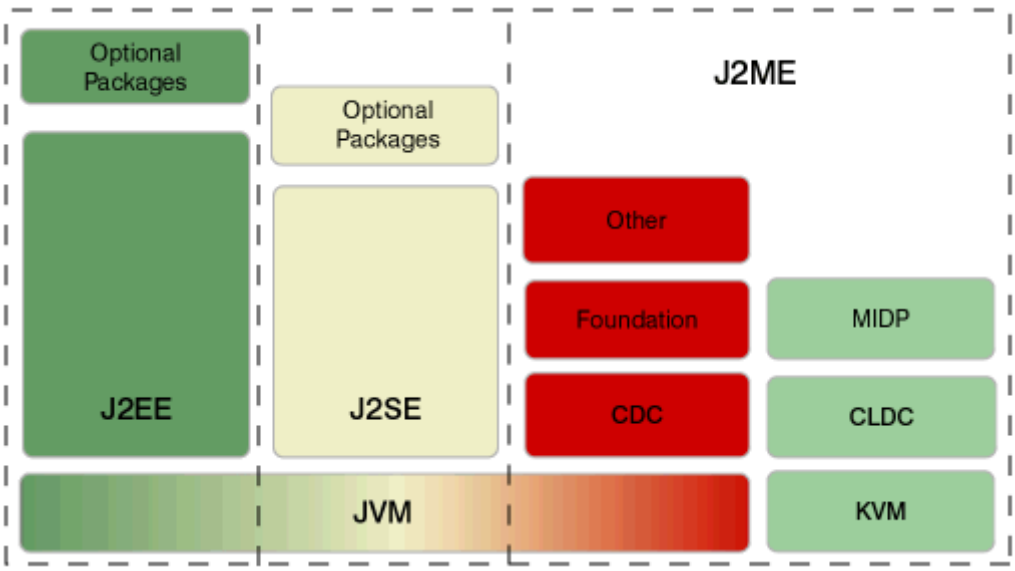


図 1. J2EE、J2SE、J2ME の関係

また CLDC 仕様に用意された JavaVM を「KVM（K Virtual Machine)」と呼ぶ。ここで「K」という名称がついているのは、K オーダー（64KB～128KB 程度）のメモリで動作するという意味である。

2.3 CLDC と CDC

（「IBM Java technology」より引用）

CLDC：512 KB 以下の機器に対応するアプリケーションのこと。

CLDC は、512 KB 以下のメモリー、電力の制限 (電池など)、制限されたまたは断続的なネットワーク接続、単一の (または無い) ユーザー・インターフェースを持つ機器を対象としており、携帯電話、ポケットベル、PDA、またそれに類似した機器のためのアプリケーション開発に最も適している。

厳しい制約に合わせるために、CLDC は J2SE の多くの機能を捨てなければなりませんでした。その結果、CLDC には、標準 Java 仕様のうちの 3 つ (java.lang、java.util、java.io) と CLDC 用の 1 つの仕様 (javax.microedition) の 4 つのパッケージのみが残された。

また、3 つの標準パッケージの内容も縮小されている。java.util パッケージには、47 のクラスと J2SE のインターフェースがありますが、CLDC では、10 クラスに縮小された。この 10 クラスによって、アプリケーション構築に最小限必要とされる機能のみが提供され、他のクラスの機能は、後に説明する MIDP によって提供される。

表 1 には、CLDC の各パッケージのクラス数 / インターフェース数が示されている。

パッケージ	説明	クラス／インターフェース
java.io	システム入出力	18
java.lang	Java プログラミング言語の基礎となるクラス	38
java.util	Collections、日付 / 時刻のサポート、さまざまなユーティリティー・クラス	10
javax.microedition	一般接続	10

表 1. CLDC の各パッケージのクラス数 / インターフェース数

CLDC には、ほとんどのアプリケーションに必要とされるユーザー・インターフェースのサポートが欠如している。このように、CLDC は完全なソリューションを目指したものではなく、プロファイルによって機能や特定の製品クラスを追加するための共通のベース、またスタート・ポイントとして設計されている。そのようなプロファイルの 1 つ

に MIDP がある（MIDP については３.１.２章で記述)。

機能	内容
浮動小数点	小数点演算
Java Native Interface (JNI)	ネイティブコード呼び出し
ユーザー定義のクラスローダ	独自にクラスファイルを呼び出して実行する機能
リフレクション	VM 内のクラス、オブジェクト、メソッド、フィールド、スレッド、実行スタック、その他の実行時構造体の数および内容を検査する機能
スレッドグループとデーモンスレッド	スレッドのグループ管理機能
ファイナライズ	Object.finalize()メソッド
弱参照	ガベージコレクタとの制限つきの対話をサポートする機能

表２.削除されている機能

CLDC 対象外の機器に対応する CDC

現在定義されている 2 つの J2ME コンフィギュレーションのうちの大きいほうは CDC であり、TV セット・トップ・ボックス、エンターテインメント・システム、カー・ナビゲーション・システム、その他の機器に対応するアプリケーション構築に最も適している。

リソースの制約という点では、CLDC が対象とする機器の上限から上を CDC は対象としており、2MB 以上のメモリー、JavaVM と Java プログラミング言語の完全なインプリメンテーションをサポートできる機器を対象としている。CDC は一般的な Java 仕様により近いものであると言える。

CLDC 準拠の仮想マシンは、標準 JavaVM 機能のサブセットに対してのみのサポートが必要とされるが、CDC 準拠の JavaVM は、標準 JavaVM との機能的な互換性が要求される。すなわち、ネイティブ・メソッドの呼び出しのサポートが必要な場合には、CDC JavaVM (または CVM) は JNI (Java Native Interface) 1.1 への準拠が必要となり、また、デバッグのサポートには、JVMDI (Java Virtual Machine Debugging Interface) への準拠、また、プロファイルのサポートには、JVMPI (Java Virtual Machine Profiling Interface) への準拠が必要である。

クラス・ライブラリー・レベルにおいては、CDC は、完全準拠の Java2 仮想マシンのサポートに必要とされる API の最小限のセットを提供する。この API セットは、ファイル

I/O、ネットワーキング、きめ細かなセキュリティー、オブジェクトのシリアル化などとともに、CLDC に定義された API もすべて含んでいる。表 3 には、CDC 仕様のパッケージとそれらの説明、およびそのパッケージに含まれるクラスとインターフェースの数が示されている

パッケージ	説明	クラス / インターフェース
java.io	システム入出力	62
java.lang	Java プログラミング言語の基礎となるクラス	77
java.lang.ref	特別参照クラス	5
java.lang.reflect	レフレクション・サポート	12
java.math	演算サポート	1
java.net	ネットワーク・クラス / ツール	23
java.security	セキュリティー・サポート	36
java.security.cert	認証サポート	4
java.text	テキスト処理クラス	13
java.util	Collections、日付 / 時刻のサポート、さまざまなユーティリティー・クラス	47
java.util.jar	Jar ファイル・サポート	7
java.util.zip	Zip ファイル・サポート	9
javax.microedition	一般接続	10

表 3. CDC 仕様の各パッケージのクラス数 / インターフェース数

表 3 に、java.awt パッケージのクラスおよびインターフェースがありません。CLDC と同様、CDC もいかなるタイプのユーザー・インターフェースもサポートしていない。これは、CLDC の場合と同じく、機器によってそのユーザー・インターフェースが多岐にわたるためである。ユーザー・インターフェースをサポートするためには、CDC に対する適切なプロファイルの追加が必要である。

CDC に使用する JavaVM である CVM は、CDC API のサポート以上の多くの興味深い機能を備えている。CVM は、携帯性に優れ、ROM 化可能なクラスをサポートし、また高速スレッドの同期操作を可能とし、ネイティブ・スレッドをサポートする。すなわち、

CVM は、組み込みアプリケーションに多く見られるタイプのオペレーティング・システムをサポートするように設計されている。

２.４ Java VM

「http://feeling.zive.net/javapages/touch/jvmandsoft.html」によれば、 Java で書かれたプログラムの実行概念は他の実行環境と大きく異なっており、Java Vartial Machine（以下 JavaVM）と呼ばれる仮想コンピュータ上で実行される。JavaVM は Java のプログラムから見ると、仮想のコンピュータであり。JavaVM は外界、ようするにホストとなるオペレーティングシステムとやり取りができる。JavaVM はユーザが使っているオペレーティングシステムで実行され、Java で書かれたプログラムはその上で実行されます。概念的にはエミュレータと非常に似ている。

この実行概念が意味をしていることは JavaVM があればどこでもプログラムを実行できる、ということである。開発者はたった一度のコーディングで済むということは、「Java の JavaVM のバージョンに注意しながら、仕様通りのプログラムを組むことで、さまざまなプラットフォームで動作するプログラムを作ることができる」ということである。JavaVM は沢山のプラットフォームで動作する Java で書かれたプログラムのための実行環境である。

Java のプログラムの実行について通常、ソフトウェアの実行ファイルはバイナリファイルと呼ばれ、その CPU、OS でしか実行できないようにコンパイルという過程を経ている。Java の実行ファイルはクラスファイルと呼ばれている。

コンパイルはプログラムを作成する上で欠かせない作業であり、コンパイルすることでそのプログラムは初めて実行できるようなファイルになる。しかし、その時点でソフトウェアは動作する環境が決まってしまう。ここが静的処理系と呼ばれる実行環境の問題点である。Java のプログラムもコンパイルという過程を踏むが、Java の場合は特殊である。

Java のコンパイルをするとクラスファイルを書き出す。書き出されたクラスファイルは CPU 固有のマシン語ではないため、どのプラットフォームでも動作することを意味している。JavaVM はコンパイラによって書き出された、クラスファイルを実行時にバイトコードにコンパイルをして実行する。

また JavaVM にはクラスの動的検索とロード履歴からの予測がある、JavaVM はリクエストされたクラスを動的に検索する、メソッドも同様。リクエストがある度に JavaVM は検索をしなければない。しかし、毎回クラスの検索をしていては動作のスピードに影響を与えかねる。そのために JavaVM はクラス、メソッドのロード履歴を作成し、次にロードされるクラスを予測する。そして、それらのクラスを JIT コンパイラでネイティブコード

にコンパイルをして、キャッシュに溜めておく。リクエストがあればクラスのロード履歴から、クラスを検索する。ヒットすればネイティブコードでコンパイルされたコードを実行する。JIT コンパイルは Just In Time コンパイルの略で高速化コンパイル技術のことをいい、クラスファイルをコンパイルすると通常はバイトコードに変換するが、ネイティブコードに直接変換をする技術である。しかし、クラスのロード履歴からリクエストされたクラスがなかった場合は JavaVM はクラスを再び動的に検索する。そして、今度は JIT コンパイラでコンパイルをせずにバイトコードのまま実行をする。

３．キャリア（ NTT DOCOMO、KDDI、J-PHONE ）によるシステムの相違

３.１　Java プロファイル

「明解 携帯電話 Java」（橋本、２００１）より抜粋

3.1.1 DoJa

NTTDocomo（以下 Docomo）のｉモード対応 Java は、J2ME の CLDC をコンフィグレーションとして採用しているが、プロファイルは Docomo 独自のプロファイルを作成、採用している。このプロファイルを DoJa（Docomo Java）と呼ぶ。以下で DoJa の特徴を示す。

このほか Docomo の携帯端末には各メーカーが実装した独自機能があり、メーカー独自のライブラリを実装している。つまり Docomo の Java は

CLDC＋DoJa（ｉモード対応 Java）＋端末メーカー独自のライブラリ

という形態で作動している。

3.1.2 MIDP

MIDP は、Sun、Motorola,Nokia,を中心とした MIDPEG（Mobile Information Profile Expert Group）によって策定された CLDC 上で使用するプロファイルである。

J-PHONE、KDDI はこちらを採用している。

J-PHONE の形態

CLDC＋MIDP＋JSCL（J-PHONE Specific Class Liblary）

KDDI の形態

CLDC＋MIDP＋KDDI 拡張ライブラリ

MIDP は、携帯電話、ポケットベル、簡単な PDA などの機器のアプリケーション構築に際して、CLDC に必要な機能を追加する。MIDP の機能には、タイマー、簡単な持続ストレージ、HTTP によるネットワーキング、また、ユーザー・インターフェースに対するサポートなどがある。

標準 Java 仕様にはあって CLDC に欠如している 3 つのクラスが、java.lang パッケージ

と java.util パッケージに再び追加され、MIDP によるタイマーのサポートが可能となる。以下がその 3 つのクラス。

java.util.Timer

java.util.TimerTask

java.lang.IllegalStateException

MIDP のその他の機能はすべて、J2ME に特別に用意された 4 つのパッケージに含まれている。表４には、4 つのパッケージとそれらの説明、およびそのパッケージに含まれるクラス数 / インターフェース数が示されている。

パッケージ	説明	クラス / インターフェース
javax.microedition.rms	持続レコード・ストレージ	10
javax.microedition.midlet	MIDlet とその環境とのインターフェース	2
javax.microedition.io	HTTP プロトコルのサポート	1
javax.microedition.lcdui	ユーザー・インターフェース・ツールキット	24

表 2. MIDP を構成する 4 つの javax.microedition パッケージ

MIDP は世界の携帯電話キャリアが採用するプロファイルであり、将来的に NTT Docomo は自社仕様の DoJa ではなく世界標準の MIDP に移行するという話があるので、卒業研究は MIDP を使用している。

３.２携帯電話における Web コンテンツ

3.2.1 C-HTML

C-HTML は ACCESS／NEC／富士通／松下電器／三菱電機／ソニーの 6 社が 1998 年に規格を設定し、「W3C」（WWW コンソーシアム）に提案してモバイル端末向けの記述言語のひとつとして認可された。C-HTML はパソコンとは違い、表示画面やメモリ容量に制限のある携帯電話や PDA で Web ブラウジングを実現する言語として開発されている。そのため、HTML が規定する各種タグのうち、フレームやテーブル、各種のフォント（文字の種類やサイズの指定)、GIF 以外の画像に関するタグを除き簡略化している。
NTT ドコモの i モードはこの C－HTML をベースとしているが、さらに NTT ドコモ独自の拡張を行なっているので C-HTML と完全に互換しているわけではない。NTT ドコモでは「i モード対応 HTML」と表記し、C-HTML から携帯電話に必要な機能を絞込み、独自の規格展開を行っている。また、携帯電話の機能向上に合わせるカタチで規格自体も拡張されている。

3.2.2 HDML

WAP に準拠した携帯電話（KDDI 用携帯電話など）や PDA のブラウザを通じて閲覧できる、コンテンツやホームページを制作するためのタグ形式の記述言語。オープンウェーブ社が開発したもので、WAP コンテンツの記述言語「WML」(Wireless Markup Language）の原形。HDML はディスプレイサイズが小さい携帯電話で表示されることを前提にしているため、表示される部分を 1 枚のカードに見立てたカード形式が基本となっている。さらに、このカードを複数まとめたものがデッキと呼ばれ、サーバーと携帯電話端末がデータやり取りする際には、このデッキがひとつの単位となる。携帯電話端末では、サーバーから受信したデッキ内容をカード単位で表示している。i モードがページごとにサーバーにアクセスするのとは異なり、複数ページを一括ダウンロードできるためスムーズにコンテンツを閲覧することが出来る。

3.2.3 MML

Mobile Markup Language の略。携帯端末向けに考案されたマークアップ言語の 1 つ。

慶應義塾大学の MOBiDY プロジェクトと J-Phone によって開発された。J-Sky サービスのコンテンツ表示に使われている。HTML をベースにタグの文字列を短縮するなどしてデータ量を少なく抑えられるようにしている。J-PHONE 以外の各社の携帯電話が MML に対応していないため、J-PHONE のネットワークには，HTML を MML に翻訳するゲートウェイがあるため、それを通してほとんどの i モード用サイトの閲覧が可能である。なお MML の記述に関して J-PHONE から一般向けに資料が公開されていない。

また C-HTML,HDML,MML について詳しくは

「W３C」http://www.w3.org/TR/1998/NOTE-compactHTML-19980209/

「Docomo　Net」http://www.nttdocomo.co.jp/p_s/imode/

「EZweb ホームページを作ろう」http://www.au.kddi.com/ezfactory/

「J-PHOME 技術情報」http://www.dp.j-phone.com/jsky/tech.html

に掲載されているので詳しくはこちらを参照。

４.画像情報提供システム

４.１　システムの設計

4.1.1 要求仕様

システムが利用可能な携帯電話としては、世界標準である MIDP を使用しており、技術資料が提供されている KDDI 用携帯電話での使用が可能なシステムを作成する。また個人情報管理にはセキュリティの面でデータベースを使用する。

システムの自体は、最初にログイン管理システムを設置し、パスワードと氏名による検査を行う。

料金のシステムとしては、携帯電話の画面上からリンクして画像が画面上に表示された時に一定金額をデータベースの個人情報の欄に加算する、なおその画像を閲覧しているのであれば履歴表に入力し一定期間は同じ画像であれば料金の加算はしない。

また新規の登録は携帯電話の画面上から行い、同時にデータベースに登録、登録後即座にシステムが利用できるようにする。

4.1.2 製品仕様

HDML、MIDP、を使うことで KDDI 用携帯電話でのシステムの利用が可能、その反面 NTTDocomo、J-PHONE 用携帯電話での実行は不可能となっている。

データベース利用と Web 上での登録及び即実行の２つを可能とするため、このシステムでは JavaServlet を利用している、データベースは HitachiHiRDB を使用し、Servlet は Tomcat3.3.1、Windows2000Server での実行確認済み、なおログイン、料金加算、閲覧履歴、新規顧客加入の等の要求された機能を備える。

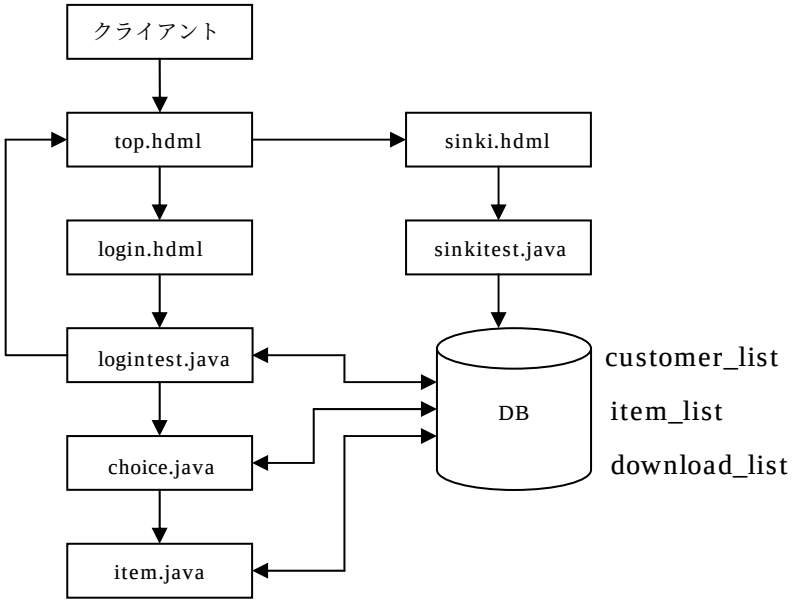
クライアント用機種	KDDI 用携帯電話 Web 対応版
システム搭載サーバ OS	Windows2000Server
Servlet　Container	Jakarta-Tomcat3.3.1
システム実行用シミュレータ	＊3　UP.SDK.3.3.1　UP.Simulator
Java	j2sdk1.4.0_01
データベース	HITACHIHiRDB

＊３　UP.SDK.3.3.1 UP.Simulator

米 Openwave Systems の「UP.Browser」は HDML に対応したブラウザである。なおこの UP.Browser は携帯電話に搭載されているので、コンテンツを制作する PC 上で動作するわけではない。そこで、Openwave Systems は HDML 上で動作するコンテンツを作成する人々向けに、Web 上で開発者向けプログラム（Openwave ディベロッパープログラム）により、UP.Browser とほぼ同様の機能が再現できる「UP.Simulator」を提供している。また、日本語完全対応となっている。これにより PC 上で HDML などの携帯端末用 Web ページの閲覧、テストが行うことが出来る。

４.２　システムの内容

4.2.1 概要



top.hdml: トップページ

新規利用者はsinki.hdmlへ登録済者はlogin.hdmlへ移動するよう表示。

sinki.hdml: 個人情報入力

名前、パスワード、ｅメールアドレス、電話番号、携帯電話番号、を入力するよう表示する。その後そのパラメータを sinkitest.java へ送る。

sinkitest.java:　個人情報登録

sinki.hdml から送られたパラメタをデータベースの customer_list に入力、その後 top.hdml へ戻るよう servlet で表示。

login.hdml:　ログイン画面

登録済者はここで名前とパスワードを入力するよう表示しその２つのパラメータを logintest.java へ送る。

logintest.java: パスワード確認

login.hdml から送られた２つのパラメータをデータベースから検索、一致することを確認した後、choice.java へのリンクを表示。
同時に送られてきた名前、パスワードの２つのパラメータも choice.java へ送る。

choice.java: 画像選択

画像選択画面を表示。選んだ画像のＩＤと logintest.java から送られてきたパラメータの 3 つのパラメータを item.java へ移動するときに送る。

item.java: 画像表示

送られてきたパラメータの画像ＩＤに対応する画像を表示、また名前、パスワードの一致する顧客が過去その画像を閲覧していないか download_list で確認後 download_list に画像ＩＤ、顧客ＩＤ、日付を入力し、customer_list の total_price を更新する。

・データベース内のテーブル

テーブル１：customer_list（会員情報）：アクセス権限　”root”

primarykey　→　customer_id、索引無

列名	型	内容
customer_id	char(6)	会員ＩＤ
password	char(6)	パスワード
customer_name	char(10)	名前
e-mail	char(50)	ｅメールアドレス
phone_no	char(15)	電話番号
celllar_no	char(15)	携帯電話番号
total_price	int(10)	利用料金
countory_id	char(6)	国 ID

テーブル２：item_list（商品情報）：アクセス権限 “root” primarykey → item_id、索引無

列名	型	内容
item_id	char(6)	商品ＩＤ
item_name	char(10)	商品名

テーブル３： download_list（商品利用履歴）アクセス権限：”root” primarykey → download_id、索引無

列名	型	内容
download_id	char(6)	履歴 ID
customer_id	char(6)	会員ＩＤ
item_id	char(6)	商品ＩＤ
the_date	date	日時

プログラムの内容及びデータベース内のテーブルの概要は４.２章：システムの内容の４．２．１項：概要で述べているのでそちらを参照

4.2.2 Servlet と HDML

Servlet の特徴は、Web アプリケーションでの使用を想定し、Web サーバと連携してクライアント（今回は携帯端末なので以下では携帯端末と書く）と情報のやりとりを行いながら、処理を行っていくことである。

Web 上のページ（今回のシステムは HDML）などは、Web サーバ上のファイルとして格納されており、携帯端末からのリクエストに応じて、Web サーバが指定されたファイルの内容を携帯端末に送信することで、ページが表示される。Web サーバだけでページを提供している場合、その内容は固定されたもので、どの携帯端末に対しても同じ内容しか提供できない。

しかし、今回のシステムは個人によってデータベースの登録や更新などといったことが必要なので、静的な内容の提供だけでなく、携帯端末ごとに異なる内容を提供しなければならない。そのためには、入力された内容をサーバ側で解析し、その内容に応じた処理を行い、その処理結果を携帯端末に送信できる仕組みが必要となる。

そのような処理は、Web サーバだけでは不可能に近いので、Web サーバの代わりに処理を行ってくれる機能が必要となる。そのため今回は Java Servlet を利用する。

Servlet が通常の Java Application と異なるところは、JavaVM 上で単独で動作しているのではなく、Servlet Container（今回のシステムは Tomcat3.3.1 を使用以下 Tomcat）という実行環境の上で動作しているということ、そして、その起動・実行がコマンドラインの指示ではなく、携帯端末からのリクエストに応じて行われる。

Web サーバはファイルとして格納されている情報しか送信することができない。そこで、Web サーバに対して拡張(プラグインの追加など)を行うことによって、Tomcat との通信を行うことができるようにする。こうすることによって、以下の動作を実現する。

Web サーバ自身で処理できる内容(HDML ファイルや画像ファイルなど)は自分で処理する。Web サーバでは処理できない内容(Servlet の実行など)は、Tomcat に処理を依頼する。

今回は、Servlet に対して特定の URL のパスを割り当てて、そのパスが指定された場合は、Tomcat に処理を依頼するようにしている。

Web サーバから Tomcat に対してリクエスト(URL 指定やパラメータ内容)が送信されると、Tomcat は、指定された URL をもとに必要な Servlet をメモリ上にロードし、インスタンス化します。その後、Servlet が実行される。

Servlet では、

・リクエスト内容からのパラメータの受信

・クライアント側への表示内容の送信

という処理を行うことができる。つまり、携帯端末から送信されたパラメータを解析し、その内容に応じた処理を行い、その処理結果によってデータベース操作 HDML 表示などをする。

今回 Servlet で HDML を表示させるとき以下のプログラムで行っている

response.setContentType("text/x-hdml;charset=Shift_JIS"); ←文字コードを設定、今回のシステムは HDML なのですべての Servlet の設定はこう記述している HTML のときは response.setContentType("text/x-html")というように out.println で表示する形式によってこの記述を変更する。

out.println(" 表示させるページを（タグを含めて）記述する ");

例：

```
out.println("<HDMLVERSION=3.0MARKABLE=TRUE><DISPLAY><IMGSRC=¥"http://rena.cis.shimane-u.ac.jp/cgi-bin/test/logo.png¥"></DISPLAY></HDML>");
```

4.2.3 Servlet とデータベース

Servlet でデータベースに接続するにはプログラム言語に SQL 文を記述する方法があるがそのためには JDBC が必要となる。JDBC は、Java プログラミング言語から表形式データソースにアクセスするための API であり。JDBC は、広範な SQL データベースへの DBMS の接続性を提供するだけでなく、現在では新しい JDBC API を使用して、スプレッドシートやフラットファイルなどの他の表形式データソースへのアクセス手段も提供している。

今回のシステムでは以下を基準としてデータベースへの接続、操作を行っている、

Class.forName(" 「ドライバクラスの記述」 ");
ドライバクラスをロード今回は HiRDB を使用しているので「ドライバクラス記述」の位置に JP.co.Hitachi.soft.HiRDB_Driver_for_JDBC.JdbcHirdbDriver

con = DriverManager.getConnection(url + system, user,password);
データベースへの接続を表すオブジェクトを作成。データベースの URL、システム名にユーザープロファイル名とパスワードを引数にして実行する。

Statement stmt = con.createStatement();
ステートメント・オブジェクトは、SQL ステートメントをデータベースに送信する

```
String sql = "「SQL ステートメントを記述」";
int num = stmt.executeUpdate(sql);
ResultSet rs = stmt.executeQuery(sql);
```

SQL ステートメントを実行するためには、その SQL ステートメントを格納するコンテナ（ステートメント・オブジェクト）を作成する必要がある。そしてその SQL ステートメントを格納したコンテナに対してその実行を行うメソッドを使用して結果を得る。
update や insert ステートメントの場合は executeUpdate メソッドを使用します。
select の場合は executeUpdate ではなく、executeQuery メソッドを使用します。
結果は executeUpdate 時は int 型、executeQuery 時は ResultSet 型で得る。

```
stmt.close();
con.close();
```

データベースから接続する時はステートメント・オブジェクトとデータベース接続を閉じ

る。

4.2.4 実行画面

・新規登録

top.hdml から２．初めての方を選択し sinki.hdml に移る、ここでなまえ、パスワード、email アドレス、電話番号、携帯電話番号、国籍を入力し、入力確認の後 OK で sinkitest.java でデータベースに個人情報を登録する。**図４．１**

データベースは sinkitest.java により個人データが入力、更新される。**図４．２**

図４．１新規登録の流れ

CUSTOMER_ID	PASSWORD	CUSTOMER_NAME	E_MAIL
0	test	test	test
1	1111	Takashi Hori	s99478@cis.shim
2	test2	test2	test2
3	test4	test4	test4
4	test3	test3	test3
5	hori	takashi	1111
6	0987	TH	email
7	HORI	TAKASHI	email
8	123456	osa	shimane

↓

2	test2	test2	test2
3	test4	test4	test4
4	test3	test3	test3
5	hori	takashi	1111
6	0987	TH	email
7	HORI	TAKASHI	email
8	123456	osa	shimane
9	PASS	NAME	EMAIL
10	masano	Osada	testmail

図４．２登録データベース

・　ログイン及び商品選択

top.hdml からすでに登録しているクライアントは　１．登録済みの方　を選択した後、login.hdml に移る、login.hdml では、なまえ、パスワードを入力して OK ならば２つのパラメータとともに logintest.java に移る、logintest.java では受け取った２つのパラメータを用いてデータベース customer_list に登録してあるか検索して、なまえ、パスワードが位置すれば図４．３の画面４番目を出力する。

画像を選択する際は　１．情報一覧画面へ　を選択後画像を選択する、choice.java に移る図４．３の画面 5 番目。

アイテム０１を選択すると、item.java に選んだ画像に対応する番号（以下 item_id）と個人情報（なまえとパスワード）をパラメータとして渡す、item.java は item_id に対応する画像を出力すると同時に、データベースにアクセスして customer_list から個人情報をもとに customer_id を受け取り download_list を customer_id と item_id を元に検索し download_list になければ、新たに download_id、customer_id、item_id、the_date、を入力し customer_list の total_price を一定料金分加算する。**図４．４、図４．５**

download_list に検索した、個人情報と item_id があれば、download_list の入力や total_price の加算などをせず画像を携帯端末に出力し終了する。

図４．３アイテムの選択

```
 ----------- ----------- ------- ----------
 0           1           1       2003-01-23
 1           0           1       2003-01-23

KFPX27010-I             2 rows selected

 DOWNLOAD_ID CUSTOMER_ID ITEM_ID THE_DATE
 ----------- ----------- ------- ----------
 0           1           1       2003-01-23
 1           0           1       2003-01-23
 2           1           2       2003-01-29
```

図４．４　download_list の追加入力

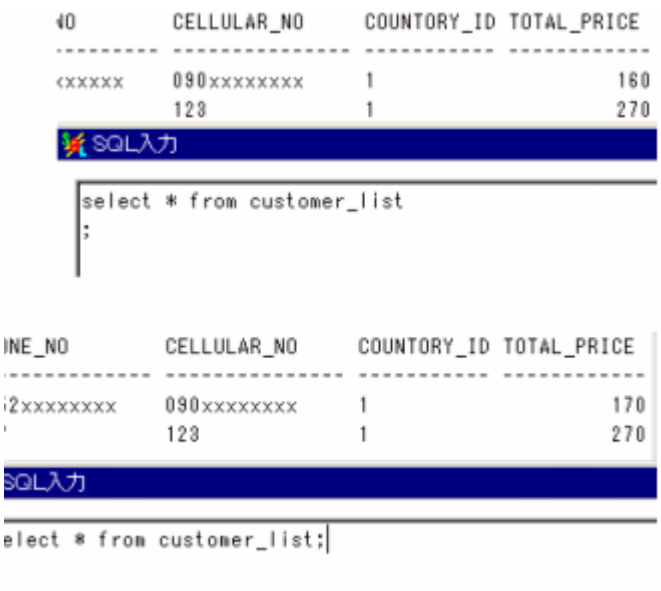


図４．５料金の加算

プログラムのフローチャート及び内容は４.２.１項：概要で述べているのでそちらを参照。

結論

本研究では、実際の携帯電話から Web サーバに接続し、リクエストに応じてデータベースを参照して画像を表示できるシステムが作成できた。一連のシステムを作成した結果、現在の携帯電話に搭載されている Java の技術的な状況が理解できた。今後の課題は、本システムでは対応する端末が KDDI 用に限られるので、NTT Docomo、J-PHONE の携帯端末が接続できるシステムにする事である。

謝辞

本研究において最後まで暖かく丁寧な指導、又助言をしてくださった田中章司郎先生に深く感謝の意をここに表します。

そして、田中研究室の辰巳圭介さん、貫目洋一さん、高木明さん、中村吉郎さん、長田昌訓くん、喜代吉容大くん、垰田千帆さん、田辺知里さんには研究に関して数々の助言をいただきました。厚く御礼申し上げます。

さらに、本論文と本研究で作成したプログラムなどのすべての著作権は田中章司郎教授に譲渡いたします。

引用文献・ホームページ・資料

橋本賢一著「携帯電話 Java」リックテレコム　２１１PP　２００１

「IBM Java technology」

http://www-6.lim.com/jp/developerwrks/java/011019/j_j-j2me.html

http://feeling.zive.net/javapages/touch/jvmandsoft.html

「Sun Microsystems」http://java.sun.com/

システム作成時参考としたホームページ

Tomcat：　http://jakarta.apache.org/

Tomcat：http://www.a.mei.titech.ac.jp/~eakagawa/tomcat.html

JavaServlet：http://java.sun.com/j2ee/ja/servlet/download.html

JDBC：http://java.sun.com/j2ee/ja/jdbc/index.html