南山大学 大学院 数理情報研究科 2006 年度修士論文 要旨集

非同期メッセージ交換パターンに基づく 非同期サービス指向アーキテクチャの提案と評価

M2005MM016 森 晃

指導教員 青山 幹雄

1. はじめに

Web サービスの多くは同期型であるため，リクエスタはリクエストを送信後，レスポンスを受信するまで待ち状態になる問題がある．この問題を解決する一方法として非同期型のメッセージ交換を用いた非同期型 Web サービスが提案されている．非同期型 Web サービスにおけるメッセージの交換方式はサービス利用形態によって異なるため，各サービスに適したメッセージ交換方式の選択が必要となる．

本稿では，サービス間のメッセージ交換をパターン化し，その組み合わせによって構築される，非同期型メッセージ交換を用いたサービス指向アーキテクチャを非同期サービス指向アーキテクチャ(非同期 SOA)として提案する．

2. 非同期サービス指向アーキテクチャ

非同期 SOA を実現する方法として，以下に示すメッセージ交換のモデルとパターンを定義し，パターンに基づく非同期 SOA の構築方法を提案する．

(1) メッセージ交換共通モデル(メタモデル)
(2) 非同期メッセージモデル
(3) 非同期メッセージ交換パターン
(4) 非同期 SOA パターン

図1に提案するメッセージ交換のモデルとパターンの関係を示す．まず，すべての Web サービスのメッセージ交換に共通したメタモデルをメッセージ交換共通モデルとして定義する．さらにメタモデルに基づき，非同期メッセージ交換を非同期メッセージモデルとして定義する．

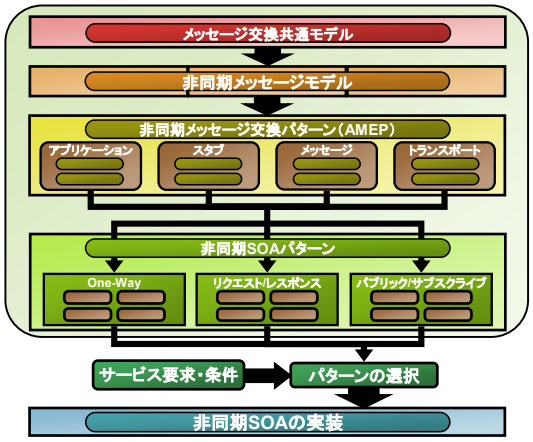

図 1 提案するメッセージ交換のモデルとパターン

また，本稿では，メッセージ交換を 4 階層モデルで表し，各階層のメッセージ交換の性質をそれぞれ非同期メッセージ交換パターン AMEP（Asynchronous Message Exchange Pattern）としてパターン化する．さらに，4 階層で定義した AMEP を組み合わせ，非同期メッセージ交換のアーキテクチャを実現する非同期 SOA パターンを定義する．

サービスの開発者は，サービスの要求や条件を与えることでパターンとのマッチングをとり，選択されるパターンが実現可能かを判断する．選択されたパターンに基づき非同期 SOA の構築が可能となる．

3. メッセージ交換共通モデル

図 2 に Web サービスのメッセージ交換のメタモデルとして提案するメッセージ交換共通モデルを示す．本稿が提案する非同期 SOA パターンは，「Message Exchange Pattern」を特殊化した「Asynchronous Message Exchange Pattern」により構成される．さらに，「Asynchronous Message Exchange Pattern」は 4 階層のパターンに特殊化され，各レベルはメッセージ交換の各性質に関連する[2].

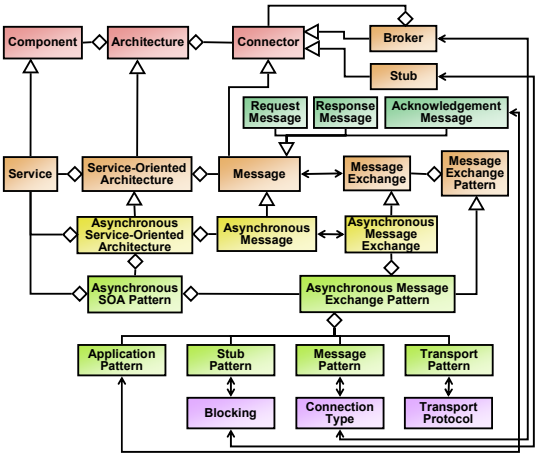

図 2 メッセージ交換共通モデル

4. 非同期メッセージモデル

非同期メッセージ交換は，同期型のリクエスト/レスポンス型とは異なり，通信形態が様々である．メッセージの形態や

性質によって非同期メッセージ交換の分類が必要である.

本稿では, 図 3 に示すようにメッセージ交換の分類基準を, メッセージの配信構造であるネットワークトポロジとメッセージ交換スタイルであるインタラクションとして, 以下に示す3つの非同期メッセージモデルを定義する[6].

1) One-Way 型メッセージモデル
2) リクエスト/レスポンス型メッセージモデル
3) パブリッシュ/サブスクライブ型メッセージモデル

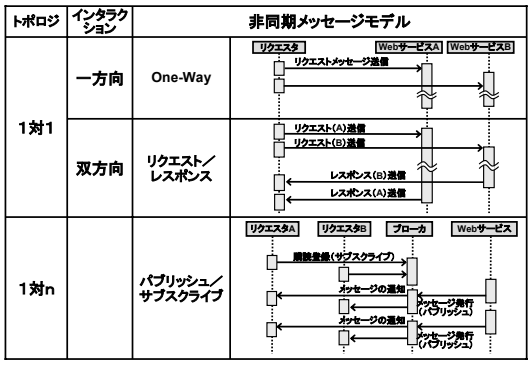

図 3 非同期メッセージモデル

5. 非同期メッセージ交換パターン

5.1. メッセージ交換 4 階層モデル

本稿では, Web サービスのメッセージ交換を図 4 に示すメッセージ交換4階層モデルを用いて 4 つの階層に分け,メッセージモデルを定義する分類基準以外のメッセージ交換の性質を各階層の AMEP として定義する. 提案する 4 階層のカテゴリと分類基準となるメッセージ交換の性質を表 1 に示す.

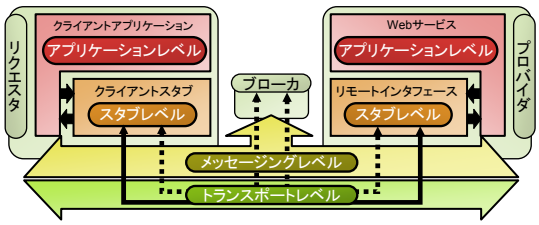

図 4 メッセージ交換 4 層モデル

5.2. 4 階層モデルに基づく AMEP

前節に示した 4 階層モデルの各階層のメッセージ交換の性質を基準として, 以下に示す AMEP を定義する[5].

(1) アプリケーションレベル

アプリケーションレベルでは, クライアントが送信したリクエストメッセージが正しく送信されたかを確認するためのプロバイダからの確認メッセージを必要とするかしないかにより, In-Only パターンと In-Out パターンに分類できる.

(2) スタブレベル

スタブレベルでは, リクエストメッセージを送信後, リクエスタがレスポンスメッセージを受信するまで待ち合わせをするか否かにより, Blocking パターンと Non-Blocking パターンに分類できる. Non-Blocking パターンを選択することで, スタブレベルの非同期を実現する.

また, Non-Blocking パターンはレスポンスを待機しないため, どのような方法で, どのタイミングでレスポンスメッセージを受信するかにより, コールバックパターンとポーリングパターンに分類できる.

(3) メッセージレベル

メッセージレベルでは, リクエスタとプロバイダ間が直接接続されているのか, または, ブローカ等を介して接続されているかによって, 直接接続パターンと間接接続パターンに分類できる. 間接接続ではリクエスタとプロバイダはブローカを介した 2 つの接続によってメッセージ交換が行われる疎結合の関係にあるため, メッセージレベルでの非同期を実現する.

(4) トランスポートレベル

トランスポートレベルでは, リクエスタとプロバイダ間のトランスポートの接続回数により SingleTransport パターンと DualTransport パターンに分類できる.

例として, 同期型のプロトコルである HTTP を用いた場合, 通常はシングルトランスポートとして同期型のメッセージ交換を行うが, 2 回の接続によってリクエストとレスポンスを分離することによりトランスポートレベルの非同期メッセージ交換を実現する.

6. 非同期 SOA パターン

前章で定義した 4 階層の AMEP を図 5 に示すように組み合わせることで非同期 SOA パターンを定義する. しかし, 各階層の AMEP は表 2 に示すように, 非同期メッセージモデルによっては対応しない AMEP が含まれるため, 各モデルに対応した 3 つの非同期 SOA パターンを提案する.

表 1 4 階層モデルのメッセージ交換の性質

4 階層モデル	カテゴリ	メッセージ交換の性質
アプリケーションレベル	各サービスに依存する性質	・送信確認メッセージの有無
スタブレベル	メッセージの受け渡しに関する性質	・レスポンスメッセージの待ち合わせの有無 ・レスポンスメッセージの受信方法
メッセージレベル	ルーティングに関する性質	・エンドポイント間の接続形態
トランスポートレベル	プロトコル依存の性質	・トランスポートプロトコルの通信方式

表 2 非同期メッセージモデルと AMEP の対応関係

			One-Way 型	リクエスト/レスポンス型	パブリッシュ/サブスクライブ型
アプリケーションレベル	In-Only		○	○	○
	In-Out		○	○	○
スタブレベル	Blocking		×	○	○
	Non-Blocking	コールバック	○	○	○
		ポーリング		○	×
メッセージレベル	直接接続		○	○	×
	間接接続		○	○	○
トランスポートレベル	Single Transport		○	○	×
	Dual Transport		×	○	○

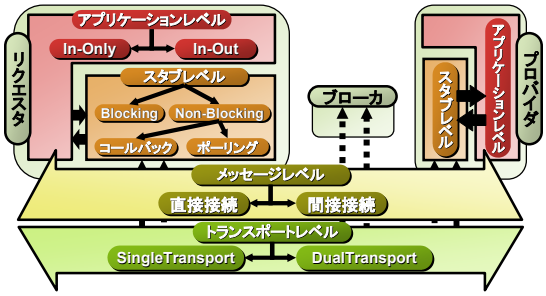

図 5 非同期 SOA パターン

(1) One-Way 型非同期 SOA パターン

一方向の非同期メッセージ交換を実現するパターン．アプリケーションレベル，メッセージレベルでは任意の AMEP の選択が可能である．しかし，One-Way 型は一方向メッセージであるため，スタブレベルの AMEP の選択肢は Non-Blocking パターン，トランスポートレベルは，Single Transport パターンのみとなる．

(2) リクエスト/レスポンス型非同期 SOA パターン

リクエスタとプロバイダが 1 対 1 である双方向型の非同期メッセージ交換を実現するパターン．最もスタンダードなメッセージ交換方式であり，4 階層すべてのレベルで任意の AMEP の選択が可能である．

(3) パブリック/サブスクライブ型非同期 SOA パターン

プロバイダが複数のリクエスタに同一のメッセージを通知する 1 対多のメッセージ交換パターン．アプリケーションレベルとスタブレベルでのパターンの選択が可能である．しかし，メッセージ購読要求・発行はブローカを介して行われるため，メッセージレベルの AMEP の選択肢は間接接続パターン，トランスポートレベルは Dual Transport パターンのみとなる．

7. パターンに基づく非同期 SOA の構築

7.1. パターンの選択

非同期 SOA の実現のためには，開発するサービスの要求に基づいて，4 階層 AMEP の選択が必要となる．パターンの選択は，まず外部仕様として，開発するサービスのトポロジとインタラクションを与えて，メッセージモデルを決定する．次に，各メッセージモデルの AMEP の制約条件とサービスの要求のマッチングを取り，選択される AMEP が実現可能かを判断する．AMEP の選択の可否を判断するためのサービス要求は，直接 4 階層の性質に関連する要求だけでなく，信頼性や待ち時間の制限といった非機能的な要求も必要となる．

7.2. パターンに基づく非同期 SOA の構築

4 階層から選択された AMEP のメッセージ交換シーケンスを組み合わせた，非同期 SOA パターンのメッセージ交換シーケンスと，パターンに基づく非同期 SOA を Apache Axis を用いて実装する一例を図 6 に示す．

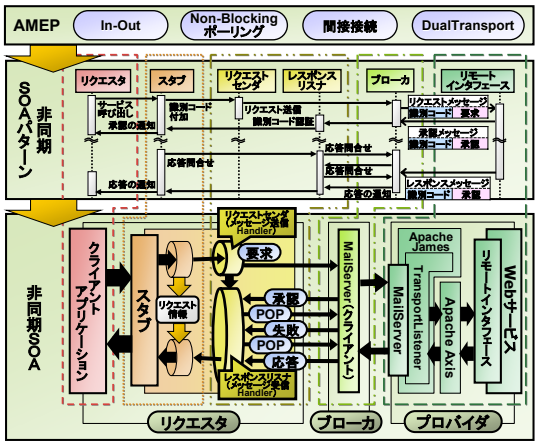

図 6 パターンに基づく非同期 SOA

非同期 SOA パターンのメッセージ交換シーケンスは上位層の AMEP からトップダウンで組み合わせ，各層の性質である，承認メッセージの受信，問い合せによるレスポンスの受信，ブローカ，2 回のトランスポート接続を順に追加して決定する．

さらに，非同期 SOA パターンを用いて非同期 SOA を構築するときは，非同期 SOA パターンの各要素に対応した機能を持つコンポーネントとの対応付けを行う．例では，パターンの各要素を Apache Axis の該当する Handler に対応付けて非同期 SOA を構築する[1].

8. 非同期 SOA パターンの評価・考察

8.1. Apache Axis を用いたパターンの評価・考察

本稿では，Apache Axis の 2 つのバージョンの各 Handler の機能を分析し，AMEP の対応範囲を比較した(表 3).

Axis 1.x 系は，One-Way 型とパブリッシュ/サブスクライブ型に非対応であるため，いずれのパターンも実現できない．リクエスト/レスポンス型では，一回の HTTP 接続を用いてレ

表 3 Apache Axis の AMEP への対応と関連研究の実現する範囲

			One-Way 型			リクエスト/レスポンス型			パブリッシュ/サブスクライブ型		
			非同期 SOA パターン	Axis 1.x	Axis2	非同期 SOA パターン	Axis 1.x	Axis2	非同期 SOA パターン	Axis 1.x	Axis2
アプリケーション	In-Only		○	×	○	○	○	○	○	×	×
	In-Out		○		○	○	○	○	○		
スタブ	Blocking		×		×	○	○	○	○		
	Non-Blocking	CB	○		○	○	×	○	○		
		PLL				○		×	×		
メッセージ	直接接続		○		○	○	○	○	×		
	間接接続		○		○	○	○	○	○		
トランスポート	Sngle Transport		○		○	○	○	○	×		
	Dual Transport		×		×	○	×	○	○		

スポンスを待ち合わせる，Bloking と SingleTransport のみに対応しているため，Non-Blocking と DualTransport は選択不可となる．一方，Apache Axis 2 では，One-Way 型で選択可能なすべての AMEP に対応している．また，リクエスト/レスポンス型では，2 回のトランスポート接続に対応しているため，DualTranspurt の選択が可能である．

しかし，標準で対応していない AMEP も，独自の Handler を追加することで，非同期 SOA の実装が可能となる．本稿では，One-Way 型非同期 SOA パターンに基づく非同期 SOA を Apache Axis 1.2 の Handler を拡張し，SOAP over SMTP を用いて実装できることを確認した[4].

8.2. 関連研究との比較・考察

非同期メッセージ交換に関する関連研究で提案されているパターンやモデルが実現できる範囲を表 3 に示す．

(1) 非同期呼び出しパターン（実現範囲：表 3 実線枠内）

既存の Web サービスフレームワーク上で非同期呼び出しを提供する 4 つの非同期呼び出しパターンが提案されている[7]. このパターンは，一方向メッセージを確認メッセージの有無，双方向メッセージをレスポンスの受信方法のみで分類している．

(2) 2 層非同期通信モデル（実現範囲：表 3 破線枠内）

Web サービス呼び出しを 2 層のメッセージ交換で行う 2 層非同期通信モデルが提案されている[3]. このモデルはメッセージ交換を API 層とトランスポート層に分離し，API 層でレスポンスの待ち合わせの有無と受信方法，トランスポート層で HTTP の接続回数を定義している．

2 つの関連研究は，非同期メッセージ交換を分類したパターンとモデルを定義しているが，いずれの研究も本稿が提案する非同期 SOA パターンの一部に限定される．

また，非同期呼び出しパターンでは，1 つの分類基準によってのみメッセージ交換が決定される．このため，例えばリクエスト/レスポンス型で確認メッセージの有無を指定しつつ，レスポンスの受信方法を指定することができない問題がある．一方，提案する非同期 SOA パターンは，メッセージ交換を 4 階層モデルで定義しているため，各階層のパターンを組み合わせた多様なメッセージ交換を実現できる．

9. 今後の課題

(1) サービスの要件定義と AMEP への対応付け

AMEP を決定するための外部仕様であるサービスの要件の定義と，AMEP の分類基準となる性質との関連性を明確にする必要がある．

(2) Apache Axis 以外へ非同期 SOA パターンの適用

本稿では，Apache Axis を用いたアーキテクチャを提案したが，他のプラットフォームでも，パターンの要素とアーキテクチャとの対応付けを明確にする必要がある．

10. まとめ

本稿では，同期型 Web サービスの問題点を解決する方法のひとつとして非同期型のメッセージ交換を用いた非同期 SOA を提案した．メッセージの交換を 4 階層に分割し，各レベルのメッセージの特性を AMEP として定義した．

さらに，4 階層の AMEP を組み合わせた非同期 SOA パターンを定義し，パターンに基づく非同期 SOA を構築する方法を提案した．非同期 SOA パターンを用いることによって，各サービスの性質に適する非同期 SOA を実現する．

参考文献

[1] The Apache Software Foundation, *Axis Documentation*, http://ws.apache.org/axis/java/ index.html.

[2] H. Jonkers, et al., Towards a Language for Coherent Enterprise Architecture Descriptions, Proc. of EDOC '03, IEEE Computer Society, Sep. 2003, pp. 28- 37.

[3] 木村 利幸, 非同期 Web サービス実現アーキテクチャ提案, 2004, http://ws.apache.org/~toshi/docs/JSR224-F2F-no1-jp.pdf.

[4] 森 晃, 青山 幹雄, SMTP を用いた非同期型 Web サービスの提案と評価, 情報処理学会ソフトウェア工学研究会, Vol. 2005-SE-147, Mar. 2005, pp. 73-80.

[5] 森 晃, 青山 幹雄, 非同期型メッセージのパターン化に基づく Web サービスアーキテクチャの提案と評価, SES2006 論文集, Oct. 2006, pp. 137-144.

[6] M. Narayanan, Approaches to Asynchronous Web Services, Sep. 2003, http://www-128.ibm.com/developerworks/webservices/library/ws-asoper/.

[7] U. Zdun, et al., Pattern-Based Design of an Asynchronous Invocation Framework for Web Services, *Int'l J. of Web Service Research*, Vol. 1, 2004, pp. 1-14.