O-046

ESB ログからのサービス障害検出の実装と性能に関する考察
Implementation and Performance Evaluation of Service Error Detection from Logs of Enterprise Service Bus

竹田 義聡† 菅野 幹人† 郡 光則†
Yoshisato TAKEDA† Mikihito KANNO† Mitsunori KORI†

1. はじめに

我々は、SOA (Service Oriented Architecture) に基づき構築した業務システムを対象とし、運用上の異常監視を実現するサービス運用管理システムを開発している。このシステムにおいて、今回、サービス内部の機能や SOA の基盤ミドルウェアである ESB (Enterprise Service Bus) の提供する監視機能を基本的に使わず、ログを監視・解析しサービスの障害を検出する方法を検討し、プロトタイプ処理系を実現した。この方法は、既存のサービスを改修せずに運用管理を実現できる、社外のサービスや他社サービスも監視できる、などの特長を持つ。これらは特に、企業や組織の合併や提携などにおいてサービスシステム同士を相互利用する際に有効であると考えられる（図 1）。

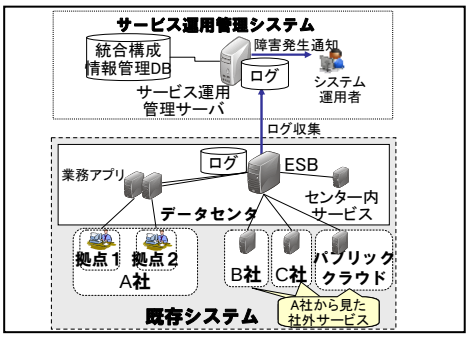

図 1 サービスシステム同士の相互利用

本論文では、障害検出方法および実装について報告する。また、プロトタイプ処理系の評価を元にサービス障害検出性能に関して考察した結果、実用的なスループットで実際の業務サービスシステムを監視できる見込みを得た。この評価および考察について述べる。

2. 関連研究

文献[1]は、企業合併などで必要となる複数の ESB を含むサービスシステムにおいて、今回と同様にサービスそのものや ESB を改修せずにサービスインスタンスを識別したりプロセス稼動状況を監視したりする方法を提案している。本研究は、この研究の発展系にあたり、単一のサービスの実行結果や、複数のサービス同士の実行順序・タイミングの不正に踏み込んだ障害検出方法を提案し、評価するものである。

文献[2]は、SOA に対応していないレガシーなシステムの画面をラッピングしサービス化したシステムにおいて、エラー検出と影響範囲の把握を可能にする方法を提案している。本研究は、SOA に基づき設計され、BPEL(Business Process Execution Language) でプロセス定義されたサービスシステムを対象としており、文献[2]と補完関係にある。

† 三菱電機株式会社　情報技術総合研究所 Information Technology R&D Center, Mitsubishi Electric Corporation

3. サービスからの障害検出

3.1 想定するサービス障害と、ログ内情報

ESB のログに記録される情報をもとに、検出可能な業務サービス（以下「サービス」と略記）の障害について検討し、抽出した（この詳細については別途発表予定）。この一覧を示す。

表 1　業務サービスの障害の種類

分類	内容
値の不正	応答された値が正しい範囲内に収まっていない、またはフォーマット不正。
サービスエラー	サービスにて検出済みのエラー。ログに特定のエラーコードおよびメッセージが含まれる。
サービス実行順序不正	業務（サービスの集合）のフロー定義から逸脱したサービスの呼び出しが行われた。
タイミング不正	指定された間隔で応答されていない（早すぎ、遅すぎ）。

3.2 今回方式の概要

今回方式では、まず SOA でサービス同士が授受する SOAP (Simple Object Access Protocol) メッセージのログ、および ESB の実行ログを DB にロードする時に解析し、障害検出に必要な情報を取り出す。この情報をログ本文と関連付けて「前回処理情報」として DB に記録するとともに、次回以降のログロード時にこの「前回処理情報」を取り出し、複数のサービスの実行情報解析が必要となるサービス実行順序不正・タイミング不正の検出に利用する。障害検出に必要な情報については後述する。

3.2.1 値の不正、サービスエラー

値の不正、サービスエラーは、SOAP メッセージの要素の値やフォーマットなどが検出可能なサービス障害である。値の不正であれば、戻り値を示すタグの要素の値の条件を設定し、この条件を満たさない SOAP メッセージを生成したサービスでは障害が発生したと判断できる。サービスエラーは、SOAP メッセージ中にエラーコードやエラーメッセージが含まれていれば検出できる。

この際、XML (eXtensible Markup Language) 形式の SOAP メッセージを扱うため、サービス障害解析の過程では XML 解析が必要となる。これは、サービス障害の検出に必要なタグ、XML 要素・属性の情報を抽出するのに XML 名前空間やコメントなどを考慮する必要があるためである。今回の XML 解析の実装については後述する。

3.2.2 サービス実行順序不正

サービス実行順序不正は、サービスが、BPEL によるフロー定義の順序に従い実行されていない障害である。今回、WSDL (Web Service Description Language) によるサービス定義、およびサービスの集合としての業務プロセスの BPEL による定義からサービスの構成情報を抽出し保管する統合構成情報管理 DB を実現した。サービスの実行順序不正は、この DB の管理する情報と、ESB のログに記録されている同一セッションで実行されたサービスの実行順序を比較することにより検出するよう実装した。

サービス運用管理システムでは、ログから

① セッション ID
② サービス ID
③ 当該ログを生成したサービスの属す業務が次に実行する可能性のあるサービスの識別情報のリスト

を取り出し、ログ本文と関連付けて保管する。そして、同一セッションで次に実行されたサービスの属す業務が直前のログに関連付けられた③の情報に含まれるかをチェックすることにより、サービス実行順序不正の検出処理を行う。

3.2.3 タイミング不正

タイミング不正は、複数のサービスの入力または出力の発生した間隔に関するエラーを検出するものである。サービス運用管理システムにおいては、あらかじめ、2 つのサービスを特定する情報、およびそれらが満たすべきタイミングの制約（例：サービス A への SOAP リクエストが発せられてから、（サービス A により呼び出される）サービス B の SOAP レスポンスが発せられるまでの経過時間が 5 分以内）をタイミング不正の検出ルールとして定義しておき、新たなログを入力として受け取る度に、当該ログを生成したサービスがルールに該当するかをチェックする。この際、ログに含まれる以下の情報が必要となる。

① セッション ID（同一セッション内のサービス同士にタイミング不正検出ルールを適用する場合）
② サービス ID
③ 当該ログを生成したサービスが、タイミング不正検出ルール上先に実行されるサービス（以下「サービス 1」）に該当する可能性がある場合の、タイミング不正検出ルールの ID

サービス運用管理システムは、これらの情報をログから取り出し、ログ本文と関連付けて保管する。そして、「サービス 2」に該当する可能性のあるログを発見すると、同一セッションで直前に実行されたサービスのログを検索し、当該ログに関連付けた「サービス 1」となる可能性のある検出ルールの ID のリストを取り出す。そして、「サービス 2」に対応する検出ルールの ID と比較し、両者が一致するものを発見することにより、タイミング不正を検出する。

なお、サービスのタイムアウトもタイミング不正の一種として検出可能である。ただし、上述の仕組みに加えて、時間経過とともにサービス完了の有無を定期的にチェックする必要がある。今回はスコープを絞るためタイムアウトの実現性確認および評価は省略した。

3.3 実装

3.3.1 XML 解析

前述の通り、SOAP メッセージからサービス障害検出に必要な情報を取り出すためには、XML の構文解析を行う必要がある。標準的な XML パーザとしては、DOM (Document Object Model) と SAX (Simple API for XML) が代表的である。今回は、DOM および DOM とともに用いられる XPath のほうが障害検出ルールの記述が容易であり、システムの保守・改修を SAX よりも高効率で実現可能なため DOM を採用した。XML 処理系としては、広く普及している Microsoft Windows OS が備える MSXML3.0 を利用した。

障害検出ルールのうち、値の不正、サービスエラーの検出ルールについては、XML 文書のツリー構造におけるノードなどを特定するための標準的なインタフェースである XPath を用いて記述した。サービス実行順序不正は、統合構成情報管理 DB に定義されたサービス実行順序が障害検出ルールに相当する。

タイミング不正の検出ルールとしては、タイミングに関わる複数の XML 文書の関係を表記する必要がある。今回は、障害検出ルールを適用するサービス（前述の「サービス 1」、「サービス 2」）を特定する情報を指定する XPath 式に加え、両サービスのタイミングに関する付加情報（閾値、閾値の単位、比較演算子）を記述可能なインタフェースを用意した。3.2.3の例では、

「サービス B のレスポンス- サービス A のリクエスト ≤5 分」

という情報を記述する。

3.3.2 ログデータベース LDB

ログを格納する DBMS には、当社が研究開発しているログデータベース LDB[3]を用いた。効率的データ配置と高速・高圧縮率データ圧縮によるストレージアクセス技術、データ駆動型の並列処理技術などにより、ログをはじめ更新を伴わず追記される大量データの高速処理、効率的な保管を実現する。

ログのリアルタイムロードのインタフェースとしては、この LDB 用にデータをリアルタイムロードするための専用 API を実装した。これは、ODBC などの汎用インタフェース、或いは大量データ向けのバッチ用インタフェースを利用すると、それぞれの本来の目的と異なり、短い周期で短いデータを単位としてデータベースに書き込むため、オーバーヘッドが発生し書込みスループットが低下することが予想され、これを避けるためである。

3.3.3 機能構成

今回開発したサービス運用管理システムのプロトタイプの機能構成を図 2に示す（図 2中、点線部はサービス障害検出性能に影響しない部分）。

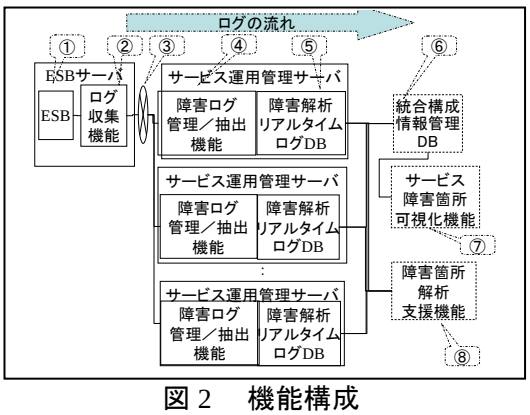

図 2　機能構成

各機能について表 2で簡単に説明する。

表 2 プロトタイプの機能の説明

番号	機能名	説明
①	ESB	サービス同士のメッセージ仲介やルーティング、プロトコル変換などを行うミドルウェア
②	ログ収集機能	ESB のログファイルを収集し効率的に④に送付するアプリケーション
③	-	ESB サーバとサービス運用管理サーバの間のネットワーク
④	障害ログ管理／抽出機能	ログからの障害情報検出およびロードを実行するアプリケーション
⑤	障害解析リアルタイムログ DB	LDB 上に、ログをサービス障害検出に必要な情報と関連づけ保管する仕組みを構築した DB
⑥	統合構成情報管理 DB	サービス、業務の呼び出し関係、依存関係などを管理する DB
⑦	サービス障害箇所可視化機能	障害箇所解析支援機能の出力をもとに業務影響範囲をグラフ表示するアプリケーション
⑧	障害箇所解析支援機能	サービス障害情報を保管するとともに、業務影響範囲を算出するライブラリ

4. 評価

サービス運用管理システムを実際の業務サービスシステムの監視に適用するに当たっては、大量のログを連続的に処理する必要がある。この際、実用的なスループットを実現できるかどうか検証する必要がある。今回はリソースの制約のため、サービス障害検出性能の評価を行い、これを元に、実用的な規模でのサービス監視性能を理論的に考察した。

4.1 サービス障害検出性能の評価

今回実装したサービス運用管理システムのプロトタイプを用いて、図 2のサービス運用管理サーバ上でのサービス障害検出性能を測定した。サービス障害を 5 件検出するのに要した時間（SOAP メッセージのログを読み込み障害が発生しているか判断するとともに、障害解析リアルタイムログ DB にログを書き込むのに要した時間）の平均を算出した。処理時間の計測は、Microsoft Visual C++の関数 GetSystemTime をサービス障害検出処理の前後で取得し、その差を求めることにより行った。

障害検出の対象となるログのサイズは下記の通り。

表 3　障害検出の対象となるログのサイズ

種類	形式	サイズ
ESB の実行ログ	テキスト形式（論理構造なし）	17,512 バイト
SOAP メッセージのログ	XML 形式	730 バイト（平均）

HW、SW の構成の詳細については本稿の付録に示した。

サービス障害検出ルールごとのサービス障害検出性能の測定結果を示す。

表 4 サービス障害検出性能

障害の種類	説明	平均所要時間
値の不正	サービスの戻り値 (SOAP Response の要素 Return の含む値)が制限を超える	0.011 秒
実行順序不正	統合構成情報管理 DB に定義されている順序と異なる順序でサービスが実行される	0.036 秒
タイミング不正	サービスの開始（SOAP Request）後、終了（SOAP Response）までの経過時間が制限を超える	0.034 秒

値の不正の検出に要する 0.011 秒は、ほぼ XML 解析に要する時間である（別途、今回用いた XML 処理系による XML 文書解析の性能を測定し、今回と同様の環境で 0.01 秒で DOM ツリーの生成と情報抽出を実行することを確認しており、値の不正検出において XML 解析以外のオーバーヘッドはほぼ発生していない。また、XML 文書のサイズが 50K バイト程度までは 0.010～0.015 秒程度で DOM ツリーの生成と情報抽出を実行することも確認している）。

サービス実行順序不正、タイミング不正については、XML 解析のほか、前回処理情報の取り出しやサービス障害検出ルールとの照合に 0.02 秒程度の時間を要している。

4.2 性能目標および実現性に関する考察

4.2.1 性能目標

現状では、SOA は大企業を中心に導入が進んでいる。ここでは、サービス運用管理システを導入する大企業の例として、1 万人規模の企業で常時 10 人に一人が業務サービスを利用するとして、サービスのログが平均毎秒 1000 件発生すると仮定する。これを 1 秒以内に処理しサービス障害を検出することが、今回の目標性能である。

4.2.2 実現性に関する考察

上述の目標の実現可能性に対し、図 2に示したサービス運用管理システムの機能ごとに分割して考察する。

図 2および表 1のうち、①～⑤は、性能評価の対象となる機能または機能間インタフェースを示す。それぞれについて、目標性能から導かれる条件、考察を行う上での仮定を示し、理論上の性能について検討する。

① ESB

本プロジェクトで扱う SOAP メッセージのログの平均サイズを簡単のため 1K バイトと仮定すると、平均毎秒 1M バイトの SOAP メッセージのログが ESB より発生する。今回使用した ESB である IBM WebSphere ESB では、実行ログ（テキストログ）と SOAP メッセージのログのファイル

を別々に保存する（図 3）。前者は、SOAP メッセージのログのファイル名やセッション ID、メッセージ生成時刻情報（タイムスタンプ）その他の経過情報を含む。

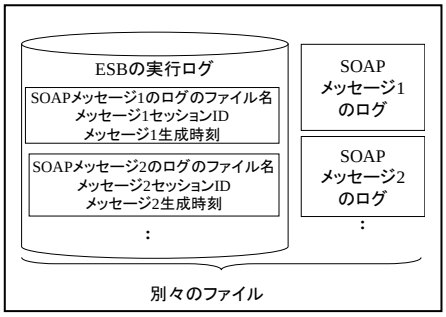

図 3 ESB のログのファイル構成

SOAP メッセージのログのファイル名の長さが平均 10 バイト程度、SOAP メッセージのログ生成時刻情報が 10 バイトとして、1 秒当たり平均 20 バイト×1000 件=20K バイトのテキストログが発生するものと仮定する。

② ログ収集機能

テキストログの内容から SOAP メッセーのログの名称を、ファイル名から SOAP メッセージのログがサービスへの入力（in）か、サービスからの出力（out）かの別を、それぞれ取り出す。

ファイル読み込み性能が本機能の性能と仮定すると、バースト転送速度[1][4]は、広く用いられている SCSI や ATA では 300～320M バイト／秒程度である。

サイズが小さい（数 M バイト程度を想定）テキストログの入力性能劣化、およびテキストマッチング処理の負荷を考慮し、本機能の性能を 100M バイト／秒と仮定する[2]。

1 件 20 バイトのテキストログから SOAP メッセージのログのファイル名、SOAP メッセージのログの生成時刻を取り出すためのテキストログ読み込みに要する時間は $20/100M=2\times10^{-7}$ 秒程度である。

③ ログ収集機能と障害ログ管理／抽出機能の間のネットワーク

以下を仮定する。

- ログの障害ログ管理／抽出機能への到着頻度はポワソン分布（ランダム到着）に従う
- 個々の障害ログ管理／抽出機能の処理時間（サービス障害検出＋ログの障害解析リアルタイムログ DB への書込み）は指数分布に従う
- ログ収集機能と障害ログ管理／抽出機能の間のネットワークは理論値 1Gbps、実行性能は半分
- [5]の情報より、MSXML3.0 は今回用いた 4 プロセッサのサーバでは 4 スレッドまで線形に性能がスケールアップする

これらの仮定のもとで、障害ログ管理／抽出機能の XML 解析サーバ数（1 サーバ 4 プロセッサと仮定）を x とすると、M/M/n の待ち行列モデルで n=4x の場合とみなすことができる。このため x についての数値計算により理論上の処理待ち時間を算出できる。

このとき、①より平均到着間隔 $Ta=1/1000=1\times10^{-3}$ 秒

また、平均毎秒 1000 件の SOAP メッセージのログの解析に要する時間を 1 件 P 秒（4.1に示した通り、検出対象となるサービス障害の種類により可変）であるとする。なお、SOAP メッセージ 1 件の伝送時にネットワーク遅延が $2\times10^{3}\times8/2/10^{9}=8\times10^{-6}$ 秒発生するが解析時間に比べて十分小さいので無視する。

1000 件の SOAP メッセージの処理をサーバ x 台のスケールアウト、サーバ 1 台当たり 4 プロセッサを用いてスケールアップすることにより、1000 件の SOAP メッセージのログの処理時間は $1000\times P/4/x=250\times P/x$ 秒

また、平均応答時間 $Tqx_{③}$（③の待ち時間＋④の処理時間）は M/M/n の算出式より数値計算可能である。この $Tqx_{③}$ の具体的な値については②⑤の処理時間についての考察と合わせて後述する。

④ 障害ログ管理／抽出機能

上述した、サービス障害検出性能評価結果（障害 1 件の検出につき P 秒）を、本機能の性能として用いる。

本機能のスケールアウト・スケールアップを考慮した性能については③で考え方を示した。具体的な値については②⑤の処理時間についての考察と合わせて後述する。

⑤ 障害解析リアルタイムログ DB

障害解析リアルタイムログ DB のリアルタイムロード機能の書込みスループットは、3.3.2で述べたリアルタイムロード専用 API による効率的な書き込み処理により、ピーク時 20MB/秒と仮定する。1 件の SOAP メッセージのログ書込みに要する平均サービス時間

$Ts=1K/(20M/秒)=5\times10^{-5}$ 秒

である。

以上より、SOAP メッセージのログ 1 件平均 1K バイトの処理に要する時間は、

②＋③＋④＋⑤

$=2\times10^{-7}+Tqx_{③}+5\times10^{-5}$

1 秒間に発生する 1000 件の SOAP メッセージのログの処理に要する時間は、④のスケールアウト・スケールアップを考慮し、

②×1000＋（③＋④＋⑤）×(1000/4x)

$=2\times10^{-7}\times1000+(Tqx_{③}+5\times10^{-5})\times1000/4x$

$\simeq 250/x\times Tqx_{③}$ 秒

これを 1 秒以内に処理完了するには、

$250/x\times Tqx_{③} \leq 1$

つまり、

$Tqx_{③} \leq x/250 = 0.004\times x$ …式 A

である必要がある。

式 A における P は、表 4に示したサービス障害検出処理をログ 1 件に対して行うのに要する時間である。このサービス障害のうち、実行順序不正、タイミング不正について

[1] ドライブのホストコントローラからインターフェイスコントローラに転送可能なデータ量

[2] 我々の開発した高速文字列照合技術 sDFA[3]（1 億字を 1 秒で処理）において 1 文字 1 バイトと仮定したときの処理性能と同等

は、サービス障害検出処理において、解析対象となっているログに関連する前回処理情報を取り出す必要がある。この処理に要する時間は、一般的には、蓄積されている前回処理情報の量が増えるほど長くなる。一方、3.3.2で紹介した LDB は、データ配置やストレージアクセスの技術により検索性能が LDB に蓄積したデータ量に影響されない、という特長を持つ。そのため、M/M/n モデルにおいて、表 4の値をそのまま実行順序不正、タイミング不正検出処理の平均サービス時間と考えることができる。

式 A において、表 4のサービス障害の種類のそれぞれについて数値計算を行い、①で述べた目標性能をサービス運用管理サーバのスケールアウトにより達成可能かどうか検証する。

- P=0.011（値の不正）

  数値計算により、

  サーバ数 2 のとき、待ち行列が無限大になり処理不可

  3 のとき、$Tqx_{③}$ (x=3)=0.012<=0.004×3

となり、サービス運用管理サーバ 3 台のクラスタ構成で実現可能である。

このとき、1000 件の SOAP メッセージのログの処理には

250× $Tqx_{③}$/4=0.75 秒

かかる。発生頻度（1 秒）より短い所要時間で障害解析リアルタイムログ DB への書込みを完了できる。

- P=0.036（実行順序不正）

  数値計算により、

  サーバ数 9 のとき、待ち行列が無限大になり処理不可

  10 のとき、$Tqx_{③}$ (x=3)=0.015<0.004×10

となり、サービス運用管理サーバ 10 台のクラスタ構成で実現可能である。

このとき、1000 件の SOAP メッセージのログの処理には

250 $Tqx_{③}$/10=0.375 秒

かかる。発生頻度（1 秒）より短い所要時間で障害解析リアルタイムログ DB への書込みを完了できる。

- P=0.034（タイミング不正）

  実行順序不正と同じく、サービス運用管理 10 台のクラスタ構成を取ることにより、発生頻度（1 秒）より短い所要時間で障害解析リアルタイムログ DB への書込みを完了できる。

以上の計算より、今回測定に用いた Intel Xeon 2.0GHz×4 台搭載クラスのサーバを、値の不正については 3 台、実行順序不正およびタイミング不正については 10 台のクラスタ構成とすることで、目標性能である毎秒平均 1000 件のログに対するサービス障害検出と障害解析リアルタイムログ DB へのロードの処理を実現できる。

5. サービス障害検出処理の改良策

前述の通り、今回開発したプロトタイプは目標性能を達成する見込みを得たが、ここでは今回成果の改良策を検討する。

5.1 XML 解析の改良

サービス障害の種類を問わず必要な XML 解析を高速化すると、障害検出処理そのものの効率を改善できる。具体的には、プロトタイプ実装において採用した DOM に代えて SAX を用いて、XML 解析の高速化を図ることができる。なお、3.3.1に述べた通り、SAX を採用しなかった理由は主に障害検出ルールの記述が複雑になるためである。このため、SAX への移行にあたっては、XML 解析性能のほか、SAX による障害検出ルール記述をサポートする機能の提供などもあわせて検討する必要がある

5.2 サービス実行順序不正・タイミング不正の検出処理の改良

値の不正、サービスの不正の検出は必須であるが、サービス実行順序不正・タイミング不正については、必ずしも検出する必要がない場合がありうると考えられる。具体的には、サービス障害検出処理の影響範囲や、監視対象となるサービスに優先度をつけ、業務上重要なサービスに絞って監視することにより、処理を効率化できる可能性がある。

6. 結論と今後の課題

サービス内部の機能や ESB の機能を基本的に使わず、ログを監視しサービスの障害を検出する方法を提案し、プロトタイプ処理系を用いて性能評価を行った。また、これに基づき、ESB で処理されるサービスが平均毎秒 1000 件のログを生成すると仮定し、このログからのサービス障害検出について実現可能なサーバ構成をした。

今後の課題としては、5章で示した、改良策の実装と実現性・有効性評価が挙げられる。また、プロトタイプ処理系の汎用性向上が挙げられる。SOAP メッセージのログからの情報抽出は標準的な XPath を用いているが、ESB の標準出力ログからの情報抽出は ESB 処理系（今回は IBM WebSphere ESB）の提供するログの形式に依存した実装になっている部分がある。種類の異なる ESB に容易に対応可能とするための、標準的な方法での ESB の実行ログからの情報抽出を実現する必要がある。

更に、今回方式と他のサービス障害検出方法との併用や役割分担について検討する必要がある。本研究においては、成果適用例の一つとして、サービスや ESB の改修・設定変更が難しい、企業合併などによる業務サービスの統合・相互利用を想定している。一方、新規業務サービスシステム構築など、障害検出にサービスや ESB、アプリケーションサーバなどの機能を利用できる場合もある。本研究にて示した方式と、これらの適切な役割分担を検討していく必要もある。

今後は、これらの課題への対処を通じて、本研究成果の実用化に取り組む所存である。

参考文献

[1]吉村礼子,馬場昭宏,山足光義,砂田英之,塚本良太,”ESB サービス稼動監視方式”, 第 70 回情報処理学会全国大会講演論文集 (2008)

[2]小川康志,吉村礼子,馬場昭宏,山足光義, “スクリーンラッピングサービスの監視および影響分析の提案”, 第 71 回情報処理学会全国大会講演論文集 (2009).

[3]郡光則, 中村隆顕,山岸義徳,” 高性能並列情報検索技術”, 三菱電機技報，83， No.12 (2009)
http://support.microsoft.com/kb/172338/ja
[4]シーゲイト社,” 速度の確認”, シーゲイト社日本語ホームページ (2010)
http://www.seagate.com/ww/v/index.jsp?locale=ja-JP&name=Performance_Considerations&vgnextoid=b018cac4bdfce010VgnVCM100000dd04090aRCRD
[5]Chris Lovett,” インサイド MSXML3 パフォーマンス”, マイクロソフト株式会社ホームページ (2000)
http://msdn.microsoft.com/ja-jp/library/ms950768.aspx

付録　サービス障害検出性能測定環境

機能構成は本文参照。

表 5　H/W 環境（サービス運用管理サーバ）

項目	内容
機種	VMWare ゲスト (物理サーバは HP BL460c G7)
ハイパバイザ	vCenter Server 4.0.0
CPU	vCPU 2.0GHz×4 (物理 CPU は Intel Xeon X5670 2.93GHz 6core/chip)
メモリ	6144MB（ゲスト OS） （物理メモリは 72GB）
HDD	280GB（ゲスト OS） （物理ディスクは 300GB 10Krpm SAS ×2(RAID1)）

表 6　S/W 環境（サービス運用管理サーバ）

項目	内容
OS	Windows Server 2008 Enterprise(x64)
RDBMS	SQLServer 2005
アプリケーションサーバ	Internet Information Service 7.0
アプリケーション実行環境	.NET Framework 3.0
その他	LDB 2.10