アスペクト指向ソフトウェアアーキテクチャに基づくPLSEに関する研究
–携帯電話制御ソフトウェアを例にして–

2004MT034 伊藤 晴香　　2004MT121 山口 さやか
指導教員　野呂 昌満

1　はじめに

携帯電話制御ソフトウェアの多機能と複雑化が進んでおり, それに対応するために開発時間の短縮化と系統的な開発方法が求められている. ソフトウェアを系統的に開発する方法の一つとして, プロダクトラインソフトウェアエンジニアリング (以下, PLSE)[2] が提案されている. PLSE を支援する開発方法論の一つとして, ユーザ要求のモデル化をおこなうためにフィーチャ指向プロダクトラインエンジニアリング (以下, FOPLE)[1] が提案されている. PLSE の問題点として, ドメイン工学における仕様モデルとアーキテクチャ間の関係が不明瞭であることが挙げられる.

本研究室では, 組込みソフトウェアにおけるアスペクト指向ソフトウェアアーキテクチャスタイル (以下, E-AoSAS++) を提案している. また, 仕様モデルとソフトウェアアーキテクチャの対応関係を明確にするために, PLSE に E-AoSAS++ を適用することを提案している. 昨年の研究では, フィーチャとアーキテクチャのコンポーネントを 1 対 1 で対応づけたが, 不十分であった. 本研究では, 仕様モデルとアーキテクチャとの対応が不明瞭であるという PLSE の問題点を解決するために, 携帯電話制御ソフトウェアを事例として, 仕様モデルとアーキテクチャとの対応関係を明確にすることである. 仕様モデルであるフィーチャ図とアーキテクチャの関係に仮説を立て, 仕様モデルとアーキテクチャの対応関係について考察をおこなった. その結果, フィーチャとアーキテクチャの構成要素との関係を明確にすることができた.
本研究は，以下のように進める.

- 既存の論文からフィーチャ図の問題点を挙げ, 解決策の提案
- 携帯電話制御ソフトウェアの分析
- フィーチャ図とアーキテクチャの関係について仮説の提案と事例検証
- 対応関係の一般性に関する考察

2　関連研究

関連研究として，PLSE, FOPLE, E-AoSAS++,UMP, プロダクトラインアーキテクチャ, プロダクトアーキテクチャについて説明する.

2.1　PLSE

PLSE とは, 系統的なソフトウェア開発を支援する手法の一つとして提案されており, 再利用性を重視した開発手法である.
PLSE を適用することにより, ドメインに共通して使用できる資産を蓄え, 製品の生産性の向上をはかることができる.

2.2　FOPLE

FOPLE とは, PLSE を支援する開発方法論の一つであり, フィーチャモデリングを提案している. フィーチャとは, システムを開発する側の視点からみた実現可能なユーザ要求である. フィーチャ図では, フィーチャを 4 階層にわけ, ユーザ要求の整理をおこなう. フィーチャ図の 4 階層を次に示す.

- 特性層
  組込みソフトウェアの機能特性や非機能特性
- 操作環境層
  組込みソフトウェアのハードウェアに関する部分
- ドメイン技術層
  ドメインに特化した技術
- 実現技術層
  一般的なソフトウェアなどの実現に利用される技術

FOPLE で提案されているフィーチャ図を図 1 に示す.

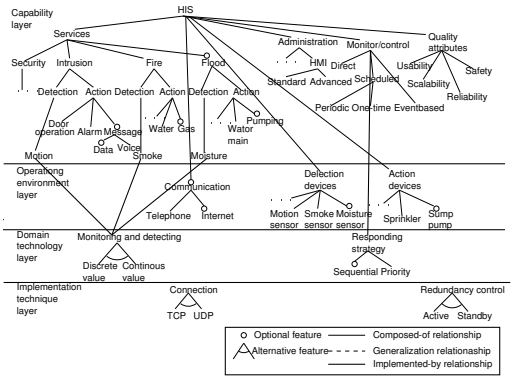

図 1　HIS のフィーチャ図

2.3　E-AoSAS++

E-AoSAS++ は, 本研究室が提案している組込みソフトウェアにおけるアスペクト指向ソフトウェアアーキテクチャスタイルである. E-AoSAS++ では, 組込みソフトウェアアーキテクチャを並行に動作する状態遷移機械の集まりとして規定する.

2.3.1　UMP

UMP はモジュール化の手法である. UMP を用いることによって, 複数の CSTM の協調動作をポリシーによる

CSTM の構成の切替えによって記述することができる.

2.3.2 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とは, ソフトウェアファミリ全体に共通したアーキテクチャである. プロダクトラインアーキテクチャは UMP の記述法とコンポーネント図を用いて現される.

2.3.3 プロダクトアーキテクチャ

プロダクトアーキテクチャとは, 個々のソフトウェアを開発をする際に, プロダクトラインアーキテクチャから必要なコンポーネントを抽出することで構築されるアーキテクチャである.

3 フィーチャ図の再定義

FOPLE の論文 [1] で, 提案されているフィーチャ図の問題点を明らかにし, 再定義をおこなう.

3.1 FOPLE のフィーチャ図の問題点

FOPLE のフィーチャ図の記述法を基にフィーチャ図を作成すると, 次の問題点が挙げられる.

- どのようなものが機能となるかフィーチャの定義が曖昧である
- 図 1 に示したように使用性や安全性などの非機能フィーチャは, 抽象度が高く, 他のフィーチャとの関連が不明瞭である

3.2 解決策

3.2.1 フィーチャの定義の明確化

既存のフィーチャ図の特性層を, 特性層とアトミック特性層の二層にわけて記述する. 二層にわけて記述することによって, FOPLE で提案されているフィーチャ図の問題点を解決できると考えた. 新たな階層の定義は次の通りである.

- 特性層
  ユースケース図のユースケースとそれに関する非機能をフィーチャとして抽出する
- アトミック特性層
  特性層に現れたフィーチャを実現するのに必要なシステムの要求をフィーチャとして抽出する

3.2.2 非機能フィーチャの詳細化

非機能フィーチャの詳細化をおこなうことで, 我々は, 非機能フィーチャが機能フィーチャの関連を記述することが可能になるのではないかと考えた. 詳細化の基準として, 非機能フィーチャを実現するコンポーネントを, アーキテクチャ上におけるコンポーネントとの関連が分かるまで 非機能フィーチャの詳細化をおこなう. 図 2 に詳細化の基準の例を示す.

4 携帯電話制御ソフトウェアの分析

提案したフィーチャ図に基づいて携帯電話制御ソフトウェアを例に, フィーチャ図を作成し, 分析をおこなう.
携帯電話には, 多数の機能が存在している. 本研究では, そのうちの 通話機能, メール機能, アドレス帳機能, カメラ撮影機能についてあつかう.
非機能の特性として, 安全性と実時間性を例に挙げる.

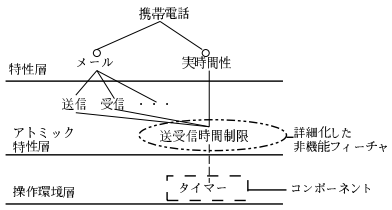

図 2 非機能の詳細化の例

4.1 携帯電話のフィーチャ図

提案したフィーチャ図に基づいた携帯電話制御ソフトウェアを構成するフィーチャ図を**図 3** に示す.

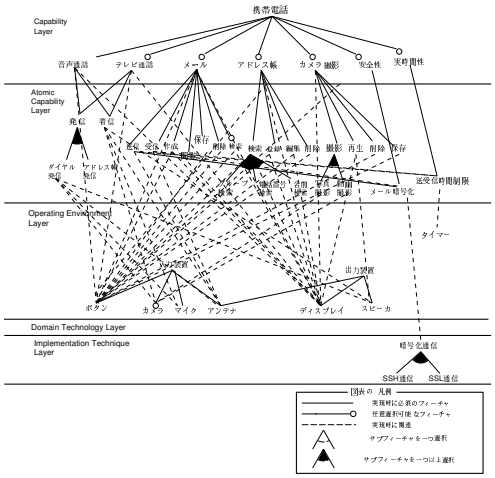

図 3 携帯電話のフィーチャ図

携帯電話に必須なフィーチャを音声通話として示している. メールやアドレス帳など, 他のフィーチャは, オプションになっており, 任意選択可能である発信とアドレス帳検索に現れるフィーチャはオアフィーチャとした.

5 アーキテクチャの構成要素とフィーチャの関係

携帯電話制御ソフトウェアを事例に前章での記述例に基づきフィーチャ図とプロダクトラインアーキテクチャの対応関係について考える.E-AoSAS++ のアーキテクチャはアスペクトをモジュールとするので対応関係を考える際に, 以下の仮説を立てる.

> フィーチャ図の特性層, アトミック特性層に現れるフィーチャはアスペクトとして扱う.

仮説に基づいてプロダクトラインアーキテクチャを構築したところ, 全てのフィーチャをアスペクトとして捉えることで, アーキテクチャが複雑になるという問題が明らかになった. どのようなフィーチャをアスペクトにする必要があるのかについて考える必要があり, その選択基準が必要となる.

5.1 アーキテクチャの構成要素と操作環境層のフィーチャとの関係

アーキテクチャの構成要素と操作環境層のフィーチャとの関係を**図 4** に示す.

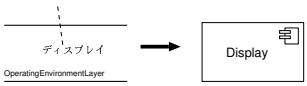

図 4　アーキテクチャの構成要素と操作環境層のフィーチャとの関係

操作環境層に現れるフィーチャはオブジェクト指向設計した際のオブジェクトとなるのでハードウェアコンポーネントに 1 対 1 で対応する. 例えば ディスプレイフィーチャは Display コンポーネントに対応する.

5.2　アーキテクチャの構成要素と特性層, アトミック特性層の必須フィーチャとの関係

アーキテクチャの構成要素と特性層, アトミック特性層の必須フィーチャとの関係について考える. 文献 [3] を参考に, 横断的コンサーンをアスペクトとして取り扱う. 横断的コンサーンは同種 (homogeneous) と異種 (heterogeneous) の 2 つに分類される. 同種横断 (例, ロギング) は, 複数のコンポーネントの異なるジョインポイントに同じコードが散在している. 一方, 異種横断 (例, サービス) は, 異なるコードが散在している.

5.3　横断的コンサーン：同種 (homogeneous)

同種横断 (homogeneous crosscut) は図 5 のフィーチャに対応する.

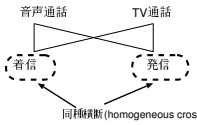

図 5　同種横断 (homogeneous crosscut)

着信, 発信を実現するコードがそれぞれ音声通話,TV 通話を実現するコンポーネントに分散されている. このようなフィーチャをアスペクトとする. フィーチャ図では, 図 5 のように複数のフィーチャが 1 つのサブフィーチャを共有するような形になっている場合, サブフィーチャがアスペクトとなる.

5.4　横断的コンサーン：異種 (heterogeneous)

携帯電話制御ソフトウェアを例にした際, 異種横断 (heterogeneous crosscut) となるフィーチャを図 6 に示す.

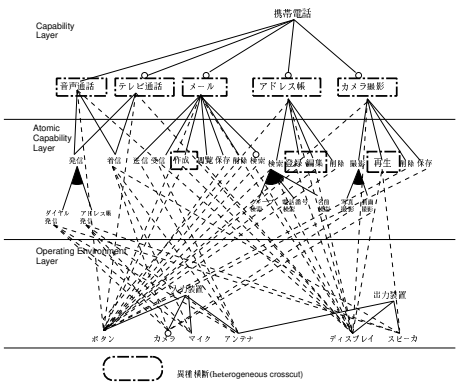

図 6　携帯電話の異種横断 (Heterogeneous crosscut)

音声通話,TV 通話, メールフィーチャなどはそれぞれを実現する際に複数のハードウェアコンポーネントに横断している. このようなフィーチャをアスペクトとする. フィーチャ図で異種横断となるフィーチャは複数のハードウェアと関連して記述されている. このようなフィーチャがアスペクトとなる.

5.5　アーキテクチャの構成要素とオルタネイティブ, オア, オプショナルフィーチャとの関係

ユーザ要求の変更に伴って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構築するには, 選択されるフィーチャとコンポーネントが 1 対 1 で対応している必要がある. よって, オア, オルタネイティブ, オプショナルフィーチャはアスペクトとしてコンポーネントと 1 対 1 で対応させる.

オルタネイティブ, オア, オプショナルフィーチャとプロダクトラインアーキテクチャの対応関係を図 7 に示す.

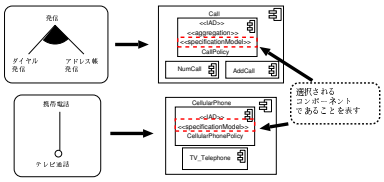

図 7　アーキテクチャの構成要素とオルタネイティブ, オア, オプショナルフィーチャとの関係

オアフィーチャである発信フィーチャは Call コンポーネント, ダイヤル発信フィーチャは NumCall コンポーネント, アドレス帳発信フィーチャ AddCall コンポーネントにそれぞれ対応する. オプショナルフィーチャ, オルタネイティブフィーチャもオアフィーチャと同様に 1 対 1 でフィーチャとコンポーネントを対応させる.

また, オア, オルタネイティブ, オプショナルフィーチャなど選択するフィーチャに対応するコンポーネントを管理する Policy コンポーネントにステレオタイプとして SpecificationModel を付加することによって管理されるコンポーネントが選択であることを示す.

5.6　アーキテクチャの構成要素と非機能フィーチャとの関係

非機能フィーチャは横断的に現れるのでアスペクトとして取り扱う. しかし,FOPLE の非機能フィーチャをアスペクトとした際, 関連がわからないので,3.2 節で示した解決策に基づいて, 非機能フィーチャの詳細化をおこなう.

携帯電話制御ソフトウェアを例にした場合, 非機能フィーチャとアーキテクチャの構成要素との関係を図 8 に示す.

実時間性はメール送受信時間制限に詳細化され, タイマを必要とすることがわかった. タイマは操作環境層のフィーチャなのでコンポーネントになる. 実時間性はタイマコンポーネントに処理を記述することで実現可能である. また安全性を詳細化すると暗号化通信である SSH 通信,SSL 通信で実現できることがわかった. 安全性の例では関連している送信、受信コンポーネントに処理を追

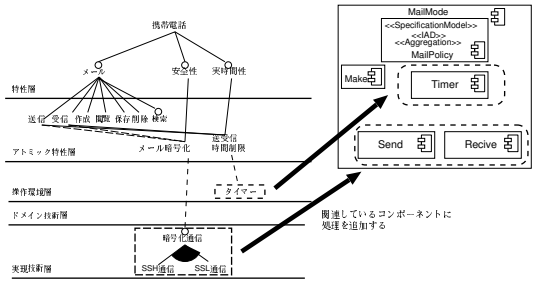

図 8　アーキテクチャの構成要素と非機能フィーチャとの関係

加することで実現可能である．携帯電話を例にした際, 非機能フィーチャをアスペクトとして記述しなくても実現可能であることがわかった.

6　考察

携帯電話制御ソフトウェアの事例を基に, プリンタ制御ソフトウェアを例に対応関係の一般性の考察をおこなう.

6.1　操作環境層のフィーチャに関する考察

操作環境層のフィーチャに関する考察をおこなう. 操作環境層のフィーチャはコンポーネントと 1 対 1 で対応した. 操作環境層のフィーチャはオブジェクト指向設計をした際のオブジェクトとなる. 操作環境層のフィーチャはハードウェアであり他のドメインにおいてもオブジェクトとなり, コンポーネントになると考える.(図 9)

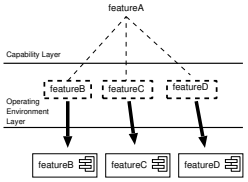

図 9　操作環境層のフィーチャ

6.2　特性層, アトミック特性層に関する考察

アーキテクチャの構成要素と特性層, アトミック特性層のフィーチャとの考察をおこなう.

6.2.1　必須フィーチャに関する考察

必須フィーチャの中でアスペクトとするフィーチャは 2 種類あり, 同種と異種がある．プリンタ制御ソフトウェアにおいても同様の形となった．フィーチャ図の形上, 横断的に現れるフィーチャを見付けることは可能であると考える. フィーチャ図上で横断的コンサーンとなるフィーチャの例を図 10 に示す.

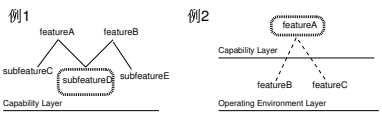

図 10　フィーチャ図上で横断的コンサーンになるフィーチャ

6.2.2　オア, オルタネイティブ, オプショナルフィーチャに関する考察

オア, オルタネイティブ, オプショナルフィーチャは, 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構築する際に, ユーザ要求によって選択される. よって, オア, オルタネイティブ, オプショナルフィーチャはアスペクトとしてコンポーネントと 1 対 1 で対応させることが必要であると考えた.

6.3　非機能フィーチャに関する考察

携帯電話制御ソフトウェアの例では, 非機能の詳細化をおこない, 関連している機能に処理を追加することで実現可能であることがわかった．プリンタ制御ソフトウェアの信頼性の例を図 11 に示す.

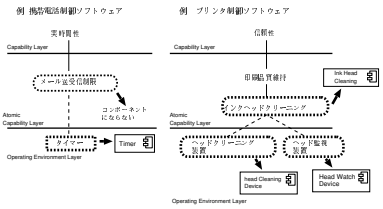

図 11　プリンタ制御ソフトウェアの例

プリンタ制御ソフトウェアの例では非機能フィーチャを詳細化した際に, 非機能を実現する機能が現れた. 図 11 では信頼性を実現するにはインクヘッドクリーニング機能が必要となることがわかった. このような場合は新たに現れた機能フィーチャをアーキテクチャのコンポーネントに対応させ実現する．非機能フィーチャはコンポーネントに対応する場合と対応しない場合が存在する．非機能フィーチャはアーキテクチャの構成要素との対応関係は不明確であり, 開発者がアーキテクチャを構築する際, 開発者の経験則に基づいて考える必要がある.

7　おわりに

本研究では，携帯電話制御ソフトウェアを PLSE に適用し，さらに，再利用性・柔軟性を高める目的で，E-AoSAS++ を適用してアーキテクチャを構築した．アーキテクチャの構成要素とフィーチャとの関係について考察をおこなった. 今後の課題として, フィーチャ図からアーキテクチャの自動生成ツールの作成があげられる.

謝辞

本研究を進めるにあたり，熱心な御指導を頂いた野呂昌満教授，沢田篤史教授，蜂巣吉成講師，有益なアドバイスを下さった大学院生の皆様に深く感謝いたします.

参考文献

[1] K. C. Kang, J. Lee, and P. Donohoe, “Feature-Oriented Product Line Engineering,”IEEE Software, Vol.19, No. 4, pp. 58-65, July/August.2002.

[2] Linda M. Northrop, “SEI’s Software Product Line Tenets”,IEEE Software, Vol.19, No. 4, pp. 32-640, July/August.2002.

[3] Kwanwoo Lee, Kyo C. Kang, Minseong Kim, Sooyong Park, “Combining Feature-Oriented Analysis and Aspect-Oriented Programming for Product Line Asset Development”, IEEE Software, 2006.