Vol.2014-SE-183 No.3
2014/3/19

ユースケースからのフィーチャモデル導出の提案

田原圭祐[†1]　　野田夏子[†1]

ソフトウェアプロダクトライン開発では共通性と可変性をモデル化するためにフィーチャモデルを用いるが、開発初心者では作成が難しい。その問題を解決するためにプロダクトラインに含まれるプロダクトごとに作成したユースケース群からフィーチャモデルを導出する手法を提案する。本手法では、個々のユースケース図からフィーチャ候補、フィーチャの共通性と可変性を導出し、同じ振る舞いのユースケースのユースケース記述を比較して差異から可変性の情報を導出する。提案手法をもとにツールを実装し、例題に手法を適用したところ、フィーチャモデルの導出を確認できた。

A Method of Deriving Feature Models from Use Cases

KEISUKE TAHARA[†1]　NATSUKO NODA[†1]

This paper proposes a systematic approach to derive a feature model of a software product line from use cases of the members of the product line. In software product line development, feature model is used to represent commonality and variability. But it is difficult to create the feature models for novice developers. In order to solve the problem, we introduce a method to derive a feature model from use cases of the members of the product line. In our approach, the common and variable feature candidates are derived from the use case diagrams of the same name use case of different member products are utilized to derive the information of variability. We implemented a tool based on the proposed method, and we confirmed that feature models are derived from use cases.

1. はじめに

ソフトウェア開発では、ソフトウェア資産の再利用が重要となる。ソフトウェア資産の再利用は古くから広く行われてきたが、場当たり的な流用はソフトウェア生産性向上にそれほど寄与しないばかりか、管理されないバリエーションや重複の増大、コードの複雑化、アーキテクチャの不整合などの問題を生む。

こうした問題の解決のために、ソフトウェアプロダクトライン開発が注目されている[1]。ソフトウェアプロダクトライン開発では、プロダクトラインの共通性と可変性を表現するためフィーチャモデルが使用されるが、開発初心者には作成が難しい。そこで本研究では、フィーチャモデル作成を支援するために、ユースケース図とユースケース記述からフィーチャモデルを生成する方法を検討する。

本論文の構成は以下の通りである。第 2 章で、背景となるソフトウェアプロダクトライン開発とその適用における問題を述べ、本研究の目的について述べる。第 3 章でフィーチャモデルとユースケースについてそれぞれ説明する。第 4 章で提案手法の説明に用いる例題について説明し、第 5 章で提案手法について述べる。第 6 章で手法を実装したツールとそのツールを用いた実験を行う。第 7 章で関連研究について述べ、最後に第 8 章でまとめる。

†1 芝浦工業大学
Shibaura Institute of Technology

2. 背景と目的

2.1 背景

プロダクトラインとは、共通の管理された特徴を持ち、特定のマーケットやミッションのために、共通の再利用資産に基づいて作られる、ソフトウェア集約的なシステムの集合のことである[1]。

プロダクトライン開発においては、プロダクトラインに含まれる全てのプロダクトに共通する事柄(共通性)と、プロダクトごとに変化しうる事柄(可変性)とを明確にとらえることが重要である。こうした共通性と可変性を明示的にモデル化する手法がいくつか提案されている。Kang らは、要求管理の単位であるフィーチャを識別し、共通性・可変性をモデル化する手法として FODA(Feature-Oriented Domain Analysis)を提案し、またモデル化する記法としてフィーチャモデルを提案した[2]。フィーチャモデルは、末端利用者から可視な特徴であるフィーチャを階層的に表すモデルであり、製品間で共通する要求と異なる要求を明確に分離し可視化できることから、多くのプロダクトライン開発で採用されている。しかし、適切なフィーチャを見つけるのが困難であるなどの理由からプロダクトライン開発に新規に取り組む開発者等にとっては、フィーチャモデルを作成することは難しい。

2.2 目的

フィーチャモデルの作成は、共通性と可変性を明確にとらえる必要があるプロダクトライン開発においては非常に重要である。しかし、フィーチャモデルを作成することはプロダクトライン開発初心者にとっては難しい。そこで、

本研究ではフィーチャモデル作成を支援するために、プロダクトラインのメンバである各プロダクトについて個別にユースケースを整理し、それらのユースケースの集合からフィーチャモデルの素案を生成する方法を考える。

フィーチャモデルはプロダクトラインに対する要求を、フィーチャを単位に整理しているものである。プロダクトラインに対する要求は、各プロダクトへの要求をベースに分析できる部分があると期待される。そこで、本研究では図 1 に示すようにプロダクトの要求を整理するユースケースをベースにフィーチャモデルを導出する手法を検討する。なお、各プロダクトへの要求を集めただけではプロダクトラインへの要求の分析には不十分な点があるので、完全なフィーチャモデルを導出することは目的とせず、素案となるフィーチャモデルを導出することを目的とする。

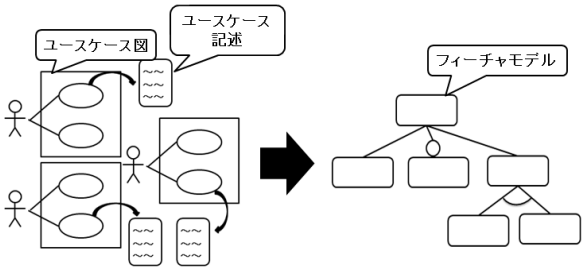

図 1 ユースケースからのフィーチャモデル導出

3. ユースケースとフィーチャモデル

本章では、ユースケースとフィーチャモデルについて説明する。ユースケースやフィーチャモデルの一般的な説明に加えて、本手法で利用する際の前提についてもあわせて説明する。

3.1 フィーチャモデル

プロダクトライン開発においてプロダクトラインの共通性と可変性を表現するためにフィーチャモデルが広く用いられている。フィーチャモデルには様々な拡張も提案されているが、本論文では FODA で提案された一般的なフィーチャモデル表現を対象にする。

フィーチャとは末端利用者から可視な特徴を表している。またフィーチャはより具体的な特徴を表すサブフィーチャを持つことができ、フィーチャモデルはツリー構造で表現される。このツリー構造のルートとなるフィーチャは製品全体を表す概念的なフィーチャであり、ルートフィーチャと呼ばれる。

フィーチャモデルはフィーチャが共通であるか可変であるかを表現するために、親フィーチャと子フィーチャの間を必須、選択、代替のいずれかの関係で繋ぐ。

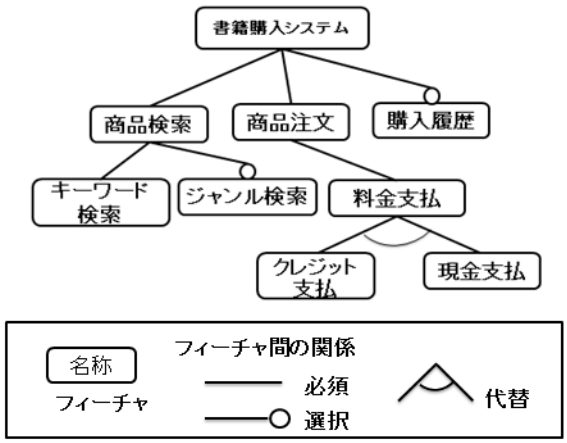

図 2 フィーチャモデル記述例

図 2 にフィーチャモデルの記述例を示す。必須は、子フィーチャが親フィーチャにとって必ず必要であることを示している。つまり、必須とされる子フィーチャは親フィーチャが存在するなら必ず存在する。選択は、子フィーチャが親フィーチャにとって選択可能であることを示している。つまり、親フィーチャが存在するとしても選択とされる子フィーチャは存在しなくてもよい。代替は 2 つ以上の子フィーチャ群との間で定義され、親フィーチャにとって子フィーチャ群のいずれか 1 つが必要であることを示している。つまり、親フィーチャが存在するならその代替とされる子フィーチャ群のいずれか 1 つが存在する。

3.2 ユースケース

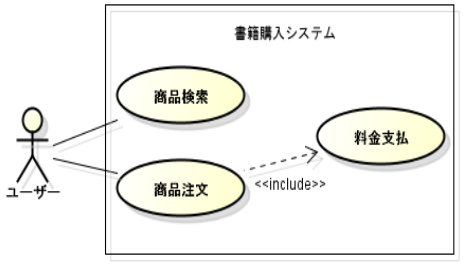

図 3 ユースケース図例

ユースケース(Use Case)とは、システムの機能要求を理解するための手段である[3]。ユースケース図は、UML で定義された図の 1 つであり、システムのユースケースの概要を示す図式表記法である[4]。図にはアクターとユースケースの間の関係だけでなく、ユースケース間の相互関係も示される。ユースケース図の例を図 3 に示す。

本稿で提案する手法では、プロダクトラインに含まれる各プロダクトの要求をまずユースケース図でモデル化する。その際に、異なるプロダクトでも同じ振る舞いのユースケースであれば同じ名前にすることとする。

ユースケース間の相互関係として、関連先のユースケースとそのユースケースをより具体化したものとして作成さ

れるユースケースの関係である汎化関係(Generalization)、関連先のユースケースに対して機能を追加するようなユースケースの関係である拡張関係(Extend)などもあるが、本研究では基本的な表現のみを対象とし、拡張と汎化は除外する。

ユースケース図を利用することでシステムが提供する機能を開発者にも利用者にも視覚的に分かりやすく表現することが出来る。しかし、個々のユースケースが具体的にどういうものであるのか、その内容を表すことが出来ない。そのために用いられるのがユースケース記述である。

ユースケース記述は、ユースケース図の中の 1 つのユースケースについて、アクターとシステムのやりとりをストーリーとして書いたものである。ユースケースを記述する方法としては様々なものがあり、テキストやシーケンス図、アクティビティ図等、いくつか存在する。次章で詳しく説明するが、本研究の提案手法では同じ振る舞いのユースケースのユースケース記述同士を比較し、差異が出る場合はその部分を抽出する必要がある。差異を抽出するにはアクターとシステム間のやりとりを部分的に抽出できることが望ましい。したがって、本研究では各ステップをアクションノードとして表現できるアクティビティ図を用いる。

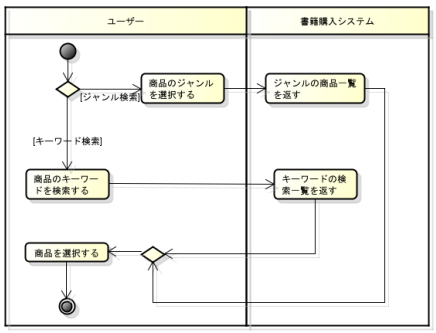

図 4 ユースケース記述例

図 4 に、アクティビティ図の例として書籍購入システムのユースケース「商品検索」のユースケース記述を示す。アクティビティ図内の開始ノードから機能が実行され、終了ノードで実行が終了する。開始ノードから終了ノードの間に存在するアクションノードは、アクターとシステムのやりとりの各ステップを表している。本研究では、ユースケース図と同様に、異なるプロダクトのアクティビティ図群の中で同じ振る舞いのアクションは同じ名前を持つという前提を置く。

4. 例題

本稿では、提案手法について例題を用いて説明する。そこで、本稿で利用する例題について本章で説明する。

図書購入システムのプロダクトラインを例題として用いる。製品 A では商品をキーワードで検索し、商品を選択して注文することが出来る。商品を注文する際に支払は現金で支払う。製品 B では商品をキーワード、またはジャンル別で検索し、商品を選択して注文することが出来る。商品を注文する際に支払はクレジットで支払う。製品 C では商品をキーワード、またはジャンル別で検索し、商品を選択して注文することが出来る。そして今まで購入した商品の履歴を確認できる。商品を注文する際に支払はクレジットで支払う。それぞれの製品について、ユースケース図を記述したものが図 5 である。

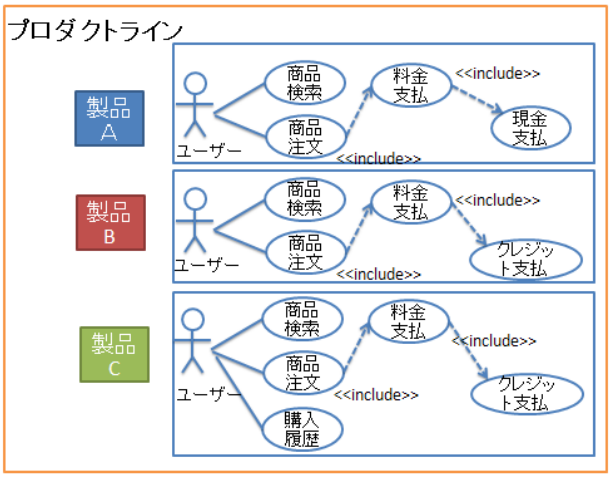

図 5 例題プロダクトラインのユースケース図

5. 提案手法

5.1 概要

プロダクトラインに含まれるプロダクトごとに作成されたユースケース図・ユースケース記述からフィーチャモデルを生成する手法を提案する。図 6 に本手法の概要を示す。

本手法では、まずプロダクトラインに含まれる複数のプロダクトのユースケース図からユースケースを抽出する。ユースケースとフィーチャには類似性があるので[5]、1 ユースケース 1 フィーチャとしてフィーチャ候補を導出する。そして、全ての製品のユースケース図に含まれるユースケースは共通部、一部の製品にしか含まれないユースケースは可変部として導出する。さらに、同名ユースケースのユースケース記述を比較して、異なるアクションを抜き出し、サブフィーチャの情報として抽出する。このようにして導出したフィーチャ情報やサブフィーチャの情報を組み合わせてフィーチャモデルを作成する。以降の節では、提案手法を構成する、フィーチャ候補の導出、フィーチャの共通性と可変性の導出、ユースケース記述の比較について順に例題を用いながら説明する。

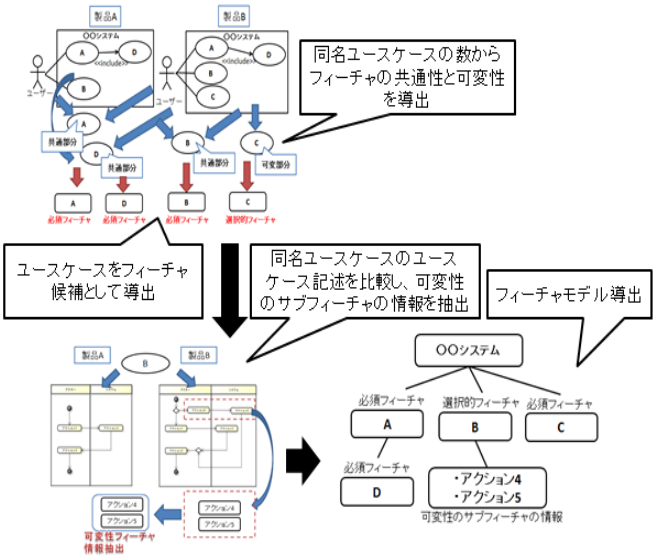

図 6 提案手法の概要

5.2 フィーチャ候補の導出

本手法ではまずプロダクトラインに含まれる各プロダクトのユースケース図のユースケースを抽出する。そして、ユースケースとフィーチャの類似性から、その抽出されたユースケースをフィーチャ候補として導出する。

図 7 に、例題システムにおけるフィーチャ候補の導出の手順を示す。各々のユースケース図からユースケースを抽出する。次に、製品 A,B,C の中で同じ名前を持つユースケースは、ユースケースの数を数える。「商品検索」、「商品注文」、「料金支払」は全ての製品に存在するユースケースであるからユースケース数が 3、「クレジット支払」は製品 B,C に存在するユースケースであるからユースケース数が 2、「現金支払」「購入履歴」は製品 B と製品 C のそれぞれ 1 つのみであるから、ユースケース数は 1 となる。

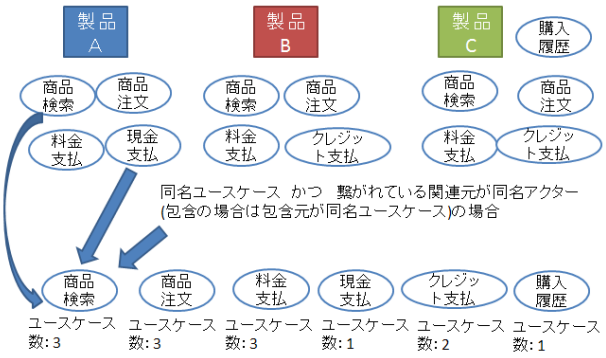

図 7 フィーチャ候補の導出

5.3 フィーチャの可変性と共通性の導出

導出された各フィーチャ候補がプロダクトラインの共通部分であるか、もしくは可変部分であるかを調べる。あるフィーチャ候補がもし共通部分である場合は、すべての製品に必要なフィーチャであるため、必須フィーチャとなる。第 5 章の冒頭でも述べたように、同じ振る舞いのユースケースは同じ名前にしている。もし、ユースケース名が一致していて、かつ、繋がれている関連元が同名アクター(包含ユースケースの場合は包含されたユースケースが同名ユースケース)の場合は同名のユースケースの数を把握しておく必要がある。プロダクトラインに含まれる製品の数をN個とすると、各フィーチャ候補の同名のユースケースの数が N である場合は、すべての製品に必要なフィーチャとなる。

図 8 に、例題システムにおける必須フィーチャ導出の手順を示す。今回の例では、プロダクトラインに含まれる製品は書籍購入システム A,B,C の 3 つである。製品の数と同名ユースケースの数が一致する場合は、すべての製品に必要なフィーチャであるため、「商品検索」、「商品注文」、「料金支払」の 3 つのユースケースは必須フィーチャとなる。他のユースケース「現金支払」、「クレジット支払」、「購入履歴」は、同名ユースケース数は製品の数 3 以下であるため、必須フィーチャとはならない。

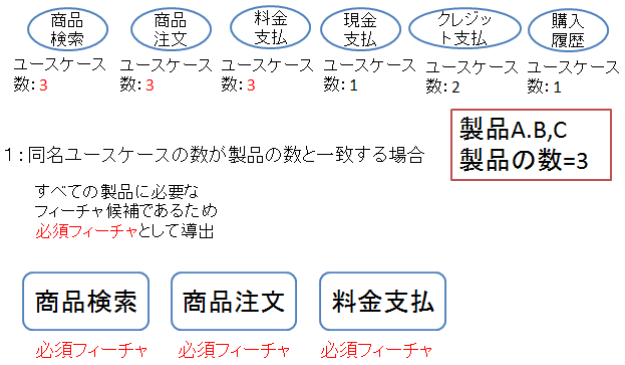

図 8 必須フィーチャの導出

プロダクトラインに含まれる製品の数をN個とすると、同名ユースケースがない、もしくは同名ユースケースの数がプロダクトラインに含まれるすべての製品の数N個以下の場合は、製品ごとに選択可能なフィーチャ候補であるため、可変部分のフィーチャとして導出する。あるフィーチャ候補が可変部分である場合は、製品ごとに選択することが可能である選択フィーチャ、選択制限のある代替フィーチャのどちらかとなる。

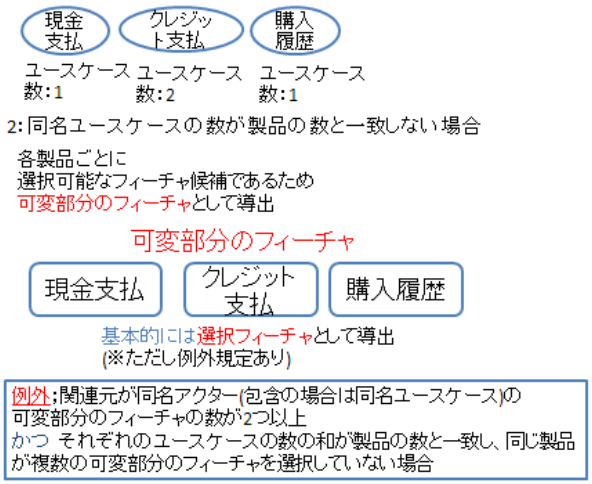

図 9 可変部分のフィーチャの導出

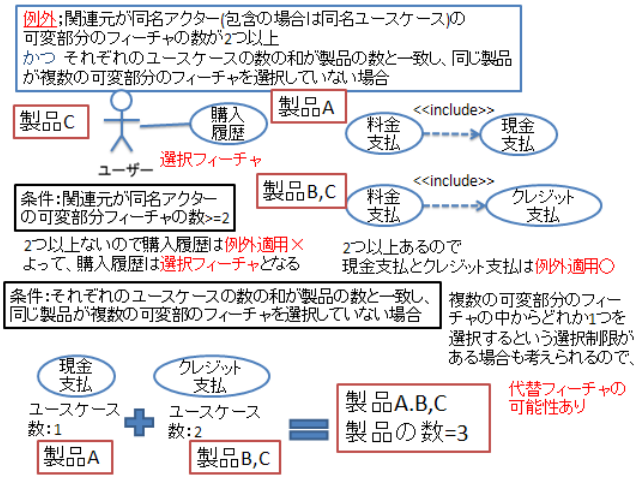

図 10 選択フィーチャ・代替フィーチャの導出

図 9 に、例題システムにおける可変部分のフィーチャ導出の手順を示す。それぞれのユースケース「現金支払」、「クレジット支払」、「購入履歴」は、同名のユースケース数がプロダクトラインに含まれる製品の数 3 以下である。そのため、製品ごとに選択可能なフィーチャであるから、「現金支払」、「クレジット支払」、「購入履歴」は可変部分のフィーチャとして導出される。

基本的には可変部分のフィーチャは選択フィーチャとなるが、例外規定に当てはまる場合は代替的フィーチャとなる可能性がある。その例外規定について説明する。代替は、2 つ以上の子フィーチャ群との間で定義され、親フィーチャにとって子フィーチャ群のいずれか 1 つが必要であることを示している。まず、代替フィーチャは可変部分であるので、必須フィーチャは対象から外されて選択的フィーチャとして導出されたものを対象とする。次に、2 つ以上の子フィーチャ群との間に定義されるものであることから、可変部分のフィーチャが 2 つ以上あり、かつ同じ親フィーチャと関連していることが条件となる。最後に、代替フィーチャは対象としている子フィーチャ群のいずれか 1 つを選択するという選択制限のフィーチャであることから、子フィーチャ群が同じ製品に複数選択されている場合は対象から外される。よって、代替フィーチャとなる条件は、関連元が同じアクター(包含の場合は同名ユースケース)の可変部分のフィーチャの数が 2 つ以上であり、それぞれの同名ユースケースの数の和がプロダクトラインに含まれる製品の数と一致し、同じ製品が複数の可変部分のフィーチャを選択していないことである。条件に当てはまるフィーチャ群が代替的フィーチャになる可能性がある。このことから、条件に当てはまるフィーチャ群は、選択フィーチャ、または代替フィーチャであると表現される。

図 10 に、例題システムにおける代替フィーチャ導出の手順を示す。代替フィーチャに該当する条件として、関連元が同名アクター(もしくは、同名ユースケース)の可変部分のフィーチャの数が 2 つ以上である必要がある。可変部分のフィーチャの中で関連元がユーザーであるのは「購入履歴」1 つのみであるから、購入履歴は例外規定に当てはまらないので選択フィーチャとなる。残りの可変部分のフィーチャ「現金支払」と「クレジット支払」の関連元は、両方ともユースケース「料金支払」であるから、条件に該当する。さらに他の条件として、それぞれのユースケースの数の和が製品の数と一致し、同じ製品が複数の可変部分のフィーチャを選択していない複数の子フィーチャ群というものがある。「現金支払」の数が 1 であり、「クレジット支払」の数が 2 であることから、その 2 つの同名ユースケースの数の和は製品の数 3 と一致する。そして、製品 A,B,C のどの製品も「現金支払」と「クレジット支払」のどちらか 1 つのみ選択しているので、2 つのフィーチャ候補「現金支払」と「クレジット支払」は代替フィーチャになる可能性がある。

5.4 ユースケース記述の比較

プロダクトラインに含まれる各製品のユースケース図からフィーチャの共通性と可変性を導出した後に、ユースケース図上でユースケース名が一致しているユースケースに着目する。ユースケース名が一致していても、その記述を詳細に見た場合に差異があることもあるので、それぞれのユースケース記述を比較する。ユースケース記述を比較するのは、同名ユースケースの数が 2 以上のフィーチャ候補である。

図 11 に、例題システムにおけるユースケース記述比較の手順を示す。同名ユースケースの数が 1 つのみの場合はユースケース記述を比較することが出来ないので、フィーチャ候補である「現金支払」と「購入履歴」の 2 つは記述比

較の対象外となる。よって、残りのフィーチャ候補はユースケース記述であるアクティビティ図を比較する。

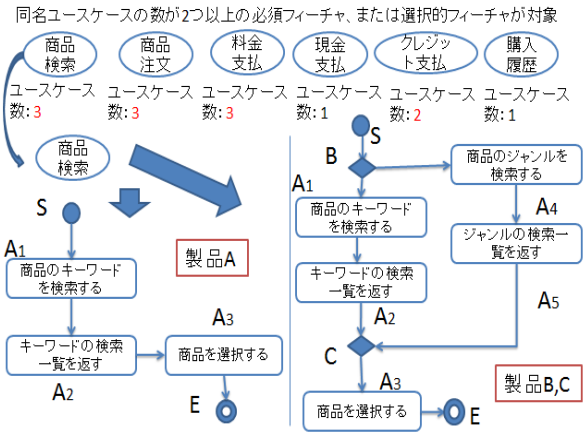

図 11　ユースケース記述比較

まず各製品のユースケース記述であるアクティビティ図の開始ノードから終了ノードまでに含まれるすべてのアクションを抽出する。ユースケースと同様、ユースケース記述のアクションも同じ振る舞いの場合は同じ名前を持つとしている。抽出されたアクションの中に同じ名前を持つアクションが存在する場合、それぞれの同名アクションの数を把握する必要がある。同じ振る舞いのユースケース記述を比較して差異が出る場合は、その差異の部分が可変部分となる可能性があるので、差異の部分をサブフィーチャの情報として導出する。同じ振る舞いのアクションの数が比較する製品の数と一致しない場合、それらのアクションをまとめて 1 つのフィーチャとして扱う。ただし、フィーチャとしてどのような名前のフィーチャにすべきか、といった情報まではユースケース記述から単純に作成することが出来ないので、フィーチャ候補の情報として導出する。

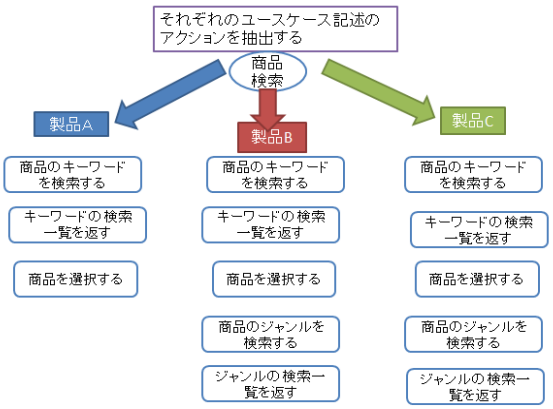

図 12　アクション抽出例

図 12 と図 13 に、例題システムにおけるアクション抽出と可変性のサブフィーチャ情報導出の手順を示す。「商品検索」のユースケース記述の書かれているアクションをそれぞれ抜き出し、同じ名前を持つアクションの数を数える。ユースケース「商品検索」を持つ製品は A,B,C の 3 つであるから、同名アクションの数が 3 以下である「ジャンル検索一覧を返す」と「商品のジャンルを検索する」の 2 つのアクションが可変部分である選択、代替サブフィーチャの情報として導出される。

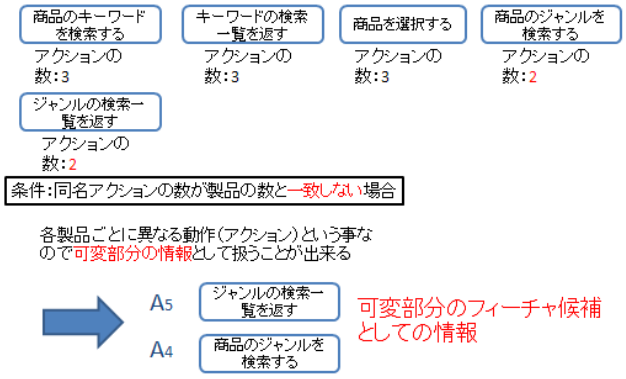

図 13　サブフィーチャの情報導出

5.5　フィーチャモデルの導出

最後に、今までに導出したフィーチャ、フィーチャの共通性と可変性、フィーチャ情報を組み合わせることでフィーチャモデルを作成する。アクターと直接関連を持つユースケースから導出されたフィーチャを、ルートフィーチャ(作成するシステム全体を表すフィーチャ)のサブフィーチャとする。そして包含されたユースケースから導出されたフィーチャは包含しているユースケースから導出されたフィーチャのサブフィーチャとなり、ユースケース記述同士を比較して抽出されたアクションは、比較したユースケースから導出されたフィーチャのサブフィーチャの情報となる。図 14 に、導出されたフィーチャモデルを示す。

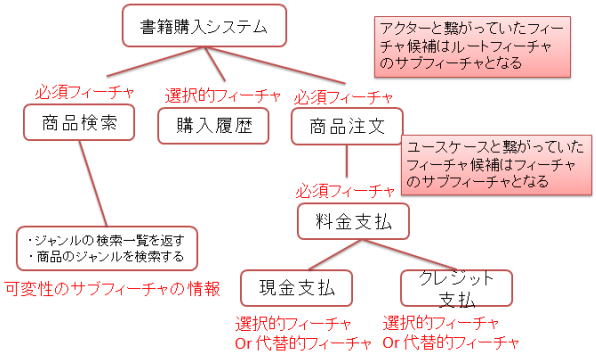

図 14　導出されたフィーチャモデル

6.　適用実験

本章では、提案手法を実装したフィーチャモデル導出ツールとそのツールを用いた導出実験、そしてその結果と考察について述べる。

6.1 フィーチャモデル導出ツール

提案手法をもとに、ユースケースからフィーチャモデルを作成するツールを実装した。本ツールは、XMI 形式で表現したユースケース図とユースケース記述を入力とし、フィーチャモデルの情報を XML 形式で出力する。本ツールは、他ツールとの連携等を考慮し、入力には XMI 形式、出力には XML 形式を用いることにした。

6.2 ツールを用いた導出実験

実装したツールを用いてユースケースからフィーチャモデルを生成する適用実験を行った。例題の章で説明したプロダクトラインに 1 製品加えたシステム「書籍購入システム」を実験に用いる。このプロダクトラインは、4 つの製品から構成され、各製品はそれぞれ 3~5 個のユースケースを持つ。また、[5]で例題として取り上げられている「Electronic Commerce System」についても適用実験を行う。なお、[5]では各プロダクトの詳細は示されずユースケース図のみが示されているため、それらのユースケース図のみを入力としてどのようなフィーチャモデルが導出されるかを実験した。

6.3 結果と考察

「書籍購入システム」について、ツールを適用して導出されたフィーチャモデルを図 15 に示す。ツールでは、フィーチャモデルの情報が XML 形式で出力されるが、ここではわかりやすさのために、出力された情報と同等の情報を図形式で示す。ユースケース図から全ての製品に必要なフィーチャは必須フィーチャとして、製品ごとに異なるフィーチャは選択、または代替フィーチャとして導出され、ユースケース記述同士を比較したときに差異がある場合はサブフィーチャの情報として導出することが出来た。

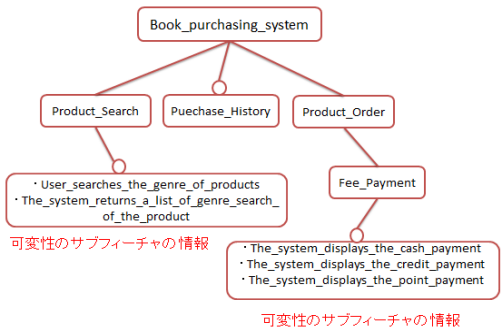

図 15 出力結果から作成したフィーチャモデル
「書籍購入システム」

また、図 16 に「Electronic Commerce System」のユースケース図から導出されたフィーチャモデルを示す。

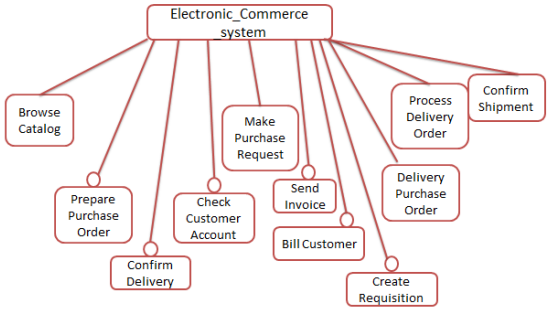

図 16 出力結果から作成したフィーチャモデル
「Electronic Commerce System」

今回、同じ振る舞いのユースケースのユースケース記述を比較して差異がある場合はアクティビティ図内の可変部分であるアクションを抽出し、複数ある場合でもひとまとめにして 1 つの可変性のサブフィーチャの情報として導出したが、アクション間の関係によって複数のサブフィーチャになる可能性もあるので、今後検討する必要がある。

7. 関連研究

Alexandre らはユースケース図からフィーチャモデルへ自動変換する手法を提案した[6]。この手法では、プロダクトライン全体を示すユースケース図とフィーチャモデル間のマッピングの自動化を目的としてユースケースからフィーチャモデルへ自動的に変換している。我々は、各プロダクトのユースケース群を入力としているのに対し、Alexandre らは可変性の表現を拡張したユースケース図を用いたプロダクトライン全体についてのユースケースを入力としている。Alexandre らの手法では、プロダクトラインのユースケース図から自動的にフィーチャモデルを得ることができるが、プロダクトライン全体についてもれのないユースケース図を作成する必要がある。それに対して、我々の手法では自動で完全なフィーチャモデルを得ることはできないが、プロダクトライン全体のユースケース図よりはより容易に作成できると思われる個々のプロダクトのユースケース図をもとにフィーチャモデルの素案を導出することができる。

Gomma はユースケースとフィーチャについての類似性と相違性について述べている[5]。ユースケースとフィーチャの類似性と相違性を整理し、例題を用いてユースケースからフィーチャモデル作成しているが、自動的にフィーチャモデルを導出するなどはしていない。我々の提案手法では、１ユースケース１フィーチャとして、必須・選択・代替フィーチャを導出する。それに対して[5]では全ての製品に共通した振る舞いのユースケースを全部ひとまとめにして１つの必須フィーチャとして導出し、残りのユースケー

スは関連元として繋がれたアクターごとにユースケースをひとまとめにして選択、または代替フィーチャとして導出している。

宇野らはゴールグラフからフィーチャモデルを導出する手法を提案した[7]。宇野らは既存製品に対して製品毎の開発からプロダクトライン開発に移行する際のフィーチャモデル作成を目的としている。それに対して本研究は、既存製品からの移行に限定していない、主に新規にプロダクトラインを定義する場合を目的に考えている。ユースケースでは機能をフィーチャとして導出するのに対し、ゴールでは操作などの特徴をフィーチャとして導出するので導出されるフィーチャの性質や粒度に違いがある。どのような場合にどちらのフィーチャが適切であるのか等、検討の必要があると思われる。

8. おわりに

フィーチャモデル作成を支援するために、プロダクトラインに含まれるプロダクトごとのユースケース図とユースケース記述からフィーチャモデルを生成する手法を提案した。また、提案手法をもとに実装したツールに例題を適用したところ、フィーチャモデルを導出することが出来た。

最後に今後の課題について述べる。今回はユースケース図の基本的な表現のみを対象としたが、拡張と汎化も活用したユースケース図の利用も考える必要がある。また、フィーチャモデルの依存・排他的関係を含めた場合のフィーチャモデル導出も考慮する必要がある。

ユースケース図のユースケース記述の差異から抽出される複数アクションの関係性も把握する必要がある。今回は全てまとめてひとつのフィーチャ候補としていたが、差異のあるアクション間の関係等により、複数のフィーチャになる可能性も考えられる。アクションから導出されるフィーチャの単位について今後検討する必要がある。

参考文献

1) Clements,P. and Northrop,L. : Software Product Lines : Practices and Patterns , Addison Wesley[2001]
2) K. C. Kang el.al :Feature-Oriented Domain Analysis(FODA) Feasibility Study, Technical Report SEI/CMU-90TR-21,Software Engineering Institute, Carnegie Mellon University[Nov. 2000]
3) マーチン・ファウラー(著),羽生田栄一(監訳):UML モデリングのエッセンス第３版,翔泳社[2005]
4) OMG－UML 仕様書:http://www.omg.org/spec/
5) Hassan Gomaa: Designing Software Product Lines with UML-From Use Cases to Pattern based Software Architectures, Addison -Wesley[2004]
6) Braganca, A. and Machado, R.J.: Automating Mappings between Use Case Diagrams and Feature Models for Software Product Lines, In Proc.11th International Software Product Line Conference, pp.3-12[2007]
7) 宇野耕平、林晋平、佐伯元司:ゴールグラフからのフィーチャモデル導出、情報処理学会研究報告．ソフトウェア工学研究会報告 2009(31), 1-8, [2009-03-11]