(2T－05) Java GUIリモート化システムの開発

デモ24

村松　孝治　　小野　泰志

(株) 東芝 デジタルメディア機器社 コンピュータソフトウェア部ソフトウェア設計第2担当

1. はじめに

近年、Java 言語はその機能充実・性能改善と共に急速に浸透してきており、現在では基幹業務系システムにも耐え得る開発言語としての地位を確立しつつある。現在最も一般的な Java の利用形態として、サーバアプリケーションとクライアントアプレットとの組み合わせが挙げられる。この形態のプログラムの開発には通信関連の設計／開発に労力がかかる。又、既存のアプリケーションをこの形態に修正するという場合であってもかなりの修正量がある事が多い。そこでこれらの問題を解決する為、Java アプリケーションの GUI 部分を実行時に他のマシンに表示する、表示中継ライブラリ・表示デーモンを開発した。

2. Java GUI のリモート化の考え方

Java GUI は下記の3層構成を取っている。

1. AWT (Abstract Window Toolkit)
   GUI を抽象表現したプラットフォーム非依存の API 群
2. Peer
   AWT と Native Window System との命令を仲介する API 群（プラットフォーム依存）
3. Native Window System
   プラットフォーム特有の API 群

1 と 2 の間を取り持つ API は "java.awt.Toolkit" と呼ばれる単一のクラスに集中している。このクラスを変更することで 1 から 2 以外の API、すなわち 2 と互換性のある独自作成 API を呼ばせることも可能である。本開発ではこの方式により、GUI を他のマシンに表示させる機能を実現している。この方式は 1 の API 自体を変更する方式と比較して変更量が少なく、又 1－2 間の API 呼び出しのみをリモート呼び出しの対象とするため通信量が少ないという特徴がある。一方、本方式を X プロトコルと比較すると、オブジェクト単位のマクロ化 API をリモート呼び出しするため通信量が少ないという特徴がある。

3. システム構成／動作

本システムのシステム構成図を以下に示す。

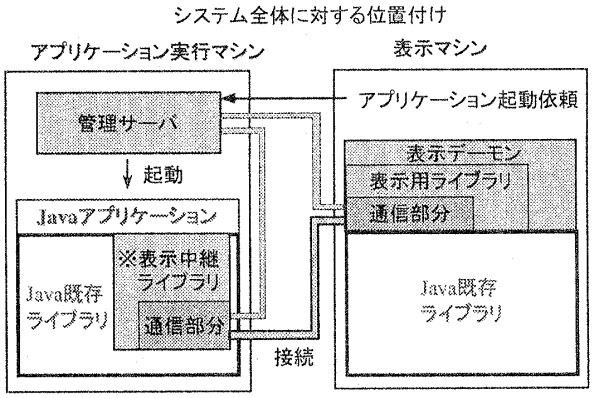

図 1 システム構成図

本システムでは2つの必須構成要素（表示中継ライブラリ、表示デーモン）と1つのオプション構成要素（管理サーバ）が存在する。各構成要素は以下の機能を提供する。

A Development of Java-GUI Remote Control System

by Koji Muramatsu / Yasushi Ono

Toshiba Corporation Digital Media Equipment & Services Company Computer Software Dept. Software Designing Group 2, 1 Toshiba-cho, Fuchu, Tokyo 183-8511, Japan

- 表示中継ライブラリ（必須）
  Java アプリケーション実行時に表示デーモンと接続。又、Java アプリケーションからの表示関連の命令を表示デーモンに中継し、逆に表示デーモンからのイベント発生通知を受信・処理する。
- 表示デーモン（必須）
  表示中継ライブラリと接続。又、表示中継ライブラリからの表示関連の命令の受信と実行を行い、逆にイベント発生時に表示中継ライブラリにイベント発生通知を行う。
- 管理サーバ（オプション）
  各マシン上で動作する表示デーモンを管理。又、アプリケーション実行依頼の受信と実行を行う。本機能は CGI 等によっても代用可能であるためオプションとしている。

実動作は以下になる。

- 管理サーバを起動
- 表示デーモンを起動
- 表示デーモンから管理サーバに対し、アプリケーション実行依頼を発行
- 管理サーバがアプリケーションを実行
- アプリケーションから表示中継ライブラリへの API がコールされる
- 表示中継ライブラリが表示デーモンと接続
- この後表示中継ライブラリへの表示命令は表示デーモンに中継・実行され、マウス移動等のイベント発生通知は表示デーモンから表示中継ライブラリに送信・処理される

4. 実装上の特徴

本方式には主に以下の特徴がある。

- 速度性能・メモリ節約のための独自プロトコル（Socket 上に実装）
- API コールのマクロ化
- デッドロック防止のためのスレッド管理
- 速度性能改善のための非同期描画機能
- 通信量改善のためのリアルタイム圧縮機能

5. 性能

性能としては以下の計測結果が出ている。

- 単純描画速度性能
  単純線描画・文字描画・画像描画等につき、単一マシンで実行する場合と比較した際の速度劣化は 1.5～2 倍程度。
- API 遠隔呼出し速度性能
  通信経由の API 呼出し時間は 3～4[msec/回]（＝単一マシン上の処理の約 2000 倍）
- 通信量
  X Window の場合と比較して 1/2～1/5 程度。
- メモリサイズ
  単一マシンでの実行の場合と、本システムの表示マシン上の場合とで比較すると、GUI のみのアプリケーションにて 80～110%。
  つまり、GUI 以外のメモリを大量に(α%)消費するアプリケーションでは(約 100－α)%。
- 体感速度性能（参考）
  実際にアプリケーションを操作した際の体感的な速度劣化は約 5 倍程度。

※計測は全て通信速度 10Mbps の LAN 上。

6. まとめ

単一のマシン上でそのまま動作する Java アプリケーションの表示を実行時のオプション指定のみで別のマシン上に表示する事が可能なシステムを構築する事が出来た。開発量軽減、既存アプリケーションの流用運用等に効果を発揮する。今後は性能改善を課題として進め、又組み込み機器等への応用をはかる予定である。

7. 参考文献

・JDK1.1.x Documentation（英語版）
http://www.javasoft.com/products/jdk/1.1/docs/
