システムソフトウェアと
オペレーティング・システム 85－10
（２０００．８．４）

インターセプタ機能を用いたCORBAオブジェクトのオンライン入れ替え方式の検討

楊 巍　　横山 和俊　　箱守 聰
（株）NTTデータ　情報科学研究所

概 要

サービスを２４時間継続して提供するためには、サービスを提供しながら、サービスを実現しているアプリケーションプログラムを入れ替えることが必要である。本稿では、CORBAオブジェクトを対象に、サービスの継続を保証したオブジェクトアプリケーションのオンライン入替え方式を提案する。具体的には、CORBAオブジェクトのオンライン入替え契機とオンライン入替えを実現する重要な機能を明確にする。重要な機能は、(1)入替え契機の検出、および、(2)入替え中のサービス要求転送である。また、これらの機能の実現にあたり、CORBA標準機能であるインターセプタを用いた入れ替え方式を提案する。提案する方式では、OSやミドルウェアに改造が不要であり、かつ、オブジェクトアプリケーションへ変更を加えることなく(1)(2)の機能が実現できる。試作と評価により、提案方式を用いることによるオーバヘッドの増加は小さく、サービスの走行に影響を与えないことを示す。

An Approach of Live Replacement for CORBA Objects with Interceptor

Wei Yang, Kazutoshi Yokoyama and Satoshi Hakomori
Laboratory for Information Technology, NTT DATA Corporation

ABSTRACT

Aimed at 24-hour services continuation, it is indispensable to replace programs, which provide services, without service interruption. In this paper, we propose an approach of live replacements for CORBA object. Detection of replacement trigger and client redirection are needed to achieve this goal. We use interceptor, a standard function of CORBA, to solve these problems. Therefore, customization of Operating Systems or communication wares and change of application are not required. Our evaluation results of a prototype show that our approach using interceptor will not cause performance down distinctly.

1. はじめに

近年、インターネットの普及により、株売買や銀行の預金業務などの重要サービスがWWWを通して利用できるようになっている。これらのサービスは、迅速にサービス内容を変更する必要があり、JavaやCORBAに代表されるオブジェクト指向技術を用いたシステム構築が行われている。

重要なサービスを提供する場合、サービスの停止が顧客への不利益を生じるため、２４時間継続したサービスが求められる。２４時間無停止を実現するためには、障害時に高速にオブジェクトによるサービスを再開する機能[1,2,3]に加え、プログラムのバージョンアップなどの保守時にも、サービスを継続することが必要である。本稿では、サービスを提供しながら、サービスを実現しているアプリケーションプログラムを入れ替えること方式について述べる。特に、CORBAオブジェクトを対象に、サービスの継続を保証したオブジェクトアプリケーションのオンライン入替え方式を提案する。

サービスを継続しながら、プログラムを入れ替える方式については、OSの機能を利用してプログラムの一部分を入れ替える研究が行われている。メモリパッチによる方法は、OSの処理より実現されているが、入れ替え対象のアプリケーションプログラムが、入れ替える契機を意識する必要があり、アプリケーション構築時の制約がある。これに対し、文

献[4]によるプログラムの部分入替え方式では、アプリケーションへ制約を与えない OS 処理の手法が提案されている。

オブジェクトプログラムを対象としたものとして、プログラム全体を入れ替える方式が提案されている[5]。この方式では、オブジェクトの冗長構成を利用して入れ替えを実現している。具体的には、古いオブジェクトを含むプログラムと新しいオブジェクトを含むプログラムを両方実行させ、古いオブジェクトへの要求を新しいオブジェクトへも転送することにより、新しいオブジェクトへ処理を移行する。この方式では、入れ替え契機の検出や要求の転送処理をアプリケーションが意識することがないが、マルチキャストを利用した専用の通信ミドルウェアが必要である。また、通信ミドルウェアのオーバヘッドが無視できない。

本稿では、CORBA (Common Object Request Broker Architecture)[6]オブジェクトを対象に、プログラム全体のオンライン入れ替え方式を提案する。提案方式では、CORBA の標準機能であるインターセプタ機能だけを用いて入替えを実現するので、OS 機能や専用の通信ミドルウェアを用いる必要がない。また、入れ替え契機や要求の自動転送処理をアプリケーションが意識することはない。また、提案方式を用いることによるオーバヘッドの増加は小さく、サービスへの影響が小さいことを示す。

2. 研究の背景

2.1. CORBA

CORBA とは、OMG(Object Management Group)によって作成された分散オブジェクトアーキテクチャの仕様である。CORBA の基本構成を図 1 に示す。クライアントは、アクセスしたいオブジェクトへのアクセス情報が格納されている IOR(Interoperable Object Reference)を持つ。クライアントからサーバオブジェクトへのアクセスは、ORB(Object Request Broker)を介して行われ、ORB は、IOR に格納されている情報を元に、該当するオブジェクトへリクエストを渡す。これらの通信は、IIOP(Internet Inter-Orb Protocol)で実現されている。

2.2. システムモデルと要求条件

プログラム全体を対象としたオブジェクトのオンライン入れ替えのシステムモデルを図 2 に示す。クライアントは、古いプログラムでサービスを提供するオブジェクト（旧オブジェクトと呼ぶ）の IOR を保持している。プログラムのオンライン入れ替えは、新しいオブジェクトによるサービス（新オブジェクトと呼ぶ）を起動し、旧オブジェクトによるサービスを新オブジェクトへ移行する。このとき、オンラインで入れ替えるためには、以下の要求条件を満たす必要がある。

[サービス中断の隠蔽]

オブジェクトにより実現されているサービスがオブジェクトの入替え中でも中断されない。具体的には、入れ替え中でもクライアントからのサービスリクエストを必ず受け付け、処理が継続される

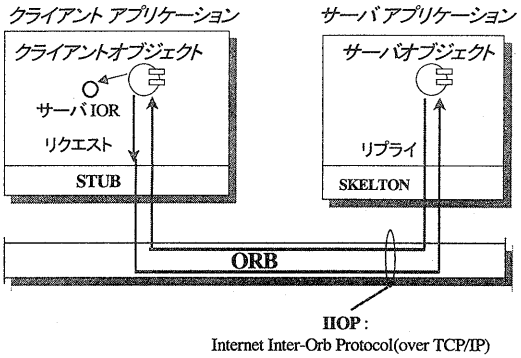

図 1 CORBA の基本構成

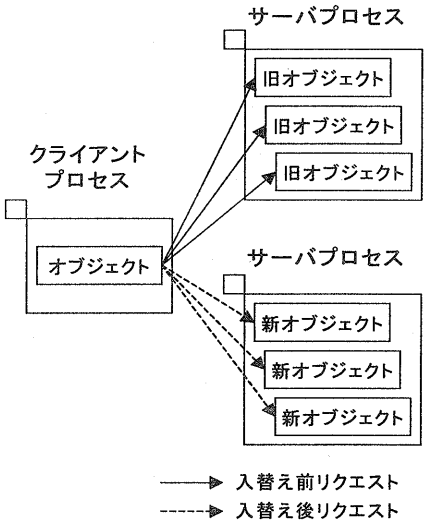

図 2 システムモデル

ことを保証する。

[処理の整合性保証]

オブジェクトの入替え中や入替え後もクライアントからのリクエストを正しく実行できる。具体的には、オブジェクトの内部情報を入れ替え中に引き継ぎ、入替え前と入替え後の処理に矛盾がないことを保証する。

2.3. 課題

上記で述べた要求条件を満たす入替えを実現するためには、以下の課題を解決する必要がある。

(1) オンライン入れ替え処理モデル

処理の整合性を保証するため、旧オブジェクトから新オブジェクトへ処理が移行できる条件を明確にする必要がある。具体的には、入替え中のオブジェクトが取るべき状態遷移と状態遷移中に実行されるべき処理を明らかにする。

(2) オンライン入れ替え処理の実現

(a) 入れ替え契機の検出

旧新オブジェクトの状態遷移を管理し、入れ替え契機を検出する機能が必要である。

(b) リクエスト転送機能

サービス中断を隠蔽するためには、旧オブジェクトへのリクエストを新オブジェクトへ転送機能が必要である。

(c) 内部情報引継ぎ

処理の整合性を保証するためには、旧オブジェクトの内部情報を新オブジェクトへ引き継ぐことが必要である。

これらの処理の実現に当たっては、以下の 2 点を考慮する必要がある。

(1) オブジェクトを実装するアプリケーションが入替えを意識する必要がない。

(2) 入替えを実現する処理のオーバーヘッドが少ない。

3. オンライン入れ替えモデル

3.1. オブジェクトの状態遷移

入替え中の旧新オブジェクトの状態遷移を図 3 に示す。また、それぞれの状態で必要となる処理機能を表 1 に示す。

(1)サービス中(状態 1)

旧オブジェクトがサービスを提供している状態である。システム管理者が新オブジェクトを起動し、状態 2 へ遷移する。

(2)入替え準備完了(状態 2)

新オブジェクトを含む実行モジュールを起動した状態であり、クライアントへのサービスは旧オブジェクトにより提供されている。システム管理者が入れ替え指示を出すことにより、状態 3 へ遷移する。

(3)終了中(状態 3)

旧オブジェクトでのサービス終了待ちの状態である。旧オブジェクトは新規リクエストの受付を中断

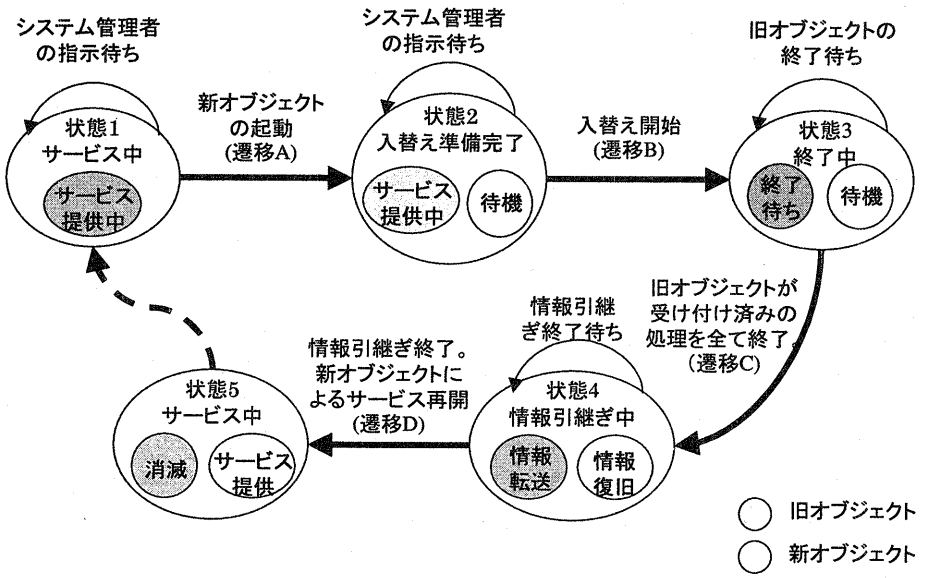

図3 入替え中のサービスを提供するオブジェクトの状態遷移

表 1 状態遷移を実現する機能

状態	入替え処理	状態遷移管理処理	
		遷移先	機能
状態 1	①オブジェクト起動機能 システム管理者が新オブジェクトを起動する機能	状態 2 (遷移 A)	新オブジェクトの検出
状態 2	①入替え指示機能 システム管理者が入替えを指示する機能	状態 3 (遷移 B)	システム管理者の入替え指示の検出
状態 3	① リクエスト転送機能 旧オブジェクトへのリクエストを新オブジェクトへ転送する。新オブジェクトはリクエストを保留する。	状態 4 (遷移 C)	旧オブジェクトの受け付け済みリクエストの終了検出
状態 4	① リクエスト転送機能 ② 情報引継ぎ機能 旧オブジェクトから新オブジェクトへ内部情報を転送する機能	状態 5 (遷移 D)	内部情報の引き継ぎ検出
状態 5 (状態 1)	① オブジェクト削除機能 システム管理者が旧オブジェクトを削除する機能	−	−

し、新規リクエストを新オブジェクトへ転送する。新オブジェクトは処理の整合性を保証するため、リクエストを保留する。

(4)状態引継ぎ中(状態 4)

旧オブジェクトの処理が終了し、引き継ぐべき内部情報が確定した状態である。旧オブジェクトから新オブジェクトへ内部情報が引き継がれる。

(5)サービス中(状態 5)

旧オブジェクトが消滅し、新オブジェクトがサービスを提供している状態である。状態 1 と同じ振る舞いをする。

3.2. 実現方式の検討

オブジェクトオンライン入替えを実現するために必要な機能の実現方式は、以下の 3 つがある。

(a) ORB 方式

CORBA の中核機能である ORB で実現する方式である。

(b)オブジェクトアプリケーション方式

オブジェクトで実現されているアプリケーションプログラム部で実現する方式である。

(c)インターセプタ方式

インターセプタはクライアント側で定義するもの（クライアントインターセプタ）とサーバ側で定義するもの（サーバインターセプタ）がある。図 4 にインターセプタの概要を示す。クライアントインターセプタは、クライアントと ORB の間で動作し、クライアントが ORB にリクエストを渡す直前に呼び出される。サーバインターセプタは、サーバオブジェクトと ORB の間で動作し、ORB がサーバオブジェクトにリクエストを渡す直前に呼び出される。インターセプタでは、リクエストの内容を変更したり、リクエストの内容を解析することができる。インターセプタは、クライアントとサーバオブジェクトにリンクされるライブラリである。この機能は、クライアントとサーバオブジェクトのプログラムを修正することなく利用することができる。

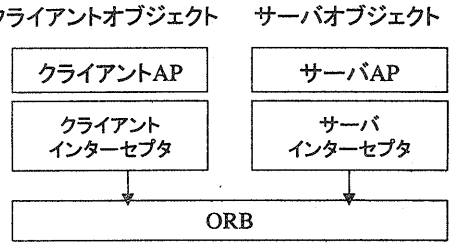

図4 インターセプタの概要

3 つの方式を比較した結果を表 2 に示す。オブジェクトアプリケーションの修正は、ORB 方式とインターセプタ方式が有効である。ORB 方式を用いる場合、オーバーヘッドが最も少ないが、ミドルウェア

表 2 方式の比較

オンライン入替え機能	実現方式		
	ORB	ｵﾌﾞｼﾞｪｸﾄｱﾌﾟﾘｹｰｼｮﾝ	インターセプタ
状態遷移管理	(1) AP の修正が不要 (2) オーバーヘッドが同程度	(1) AP の修正が多 (2) オーバーヘッドは同程度	(1) AP の修正が少 (2) オーバーヘッドは同程度
リクエスト転送	(1) AP の修正が不要 (2) オーバーヘッドが小	(1) AP の修正が多 (2)オーバーヘッドが大。	(1) AP の修正が不要 (2)オーバーヘッドが大
情報引継ぎ	AP 依存のため実現困難	AP 依存のため実現困難	実現が容易

の改造が必要である。本方式では、実現の容易性とオーバヘッドの少なさにより、インターセプタ方式を選択した。また、オブジェクトの情報引継ぎについては、AP プログラム部の処理内容によって修正する内容が変化するため、アプリケーションで実現する必要がある。さらに、表1に示した「新オブジェクトの起動」、「入替え指示」と「旧オブジェクトの削除」」機能はシステム管理者が指示を発行する外部ツールとして必要である。

4. インターセプタを用いた実現方式

4.1. システム構成

図5にインターセプタを用いたオンライン入替えのシステム構成を示す。入替えの対象となるオブジェクトには、サーバインターセプタがリンクされている。また、システム管理者が入替え契機を発生させ、旧新オブジェクトの情報引継ぎを実現するため

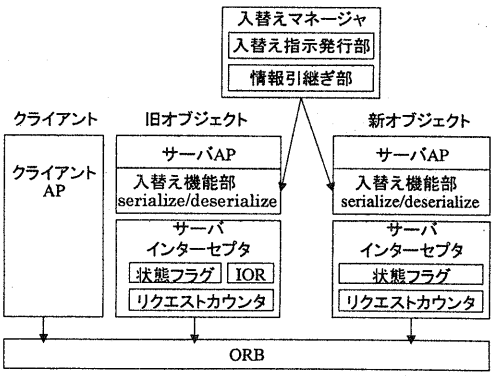

図5 システム構成

に、オブジェクトとシステム管理者間の通信インタフェースとして入替えマネージャを導入する。

4.2. オブジェクトの構造

サーバオブジェクトは、サービスを実行する AP 部と入替え時の状態引継ぎを実現する入替え機能部から構成されるオブジェクトの開発者は、オンライン入替えのために、入替え機能部に以下の2つの機能を実現する必要がある。

(1)serialize

旧オブジェクトでの処理が終了した後に、その状態を読み出すための機能である。

(2)deserialize

旧オブジェクトの状態を入替えマネージャより取得し、新オブジェクトでサービスを再開するための環境を設定するための機能である。

4.3. 状態遷移管理

図6にインターセプタを用いた状態管理と遷移管理の概要を示す。オブジェクトの遷移状態は、サーバインターセプタ中に定義された状態フラグで管理する。オブジェクト状態の遷移イベントは、以下のように実現できる。

[遷移 A]

システム管理者が入替えマネージャを通して、新オブジェクトを含むモジュールを実行することによって新オブジェクトを起動する。

[遷移 B]

システム管理者が入替えマネージャを通して、旧オブジェクトの serialize 機能を呼び出す。CORBA で使用されている IIOP では、リクエストヘッダ中に呼出し先のオブジェクト名、呼ばれ

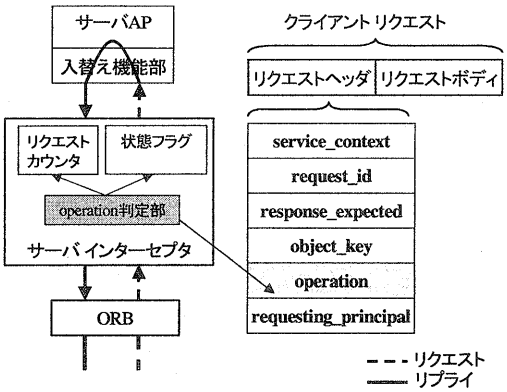

図6 状態遷移管理

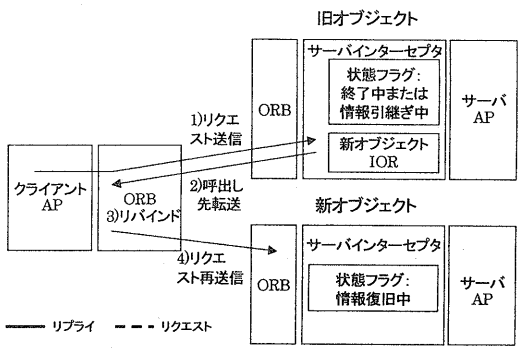

図7リクエスト転送の実現方式

るインタフェース名が含まれている。遷移Bの検出はサーバインターセプタがIIOPのリクエストヘッダ中のoperationが「serialize」であることを検出すればよい。このとき、サーバインターセプタは旧オブジェクトの状態フラグを「終了中」に設定することで状態「終了中」へ遷移する。

[遷移C]

実行中の処理数は、リクエストカウンタで管理する。オブジェクトが「serialize」と「deserialize」以外のリクエストを受け付けたときにインクリメントし、リプライを返却するときにデクリメントすることで現在実行中の処理数を把握することができる。オブジェクト状態が「終了中」にリクエストカウンタが0になることで、遷移Cを検出する。サーバインターセプタは、リクエストカウンタが0になると、オブジェクトのserialize機能へ制御を渡し、「情報引継ぎ中」へ遷移する。

[遷移D]

入替えマネージャが旧オブジェクトの状態を読み出し、新オブジェクトのdeserialize機能を呼び出す。内部情報引継ぎ部からの通知を検出する。前述の遷移Bと同様に、IIOPのリクエストヘッダをサーバインターセプタが解析することで、遷移Dを検出する。

4.4. リクエストの転送

CORBAの場合、オブジェクトへのアクセスは、IORにより制御される。そのため、リクエストの自動転送は、旧オブジェクトのIORを新オブジェクトのIORに自動的に変更する機能により実現できる。

リクエスト転送機能の実現方式を図7に示す。

(1)転送先の通知

サーバインターセプタが旧オブジェクトの状態フラグを参照しながら、全てのリクエストを解析する。オブジェクトの状態フラグが「終了中」または「情報引継ぎ中」の場合、新オブジェクトのIORを含む呼出し先を返却する。

(2)リクエストの転送

呼出し先が返されると、クライアント側のORBがそれを検出し、新オブジェクトに自動的に再接続(リバインド)する。入替え後オブジェクト間に新たなコネクションが成立し、クライアント側のORBが旧オブジェクトへのリクエストを新オブジェクトに再送する。クライアントには再接続処理が隠蔽されるため、オブジェクトが入れ替わったことを意識することがない。

(3)リクエストの保留

新オブジェクトは、旧オブジェクトが終了し、内部情報を引き継ぐまで、クライアントからのリクエストを一時的に保留する必要がある。新オブジェクトのサーバインターセプタは、状態フラグを参照し、「情報引継ぎ中」であれば、インターセプタ内でリクエストを滞留させ、オブジェクトへリクエストを渡さない。

5. 実装と評価

5.1. 実装

本手法の有効性を検証するために、CORBAの製品の1つであるVisiBrokerを用いて実装した。実現

した環境を表3に示す。

表3　プロトタイプの実装環境

項目	内容
ハード	Sun Enterprise450 (CPU:Ultra II 400MHz x 4 メインメモリ: 512MB NW: FastEther)
OS	Solaris2.6
CORBA	VisiBroker for Java3.4
言語	Java(JDK1.2)

実装時には、以下の点について留意した。

(1)サーバインターセプタ

サーバインターセプタはサーバ側とクライアント間のコネクション数分生成される。オブジェクトのオンライン入替え時に、生成された複数のサーバインターセプタ内でプログラム全体の状態フラグとリクエスト数を確認しながらオブジェクトの状態遷移を行う。従って、オブジェクトの状態フラグとリクエストカウンタをサーバインターセプタ間で共有する必要がある。また、複数のサーバインターセプタからの同時アクセスでデータの整合性が破壊されないよう、状態フラグとリクエストカウンタを操作するためのメソッドに排他制御を持たせた。

(2)リクエストの自動転送

今回の実装で用いたCORBA製品では、リクエスト転送機能におけるリバインドがクライアント とサーバ間のコネクションが切断された時のみ動作する。そのため、旧オブジェクトでの情報転送を終了した後に、クライアントとのセッションを切断することでORBによるリバインドを動作させている。

(3)内部情報の引継ぎ

serialize機能はオブジェクトの内部情報を読み出し、入替えマネージャに送信する。読み出しは、オブジェクトの内部状態をバイト列に変換するシリアライゼーション機能を利用することで実現できる。一方、deserialize機能では、送信された内部情報(バイト列)をアプリケーションにより復旧する。

5.2.　評価結果

ここでは、入替えを実現するインターセプタを導入した場合のオーバーヘッドと入れ替え時間を測定した。測定に用いたオブジェクトアプリケーションはリクエストを受け付けると、String型の変数を返却するものである。

(1)インターセプタのオーバーヘッド

図8にインターセプタのオーバーヘッドを示す。データのサイズが増加しても、約10msec以下のオーバーヘッドである。一方、IIOPの通信時間は約30msecであり、インターセプタのオーバーヘッドが小さいことがわかる。しかしながら、高トランザクション処理では、数10msecのアプリケーション処理時間が要求され、インターセプタのオーバーヘッドは無視できなくなる。

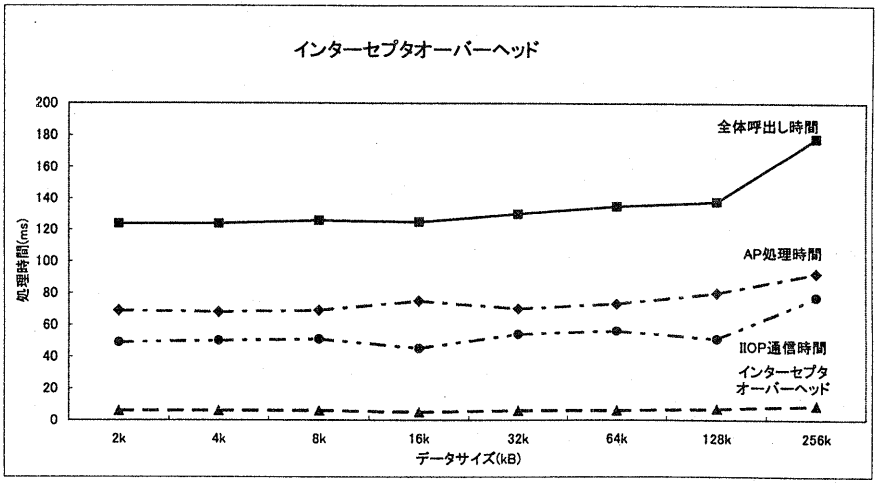

図8 インターセプタのオーバーヘッド

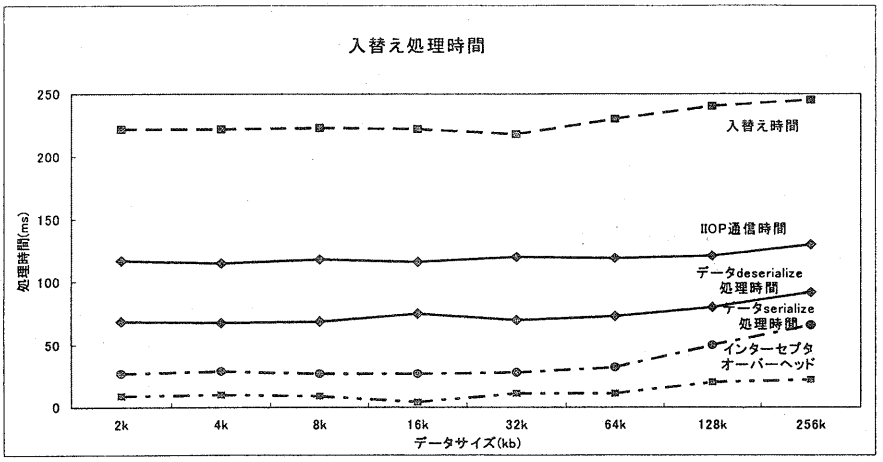

図9 入替え時間の変化

(2)入替え時間

図9にオブジェクト入替え時間を示す。引き継ぐ情報が256kbのとき約180msecで入れ替えが完了する。そのうち、アプリケーションに依存するserializeとdeserializeが40%を占めている。100TPS以上の高トランザクション処理へ適用する場合は、新オブジェクトで十分な数のリクエストを保留するバッファー機能が必要である。

6. 終わりに

本稿では、CORBAオブジェクトを対象に、オブジェクトオンライン入替え手法について述べた。オブジェクトのオンライン入替えを可能にするために、入替え可能な条件を明らかにし、オンライン入替えに必要な機能を示し、インターセプタによる実現方式を提案した。また、本方式の評価により、インターセプタによるオーバーヘッドがアプリケーション処理に影響を与えないことを確認した。

今後は、1つのサーバプロセスに複数のオブジェクト実装といった構成に対して、その中のオブジェクトを部分的に入れ替えるための方法の検討や実現を進める。

参考文献

[1] 白井,ほか,"障害の影響を抑制するトランザクション処理サービスの設計",2000年電子情報通信学会総合大会 '00/03 SD-2

[2] Takemoto, "Implementation and evaluation of fault-tolerant mechanism on network-wide distributed object-oriented systems", Transactions of Information Processing Society of Japan, 41(2), Feb. 2000

[3] P. Narasimhan, L. E. Moser and P. M. Melliar-Smith, "The Interception Approach to Reliable Distributed CORBA Objects", Panel on Reliable Distributed Objects, Third USENIX Conference on Object-Oriented Technologies and Systems, Portland, Oregon (June 1997), pp 245-248.

[4]谷口,ほか,"プロセス走行時におけるプログラムの部分入替え法",電子情報通信学会論文誌 '95/5 Vol.J78-D-I

[5] P. Narasimhan, L. E. Moser and P. M. Melliar-Smith, "Exploiting the Internet Inter-ORB Protocol Interface to Provide CORBA with Fault Tolerance", Third USENIX Conference on Object-Oriented Technologies and Systems, Portland, Oregon (June 1997), pp 81-90.

[6] Object Management Group: The Common Object request Broker Architecture – v2.1, Architecture and Specification