組み込み機器の派生開発におけるフィーチャーモデルとXDDPを組み合わせた開発効率化の試み

～アジャイルの風味も添えて～

株式会社ニコン
石川 琢海

2023/5/26

株式会社 ニコン

目次

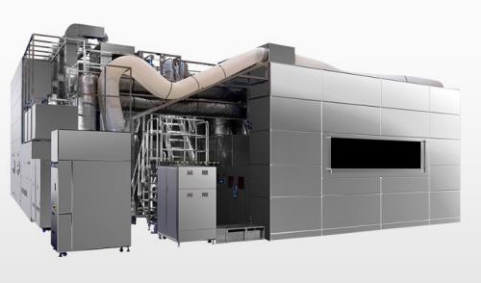
- 会社紹介・製品紹介
- 背景
- XDDPの導入
- フィーチャーモデルの活用
- 改善効果
- まとめ・所感

会社紹介・製品紹介

株式会社ニコン

https://www.jp.nikon.com/company/

半導体露光装置



FPD露光装置

超解像顕微鏡

双眼鏡

カメラ用交換レンズNIKKORレンズ

デジタル一眼レフカメラ

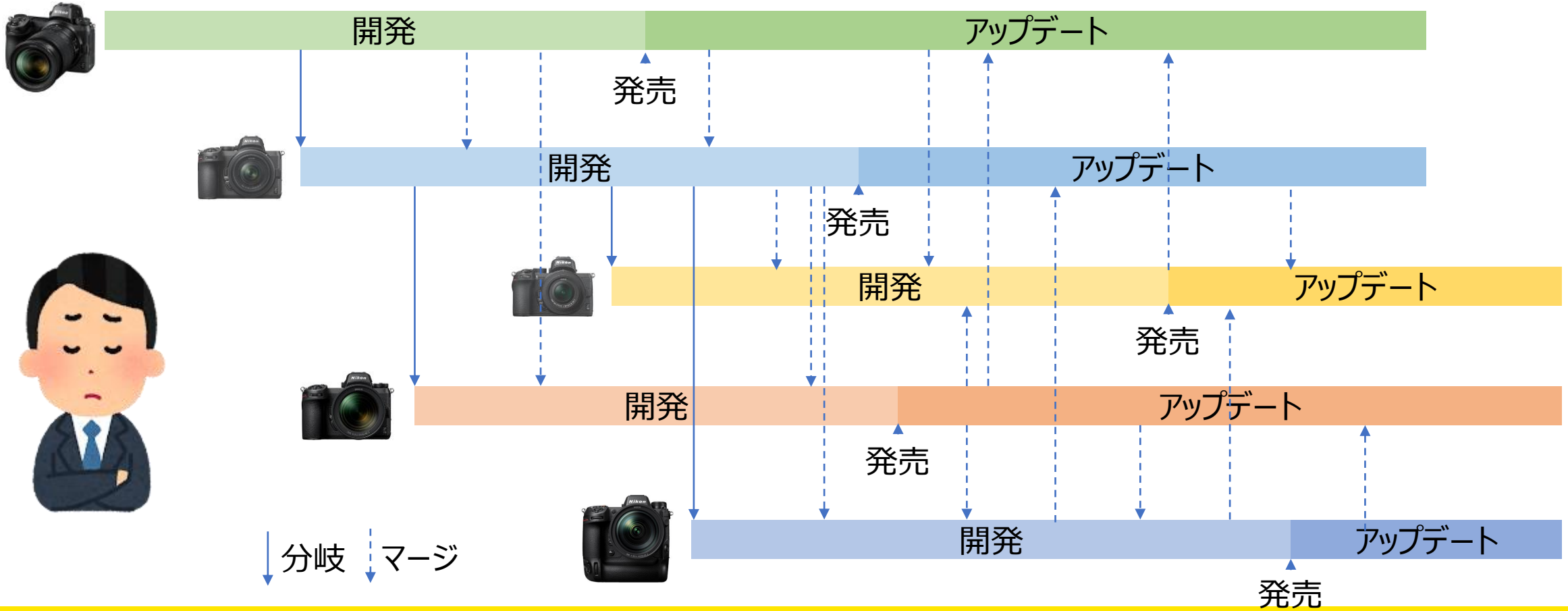
ミラーレスカメラ

今回の説明対象

背景

混沌とした派生開発の実情

並行機種開発でのソフトウェアの分岐とマージの繰り返しが開発効率向上の課題に。



多彩な”特徴”を持つ製品

専門性の高い特徴が数多く組み合わさることによりソフトウェア構造の複雑化が進み、ソースコード管理や開発者の適正配置の難易度が上昇が課題に。

XDDPの導入

XDDP本格導入の決断

XDDPとは

・品質と生産性を追求したプロセス

　・ベースとなる製品から”差分情報“に基づいた開発

　・ムダの徹底排除

※日科技連 実践！派生開発を成功させるXDDPセミナー　（講師：古畑 慶次氏）より。

よし！XDDPやろう！

でも

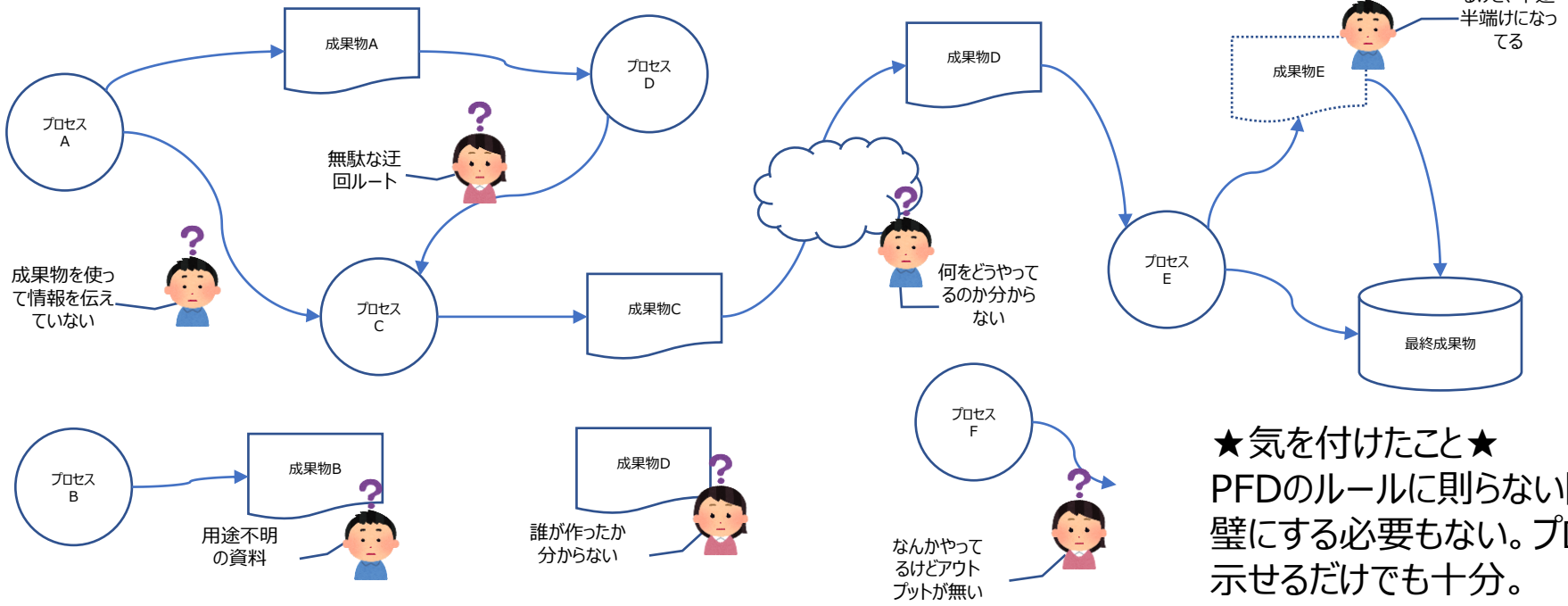
開発者は大勢いる。
みんなにやってもらえるだろうか？

導入準備：プロセス

現状をプロセスをPFD(Process Flow Diagram)に書き起こす

⇒現状の課題点を可視化し「このままではよくない。とにかく変えよう。」という意識をもってもらうための情報として利用する。

【課題があるプロセスのイメージ】



バイヤーだね。とにかくなんか変えなきゃ

★気を付けたこと★

PFDのルールに則らない図になっても気にしない。完璧にする必要もない。プロセスが不透明であることが示せるだけでも十分。

導入準備：認知拡大

そもそもXDDPを知らない開発者が大半。

⇒まずはXDDPを知ってもらう必要がある。

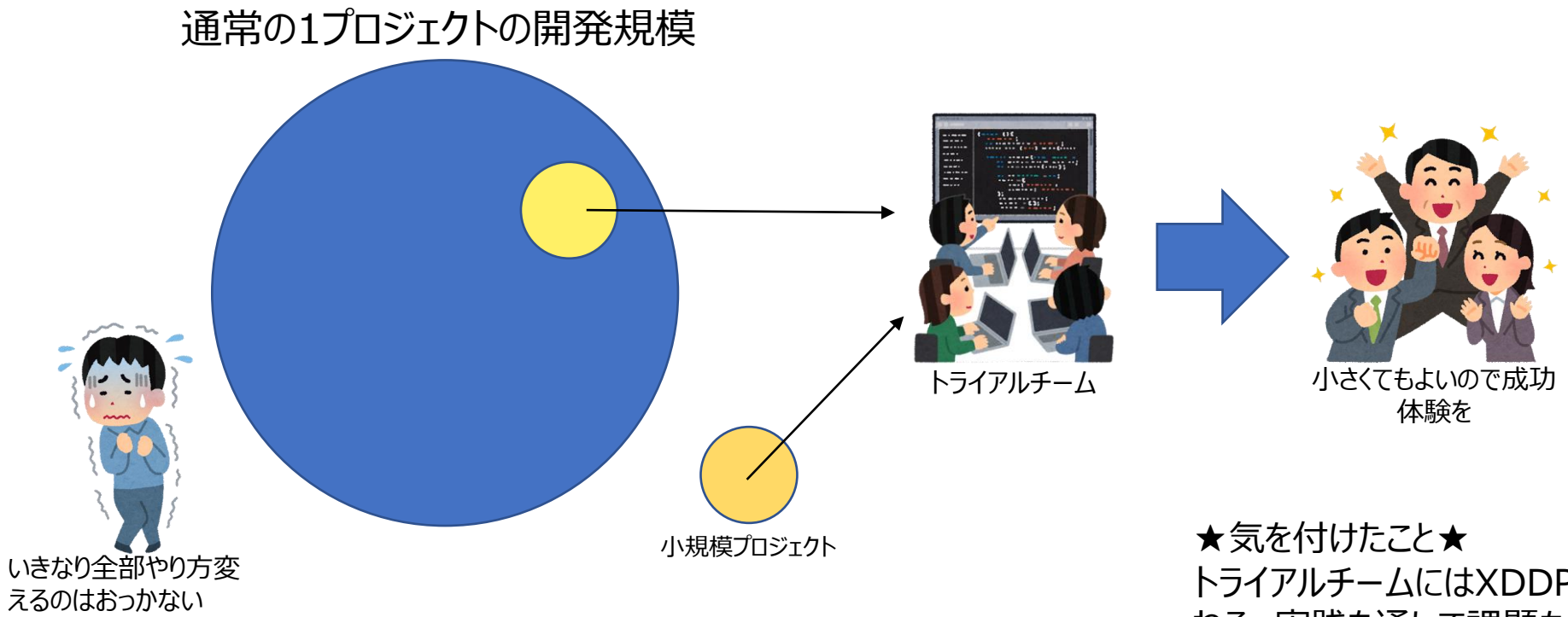
- コアメンバーの研修参加
  ⇒意識高い系のメンバーに研修を受講してもらい、XDDP普及の推進役になってもらう。
- 全開発者へのXDDPゲーム(※)の実施
  ⇒「なんとなく良さそう」という実感を持ってもらう。

※SPI Japan 2018 XDDPゲーム ～派生開発手法「XDDP」の原理をゲーム感覚で理解できるワークショップ～（八木 将計氏(株式会社日立製作所)）
を参考にさせていただきました。

★気を付けたこと★
数百人いる開発者全員にXDDPを完璧に理解してもらうのは現実的ではない。実践しながら感覚をつかんでもらえるよう「とりあえずやってみるか」と思ってもらえるように推進した。

導入準備：訓練

小規模プロジェクト、プロダクトの一部を抜き取りトライアルを実施。少人数でXDDPを実践し効果を報告。

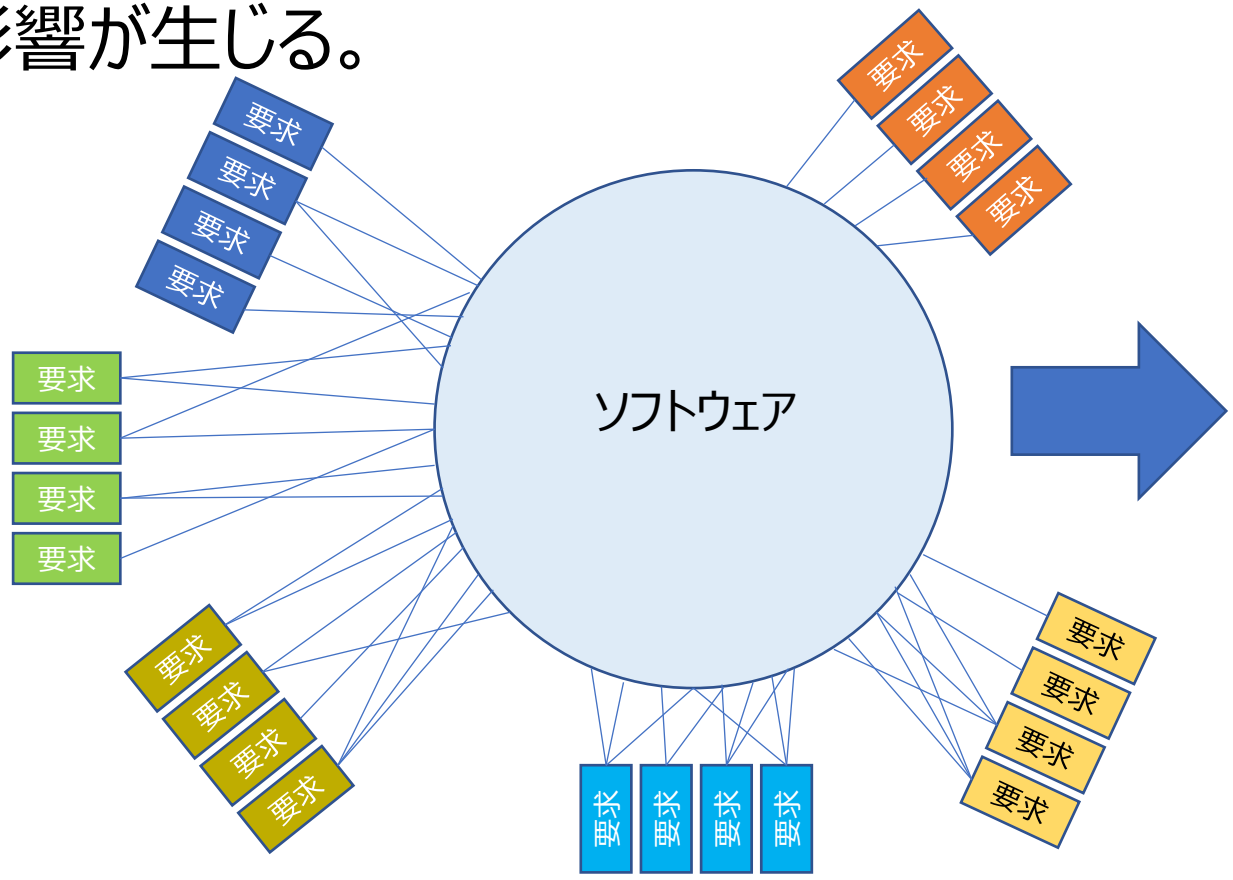


★気を付けたこと★
トライアルチームにはXDDP推進者自身が直接かかわる。実践を通して課題も抽出し本運用で対策を講じる。

フィーチャーモデルの活用

差分抽出の単位

組み込みソフトウェアを開発するうえでの差分要求はユーザー要望だけでなく、メカ制御、電気制御、アルゴリズムの変更など様々な要因から発生する。毎回異なる箇所に影響が生じる。



全方位からくる変更要求すべてをバラバラに扱うとコンフリクトが起きやすいし、無駄が多くなる。
↓
ある程度の単位でまとめて分業したい。
↓
単位が毎回変わると混乱のもとになってしまう。
↓
できれば不変的な単位が欲しい。
↓
変更するための開発項目単位を変更したくないという矛盾。

フィーチャーモデルの作成①

XDDPにおける差分抽出の不変的な基準として「フィーチャーモデル」を活用する。

【フィーチャーモデル】とは

製品ラインに含まれる製品セットを構成する特徴を表現したもの。プロダクトラインエンジニアリングにおいて、製品ラインに含まれる製品のバリエーションを表現するために使用する。

チョットナニイッテルカ
ワカリマセーン

フィーチャーモデルの作成②

「フィーチャーモデル」は掴みどころがなく、人によって解釈も分かれてしまいがち。

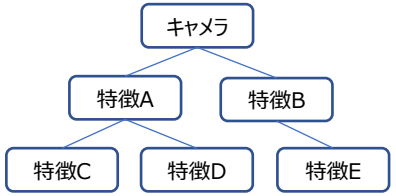
大人数で検討すると。

意見が合わず決められない。

何を”特徴”と捉えるのかは、製品開発にどのような立ち位置で関わっているかによって感覚が大きく異なる。意見が合わないのは当然。人数が増えれば増えるほどすり合わせの時間は増加していく。しかも、「すり合わせ」自体が目的になってしまい、フィーチャーモデルを構築する目的を見失ってしまう。

あくまで立ち上げ時の話。

極力少ない人数で検討する。ただし一人で決めるのはだめ。最低二人以上。

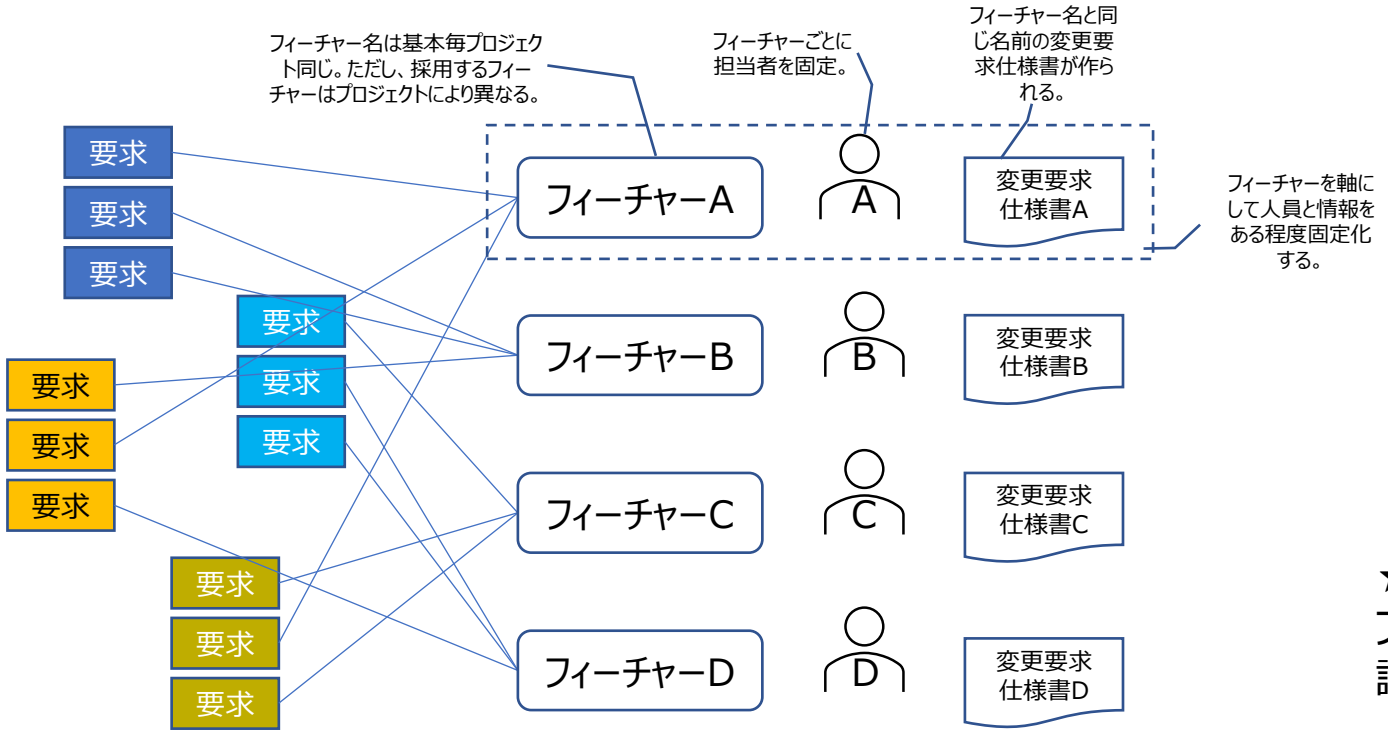


すり合わせの時間が最小限で済み早く決まる。

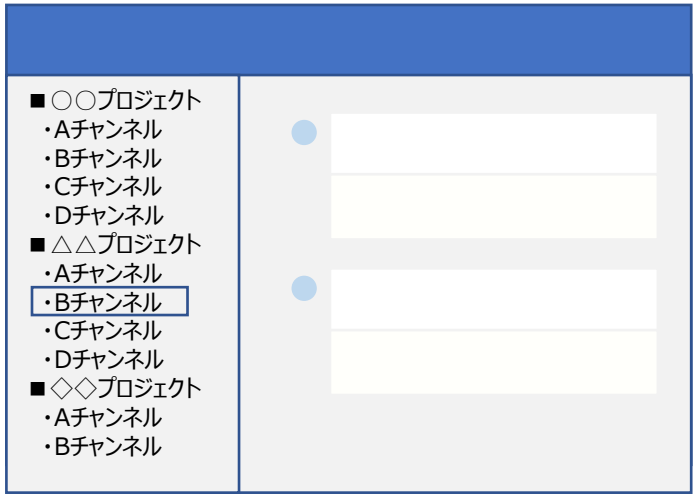
★気を付けたこと★
製品に対する最も広い知見を持ったメンバーをフィーチャーモデルの検討に引き込む。いきなり完璧はもとめず最初は抽象度の高いレベルにとどめておく。一度運用に乗せたあと、実務上の不都合を広く意見収集しブラッシュアップしていく。

フィーチャーモデルの利用方法

変更要求をフィーチャーに関連付けし、フィーチャー単位で変更要求仕様書を作成。コミュニケーションのルートもフィーチャーを基準にする。



チャットツールのチャンネル構成をフィーチャー単位にする。



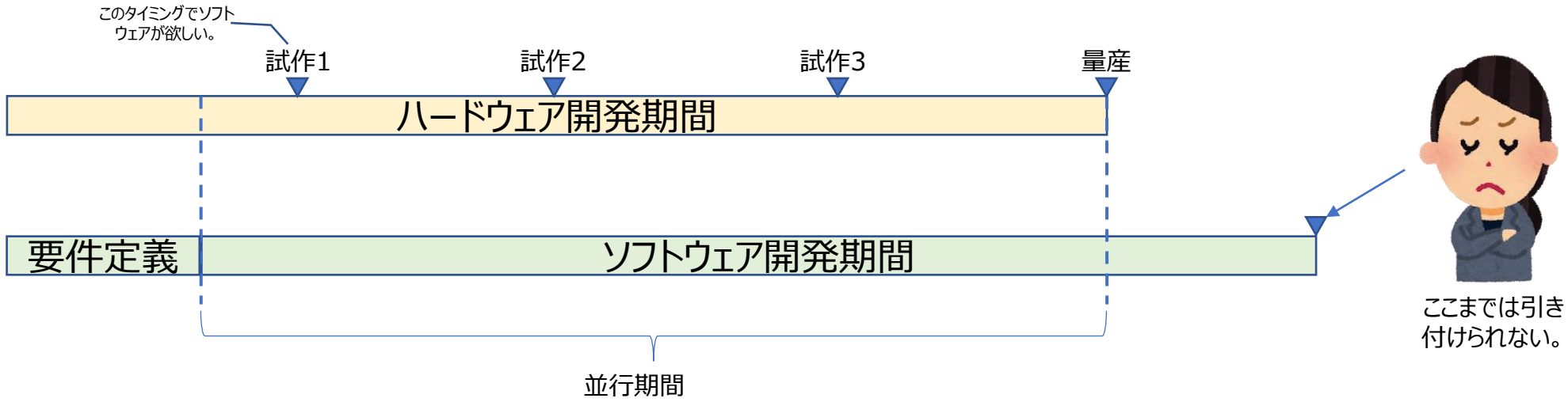
★気を付けたこと★
フィーチャー自体には意味を定義せず、あくまで「標識」として使用する。同じフィーチャーに対する解釈が人によって異なることを許容し、実務担当者がやりやすくなるようフィーチャーに情報が紐づけば良しとする。

本格実践

プロジェクト全体でのXDDP実践①

XDDPの基本は「全ての変更設計が完了してから一気にソースコードを変更する」

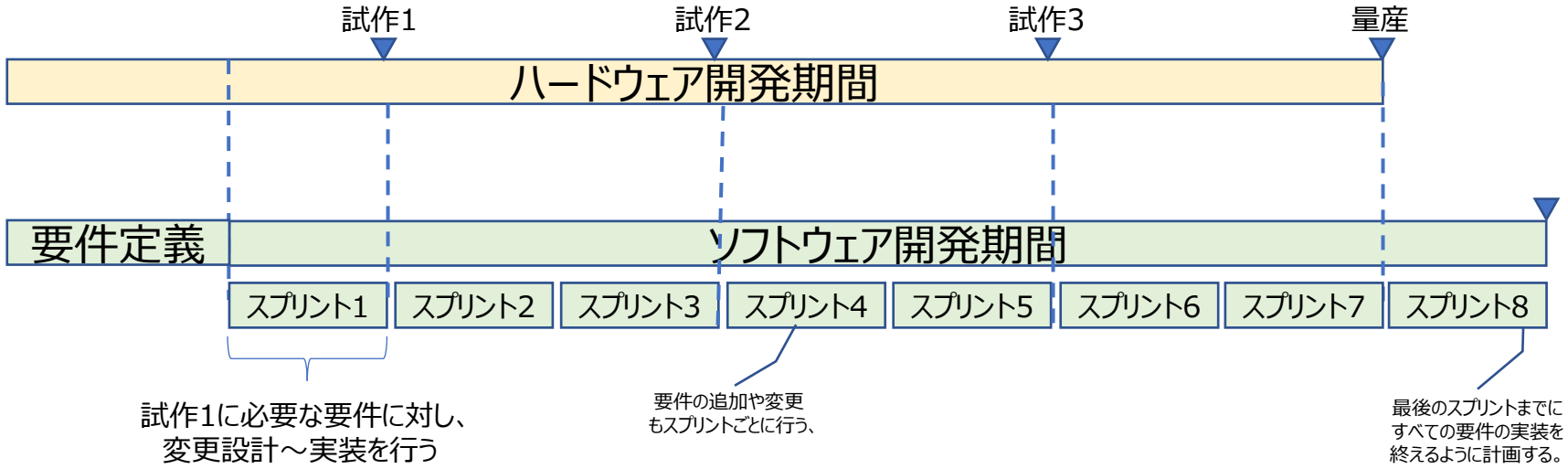
⇒コンカレント開発では開発途中にハードウェア部門からの要望である程度動くソフトウェアのリリースが求められることがある。また、実機レベルでの使用感を試してから仕様を確定したい場合もある。



プロジェクト全体でのXDDP実践②

全開発期間を分割して段階的にソフトウェアをリリースする。

⇒分割した1開発期間の中でXDDPのプロセスを1回実施するイメージ。1回の期間をスクラム開発になぞらえて「スプリント」と呼ぶ。（厳密な意味でのスクラム開発ではない）



プロジェクト全体でのXDDP実践③

変更要求仕様書にどのスプリントで対応するか計画を記入して管理する。

■フィーチャーAの変更要求仕様書

項目		内容	スプリント
要求		おにぎりをつくる	1
	理由	腹が減っては戦はできないため	
要求		豚汁をつくる	2
	理由	おにぎりだけでは物足りないので	
要求		漬物を切る	3
	理由	アクセントが欲しいので	

全体の開発期間の中でどの時期までに必要な要件なのかを加味して対応スプリントを決める。変更要求仕様書に対応スプリントを明記することで、フィーチャーの関係メンバーが同じタイミングで同じ目標を目指しやすくなる。

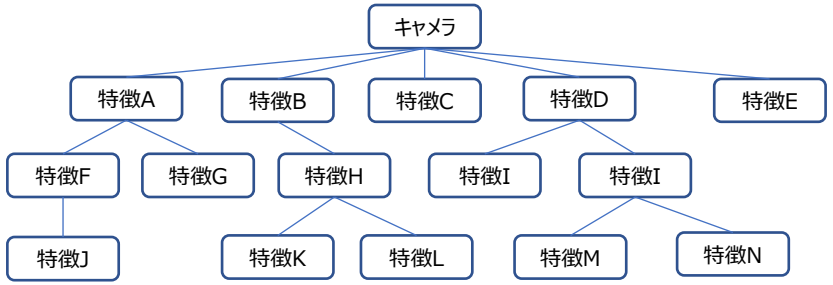
最後のスプリントですべてが揃えばよい。スプリントごとに作業を集中でき、要件の変更や追加にも対応しやすくなる。

プロジェクト全体でのフィーチャーモデルの役割①

プロジェクトの最初期に対象製品が採用するフィーチャーを選定する。

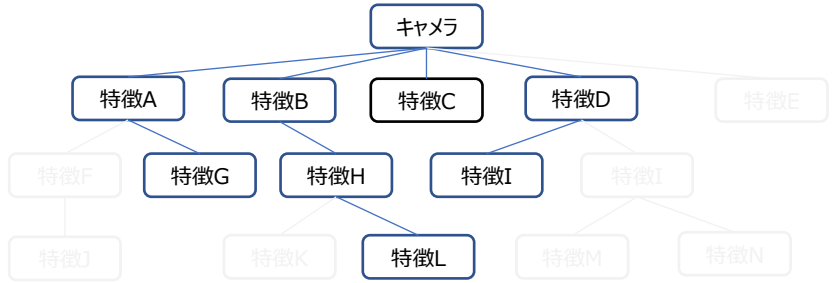
⇒対象製品のアウトラインが可視化され、変更要求仕様書の作成単位が決まる。

フィーチャーモデル（全体）

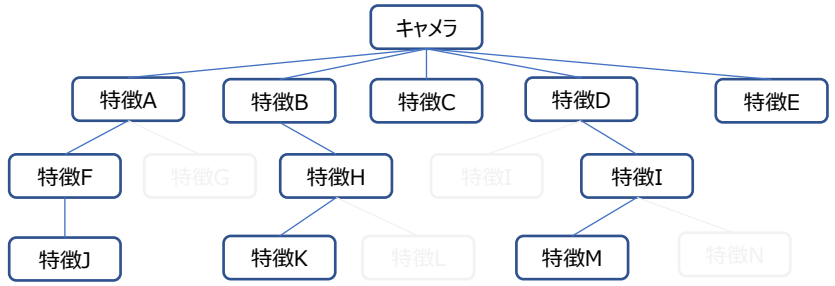


採用するフィーチャーを選定

フィーチャーモデル（製品A）



フィーチャーモデル（製品B）

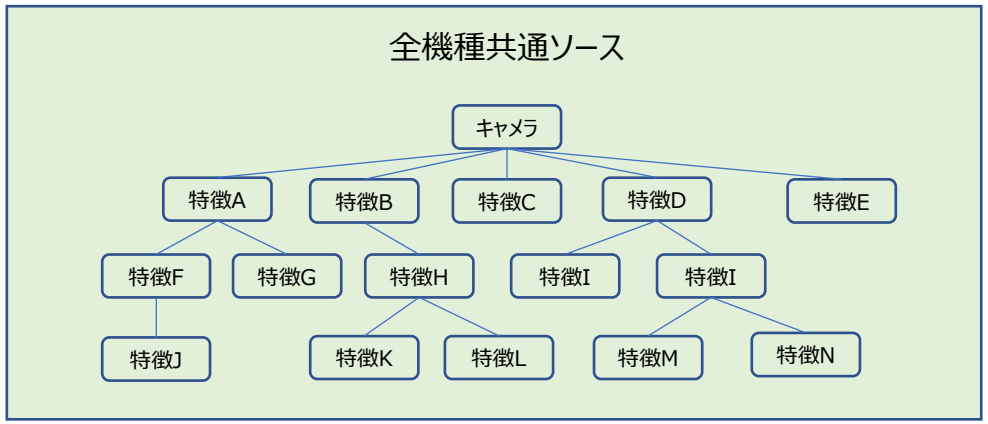


フィーチャーの選定作業はマトリクスに変換して行う。

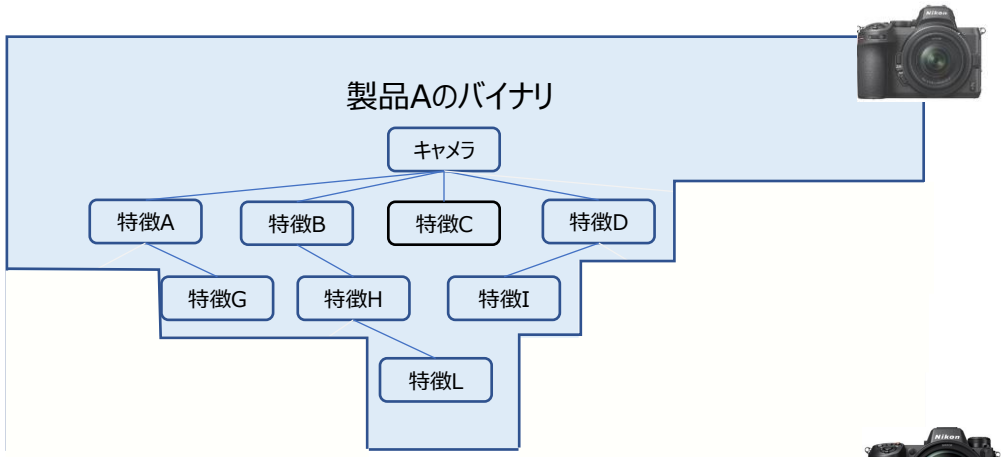
フィーチャー名			製品		
			製品A	製品B	製品C
特徴A			○	○	○
	特徴D		○	○	○
	特徴D			○	
			○	○	○
	特徴E	特徴E-1	○		
		特徴E-2	○	○	
	特徴F		○	○	○
	特徴G			○	
	特徴H				
特徴B			○	○	○
			○	○	○
	特徴I	特徴I-1	○	○	
		特徴I-2	○		○
	特徴J			○	○
	特徴K			○	
	特徴L				
	特徴M			○	○
特徴C			○	○	○
	特徴N		○	○	○
		特徴N-1	○	○	○
		特徴N-2	○	○	○
		特徴N-2		○	○
		特徴N-2		○	
		特徴N-2			
		特徴N-2			
		特徴N-3	○	○	○
		特徴N-4		○	○
		特徴N-4		○	○
		特徴N-5	○		
		特徴N-5			
		特徴N-6	○	○	○
		特徴N-7	○	○	○
		特徴N-7	○	○	○
		特徴N-8			
		特徴N-9			

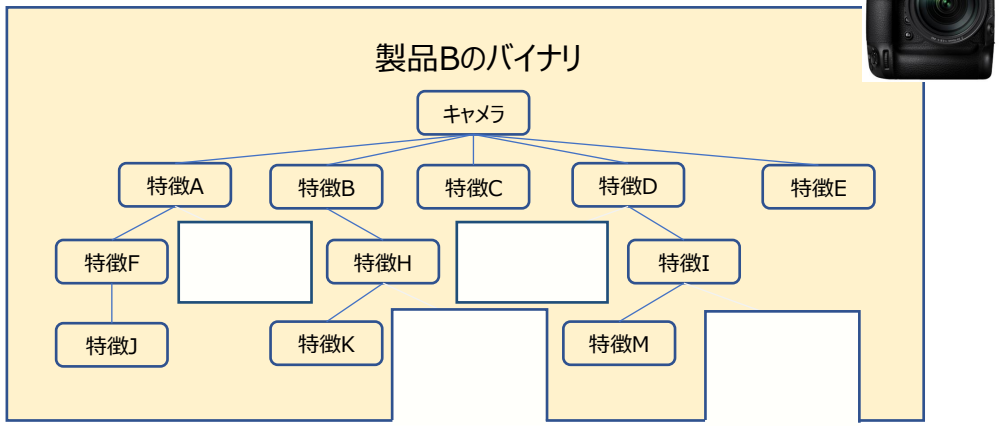
プロジェクト全体でのフィーチャーモデルの役割②

ソースコードのリポジトリを1本化し、フィーチャーをコンフィギュレーションにして製品差分にあたる処理を切り分ける。



フィーチャーの有無に従ってコンフィグを切り替えビルド





進行中のプロジェクトの前に別のプロジェクトが割り込むような突発的な計画変更があると、コンフィグの定義や共通処理のFBが必要になってしまうなど、運営面での課題も多く、計画通りに運用するのは難しい。さらなる改良を模索中。

改善効果

- XDDPを導入したことによって、開発を進めるうえでの成果物が明確になった。これにより、成果物の完成数をカウントすることで進捗状況を機械的に把握することができるようになった。
- 変更要求仕様書をフィーチャー単位で作成することにより、開発項目の全体像を俯瞰して可視化することができるようになった。また、フィーチャーモデル＞ 変更要求仕様書＞ 変更設計書、という流れで抽象度の高い成果物から、具体化した成果物にストレートにブレイクダウンして成果物を追うことができるようになり、状況に応じて必要な情報を探しやすくなった。
- 分割リリースするスタイルにしたことにより、要件単位での開発順序や要員計画を見据えた計画を立てやすくなった。
- リポジトリの1本化により複数プロジェクト間で共通の処理をまとめて開発しやすくなった。（開発成果物のコア資産化）

まとめ・所感

XDDPとプロダクトラインエンジニアリングは相性がよい。アジャイルの思想と完全に相反するものではない。

- フィーチャーモデルは成果物だけでなく、組織やコミュニケーションの軸としても有効に使うことができる。
- フィーチャーモデルを使うことで、小さな単位で自律分散的に開発を進行することができる。
- XDDPを単一の手法としてとらえるのではなく、いくつかの手法やパラダイムをブレンドすることで、より実践的な開発スタイルを構築することができる。また、アジャイル開発を実践するための土壌にすることができるかもしれない。
- よりアジャイルな開発を実践するためには、チームビルディングの観点からのアプローチも必要。

参考文献・出典

- 【書籍】「派生開発」を成功させるプロセス改善の技術と極意（著者：清水 吉男）
- 【書籍】[改訂第2版][入門+実践]要求を仕様化する技術・表現する技術 ～仕様が書けていますか？（著者：清水 吉男）
- 【書籍】プロセスを自在に設計するーPFDを使いこなそうー（著者：梶本 和博、派生開発推進協議会 T21研究会　編集：八木 将計、八木 香織）
- USDMを簡単に理解してもらうための小冊子（AFFORD：T2研究会活動成果）
- 【セミナー】日科技連 実践！派生開発を成功させるXDDPセミナー （講師：古畑 慶次）
- 【セミナー】XDDPゲーム ～派生開発手法「XDDP」の原理をゲーム感覚で理解できるワークショップ～（講師：八木 将計氏）（SPI Japan 2018）

ご清聴ありがとうございました。


Nikon