アスペクト指向に基づくSOAの考察
– PLSEを適用したSOAに基づくシステムのアーキテクチャ設計支援 –

2008MI089 鹿島 一彦　　2008MI181 岡田 大輝　　2008MI289 吉川 正晃
指導教員　野呂 昌満

1　はじめに

Service-Oriented Architecture(以下, SOA) はサービスを組み合わせることでシステムを構築するアーキテクチャである．Georgakopoulos らの研究 [2] では，SOA に基づくシステムの参照アーキテクチャを提案している．一般に，SOA に基づくシステムアーキテクチャは参照アーキテクチャから設計される．Product Line Software Engineering(以下，PLSE)[4] のドメイン工学では，仕様変更やアーキテクチャが存在しない場合にアーキテクチャを設計する．一般に，非機能特性を考慮してソフトウェアアーキテクチャを設計することは容易な作業ではない [5]．仕様モデルとアーキテクチャ間の追跡性の確保によりアーキテクチャ設計を支援することができる．

SOA に基づくシステム開発において，非機能特性を考慮したシステムアーキテクチャの設計は困難である．求められる非機能特性に応じたシステムアーキテクチャの設計方法が整理されていない．

本研究の目的は，非機能特性を考慮した SOA に基づくシステムアーキテクチャ設計のガイドラインの提案である．PLSE の概念に基づき，非機能特性を考慮した仕様モデルと非機能特性を実現するプロダクトラインアーキテクチャを定義する．定義した仕様モデルとプロダクトラインアーキテクチャの追跡性を確保する．追跡性を確保した仕様モデルとアーキテクチャをシステムアーキテクチャ設計のガイドラインとして提示する．定義したガイドラインを利用することで，求められる非機能特性に応じたアーキテクチャのコンポーネントの追跡が可能となる．

本研究では，アーキテクチャ設計にアスペクト指向技術を用いることで仕様モデルとアーキテクチャ間の追跡性を確保した．一般に，仕様モデルとアーキテクチャが扱う関心事が異なるので，仕様モデルとアーキテクチャ間の追跡性の確保が困難である．我々は，アーキテクチャが扱う関心事を仕様モデルに揃えることで仕様モデルとプロダクトラインアーキテクチャ間の追跡性を確保した．プロダクトラインアーキテクチャが扱う関心事を仕様モデルに揃えるために，プロダクトラインアーキテクチャ設計にアスペクト指向技術を適用した．仕様モデルが扱う関心事をアスペクトとして分離し，設計することで仕様モデルとプロダクトラインアーキテクチャ間の追跡性を確保した．

仕様モデルに PLSE で一般に用いられている FORM[3] に基づくフィーチャモデルを用いた．アスペクト指向アーキテクチャの記法に Views&Beyond[6] の記法を用いた．

本研究の結果として，ガイドラインに基づくことで SOA に基づくシステムに求められる非機能特性に応じたコンポーネントの追跡が可能となった．ガイドラインを提案することで，非機能特性を考慮した SOA に基づくシステムアーキテクチャの設計支援を実現した．

2　背景技術

2.1　SOA

SOA とは業務処理の単位であるサービスを組み合わせることにより，システムを構築する設計手法である．SOA に基づくシステムアーキテクチャは参照アーキテクチャを基に設計される．参照アーキテクチャは，サービスプロバイダ，サービスリクエスタ，サービスブローカーの関係に関するアーキテクチャである [2]．これらの関係を図 1 に示す．

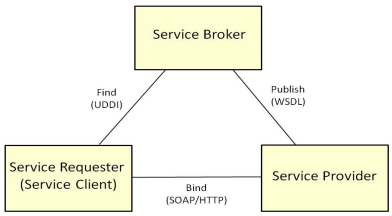

図 1　参照アーキテクチャ

2.2　PLSE

PLSE[4] とは, ソフトウェアの複雑化を解決し，ソフトウェア開発にかかる工数削減と品質向上を重視した開発手法である．PLSE の開発においてフィーチャモデルが仕様モデルとして代表的に用いられている．フィーチャモデルとは製品系列の共通性と可変性を表した仕様モデルである．フィーチャとは外部から認識可能なシステムの機能や品質，特性である．

FORM(Feature-Oriented Reuse Method)[3] に基づくフィーチャモデルは，フィーチャの性質を 4 層に分離して記述する．Capability Layer はユーザが要求する機能，非機能を示す．Operating Environment Layer はハードウェア，ソフトウェアなどの動作環境を示す．Domain Technology Layer は特定のドメイン内で使用される技術を示す．Implementation Technique Layer では特定のドメインに依存しない一般的な技術を示す．

2.3 アスペクト指向技術

アスペクト指向技術とは，支配的分割によって得られる構造に散在する横断的関心事を分離する技術である．横断的関心事とは，アーキテクチャ上の複数のコンポーネントに散在し，処理内容が他の関心事に依存する関心事である．アスペクト指向技術を用いて横断的関心事を分離することにより，仕様モデルとアーキテクチャの関係が明確化され，追跡性の確保が可能になる [1]．

3 システムアーキテクチャ設計のガイドライン

本研究では，非機能特性を考慮した SOA に基づくシステムアーキテクチャ設計のガイドラインを提示する．非機能特性を考慮した仕様モデルと，非機能特性を実現するシステムアーキテクチャのプロダクトラインアーキテクチャを定義する．定義した仕様モデルとプロダクトラインアーキテクチャの対応関係を明確化し，追跡性を確保することにより，それらをガイドラインとして定義する．ガイドラインの概要を図 2 に示す．

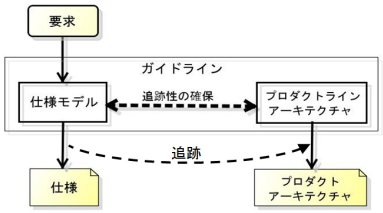

図 2　ガイドラインの概要

技術者はガイドラインを利用し，要求と仕様モデルから仕様を作成，仕様とプロダクトラインアーキテクチャからプロダクトアーキテクチャを設計する．仕様モデルとプロダクトラインアーキテクチャ間の追跡性が確保されていることから，非機能特性に応じたアーキテクチャのコンポーネントの追跡が可能となる．仕様で扱う非機能特性から必要となるアーキテクチャのコンポーネントを追跡することが可能となり，プロダクトアーキテクチャ設計が支援できる．

3.1 SOA に基づくシステムに必要な非機能特性の整理

システムアーキテクチャ設計のガイドラインを提示するために，SOA に基づくシステムに必要となる非機能特性の整理を行なう．Georgakopoulos らの研究 [2] から，SOA に基づくシステムに求められる非機能特性を整理した．本研究で提案するガイドラインでは，以下に示す非機能特性を対象とする．

- スケーラビリティ (Scalability)
- 位置透過性 (Location Transparency)
- 相互運用性 (Interoperability)
- 並行性 (Concurrency)
- 信頼性 (Reliability)
- セキュリティ (Security)
- 正確性 (Accuracy)
- フォールトトレランス (Fault Tolerance)

Georgakopoulos らの研究 [2] から，非機能特性と同様に非機能特性の実現方法，非機能特性の実現に必要となるアーキテクチャ上のコンポーネントを整理した．

3.2 フィーチャモデルの作成

整理した非機能特性と非機能特性の実現方法を基に，仕様モデルとしてフィーチャモデルを作成する．フィーチャモデルの記述は FORM[3] に従う．Capability Layer に整理した非機能特性を表し，Implementation Technique Layer に整理した非機能特性の実現方法を表す．作成したフィーチャモデルを図 3 に示す．

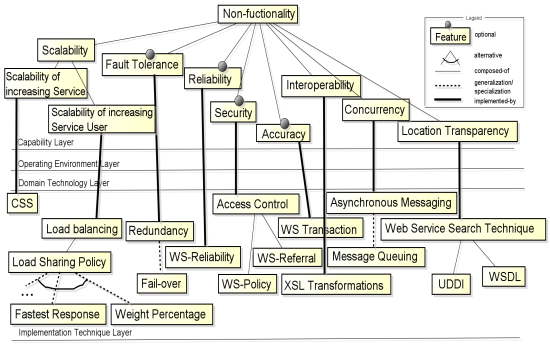

図 3　非機能特性を考慮したフィーチャモデル

3.3 プロダクトラインアーキテクチャの設計

3.3.1 アスペクト指向技術の適用

仕様モデルとプロダクトラインアーキテクチャ間の追跡性を確保するために，プロダクトラインアーキテクチャ設計にアスペクト指向技術を適用する．仕様モデルとプロダクトラインアーキテクチャが扱う関心事を揃えることで対応付けが容易になり，追跡性が確保できると考える．アスペクト指向技術をアーキテクチャ設計に適用することで，仕様モデルとプロダクトラインアーキテクチャで扱う関心事を揃える．SOA に基づくシステムの支配的分割から非機能特性を実現するコンポーネントをアスペクトとして分離し，対応付けが容易なアーキテクチャを設計する．アーキテクチャ上のアスペクトの表現は Views&Beyond[6] の記法を用いる．

サービスの機能を提供する Service Provider とその機能を要求する Service Requester が連携してシステムを構築する．Service Requester から Service Provider への通信にアスペクトを織り込むことで非機能特性を満たす通信を実現する．

3.3.2 プロダクトラインアーキテクチャの定義

整理した非機能特性と，非機能特性を実現するアスペクトを基にプロダクトラインアーキテクチャを定義する．アスペクトの詳細な織込み箇所は，プロダクトに依存するので，プロダクトラインアーキテクチャには表現しない．定義したプロダクトラインアーキテクチャを図 4 に示す．

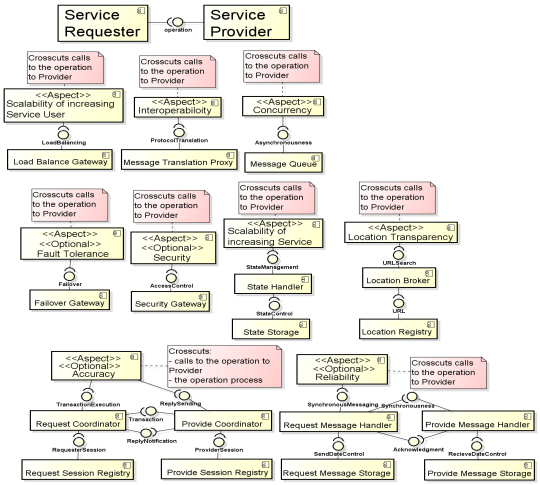

図 4　定義したプロダクトラインアーキテクチャ

3.3.3　フィーチャモデルとプロダクトラインアーキテクチャ間の追跡性の確保

非機能特性を表すフィーチャと非機能特性を実現するアスペクトを対応付けることにより追跡性の確保を行なう．仕様モデルとプロダクトラインアーキテクチャが扱う関心事をアーキテクチャ設計にアスペクト指向技術を用いて揃えたことから対応付けが容易になる．フィーチャとアスペクトの対応付けにより，プロダクトラインフィーチャモデルとプロダクトラインアーキテクチャ間の追跡性を確保できたと考える．例としてフィーチャとアスペクトの対応関係を一部抜粋して図 5 に示す．

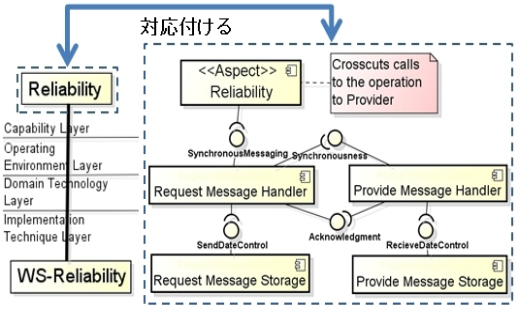

図 5　フィーチャとアスペクトの対応関係

4　事例検証

出庫受付システムを事例とし，ガイドラインに基づいてシステムアーキテクチャを設計する．出庫受付システムの分析結果から，出庫受付システムのフィーチャモデルを作成した．作成したフィーチャモデルを図 6 に示す．

仕様モデルとアーキテクチャで扱う関心事を揃えたことで，非機能特性を表すフィーチャから非機能特性を実現するアスペクトが追跡できた．出庫受付システムに必

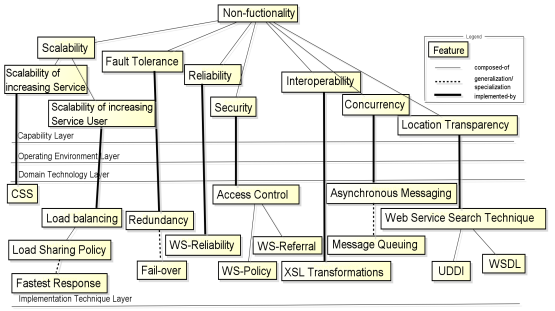

図 6　出庫受付システムのフィーチャモデル

要となる非機能特性を実現するアスペクトを選択し，出庫受付システムのシステムアーキテクチャを設計できる．設計したシステムアーキテクチャを図 7 に示す．

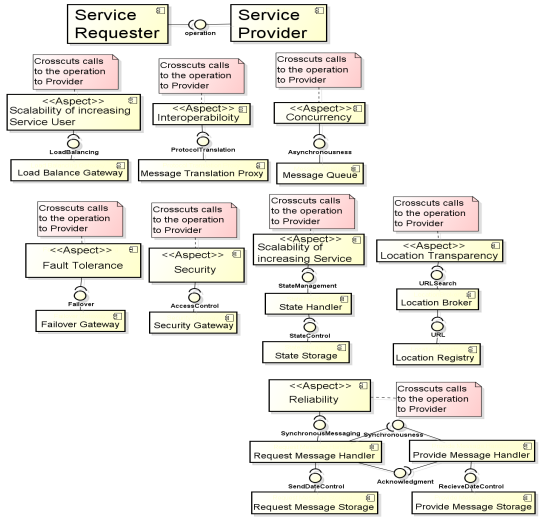

図 7　出庫受付システムのシステムアーキテクチャ

5　考察

5.1　ガイドラインに対する考察

扱う関心事を揃えることで，フィーチャモデルとプロダクトラインアーキテクチャ間の追跡性を確保した．追跡性の確保により，システムに求められる非機能特性に応じたアーキテクチャのコンポーネントを追跡することが可能となった．ガイドラインは，求められる非機能特性に応じたシステムアーキテクチャの設計を支援できると考える．

事例検証を通して，ガイドラインを利用してシステムアーキテクチャの設計を行なった．フィーチャモデルのフィーチャからプロダクトラインアーキテクチャ上のアスペクトを追跡することが出来た．事例で扱ったフィーチャからアスペクトへの追跡を図 8 に示す．

ガイドラインの利用により，非機能特性を表すフィーチャの選択から非機能特性を実現するシステムアーキ

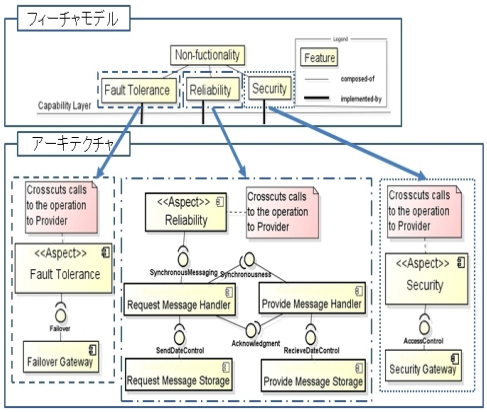

図 8　フィーチャからアスペクトへの追跡

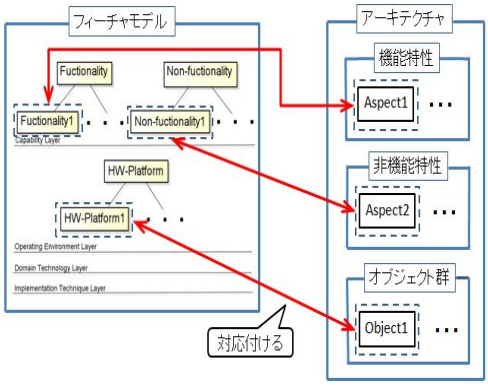

図 9　組込みシステムにおける対応関係

テクチャの設計が可能であることを確認した．ガイドラインが，非機能特性を考慮した SOA に基づくシステムアーキテクチャの設計を支援可能であり，妥当であると考える．

5.2　追跡性確保の方法に対する考察

5.2.1　追跡性確保の方法の妥当性

本研究では，アーキテクチャ設計にアスペクト指向技術を適用し，アーキテクチャと仕様モデルが扱う関心事を揃えることで追跡性を確保した．仕様モデルとアーキテクチャが扱う関心事を揃える代替案として，アーキテクチャで扱う関心事を考慮し仕様モデルを拡張する方法が挙げられる．この方法には，仕様モデルの本質を損なう可能性があるという問題がある．

アーキテクチャ設計にアスペクト指向技術を適用し，仕様モデルと関心事を揃えることで仕様モデルやアーキテクチャの本質を損なわず追跡性の確保が可能である．本研究の追跡性確保の方法は妥当であると考える．

5.2.2　追跡性確保の方法の一般性

組込みシステムを対象とし，FORM に基づくフィーチャモデルとアーキテクチャ間の追跡性の確保に本研究で用いた方法が適用可能であるか考察する．一般に，組込みシステムの支配的分割はオブジェクト指向である．組込みシステムのアーキテクチャと仕様モデルで共通して扱う関心事は，機能特性，非機能特性，オブジェクトである．組込みシステムのアーキテクチャ上のオブジェクトはフィーチャモデル上のハードウェアプラットフォームと対応付く．組込みシステムの支配的分割であるオブジェクト指向に対し，機能特性と非機能特性は横断的関心事となるので，アスペクトとして分離する．機能特性と非機能特性を実現するコンポーネントをアスペクトとして分離することにより，フィーチャモデル上の Capability Layer のフィーチャと対応付く．組込みシステムにおけるフィーチャモデルとアーキテクチャの対応関係を図 9 に示す．

組込みシステムにおいても，アスペクト指向技術により仕様モデルとアーキテクチャ間の追跡性確保が可能であり，本研究で用いた方法は一般性があると考える．

6　おわりに

本研究は非機能特性を実現する SOA に基づくシステムアーキテクチャ設計のガイドラインの提案を行なった．PLSE の概念に基づき，非機能特性を考慮した仕様モデルと非機能特性を実現するプロダクトラインアーキテクチャを定義した．アーキテクチャ設計にアスペクト指向技術を適用することで仕様モデルとプロダクトラインアーキテクチャ間の追跡性を確保した．追跡性を確保した仕様モデルとプロダクトラインアーキテクチャをシステムアーキテクチャ設計のガイドラインとして提案した．ガイドラインを提案することで，非機能特性を考慮した SOA に基づくシステムアーキテクチャの設計支援を実現した．

参考文献

[1] A. NyBen, S. Tyszberowicz, and T. Weiler, “Are Aspects useful for Managing Variability in Software Product Lines? A Case Study,” *Early Aspects Workshop at SPLC 05,* pp. 1-7, 2005.

[2] D. Georgakopoulos, and M. Papazoglou, *Service-Oriented Computing,* The MIT Press, 2009.

[3] K. C. Kang, S. Kim, J. Lee, K. Kim, G. J. Kim, and E. Shin, “FORM:A Feature-Oriented Reuse Method with Domain-Specific Reference Architechtures,” *Annals of Software Engineering,* vol. 5, no. 1, pp. 143-168, 1998.

[4] K. Pohl, G. Bockle, F. Linden, *Software Product Line Engineering,* Springer, 2005.

[5] M. Shaw, D. Garlan, *Software Architecture: Perspectives on an Emerging Discipline,* Prentice Hall, 1996.

[6] P. Clements, F. Bachmann, L. Bass, D. Garlan, J. Ivers, R. Little, P. Merson, R. Nord, and J. Stafford, *Documenting Software Architectures: Views and Beyond,* Addison-Wesley, 2010.