アジャイルプロダクトライン開発における ポートフォリオ駆動開発モデルと管理方法の提案と適用評価

林 健吾[†1]　青山 幹雄[†2]

概要：近年のソフトウェアシステムの開発では，製品のグローバル化や市場要求の高度化，複雑化により，多様性と俊敏性の実現が競争の鍵となっている．本稿では製品の俊敏な進化と多様性の実現を包括的に解決するために，APLE（Agile Product Line Engineering）の開発全体を包括した管理可能な開発モデルと管理方法を提案する．APLE 上で複数のプロダクトラインの開発全体を包括管理するためには，組織が保有する開発資源のポートフォリオ管理の確立が課題である．本提案では，APLE に対する 3 階層のポートフォリオ駆動開発モデルを提案し，マネジメントライフサイクルに基づく管理方法を提案する．APLE の開発モデルと管理方法を提供することで，各分野への適用を可能とし，APLE を運用する組織に対して包括的に管理可能な開発活動の推進に貢献することを期待する．

A Portfolio-Driven Development Model and Management Method of Agile Product Line Engineering

KENGO HAYASHI[†1]　MIKIO AOYAMA[†2]

1. はじめに

SPLE（Software Product Line Engineering）は，低コスト，高品質で SPL（Software Product Line）における多様な製品を開発するアプローチである[17]．複数の SPL を並行して開発する MPLE（Multiple Product Line Engineering）も実践されている[21]．自動車ソフトウェア開発では Bosch のガソリンシステム開発が，市場セグメントごとに SPL を構築することで MPLE を実践している[20].

自動車システムを含むソフトウェアシステムの開発では俊敏な製品進化が求められている．俊敏な製品進化に対しては，短いサイクルでインクリメンタルな開発を駆動する ASD（Agile Software Development）が実践されている[4][9][14].

製品の多様性とその俊敏な進化に対応するために ASD と SPLE を統合した APLE（Agile Product Line Engineering）も実践されている[5][9]．しかし，製品のグローバル化や市場要求の高度化，複雑化により[6]，SPLE におけるドメイン開発とアプリケーション開発が並行化するなど，さらなる俊敏性が必要となり，従来の開発方法では対応できない状況が生じている. このようなコンテキスト上の組織では，ビジネス上の要求と，要求を実現する製品ロードマップ，組織の資源，これらのトレードオフトライアングルを組織活動が継続できるよう最適化する方法の確立が必要である[3].

SPLE におけるアプリケーション開発では，単一システムのソフトウェア開発の規模は比較的小さい[17]．しかし，SPL から生成する製品が増加すると，個々の製品の納期が近接し，開発に割り当て可能な期間と必要な資源が短期間に集中する．加えて，俊敏な製品進化への要求によって，単一の SPL では要求の多様性を吸収できなくなり，複数の SPL の並行開発が必要となる[1].

SPLE においてトレードオフトライアングルを最適化するためには，単一の SPL 開発を管理するだけでは不十分である．なぜならば，組織が開発に割り当てられる資源には限りがあり，製品ごとの開発期間の重複は資源制約によって相互に依存するためである．トレードオフトライアングルを複数の製品開発に拡張して最適化を可能とするためには，複数の SPL を包括した開発モデルと管理方法の確立が必要である.

組織の限られた資源の投資や運用に対する意思決定を支援する活動としてポートフォリオマネジメントが実践されている[16]．ポートフォリオはビジネスゴールや戦略的な目標を達成する手段としてのプロジェクトやプログラム，およびその他のオペレーションの集合である（図 1）．ポートフォリオマネジメントは資源制約の下で複数の製品開発を包括して管理可能とする.

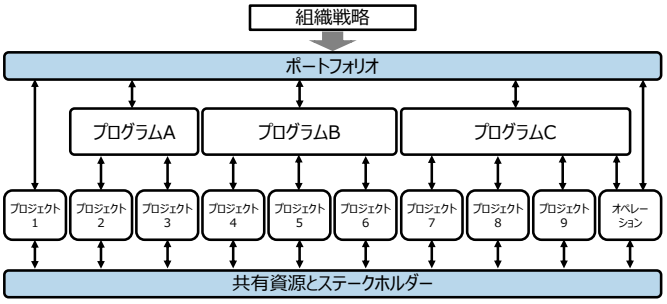

図 1 ポートフォリオの概念図

Figure 1 Portfolios High Level View

†1 (株)デンソー
DENSO CORPORATION
†2 南山大学
Nanzan University

本稿では，資源制約が複数の SPL 開発の最大制約であることに着目し，複数の SPL を包括してトレードオフトライアングルを最適化することを目的に,以下の課題を定める．

Q1:複数の SPL を包括管理する手段としてポートフォリオマネジメントを開発モデルに取り込むことは可能か？

Q2:提案する開発モデルに基づいて複数の SPL の包括管理を実現する管理方法はどのようなものであるか？

Q3：提案方法は実際の APLE の開発で有効であるか？

本課題の解決方法として，ポートフォリオ駆動開発モデルと管理方法を提案する．提案する開発モデルと管理方法を自動車システムのソフトウェア開発の実開発に適用し，提案方法の有効性を示す．本提案方法を提供することで，各分野の SPL のコンテキストへの適用を可能とし，SPLE を運用する組織に対して包括的に管理可能な開発活動の推進への貢献を期待する．

2. 関連研究

2.1 SPLE（Software Product Line Engineering）

SPLE は，ドメイン開発とアプリケーション開発の 2 つの開発領域で SPL の製品の多様性を低コストで実現するアプローチである[17]．ドメイン開発では SPL における共通性と可変性を分析してコア資産を構築する．共通性とは SPL の製品間で同一である特徴点の集合である．可変性とは SPL の製品間で差異が現れる特徴点の集合である．

SPL を投入するビジネス領域を製品ポートフォリオ分析によって決定する方法が提案されている[17]．製品ポートフォリオ分析では，製品の市場シェアと市場の成長度合いの 2 軸で市場領域を 4 象限に分け，製品戦略としての SPL 投入時の投資対効果を分析する．

製品戦略を立案する具体的な方法は提案されているが，SPL を投入してからのポートフォリオの運用管理の研究は少ない[18][19]．個々のアプリケーション開発を包括的に管理する開発モデルや管理方法の研究も少ない[7]．本研究では我々の研究を発展させた内容を報告する．

2.2 ポートフォリオマネジメント

ポートフォリオマネジメントは組織における投資や資産運用に対する意思決定を支援する活動である[16]．ポートフォリオの構成コンポーネントは定量化可能であり，識別，分類，評価，優先付け，認可される対象である．各コンポーネントは組織の限られた資源を競合する．

Jagroep は，製品やプロジェクトを包括したポートフォリオは，個別のプロジェクトではなくビジネスの短期的かつ長期的な継続を確保するためのコンセプトである，と定義している[10]．Bekkers は，ソフトウェア製品管理の 4 つのビジネス上の機能において，ポートフォリオマネジメントを製品ロードマップ，要求管理，リリース計画の前段に位置づく機能として提案している[2]．

製品やプロジェクトを包括してビジネス領域に関連付けるためにポートフォリオの概念は有効である．しかし，開発モデルと統合した管理方法は未確立である．

2.3 アジャイルポートフォリオマネジメント

ポートフォリオマネジメントでは資産と投資の運用を調整して適切に管理する必要がある．Krebs はプロジェクトと資源，資産の 3 つをポートフォリオの構成コンポーネントと定義し，ASD のフレームワークを利用して管理する方法を提案している[11]．

(1) プロジェクト：プロジェクトの投資対効果．
(2) 資源：人的資源の量と質（スキル）．
(3) 資産：開発済みの製品などの資産の保守や廃棄．

これらのポートフォリオを測定，監視して管理するために，Scrum などの ASD の開発フレームワーク[4][9][13][14]を利用する．ASD では，プロジェクトの開発量と組織の生産性を普遍量で見積もるのではなく，2〜3 か月間の実績から統計的に推定し，計画を段階的に詳細化していく．

Leffingwell は ASD のアプローチをトレードオフトライアングルを用いて価値駆動として提示している[13]．価値駆動では，固定されたリソースと納期に基づいて，要求を見積もって決定するアプローチを採る（図 2）．

価値駆動に基づいて，Leffingwell は大規模開発を複数チームで協調して開発するためのフレームワークである SAFe（Scaled Agile Framework）を提案している[13][14]．SAFe ではポートフォリオレベル，プログラムレベル，チームレベルの 3 階層で開発資源を運用する開発モデルと管理方法を提案している．

単一製品を複数チームで開発するマネジメントフレームワークは ASD において提案され実践されている[12][13]．また，複数チームのプロジェクトを包括して開発管理する方法も提案されている[13][22]．しかし，SPLE のように製品系列内の製品同士が相互に依存する場合の方法は報告されていない．価値駆動に基づくマネジメントフレームワークでは，製品同士の相互依存によって資源制約が大きくなると適用が難しくなると考えられる．本稿では後述するように資源制約をより重視することで適用可能性の向上を図る．

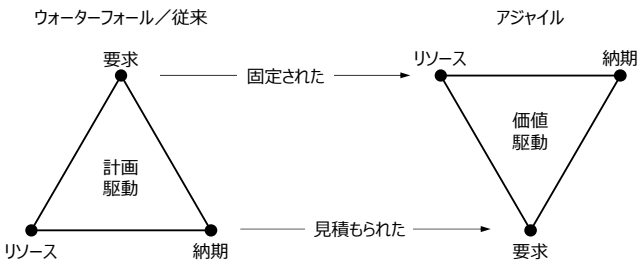

図 2 価値駆動のトレードオフトライアングル
Figure 2 Trade-off Triangle for Value Driven

3. アプローチ

3.1 研究課題とアプローチ

本稿では，前述の 3 つの研究課題に対して以下のアプローチを提案する．

Q1 : 複数の SPL を包括管理する手段としてポートフォリオマネジメントにおける共有資源の競合に着目して，ポートフォリオ駆動の開発アプローチを提案し，開発モデルとして展開する．

Q2 : 提案した開発モデルに基づき，ポートフォリオレベル，プロダクトレベル，スプリントレベルの 3 階層で複数の SPL を包括してマネジメントする方法を提案する．

Q3 : 自動車システムの APLE として，筆者らが従事している超音波センサシステム開発に提案方法を適用し，その効果を評価する．

本章では，複数の SPL を包括管理する上で，従来の計画駆動と ASD における価値駆動アプローチにおける問題点を示した上で, ポートフォリオ駆動アプローチを提案する．

3.2 SPL と計画駆動・価値駆動アプローチの問題

開発マネジメントに取り組む上で，要求と納期，リソースを要素としたトレードオフトライアングルにおいて，どの要素を可変点と定めるかの制約は重要である [3]．Leffingwell はトレードオフトライアングルの可変点によって計画駆動と価値駆動のアプローチを区別した [13]．計画駆動アプローチは，固定された要求に基づいて，リソースと納期を見積もって決定する．価値駆動アプローチは，固定されたリソースと納期に基づいて，要求を見積もって決定する．

SPL として生成する製品間には依存関係が生じる．図 3 に SPL の製品間の依存関係を示す．SPLE に限らず同時期の 2 つの製品開発では開発リソースの共有と競合が生じる．一方，開発元のコア資産を共有することで SPLE 特有の要求間の依存関係が生じる．先行した製品開発によってコア資産が成長すると後続製品の要求に影響する．例えば，先行製品で新たな可変点が生じると，新たに変異体を選択する要求が後続製品では必要となり，最低限実現必要な要求の量が増加する．

要求の依存関係は計画駆動アプローチで問題を生じる．要求を固定する計画駆動アプローチでは，先行製品の開発で後続製品の要求が変動すると，必要な開発リソースが増加し，開発を遅延させる方策に陥りやすい．その結果，開発の管理性の悪化を招く．価値駆動アプローチでは要求が可変であるため実現する要求量を切り詰めることで管理性の維持が可能である．

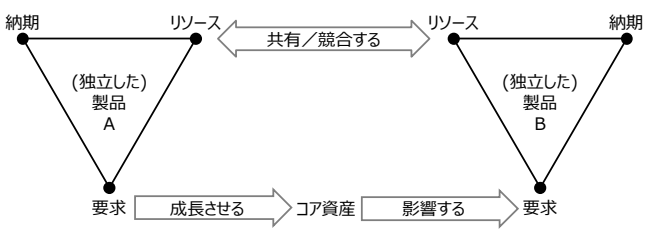

図 3 SPL のトレードオフトライアングルの相互依存

Figure 3 Interdependence of Trade-off Triangle for SPL

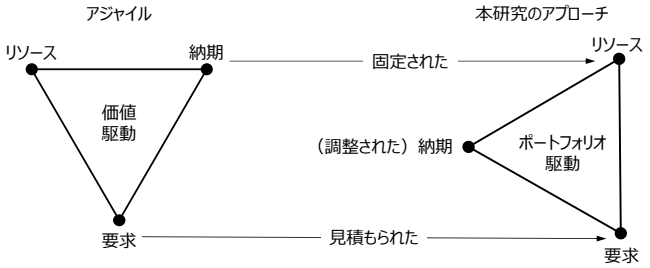

図 4 ポートフォリオ駆動のトレードオフトライアングル

Figure 4 Trade-off Triangle for Portfolio Driven

リソースの依存関係は価値駆動アプローチで問題を生じる．単一製品であれば要求を変動させることで固定した提供期間を維持することができる．しかし，複数製品の並行開発では個別の製品で顧客も異なる可能性があり，提供時期は開発組織だけでは決定できない．顧客要求に応じて要求以外の要素の変動を制約とするような，価値駆動とは異なるアプローチが必要である．

3.3 ポートフォリオ駆動アプローチ

本稿で提案するポートフォリオ駆動アプローチの概念を図 4 に示す．ポートフォリオ駆動アプローチでは，リソースを固定し，要求は可変可能な要素であるとして見積もって決定する．その上で製品の納期が調整される．

コンセプトは価値駆動アプローチと類似している．納期は ASD に基づき固定期間を単位として調整され得るが, 顧客要求に応じて調整される．“見積もった” 納期に定めるのではなくリソースの制約を受けて要求量で調整されるため, トレードオフトライアングルを傾け，納期を中間地点に配置することで表現している．

ソフトウェア開発では開発ドメインの開発スキルが重要であり，開発リソースの変動は容易でない．そこで，本アプローチでは組織の保有する開発リソースを最も優先する．開発リソースに基づいて複数製品を包括して要求の制約と納期の調整を実行し，開発を駆動することからポートフォリオ駆動アプローチと呼称する．

4. ポートフォリオ駆動開発モデル

4.1 開発モデルの概要

本稿で提案する，ポートフォリオ駆動アプローチに基づくポートフォリオ駆動開発モデルを図 5 に示す. 図 6 に本開発モデルで運用する成果物と組織におけるロールの関係を示す.

ポートフォリオ駆動開発モデルは，開発プロダクトをポートフォリオの構成コンポーネントとし，構成コンポーネントをプロダクトを実現するための要求量で定量化する．

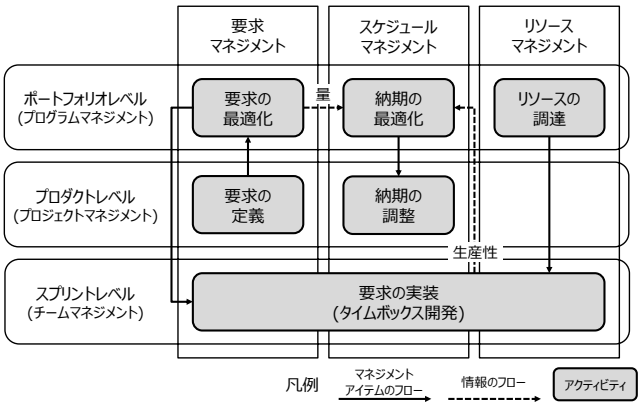

図 5 ポートフォリオ駆動開発モデル

Figure 5 Portfolio Driven Development Model

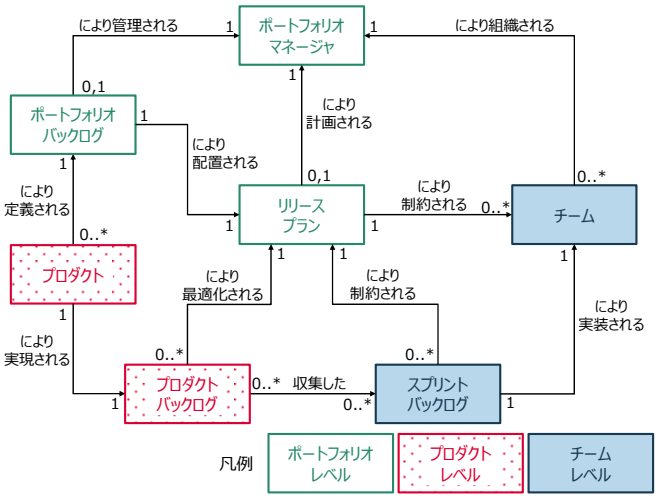

図 6 ポートフォリオ駆動開発モデルの成果物とロール

Figure 6 Artifacts and Roles of Portfolio Driven Development Model

競合される共有資源をリソース量ではなく，要求の時間当たりの実現量として生産性で定量化する．要求量と生産性の定量化は，ASD と同様にストーリポイントを用いる[4][13][14]．構成コンポーネントと共有資源の定量化単位をストーリポイントで統一することで複数コンポーネントの包括管理を実現することを特徴とする．

ポートフォリオ駆動開発モデルは，ポートフォリオレベル，プロダクトレベル，スプリントレベルからなる 3 層のマネジメントレベルで構成される．各層が協調して機能することで，複数の SPL を包括管理して要求，納期，リソースのトレードオフトライアングルを組織活動が継続できるよう最適化する．以下に各マネジメントレベルの詳細を示す．

4.2 スプリントレベル（チームマネジメント）

スプリントレベルは共有資源を生産性として定量化する責務を担う．共有資源が安定した生産性を実現できるようチームマネジメントする．チームマネジメントは ASD[4][9][13][14]のフレームワークと規律に則り，タイムボックス開発により実現する．

チームはポートフォリオマネージャによって組織され，チームごとに開発と生産性がマネジメントされる．生産性を測定するためのスプリントはタイムボックス(固定期間)の反復であり，固定期間は 1〜4 週間で任意に定めることができる．スプリントの開発アイテムの集合であるスプリントバックログは，複数のプロダクトバックログからリリースプランの制約に従って収集して作成する[7][22].

ポートフォリオ駆動として，スプリントレベルのチームマネジメントの目標は生産性の安定性の実現である．SPLE では可変性を中心に開発が反復されることを利用してプロセス資産を活用する[7]，可変性を分析してインクリメンタル開発を実現する[8]など，APLE としての施策を取り入れて生産性を安定させる．

4.3 プロダクトレベル（プロジェクトマネジメント）

プロダクトレベルはポートフォリオの構成コンポーネントである開発プロダクトを要求量で定量化する責務を担う．ポートフォリオレベルで調整を受けた要求と納期に適合させられるようにプロジェクトマネジメントする．プロジェクトマネジメントのアクティビティとして要求のスコープ定義と納期の調整を実行する．

要求のスコープ定義では，プロダクトオーナーあるいはプロジェクトマネージャがプロダクトの開発アイテムであるプロダクトバックログを定義する[4][9][13][14]．定義されたプロダクトバックログはポートフォリオレベルでリリースプランによって最適化される．

最適化された要求の量と，他のプロダクトが必要とするリソースとその納期によって，当該プロダクトの納期をステークホルダと調整する．

本開発モデルでは，プロダクトレベルのプロジェクトマネジメントは単独では完結しない．ポートフォリオレベルのプログラムマネジメントの統制を受けながらマネジメントを履行する．

4.4 ポートフォリオレベル（プログラムマネジメント）

ポートフォリオレベルは共有資源の生産性と開発プロダクトの要求量，開発プロダクトの要求納期に基づいて，ポートフォリオの構成コンポーネントである開発プロダクトを優先付けする責務を担う．また，構成コンポーネント間の要求の相互依存を反映して要求量と納期を最適化する責務を担う．プログラムマネジメントのアクティビティとして，リソースの調達，要求の最適化，納期の最適化を実行する．本節ではアクティビティの概要を示し，具体的な管理方法については次章にて詳細を示す．

リソースの調達では，1〜3 年単位での SPL の投資計画に基づいて人的資源を調達し，チームを編成する．SPL のアプリケーション開発量を見積もってチームの増員，減員或いは編成の見直しを実施する．

要求の最適化では，プロダクトレベルで定義されたプロ

ダクトバックログを検証し，チームの生産性とプロダクトのリリース，同一 SPL のその他プロダクトへの影響を考慮して，プロダクトバックログを最適化する．要求の最適化は，ポートフォリオに存在するすべての構成コンポーネントを実現する要求量を最小化しながら，各プロダクトの納期の調整幅に収まる要求量を極大化することである．

納期の最適化では，チームの生産性とすべてのプロダクトを包括的に評価して，リソースの平準を保つことを目指す．個々のプロダクトのリリース計画と，各チームに対してスプリントごとのプロダクトの割当てをリリースプランとして作成する．

各プロダクトは企画が承認された時点でポートフォリオマネージャによってポートフォリオバックログ上に定義される．ポートフォリオマネージャはロールである．個別の人員が担ったり，プロジェクトマネジメントオフィスなどのチームで担ったり，実現方法は任意である．複数の SPL を包括して意思決定できる規律があればよい．

5. ポートフォリオ駆動の管理方法

5.1 マネジメントライフサイクル

本稿で提案したポートフォリオ駆動開発モデルに基づいた管理方法のライフサイクルを図 7 に示す．本ライフサイクルは以下 6 つのフェーズで構成される．

(1) Establish（起案）
(2) Evaluate（評価）
(3) Prioritize（優先順位付け）
(4) Select（選択配置）
(5) Manage（プロダクトマネジメント）
(6) Feedback（フィードバック）

Establish から Manage までの 5 つのフェーズは Wysocki が提案したプロジェクトポートフォリオマネジメントのライフサイクルに基づいている[23].

複数 SPL を包括管理する場合，SPL としての相互作用を考慮する必要がある．個別のプロジェクトにおけるポートフォリオマネジメントに複数 SPL の相互作用を調整しマネジメントする方法を各フェーズにおいて規定する．

5.2 Establish（起案）

Establish はポートフォリオレベルで実行されるフェーズである．組織のポートフォリオ戦略から，SPL を利用したアプリケーションをリリースする量から見積もって，1〜3 年の期間範囲でのリソース調達計画を立案する．

初期のリソース調達計画は単にリソース量とチームの人員数を計画する概要計画である．以降のライフサイクルフェーズでは，ここで定めたリソースの制約に則って活動する．Feedback フェーズから得られる生産性やプロダクトの開発量の実績データを利用して，3 か月などの期間ごとに計画を洗練させていく．

本管理方法において，Establish はプロジェクト開発に先立って実行されるフェーズであると共に，反復して実行されるフェーズである．

5.3 Evaluate（評価）

Evaluate はポートフォリオレベルとプロダクトレベルで協調して実行するフェーズである．プロダクトレベルで見積もったプロダクト要求を，ポートフォリオレベルで SPL として最適化して見積りを洗練する．本フェーズは，プロダクト要求の最適化と管理対象からのプロダクトのリジェクト判定，プロダクト開発量の見積りの 3 つのプロセスから構成される．

5.3.1 プロダクト要求の最適化

プロダクト要求は，SPL の選択と同一 SPL の他プロダクトへの影響を評価して最適化する．

組織が複数の SPL を運用する場合，プロダクトをどの SPL から開発するかによって要求を実現するコストが変動する．優先度が高い要求を満たす上で最もコストが小さくなる SPL を選択する．例えば，2 つの SPL のプラットフォームが異なり，SPL 間の機能差を解消してプロダクト向けの新たな要求を実現するよりもプラットフォームへの適合にリソースを要するのであれば，プラットフォームの適合を不要とする SPL を選択する．

選択された SPL から開発されるプロダクトに未着手プロダクトに含まれる場合，3.2 で示したようにコア資産を通して開発が相互に依存する．共通性の変更を伴う要求開発，あるいは新たな可変性の追加を伴う要求開発が含まれる場合，未着手プロダクトの開発コストの変動を評価する．例えば，追加されたプロダクト A と未着手プロダクト B があった場合．A が B に先行しており，B で実現必要な要求が先に A で実現されるのであれば，B の開発量はその分低減する．また，A で新たな可変点が開発される場合，B では可変点を新たにコンフィグレーションする必要が生じるため，B の開発量はその分増加する．

SPL の選択とその他プロダクトへの影響評価によって，

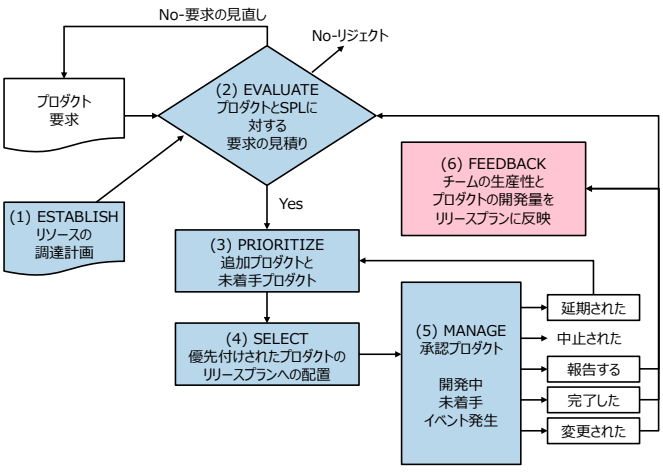

図 7 ポートフォリオ駆動マネジメントのライフサイクル
Figure 7 Portfolio Driven Management Life Cycle

優先度の低い, すなわち価値の低い要求が識別された場合, これを取り除いてプロダクト要求の最適化を完了する．この時点で開発量と計画への配置は仮実行する．仮として計画に配置した結果，すべての要求を実現しても計画に収まる見込みがあれば要求は取り除かない．計画に収まらない場合に要求を除くよう意思決定する．リソースの圧迫を軽減し，組織の投資対効果を高める．

5.3.2 管理対象からのプロダクトのリジェクト判定

最適な SPL がなく，大きなコア資産の進化が求められる場合は，SPL のアプリケーション開発として管理すべき対象ではないプロダクトとしてリジェクトする．この場合は従来のプロダクトのマネジメント対象として開発投資の判断が下される．

5.3.3 プロダクト開発量の見積り

筆者らの経験では，同一の SPL から開発されるプロダクトは開発量の見積りが類似となることがわかっている[7]. Feedback フェーズで同一 SPL の開発量の実績データが蓄積されている場合は，実績データと該当プロダクトのプロダクトバックログ並びに当該プロダクトの最適化された要求を比較することで正確性の高い見積り値を推定できる．

選択した SPL から初めてアプリケーション開発する場合は，異なる SPL のプロダクト開発を比較対象にしたり，プロダクトレベルで開発アイテムを詳細化したり，従来の方法で見積り値を決定する．

5.4 Prioritize（優先順位付け）

Prioritize はポートフォリオレベルで実行されるフェーズである．新たに追加されたプロダクトと未着手プロダクトを比較してプロダクト開発の優先順位を決定する．各プロダクトには個別に納入計画がある．各納入から得られる投資のリターンと納期を基準として優先順位を決定する．

5.5 Select（選択配置）

Select はポートフォリオレベルとプロダクトレベルで協調して実行されるフェーズである．プロダクトのリリースプランへの配置，納期調整のプロセスを実行して納期の最適化が図られる．

5.5.1 プロダクトのリリースプランへの配置

Evaluate で見積もられたプロダクトの開発量と, Prioritize で決定された優先度に従って，複数 SPL を包括したリリースプラン上にプロダクトを配置する．Feedback フェーズでチームの生産性が得られていれば統計値を，得られていなければ推定値を利用してガイドラインを規定する．

リリースプランは固定期間としてのスプリントで区切られる．プロダクトの納期から逆算してプロダクトの開発量だけスプリントごとにストーリポイントを配分する．

5.5.2 プロダクトの納期の調整

プロダクトのリリースプランへの配置が完了した時点で, チームの生産性を超えたスプリントが存在する場合は, プロダクトの納期を調整する．調整方法には単一の納期の調整とプロダクトのインクリメンタルな納入計画の調整である．

(1) 単一の納期の調整：プロダクトの単一の納期を早期化，もしくは延期して開発量がリソースに収まるように調整する．
(2) 納入計画の調整：プロダクトの納入計画上の複数の納期を統廃合して納入を減らす，あるいは納入を分割して増やす代わりにリソースが競合している納入の開発量を減らすよう調整する．

いずれの調整も納期を最適化して固定したリソース内に開発量を収める代わりに, 価値創出に遅延が発生したり, イベント追加のコストのオーバーヘッドが発生したり，トレードオフが伴うためにポートフォリオレベルでの判断が求められる．組織活動を継続するために，リソース不足やリソース競合を理由としたプロダクトの遅延が，許容を超えないことを目標とする．

5.6 Manage（プロダクトマネジメント）

Manage はプロダクトレベルとスプリントレベルで実行されるフェーズである. 従来の ASD の規律に従ってスプリントレベルでタイムボックス開発し，プロダクトレベルでプロジェクトマネジメントを実行する．

開発中，未着手プロダクトを対象に実行され，開発の要求スコープや納期に変化が生じたなどのイベント発生を検出してその他のフェーズへの移行を実行する．ポートフォリオレベルでのフェーズ移行を引き起こすイベントは，延期，中止，報告，完了，変更の 5 つである．

(1) 延期：プロダクトの開発が延期された場合，未着手プロダクトとして Prioritize フェーズに遷移する．
(2) 中止：プロダクトの開発が中止された場合，プロダクトのライフサイクルを終了する．
(3) 報告：チームが 1 スプリントの開発を終えたことを報告し，一時的に Feedback フェーズに遷移する．
(4) 完了：プロダクトの開発が完了した場合，Feedback フェーズに遷移してその後プロダクトのライフサイクルを終了する．
(5) 変更 : プロダクトの要求スコープが変更された場合, Evaluate フェーズに遷移して要求の最適化からフェーズを実行し直す.

5.7 Feedback（フィードバック）

Feedback はプロダクトレベルからのプロダクトの開発量とスプリントレベルからのチームの生産性の報告に基づいてポートフォリオレベルで実行されるフェーズである．リリースプランを洗練させて複数 SPL の包括した管理性を向上するためのフェーズである．

プロダクトの開発量は Establish，Evaluate でのプロダクトの開発量見積りや Select での納期調整時の分割・統廃合後の開発量の見積りに利用する．各開発ドメインや組織の資産運用に基づいて，SPL やリリース目的，リリースフェ

ーズなどで分類して実績値を蓄積しておく．

チームの生産性は Establish，Select での生産性のガイドラインに利用する．チームの生産性は 7 スプリント分の移動平均を利用する方法が提案されている[4][7]．生産性は未学習の新たな SPL の開発に取り組む場合や，人員の増減によって変動するため，チームと協調して標準な生産性のガイドラインを決定する必要がある．

6. 自動車システム APLE への適用

6.1 適用開発のコンテキスト

複数の SPL を包括管理するためのポートフォリオ駆動開発モデルと管理方法を，自動車システムの APLE として超音波センサシステム開発に適用した．図 8 に開発環境を示す．適用した APLE では 2 つの組織で開発を分担している．コア製品開発チームは各 SPL のコア資産を開発し，いくつかのアプリケーションをコア製品として開発する．派生製品開発チームは各 SPL を継承して派生製品を開発する．

適用対象は派生製品開発チームにおける複数 SPL を利用した開発である．筆者らの一人はポートフォリオマネージャとして開発に参加している．各プロジェクトで得られた実データで本方法の有効性を評価した．

6.2 適用期間と SPL・チームの運用

適用期間は 2016 年 10 月～2018 年 2 月の 15 か月間である．1 スプリントは 2 週間である．この期間内に運用された SPL は 4 本であった．この期間内に開発を終えた 12 のプロダクトを対象に評価した．

開発リソースとして編成されたチームは 2 チームである．1 つ目のチームは 52 スプリント運用し，人員は 2 名，5 名，3 名と増減させている．2 つ目のチームは 35 スプリント運用し，人員は 2 名，3 名，5 名，7 名と順次増員している．

ポートフォリオレベルのポートフォリオマネージャは 1 名，プロダクトレベルのプロジェクトマネージャは 3 名配置し，各アクティビティを担った．

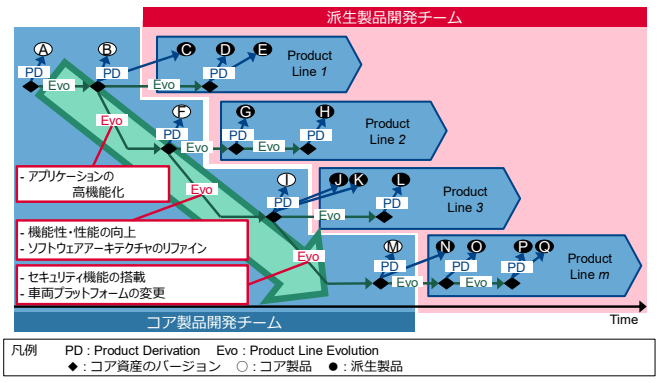

図 8 自動車システム APLE の開発環境

Figure 8 Automotive System Products Development on APLE

7. 評価

7.1 評価方法

提案方法の効果を以下の観点で評価した．

(1) リソースの生産性の予測可能性

(2) プロダクトの開発量の予測可能性

(3) トレードオフトライアングルの制御可能性

(1)，(2)にて，ポートフォリオ駆動開発モデルにおける，包括管理のための制御パラメータの正確性を評価する．(3)にて管理方法としての妥当性を評価する．

7.2 リソースの生産性の予測可能性

ポートフォリオ駆動開発モデルでは，各チームの生産性を基にリソース調達計画やリソースプランを策定する．生産性が安定していて予測可能であることは，ポートフォリオ駆動の管理性を確保するための必要条件である．

生産性の予測可能性を評価するために生産性の変動性を指標として用いる[7]．チームの生産性は従来の ASD と同様にスプリントごとの開発量から求める[4][13]．本稿ではそれぞれ以下の計算式により算出した．

1 スプリント＝2 週間 (1)

$$生産性(k)=\sum_{n=k-6}^{k}(スプリントnの開発量)/7 \quad (k\geqq 7) \qquad (2)$$

$$生産性の変動性(k)= \left|生産性(k)-生産性(k-1)\right| / 生産性(k-1)\times 100 \quad (k\geqq 8) \qquad (3)$$

図 9 に 2 つのチームの開発量と生産性，生産性の変動性を示す．チームの人員数は評価期間において増減しているため，人員数の切り替わりに合わせて開発量の色の濃淡を変えて区別している．

表 1 に 2 チームを合計した生産性の変動性の分布を示す．変動性が 10%以下のスプリント数は 63 で全体の 85.1%の割合であった．10%超の変動性を示すスプリントの多くは，

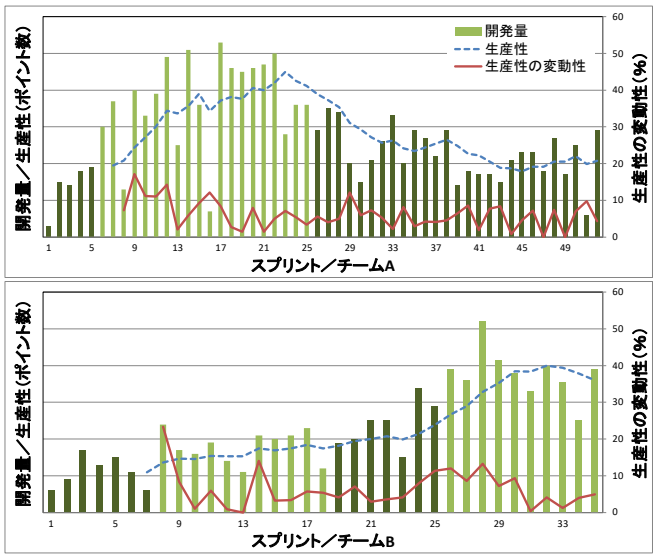

図 9 スプリントごとのチームの生産性と変動性

Figure 9 The Productivity and Productivity Volatility of Each Sprint

表 1 生産性の変動性の分布
Table 1 Distribution of Productivity Volatility

生産性の変動性	＜10%	＜15%	＜20%	≧20%	合計
スプリント数	63	9	1	1	74
割合（%）	85.1	12.2	1.4	1.4	100.0

チーム人員数の増減で生産性が変化したことが要因である．

リソースを一定に保っている期間は生産性の変動性が低く，生産性が安定しており，生産性の予測可能性が高くなる結果が得られた．

7.3 プロダクトの開発量の予測可能性

プロダクトの開発量はポートフォリオ駆動開発モデルにおいてリソースプランの策定に利用する．適切に納期を調整し，固定したリソース内に開発量を収めるために必要である．プロダクトの開発量の予測可能性はポートフォリオ駆動の管理性を確保するための必要条件である．

表 2 に開発量の推定値と実績値の比率を示す．対象期間中の 48 回の製品リリースが母集団である．各リリースは，開発元の SPL，対象プロダクト，プロダクト中のインクリメント数が異なっている．母集団を以下 3 つに分類した．

(1) F/B（Feed Back）なし：見積りの参考実績値なし．
(2) F/B あり（初期）：見積りの参考実績値が 1 件あり．
(3) F/B あり（後期）：見積りの参考実績値が 2 件以上．

F/B なしに対して F/B ありの比率は平均値が低く見積りの正確性が高い．標準偏差も小さくばらつきが改善されている．また，F/B ありの初期に対して後期は F/B の繰り返しで正確性もばらつきも改善され，プロダクトの開発量の予測可能性が向上する結果が得られた．

7.4 トレードオフトライアングルの制御可能性

トレードオフトライアングルの制御可能性として，要求，納期，リソースの変動係数を比較する．変動係数は各パラメータの値の標準偏差を平均値で割り無単位とすることで，変動の程度を複数のパラメータの変動係数で比較できる．各パラメータの値の定義を以下に示す．

(1) 要求：48 回の製品リリースの開発量．
(2) 納期：1 スプリントの固定期間である 14 日間を基準とした場合の納期調整日数．調整なしであれば 14，7 日延期した場合は 21，7 日前倒した場合は 7．
(3) リソース：リソースはスプリント期間内で増減しないため変動係数は 0 である．加えてリソースの能力を比較する．各チームで同一人員数である期間の 1 スプリント辺りの開発量をリソースの能力とした．複数期間が対象となるため全期間の変動係数の平均を求めている．

表 2 開発量の推定値と実績値の比率
Table 2 Ratio of the Estimated and Actual Value of Development Amount

	F/Bなし	F/Bあり（初期）	F/Bあり（後期）
標本数	15	11	22
平均（%）	124.3	118.4	112.7
標準偏差（%）	53.3	31.5	24.2

表 3 トレードオフトライアングルの変動係数
Table 3 Coefficient of Variation for Trade-off Triangle

	要求	納期	リソース	リソース（能力）[参考]
変動係数	0.70	0.38	0.00	0.25

表 3 にトレードオフトライアングルの各変動係数を示す．要求，納期，リソース能力，リソースの順に変動係数が小さくなっている．提案した管理方法では，リソースとリソースの能力のばらつきを抑えて固定化し，要求と納期を可変させてトレードオフトライアングルを最適化するよう制御している結果が得られた．

8. 考察

8.1 複数の SPL を包括管理可能とする開発モデル（Q1）

8.1.1 リソースの生産性の予測可能性

7.2 の結果から，提案した開発モデルにおいてリソースの生産性は安定し，高い予測可能性を示した．生産性の安定は APLE としての開発方法に基づいて実現されている[7][8]．SPLE の資産の再利用と ASD の規律による適切な学習効果の利用が生産性の予測可能性を実現したと考えられる．

8.1.2 プロダクトの開発量の予測可能性

7.3 の結果から，提案した開発モデルにおいてプロダクトの開発量は高い予測可能性を示した．SPL を用いた開発は再利用性が高い．開発モデルのフィードバック構造が予測可能性の向上を実現するために機能したと考えられる．

開発量の定量化にストーリポイントを利用していることも予測可能性の向上に寄与していると考えられる．ストーリポイントは実工数の見積りに対して精度は低下するがばらつきを低減させる効果がある[4]．

ばらつきを低減したストーリポイントによる見積りと SPL の高い再利用性から，フィードバックによる開発量の高い予測可能性が実現できたと考えられる．

8.1.3 SAFe との共通点と相違点

提案した開発モデルは，ASD の規律に基づいて 3 階層で開発を管理しており SAFe との共通点が多い[13][14]. SAFe は価値駆動で製品のリリース周期を固定している．組織が自律的かつ線形に製品を進化させる場合に有効である．

提案モデルではポートフォリオ駆動で製品のリリースタイミングを調整する．共通性を有した複数のバリエーションを備えたプロジェクトを運用する場合に有効である．

8.2 複数の SPL の包括管理する管理方法（Q2）

8.2.1 トレードオフトライアングルの制御可能性

7.4 の結果から，提案した管理方法ではトレードオフトラインアングルを要求と納期を可変して制御している．その結果, 15 ヶ月間継続的に 48 回の製品リリースを実現し，包括的に 4 本の SPL を運用することを可能とした．

要求と納期の最適化は納入先への申し入れだけではなく，リリースプランを共有することで実現した．リソース

の状態を共有し，その他製品を考慮して要求と納期が最適化されている状態も公開して共有することで，納入先の理解を得て協調した最適化を実現することができた．

8.2.2 適用コンテキストの前提条件

提案した管理方法は要求と納期が可変できることが前提である．適用した自動車システム開発では，フェーズ開発であることから要求を調整する余地があり，組み込み製品であることからハードウェアの製造期間を納入バッファとして利用できた．要求と納期が固定された開発の場合，提案した管理方法は適切に機能しない可能性が高い．

複数製品が相互依存関係を有していること，また製品への要求と納期を開発組織が自律的に制御できないことも前提となる．相互依存関係を有していない場合はより簡便な管理方法が望ましい．要求と納期を自律的に制御できるのであれば，SAFe など従来の価値駆動アプローチの開発モデルが適している．

SPL として開発コストが見通せるアーキテクチャを備えていることも前提である．可変点が整理されており類似の開発が繰り返されることを前提としている．この前提が満たされない場合，本開発モデルを適用しても生産性と開発量が安定しない．

これらの前提を満たせば，MPLE ではなく SPLE のコンテキストでも本管理方法は適用可能である．

8.3 APLE における有効性（Q3）

8.3.1 提案方法の APLE における有効性

8.2.1 で述べた通り，提案した開発モデルと管理方法は，実際の APLE で有効であった．また，定性的な有効性として納入に対する高いコミット性が挙げられる．開発量と生産性が高い予測可能性を備えることで，複数の SPL を運用した複雑な開発でも確信をもって納期を提案できた．コミット遵守は納入先との信頼関係の醸成にも有効である．

8.3.2 スケーラビリティ

本稿での適用評価は 2 チームで最大 10 人規模の開発を対象とした．ポートフォリオ駆動であることから，チーム数の増加へは SAFe と同様に対応可能と考えられる．但し，運用する SPL の数が増えると同一のコア資産を進化させる衝突が生じ得る．また，製品リリースの本数が増大すると納期を最適化するための管理が複雑化する．スケーラビリティの確保には管理ツールの整備が必要である．

9. まとめ

APLE において，ポートフォリオ駆動のアプローチに基づき，複数のプロダクトラインの開発全体を包括した管理可能な開発モデルと管理方法を提案し，実プロジェクトに適用した．その結果，チームの生産性とプロダクトの開発量の予測可能性を高めて開発の管理性を向上した．また，要求と納期を制御することでトレードオフトライアングルを最適化することに成功した．

今後はドメイン開発も包括した管理可能な開発モデルの確立を目指す．

参考文献

[1] M. Aoyama, Continuous and Discontinuous Software Evolution, Proc. of IWPSE 2001, ACM, Sep. 2001, pp. 87-90.
[2] W. Bekkers, et al., A Framework for Process Improvement in Software Product Management, Proc. of EuroSPI 2010, Springer, Sep. 2010, pp. 1-12.
[3] B. Blackman, et al., Managing Agile Open-Source Software Projects with Microsoft Visual Studio Online, Microsoft Press, 2015.
[4] M. Cohn, Agile Estimating and Planning, Prentice Hall, 2005.
[5] J. Díaz, et al., Agile Product Line Engineering- A Systematic Literature Review, Software: Practice and Experience, Vol. 41, No. 8, Jul. 2011, pp. 921-941.
[6] C. Ebert and J. Favaro, Automotive Software, IEEE Software, Vol. 34, No. 3, pp. 33-39, May-Jun. 2017.
[7] K. Hayashi, et al., Agile Tames Product Line Variability, Proc. of SPLC 2017, ACM, Sep. 2017, Sevilla, Spain, pp. 180-189.
[8] 林 健吾, 青山 幹雄, マルチプロダクトライン開発における可変性の構造分析に基づくアジャイルアプリケーション開発方法の提案と評価, SES 2017 論文集, 情報処理学会, Aug.-Sep 2017, pp. 190-197.
[9] P. Hohl, et al., Searching for Common Ground, Proc. of ICSSP 2017, ACM, Jul. 2017, pp. 70-79.
[10] E. Jagroep, et al., Framework for Implementing Product Portfolio Management in Software Business, Software Project Management in a Changing World, Springer, 2014, pp. 193-221.
[11] J. Krebs, Agile Portfolio Management, Microsoft Press, 2008.
[12] M. Laanti, Is Agile Portfolio Management Following The Principles of Large-Scale Agile?, Proc. of IEEE Agile 2015, IEEE, Aug. 2015, pp. 92-96.
[13] D. Leffingwell, Agile Software Requirements, Addison-Wesley, 2011.
[14] D. Leffingwell, SAFe® 4.0 Reference Guide: Scaled Agile Framework® for Lean Software and Systems Engineering, Addison-Wesley, 2016.
[15] F. van der Linden, et al., Software Product Family Evaluation, Proc. of SPLC 2004, LNCS Vol. 3154, Springer, Aug.-Sep. 2004, pp. 110-129.
[16] PMI, The Standard for Portfolio Management 4th ed., PMI, 2017.
[17] K. Pohl, et al., Software Product Line Engineering, Springer, 2005.
[18] J. Savolainen, et al., Combining Different Product Line Models to Balance Needs of Product Differentiation and Reuse, High Confidence Software Reuse in Large Systems, Proc. of ICSR 2008, LNCS Vol. 5030, Springer, May 2008, pp.116-129.
[19] K. Schmid, Scoping Software Product Lines: An Analysis of an Emerging Technology, Proc. of the 1st Software Product Lines Conference, Kluwer Academic, Aug. 2000, pp.513-532.
[20] M. Steger, et al, Introducing PLA at Bosch Gasoline Systems: Experiences and Practices, Proc. of SPLC 2004, LNCS Vol. 3154, Springer, Aug.-Sep. 2004, pp. 34-50.
[21] B. Tekinerdogan, et al., Supporting Incremental Product Development using Multiple Product Line Architecture, Int'l J. of Knowledge and Systems Science, Vol. 5, No. 4, Oct.-Dec. 2014, pp. 1-16.
[22] M. Turek, et al., Multi-Project Scrum Methodology for Projects Using Software Product Lines, Proc. of ISAT 2015 - Part III, AISC Vol. 431, Springer, 2016, pp. 189-199.
[23] R. K. Wysocki, Effective Project Management, 7th ed., Wiley, 2013.