軽量 Web サービスアーキテクチャに関する研究

2003MT025 池﨑 崇　2003MT065 永橋 陽一郎

指導教員:青山 幹雄

1. はじめに

現在，普及しつつある Web サービスのメッセージ交換には，主に SOAP が利用されている．しかし，SOAP 構文の生成や解析に多くの時間が必要となるため，性能の低下が問題視されている．

本研究では，上記の問題に対し，メッセージ交換の手段として REST(REpresentational State Transfer) [5]を利用したアーキテクチャを提案する．そして，提案を検証するために，プロトタイプを開発し，スループットや応答時間の比較を行い，REST の有効性を検証する[2].

2. Web サービスの問題

2.1. メッセージ交換における問題

リクエスタ/プロバイダ間における SOAP のメッセージ交換は，SOAP エンベロープに XML 構文を用いる．SOAP エンベロープの構文は，SOAP ヘッダと SOAP ボディによって構成される．

一般的に XML の処理には，多くの時間を必要とする．その主な要因として，XML の生成(シリアライズ)と解析(デシリアライズ)がある．これは，SOAP エンベロープのタグ名や名前空間の重複による冗長性などによって，シリアライズとデシリアライズに多くの時間を必要とすると考えられる．

そのため，SOAP メッセージの処理は，XML 処理に伴う性能低下を招いていると思われる[4].

図 1 に SOAP メッセージの生成から解析の流れを示す．

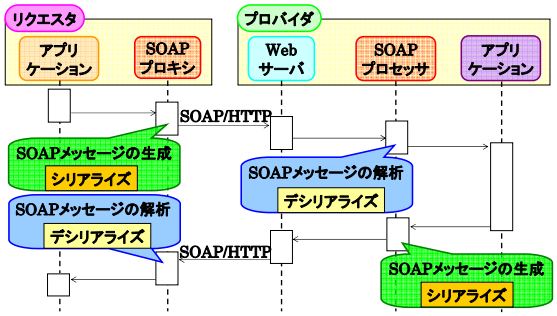


図 1 ： SOAP メッセージの生成から解析の流れ

2.2. 性能問題の関連研究

SOAP メッセージにおける性能問題の関連研究として，プロバイダに 5000 件の顧客詳細レコードを送り，RPC/Encoded とドキュメント/リテラルの 2 つの SOAP 形式による処理速度の違いに関する検証を行っている[1]．2 つの SOAP 形式において，プロバイダ側の XML シリアライズ処理時間は総処理時間の 10.7%，XML デシリアライズ処理時間は総処理時間の 30.7%という検証結果から，SOAP 構文の生成や解析はメッセージ交換のボトルネックとなっている．

2.3. 性能低下を招くメッセージ交換

現在，Web サービスにおけるメッセージ交換の多くは SOAP が利用されている．しかし，以下のような場合，SOAP メッセージ交換による性能低下が予想される．

(1) 類似したメッセージを複数回交換
　SOAP ヘッダなどに冗長な情報が含まれる．そのため SOAP メッセージの生成や解析に余分な時間が必要となり，メッセージ交換の性能が低下する．

(2) 少量の情報を含むメッセージ交換
　SOAP エンベロープには，SOAP 構文に関するデータが余分に付加される．これにより，データ量が増加し，通信の負荷が余分にかかるため，メッセージ交換の効率低下を招く．

近年，少量の類似した情報を複数回交換する Web サービスが増加傾向にあるため，上記の状況は頻繁に発生すると考えられる．

3. 問題に対するアプローチの提案

3.1. REST による Web サービスアーキテクチャ

本研究では，SOAP メッセージ交換において性能低下が予想される場合に，REST の概念と制約に基づくメッセージ技術と REST の制約を適用したアーキテクチャを提案する．特に図 2 に示すように，REST の概念と制約に基づくメッセージ技術の適用を中心に，SOAP メッセージと REST メッセージを比較し，性能問題に対する改善を図る．

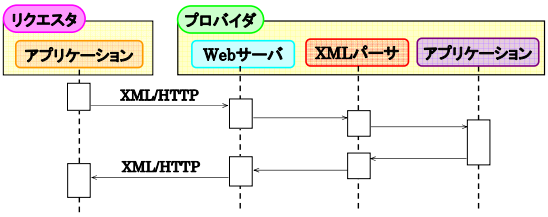


図 2 ： REST のメッセージアーキテクチャ

3.2. REST のメッセージ技術の適用

REST のメッセージ技術は SOAP と異なり，標準化されていないため，REST のメッセージ交換には様々な方式がある．一般的に REST のメッセージ交換は，HTTP 上で XML 文書をそのまま交換する方式である．

REST のメッセージ交換は，SOAP エンベロープのような構文がないため，図 1 に示す SOAP メッセージ構文の生成や解析を行う必要がない．そのため，少量の情報を送信する Web サービスのメッセージ交換に REST メッセージ技術を適用することは，性能向上に有効であると思われる．

3.3. REST の制約の適用

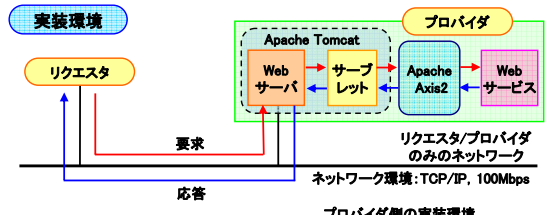
REST の制約の 1 つであるキャッシュは，HTTP GET リクエストに対してのみ有効であり，Web サービスにキャッシュを適用することは，性能向上に有効である．

しかし，SOAP メッセージは，HTTP リクエストに POST を利用し，サービスに対するターゲット URL が SOAP プロセッサである[3]．この点から，SOAP による Web サービスにキャッシュを効果的に適用することは困難である．

そこで性能向上のために，効果的なキャッシュが利用可能な REST の Web サービスアーキテクチャを適用する．

4. プロトタイプの開発

REST メッセージ技術を適用したアーキテクチャの性能を検証するために，SOAP メッセージ交換による性能低下が予想される，少量の情報を繰り返しメッセージ交換する場合を想定したプロトタイプを開発した．開発したプロトタイプは，文字列を返す Echo サービスと 2 つの値の和を求める Calc サービスである．この 2 つのプロトタイプを用いて，SOAP メッセージ交換と REST メッセージ交換の性能を比較した．検証を行った実装環境を，図 3 に示す．



リクエスタ側の実装環境

実行環境	Java 2 SE 5.0
OS	WindowsXP Professional SP2
メモリ	256MB
CPU	Mobile Intel Celeron(TM) 1.2GHz

プロバイダ側の実装環境

SOAPサーバ	Apache Axis2 1.0
Webサーバ	Apache Tomcat 5.5.20
実行環境	Java 2 SE 5.0
OS	WindowsXP Professional SP2
メモリ	768MB
CPU	Intel Pentium(R) 4 2.00GHz

図 3 ： 実装環境

開発した 2 つのプロトタイプは，SOAP サーバの Apache Axis2 に配置し，検証を行った．Apache Axis2 では，受信したメッセージが SOAP メッセージか REST メッセージかを判断し，それぞれのメッセージに関する処理を行う．

5. 検証と評価

5.1. 評価方法

プロトタイプの検証結果を評価するには，様々な方法がある．本研究では，リクエスタ側の応答時間と，プロバイダ側の要求処理時間，スループットの 3 つを測定し，評価を行う．図 4 に本研究で用いる評価方法を示す．

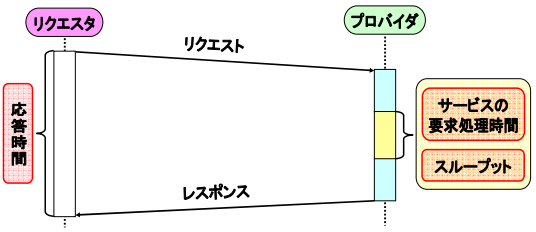


図 4 ： 評価方法

(1) 応答時間

リクエスタがリクエストを送信し，プロバイダがリクエストに対する処理を行い，その処理結果をレスポンスとして返信する一連の処理に要する時間である．

(2) スループット

単位時間あたりに行う要求処理数である．

(3) 要求処理時間

プロバイダが受け取ったリクエストに対する処理を行う時間である．要求処理時間はプロバイダにおけるスループットと反比例の関係にある．

5.2. 検証結果

5.2.1. Echo サービスの検証結果

Echo サービスによる検証は，リクエスタがプロバイダに対して，文字列データ(10 バイト)を送信し，受信したデータをそのままレスポンスとして返信する時の図 4 に示す 3 つの評価方法を測定する．Echo サービスは，1 回の処理時間が短いため実行回数を増加させて測定し，SOAP メッセージと REST メッセージの処理の比較を行う．SOAP，REST による 1 回毎の応答時間の推移を，図 5，6 に示す．

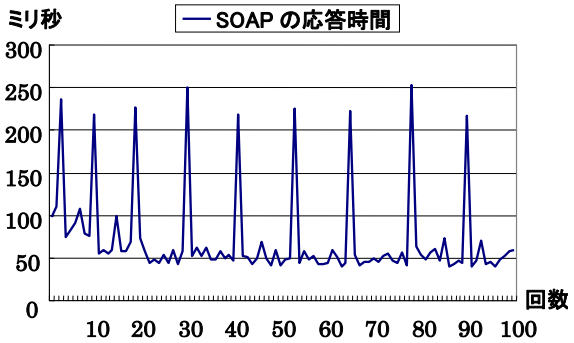


図 5 ： SOAP による 1 回毎の応答時間の推移

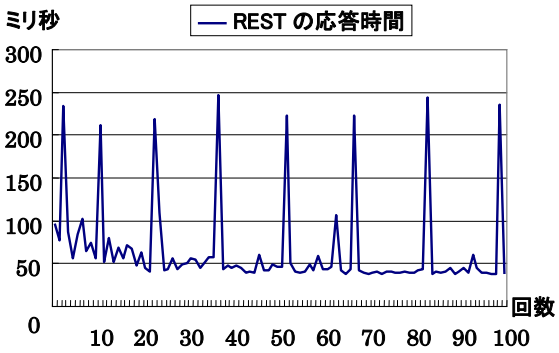


図 6 ： REST による 1 回毎の応答時間の推移

図 5, 6 より, ほぼ一定周期で異常な遅延を含む. これは Java におけるガベージコレクションと推定できる.

そこで, 150 ミリ秒以上の特異値である応答時間を除いた平均応答時間を補正値とし, SOAP, REST の補正値による平均応答時間の値, 応答時間の差[ミリ秒]の値, 応答時間の差[比率]の値を表1に, また応答時間の差[比率]を図7に示す. 応答時間の差[比率]は, 次式で表される.

$$\text{応答時間の差[比率]} = \frac{SOAP\text{の応答時間} - REST\text{の応答時間}}{SOAP\text{の応答時間}}$$

表 1 ： 補正値による平均応答時間とその差

実行回数	1000	5000	10000
SOAP[ミリ秒]	45.0	40.2	39.6
REST[ミリ秒]	38.2	34.0	33.4
差[ミリ秒]	6.8	6.2	6.2
差[比率]	0.151	0.154	0.157

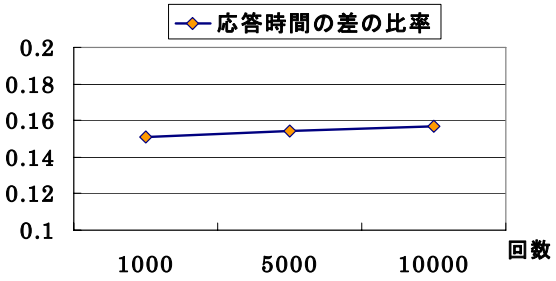


図 7 ： 応答時間の差[比率]

表 1, 図 7 より応答時間の差の比率は, ほぼ一定となっている. この結果から, SOAP と REST の応答時間の差は実行回数に依存せず, REST の応答時間は SOAP の応答時間に対して, 約 15%短縮されている.

また, Echo サービスの 1 秒あたりの要求処理数であるスループットを比較した結果を図 8 に示す.

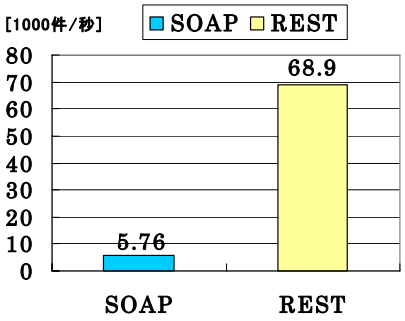


図 8 ： スループット

図 8 より, REST は SOAP のスループットの約 12 倍という高い数値となった. この結果から, SOAP のメッセージ交換よりも REST のメッセージ交換が効率的に要求に対する処理が行われ, プロバイダの負担も軽減されている.

また, 送信するデータ量の増加による SOAP メッセージと REST メッセージの応答時間の変化を検証した. 変化の検証には, 図 9 に示すデータ量を用いてそれぞれ 100 回実行し, 1 回の平均応答時間を比較した.

図 9 にデータ量による応答時間を示し, データ量による応答時間の差[比率]を図 10 に示す.

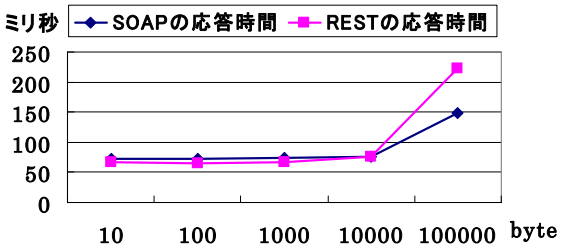


図 9 ： データ量による応答時間

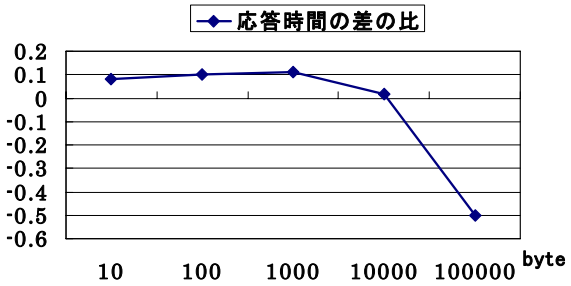


図 10 ： データ量による応答時間の差[比率]

図 9, 10 より, 少量のデータのメッセージ交換には, REST メッセージの応答時間が短いため, 効率的にメッセージ交換が行える. しかし, データ量が増加するにつれて, SOAP の方が効率的にメッセージ交換を行える.

5.2.2. Calc サービスの検証結果

Calc サービスによる検証は, リクエスタが, プロバイダに 2 つの値を送信し, その和を求めて結果を返す. この処理に対して, 図 4 に示す 3 つの評価方法を測定する. Calc

サービスも実行回数を増加させて測定し，SOAP メッセージとRESTメッセージの比較を行う．また，Echo サービスと同様に，SOAPとRESTのガベージコレクションによる特異値を除いた補正値による平均応答時間とその差[ミリ秒]の値，応答時間の差[比率]の値を表 2 に，応答時間の差[比率]を図 11 に示す．

表 2 ： 補正値による平均応答時間とその差

実行回数	1000	5000	10000
SOAP[ミリ秒]	42.8	39.8	38.7
REST[ミリ秒]	37	34.5	33.6
差[ミリ秒]	5.8	5.3	5.1
差[比率]	0.136	0.133	0.132

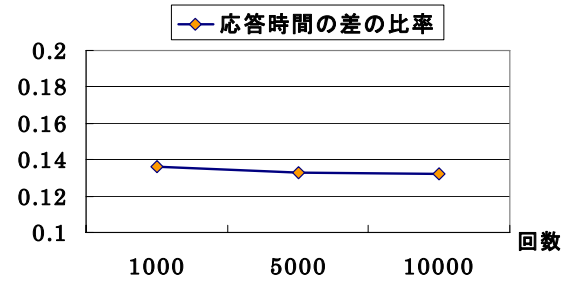


図 11 ： 応答時間の差[比率]

表2，図11より，Echoサービスと同様に応答時間の差の比率がほぼ一定になっている．この結果から，応答時間の差は実行回数に依存せず，RESTの応答時間はSOAPの応答時間に対して，約 13%短縮されている．また，Calc サービスのスループットの比較を図 12 に示す．

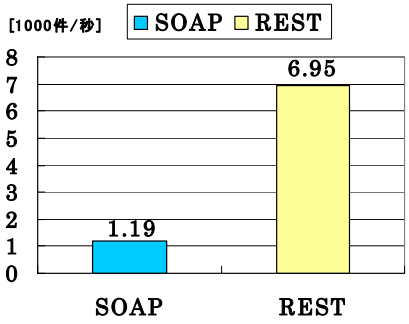


図 12 ： スループット

図 12 より，Echo サービスと同様に，REST のスループットはSOAPの約6倍という高い数値となった．この結果より，REST のメッセージ交換の方が効率的に処理を行える．

6. 考察と今後の課題

Echo サービスと Calc サービスによる検証で，REST は SOAP よりも応答時間が短く，スループットも高い数値となった．これにより，軽量なWebサービスに対して，RESTのメッセージ交換は，性能向上に有効である．

しかし，データ量の変化による検証結果から，データ量が増加すると，REST よりも SOAP の応答時間が短くなる．この結果から REST のメッセージ交換は，データ量が多い場合，効率的ではない．そのため，状況に応じて SOAP メッセージと REST メッセージを使い分ける必要がある．

今後は以下の課題を検討する必要がある．

(1) REST のメッセージ技術

XML を用いずにメッセージ交換する REST メッセージ技術について考察し，一層の性能向上に繋げる．

(2) Web サービスキャッシュの適用による検証

性能の向上を図る上で，Webサービスアーキテクチャにキャッシュを適用することは有効である．一層の性能向上を図るために状況に応じてキャッシュを適用し，性能の向上を検証する必要がある．

7. まとめ

本研究では，SOAP のメッセージ交換における性能の問題を解決する方法として，REST を用いた Web サービスアーキテクチャを提案した．

SOAP のメッセージ交換が性能低下を招く場合において，Echo サービスと 2 つの値の和を求める Calc サービスのプロトタイプを開発し，SOAP メッセージとREST メッセージを 3 つの評価方法を用いて性能を比較し，REST の有効性を検証した．

参考文献

[1] A. Ng, S. Chen, and P. Greenfield, An Evaluation of Contemporary Commercial SOAP Implementations, Proc. of the 5th Australasian Workshop on Software and System Architecture, 2004, pp.64-71.

[2] 池﨑 崇 ほか, REST を用いた軽量 Web サービスアーキテクチャの提案と評価, 情報処理学会第 69 回全国大会論文集, No. 3T-6. Mar. 2007(発表予定)

[3] K. Janowicz, C. Riedenmann, and M. Wilde, Technology Watch Report 1, BRIDGE-IT Project, 2002, http://www.Bridge-it.info/download/reports/BRIDGE_IT_RE_7200_004_UM_Technology_Watch_Report_1.pdf.

[4] 松村 郁生, コンテキスト依存な通信による Web サービスの高速化, 特別研究報告書, 2006, http://www.ai.soc.i.kyoto-u.ac.jp/publications/thesis/BH17_matumura.pdf.

[5] R. T. Fielding, Architectural Styles and the Design of Network-based Software Architectures, PhD Dissertation, 2000, http://www.ics.uci.edu/~fielding/pubs/dissertation/top.htm.