ＳＯＡアプリケーションプラットフォームのプロダクトライン化に関する研究

―アスペクト指向に基づくプロダクトアーキテクチャの自動生成―

2010SE015 伴昌樹　2010SE281 吉川翔平

指導教員：野呂昌満

1　はじめに

本研究室では SOA に基づくシステムのためのアプリケーションプラットフォームを提案し，そのプロダクトライン化に関する研究 [4] を行なっている． SOA アプリケーションプラットフォームのプロダクトライン化ではプロダクトアーキテクチャの自動生成が目標とされている．プロダクトアーキテクチャを自動生成するには仕様モデルとアーキテクチャの対応関係を明確にし，追跡性を確保することが重要である．仕様モデルにはプロダクトラインアーキテクチャの Variation Point(変動点) に対応した選択肢が記述してある．Variation Point には Variant があり，その Variant の組み合わせによってプロダクトアーキテクチャの Component が決定する．

SOA アプリケーションプラットフォームのプロダクトラインの仕様モデル，アーキテクチャの関係は不明確である． 仕様モデルとアーキテクチャの関係が不明確であると，追跡性が確保できずプロダクトアーキテクチャの自動生成が困難である．

本研究の目的は，アプリケーションプラットフォームのプロダクトラインにおける仕様モデルとアーキテクチャの対応関係を明確にすることである． 対応関係を明確にすることで，追跡性が確保でき，プロダクトアーキテクチャ自動生成の基礎となる．

本研究の目的達成のためのアイディアとして以下の 3 つのことがあげられる．

- Variation Point の再整理
- Variation Point と仕様モデルの対応関係整理
- Variation Point とアーキテクチャとの対応関係の整理

Bass，Bachmann[1] を参考に，Case Study に基づいて Variation Point を整理した．Feature Oriented Reuse Method[2](以下，FORM) の仕様モデルの分類と Bass らの研究で定義される Variation Point の分類の対応関係を整理した．FORM と Bass らの研究の対応関係に基づいて整理した Variation Point を仕様モデルに配置し，仕様モデルを生成した．Variation Point とアプリケーションプラットフォームのプロダクトラインの Concern によって定義される Component(以下，アスペクト) との対応関係を整理した．アスペクトを決定付ける Variation Point を整理した．最後に事例検証を行なうことで，本研究の妥当性を確認した．

本研究の成果として，SOA アプリケーションプラットフォームの仕様モデルとアーキテクチャの対応関係を明確にしたことで，追跡性を確保できた． 仕様モデルとアーキテクチャの追跡性が確保することにより，プロダクトアーキテクチャの自動生成の基礎となる部分が提示できた．

2　SOA アプリケーションプラットフォームのプロダクトライン

2.1　プロダクトラインアーキテクチャ

プロダクトラインアーキテクチャはすべての SOA システムに共通なアプリケーションプラットフォームの基本的な構造を示したものである．プロダクトラインアーキテクチャは以下の 2 つで構成されている．

- アスペクト間の関係
- アスペクトにおける Variation Point

本研究で定義したアスペクト間の関係を図 1 に示す．

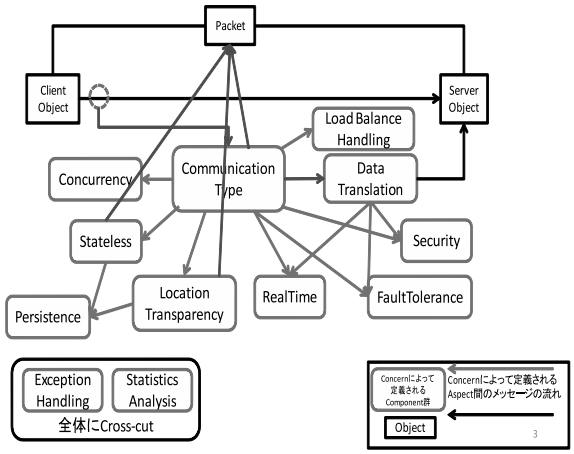

図 1　アスペクト間の関係

Variation Point には Variant(候補となる値) があり，その複数の Variant の組み合わせにより，アスペクトが決定する．

2.2　プロダクトアーキテクチャ

プロダクトアーキテクチャはプロダクト特有の要求を反映させたアーキテクチャである．プロダクトラインフィーチャモデルは要求を入力として生成することを想定している．Variation Point のとる Variant によって，プロダク

トアーキテクチャは異なる．

2.3 プロダクト自動生成の課題

プロダクトの自動生成を実現するための課題として，プロダクトアーキテクチャの自動生成がある．プロダクトアーキテクチャ自動生成を実現するための課題は，仕様モデルとアーキテクチャの対応関係を明確にすることがある．プロダクトラインアーキテクチャには複数の Variation Point があり，Variant の組み合わせによって Component が決定する．仕様モデルとアーキテクチャの対応関係を明確にするための課題として以下の 3 点が挙げられる．

- Variation Point の整理
- Variation Point と仕様モデルの対応関係を整理
- Variation Point とアスペクトの対応関係を整理

3 Variation Point の整理

アプリケーションプラットフォームのプロダクトラインの Variation Point の定義は Case Study に基づき，Bass らの論文を参考に行なった．Variation Point の定義は Bass，FORM のどちらでも行なっているが，FORM の分類は粒度が大きく Variation Point を識別するのが困難であったので，Bass を参考に Variation Point の定義を行なった．Bass らの研究では，Variation Point は 6 つの種類に分けられる．

- Function
- Data
- Control Flow
- Technology
- Quality Goal
- Environment

Variation Point とその Variant，Variation Point 同士の依存関係の一部を表 1 に示す．

表 1 Variation Point の一部

分類	Variation Point	Variant(実現候補)	依存関係
Function	送信方法	1 対 1 通信，1 対多通信	
Data	セッションデータ	有，無	Session 管理
Control Flow	障害回復処理	Retry，Recovery，リカバリブロック，N-Version，Programming	複数回実行
Technology	オペレーティングシステム	Linax，MacOS	
Quality Goal	合目的性	非機能特性の優先度合い	
Environment	言語	Java，C	

4 Variation Point と仕様モデルとの対応関係

本研究室で提案されている仕様モデルは，アーキテクチャの自動生成が考慮されていない．アーキテクチャと仕様モデルの対応関係を明確にするために仕様モデルの見直しを行なった．アーキテクチャと仕様モデルの対応関係を明確にするためには，Variation Point と仕様モデルの対応関係を明確にする必要がある．Variation Point と仕様モデルの対応関係を明確にするために，まず Bass で整理した Variation Point の分類と FORM の分類との対応関係を整理した．

Bass らの研究で定義している 6 種類の Variation Point を FORM で定義している 4 層にそれぞれ分類した．この分類に基づいて Variation Point に対応するフィーチャを仕様モデルに定義した．そのさい，機能フィーチャと関係のあるフィーチャを整理した．フィーチャモデル上に記述可能なものを関連線を用いて記述した．フィーチャモデル上に記述できないものを，ジェネレーティブプログラミング [3] で定義されている補足情報として記述した．この関連線と補足情報によってアスペクトごとに決定すべき Variation Point の関係が明確化できる．Bass の分類と FORM の分類の関係を表した図が図 2 となる．

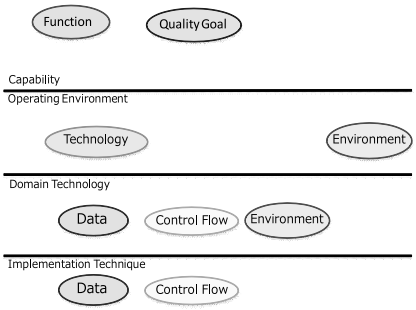

図 2 Bass の分類と FORM の分類の関係

Bass の分類と FORM の分類の関係の図に基づいて Variation Point を仕様モデルに配置した．生成した仕様モデルの一部が図 3 となる．

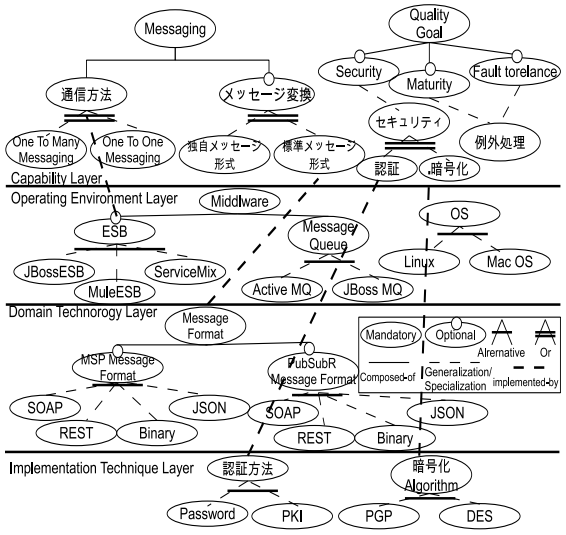

図 3 仕様モデルの一部

5 Variation Point とアスペクトとの対応関係

Variation Point とアーキテクチャの関係を対応明確にするために，アスペクトを決定する Variation Point を整理した．

Variation Point とアスペクトの対応関係，アスペクトを決定する Variation Point の整理を Data Translation の例で説明する．メッセージ変換に関する Variation Point とアスペクトの関係を整理した．Data Translation において整理した Variation Point が表 2 となる．

表 2 Data Translation における Variation Point

Variation Point	Variant (候補となる値)
Function. メッセージ形式	独自メッセージ形式変換/ 標準メッセージ形式変換/ instanceSerialization 無
Data.Message Format	SOAP/REST/JSON /Binary/RMI
Technology. Middleware	JIBX/CXF/ MessageFormat/Jackson/ MessageProtocol/JavaRMI

Data Translation のアスペクトを決定する Variation Point の整理をした．その図が図 4 となる．

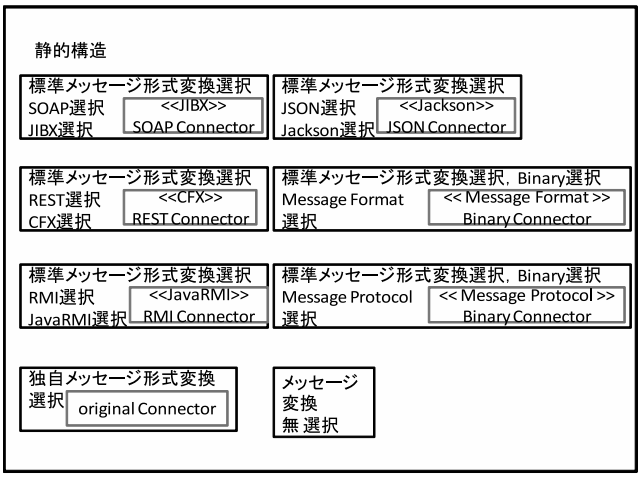

図 4 メッセージ変換のプロダクトアーキテクチャ

表 3 と図 4 と同様な整理をすべてのアスペクトに対して行なった．結果，Variation Point とアスペクトとの対応関係を整理できた．

6 仕様モデルとアスペクトの対応関係

3.1 章から 3.3 章までの調査により仕様モデルとアスペクトの対応関係を整理できた．整理した対応関係を元に仕様モデルのサブフィーチャの選択と，プロダクトアーキテクチャの構造の対応関係を図に表した．整理した対応関係を Communication Type の例で説明する．仕様モデルのサブフィーチャの選択と，プロダクトアーキテクチャの構造の対応関係が図 5 となる．

Feature	Component / SubFeature	Message Service Provider	<<JBOSSESB>> Message Service Provider	<<MuleESB>> Message Service Provider	<<Service Mix>> Message Service Provider
送信方法	One To One	○	○	○	○
	One to Many	○	○	○	○
	One To One, One To Many	○	○	○	○
ESB	ESB 無	○	-	-	-
	Jboss ESB	-	○	-	-
	Mule ESB	-	-	○	-
	Service Mix	-	-	-	○

図 5 サブフィーチャの選択と，プロダクトアーキテクチャの構造の対応関係

Communication Type の Message Service Provider は送信方法の選択によって構造が変化しない．しかし ESB の選択の場合においてアーキテクチャの構造が変化する．ESB が選択された場合は選択されたミドルウェアの名前をステレオタイプで記述する．ESB が選択されなかった場合はステレオタイプの記述を行なわない．このようにサブフィーチャの選択によってプロダクトアーキテクチャの構造は変化する．

7 事例検証

仕様モデルとアーキテクチャの対応関係を明確にすることで追跡性を確保することができた．確保した追跡性の妥当性の確認のために事例検証を行なう．Variation Point の Variant を表 3 のように決定し，プロダクトアーキテクチャを生成した．プロダクトラインフィーチャモデルと Variation Point の Variant を照合して決定したプロダクトフィーチャモデルの一部を図 6 に記す．プロダクトフィーチャモデルと，6 章で定義した対応関係から作成したプロダクトアーキテクチャが図 7 となる．

8 考察

8.1 Variation Point の整理と仕様モデルの生成

Variation Point と仕様モデルの関係を明確にするために，CaseStudy に基づき，Bass の研究を参考にして Variation Point の分類を行なった．その分類と FORM の 4 層の分類を参考に Variation Point に対応したフィーチャを仕様モデルに配置した．結果，Variation Point の分類を明確にし，仕様モデルを生成できた．

新たな Variation Point が追加された場合，本研究と同様に Variation Point を分類し，Variation Point に対応するフィーチャを仕様モデルに配置する必要がある．フィー

表 3 決定した Variation Point の Variant

Variation Point	Variant(実現候補)
Function. 送信方法	One To One One To Many
Function. メッセージ変換	無
Function. 並列化	有
Function. 負荷分散	無
Function. サービス特定.	ObjectID 特定
Function.Data 管理	永続化無 Session 管理無
Middleware.MassageQueue	JBossESB
Environment	Local Object

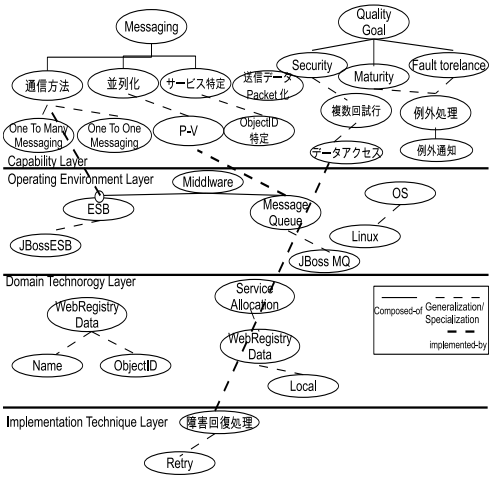

図 6 プロダクトフィーチャモデルの一部

チャを仕様モデルに配置するさい，フィーチャ間の依存関係も記述する必要がある．

8.2 仕様モデルとアーキテクチャの対応関係の考察

仕様モデルとアーキテクチャの対応関係を明確にするために，Variation Point の Variant の決定がアーキテクチャに与える影響について調査した．章 5 の図 4 と表 2 をアスペクトごとに作成することで，仕様モデルとアーキテクチャの対応関係が明確になった．対応関係が明確になったことで，仕様モデルとアーキテクチャの追跡性が確保できた．プロダクトアーキテクチャ自動生成の基礎となる部分になる．

新たに Variation Point が追加された場合，新たな Variation Point と以下の 2 つの関係を調査する必要がある．

- プロダクトラインアーキテクチャと仕様モデルの機能フィーチャとの対応関係
- Variation Point の Variant の決定によって変化するプロダクトアーキテクチャの構成

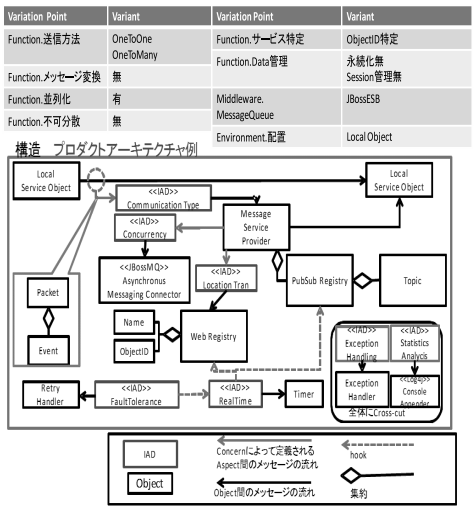

図 7 プロダクトアーキテクチャ

8.3 事例検証

プロダクトアーキテクチャの生成の事例検証を行なった．プロダクトフィーチャモデル，プロダクトラインアーキテクチャと対応関係の表を照らし合わせ，プロダクトアーキテクチャを生成した．プロダクトフィーチャモデル，対応関係の表を参考にプロダクトアーキテクチャを生成することが可能であり，妥当性が確認できた．

9 おわりに

Variation Point，仕様モデル，アーキテクチャの対応関係を明確にしたことにより，プロダクトアーキテクチャの自動生成の基礎ができた．プロダクトアーキテクチャの自動生成を実現するには，本研究で明確にした対応関係を機械処理可能な形式で記述し，生成ルールを提示することが課題としてあげられる．

参考文献

[1] F. Bachmann, L. Bass, ”Managing Variability in Software architectures,” *ACM SIGSOFT Software Engineering Notes,* vol. 26, no. 3, pp. 126-132, 2001.

[2] 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 Architectures,” *Annals of Software Engineering,* vol. 5, no. 1, pp. 143-168, 1998.

[3] K. Czarnecki, and U. W. Eisenecker, *Generative-Programming: Methods, Tools, and Applications,* Addison-Wesley, 2000.

[4] 江坂篤侍，野呂昌満，沢田篤史，“SOA に基づくシステムのためのアプリケーションプラットフォームのプロダクトライン化に関する研究，”情報処理学会研究報告. ソフトウェア工学報告，vol.2013-SE-179，no. 25，pp. 1-6，2013.