組込みシステムへのプロダクトライン型開発導入容易化に向けた可変性モデリング手法「機器-パッケージモデル」の提案

大島浩資[1]　秋下耀介[1]　若林昇[1]　川上真澄[1]

概要：組込みシステムへプロダクトライン型開発を適用する場合，システムの可変性に応じてソフトウェアの可変性が発生する．このような可変性の表現を容易にするため，可変性モデリング手法「機器-パッケージモデル」を提案する．システムの可変性の表現として機器レベルフィーチャを，システムの可変性に対応するソフトウェアの管理単位としてソフトウェアパッケージを導入し，これらの対応関係を管理する．また，機器-パッケージモデルを用い，既存のシステム仕様やソフトウェアを活用してプロダクトライン型開発を導入するプロセスを示した．そして，このプロセスを試行することで，機器-パッケージモデルの有効性を確認した．

キーワード：ソフトウェアプロダクトライン，組込みシステム，フィーチャモデル

Equipment-Package Model: Variability Modeling Method for Easy Introduction of Product Line Engineering for Embedded System

KOSUKE OSHIMA[†1]　YOSUKE AKISHITA[†1]
NOBORU WAKABAYSHI[†1]　MASUMI KAWAKAMI[†1]

Abstract: In application of software product line engineering (SPLE) to embedded system, software variability derives from system variability. In order to express such variability by utilizing existing specification of system and software, we proposed variability modeling method “Equipment-Package Model”. We introduce Equipment-level Feature for expressing valiability of system specification and Software Package for bundling software by Equipment-level Feature. We defined process to introduce SPLE by applying Equipment-Pakcage Model and existing specification and software. By trial of applying our method, we confirmed its effectiveness.

Keywords: software product line engineering, embedded system, feature model

1. はじめに

輸送用機器や産業用機器のような組込みシステムの開発において，機器に搭載するソフトウェアが機械や電気回路の制御に関わる場合，機械や電気回路と合わせてシステムが要求を満たすようソフトウェアを開発する必要がある．また，組込みシステムには，機能を実現するために複数のシステムが接続するものがある．接続手段としては，Ethernet や CAN (Controller Area Network) のような通信路を多く用いるが，機械的もしくは電気的な接続を用いる場合もあるため，接続手段の仕様を踏まえてシステムを開発する．

また，システム開発において，同種のシステムであってもユーザによって満たすべき要求が異なる場合があり，これらの要求を満たすようハードウェアおよびソフトウェアを開発する必要がある．このような状況下で，複数製品のソフトウェアを効率的に開発する技術としてソフトウェアプロダクトライン(SPL: Software Product Line) [1]がある．SPL による開発では，開発対象の製品系列におけるソフトウェアの共通性と可変性(製品ごとの差異)を分析し，それを基に複数製品で利用可能なソフトウェア資産を構築する．

可変性のモデリング手法として，フィーチャモデル[2]や直交可変性モデル[1]が用いられる．これらのモデルはシステムの特徴をフィーチャとして表現する．システムの特徴は製品分野によって異なるため，可変性モデリングにおいてはこれらのモデルをベースとしながらも，製品分野ごとに様々な応用がなされてきた．組込みシステムにおいては，システム，機械，電気など複数のドメインにわたる特徴を表現することから，ドメインごとに可変性モデルを構築し，それらを統合することで可変性をモデリングする方法を適用できる[3][4][5]．しかし，ドメインの設定や可変性モデルの構築方法によっては，システムの可変性に対してモデルの表現力が不足する可能性がある．また，複雑な可変性モデルを構築した場合には，モデルを実現するソフトウェアの開発が難しく，SPL 導入の難易度が上昇する可能性もある．

そこで本稿では，輸送用機器や産業用機器に搭載する組込みシステムの開発を想定し，以下の特徴を有する可変性

1 (株)日立製作所
Hitachi Ltd.

モデリング手法を提案する．

- 機械，電気やシステムの，ソフトウェアへ影響する可変性を表現する．ソフトウェアに影響する可変性に特化したモデルとすることで，SPL の実現に必要な情報は表現しつつ，可変性モデリングに要する工数や可変性モデルの規模を削減する．また，可変性モデリングの実施時に機器やインタフェースに関する設計情報を利用できるようにすることで，モデリングに必要となるドメイン知識の範囲を狭め，可変性モデリングの難易度を低減する．
- システムの可変性とソフトウェアの可変性の関係を表現する．ソフトウェアの可変性をシステムの可変性と対応付けて管理することで，多種の可変性や変異体を効率良く管理できるようにする．また，可変性モデリング以外のプロセス(例：既存ソフトウェアの分析)で得られた機能の構成を維持することで，開発者のドメイン知識や開発プロセスの経験を活用する．

さらに，この可変性モデリング手法を適用した上で，既存製品のソフトウェアを活用して再利用可能なソフトウェア資産を構築するプロセスと，構築した資産を利用して製品ソフトウェアを開発するプロセスを提案する．既存製品のソフトウェアを可能な限り流用することで，ソフトウェア資産の開発工数を削減して SPL の導入を容易にする．

2. 関連研究

2.1 可変性モデリング

文献[1]によれば，SPL の開発プロセスはドメイン開発とアプリケーション開発に大きく分けられる．ドメイン開発では，製品系列の可変性モデルを構築し，モデルに含まれる変異体(可変性の一つに対応するシステム仕様や実装)を作成する．アプリケーション開発では，ドメイン開発で作成した可変性モデルを利用し，製品系列に含まれる個別製品のソフトウェアを開発する．

可変性のモデリング手法として，フィーチャモデル[2]や直交可変性モデル[1]が用いられる．これらのモデルはシステムの特徴をフィーチャとして表現する．フィーチャはいくつかの文献で異なる定義がなされており[6]，例えば文献[1]では文献[2]を引用し「システムの，末端利用者から可視な特徴」と定義されている．これらのモデリング手法を利用する場合には，表現したい可変性に応じて適切に手法を利用する必要がある．

組込みシステムの開発においては，機械，電気やシステムの可変性とソフトウェアの可変性が関連する．このような関連を表現する方法の一つとして，ドメイン駆動開発[7]が考えられる．ドメイン駆動開発では，システムが扱うドメインを表現したドメインモデルと，ソフトウェアの設計および実装を一致させることで，システムの要求の変化に強いソフトウェアを実現する．しかし，導入の容易さの観点では，ドメインモデルに一致するソフトウェアを新たに開発するか，ドメインモデルに一致するように既存のソフトウェアをリファクタリングする必要があるため，導入工数，難易度の両方が高いと考えられる．

一方，可変性の範囲でシステムとソフトウェアの関連を表現するモデリング手法も提案されている．文献[8]ではシステムの可変性を直交可変性モデルで，ソフトウェアの可変性をフィーチャモデルで表現し，モデル間のリンクを追加することで可変性を表現する手法が提案されている．

組込みシステムを対象とする場合には，フィーチャは機械や電気など複数のドメインで定まる場合もあるため，問題領域ごとにフィーチャモデルを構築する手法がある．文献[3]では機能，非機能，物理の各ドメインについて SysML による要求モデルと直交可変性モデルを構築し，二つのモデルを対応させて管理している．文献[4]ではシステムに関するフィーチャモデルを機械，電気，ソフトウェアの各ドメインで詳細化している．文献[5]では機械や電気，熱などに関するフィーチャモデルを統合することでシステム全体のフィーチャモデルを構築している．

2.2 SPL 導入の難易度の抑制

企業において新たに SPL を導入する場合，可変性モデルやソフトウェア資産の構築に関する工数や難易度の予測が難しく，初期投資に踏み切れないことが SPL の導入を避ける理由となる．そのため，企業において SPL の導入に先行して XDDP(eXtreme Derivative Development Process)[9]を導入し，SPL の導入に必要な開発工数を確保する事例がある[10][11]. XDDP は，ある製品のソフトウェアをベースとし，それに変更を加えて派生製品のソフトウェアを開発する(派生開発). このとき，変更内容を所定のフォーマットの文書に記述し十分にレビューすることで，コーディング時の変更漏れや変更誤りを防止し，手戻り防止による工数削減を図る．XDDP はソフトウェア開発部門のみで着手できることや，初期投資が少ないことから導入しやすいと考えられている[10].

さらに，XDDP から SPL へ段階的に移行するプロセスを体系化した XDDP4SPL[12][13]が提案されている．XDDP4SPL では，まずいくつかの製品に対して XDDP を適用する．続いて，XDDP で作成された変更要求仕様書をもとに，変更が発生した箇所に対して局所的な可変性を分析し，局所的フィーチャモデルを構築する．さらに，局所的フィーチャモデルの構築を繰り返した後これらを統合し，大域的フィーチャモデルを構築する．そして，大域的フィーチャモデルと設計書やソースコードとのトレーサビリティを記録することにより，複数製品で利用可能なソフトウェア資産を構築し SPL へ移行する．

3. 課題

3.1 組込みソフトウェアの仕様に影響するシステムの可

変性を簡易な方法でモデリングすること

組込みシステムのソフトウェアはその要求を満たすため，システム内外の機械や電気回路に対する状態監視や制御を行う．製品系列を開発する場合には，機械や電気回路の可変性がソフトウェアの仕様に影響を与えることもある．可変性をソフトウェアにより漏れなく実現するため，ソフトウェアの仕様に影響を与えるシステムの可変性をモデリングすることを考える．

文献[8]の手法は，システムとソフトウェアの可変性を分けてモデリングし，それぞれの関係を表現している．ただし，システムの可変性モデルで表現する内容は手法の利用者に委ねられている．そのため，文献[4]では機械，電気，ソフトウェアの上位概念にシステムを位置づけてフィーチャモデルを構築している．この手法は，システム全体の可変性を分析できるが，機械，電気の可変性とソフトウェアの可変性の関係が複雑になりやすく，SPL 導入の難易度の上昇につながると考える．また，文献[5]では複数ドメインのフィーチャモデルを統合しているが，モデルの構築に広範なドメイン知識を必要とするため難易度が高いと考える．

このため本稿では，組込みソフトウェアの仕様に影響するシステムの可変性を簡易な方法でモデリングすることを研究課題の一つとする．

3.2 予め構成の定まったソフトウェア機能について，その可変性を組込みシステムの可変性と対応付けてモデリングすること

ソフトウェアを再利用し個別製品の開発工数を削減するためには，システムの可変性が引き起こすソフトウェアの可変性を特定し，変異体を実装する必要がある．

文献[3][5]の手法はシステムとソフトウェアの関係を明示していない．ソフトウェアの可変性は，文献[8]では個別のフィーチャモデルとして，文献[4]ではシステムのフィーチャモデルと同様の構造のフィーチャモデルを作成して管理している．これらの方法は，ソフトウェアの可変性をフィーチャとして表現できる場合は有効である．

ここで，組込みシステムが状態監視や制御の機能を実現するとき，それぞれの機能で，複数の機械や電気回路に関する情報について総合的な処理が求められる．このような機能を，機械や電気回路，システムの可変性に対応する単位で分割するのは難しい．特に，既存のソフトウェアを利用して SPL を導入する場合には，新たな分割方法の導入は開発者の理解や工数の面で不利である．

そこで本稿では，既存ソフトウェアの分析などによりソフトウェア機能の構成が定まっている場合に，その構成を利用した上で，ソフトウェアの可変性を組込みシステムの可変性と対応付けてモデリングすることを課題とする．

4. 組込みシステムの可変性モデリングへのアプローチ

4.1 対象システム：機器と接続する組込みシステム

本稿において可変性モデリングの対象とする組込みシステムの構成の例を図 1 に示す． 図 1 のシステムの特徴を以下に述べる．

- システムは，一つのコントローラと二つのセンサ，二つのアクチュエータから成り，コントローラは二つの外部システムと接続する．
- コントローラは CPU と記憶装置を有しており，コントローラ上でソフトウェアが動作する．
- コントローラ上で動作するソフトウェアは，センサ 1, 2 および外部システム 1, 2 から受け取ったデータをもとに，アクチュエータ 1, 2 および外部システム 1, 2 を制御する．

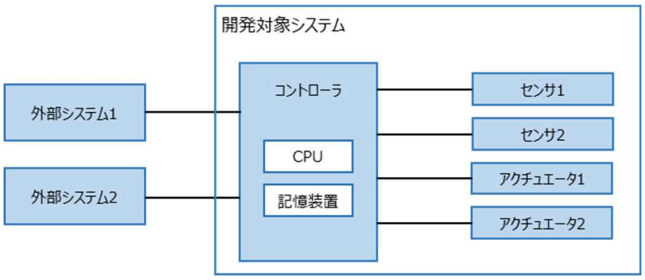

図 1 可変性モデリング対象のシステム構成例

図 1 のシステムは以降の説明で参照する．

4.2 課題解決のアプローチ

本稿では，3 章の課題を解決する組込みシステム向け可変性モデリング手法を提案する．提案に先立って，3 章の課題を解決するアプローチを以下に述べる．

(A) 組込みソフトウェアの仕様に影響するシステムの可変性を簡易な方法でモデリングすること

本稿で提案する可変性モデリング手法では，システムや機械，電気に関する可変性を，ソフトウェアに影響する範囲について表現する．機器やインタフェースなど，ソフトウェア開発時に参照する設計情報を利用してモデリングを行うことで，モデリングに必要となるドメイン知識の範囲を狭め，可変性モデリングの難易度を低減する．また，システムとソフトウェアの可変性を対応付けることを前提としたモデルにより，システムの可変性がソフトウェアの仕様に影響していることを確実にする．これにより，SPL の実現に必要十分な範囲でシステムの可変性をモデリングする．

(B) 予め構成の定まったソフトウェア機能について，その可変性を組込みシステムの可変性と対応付けてモデリングすること

本稿で提案する可変性モデリング手法では，ソフトウェアの可変性を，システムの可変性とソフトウェアの機能の

両方に対応させて管理する．これにより，既存のソフトウェアにおける機能の構成を維持することを可能とし，開発者のドメイン知識や開発プロセスの経験を活用を可能とする．これにより，可変性モデリングや，可変性モデルを利用したソフトウェア開発の難易度を抑制する．

4.3 可変性モデリングの前提となる組込みソフトウェアの可変性の特徴

4.3.1 システムの可変性

機器と接続する組込みシステムにおけるソフトウェアの仕様は，接続する機器に関する仕様の影響を受けて変化する．ソフトウェアの可変性の発生要因となるシステムの可変性を，その説明と合わせて以下のように整理する．

(1) 外部システムとのインタフェース仕様の差異

製品系列の開発においては，接続する外部システムの機能や開発メーカが製品によって異なり，データ項目の追加やデータフォーマットの変更など，インタフェース仕様の変更につながる場合がある．この変更がソフトウェア仕様の可変性を発生させる．

(2) システム内機器の仕様の差異

組込みシステム内で使用するセンサやアクチュエータ，コントローラなどの機器は，機器が有する機能のほか，機械的・電気的な特性，コスト，納期などの観点で選択する．言い換えると，製品系列の開発において，機器が有するべき機能が変わらない場合でも，その他の要因で機器を変更することがある．システム内機器の仕様変更は，機器を制御するソフトウェアの仕様の可変性を発生させる．

4.3.2 システムの可変性とソフトウェアの可変性の関係

4.3.1 節で整理したシステムの可変性は，ソフトウェアの可変性を複数発生させる可能性がある．これは，一つの機器からの入力を利用するソフトウェアの機能や，一つの機器への出力を作成するソフトウェアの機能が複数にわたる可能性があるためである．

5. 組込みシステム向け可変性モデリング手法「機器-パッケージモデル」の提案

5.1 手法の概要

4.3 節に示す可変性の特徴を表現する可変性モデリング手法として機器-パッケージモデルを提案する．本手法の特長を以下に示す．

- 4.3.1 節に示すシステムの可変性を，システムの内外を問わず機器の仕様に由来する可変性ととらえ，機器レベルフィーチャとして表現するよう定める．4.3.1 節(1)のアプローチに従うことで，可変性モデリングの工数や難易度，可変性モデル規模を削減する．
- 4.3.2 節で述べたように，システムの可変性はソフトウェアの可変性を複数発生させる可能性がある．この関係についてトレーサビリティを確保するため，同時に発生する可変性を管理する単位としてソフトウェアパッケージを導入する．複数の機能にわたる可変性をソフトウェアパッケージとして管理することで，機能の構成を維持しながら，システムの可変性とソフトウェアの可変性の関係を表現する．

以降，5.2 節で機器レベルフィーチャについて，5.3 節でソフトウェアパッケージについて，それぞれ説明する．

5.2 機器レベルフィーチャによるシステム可変性の表現

機器レベルフィーチャは以下の通り定義する．

定義 機器レベルフィーチャとは，他の機器と接続するシステムの設計において，ある機器が有する特徴のうち，他の機器の特徴の決定に影響を与えるものである．

図 1 のシステムに関する機器レベルフィーチャの例を以下に示す．

(a) 外部システムに関する機器レベルフィーチャ

- ある種類の外部システムの有無
- 外部システムとコントローラの接続手段(通信路，電気的接続，機械的接続)
- 外部システムの制御方法(入力と動作との関係)
- 外部システムの動作と外部システムからの出力との関係

(b) システム内機器に関する機器レベルフィーチャ

- ある種類のセンサの有無
- センサとコントローラの接続手段(通信路，電気的接続，機械的接続)
- センサが測定した物理量とセンサの出力との関係
- ある種類のアクチュエータの有無
- アクチュエータとコントローラの接続手段(通信路，電気的接続，機械的接続)
- アクチュエータの制御方法(入力と動作との関係)

これらの機器レベルフィーチャは，組込みシステムの開発に必要な情報であるため，機器やインタフェースの設計情報として組込みソフトウェアの開発者に提示される．この情報を利用して機器レベルフィーチャを抽出する．

5.3 ソフトウェアパッケージによるシステム可変性とソフトウェア可変性の管理

ソフトウェアパッケージは以下の通り定義する．

定義 ソフトウェアパッケージとは，可変性発生要因となる機器レベルフィーチャが同一であるソフトウェアの変異体の集合である．ソフトウェアパッケージには，仕様，テスト仕様，ソースコードのそれぞれを含む．

ここで，機器レベルフィーチャとソフトウェアパッケージの関係を，図 2 を用いて説明する．図 2 の表は，図 1 のシステムの機器レベルフィーチャとソフトウェア仕様の関係を表す．例えば，センサ 1 の種類が Type A である場合，アクチュエータ 1 制御機能はロジック a によって実現され，外部システム 1 情報通知機能で通知される情報にはセンサ 1 情報が含まれる．このとき，機器レベルフィーチャ「センサ 1: Type A」が選択されたとき，選択されるソフトウェ

ア仕様の集合および，それに対応するテスト仕様とソースコードが，「センサ 1: Type A」に対応するソフトウェアパッケージである．

なお，図 2 ではソフトウェアパッケージを構成するソフトウェア仕様は機能を単位として記述しているが，複数の機器レベルフィーチャが可変性発生要因となる機能も存在し得る．その場合，単一の機器レベルフィーチャが可変性発生要因となるよう，可変性および変異体とする対象の範囲を狭める．例えば，機能に含まれる一部のロジックやパラメータを単位としてソフトウェアパッケージを構成してもよい．

また，ソフトウェアパッケージを作成し終えた時点で，ある機器レベルフィーチャに対応するソフトウェアパッケージが存在しないことが分かった場合には，その機器レベルフィーチャを機器-パッケージモデルから削除する．

機器レベル フィーチャ	ソフトウェア仕様	
	アクチュエータ 1 制御機能	外部システム 1 情報通知機能
センサ 1: Type A	ロジック a	センサ 1 情報あり
センサ 1: Type B	ロジック b	センサ 1 情報あり
センサ 1: なし	ロジック c	センサ 1 情報なし

ソフトウェアパッケージ

図 2　センサ 1 に関するソフトウェアパッケージの例

5.4 課題に対する機器-パッケージモデルの効果

3 章に示した課題に対する機器-パッケージモデルの効果を考察する．

(A) 組込みソフトウェアの仕様に影響するシステムの可変性を簡易な方法でモデリングすること

組込みソフトウェアの開発者に提示される情報を利用することで，必要となるドメイン知識の範囲を狭め，可変性モデリングの難易度を低減する．また，ソフトウェアパッケージを機器レベルフィーチャと対応付けることと，ソフトウェアパッケージと対応しない機器レベルフィーチャを削除することにより，可変性モデルに含まれるシステムの可変性がソフトウェアの仕様に影響していることを確実にする．これにより，可変性モデルを簡潔に保ち，その管理を容易にする．

(B) 予め構成の定まったソフトウェア機能について，その可変性を組込みシステムの可変性と対応付けてモデリングすること

ソフトウェアパッケージを機器レベルフィーチャに対応付ける方式とすることで，ソフトウェアの可変性のモデリングと．システムとソフトウェアの可変性の対応付けを同時に実現する．また，図 2 に示すように，ソフトウェアパッケージに含まれる変異体がソフトウェアの機能ごとに特定されており，既存のソフトウェアにおける機能の単位を維持することが可能である．

6. 機器-パッケージモデルおよび既存ソフトウェアを活用した SPL 型開発プロセスの提案

6.1 可変性モデリングおよびソフトウェア資産構築のプロセス

6.1.1 提案プロセスの全体像

5 章で提案した機器-パッケージモデルは，既存製品のソフトウェアを活用して SPL を導入する場合に利用できる．SPL の導入時には，可変性モデリングを行い，各製品の開発へ応用するソフトウェア資産を構築する．6.1 節では，これらのプロセスを提案する．

プロセスの流れを図 3 に示す．図 3 は，XDDP で使用される開発プロセス記法の PFD[9]を用いてプロセスを記述した．図 3 のプロセスは，ソフトウェア資産として，ソフトウェア仕様資産(ソフトウェア仕様書，テスト仕様書)およびソースコード資産を作成する．ソフトウェア仕様資産とソースコード資産は，個別製品の開発時にベースとして使用する成果物である．以降，可変性モデリングのプロセス(図 3 の P1～P3)について 6.1.2 節で，ソフトウェア資産の構築プロセス(図 3 の P4～P6)について 6.1.3 節で述べる．

6.1.2 機器-パッケージモデルによる可変性モデリング

機器-パッケージモデルによる可変性モデリングのプロセスでは，まず既存製品のシステム仕様から機器レベルフィーチャを抽出する(図 3 の P1)．このとき，(a)外部システムに関する機器レベルフィーチャの抽出には外部システムとのインタフェース仕様を，(b)システム内機器に関する機器レベルフィーチャの抽出にはシステム仕様や機器のハードウェア仕様を参照する．

続いて，既存製品のソフトウェア仕様から，製品間のソフトウェア仕様の差分を抽出する(図 3 の P2)．ソフトウェア仕様の差分を機器レベルフィーチャの単位でまとめてソフトウェアパッケージを定義し，ソフトウェアパッケージ一覧を作成する(図 3 の P3)．

ソフトウェアパッケージ一覧には以下を記述する．

- パッケージ ID：ソフトウェアパッケージを一意に特定する．
- パッケージ名：ソフトウェアパッケージの名称を示す．
- 対応機器レベルフィーチャ：対応する機器レベルフーチャを示す．
- 説明：ソフトウェアパッケージに含まれるソフトウェアの仕様を簡潔に説明する。

ソフトウェアパッケージ一覧の作成例を表 1 に示す．表 1 には，ソフトウェアパッケージとして「TypeA センサ制御」「TypeB センサ制御」「センサ不使用制御」の 3 種類のソフトウェアパッケージを記録している．

表 1　ソフトウェアパッケージ一覧の作成例

パッケージ ID	パッケージ名	対応機器レベルフィーチャ	説明
1	TypeA センサ制御	センサ 1: Type A	TypeA センサを使用したシステムを制御する.
2	TypeB センサ制御	センサ 1: Type B	TypeB センサを使用したシステムを制御する.
3	センサ不使用制御	センサ 1: なし	センサを使用せずにシステムを制御する.

6.1.3 ソフトウェア資産の構築

6.1.2 項で作成したソフトウェアパッケージを実現し，機器レベルフィーチャの選択に応じてソフトウェアパッケージが選択されるようにする．ソフトウェア資産構築のプロセスは，ソフトウェア仕様資産の構築(図 3 の P4)，ソースコード差分の抽出，ソースコード資産の構築(図 3 の P5, P6)のプロセスで実施する．

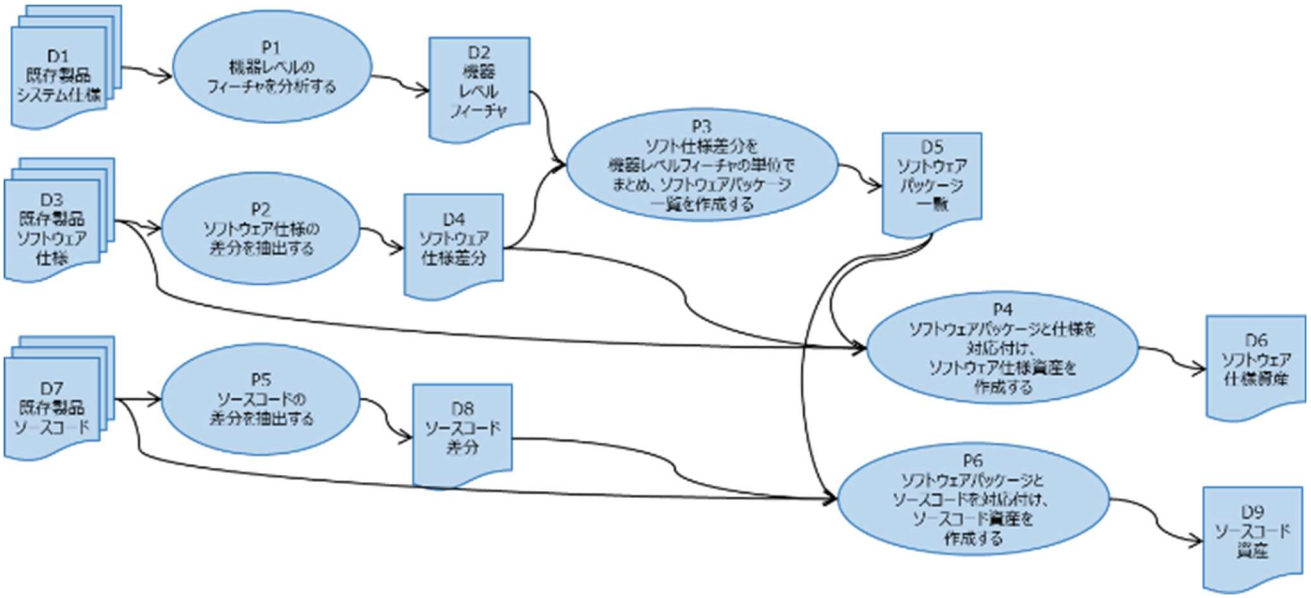

図 3　ソフトウェア資産構築プロセス

(a)　ソフトウェア仕様資産の構築

ソフトウェア仕様資産は，ソフトウェア仕様とテスト仕様を定義するドキュメントであり，ソフトウェアパッケージの選択に応じて変化する仕様についても記述したものである．ソフトウェア仕様資産は，既存のソフトウェア仕様書とテスト仕様書に対して，ソフトウェアパッケージを選択した際に定まる仕様を追記することで行う．具体的な手順を以下に示す.

(1) 仕様を記述するドキュメントに，ソフトウェアパッケージの選択によって定義される可能性のある仕様の候補をすべて併記する.

(2) ソフトウェアパッケージを選択した場合，(1)で併記された仕様の候補のうち選択されなかったものはソフトウェアの仕様から外れる．外れた仕様を仕様書から削除し，製品の仕様書を作成できるようにするため，仕様書の備考などに，ソフトウェアパッケージの選択に応じた本文の編集作業の指示を記載する．このとき，編集作業の指示にパッケージ ID を加える。これにより，開発者がパッケージ ID で指示を検索することを可能とし，編集作業の漏れを防止する.

このようにして記述したソフトウェア仕様の例を表 2 に示す．表 2 の仕様は，例えばパッケージ ID1 のソフトウェアパッケージを選択したとき，ロジック a によりアクチュエータ 1 を制御することを定義する.

表 2　ソフトウェア仕様の例

仕様	備考
ロジック{a, b, c}によりアクチュエータ 1 を制御する.	[ID1] ロジック a を選択 [ID2] ロジック b を選択 [ID3] ロジック c を選択

(b)　ソースコード差分の抽出，ソースコード資産の構築

ソースコード資産は，ソフトウェアパッケージの選択に応じて動作の変更が可能なソースコードであり，ソフトウェア仕様資産のソースコード版のような位置付けである.

ソースコード資産の構築は，既存のソースコードに対し，ソフトウェアパッケージに対応する差分を追加することで実施する．差分の追加は，ビルド時または実行時に選択可能な方法で実施する．例えば C 言語においては，#ifdef ブロックや if 文の利用が考えられる.

6.2 製品ソフトウェアの開発プロセス

6.1 節のプロセスで構築したソフトウェア資産を利用して，個別製品のソフトウェアを開発するプロセスを示す．個別製品のソフトウェアは，以下のプロセスで開発する．

(1) 個別製品に対する要求を分析する。
(2) 個別製品のシステム仕様から機器レベルフィーチャを特定する．
(3) ソフトウェアパッケージ一覧を参照し，機器レベルフィーチャからソフトウェアパッケージを選択する．
(4) ソフトウェアパッケージによって，(1)で明らかにした要求を満たすことができるか分析する．満たすことができる要求はそのソフトウェアパッケージで実現するものとし，満たすことができない場合には，追加の仕様を定義する．
(5) ソフトウェア仕様書およびテスト仕様書を作成する．ソフトウェアパッケージで実現する仕様は 6.1 節で定義した仕様を参照して記述し，それ以外の仕様については新たに定義する．
(6) コーディングを行う．ソフトウェアパッケージで実現する仕様については，対応する処理が実行されるよう設定する．それ以外の仕様については(5)で定義した仕様をもとにコーディングを行う．
(7) (5)で作成したテスト仕様書に基づきテストを実施する．

7. 機器-パッケージモデルの評価

7.1 評価の目的

本稿で提案した機器-パッケージモデルを評価する．評価の観点として以下を挙げる．

(1) 機器-パッケージモデルを利用したプロセスによる SPL 導入の効果．本節では個別製品の開発工数を削減可能であるか評価する．
(2) 可変性モデリングおよびソフトウェア資産構築に関する課題に対し，機器-パッケージモデルがもたらす効果．
(3) 機器-パッケージモデルおよび既存ソフトウェアを活用した SPL 型開発の容易性

7.2 評価の方法

輸送用機器を構成するシステムへ，機器-パッケージモデルおよび 6.1 節に示したソフトウェア資産構築プロセスを適用する．そして，以下の点について評価する．

(1) 機器-パッケージモデルを利用したプロセスによる SPL 導入の効果

新たな製品を開発することを想定し，機器-パッケージモデルを適用して SPL を導入した場合としない場合の開発工数を見積もった．なお，SPL を導入しない場合は，開発する製品に近い過去の製品からの派生開発によって開発するものとした．また，該当システムの開発ではモデルからのソースコード自動生成を利用しており，開発範囲においてコーディングを実施しなかったため，コーディングの工数は評価していない．

(2) 可変性モデリングに関する課題に対し，機器-パッケージモデルがもたらす効果

可変性モデリングを実際に実施し，5.4 節に述べた効果が実際に得られたかを確認する．これにより，関連研究を踏まえて抽出した 3 章の課題が解決されているか検討し，機器-パッケージモデルの有用性を評価する．

(3) 機器-パッケージモデルおよび既存ソフトウェアを活用した SPL 型開発の容易性

(2)に続いてソフトウェア資産構築を実施し，その容易性を考察する．

7.3 評価の結果

(1) SPL 導入の効果

(a) 評価の前提

評価の前提となるパラメータを表 3 に示す．以下，各項目を説明する．各項目は，対象製品の過去事例の開発経験を元に見積もった値である．

- 流用率 : 対象案件を派生開発で開発した場合に，流用する仕様項目の割合
- ソフトウェアパッケージ選択時間 : SPL を適用する場合に，ソフトウェアパッケージの選択にかかる時間
- 変更項目定義時間 : 派生開発の場合に，変更する仕様の項目を特定し，その概要を記述するのにかかる時間
- ソフトウェアパッケージを利用した場合の仕様更新時間 : 6.1.3 節の方法で定義した仕様を選択する時間
- ソフトウェアパッケージを利用しなかった場合の仕様更新時間 : 変更項目に応じて新たに仕様を更新する場合の時間

表 3　評価の前提

項目	値
流用率	80%
ソフトウェアパッケージ選択時間	5 分/項目
変更項目定義時間	22 分/項目
ソフトウェアパッケージを利用した場合の仕様更新時間	5 分/ページ
ソフトウェアパッケージを利用しなかった場合の仕様更新時間	22 分/ページ

(b) 評価の方法

表 3 を前提に，ソフトウェアパッケージ選択/変更項目定義，仕様定義の工数を計算する．それぞれの計算方法を以下に示す．

- ソフトウェアパッケージ選択/変更項目定義 : SPL を導入する場合について，適用ドメインにおけるすべてのソフトウェアパッケージを選択するのに要する時間を，表 3 の「ソフトウェアパッケージ選択時間」と，

ソフトウェア資産構築の結果得られたパッケージ数から見積もる. また, SPL を導入しない場合について, 適用対象システムにおいて各機能の変更項目を定義する時間を, 表 3 の「変更項目定義時間」から見積もる. このとき, 変更項目数は表 3 の「流用率」と適用対象システムの機能数に基づき算出する.

- 仕様定義 : SPL を導入する場合は, ソフトウェアパッケージの選択に基づき仕様を更新する時間を, 表 3 の「ソフトウェアパッケージを利用した場合の仕様更新時間」と適用対象システムの仕様項目数を用いて見積もる. ただし, ソフトウェア資産構築の実績に基づき, ソフトウェアパッケージの選択のみで対応できない仕様変更の割合を見積もり, それらについては「ソフトウェアパッケージを利用しなかった場合の仕様更新時間」がかかるとしている. SPL 非導入時は,「ソフトウェアパッケージを利用しなかった場合の仕様更新時間」と適用対象システムの仕様項目数から仕様更新時間を計算する.

(c) 評価の結果

評価の結果を表 4 に示す.

本稿のプロセスで SPL を導入した場合, 非導入の場合に比べて工数を 45%削減しており, SPL 導入の効果が得られていると考える.

表 4 評価の結果

項目	SPL 非導入	SPL 導入
	[h]	[h]
ソフトウェアパッケージ選択/変更項目定義	11.0	4.0
仕様定義	282.8	157.2
合計	293.8	161.2 (45%削減)

(2) 可変性モデリングに関する課題に対し, 機器-パッケージモデルがもたらす効果

機器-パッケージモデルおよびそれを用いた開発プロセスの適用効果を以下の通り考察する.

(A) 組込みソフトウェアの仕様に影響するシステムの可変性を簡易な方法でモデリングすること

機器レベルフィーチャの抽出は, 既存ソフトウェアの開発時に参照もしくは作成した仕様書と, 輸送用機器ソフト開発エンジニアへのヒアリングのみにより実施した. このことから, ソフトウェア開発の範囲で抽出が可能であり, 可変性モデリングの難易度低減を実現したと考える. また, すべてのソフトウェアパッケージを単一の機器レベルフィーチャと対応付けることができ,簡潔な可変性モデルを構築できた.

(B) 予め構成の定まったソフトウェア機能について, その可変性を組込みシステムの可変性と対応付けてモデリングすること

機能の構造を維持したままソフトウェアパッケージの作成を実施することができた. この点は, 既存のソフトウェアや輸送用機器ソフト開発エンジニアのドメイン知識の活用につながり, 仕様の検討とレビューを進めやすくしたと考える. なお, レビューにおいては, 作成したソフトウェアパッケージを今後使用する見込みがあるかも検討しており, これにより可変性モデルやソフトウェア資産の妥当性を確保している.

(3) 機器-パッケージモデルおよび既存ソフトウェアを活用した SPL 型開発の容易性

ソフトウェア資産の構築は以下の体制で実施した.

- SPL 導入担当 1 名(専任, ソフトウェア資産構築を担当, 本稿の著者)
- 輸送用機器ソフト開発エンジニア 5 名(通常は製品開発に従事し, ドメイン知識の提供とレビューについて協力)

SPL 導入担当は, 評価開始時点で輸送用機器ソフト開発の従事経験が 1 年程度であり, ドメイン知識が不足している. 一方輸送用機器ソフト開発エンジニアは, 評価開始時点では SPL に関する教育を受けていない. この体制のもと, SPL に関する知識とドメイン知識を共有し, 約 5 ヶ月でソフトウェア資産を構築することができた.

本手法の課題として, 既存製品に無い仕様からソフトウェア資産を構築できないことが挙げられる. このような仕様については, 6.2 節に示すように個別製品のソフトウェア開発時に新たに仕様定義とコーディングを実施し, その結果をソフトウェアパッケージと統合して個別製品のソフトウェアを開発する. また, この仕様を再利用する見込みがある場合には, 作成した仕様書やソースコードを流用して可変性モデルとソフトウェア資産を更新する.

以上の考察から, 機器-パッケージモデルによる可変性モデリングは組織の技術やを活用しながら短期間で SPL を導入することを可能とし, SPL 導入を容易にすると考える.

8. まとめ

本稿では, 組込みシステムにおける可変性の的確な表現と SPL 導入の容易化を目的に, 可変性モデリング手法として機器-パッケージモデルを提案した. 機器と接続する組込みシステムへの適用を前提に, システムの可変性の表現に機器レベルフィーチャを利用し, システムの可変性とソフ

トウェアの可変性の関係についての表現にソフトウェアパッケージを利用した．また，機器-パッケージモデルを適用し既存ソフトウェアを活用した SPL プロセスを示し，ソフトウェア資産の構築を試行した．試行に関する評価から，本手法によって，SPL 導入を容易にした上で，個別製品のソフトウェアの開発工数を削減する効果を得たと考える．

今後の課題として，ソフトウェア資産の構造の改善を促す開発プロセスとすることを挙げる．本稿で示した SPL 導入プロセスは，既存のソフトウェアを活用することで開発の効率化を図ったが，既存のソフトウェアの構造に問題がある場合にはソフトウェア品質の低下が見込まれる．そのため，リファクタリングなどの手法をプロセスに取り入れ，構造を継続的に改善していくことを考える．

参考文献

[1] クラウス　ポール，ギュンター　ベックレ，フランク　ヴァンデル　リンデン．ソフトウェアプロダクトラインエンジニアリング　ソフトウェア製品系列開発の基礎と概念から技法まで．株式会社エスアイビー・アクセス．2009.
[2] Kang, K. C., Cohen, S. G., Hess, J. A., Novak, W. E., and Peterson, A. S. Feature-oriented domain analysis (FODA) feasibility study. SEI, CMU, Tech. Rep, 1990.
[3] M. Li, et al. Model-based systems engineering with requirements variability for embedded real-time systems, 2015 IEEE International Model-Driven Requirements Engineering Workshop (MoDRE), pp. 1-10, 2015.
[4] B. Vogel-Heuser et al. Variability management for automated production systems using product lines and feature models. Proc. INDIN 2016, pp. 1231-1237, 2016.
[5] Chavarriaga, J. et al. Using multiple feature models to specify configuration options for electrical transformers: an experience report. Proc. SPLC 2015, pp. 216-224, 2015.
[6] Berger, T. et al. What is a feature? a qualitative study of features in industrial software product lines. Proc. SPLC 2015, pp. 16–25, 2015.
[7] エリック・エヴァンス，エリック・エヴァンスのドメイン駆動設計　ソフトウェアの核心にある複雑さに立ち向かう，株式会社翔泳社，2011
[8] Metzger, A. et al. Disambiguating the Documentation of Variability in Software Product Lines: A Separation of Concerns, Formalization and Automated Analysis In Proc. RE 2007, pp. 243-253, 2007.
[9] 清水吉男．「派生開発」を成功させるプロセス改善の技術と極意．株式会社技術評論社．2007.
[10] 筒井賢．プラットフォーム開発、XDDP 開発、そして SPL 開発へ 〜 移行のきっかけ(取り巻く環境)と成功/失敗のステップ 〜．第４回 アフォード・フォーラム，2013.
[11] 小寺良明．膨大かつ断続的な変更要求を統制し、手戻りを改善 〜手に入れた時間をアーキテクチャ再考へ〜．派生開発カンファレンス 2017，2017.
[12] Nakanishi, T., Jæger-Hansen, C. L., & Griepentrog, H-W. Evolutional development of controlling software for agricultural vehicles and robots. In Proc. on 3rd International Conference on Machine Control & Guidance (MCG 2012), pp. 101-111, 2012.
[13] 中西恒夫，久住憲嗣，福田晃．派生開発からプロダクトライン開発への漸次的移行プロセス XDDP4SPL におけるコア資産管理手法．情報処理学会技術研究報告．2013-SLDM-160， pp.1-6，2013.