インタフェースに特化したクライアントの動的生成による Web サービス呼び出し処理の高速化

住友千紗　坂田祐司　横山和俊　松田栄之
株式会社ＮＴＴデータ　技術開発本部

1. はじめに

Web サービス環境において、Web サービス利用アプリケーションは、その利用のたびにインタフェースに関する定義情報を取得し、サービスを動的に呼び出すことができる。この実行形式をダイナミックバインディングとよぶ。ダイナミックバインディングを用いることにより、サービス利用者は、処理要件に応じてサービスを呼び出すことが出来る。

現在、ダイナミックバインディングの実現方式として、さまざまな定義情報に対応可能な汎用的なコンポーネントを用い、サービスのオペレーションを呼び出す方式が存在する。この方式においては、汎用コンポーネントが定義情報から呼び出しに必要なオブジェクトを生成する処理に時間がかかり、性能面で問題がある。

本稿では、その生成処理にかかる時間を削減するために、Web サービスを呼び出す際に、呼び出すオペレーションに特化した実行コードを生成、コンパイルし、オペレーションを呼び出す手法を提案する。

2. ダイナミックバインディングの問題点

ダイナミックバインディングは、実行時に、その都度決定されるインタフェースに合わせてサービスを呼び出すために必要とされる技術である。この呼び出し方式を実現するため、A) Web サービスのインタフェースとバインディング情報は、クライアントが理解できる形式で定義され、B)それを用いて、クライアントは、サービスを実行できる仕組みが必要となる。現在、A)については、サービスのインタフェース記述言語である WSDL(Web Services Description Language)が、事実上の標準である。また、B)については、WSDL で記述されたインタフェース定義を読み込み、動的にオペレーションを呼び出す WSIF (Web Services Invocation Framework)がある。

図 1 に WSIF を用いた Web サービス実行クライアントの構成と実行手順を示す。クライアントは、ユーザクライアントとサービスファクトリとサービス処理コンポーネントから構成される。それぞれの役割は以下の通りである。

【サービス処理コンポーネント】WSDL によって定義されるサービス呼び出し処理を実現する部分
【サービスファクトリ】サービス処理コンポーネント生成部分
【ユーザクライアント】サービスファクトリに、サービス処理コンポーネント生成を要求し、受信したサービス処理コンポーネントを用いて Web サービスを呼び出す部分

また、ユーザクライアントの呼び出し手順は、以下の手順となる。

【サービス処理コンポーネント取得】ユーザクライアントが、利用したいサービス処理コンポーネントを保持していない場合、その Web サービスの WSDL 定義をサービスファクトリに送信する。サービスファクトリは、受信した情報に基づいてサービス処理コンポーネントを生成し、ユーザクライアントに返却する
【サービス実行要求】ユーザクライアントは、サービスのオペレーションを呼び出すごとに 1）返却されたサービス処理コンポーネントを用い、Web サービスのオペレーション呼び出しに必要なオブジェクトを生成し、2）Web サービスのオペレーションを呼び出す。1）は、1a）メッセージオブジェクトに設定する値オブジェクトの生成、1b）呼び出すオペレーションに関するオペレーションオブジェクトの生成と 1c）メッセージオブジェクトの生成の 3 つの処理を行う。

WSIF を用いて EJB で公開されている Web サービスを実行する際のサービス実行要求処理に要する時間の割合を図 2 に示す。サービス実行要求処理の 1b）と 1c）で示される処理は、同一オペレーションの場合は、同一オブジェクトが再利用可能であるにも関わらず、呼び出されるたびに実行されるため、同一オペレーションが繰り返し呼び出される場合に、特に性能上のオーバーヘッドとなる。

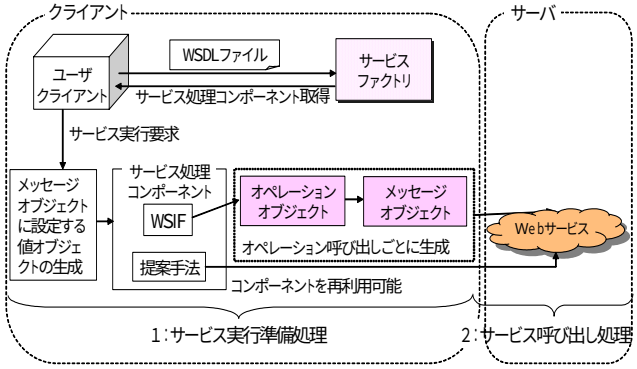

図 1　Web サービス実行処理手順

3. 提案手法

3.1. 事前コンパイル方式

前節で示した問題を解決するため、サービス処理コンポーネントを、汎用的なコンポーネントを用いて実装せず、特定のオペレーションのみに特化

A method for faster web services invocation by dynamically creating a particular interface-specializing client
Chisa Sumitomo, Yuji Sakata, Kazutoshi Yokoyama and Shigeyuki Matsuda
NTT DATA CORPORATION Research and Development Headquarters

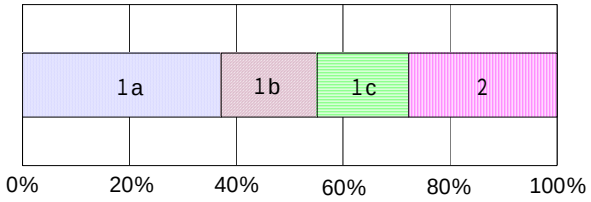

図 2 サービス実行要求の処理時間の割合

されたコンポーネントによって実現する。2 回目以降のオペレーション呼び出しは、このコンポーネントを再利用して実行する。この方式により、オペレーションを呼び出すたびにオブジェクトを生成する必要がなくなるため、WSIF において時間のかかる処理が回避される。

この処理コンポーネント生成処理を実現するため、実行時に与えられる WSDL 定義から、定義されるサービスを呼び出すためのコードを生成し、コンパイルする処理を実装した。

3.2. 具体的な実現方式

図 1 に示すように、クライアントの構成とユーザクライアントの呼び出し手順は、WSIF と同じである。しかし、サービスファクトリにおけるサービス処理コンポーネントの生成手順が異なる。事前コンパイル方式におけるサービス処理コンポーネントの生成手順を図 3 に示す。

(1) 与えられた WSDL ファイルを解析して、Home オブジェクト名や JNDI 名など必要な値を取得する。なお、EJB の定義情報は WSIF において定められた仕様に従っている。

(2) 取得した情報を用い、EJB のリモートオブジェクトを生成する。

(3) 取得した情報と、WSDL に依存しない呼び出しコード部分を記述した Java のファイルから、オペレーションを呼び出す特化コードを生成し、ファイルとして保存する。

(4) ファイルをコンパイルして、特化クラスを生成する。

(5) そのクラスをインスタンス化してユーザクライアントに返却する。

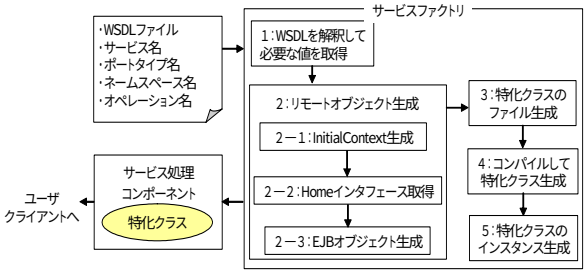

図 3 特化クラス生成処理手順

4. 評価

提案手法の、既存技術に対する有効性を定量的に見積もるために評価実験を行った。

4.1. 実験環境

実験環境は以下に示す通りである。

【クライアント機】
1CPU (PentiumIII 850GHz), 主記憶 512M バイト
【サーバ機】
1CPU (PentiumIV 2.40GHz), 主記憶 1.0G バイト
また、サーバ、クライアントは同一 LAN 内である。

測定プログラムは、引数が一つで処理も行わない、EJB サーバ側のオブジェクトによって実現するサービスのオペレーションを呼び出す。処理時間は、同一オペレーション呼び出しを 10000 回繰り返し、その平均から算出した。

4.2. 実験結果と考察

提案手法と WSIF を用いた場合の処理時間を表 1 に示す。表 1 は、オペレーションの引数を、文字列長 20000 バイトにした場合の処理時間である。なお、本稿では実験結果を載せていないが、オペレーション数やオペレーションの引数の大きさを可変にした場合もほぼ同様の結果が得られている。

表 1 より、以下のことがわかる。

(a) 提案手法のサービス実行要求全体の処理時間は、WSIF の場合と比べて約 50%である。

(b) サービス実行要求処理の 1b) と 1c) に着目すると、提案手法のオペレーションオブジェクト生成とメッセージオブジェクト生成処理を削減した効果がわかる。評価結果では、約 34%の処理時間の削減を達成している。

(c) サービス実行要求処理の 2) に着目すると、サービス呼び出し処理時間が短縮され、全体の処理時間が約 20%削減されたことがわかる。WSIF では、サービス呼び出しをする際に、処理に時間のかかるリフレクションを利用している。一方、提案手法では、サービスを呼び出すための特化クラスを生成している。このため、汎用呼び出しであり、処理オーバーヘッドの大きいリフレクションを利用しなくて済むためである。

以上のことより、提案する方式は、ダイナミックバインディングのサービス実行要求の処理時間の削減に効果があるといえる。

次に提案手法の有効領域について考察する。表 1 からわかるように、提案手法では、サービス処理コンポーネント取得処理は WSIF に対して時間がかかるが、サービス呼び出し処理時間は短縮される。サービス処理コンポーネント取得処理は、特定のオペレーションに対して、一回のみ行われるため、ユーザクライアントが、特定のオペレーションを高頻度に呼び出す場合、提案手法が有効となる。表 1 に示した実験条件での処理時間を例にして考えると、特定オペレーションを 42 回以上実行する場合に提案手法が有効となる。

表 1 処理時間の比較 (ミリ秒) 引数：文字列長 20000 バイトの場合

		WSIF	提案手法
サービス処理コンポーネント取得		70.0	951.0
サービス実行要求	1a)	15.2	16.5
	1b), 1c)	14.1	0
	2)	12.5	4.3

5. 今後の課題

提案手法におけるサービス処理コンポーネント取得処理時間の短縮化があげられる。