ハードウェア仕様変更に対応する ソフトウェアプロダクトラインの提案

飯田 隆博† 松原 正裕† 成沢 文雄† 山口 東馬††

ソフトウェアプロダクトライン(SPL)は、ソフトウェアを効率的に開発するための手法として注目されている。従来の SPL では、要求仕様からフィーチャモデルを用いて可変性を固定し、ソフトウェアを構築する開発プロセスが取られる。しかし、自動車電動ブレーキ等の制御システムでは、機能仕様の変更を伴わないハードウェアの仕様変更が頻繁に行われ、既存の SPL 開発プロセスでは取り扱うことができない。

本研究では、SPL にてハードウェア仕様変更に対応する手法を提案する。本手法では、ハードウェアとソフトウェアの各仕様間の依存関係を示すマトリクスを用いて、1 つのハードウェア仕様変更が影響を及ぼす他のハードウェア仕様とソフトウェア仕様の追跡を容易にすることで、頻繁なハードウェア仕様変更への対応を可能にする。提案手法は従来の SPL の開発プロセスと両立可能であり、SPL による開発効率の向上を維持することができる。

Software Product Line to Deal with Hardware Specification Changes

TAKAHIRO IIDA† MASAHIRO MATSUBARA† FUMIO NARISAWA†
TOHMA YAMAGUCHI††

Software Product Line (SPL) is an effective method to develop software. In the SPL process, software is developed by fixing variants on a feature model according to the requirements. However, the conventional SPL does not treat hardware specification changes which occur in developments of control systems such like an electric brake system for automotives. These changes happen frequently without functional specification changes.

In this paper, we propose the software development method to treat hardware specification changes in SPL. In this method, the matrices are used which show dependences between each specification of hardware and software, to trace effects of a hardware specification change to other hardware or software specifications. These matrices make the tracing easy and enable to treat frequent hardware specification changes. The proposed method is compatible with the conventional SPL process, and software development productivity obtained with SPL is maintained.

1. はじめに

近年、組込みソフトウェアの開発規模が増大し、開発工数の増大が問題となっている。ソフトウェアプロダクトライン(SPL : Software Product Line) [1]は、ソフトウェア資源の再利用により開発を効率化する手法として注目されている。SPL では製品のロードマップを元に、機能構成を共通部と可変部に分けて開発を行うことで、ソフトウェアの再利用性を高める事ができる。設計者は要求から可変部を選択しソフトウェアを構築する。

しかし実際の制御システムの開発では、機能仕様の変更を伴わずに、ハードウェアの仕様変更を繰り返す製品があり、微細なソフトウェア変更が多発する。さらに一部のハードウェアの仕様変更は、他のハードウェアの仕様変更につながり、それがソフトウェアに対しても広範囲にわたる影響を及ぼす恐れがあるため、詳細レベルでの仕様変更はさらに顕著になる。

このような細かな仕様変更が繰り返される開発は従来の SPL では考慮されておらず、フィーチャモデルやアーキテクチャの仕様変更まで遡ってしまうと、SPL による開発効率向上の効果が得られなくなる。

本研究では、組込み制御システムへの SPL 適用において、ハードウェア仕様の頻繁な変更に対応する手法を提案する。

2. 従来手法

SPL では一般的に、機能構成を管理するフィーチャモデルを用いて可変性を管理する。フィーチャモデルは、製品系列の機能構成を上位から分解する形で表現するツリーであり、共通部・可変部を表現できる。図 1 に一般的な SPL 開発プロセスを示す。設計者は、製品の要求仕様に応じてハードウェア構成を決定し、要求仕様とハードウェア仕様を元に、フィーチャモデルを用いてソフトウェアの機能を選択する。そして、選択したソフトウェアの機能に対応するソフトウェアのコア資産を組み合わせることで、製品を構築する。以上のプロセスにより開発工数を削減できる。

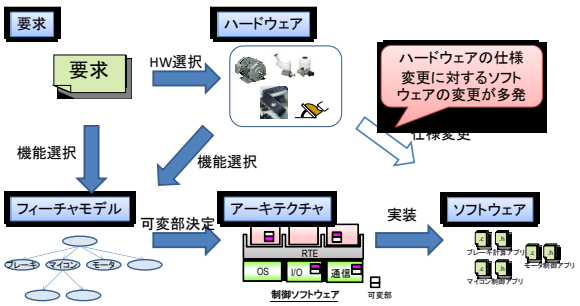

図 1 従来の SPL 開発プロセス

* † (株)日立製作所
Hitachi Ltd.
†† 日立オートモティブシステムズ(株)
Hitachi Automotive Systems, Ltd.

しかし従来の方式は、ハードウェアの仕様変更が頻発する事を考慮していない。図 2 で示すように、自動車の電動ブレーキなどでは、ハードウェアの開発を進めながらソフトウェアを開発していくため、常にハードウェアの仕様変更と対応したソフトウェアの微細な変更が行われる。またハードウェアを構成する各部品の結びつきが強いため、ハードウェアの一部の変更が広範囲の微細な変更につながる。そのため、車両毎のサイズの制約や、踏み心地、騒音などの対策でハードウェア仕様の変更が頻発する。

図 3 に仕様変更の影響が波及する例を示す。モータの回転音が問題になる場合、モータの径や筐体の変更で解決を図る。ここで、モータの仕様変更はモータ制御アプリの変更につながるが、ソフトウェアの変更で解決できない場合は、他のハードウェアに追加の変更を加える。もしモータを変更したならば、モータの回転を伝えるギアや、回転をピストン運動に変えるボールねじが変わることがあり、これらはマスタシリンダの液圧に影響する。マスタシリンダの仕様変更はブレーキペダルの動作に影響を与えうる。

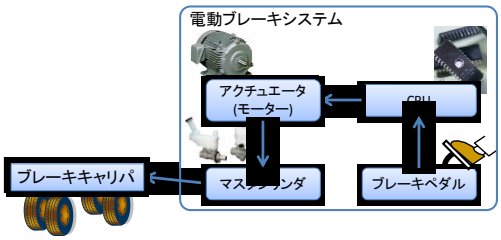

図 2 電動ブレーキシステムの構成

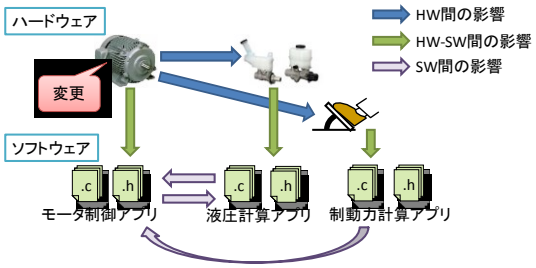

図 3 ハードウェア仕様変更の影響波及

このように、ハードウェアの仕様変更頻発による構成要素間の影響関係は把握困難であり、フィーチャモデルでは扱えない。この状況下で仕様変更を野放図に行うと、再利用性の低いソフトウェア部品が大量に出来上がる恐れがあり更に管理が難しくなっていく。上記問題を解決するためには、ハードウェアとソフトウェア間の影響関係を整理する仕組みが必要となる。

3. 提案手法

ハードウェア仕様変更の影響波及を容易に追跡するため、我々は図 4 に示す 3 つの依存関係を管理するマトリクスを用いる手法を提案する。

1. HW 間依存関係マトリクス
2. HW-SW 依存関係マトリクス
3. SW 間依存関係マトリクス

ハードウェアの仕様変更があった際に、設計者は HW 依存関係マトリクスを元に影響があるハードウェアを確認する。次に、変更のあったハードウェアが影響を与えるソフトウェアを HW-SW 依存関係マトリクスを用いて追跡する。最後に、SW 間の影響は SW 依存関係マトリクスで追跡する。以上により、1 つのハードウェア仕様変更が影響を与えるソフトウェア仕様を抽出できる。ハード仕様変更とソフト仕様変更の結果は、両者を対応させる形で管理する。なお、HW-SW 依存関係マトリクスは、ハードウェア仕様からフィーチャ選択する箇所でも利用することができる。

また上記の開発プロセスでは、フィーチャモデル等の上位設計は変更せずに再利用できるため、SPL による開発効率向上を維持することができる。図 5 に提案手法を用いた際の開発プロセスを示す。

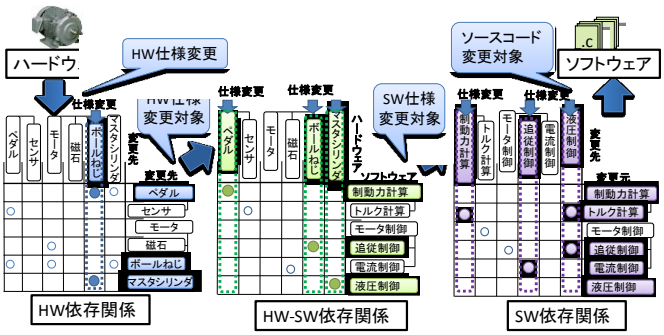

図 4 ハードウェアの仕様変更を追跡する 3 マトリクス

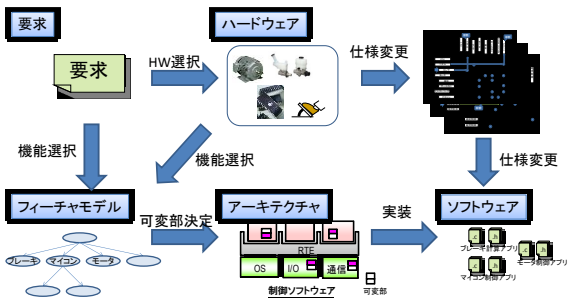

図 5 提案手法を用いた SPL 開発プロセス

4. 最後に

本研究では、開発中にハードウェアの仕様変更が多発する制御システムにおいて、ハードウェアの仕様変更がソフトウェアに及ぼす影響を管理する手法を提案した。本手法は、従来の SPL 開発プロセスとの両立が可能である。

今後の課題として、SysML 等のモデル言語により、上位の概念でハードウェアやソフトウェア間の関係を管理し、依存関係マトリクスを機械的に導出することが挙げられる。

参考文献

1) Klaus Pohl, Gunter Bockle, Frank J. van der Linden: Software Product Line Engineering German, Springer-Verlag (2005).