5A-4

大量データ連携時のＥＳＢの性能向上に関する考察

○田中 覚† 山足 光義† 塚本 良太† 米田 貴雄†
三菱電機株式会社 情報技術総合研究所†

１．はじめに

今日のシステム開発は、企業間あるいは企業内におけるシステムの統合・連携を迅速に行うことが要求されており、システム開発の効率化が非常に重要となっている。そこで、標準化された手順によって外部から呼び出すことができるサービスを組合せてシステム構築を行う SOA（Service Oriented Architecture）が普及してきている[1]。既存システムの機能をサービス化することにより再利用し、サービス連携の基盤として ESB（Enterprise Service Bus）を用いることにより、信頼性のあるシステム連携を効率よく実現することが可能となる。

しかしながら、ESB はサービス間で連携するデータをメッセージにより運んでいるため、大量のデータを転送するような場合に性能が上がらないという課題がある。

この課題を解決するためにメッセージとは別のデータバスを使用し、データをストリームで処理することで性能向上させることを検討した。本稿では、この ESB で大量データを使用するときの性能向上の方式を提案する。

２． 従来の課題

ESB では、サービス連携を実現するために、サービス間でメッセージを送受信している。従来の ESB では、そのメッセージの中にサービス間で連携するデータを格納していた。

例えば 2 つのサービスを連携させる場合、連携元サービスが終了すると、連携元サービスから取得された連携データと制御情報とを格納したメッセージを作成する。そのメッセージを連携先サービスに渡すことで連携先サービスを実行する。このようにしてサービス間でデータを運び、サービス連携を実現している。

しかし、従来の方式ではサービス間で連携するデータが大量になった場合に、以下のような 2 つの課題があった。

一つ目の課題は、メモリの消費量である。サービス間をつなぐメッセージの実体はメモリ上に確保された領域であるため、サービス間で連携するデータが大量になった場合に、メモリの消費量が増大する。そのため、メモリが豊富に使用できる環境で無ければ、一度の連携で使用できるデータ量や、同時に実行できるサービス連携の数などに制限が発生するという課題があった。

二つ目の課題は、処理時間である。サービス連携において、連携元サービスからの連携データの取得と連携先サービスへの送信とは逐次的に実行されるため、処理全体にかかる時間には、単純にそれぞれの処理時間が加算される。そのため、データ量が多く、複数のサービスを連携させるような場合には、処理時間が長くなりやすいという課題があった。

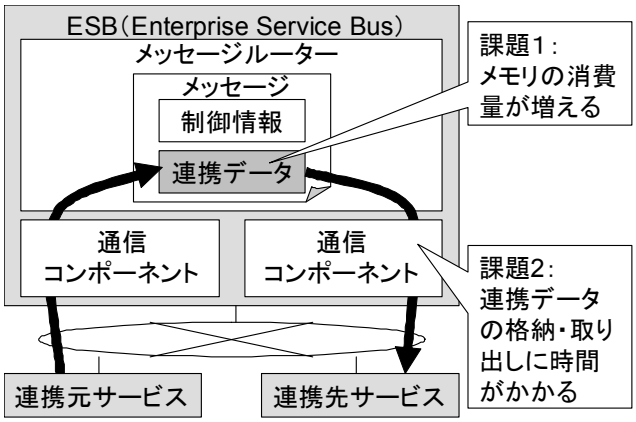

図 1：従来の ESB の課題

3. ストリームを使用したデータ連携

大量のデータを処理する方式としては、Unix のパイプのようにデータを一箇所に溜め込まずにストリームとして処理する方式と、ファイルのような領域に一時的にデータを蓄積し、蓄積されたデータを処理するという方式が知られている。

従来のメッセージに連携データを蓄積して運ぶ方式は、後者の方式として捉えることができる。しかし、前述のように性能的な課題が発生する。そこで、前者のようにデータをストリー

Improved ESB performance when large amounts of data linkage
† Satoru TANAKA, Mitsuyoshi YAMATARI, Ryota TSUKAMOTO, Takao YONETA,
Information Technology R&D Center, Mitsubishi Electric

ムとして処理することで、ESB の性能改善を提案する。

提案方式のソフトウェアの構成図を図 2 に示す。提案方式では、従来方式に連携データを運ぶためのデータバスとデータバスに対しデータを入出力するためのエンコーダとデコーダを追加する。

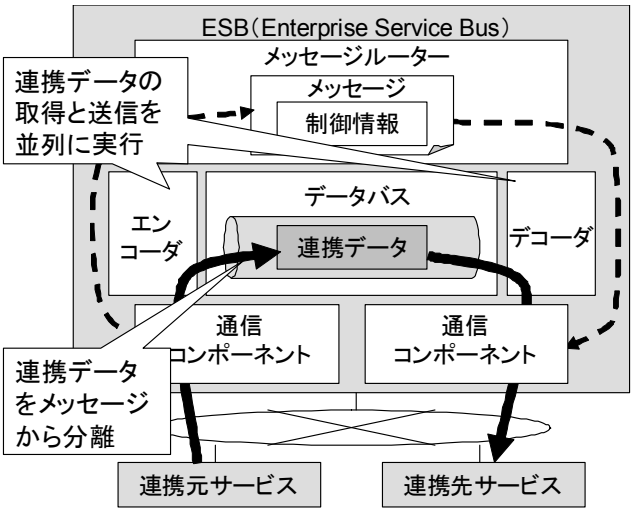

図 2：提案方式 ソフトウェア構成

前述のように従来方式では、連携元サービスから取得したデータを使用して、制御情報と連携データを格納したメッセージを作成する。一方、提案方式では、連携元サービスの処理が開始された時点で、連携先サービスを実行するための制御情報を格納したメッセージを作成する。そして、連携先サービスに送信する。

次に、エンコーダを起動して連携元サービスから連携データを取得し、取得したデータをデータバスに格納する。また、メッセージを受け取った連携先サービス側では、デコーダを起動して、データバスから連携データを取得し、連携先サービスへ送信する。ここで、エンコーダとデコーダに対して独立したプロセスやスレッドを割り当てることで、連携元サービスからのデータ取得と連携先サービスへのデータ送信が並列に実行される。

上記のようなデータをメッセージとは別のデータバスにより運ぶ提案方式では、サービス間の連携データをメッセージ上に蓄積する必要が無くなり、メモリの消費を抑えることができる。

また、連携元サービスからのデータ取得と連携先サービスへのデータ送信が並列に行われるようになり、処理時間を短縮することができる。特に近年のように CPU がマルチコア化し、並列処理が効率的に行われるような環境では、サービス連携全体の処理時間短縮の効果が大きくなることが期待される。

・複数サービスでの連携データの使用

前述のソフトウェア構成では、連携データを使用する連携先サービスが 1 つであるが、処理によっては複数のサービスで連携データを使用する場合が考えられる。この場合は、エンコーダにて同じデータを複数のデータバスに書き込むことで対応する。

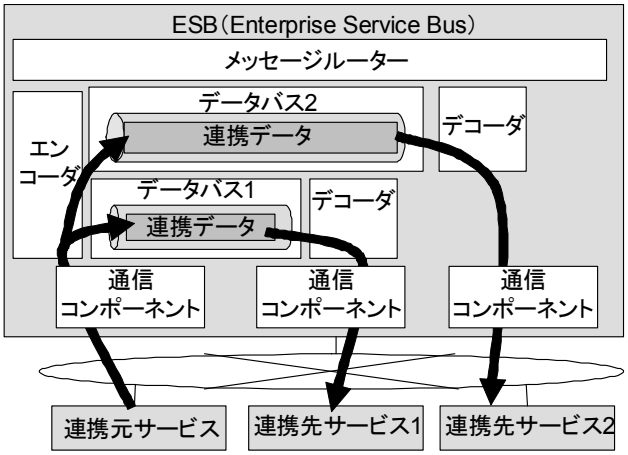

図 3：エンコーダによる連携データの複製

提案方式では、連携するサービス数が増加した場合、それぞれのサービスとのデータ入出力処理が並列に実行されるため、従来方式と比べて処理時間の短縮効果が大きくなることが期待される。

4. まとめ

本稿では、サービス間で大量データを連携するときの ESB の性能改善の方式として、サービス間の連携データをサービス間連携を行うためのメッセージから分離し、ストリームで処理する方式について提案した。本方式では、データをストリームとして処理することで、メモリ消費量の抑制と処理時間を削減することが可能である。今後は本稿で述べた技術を実装して検証を行い、具体的な効果を評価する予定である。

参考文献

[1] 野村総合研究所 技術調査部,「IT ロードマップ 2011 年版」,東洋経済新報社 (2011)

[2] Ron Ten-Hove, Peter Walker,「Java Business Integration (JBI) 1.0」, http://download.oracle.com/otndocs/jcp/jbi-1.0-fr-oth-JSpec/ (2005)