2004－EVA－8　(9)
2004／3／19

CORBA を用いたネットワーク対応型医用機器の開発とLatency測定

古瀬慶博[#○]・砂田文宏[#]・秋山朋之[#%]・八木昭彦[$1]・青木英祐[$2]
波多伸彦[$1]・佐久間一郎[$2]

ネットワーク接続を前提とした分散型医療機器システムの開発において，機器間のデータのやりとりをCORBA(ACE/TAO)を用いて実装した．システム結合試験が限られた制約の下，事前試験のデザインを工夫して基本的なLatency の測定を行った．それらの測定結果から，結合試験時の実効的なパフォーマンスを推定することを試みた．

Measurement of Latency for Distributed Surgical Equipment using CORBA

Nobuhiro FURUSE[#], Fumihiro SUNADA[#], Tomoyuki AKIYAMA[#%], Akihiko YAGI[$1], Eisuke AOKI[$2], Nobuhiko HATA[$1], and Ichiro SAKUMA[$2]

In a development of a distributed surgical robots system assuming the network connection, the data communication between equipment comes to a realization using the CORBA. The high concealmentness still exists in a primary developmental stage and it might not easy to move the place to testify a connectivity of the other developmental instrument. However, backyard communication record will be important between those equipment which is carried out through the ACE+TAO Libraries. We try to estimate a total throughput performance in the system based on the measurement of Latency between the local distributed personal computers.

1. はじめに

近年の医工連携による産学協同で分散型の医療機器の研究開発が盛んに行われてきている．これらの機器開発は，ネットワーク接続を前提としており，手術支援ロボットや画像ナビゲーションシステムはその代表といえる．とりわけ，手術支援ロボットにおけるマスター装置やスレーブ装置、画像誘導装置は，サブシステムとして閉じた機能・性能要件を満たす開発であるとともに，様々な目的や複雑な使用状況を想定した組み合わせ自由度の高い開かれたシステムとして運用されることも望まれている．それぞれの装置が異なるプラットフォームで開発されうることを前提に，プラットフォーム間の共通な通信手段として CORBA を利用した実装が行われた [4].

我々は，八木ら [3]によって実装された CORBA の同期型イベントサービスをもとに，分散コールバックのデザインパターンを適用し，装置間の通信データのログを記録する機能を開発した．ログを記録するために，装置間のデータを記録するために多くのイベントが発生する．このような同期型イベントサービスを大量に発生した場合，通信のパフォーマンスは関連する装置の使用状況や相対的な処理能力に大きく依存すると予想される．

本稿では，開発環境と総合接続環境が異なることを前提に，単体での開発段階において装置間のLatency（通信遅延時間）を測定結果について報告する．CORBA の実装にはプラットフォーム依存性はないものの，実際にはプラットフォーム依存の遅延時間の特性があることを明らかにする．さらに，最大遅延時間がもっとも小さいプラットフォームの選択により，総合接続試験の結果について触れる．

#　三菱スペース・ソフトウエア(株)
#%:現在、日本アドバンス・テクノロジー（株）
$1:東京大学大学院情報理工学系研究科
$2:東京大学大学院新領域創成科学研究科

#: Mitsubishi Space Software Co.ltd.
#%: Nippon Advanced Technology Co.ltd.
$1: Graduate school of InformationScience and Technology, The University of Tokyo
$2: Graduate school of Frontier Sciences, The University of Tokyo

2. 方法

2.1 システム概要

本システムの構成を図１に示す．スレーブロボットシステムから CORBA 通信装置を経由してログ記録装置へデータを通信するシステムである．また，この図には記載されていないが，スレーブロボットは上位のマスターロボットシステムと通信可能である．

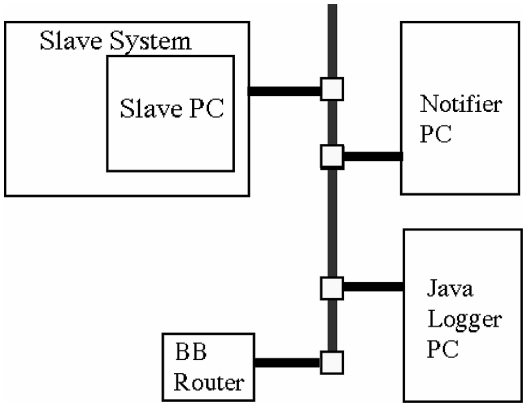

図１ システム構成概要

Fig.1 Target system configuration

2.2 イベントサービスによる通信

スレーブロボットシステムからログ記録装置へのメッセージ配信は，CORBA の Push 型イベントサービスによって実装する(Schmidt)．Push 型イベントサービスのモデルを図２に示す．メッセージは Supplier（スレーブロボット）のメッセージは Notifier を経由して，Consumer であるログ記憶装置へ送られる．ベースとなる CORBA 通信ライブラリには，ワシントン大学 [2]で開発されている ACE+TAO(ACE5.2.1+TAO1.2.1)を利用した．

2.3 システム開発における測定実験の位置づけ

スレーブロボットのシステム開発において，ログ記録装置へのメッセージ配信はシステム外へ配信機能に位置づけられる．CORBA 通信部分の開発および単体試験は独立に行い，システム開発の時間を有効に利用することが通常行われる．また開発の現実の問題として，研究室等で用意されるスレーブのターゲット PC が最新の仕様で用意されるのに対して，開発環境で使用する PC は社内の共有資産であることが多く，性能はターゲット機より劣るケースが想定される．また，ターゲット機の開発段階における可搬性の制約から，開発環境で出来る限り接続時のパフォーマンスを事前評価しておくことが望ましい．加えて，CORBA 実装にはプラットフォーム依存性はないものの，Consumer-Notifier-Supplier に相当する PC のハードウェア性能および OS に依存した通信遅延が発生すると予想される．Consumer-Notifier 間の Latency の測定によって，Consumer-Notifier-Supplier が接続された場合のトータルでの通信遅延時間の推定が可能になることを検証する．

図２ Push 型イベントサービスモデル [1)]

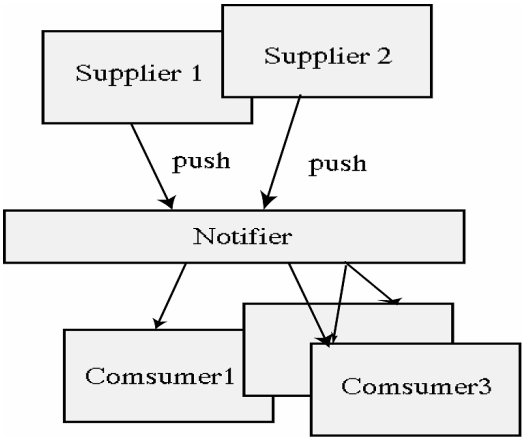

Fig.2 Event Service Model (Consumer, Notifier, and Supplier Model)

3. 測定実験と評価

3.1 試験デザイン

スレーブロボットシステムの PC ハードウェア性能以下の開発用 PC を利用して Latency の測定を実施する．試験の第一目標は，Consumer-Notifier を実装した２台ないし１台の PC で Latency の測定を行うことである（図３）．第二に，最大遅延時間が小さく安定している OS と PC の組み合わせを明らかにすることである．第三に，スレーブロボットシステムを使用して結合試験を行い，予想した遅延時間内でログの記録が可能であることを検証する．

利用した開発 PC の性能と OS を表１に示す．想定するメッセージ通信は，Supplier（スレーブロボットを想定）から Notifier に 188 個の double 型データと４個の long 型整数(IEEE 規格)である．これらを IDL で定義し，ACE+TAO IDL コンパイラを用いて C++のコードとして実装した．上記メッセージセットを Supplier から Notifier あてに投げたときの Supplier 側で捉えた開始から受け取り完了までの時間である．１メッセージセットに対する１回の Latency 時間とし，連続 1000 回のメッセージ送信を行い，１回あたりの最大遅延時間を評価することを試みた．なお、PC の OS は windows/Linux

いずれも工場出荷時またはインストール時の設定の状態で使用し，プロセス実行の nice 値や windows CPU の CAS およびネットワークバッファーサイズについての変更等は行わないものとした．

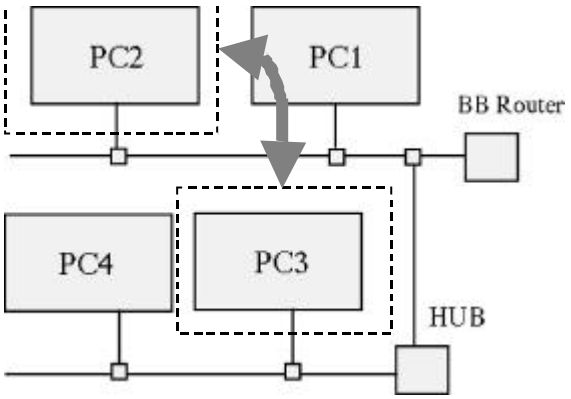

図３ Latency の測定環境（DHCP サブネット下の任意の 2 台または 3 台を使用した）
Fig.3 Metrics of Supplier-Notifier Latency between PCs

表１　開発 PC の種類と OS
Table 1 Hardware Specification of PC and OS

350 PowerEdge Dell	Celeron	850MHz,256MB	Redhat7.3 gcc2.96
260GX Dell	Pentium 4	2.53GHz,512MB	Redhat7.3 gcc2.96 winXP SP1
800r Dell	Pentium 3	797MHz,320MB	Redhat8.0 gcc 3.2
T240 IBM ThinkPad	Celeron	346MHz,192MB	Redhat8.0 gcc 3.2

第一ステップとして，Linux(Redhat)を実装した PC 間での Latency の測定を実施した．その結果をまとめて表２に示す．第二ステップとしては，WinXP および Redhat をデュアルブート可能な PC を利用して，Notifier を winXP に限定した測定をおこなった．

3.2 測定評価および考察

表２から以下の３点が指摘できる．

a) T240(IBM thinkpad)を除いて１台の PC で Suppiler-Notifier を行う場合には，見かけ上中央値は小さくなるが，トータルで平均と中央値の差が大きく現れる．これは遅延時間の部分が二層構造（bi-modal）になるためである．

b) 最大遅延時間が 20msec 以内（最初の１回目の通信を除けば 15msec 以内）のときには、Latency の中央値は 6msec となる．T240(IBM thinkpad)の note PC が Notofier か Supplier のいずれかの場合である．

c) T240(IBM thinkpad)以外の PC（デスクトップまたは 1U ラック）では，最大遅延時間が 300msec を越えるケースが現れる．

以上の結果の範囲から現象としては，T240(IBM thinkpad)最大遅延時間が最小で，中央値と平均との差が小さい（揺らぎか外れ値よる偏りがすくない）ようであることがわかった．2Ghz の 260GX との組み合わせと 800Mhz の 800r との組み合わせのいずれでも同程度である．T240(IBM thinkpad)は 500MHz 以下の低速 CPU である点を考慮すると，OS の違いでなく実装レベルでの製品での違いに関係があるのかもしれない．現時点で原因を絞り込むことは困難である．

表３からは以下の２点が明らかになった．

d) 260GX(winXP)を Notifier にした場合，上記 b)の限界性能を 800r が Supplier となる場合も達成できる(10msec 程度の最大遅延時間)．

e) 記録表示画面を可視状態することは，遅延時間の中央値および最大遅延時間を５倍程度大きくしてしまう．

260GX を Notifier にしたとき，winXP と Redhat7.3 との OS の違いによる遅延時間の挙動を評価することを試みた．260000 回の通信で実験を行った，最大遅延時間の発生状況と形態（外れ値か否か），平均値周辺の揺らぎの性質を見ることが目的である．その結果，winXP の場合，遅延時間が 20msec を越えるのは１万回に１回程度の確率(99.999%)であることがわかった．また最大遅延時間は安定して 20msec 程度に押さえることができる．一方、Redhat7.3 の場合には 100000 回を越える通信の頃から外れ値が 10 倍程度多く発生するようになった．また図４のように平均値周辺の揺らぎには winXP より大きく，連続的に発生しうるような関連性が見受けられた．

表２　Latency の測定結果（max,max w/o, min, ave, median, ave-med, max w/o -med の単位は msec）

Table 2 Result of Latency (msec unit for max,max w/o, min, ave, median, ave-med, max w/o ）

N:Notifier, Slv:Slave Simulation(Supplier), JL:Logger with Java(Consumer), Disp:Java Logger Monitor Display, and "+" means to implement different two function on the same PC

800r (rh8.0)	260GX (winxp)	260GX (rh7.3)	350 (rh7.3)	T240 (rh8.0)	Folder name	iterations	max	max w/o init	min	ave	median	ave-med	max-med
N		Slv			2004_0117_155959T	1000	580.9	580.9	3.8	4.0	4.0	0.1	540.9
N		Slv			2004_0117_160027T	1000	318.5	318.5	3.8	6.0	4.0	2.0	314.5
N		Slv			2004_0117_160045T	1000	269.1	269.1	3.7	6.0	4.0	2.1	265.1
N			Slv		2004_0117_160444T	1000	53.2	53.2	4.3	4.6	4.4	0.1	48.8
N			Slv		2004_0117_160502T	1000	7.7	5.3	4.3	4.5	4.4	0.1	0.9
N			Slv		2004_0117_160520T	1000	236.3	236.3	4.3	6.1	4.4	1.7	231.9
N				Slv	2004_0117_161030T	1000	17.9	10.5	6.1	6.6	6.3	0.3	4.2
N				Slv	2004_0117_161051T	1000	16.5	10.6	6.1	6.6	6.3	0.3	4.3
N				Slv	2004_0117_161114T	1000	15.3	11.0	6.1	6.7	6.4	0.3	4.6
N+Slv					2004_0117_161419T	1000	538.7	538.7	1.1	5.4	1.8	3.6	536.9
N+Slv					2004_0117_161438T	1000	332.5	332.5	1.1	4.9	1.8	3.1	330.7
N+Slv					2004_0117_161452T	1000	341.4	341.4	1.1	4.1	1.8	2.3	339.6
Slv		N			2004_0117_162529T	1000	301.0	301.0	3.9	5.2	4.1	1.1	296.9
Slv		N			2004_0117_162548T	1000	226.4	226.4	4.0	5.4	4.1	1.3	222.3
Slv		N			2004_0117_162608T	1000	235.4	235.4	3.9	5.2	4.1	1.1	231.3
		N+Slv			2004_0117_162841T	1000	581.0	581.0	0.6	5.8	0.8	5.0	580.2
		N+Slv			2004_0117_162901T	1000	585.6	585.6	0.5	6.0	0.8	5.1	584.8
		N+Slv			2004_0117_162921T	1000	586.4	586.4	0.6	6.0	0.8	5.2	585.6
		N	Slv		2004_0117_162908T	1000	682.7	682.7	1.6	5.5	1.7	3.8	681.0
		N	Slv		2004_0117_162928T	1000	493.1	493.1	1.6	5.8	1.7	4.1	491.4
		N	Slv		2004_0117_162948T	1000	312.9	312.9	1.6	5.3	1.7	3.7	311.2
		N		Slv	2004_0117_163239T	1000	18.8	11.0	5.9	6.5	6.1	0.4	4.9
		N		Slv	2004_0117_163302T	1000	18.5	12.6	5.9	6.5	6.1	0.4	6.5
		N		Slv	2004_0117_163322T	1000	14.6	13.7	5.9	6.5	6.1	0.4	6.6
Slv			N		2004_0117_163800T	1000	338.2	338.2	4.4	5.8	4.7	1.3	333.5
Slv			N		2004_0117_163816T	1000	277.4	277.4	4.3	5.9	4.5	1.4	272.9
Slv			N		2004_0117_163829T	1000	371.6	371.6	4.4	6.4	4.5	1.9	367.1
		Slv	N		2004_0117_163944T	1000	447.0	447.0	1.6	5.7	1.7	4.0	445.3
		Slv	N		2004_0117_164002T	1000	390.2	390.2	1.6	5.3	1.7	3.6	387.5
		Slv	N		2004_0117_164018T	1000	308.7	308.7	1.6	4.8	1.7	3.2	307.0
			N+Slv		2004_0117_164000T	1000	356.3	256.3	1.6	4.8	2.0	2.7	254.3
			N+Slv		2004_0117_164016T	1000	325.9	325.9	1.6	4.8	2.0	2.7	323.9
			N+Slv		2004_0117_164035T	1000	330.4	330.4	1.6	4.8	2.0	2.8	328.4
			N	Slv	2004_0117_164330T	1000	25.9	25.9	6.3	8.4	6.5	1.9	19.4
			N	Slv	2004_0117_164357T	1000	413.8	413.8	6.3	10.3	6.5	3.8	407.3
			N	Slv	2004_0117_164431T	1000	23.7	23.7	6.3	8.3	6.5	1.8	17.2
Slv				N	2004_0117_164831T	1000	15.5	12.7	6.0	6.8	6.3	0.5	6.4
Slv				N	2004_0117_164846T	1000	14.4	14.4	6.0	6.9	6.3	0.6	8.1
Slv				N	2004_0117_164858T	1000	14.6	14.0	6.0	6.7	6.3	0.4	7.7
		Slv		N	2004_0117_165011T	1000	29.6	29.6	5.9	6.9	6.3	0.7	23.4
		Slv		N	2004_0117_165031T	1000	12.6	12.6	5.9	6.8	6.2	0.7	6.4
		Slv		N	2004_0117_165051T	1000	34.0	34.0	5.9	6.9	6.3	0.6	27.6
			Slv	N	2004_0117_165015T	1000	13.0	13.0	6.4	7.1	6.5	0.6	6.4
			Slv	N	2004_0117_165040T	1000	15.2	15.2	6.3	7.0	6.5	0.5	8.8
			Slv	N	2004_0117_165100T	1000	11.7	11.7	6.3	7.1	6.5	0.6	5.2
				N+Slv	2004_0117_165339T	1000	12.7	10.8	2.7	4.5	4.3	0.2	6.5
				N+Slv	2004_0117_165402T	1000	18.2	10.8	2.7	4.5	4.3	0.3	6.5
				N+Slv	2004_0117_165418T	1000	16.1	13.9	2.7	4.4	4.3	0.1	9.7

表３　主として winXP Notifier の場合の Latency の測定結果（max,max w/o, min, ave, median, ave-med, max w/o -med の単位は msec）

Table 3 Result of Latency (msec unit for max,max w/o, min, ave, median, ave-med, max w/o)in case of winXP Notifier and/or with Java Logger

N:Notifier, Slv:Slave Simulation(Supplier), JL:Logger with Java(Consumer), Disp:Java Logger Monitor Display, and "+" means to implement different two function on the same PC

800r (rh8.0)	260GX (winxp)	260GX (rh7.3)	350 (rh7.3)	T240 (rh8.0)	Folder name	iterations	max	max w/o init	min	ave	median	ave-med	max-med
Slv	N				2004_0221_145559T	1000	9.8	4.1	4.1	4.3	4.3	0.0	-0.2
Slv	N				2004_0221_145619T	1000	9.7	5.2	4.1	4.3	4.3	0.0	0.9
Slv	N				2004_0221_145748T	1000	10.1	5.3	4.1	4.3	4.3	0.0	0.9
Slv	JL			N	2004_0221_192011T	1000	75.5	75.5	8.6	21.8	17.0	4.8	53.7
Slv	JL			N	2004_0221_192203T	1000	62.8	62.8	8.5	10.2	8.6	1.6	52.6
Slv	JL			N	2004_0221_192250T	1000	13.4	13.4	8.5	8.8	8.6	0.2	4.6
	N			Slv	2004_0222_035557T	1000	14.8	10.8	6.5	6.3	0.2	0.2	4.5
	N			Slv	2004_0222_035640T	1000	15.5	15.5	6.1	6.5	6.4	0.1	9.0
	N			Slv	2004_0222_035823T	1000	17.7	13.0	6.2	6.6	6.3	0.3	6.4
	N+Disp			Slv	2004_0222_040216T	1000	93.2	93.2	6.5	8.5	6.9	1.6	84.7
	N+Disp			Slv	2004_0222_040306T	1000	90.1	90.1	6.5	15.7	7.1	8.6	74.4
	N+Disp			Slv	2004_0222_040343T	1000	87.3	87.3	6.6	16.1	7.2	8.9	71.2
	N+JL			Slv	2004_0222_040645T	1000	90.1	90.1	6.3	12.4	7.2	5.3	77.7
N	JL			Slv	2004_0222_043117T	1000	20.6	20.6	8.6	10.4	8.8	1.6	10.2
N	JL			Slv	2004_0222_043212T	1000	20.2	20.2	8.5	10.3	8.8	1.5	9.9
N	JL			Slv	2004_0222_043300T	1000	21.6	21.6	8.6	10.4	8.8	1.6	11.2

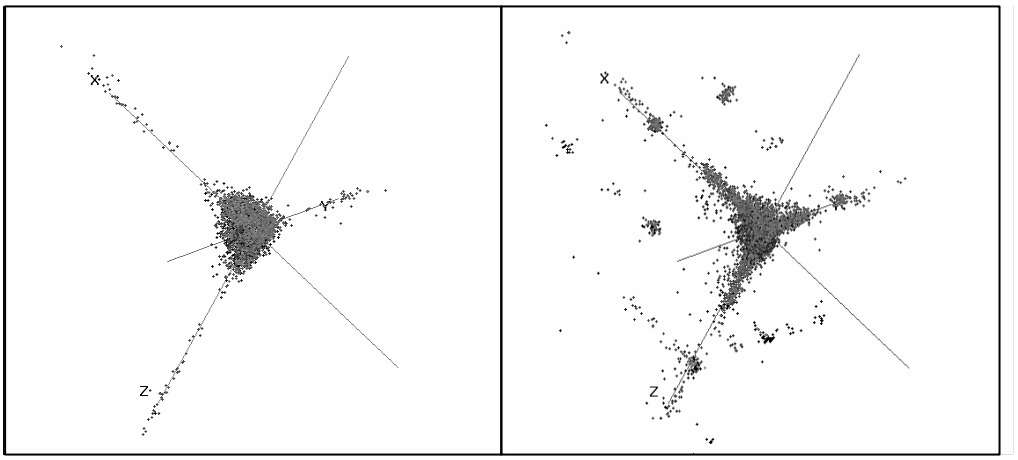

図４ 遅延時間のプロファイルを４次元の埋め込みプロット（左が winXP、右が Redhat7.3 で実装）
左の第一リアプノフ指数＝0.80 右の第一リアプノフ指数＝0.95 の状態で winXP の方がより決定論的状態（準安定）に近いと思われる
Fig.4 Examples of distribution on 4-dimentional embedded space for latency on 260GX (Left: winXP, Right: Redhat 7.3) First Lyapunov exponent on the left diagram is around 0.8 and the right diagrams takes 0.95.

これらの結果から、総合的に Notifier を 260GX(winXP)で実装することで結合試験を実施した.

3.3 統合環境におけるパフォーマンス評価

前節の Latency の測定結果を踏まえて，ターゲットのスレーブシステム(Supplier）を RT-Linux(TIMESYS 社製)、Notifier を winXP，および Java Logger（Consumer）を winXP という構成にて結合試験を実施した.

試験結果の一例を図５に示す．図の連番は Supplier 側のスレーブシステムが指定したタイミングで定期的にデータを Logger 側で受け取り，ハードディスクに記録したものである．したがって realtime モードで Supplierが動作保証されている場合でかつ Logger 側の記録の連番が抜けていない場合には，確実に指定したフレーム周期でデータを受け取っていると解釈される．図から、Supplier 内の socket 通信の送信側を 40msec フレーム、同受信側を 20msec フレーム（つまり Logger 側とこれは同期）させた場合に，800 メッセージの転送

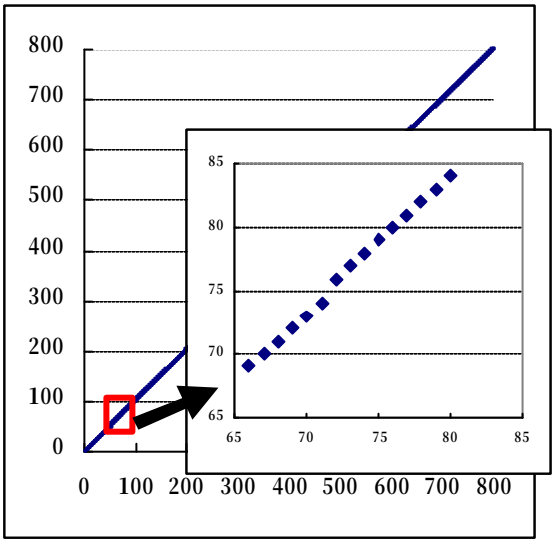

図５ 試験結果（スレーブ 40msec－＞20msec supplier）
Fig.5 Result of Logging Data (slv 40msec->20msec supplier)

において５メッセージが missing となった，800/5=160 メッセージに１回，定期的な mis-catch が発生している．これらの mis-catch は，少なくとも 100msec フレームで転送する場合には消失することが確認できた.

4. まとめ

ネットワーク接続を前提とした分散型医療機器システムの開発において、機器間のデータのやりとりを CORBA(ACE/TAO)を用いて実装した．開発環境と総合接続環境が異なることを前提に，単体での開発段階において装置間の Latency（通信遅

延時間）の測定を実施した．その結果，Push 同期型イベントサービスを利用する場合，長時間の接続において最大遅延時間を小さくするためには，CORBA 通信に介在する PC のハードウェア性能を揃えることが望ましい．大量のイベントが発生する環境下でかつ，Supplier と Consumer(Java Logger)が 2GHz 程度，かつ Notifier が 1GHz 程度の CPU のときには，winXP を Notifier にした場合に最大遅延時間は安定して 20msec 程度に押さえることができる．遅延時間が 20msec を越えるのは１万回に１回程度の確率(99.999%)であることがわかった．長期的な通信（260000 回）から 99.999%の確率で発生する大きな遅延時間は，予備的な統計解析から通常の遅延時間の揺らぎの枠を越えた外れ値であることがわかった．以上の結論は，188 個の double 型データと４個の long 型整数をメッセージ送信した場合であり，また背景トラフィックが少ない理想的な状況での計測である．イベント発生数とイベントあたりのメッセージ量との経験則を明らかにすることが今後必要と思われる．

本研究の一部は日本学術振興会・未来開拓学術研究推進事業「外科領域を中心とするロボティックシステムの開発」によるものである．

参考文献

1)D. C., Schmidt: An Overview of OMG CORBA Event Service,
http://www.cs/wustl.edu/~schmidt/PDF/coss4.pdf
2) ACE および TAO のトップページ
http://www.cs.wustl.edu/~schmidt/ACE.html
http://www.cs.wustl.edu/~schmidt/TAO.html
3) 八木昭彦，橋爪誠，土肥健純，波多伸彦：CORBA を用いた異種環境統合システムの開発及び評価，第 12 会日本コンピュータ外科学会大会・第 13 回コンピュータ支援画像診断学会大会合同論文集, pp.99-100, Dec, 2003.
4) 八木昭彦，大杉伸也，橋爪誠，土肥健純，波多伸彦: 分散オブジェクト技術を用いた外科手術支援ディスプレイシステムの開発，日本コンピュータ外科学会誌, Vol.5, No.2, pp.103-109, 2003.