HORBを利用したロボット操作環境

平松 薫†　森 啓†　納谷 太†　大里 延康†

1　はじめに

筆者らは，マルチエージェント制御ロボットの分散制御実験[1]で作成したモジュールを再利用し，Webブラウザ上の Javaアプレットから，ネットワーク経由でロボットの操作やセンサの状態の表示を可能にする仮想ロボットターミナル[2]の実装を行ってきた．仮想ロボットターミナルでは，C言語の低水準インタフェースを利用してロボットやセンサの制御と各コントローラからの情報をネットワークへブリッジするデーモンを開発し，そのユーザインタフェースを Javaアプレットで開発することで，Webブラウザ上からのロボット操作とプログラムの可搬性を実現した．そして，デーモンと Javaアプレットがソケットを利用して通信を行うようにし，各々に必要最小限の機能のみを実装することで処理の効率化を図っていたが，この構成では通信において低レベルな記述が必要なためプログラムが煩雑になる傾向があった．

そこで，仮想ロボットターミナルのデーモンや Javaアプレットの開発を容易にするために，Javaベースの分散言語システム HORB[3]を導入し，プログラムの記述性を改善することにした．また，HORBを導入するのに伴い，従来 C言語で記述していたデーモンも Javaで再構築を行った．本稿では，HORBを導入したシステムと従来のソケットを利用したシステムの性能を比較し，リアルタイム性の要求されるロボット操作環境においての Javaで構成したシステムの実用性について検討する．

2　仮想ロボットターミナルへの HORBの導入

仮想ロボットターミナルは，ロボットや各種センサに接続するコントローラ，コントローラの制御と情報のブリッジを行うデーモン，Webサーバが動作するパソコンまたはワークステーションと Webブラウザの動作する端末で構成する．そして，Webブラウザ上で動作する Javaアプレットでユーザインタフェースを構築し，プラットフォームに独立した操作環境を実現する(図1)．

† NTT コミュニケーション科学研究所，
619-02　京都府相楽郡精華町光台2-2，
Tel: 0774-95-1827, Fax: 0774-95-1871,
e-mail:{hiramatu,akira,naya,osato}@cslab.kecl.ntt.co.jp

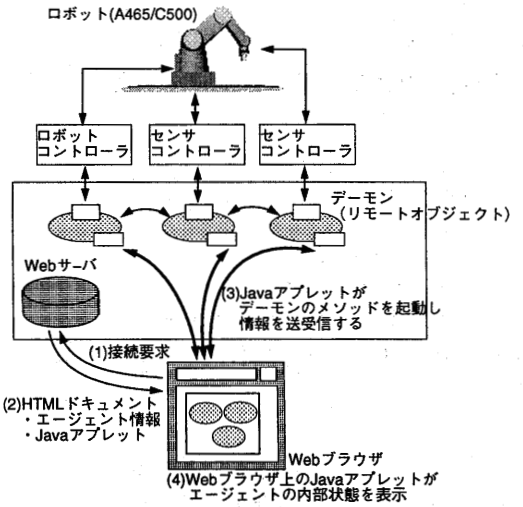

図 1: 仮想ロボットターミナルの構成

2.1　デーモンの Javaオブジェクト化

従来の仮想ロボットターミナルでは，デーモンを C言語で開発していたために，既存の分散言語システムの導入が難しかった．このため，ソケットを利用して通信を行うようにしていたが，デーモン，Javaアプレットの双方で通信に関する低レベルな記述をする必要があった．そこで，Javaベースの分散言語システムのうち記述の簡単な HORBを仮想ロボットターミナルへ導入し，通信に関連する部分を再構築することにした．HORBを利用するためにはデーモンも Javaアプリケーションとして構築する必要があるので，従来のデーモンの機能をうちの Javaで実装することが難しい各種ポートの低水準入出力は，C言語で記述したネイティブメソッドを呼ぶ形で実装した(図2)．

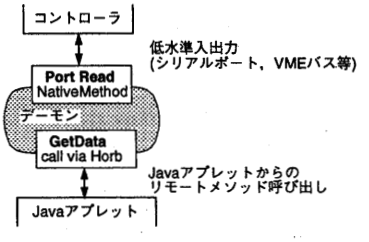

図 2: ネイティブメソッドを利用したデーモン

HORBを導入することで，ネットワーク透過なメソッド起動が可能となるので，Javaアプレットがデーモン側のメソッドを起動する形でセンサやロボットの状態を取得し，表示を行うような記述が可能になる．

3 性能比較

HORBを導入した仮想ロボットターミナルと従来のソケットを用いたバージョンを表1の環境で動作させ，比較を行った．測定にはセンサとしてデータグローブを利用し，各指に付いた合計10個の角度センサのデータを簡単なグラフィックで表示するJavaアプレット(図3)の動作で，ソケットで通信する場合とHORBを導入した場合の性能を比較した．

表 1: 測定環境

サーバ	Sun Ultra30 (Solaris2.6)
クライアント	AMiTY VP (Win95)
	InternetExplorer 3.0
ネットワーク	10BASE/T (同一ハブに接続し，他と隔離)
センサ	データグローブ (VPL社)
開発環境	JDK1.1.3, JDK1.0.2, HORB 1.3b1

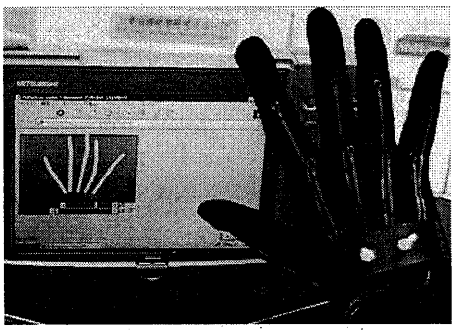

図 3: センサデータを表示する Javaアプレット

測定にはサーバ上の tcpdump コマンドを利用した．Javaアプレットからのデータ要求パケットとデーモンからの応答パケットのタイムスタンプを記録し，オフラインでそれらのパケットの間隔を計算し，それぞれ統計処理を行った．

3.1 デーモンの反応時間

従来のデーモンは C言語で実装し，内部をマルチスレッド化することで処理の効率化を図り，Javaアプレットとの通信はソケットを通じて行うように実装した．また，HORBバージョンのデーモンは，低水準入出力をネイティブメソッドとして実装し，HORBのリモートオブジェクトとして動作する Javaアプリケーションへと変更した．どちらのバージョンのデーモンも，センサから連続的に出力されるデータを一旦バッファリングし，Javaアプレットからデータを要求された時点で最新のデータを返せるように実装し，ほぼ同じ手順で処理を行うようにした．

表2は，デーモンの反応時間の測定結果である．反応時間は，データ要求パケットと応答パケットの間隔で測定した．

表 2: デーモンの反応時間

	HORBで通信	Socketで通信
平均(ms)	0.732	0.175

HORBバージョンをソケットバージョンと比較すると，反応時間の差は約0.55msであった．この差は，HORBの各オブジェクト間のメソッドコールと Javaのネイティブメソッドコールのオーバーヘッドが原因と考えられる．

3.2 アプレットの更新周期

表3はアプレットの更新周期を測定したものである．測定に利用した Javaアプレットは，通信とグラフィック表示を逐次的に行うので，Javaアプレットからのデータ要求の間隔でグラフィックの更新周期を測定できる．また，同じJavaアプレットで表示を行わない場合も併せて測定した．

表 3: アプレットの更新周期

	HORBで通信	Socketで通信
ポリゴン表示(ms)	28.71	27.58
表示なし(ms)	3.45	2.31

測定の結果，実験環境ではソケットで通信する場合，HORBを利用した場合ともに毎秒30フレーム以上でグラフィック表示を行うことができた．また，表示なしで通信のみを行う Javaアプレットとの比較から，処理時間の約90%がグラフィックの表示に費やされていることが分かった．従って，表示と通信では，表示処理の方が Javaアプレットの動作速度に対して大きな影響を与えるので，表示処理の効率化を図ることの方が重要であることが分かった．

4 まとめ

以上の測定結果から，簡単なデータであれば Webブラウザ上の Javaアプレットでも十分な表示速度が得られることが分かった．今後は，本格的なロボット操作環境を構築するために，Javaアプレット上での複雑なデータの表示やロボットの操作方法を検討していく予定である．

参考文献

[1] A.Mori, et al. "Multiagent-based distributed manipulator control". In *MFI'96*, pp. 289–296, 1996.

[2] 平松他. "マルチエージェント制御ロボットにおける Webブラウザを利用した内部状態の可視化". 第55回情報処理学会全国大会講演論文集(3), pp. 3–432,433, 1997.

[3] http://ring.etl.go.jp/oprnlab/horb/.