システムソフトウェアと
オペレーティング・システム 77－8
マルチメディア通信と分散処理 87－8
（1998. 2. 26）

青果物市況情報データベースNAPASSを用いたJAVA-RMIとHORBの性能比較†

本田 茂広[*1]、下村 道彦[*2]、南石 晃明[*3]、木浦 卓治[*4]、二宮 正士[*5]、大谷 信博[*6]

[*1,*2,*6]三菱スペース・ソフトウエア（株）情報エンジニアリングセンター
[*3]東北農業試験場、[*4,*5]農業研究センター

青果物市況情報データベースNAPASSのJAVA-RMI化およびHORB化を行なった。本稿では、NAPASSのJAVA-RMI化版およびHORB化版の一部を変更し、データ転送速度の比較を行なった。NTサーバでバーチャルマシンjviewとした場合、サーバの性能、クライアントの性能、およびVMの違いで、絶対値的には差が出るものの、傾向として配列数が少ない場合はHORBの転送速度が勝り、配列数が多い場合はJAVA-RMIの転送速度が勝る結果が観測された。

Performance Comparison of JAVA-RMI and HORB by Using NAPASS (Nationwide Agricultural products Price Analysis Support System) †

HONDA, Shigehiro[*1], SHIMOMURA, Michihiko[*2], NANSEKI, Teruaki[*3], KIURA, Takuji[*4], NINOMIYA, Seishi[*5], OHTANI, Nobuhiro[*6]

[*1,*2,*6]MITSUBISHI SPACE SOFTWARE CO., LTD.
INFORMATION SYSTEM ENGINEERING CENTER
[*3]Tohoku National Agricultural Experiment Station
[*4,*5]National Agriculture Research Center

We have made NAPASS, which is a market price database of agricultural products, accessible by JAVA-RMI or HORB. In the course of our work we conducted performance comparison tests between JAVA-RMI and HORB.

This paper reports the results of this testing. Although the performance of a server, a client or a virtual machine was dominant, we observed that HORB was faster in dealing with a small number of arrays and that JAVA-RMI transmitted a large number of arrays faster.

---

†この研究は農林水産省農林水産技術会議事務局が実施している「増殖情報ベースによる生産支援システム開発のための基盤研究」の一環として行われたものである。

[*1] honda@mi.mss.co.jp, [*2] shimomur@mi.mss.co.jp, [*3] nanseki@affrc.go.jp,
[*4] kiura@affrc.go.jp, [*5] snino@narc.affrc.go.jp, [*6] ohtani@mi.mss.co.jp

1 はじめに

1995 年にオブジェクト指向言語 Java [1] が SUN 社により発表されて以来、1995 年に Java を拡張した分散環境 HORB[2] が平野により発表され、1997 年に RMI[3] が SUN 社により発表された。

農林水産省では、今までに蓄積した分散環境にあるデータおよびプログラムの利用環境を向上させるために、これらの技術を用いたネットワーク・エージェント利用研究を研究†課題の一つとして行なっている。このネットワーク利用者の計算機環境およびネットワーク環境は理想的な状態ばかりとは限らない。

そこで、本稿は、青果物市況情報データベース NAPASS に実装した RMI および HORB の動作環境（バーチャルマシン（VM）による違い、計算機 CPU パワーの違い）における影響を調査することを目的とする。
以降 2 章では実装した「青果物市況情報データベース NAPASS の概要」を記し、3 章では計測の流れ、性能比較の計算機環境、使用した VM、結果ならびに考察を行なう。

2 青果物市況情報データベース NAPASS の概要

青果物市況情報データベース NAPASS（Nationwide Agricultural products Price Analysis Support System）は 1983 年から南石により開発されたキャラクタベースのインタフェースを持つ全国 68 市場の 150 品目・品種以上の日別データ（市場、農産物価格、入荷産地、入荷量等）を時系列分析支援するためのシステムであり、1994 年に江渡らにより WWW 化され、1996 年に南石らにより Java 化された[4]。1997 年に作成した NAPASS の JAVA-RMI 化（以下 RMI 版と記す）、HORB 化（以下 HORB 版と記す）の WWW インタフェース例（図 2.1）を示す。[5]

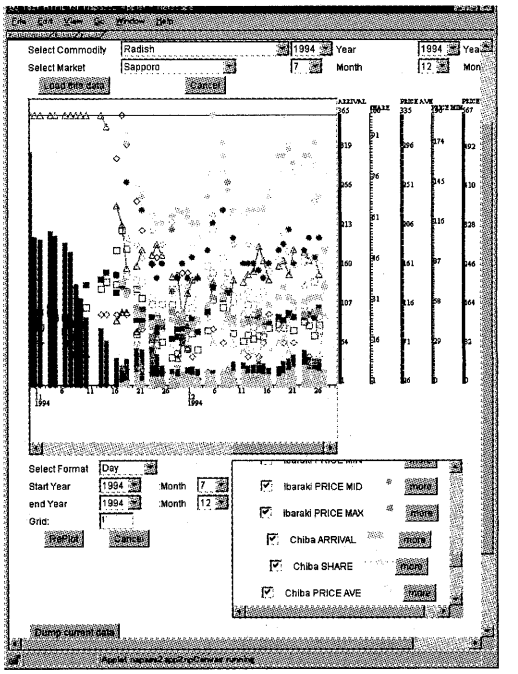


図 2.1 RMI 版、HORB 版インタフェース

2.1 プログラム機能構成

RMI 版および HORB 版（図 2.2）の機能構成を示す。

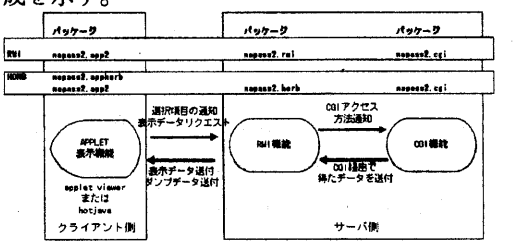


図 2.2 RMI 版および HORB 版の機能構成

図 2.2 に示す機能構成の内、RMI および HORB による通信データは、・選択項目の通知,・表示データリクエスト,・表示データ送付,・ダンプデータ送付、である。
図 2.2 中の napass2.rmi パッケージの機能概

略は図 2.3 のとおりであり、クライアント側から見た JAVA-RMI の流れは(1)～(4)である。

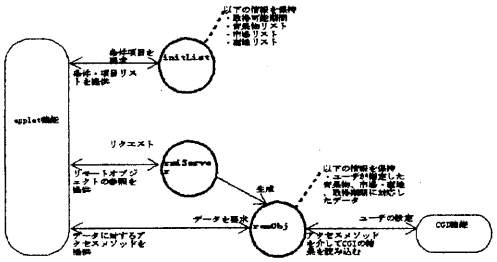


図 2.3　napass2.rmi パッケージの機能概略

(1) クライアントは、initList オブジェクト（NAPASS リストサーバ）より選択可能な条件項目のリストを得る。
(2) クライアントは、rmiServer オブジェクト（NAPASS RMI サーバ）より remObj オブジェクト（NAPASS リモートオブジェクト）を生成させ、そのスタブオブジェクトを入手する。
(3) クライアントは、(1)のリストを元に選択された条件で remObj オブジェクトを初期化する。 remObj オブジェクトは、その条件のデータを cgi を用いてデータベースサーバより入手する。
(4) クライアントは、グラフ化を行なうデータを、remObj オブジェクトより入手し、図 2.1 に示したような描画を行なう。

図 2.2 中の napass2.horb パッケージの機能概略は図 2.4 のとおりであり、クライアント側から見た HORB の流れは(1)～(4)である。

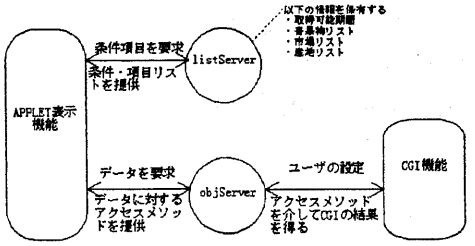


図 2.4　napass2.horb パッケージの機能概略

(1) クライアントは、listServer オブジェクト（NAPASS リストサーバ）より選択可能な条件項目のリストを得る。
(2) クライアントは、 objServer オブジェクトを生成させ、そのプロキシオブジェクトを入手する。
(3) クライアントは、(1)のリストを元に選択された条件で、objServer オブジェクトを初期化する。 objServer オブジェクトは、その条件のデータを cgi を用いてデータベースサーバより入手する。
(4) クライアントは、グラフ化を行なうデータを、remObj オブジェクトより入手し、図 2.1 に示したような描画を行なう。

3　RMI 版と HORB 版の比較環境

3.1　計測の流れ

2.1 節では、RMI 版の napass2.rmi 機能概略と HORB 版の napass2.horb 機能概略をクライアント側から見た説明を行った。ここでは転送速度の計測にあたり、2.1 節に示した流れを次のように変更する。

・RMI 版

(1) クライアントは、rmiServer オブジェクト（NAPASS RMI サーバ）より remObj オブジェクト（NAPASS リモートオブジェクト）を生成させ、そのスタブオブジェクトを入手する。
(2) クライアントは、固定条件で remObj オブジェクトを初期化する。 remObj オブジェクトは、その条件のデータを cgi を用いてデータベースサーバより入手する。
(3) クライアントは、タイマをスタートする。
(4) クライアントは、グラフ化を行なうデータを、remObj オブジェクトより入手する。
(5) クライアントは、タイマをストップする。

・HORB 版

(1) クライアントは、objServer オブジェクトを生成させ、そのプロキシオブジェクトを入手する。

(2) クライアントは、固定条件で、objServer オブジェクトを初期化する。 objServer オブジェクトは、その条件のデータを cgi を用いてデータベースサーバより入手する。

(3) クライアントは、タイマをスタートする。

(4) クライアントは、グラフ化を行なうデータを、remObj オブジェクトより入手する。

(5) クライアントは、タイマをストップする。

3.2 試験条件

(1)計算機環境（ハードウェア）

・hnd (Micron MILLENNIA)

OS : Windows NT4.0SP3

CPU : Pentium166MHz

Memory : 128MB

Ethernet : 3Com 509B (ISA, 10Mbps)

・phnd (DELL DIMENSION XPS266)

OS : Windows NT4.0 SP3

CPU : PentiumII 266MHz

Memory : 64MB

Ethernet : AlliedTelesis LA100-PCI (PCI, 10Mbps)

・bianco (SUN Ultra 30 Model 250)

OS : Solaris 2.5.1

CPU : UltraSPARC-II 248MHz

Memory : 256MB

Ethernet : (PCI, 10Mbps)

(2)計算機環境（ソフトウェア）

・jpp	java performance pack
・jview	SDK 2.01
・HORB	1.3b1
	+ ArrayAccelarator 1.0a
	+ Performance patch

・sunjit

(3)ネットワーク環境

ハブ：CenterCOM 3012TR

ネットワーク 10Mbps

(4)リモートオブジェクトのメソッド呼出し

```
dataA [ ] method (
        int data, int data, int data
)
```

なお、RMI では) の後に

throws java.rmi.RemoteException

が必要

(5)オブジェクト転送のためのクラス

```
class dataA {
        method
          ・
        method
        float data
        int data
}
```

3.3 試験結果および考察

サーバ phnd、VM jview を使用し、NAPASS データ 1992 年～1996 年（配列数 1355）+ダミーデータ（配列数 18000）を転送した結果を図 3.1 に、サーバ hnd、VM jview を使用し、NAPASS データ 1992 年～1996 年を転送した結果を図 3.2 に示す。 なお、クライアントの計算機は phnd, hnd の 2 通り、VM は jview, jpp の 2 通り、および RMI と HORB の組合せの計8通り、並びにクライアント計算機 bianco, VM は sunjit で RMI と HORB の組合せの計 2 通りである。

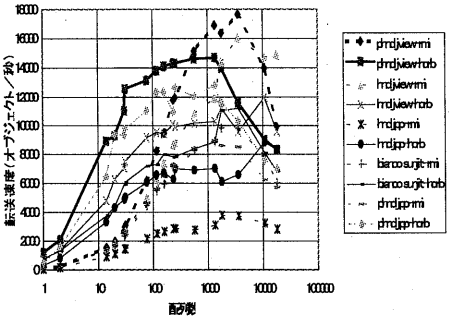


図 3.1 サーバ phnd, jview 配列数対転送速度

注：配列数は 2.2(5)で示した class の float data, int data のセットの個数を表わし、転送速度のオブジェクトは 2.2(5)で示した class を意味する。また、凡例はクライアントの計算機条件および RMI か HORB を指す。

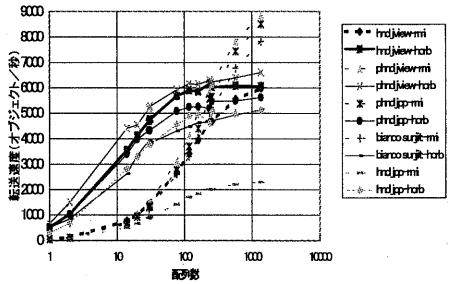


図 3.2 サーバ hnd, jview 配列数対転送速度

図 3.1 および図 3.2 のデータを HORB で規格化、すなわち（同一サーバ名 jview - 同一クライアント名 VM - RMI）/（同一サーバ名 jview - 同一クライアント名 VM - HORB）とした場合の結果を図 3.3 および図 3.4 に示す。

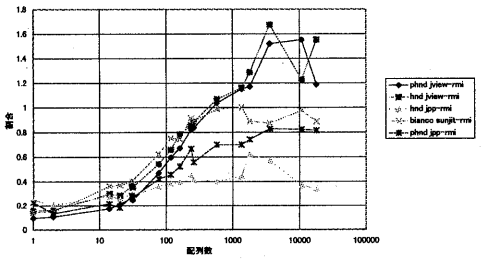


図 3.3 サーバ phnd, jview の配列数対転送速度

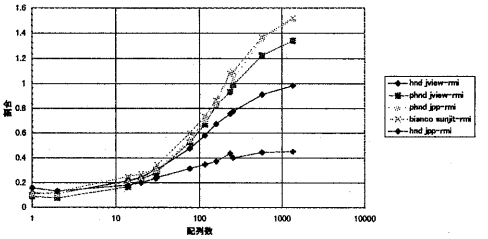
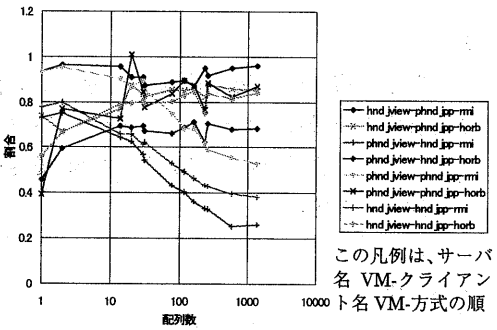


図 3.4 サーバ hnd, jview の配列数対転送

配列数約 200 までは HORB が RMI の最大 5 倍程度の転送速度が出ている。
配列数約 200 以上は RMI が HORB の最大 1.5 倍程度の転送速度が出ている。
傾向は変わらないが、計算機環境によってもかなりのバラツキを生じる。

VM の差異を見るために、図 3.1 と図 3.2 のデータを（同一サーバ名 jview - 同一クライアント名 jpp - 方式）/（同一サーバ名 jview - 同一クライアント名 jview - 方式）とした場合を図 3.5 に示す。
ここで方式は RMI もしくは HORB を指す。



この凡例は、サーバ名 VM-クライアント名 VM-方式の順

図 3.5 VM の違いによる比較

jpp RMI は転送配列数が増大するに従い、クライアントの計算機性能に依存する度合が上がる傾向にある。逆の見方をすれば、jview

RMI はクライアントの計算機性能に依存する度合いが下がる傾向にある。転送速度に関する限り、jpp に比べ、jview の方が転送速度が高い。jpp HORB は立上り性能が jview に比べ悪い。 RMI に比べた場合 HORB は VM による影響度が少ない。

4 おわりに

本稿では、青果物市況情報データベース NAPASS に実装した RMI および HORB の動作環境 (VM による違い、CPU パワーの違い) における影響を調査した。

サーバが NT 計算機で、VM jview とした場合、転送配列数の少ない領域では HORB の転送速度が勝り、転送データの多い領域では RMI の転送速度が勝る。また、VM による違い、計算機の CPU の違いにより差異が発生することも判明した。 分散環境におけるシステムを構築する場合は転送データ (配列数)、計算機環境、および VM のバランスを考えながら設計する必要がある。

謝辞

ご助言を賜った三菱電機株式会社 情報技術総合研究所 桜田 博氏、三菱スペース・ソフトウエア株式会社 渡辺 克明氏、古瀬 慶博氏に感謝致します。

引用文献

1) Sun Microsystems : The Java language specification, (1996)
2) 平野 聡 : 分散 Java 実行系 HORB の基本性能の評価, 情報処理学会 システムソフトウェアとオペレーティング・システム, 76-12, pp.67～72, (1997)
3) Sun Microsystems : Remote Method Invocation Specification, (1996)
4) 南石 晃明, 牧野 和佳, 上田 正和, 下村 道彦, 平藤 雅之 : Java 言語による対話的グラフ表示機能を備えた WWW 青果物市況情報データベース NAPASS for Web on Java の開発, 農業情報研究, 6 (1), pp.27～42, (1997)
5) 下村 道彦, 本田 茂広 : Napass for web on JAVA-RMI & HORB の事例発表, 農林水産省平成 9 年度農学情報機能部門研修資料, (1997)