

リクエスタサイドから制御可能な Web サービスの 分散型テストシステムの提案

Proposal of distributed testing system for Web Services controlled by request sides

光林 真[†]
MITSUBAYASHI, Shin

村山 隆彦[‡]
MURAYAMA, Takahiko

多田 好克[†]
TADA, Yoshikatsu

1. はじめに

近年, Web サービスと呼ばれる分散システムアーキテクチャが注目されている. Web サービス技術では, 単一または複数ホストが提供するサービスを標準化されたプロトコル (SOAP[1] など) に基づいて組み合わせることで, 粗粒度の複合サービスを合成できる. 以下, サービスの提供者をプロバイダ (Provider), サービスの利用者をリクエスタ (Requester) と呼称する. 複合サービスでは, プロバイダとリクエスタを兼ねることもある. Web サービスの概要を図1に示す.

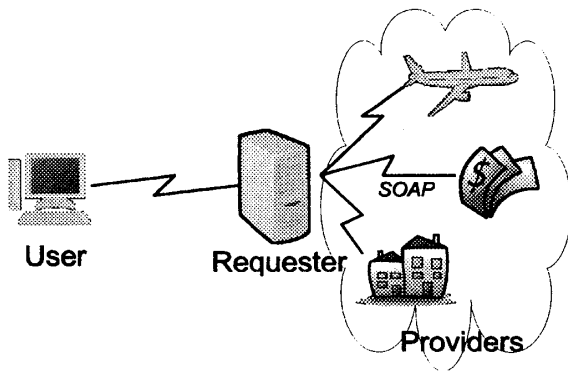


図1: Web サービスの概要

複合サービスにおいて, 基本的にリクエスタは個々のプロバイダのサービスの内部動作を知ることとはできない. これは, プログラムモジュールの抽象化という観点から見ると, 実装の詳細を知らなくてもアプリケーションを構築できるという利点がある. 一方, 実装が隠蔽されてしまうのでデバッグやテストが困難になるという問題もある. 特に複合 Web サービスの場合はサービス間に依存関係が発生するので, より問題は複雑化する.

そこで本研究では, 動的コード変換技術を用いてサービスの内部動作の制御を可能とする分散型テストシステムを提案し, システムの実現可能性について検討する.

2. 分散型テストにおける問題

Web サービスはメッセージングを用いた分散システムの一つである. 独自のプロトコルに基づく他の分散システムとは異なり, XML ベースのメッセージフォーマット (SOAP) や HTTP などの通信

プロトコルなど, 標準化された技術でレイヤが構成されている. また, 普及している技術を使用するので, 相互運用性に優れている.

この相互運用性の良さを生かし, 近年では企業内システムの構築に Web サービス技術を用いる事例も増えている. 既存の資源を多く抱える企業にとって, 各部署ごとに独立しているシステムを相互に接続し連携させられることは大きな魅力である.

しかし, システムの開発段階においてリクエスタからテストを行う際, その抽象的なアーキテクチャが障壁となる. 各サービスの内部動作は隠蔽されており, リクエスタはサービスインタフェースで公開されている情報しか得られず, 問題を特定できる決定的な情報の入手が困難になる. 複合サービスの場合は複数のプロバイダが関係する可能性も有り得るので, より問題は複雑化する.

具体的な問題として, 以下のようなものが挙げられる.

1. サービスが汎用的なテスト用インタフェースを用意していたとしても, そのインタフェースの規約に従わなければならないという根本的な問題は解決しない.
2. リクエスタからプロバイダを制御できないので, 予めプロバイダの想定しているテスト要件に干渉するようなテストケースを構築できない.

3. 解決のアプローチ

前章で述べた問題を解決するためには, テスト環境がインタフェースの制約を受けずにリクエスタサイドからサービスの内部動作に干渉し情報を取得できるような機能を備えていればよい. この機能は, 以下に述べるアプローチに基づいて実現できる. なお, 本研究は企業内システムなどの外部に公開されないネットワークシステムへの適用, かつ開発段階の措置という前提に基づいており, サービス内部への干渉は認められるものとする.

テスト用コードやツールの送信

直接サービス内部へ干渉するために, リクエスタサイドで用意したテスト用コードやライブラリ, またはサービス監視用のツール (以下, テストツール一式と呼称) をプロバイダ内部に送信する.

テストツール一式の展開とサービスへのフック

プロバイダ内部に送り込まれたテストツール一式を展開し, サービス内部へのフックおよび内部情報の取得を実行する.

[†] 電気通信大学 大学院情報システム学研究科, UEC

[‡] 日本電信電話株式会社 NTT 情報流通プラットフォーム研究所, NTT

4. 設計

分散型の形態でテストシステムの設計を行った。システムの概要を図2に示す。

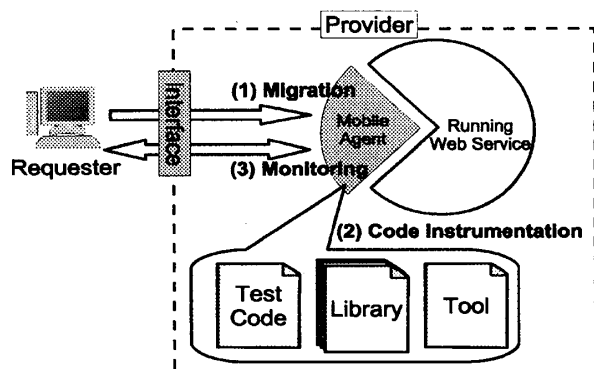


図2: システムの概要

- (1) テストツール一式はモバイルエージェント(以下エージェントと呼称)にラップされ、テスト用インタフェースを通してプロバイダ内部へ送信される。このインタフェースは特定のテストに特化されたものではなく、汎用的に使用できる(Migration)。
- (2) エージェントがテストツール一式を用いてサービスの動作に割り込む。このとき、もしサービスへのフックが必要であれば、エージェントがコード変換技術を用いて直接サービスのコードを書き換える(Code Instrumentation)。
- (3) リクエスタがエージェントを介してプロバイダを監視する(Monitoring)。

エージェントやコード変換技術を適用するためには、コードの動的なロードや変換をサポートしている処理系であることが必須要件となる。従って、本システムでは該要件を満たす処理系に対象を限定する。

5. 関連技術

5.1. 動的なコード変換技術

既存のコード変換技術として、アスペクト指向プログラミング(AOP)が注目されている。AOPは、コンパイル済みのコードに対し、実行前または実行時に別のコードを挿入することができる。挿入されるコードは挿入ポイントの指定なども含めて高級言語で記述できるので、既存のプログラムの補完や、デバッグコードの挿入などを簡単に実現できる。従って、AOPは本システムの実現に有用である。

分散ソフトウェア用のAOP言語DJcutter [2][3]は、分散テストを強く意識して設計されており、AOPの手法を用いてリモートホスト上のコードを動的に制御できる。しかし、独自に拡張した言語を使用する上に、コードのロード時に変換を行うので、既に実行中のコードの操作には向かない。

AspectWerkz[4]、PROSE[5]は、Java言語で動的なAOP(DAOP)を実現するフレームワークである。他のAOPツールと異なり、独自の構文拡張や特定のフレームワークに依存するクラス拡張を必要としないという利点がある。また、実行中のコードに対する割り込みやリモートからのコード変換もサポートしている。一方で、Webサービスとは異なる通信プロトコルを用いるので、プロバイダはサービスインタフェース以外の通信経路を開放しなければならないという問題がある。

5.2. Webサービスのセキュリティ

外部からの実行環境の制御は極めて重大なセキュリティ上の危険が伴うので、テスト用インタフェースは厳重に保護され、信頼できるホストのみが暗号化された経路を用いてアクセスできるようにしなければならない。

現在、Webサービスではセキュリティ技術を細かく分類し、それぞれについて仕様の策定を進めている[6][7]。先に述べた本システムの適用とセキュリティ確保のトレードオフに関しては、これらWebサービス固有のセキュリティ技術と、SSLなどの既存の暗号化経路の組み合わせで解決できる。すなわち、使用するプロトコルで利用可能な標準技術を用いればよく、特別なセキュリティ機構を独自に実装する必要はないと考える。

6. まとめと今後の予定

リクエスタサイドからWebサービス実行環境を制御可能な分散型テストシステムを提案し、システムの大まかな仕様を策定した。また、AOPを用いた既存の動的コード変換技術を調査し、本システムへの適用可能性を検討した。

今後は、まず関連技術を用いた小規模な試作を行い、実装における問題点を洗い出した後、システムを設計、実装する予定である。併せて評価を行い、システムの有効性を検証する。

参考文献

- [1] Simple Object Access Protocol (SOAP), <http://www.w3.org/TR/soap/>
- [2] 西澤無我, 千葉滋, 立堀道昭, “分散ソフトウェアのテストに適したアスペクト指向言語”, 情処学論, Vol.46, No.7, pp.1723-1734, 2005.
- [3] 佐藤芳樹, 千葉滋, “効率的なJava Dynamic AOPシステムを実現するJust-in-Time Weaver”, 情処学論, Vol.41, No. SIG13, pp.15-24, 2003.
- [4] AspectWerkz, <http://aspectwerkz.codehaus.org/>
- [5] PROSE, <http://prose.ethz.ch/Wiki.jsp>
- [6] Web Services Security, <http://docs.oasis-open.org/wss/v1.1/>
- [7] OASIS Web Services Secure Exchange (WS-SX) TC, http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=ws-sx