M-66

J2MEによるシンクライアントの実現

A GUI Thin Client based on J2ME

轟木伸俊† 北村操代†
Nobutoshi TODOROKI Misayo KITAMURA

1. はじめに

筆者らは，遠隔地から携帯端末を通じてプラントの状態の監視制御を行えるフレームワークを開発している．本フレームワークは，監視データサーバから取得したプラントの状態と携帯端末用の監視画面仕様から動的に監視用 GUI を生成し，携帯端末にプラントの状態を表示する．

今回，携帯端末上にシンクライアントを実装する手法を用いて，フレームワークを試作した．本稿では，本手法の詳細について説明するとともに，実行性能の評価結果を述べる．

2. システム概要

監視制御システム全体と，本フレームワークの適用部分の構成を図１に示す．破線部が本フレームワーク適用部分である．監視データサーバは，コントローラから状態を収集・蓄積する．コントローラのデータには，コントローラから定周期で上がる情報（現在値データと呼ぶ）と，異常時・状態変化時に上がる情報（イベントデータと呼ぶ）がある．また，監視データサーバは，時系列の推移データ（トレンドデータと呼ぶ）を保持している．

本フレームワーク適用部分の主な構成要素は，WWW サーバと携帯端末である．これらはサーバ/クライアントシステムとして動作する．まず，携帯端末が WWW サーバに HTTP を用いて問い合わせを行う．この問い合わせに基づき，WWW サーバは監視データサーバに問い合わせを行い，上記プラントの各状態を取得する．そして，WWW サーバは情報を適切な形に加工して携帯端末に応答する．

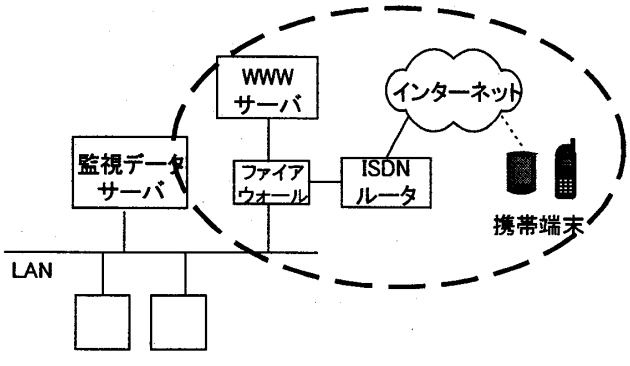


図１：システム構成

3. 設計

Java 言語のプログラム規約である J2ME(Java 2 Platform, Micro Edition)を用いてフレームワークを試作した．以下，その概要を述べる．

3．1 J2ME の採用

本システムでは，1)携帯端末の画面に様々な情報（トレンドグラフ，イベント情報等）を表示する，2)監視画面上の監視対象を示すアイコン（以下，シンボルと呼ぶ）を選択させて，その対象に対する操作を行わせる，3)複雑な操作のために，ファンクションキーに様々な機能を割り当てる，等を実現する．携帯端末の WWW 閲覧機能だけでは上記 2,3 を満足できないため，本システムでは J2ME を用いて必要な機能を実現する．

3．2 超軽量クライアント

携帯端末に搭載するソフトウェアは軽量のクライアントプログラムとする．これは，処理能力が低い，及び，搭載アプリケーションのプログラムサイズに制限がある（携帯電話では 10～30KBytes），という携帯端末の制約のためである．トレンドグラフを表示するためには，トレンドデータ取得，データの正規化，グラフ生成，及び，グラフ表示というプロセスが必要であるが，これらすべてを携帯端末内で行うと，上記制限を大幅に上回ることになる．

クライアントは，WWW サーバ（以下，サーバと呼ぶ）から受信した描画コマンド（端末画面に表示する GUI 部品を指定する）を解釈し，端末画面上への描画を行う機能のみを持つシンクライアントとする．描画コマンドとしては，ボタン部品描画コマンド（描画座標，表示文字列，押下時の振舞いを指定して描画），長方形部品描画コマンド（始点座標，線色，幅，高さを指定して描画）等がある．

サーバは，クライアントが描画すべき画面を描画コマンドの集合で表現し，クライアントに送信する．描画すべき画面は，予め作成した監視画面テンプレートと，監視データサーバから取得したプラントのデータから動的に生成する．監視画面テンプレートは JSP（JavaServer Pages）を用いて記述する．

3．3 端末依存 API の隠蔽

J2ME の API 仕様は複数あり，開発対象となる端末毎にその API 仕様が異なることがある．描画コマンドが端末の API 仕様に依存する場合，同じ画面を表示する際でも端末毎に描画コマンドを変更する必要がある．この問題の解決のために，端末依存 API を端末内に隠蔽するアーキテクチャを採用した．具体的には，1)描画コマンドは端末非依存にする．2)クライアントが描画コマンドを解釈し，画面に描画する際にのみ端末依存 API を使用する，というものである．本ソフトウェアのアーキテクチャを図３に示す．端末依存 API を用いて処理を行う部分を斜線で示している．例えば，ボタン部品の描画コマンドを受け取った場合，端末の API 仕様が Doja（ドコモ Java 拡張）ならば Button クラス，MIDP (Mobile Information Device Profile) ならば Command クラスを用いて描画を行う．

†三菱電機株式会社 先端技術総合研究所

図３における GUI 部品群は描画コマンドを出力する API の集合である．本 API 群は，監視画面テンプレートの作成時に JSP 内で使用することができる．本 API 群には，ボタン部品 API，テキストボックス部品 API，線分部品 API 等がある．部品には属性値（部品の大きさ，色，アクション発生時の振舞い等）を設定することができる．プリミティブな部品を組み合わせることで，高機能部品を実現する．例えば，線分部品を組み合わせることで，グラフ部品を作成できる．

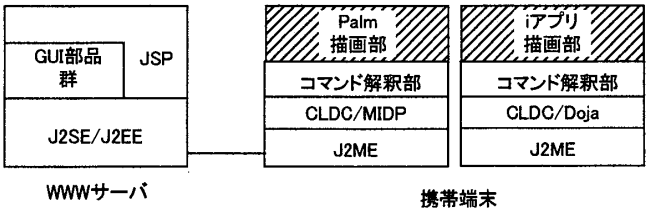


図３：ソフトウェアアーキテクチャ

４．試作と評価

携帯電話の iアプリ(Doja2 API 仕様)上に試作したクライアントを用い，実行時の性能を評価した．システム構成は２節と同様である．試用の結果，プラント監視システムの GUI 端末として，実用に耐える速度で動作することが分かった．

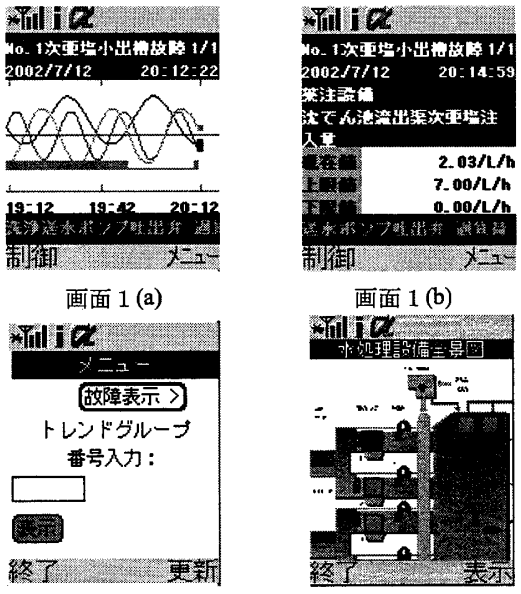
評価方法として，サーバへの問い合わせ発生時から画面出力が完了するまでの応答時間を測定した．評価対象の画面を図４に示す．画面１は関連する４本のトレンドグラフ(a)とその詳細画面(b)を表示したものである．４つのグラフそれぞれに対応する詳細画面が存在するため，画面１は，(a)×１画面+(b)×4 画面の計５画面で構成される．画面２は，ボタンとテキストボックスよりなるトレンドグラフ指定画面である．本画面のみ監視データサーバへのアクセスは発生しない．画面３は，プラントの中央監視室で表示される監視画面を模擬したものであり，プラント全景図上に監視対象を表すシンボルを重ねたものである．全体で４画面分の広さである．

測定結果を表１に示す．測定には当社製携帯電話 D504i と，その比較対象として，PC 上のエミュレータ[1]を用いた．PC とインターネットとの接続には，実機と実効通信速度が近い 14.4Kbps のモデムを使用した（本評価に先立ち実施した実機でのファイル転送試験において，実機の実効通信速度は 11Kbps であった）．試験により，実機の応答時間において，描画コマンド解釈・描画時間（CPU 処理能力に依存する．以下，解釈描画時間）が支配的であることが分かった．今後の CPU 処理能力の向上により応答時間を改善できると考えられる．

以下，考察の詳細を述べる．次の条件を仮定する．1)クライアントはサーバからの最初の応答パケットが到着すると，解釈・描画処理を始める，2)最後のパケットが到着，且つ，解釈・描画処理が終了しないと処理は完了しない，3)PC と実機の実効通信速度は等しく 11Kbps とする．画面１(実機)での応答時間の内訳は，最初の応答パケットが到着するまでの時間（以下，応答待ち時間），及び，端末での処理時間（以下，端末処理時間）である．まず応答待ち時間を求める．本時間は，携帯電話キャリアでの遅延時間，及び，インターネットでの遅延とサーバ処理時間からなる．前者は，通信が支配的な条件（画面２）での実機(2.3 秒)，及び，エミュレータ(1.4 秒)の応答時間の差より 0.9 秒と見積もることができる．後者は，画面１のエミュレータの応答時間(3.8 秒)と，通信時間(仮定 3 より 3.4 秒)の差より，0.4 秒と見積もることができる．以上より，画面１(実機)での応答待ち時間は 1.3 秒である．次に，端末処理時間は，実機での応答時間(7.3 秒)から先に求めた応答待ち時間(1.3 秒)を引いて，6 秒である．ここで仮定 1,2 より，端末処理時間は，解釈描画時間あるいは通信時間の内，より長い方である．通信時間は 3.4 秒なので，この 6 秒とは解釈描画時間であると分かる．以上から解釈描画時間は応答時間の 8 割を占めており，支配的であると判断できる．

５．おわりに

本稿では，J2ME を用いたシンクライアント構築手法について述べた．また，試作システムにおいて実行時の性能評価を行い，プラント監視分野における利用では実用的な速度で動作することが分かった．今回の試作では，J2ME における CLDC (Connected Limited Device Configuration)に基づく API 仕様を用いたが，今後は，他の API 仕様を用いた場合の拡張点について検討していく予定である．



画面１(a)　画面１(b)

画面２　画面３

図４：評価対象の画面（画面はエミュレータのもの）

表１：評価結果

	GUI 部品数	バイト数	応答時間(秒)	
			実機	エミュレータ
画面１	107	3773	7.3	3.8
画面２	9	429	2.3	1.4
画面３	13(+背景画像)	1616 (+3287)	9.5	5.5

参考文献

[1] NTT DoCoMo, Inc.: 504i 向けアプリ開発ツール，http://www.nttdocomo.co.jp/mc-user/i/java/tool_504i.html