アジャイルになるために

TIS株式会社
テクノロジー&イノベーション本部

テクノロジー&エンジニアリングセンター

全体のコンテンツ構成

1. なぜアジャイルを学ぶのか
   - 価値ある機能をどう提供するか
   - 受託開発とサービス開発の違い
   - ウォーターフォールとアジャイルの違い
2. アジャイルとは何か
   - アジャイルソフトウェア開発宣言とその背後にある原則
3. スクラムを知る
   - スクラムとは何か
   - スクラムの３・５・３
   - スクラムの品質

1. なぜアジャイルを学ぶのか

アジェンダ

1. なぜアジャイルを学ぶのか
   - 価値ある機能をどう提供するか
   - 受託開発とサービス開発の違い
   - ウォーターフォールとアジャイルの違い

学習目標

以下を理解できる

- 受託開発とサービス開発の価値観の違い
- 価値観の違いから生まれる問題
- 我々が変えていかなければいけない新たな意識

価値ある機能をどう提供するか

理想的でないアジャイルの現場でよく見られる光景

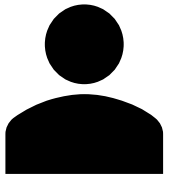
開発者

プロダクトオーナーがバックログを
整備してくれないので開発が進みません

書かれたプロダクトバックログの内容の意味が
分からないので**詳細に記述して**ください

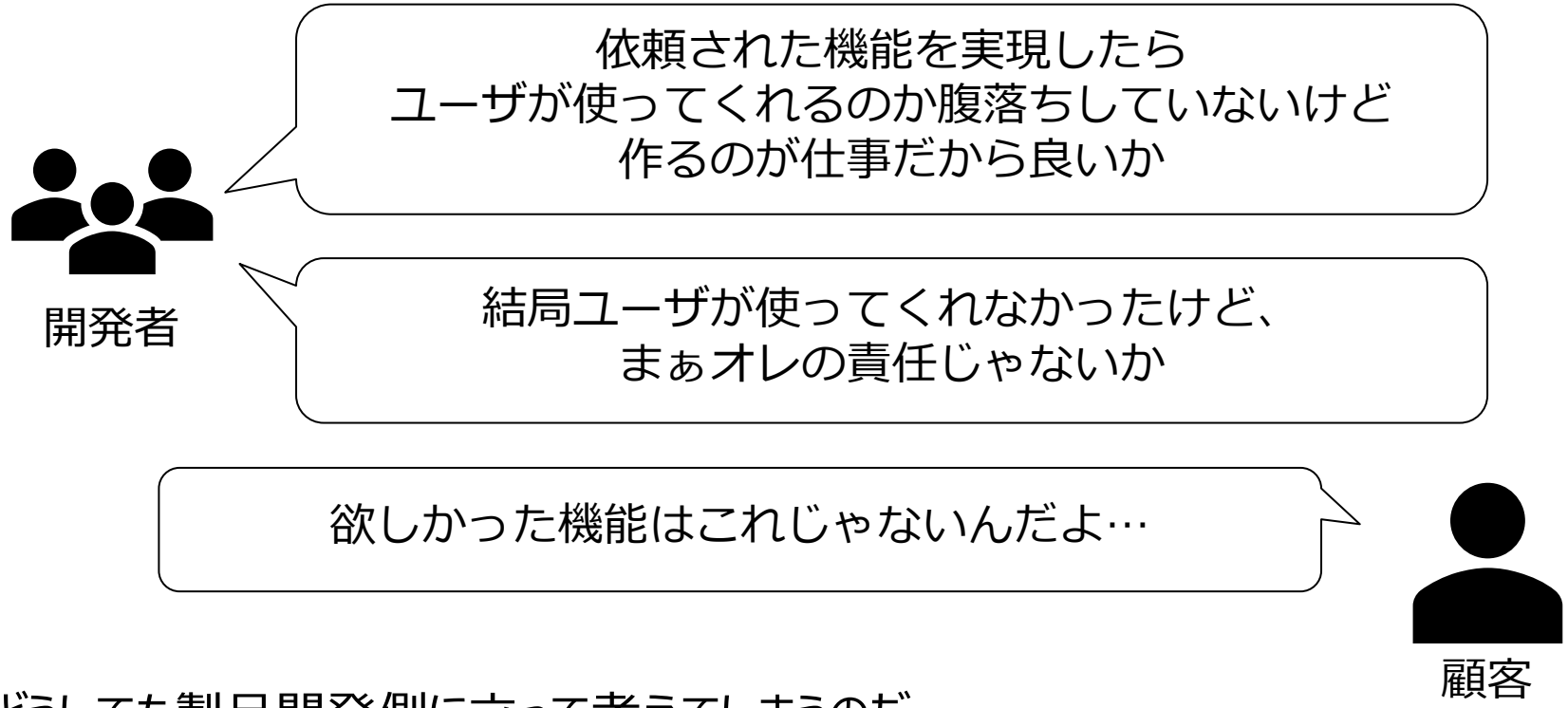
開発者が**言ったことしかやってくれない**…

システム的なところが分からないから
もっと**寄り添ってほしいん**だけど…



プロダクト
オーナー

理想的でないアジャイルの現場でよく見られる光景

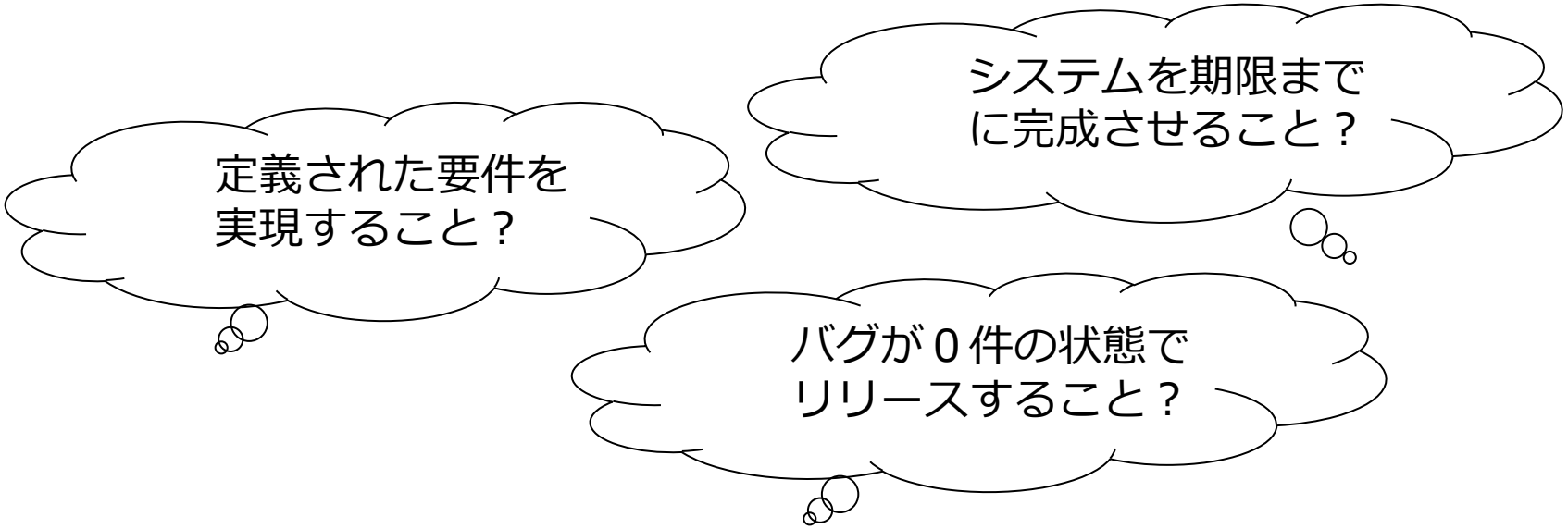


どうしても製品開発側に立って考えてしまうのだ。
マネジメントや販売はパートナーや上司の担当で、私のような人間は技術や運用といった仕事に注力する。そして、**必死に頑張った製品が市場で悲惨な結果に終わるという経験を繰り返してきた。**

エリック・リース
「リーン スタートアップ」
（日経*BP*、*2012*）

- **アジャイル開発を使って、利用者にとって価値あるもの**をどう提供するか？

- 提供しようとしている価値を**チームの中で共有**できていますか？



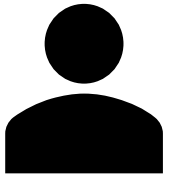
理想的なアジャイルの現場で見られる光景

開発者

バックログに書いてある内容について、機能実装以外で対処できないでしょうか？

たしかに、これは既存の機能でも実現できそうですね

こういう使い方もありますよと紹介する方針にしましょう



プロダクトオーナー

理想的なアジャイルの現場で見られる光景

開発者

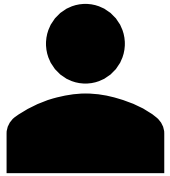
前回のアンケートでこういうニーズが見られた。
機能追加するといいかもしれない。

この機能をユーザにどう使ってもらうか？

使ってもらってなにを成し遂げられるか？

リリース後・・・

欲しかった機能はこれだ！

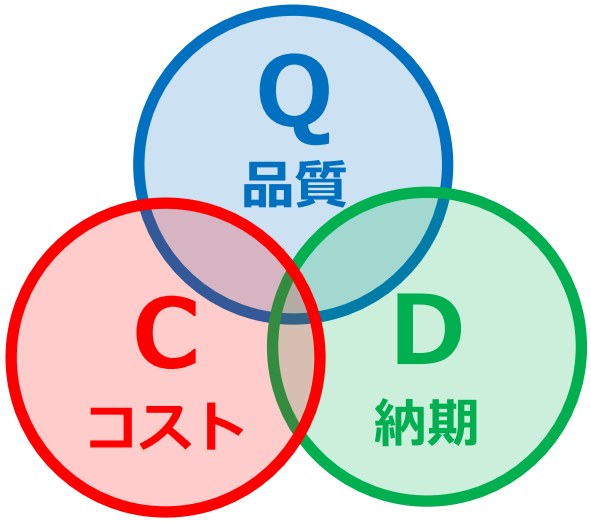


ユーザ

受託開発とサービス開発の価値の違い

顧客が要求するシステムを
如何に正確に・安く・早く完成させるか

- **<u>完成させること</u>**が価値の主体
- 高いレベルでのQCDで訴求



完成後のサービスが
ユーザに受け入れられないことに対する責任はない

※エンドユーザの志向の分析、マーケティングは顧客の役割

エンドユーザが欲する価値を
如何に発見し、提供するか

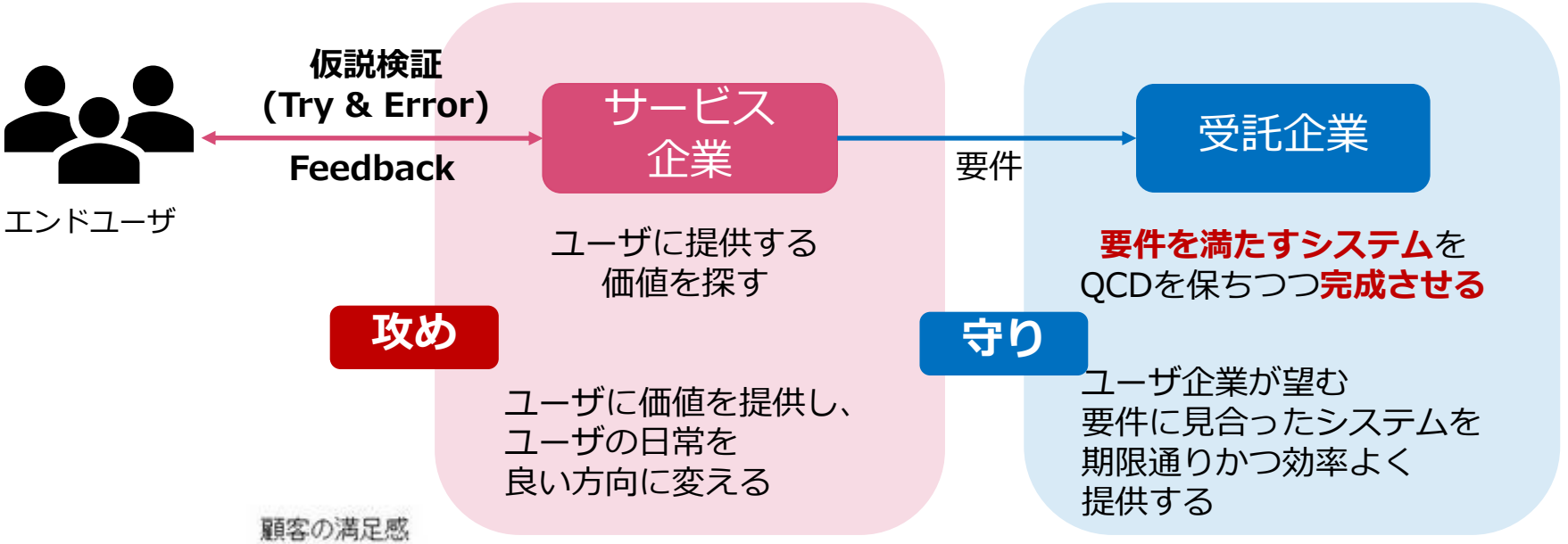
- エンドユーザが望むものを探す
  - 仮説を立てる
- 仮説をサービスに組み込み提供する
  - ユーザが望む価値を提供できているか検証する

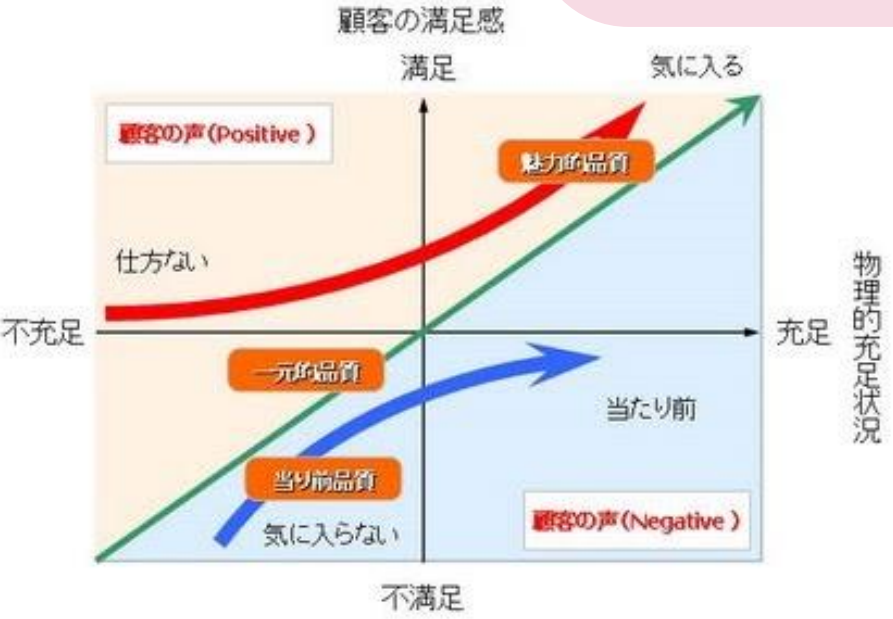
仮説 ⇄ 検証

ユーザが望むものを提供できない場合

如何にQCDを保って完成させても
そのシステムに価値はない

受託開発と自社サービス開発の違い





“ウォーターフォール型は
開発が失敗しないための手法であり、
アジャイル開発は
ビジネスが成功するための手法であると
言えなくもない”

非ウォーターフォール型(アジャイル)開発の動向と課題
-*IPA/SEC*における*4*年間の
調査・検討から明らかになったこと-

ユーザーに価値を問う難しさ

「顧客の意見に耳を傾けよ」というスローガンが良く使われるが、このアドバイスは**いつも正しいとはかぎらない**ようだ。むしろ顧客は、メーカーを持続的イノベーションに向かわせ、破壊的イノベーションのリーダーシップを失わせ、**率直に言えば誤った方向に導くことがある**。

クレイトン・クリステンセン
「イノベーションのジレンマ」
（翔泳社、*2011*）

顧客から彼らの望みを聞くことはできなかった。（中略）
ただ、我々が製品を改良しようと悪戦苦闘している間、**行動によって、あるいは行動しないことによって真実を伝えてくれていた**のだ。

エリック・リース
「リーン スタートアップ」
（日経*BP*、*2012*）

ユーザーは自分自身が
何を望んでいるか分からない

- 価値を判断するのは**「市場」**
  - 市場の価値やニーズは目まぐるしく変化
  - 複雑・不確実なニーズを持つ市場
  - 市場に価値を問わないまま
    重厚な計画を立てる意味があるか

VUCA

- Volatility
- Uncertainty
- Complexity
- Ambiguity

ビジネスでは「**スピード**」が求められている

ビジネスでは「スピード」が求められている

- 企業が求めるスピード

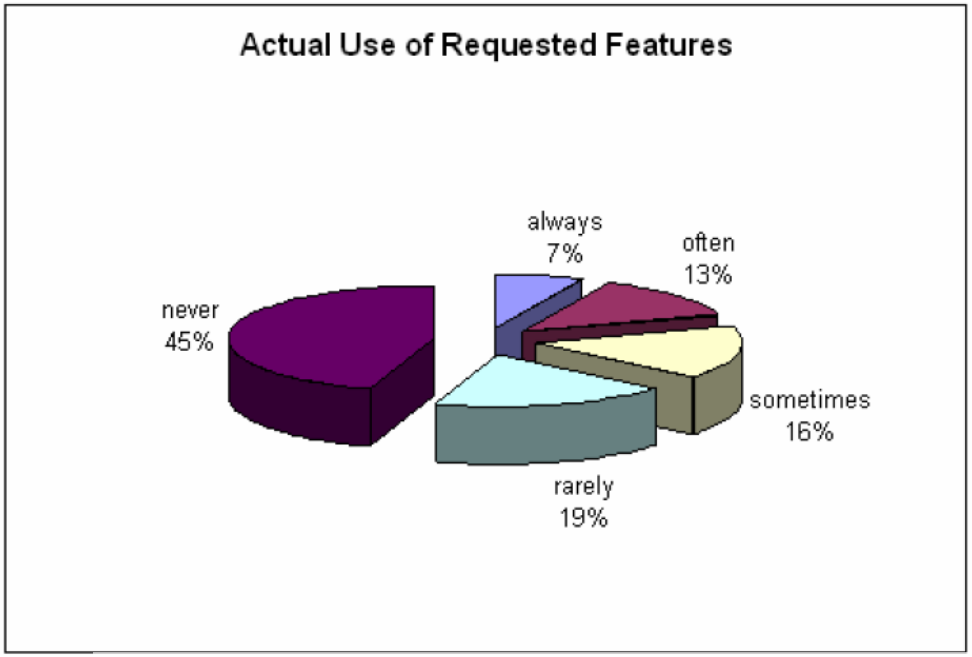
・ROIの算出
・費用の算出
・効果の明確化
・リスクの明確化

・自動化
・標準化
・効率化
・ルール定義

本当に価値ある機能とは

- 全く使わない機能　　45%
- ほとんど使わない機能　19%

せっかく作っても、
ほぼ使われない機能が半分以上



Qualid, Ktaka; Ghislain, Levesque;
Agile development: Issues and avenues requiring a substantial enhancement of the business perspective in large projects.
ACM International Conference Proceeding Series, 2009, p.59-66

- **使われない機能の開発をスピードアップさせることに意味はあるのか？**
- **誰も欲しがらないものを作っているのなら、スケジュールや予算を守ることに意味があるのか？**

ビジネスでは「スピード」が求められている

- もう一つのスピード

ビジネスでは「スピード」が求められている

- 結果を知るスピードによって得られるもの

- 仮説を実ビジネスで検証
- 判断根拠は
  具体的・リアルな結果
- 要望/要求の明確化

- 要望の即時反映による
  魅力品質の向上
- 必要な機能に集中して開発
- 結果的に不要な機能を
  開発しないことによる
  工数削減

スタートアップが本質的に持つ不確実性

- 顧客が誰なのか
- 顧客が何に価値を見出すのか

検証による学び
(Validated Learning)

顧客の望みを学ぶためにどうしても必要なものだけが必要
それ以外の努力はなくて良い

- 顧客が使ってみることさえしない機能について
  どうすべきか、どれを優先すべきか検討する
- 使われないシステムのアーキテクチャを美しくする

“顧客はこう望んでいるはずという自分の考えを正当化するのは簡単だ。的外れのことを学ぶのも簡単だ。だから、検証による学びを得るためには、現実の顧客から集めた実測データを基礎とする必要がある”

エリック・リース
「リーン スタートアップ」
（日経*BP*、*2012*）

MVP: Minimum Viable Products

- Viable: 形容詞
  - 存立できる、実行可能な、生存できる
- それ単体で存立できる最小限のプロダクト

検証による学び
(Validated Learning)

製品やサービスが本当に価値を提供できるか

アーリーアダプター

結果を知る スピード → **判断・決定する スピード** → **形にする スピード**

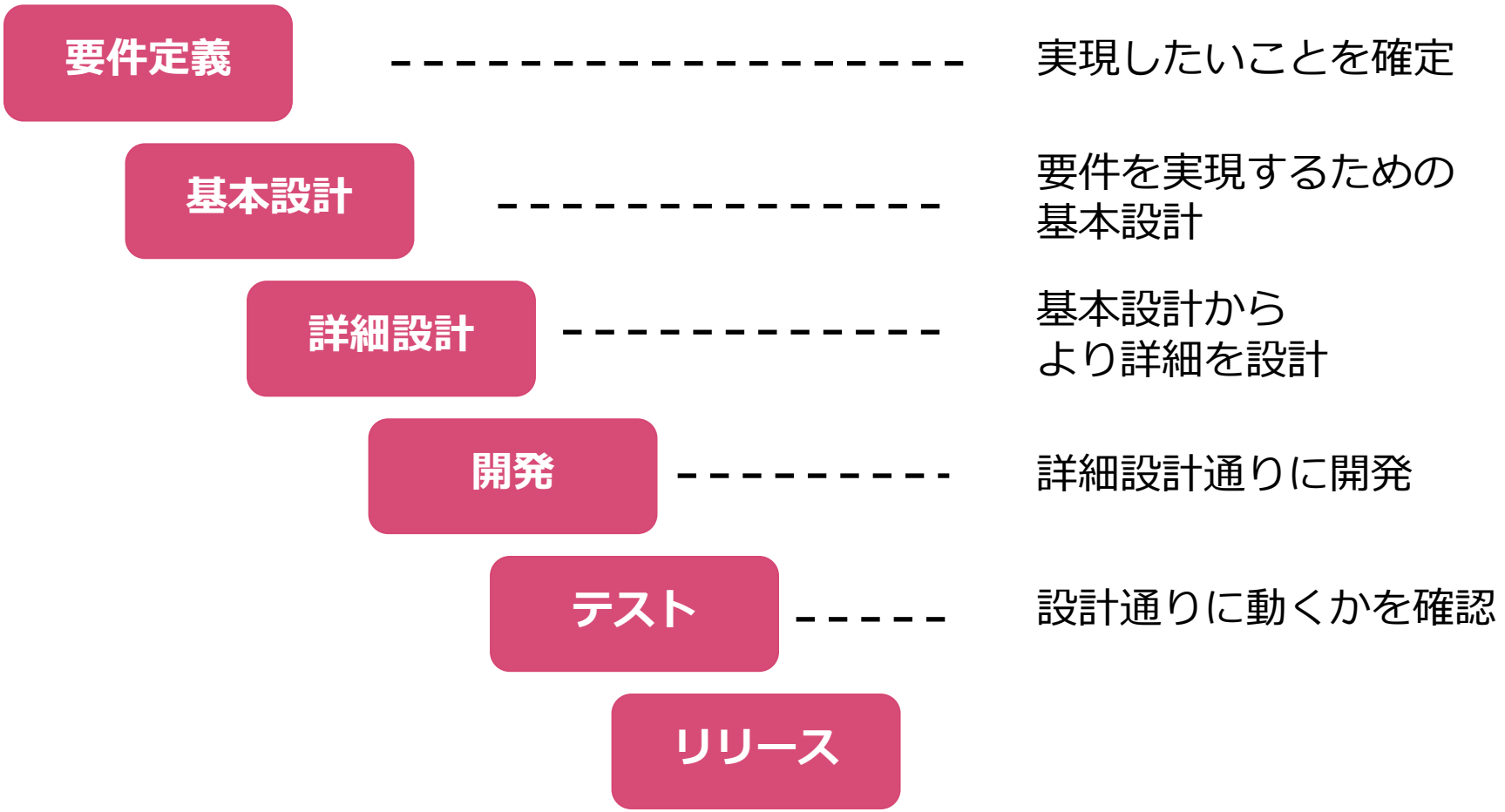
- 構築・計測・学習のフィードバックループを最も早く回せる
- 最小限の労力で回せる

短い期間で仮説・検証を繰り返してサービスを育てていく

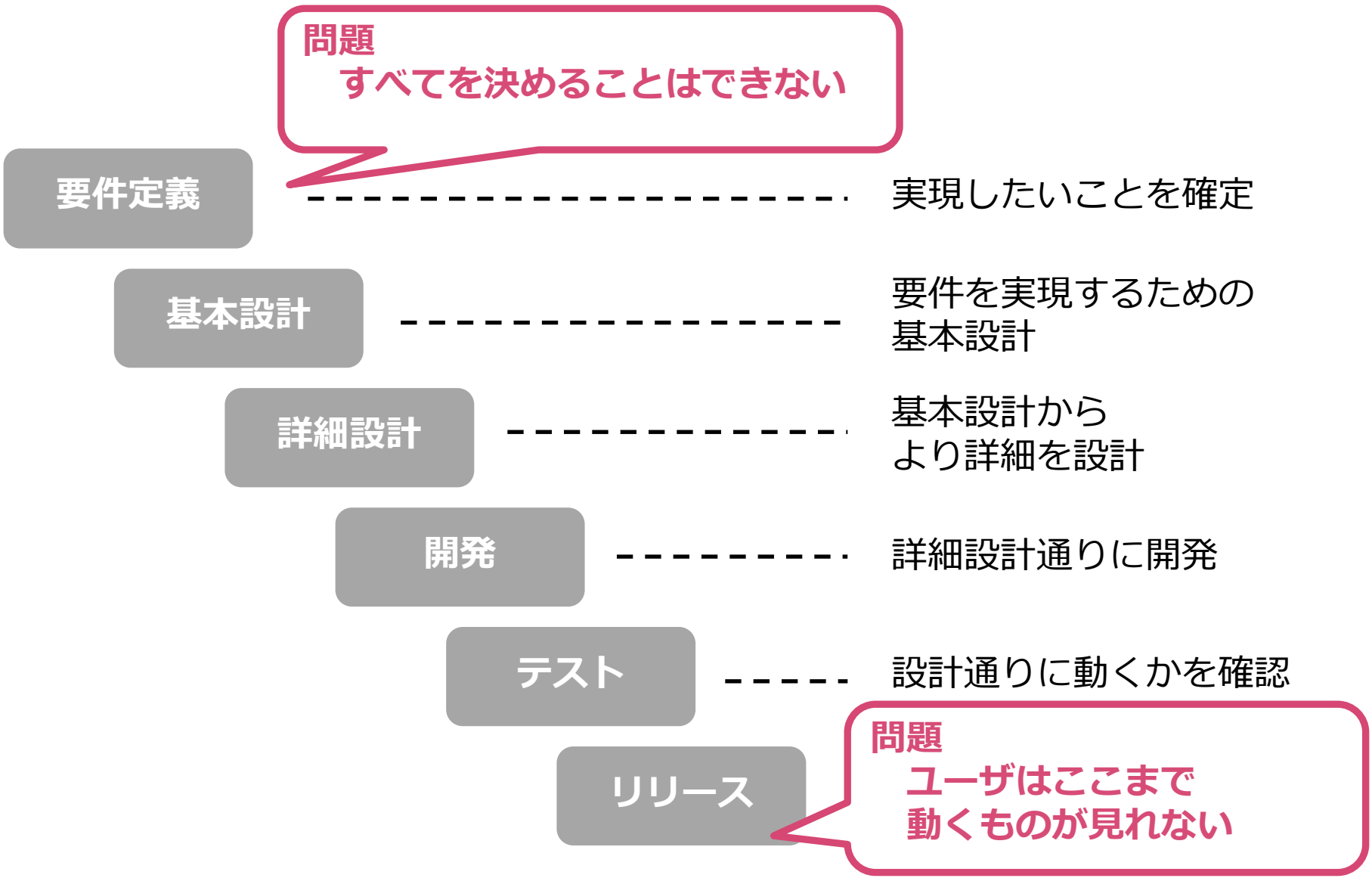
- 素早くリリースしフィードバックを得ることで、
  「結果を知るスピード」を高める
- 実際のユーザの反応を元にすることで
  「判断・決定するスピード」を高める
- 本当に必要な機能だけを作ることで
  「形にするスピード」を高める

ウォーターフォールとアジャイルの違い

前工程の正しさが前提



サービス開発における問題



サービス開発における問題

問題
すべてを決めることはできない

要件定義 ---------- 実現したいことを確定

基本設計 ---------- 要件を実現するための基本設計

詳細設計 ---------- 基本設計からより詳細を設計

開発 ---------- 詳細設計通りに開発

誰も欲しがらないものを
作っているのではないか？

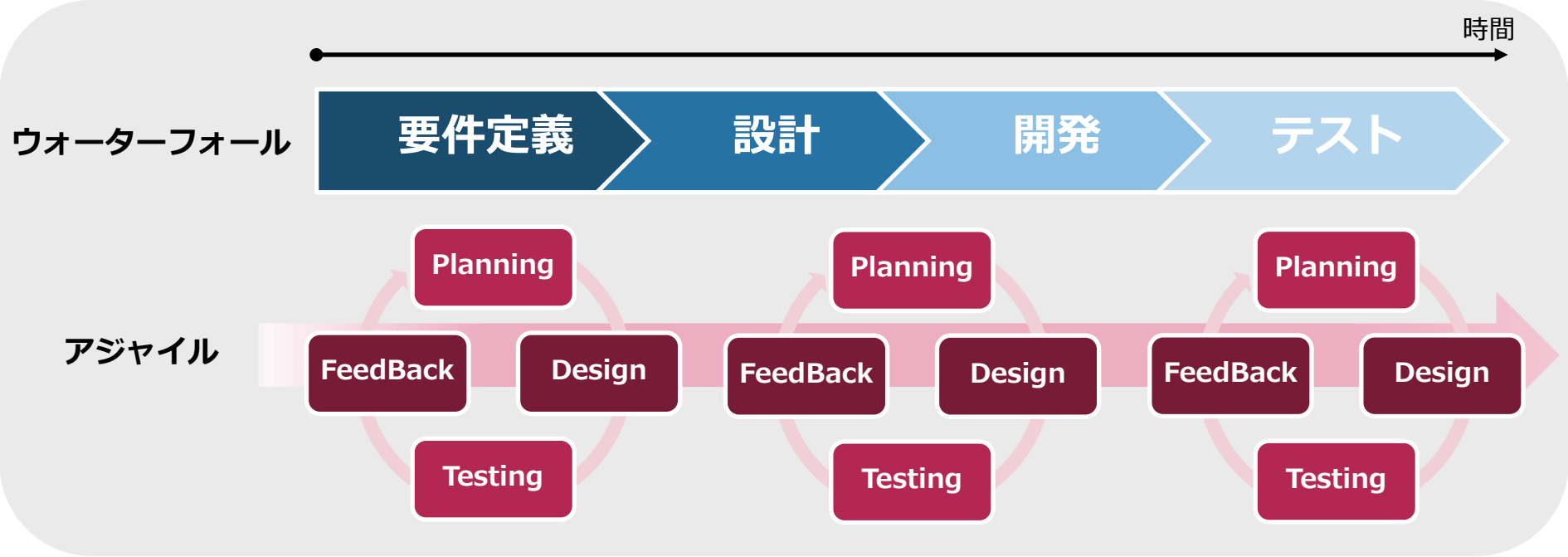
テスト ----- 設計通りに動くかを確認

リリース

問題
ユーザはここまで
動くものが見れない

従来の考え方を大きく変える

- **ウォーターフォール**
  - ドキュメントと前工程の確実性でプロセスを担保
- **アジャイル**
  - 動くシステムとコミュニケーションでプロセスを担保



職能指向

- 専門分野に応じてチームを組成

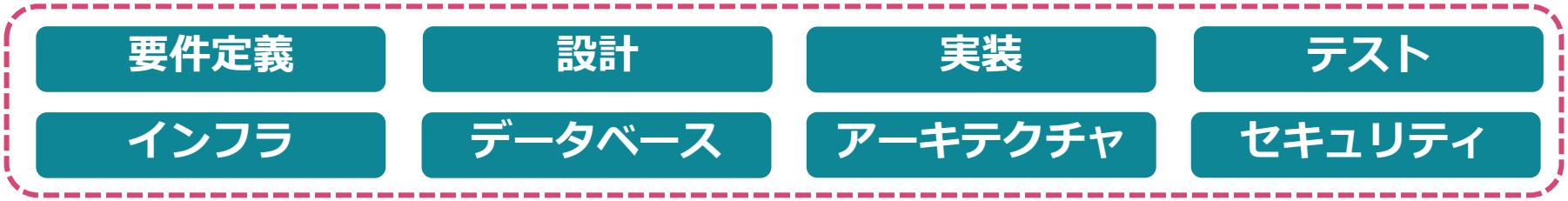
  - 反復と習熟がしやすく育成・分業に適する
  - 専門能力の一元管理によるキャリア形成

- 発生しやすい問題
  - チーム間の**作業の受け渡しの多発**
  - 作業者・チームの部分最適化（サイロ化）

→ **リードタイム悪化**
受け渡しのまずさ
手戻りの発生

市場志向

- 顧客ニーズに素早く向き合う
  - **独力で**仕事を進められるチームを組成



アジリティを持って顧客へ価値を提供できる

- 発生しやすい問題
  - 組織全体の中では機能重複が発生

2. アジャイルとは何か

アジェンダ

2. アジャイルとは何か
   - アジャイルソフトウェア開発宣言
   - アジャイル宣言の背後にある原則

学習目標

- 以下を理解できる
  - アジャイルソフトウェア開発宣言において、我々は何に価値を置くか
  - その価値の背後にどんな原則があるか
  - もっとも重要な原則はどれか

アジャイルソフトウェア開発宣言とその背後にある原則

アジャイルソフトウェア開発宣言

私たちは、ソフトウェア開発の実践
あるいは実践を手助けをする活動を通じて、
よりよい開発方法を見つけだそうとしている。
この活動を通して、私たちは以下の価値に至った。

プロセスやツールよりも**個人と対話**を、
包括的なドキュメントよりも**動くソフトウェア**を、
契約交渉よりも**顧客との協調**を、
計画に従うことよりも**変化への対応**を、

価値とする。すなわち、**左記のことがらに価値があることを認めながらも、私たちは右記のことがらにより価値をおく。**

Kent Beck
Mike Beedle
Arie van Bennekum
Alistair Cockburn
Ward Cunningham
Martin Fowler

James Grenning
Jim Highsmith
Andrew Hunt
Ron Jeffries
Jon Kern
Brian Marick

Robert C. Martin
Steve Mellor
Ken Schwaber
Jeff Sutherland
Dave Thomas

アジャイルソフトウェア開発宣言
(*http://agilemanifesto.org/iso/ja/manifesto.html*)
*2020*年8月*24*日*11*時に最新を取得

アジャイル宣言の背後にある原則

私たちは以下の原則に従う:

顧客満足を**最優先**し、**価値のあるソフトウェア**を**早く継続的に提供**します。

要求の変更は**たとえ開発の後期であっても歓迎**します。
変化を味方につけることによって、**お客様の競争力を引き上げます**。

動くソフトウェアを、2-3週間から2-3ヶ月という
できるだけ**短い時間間隔でリリース**します。

**ビジネス側の人と開発者は、プロジェクトを通して
日々一緒に働かなければなりません。**

意欲に満ちた人々を集めてプロジェクトを構成します。
環境と支援を与え仕事が無事終わるまで彼らを信頼します。

情報を伝えるもっとも効率的で効果的な方法は
フェイス・トゥ・フェイスで話をすることです。

アジャイル宣言の背後にある原則
(*https://agilemanifesto.org/iso/ja/principles.html*)
*2020年8月24日11時*に最新を取得

アジャイル宣言の背後にある原則(続き)

動くソフトウェアこそが進捗の最も重要な尺度です。

アジャイル・プロセスは持続可能な開発を促進します。
一定のペースを継続的に維持できるようにしなければなりません。

技術的卓越性と優れた設計に対する
不断の注意が機敏さを高めます。

シンプルさ（ムダなく作れる量を最大限にすること）が本質です。

最良のアーキテクチャ・要求・設計は、
自己組織的なチームから生み出されます。

チームがもっと効率を高めることができるかを**定期的に振り返り**、
それに基づいて**自分たちのやり方を最適に調整**します。

アジャイル宣言の背後にある原則
(*https://agilemanifesto.org/iso/ja/principles.html*)
2020年8月24日11時に最新を取得

もっとも守るべき原則って
どれだと思いますか？
理由とともに考えてみてください

ご自身のPCで「アジャイル宣言の背後にある原則」を
検索してみてください。

結論

原則を全て守ることが重要

ずるいよって思った方も多いと思います。
でも全部重要だということを理解してもらいたいです

ここではアジャイルの原則について触れました

この原則を意識することが実はとても重要で、
スクラムを実践する際の成否に関わってきます。

頭の片隅に置いておいてください。

3. スクラムを知る

アジェンダ

3. スクラムを知る
   - スクラムとは何か
   - スクラムの３・５・３
   - スクラムの品質

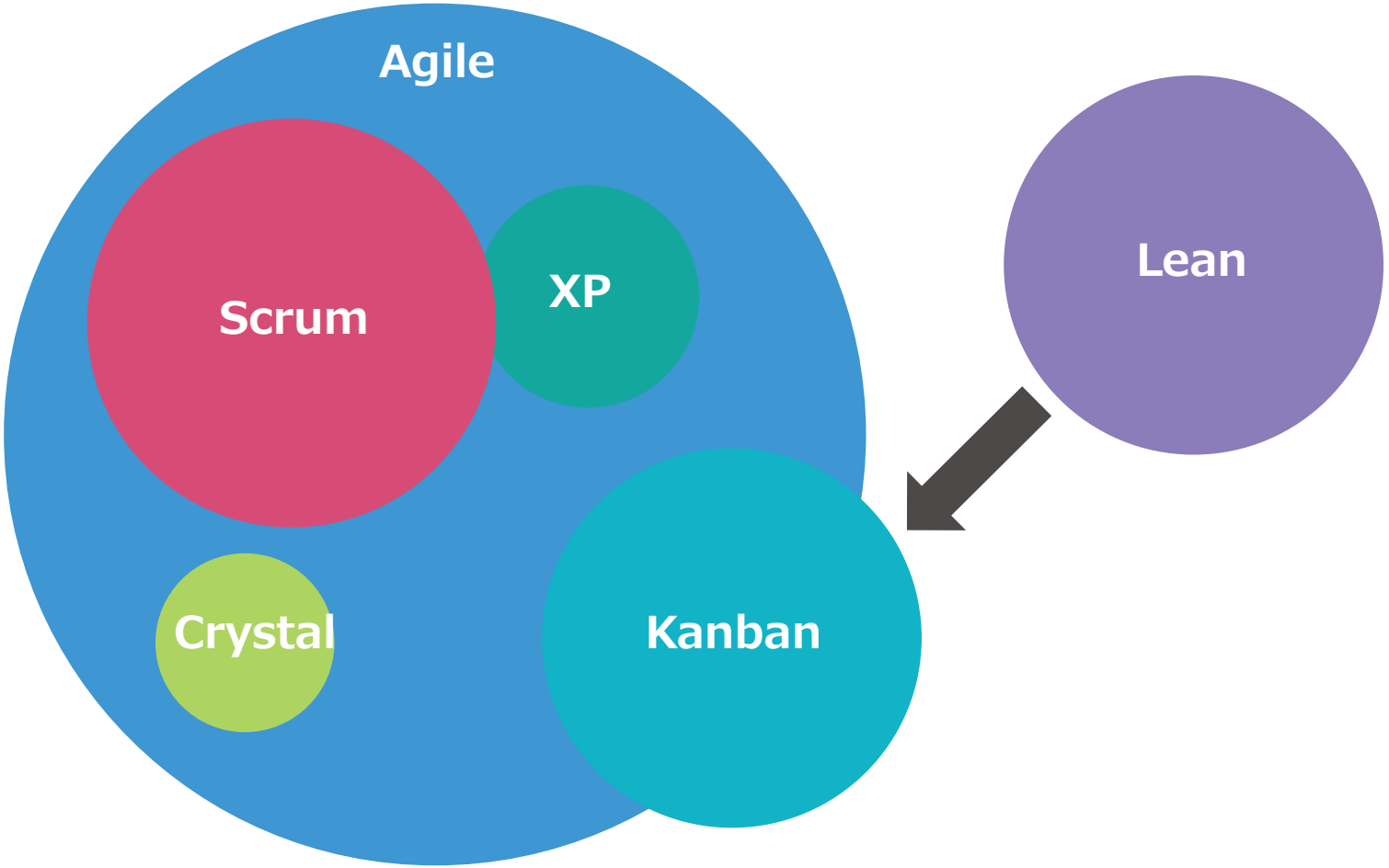
学習目標

- 以下を理解できる
  - アジャイルとスクラムの関係などの基礎知識
  - スクラムの構成要素 ３・５・３
  - スクラムにおける品質の考え方

スクラムとは何か

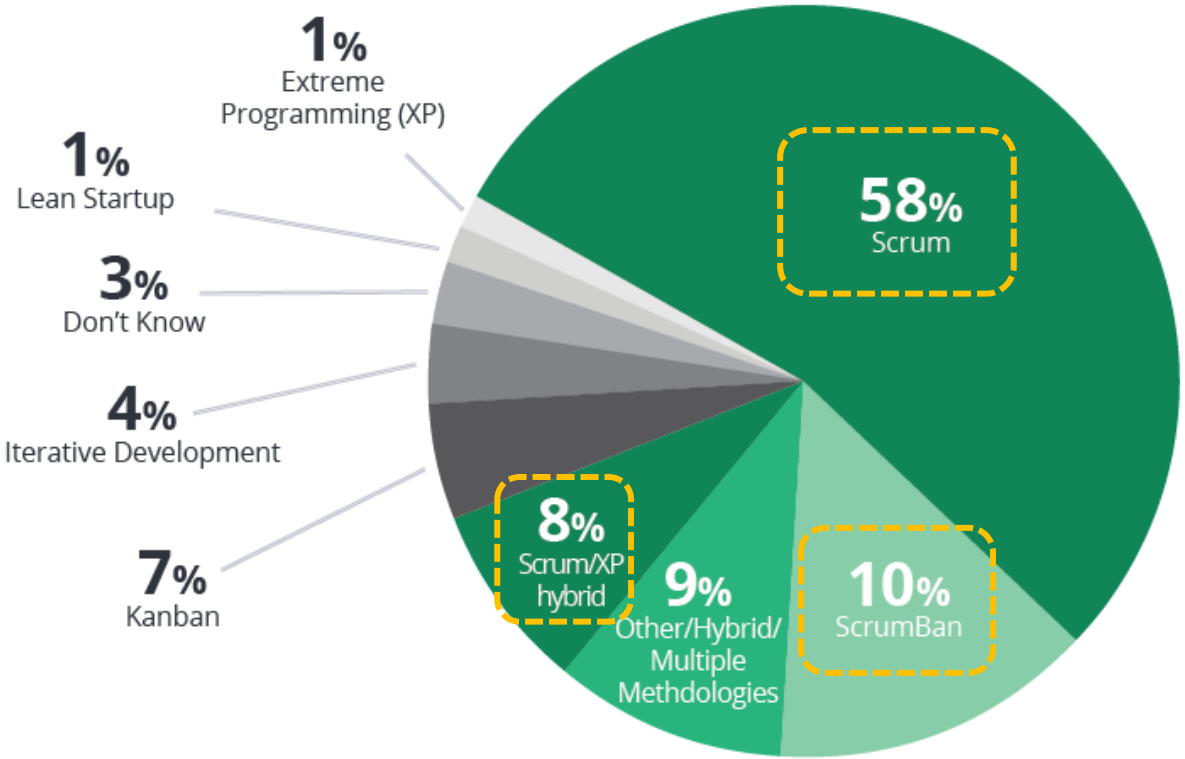
アジャイルとスクラムの関係

スクラムはアジャイルに包含される。
他にもXPやLeanの流れを汲むKanbanなどがアジャイルに含まれる。



アジャイルの中でのスクラムの採用率

ハイブリッドを合わせると、**76%**がスクラムを採用



Total exceeds 100% due to rounding.

VersionOne 14th Annual State of Agile Report
(http://stateofagile.versionone.com/)
2020年8月24日11時の最新版を取得

スクラムの起源

スクラム自体は**1995年**にJeff SutherlandとKen Schwaberによって発表されているが、
スクラムの原点となっている**野中郁次郎**、**竹内弘高**の研究論文
「The New New Product Development Game」は**1986年**に発表されている。
論文の中で、柔軟で自由度の高い日本発の開発手法をラグビーの**スクラム**に喻えて
「**Scrum**（スクラム）」として紹介した。

Luke Burgess introduces the ball into the scrum.
(https://ja.wikipedia.org/wiki/%E3%83%95%E3%82%A1%E3%82%A4%E3%83%AB:ST_vs_Gloucester_-_Match_-_23.JPG)
2020年9月7日18時に最新情報を取得

スクラムの定義

スクラムとは、**複雑な問題に対応**する適応型のソリューションを通じて、
人々、チーム、組織が**価値を生み出すための軽量級フレームワーク**である。

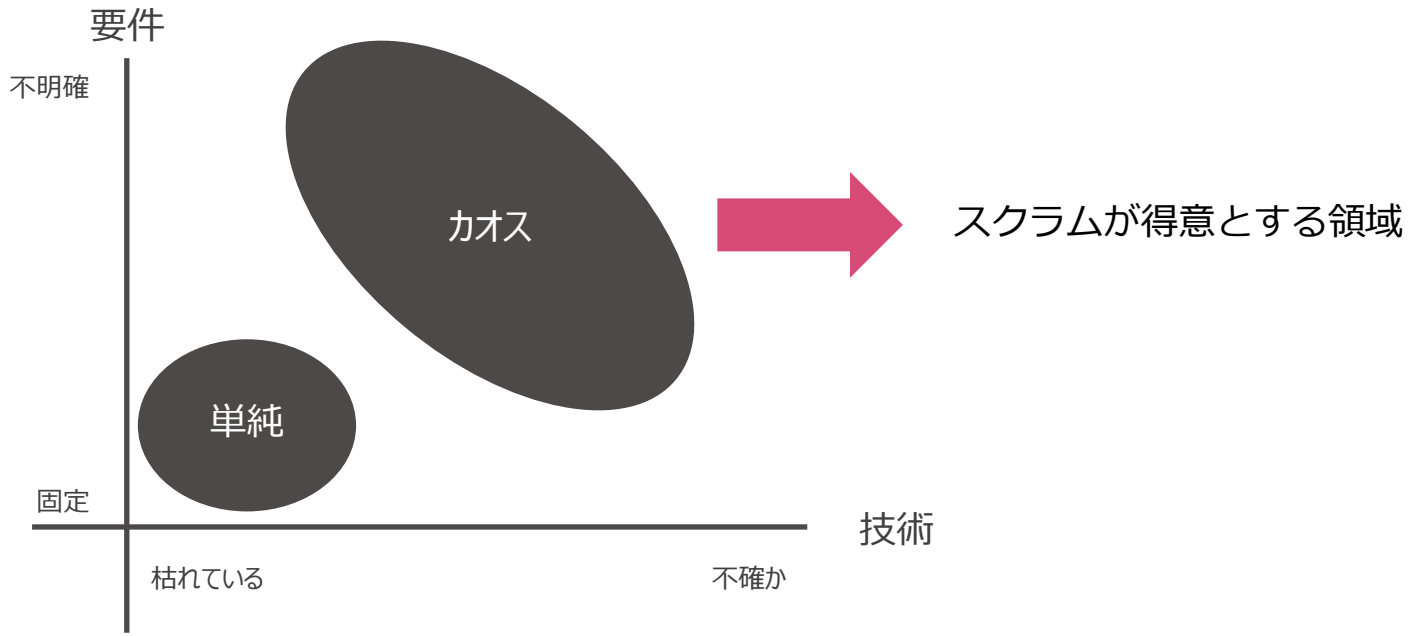
スクラムのルールは 詳細な指示を提供するものではなく、実践者の関係性や
相互作用をガイドするものである。

簡単に言えば、スクラムとは
次の環境を促進するためにスクラムマスターを必要とするものである。

1. プロダクトオーナーは、複雑な問題に対応するための作業を
   プロダクトバックログに並べる。
2. スクラムチームは、スプリントで選択した作業を価値の
   インクリメントに変える。
3. スクラムチームとステークホルダーは、結果を検査して、
   次のスプリントに向けて調整する。
4. 繰り返す。

スクラムに向かないプロジェクト

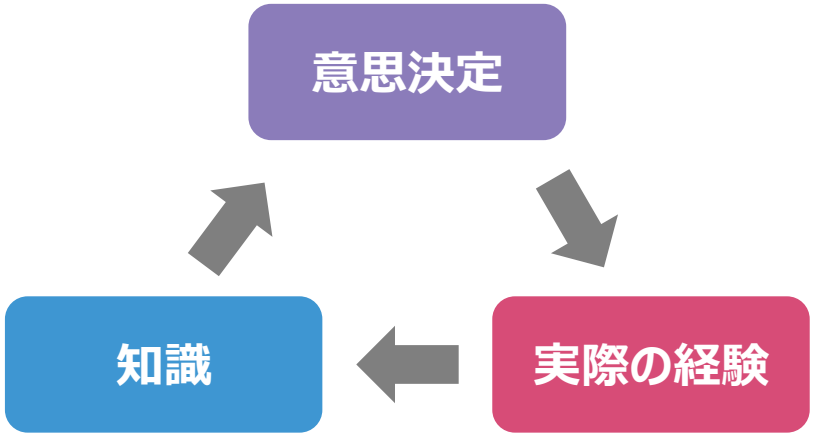
- プロジェクトの状態が**単純**
- **決まりきったモノ**を作る
- チームの**生存期間が短い**
  - ３ヶ月より短いと技術やプロセスに習熟しない



スクラムの理論

スクラムは「**経験主義**」と「**リーン思考**」に基づいている。

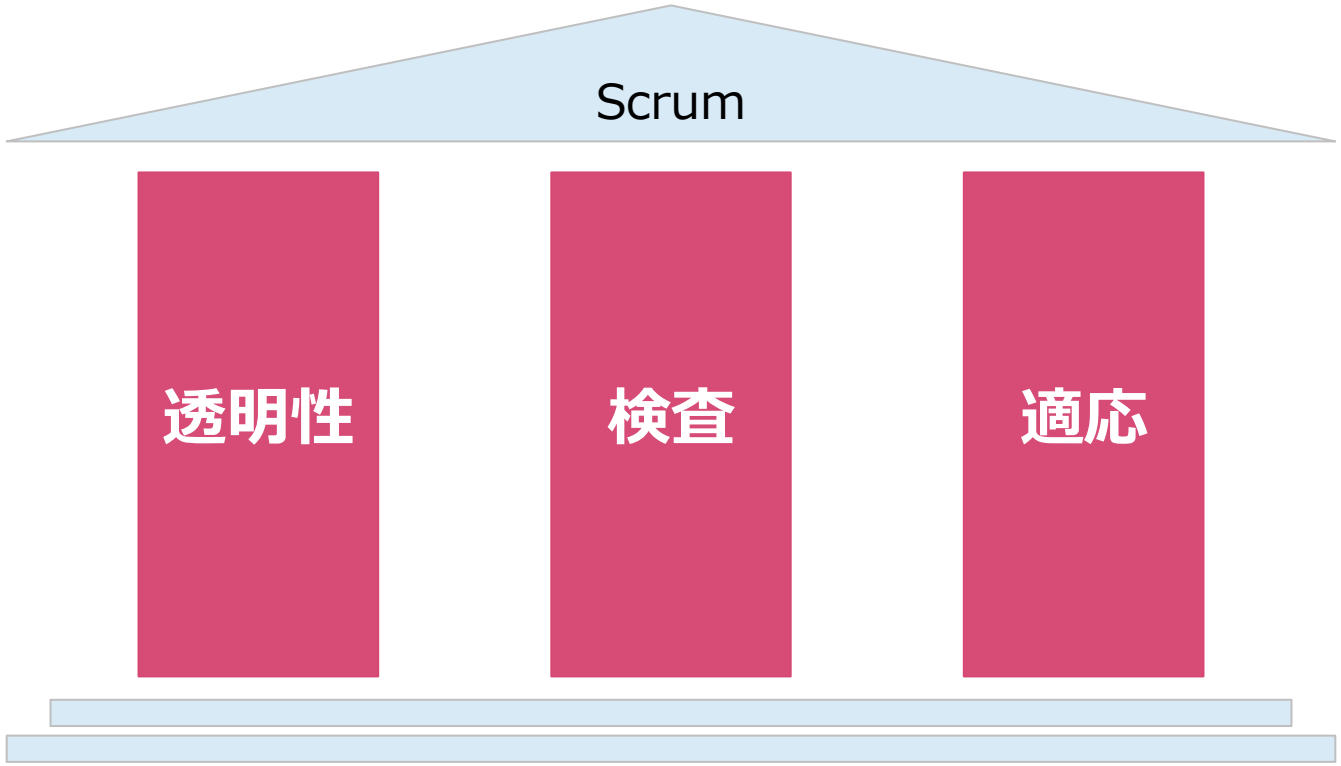
経験主義　＝ 知識は経験から生まれ、意思決定は観察に基づく。
リーン思考 ＝ ムダを省き、本質に集中する。



スクラムでは、予測可能性を最適化してリスクを制御するために、
イテレーティブ（反復的） で**インクリメンタル（漸進的）**なアプローチを
採用している。

スクラムを支える3本柱

「経験主義」であるスクラムは、以下の3つによって支えられている。
スクラムのイベントが機能するのは、このスクラムの３本柱を実現するからである。



5つの価値基準

スクラムチームに求められる**作業・行動・振る舞いの方向性**。
これらの価値基準を取り入れて実践することで「透明性」「検査」「適応」が実現可能。

スクラム開発の３・５・３

スクラムの３・５・３

3つの役割


プロダクトオーナー


スクラムマスター


開発者(複数人)

その他


ステークホルダー

5つのイベント


スプリント
（1~4週間）


スプリントプランニング


デイリースクラム


スプリントレビュー


レトロスペクティブ

３つの成果物


プロダクトバックログ


スプリントバックログ


インクリメント（作成物）

スプリントの流れ

スクラム開発の3・5・3

スクラムにおける3つのロール

スクラムの３・５・３

3つの役割

プロダクトオーナー

スクラムマスター

開発者(複数人)

その他

ステークホルダー

5つのイベント

スプリント
（1~4週間）

スプリントプランニング

デイリースクラム

スプリントレビュー

レトロスペクティブ

３つの成果物

プロダクトバックログ

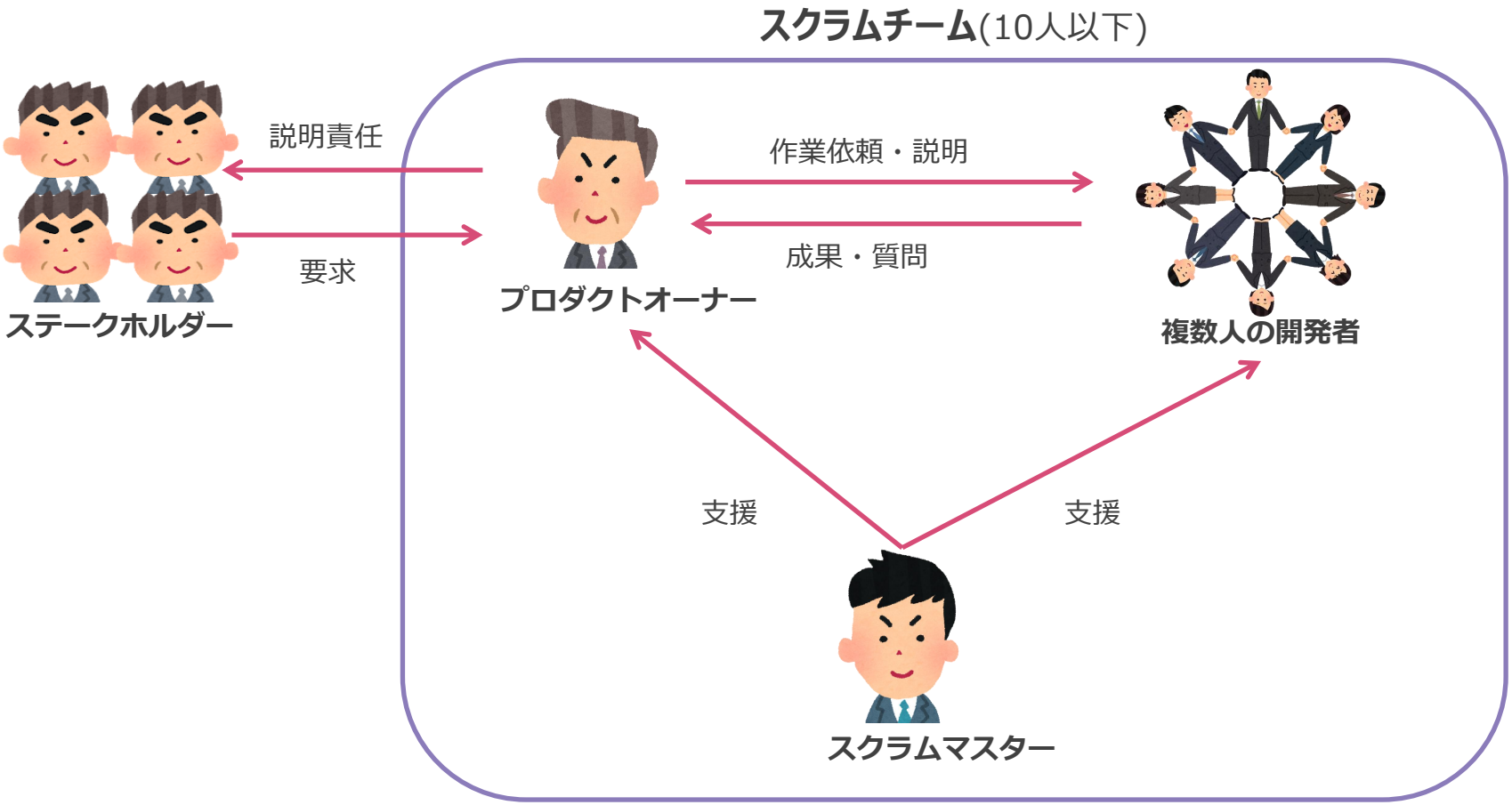
スプリントバックログ

インクリメント（作成物）

スプリントの流れ

3つの役割

プロダクトオーナー、スクラムマスター、複数人の開発者で構成される。



プロダクトオーナー(PO)

求められる能力

- コスト感覚
- ネゴシエーション力
- 説明力
- 決断力
- 一貫性
- 戦略性

役割

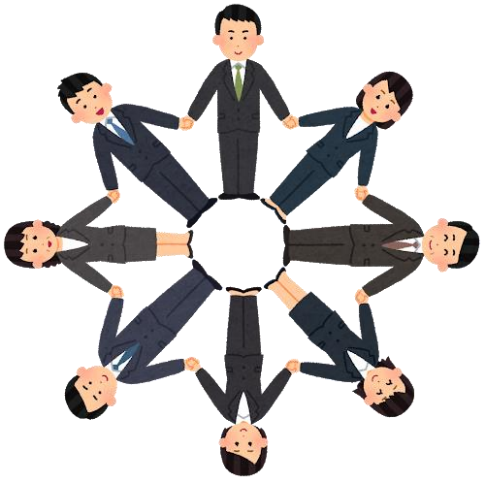
- スクラムチームから生み出されるプロダクトの**価値の最大化**に**責任**を持つ
- リリース日、リリース内容を決める
- **プロダクトバックログの管理に責任**を持つ**1人の人間**
- プロダクトバックログの優先順位の決定(委員会があっても構わないが、決定はPOの責任)
- 開発者への作業依頼(PO以外が開発者に作業依頼をしてはいけない)
- 作業結果の受入・拒否

仕事

- プロダクトゴールを策定し、明示的に伝える。
- プロダクトバックログアイテムを作成し、明確に伝える。
- プロダクトバックログアイテムを並び替える。
- プロダクトバックログに透明性があり、見える化され、理解されるようにする。

開発者

求められる能力



- 開発力
- **自己管理能力**
- 見積力
- **職能横断的**スキル(全員揃えばプロダクトを作れる)

役割

- リリース可能なモノを作成する
- 自分たちの作業の管理(**1番良いやり方を自分たちで考え、決める**)
- **生産性を向上するために努力する**

仕事

- スプリントの計画（スプリントバックログ）を作成する。
- 完成の定義を忠実に守ることにより品質を作り込む。
- スプリントゴールに向けて毎日計画を適応させる。
- 専門家としてお互いに責任を持つ。

スクラムマスター

求められる能力

- リーダーシップ
- ティーチング力
- ファシリテーション力
- コーチング力
- スクラム以外のプロセスの理解
- 集団心理の理解
- 事実(数字)を示す

役割

- **スクラムの確立、スクラムチームの有効性**に責任を持つ

仕事

- 自己管理型で機能横断型のチームメンバーをコーチする。
- スクラムチームが完成の定義を満たす価値の高いインクリメントの作成に集中できるよう支援する。
- スクラムチームの進捗を妨げる障害物を排除するように働きかける。
- すべてのスクラムイベントが開催され、ポジティブで生産的であり、タイムボックスの制限が守られるようにする。

スクラム開発の３・５・3

スクラムの3つの作成物

スクラムの３・５・３

3つの役割

プロダクトオーナー

スクラムマスター

開発者(複数人)

その他

ステークホルダー

5つのイベント

スプリント（1~4週間）

スプリントプランニング

デイリースクラム

スプリントレビュー

レトロスペクティブ

３つの成果物


プロダクトバックログ


スプリントバックログ


インクリメント（作成物）

スプリントの流れ

プロダクトバックログ

創発的かつ**順番に並べられた**、プロダクトの改善に必要なものの一覧。
スクラムチームが行う作業の**唯一の情報源**である。

優先順位	ストーリー	見積
1	AとしてXXが出来る。	2
2	BとしてYYを一覧形式で参照できる。	3
3	C処理の性能改善	5
…	…	…
100	Dとしてレポートを作成できる	8

明確

不正確で粗い

リファインメントによって
明確にしていく

プロダクトバックログの1要素をプロダクトバックログアイテム(**PBI**)と呼ぶ。

PBIがより小さく詳細になるように、分割および定義をする活動を**リファインメント**と呼ぶ。これは、説明・並び順・サイズなどの詳細を追加するための**継続的な活動**。
スプリントプランニングで選択するPBIは十分に明確になっている必要がある。

スプリントバックログ

スプリントで選択したプロダクトバックログアイテムと、それらをインクリメントとして届けるための計画を合わせたもの。

開発者が**スプリントゴールを達成するのに必要な作業の一覧**。
開発者による、開発者のための計画であり、スプリントの期間を通して更新される。

見積りは**理想時間**で行う。

ストーリー	タスク	見積
AとしてXXが出来る。	UIのコーディング	3.0h
	データモデル設計、変更、Entityの作成	3.0h
	Actionのコーディング	2.0h
BとしてYYを一覧形式で参照できる。	UIのコーディング	4.0h
	Actionのコーディング	2.0h

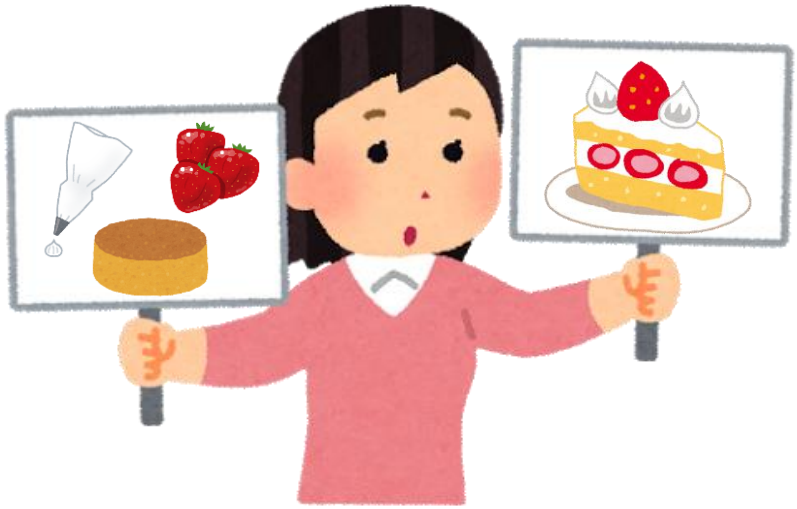
インクリメント

プロダクトゴールに向けた作成物。
インクリメントはこれまでのすべてのインクリメントに追加する。

スプリントでは、複数のインクリメントを作成可能である。インクリメントをまとめたものをスプリントレビューで提示する。

価値を提供するには、インクリメントを**利用可能**にしなければならない。
完成の定義を満たさない限り、作業をインクリメントの一部と見なすことはできない。

部品だけでは出荷できない。**小さくても使えるもの**を作る。



Done(完成)の定義

出荷可能な製品の「完成」の定義。作業が完了したかどうかの評価に使われる。
リリースするためにやらなければならないこと。

Done

毎スプリント完了しているもの

- 開発
- UT
- カバレッジ80%以上
- IT
- リグレッションテスト
- ドキュメント作成
- 静的解析

Undone

毎スプリント実施できていないもの

- シナリオテスト
- 負荷テスト
- セキュリティテスト
- リリース

UndoneをDoneにしていくことが重要

スクラム開発の３・**５**・３

スクラムにおける**５**つのイベント

スクラムの３・５・３

TIS
TIS INTEC Group

3つの役割

プロダクトオーナー

スクラムマスター

開発者(複数人)

その他

ステークホルダー

5つのイベント


スプリント
（1~4週間）


スプリントプランニング


デイリースクラム


スプリントレビュー


レトロスペクティブ

３つの成果物

プロダクトバックログ

スプリントバックログ

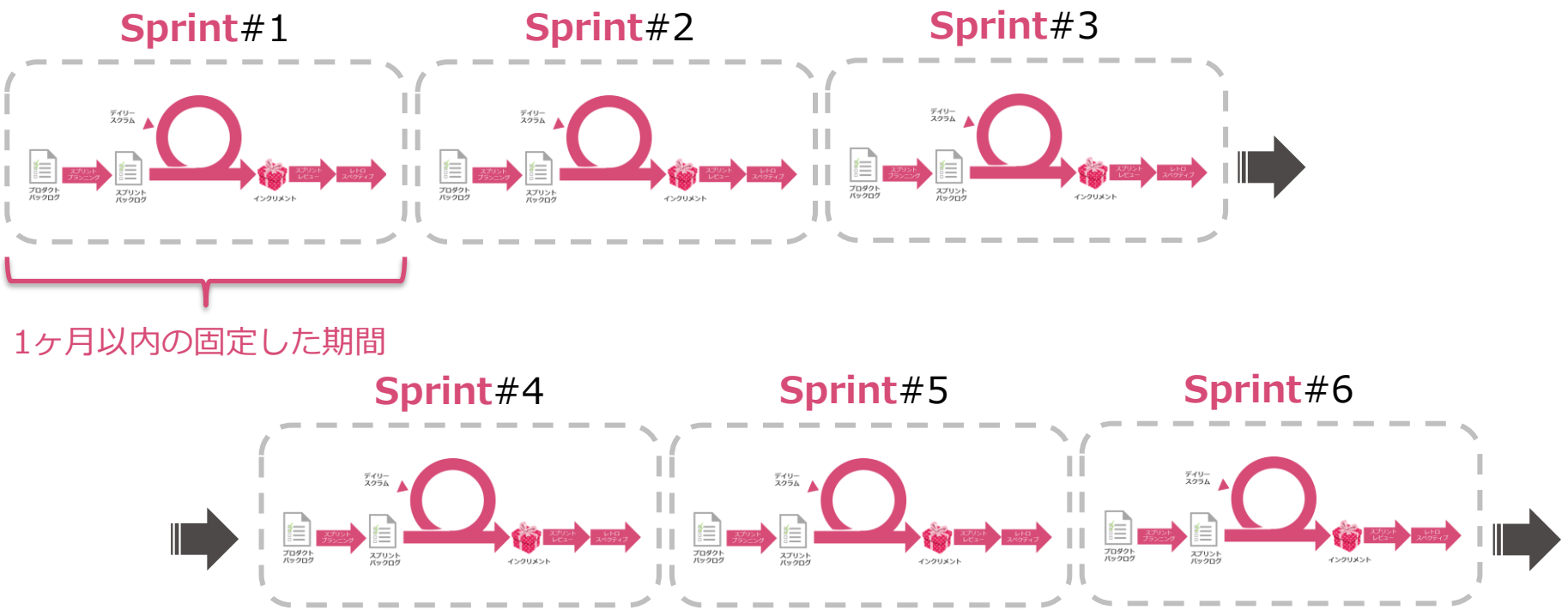
インクリメント（作成物）

スプリントの流れ

スプリント

1ヶ月以下のタイムボックスで、開発作業を行う連続した期間。

スプリントによって、プロダクトゴールに対する進捗の検査と適応が少なくとも1ヶ月ごとに確実になり、予測可能性が高まる。



スプリントプランニング

タイムボックス:　4時間 / 2週間

スプリントで実行する作業の計画を立てる。

スプリントプランニング第一部では以下のことに対応する。

1. スプリントゴールを定義する。
2. スプリントで実施する**プロダクトバックログアイテム(PBI)を選択**する。

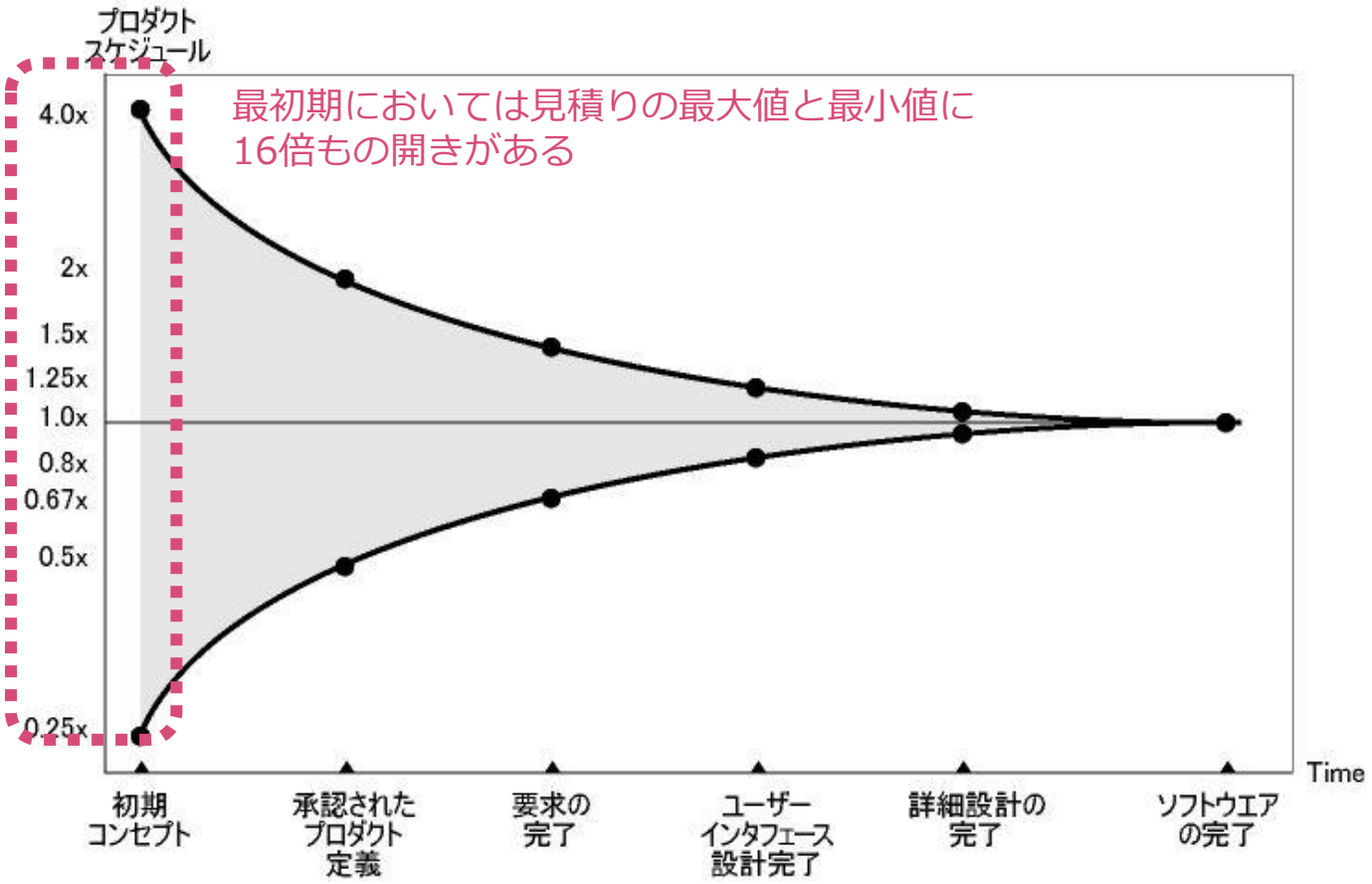
プロダクトバックログ

優先順位	ストーリー	見積
1	AとしてXXが出来る。	2
2	BとしてYYを一覧形式で参照できる。	3
3	C処理の性能改善	5
…	…	…
100	Dとしてレポートを作成できる	8

見積ができるのは開発者だけ。
どのくらいできるかの選択も開発者が行う。

不確実性コーン

プロジェクトが進行するにつれて見積もりのバラツキがどのように推移していくのかを表している。



http://itpro.nikkeibp.co.jp/article/COLUMN/20131001/508039/

スプリントプランニング

スプリントプランニング第二部では、開発者が、第一部で選択したPBIごとに、
完成の定義を満たすインクリメントを作成するために必要な作業を計画する。

プロダクトバックログ

ストーリー
AとしてXXが出来る。
BとしてYYを一覧形式で参照できる。

小さな作業アイテムに分解する

スプリントバックログ

タスク	見積
UIのコーディング	3.0h
データモデル設計、変更、Entityの作成	3.0h
Actionのコーディング	2.0h
UIのコーディング	4.0h
Actionのコーディング	2.0h

どのように行うかは、開発者だけの裁量。
作業の分解は、必要に応じてスプリント期間中も実施する。

デイリースクラム

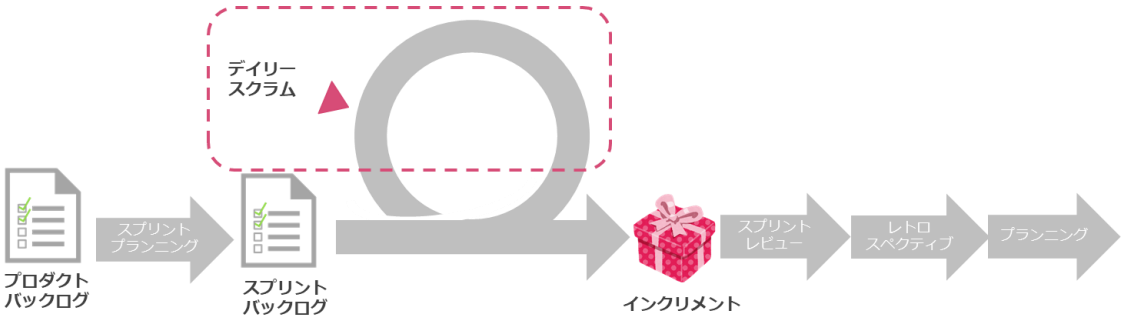
タイムボックス:　15分 / 1日

スプリントゴールへの確約(Commitment)の**進捗**と**よりよい仕事のやり方**を議論する。

デイリースクラムの目的は、計画された今後の作業を調整しながら、スプリントゴールに対する進捗を検査し、必要に応じてスプリントバックログを適応させることにある。

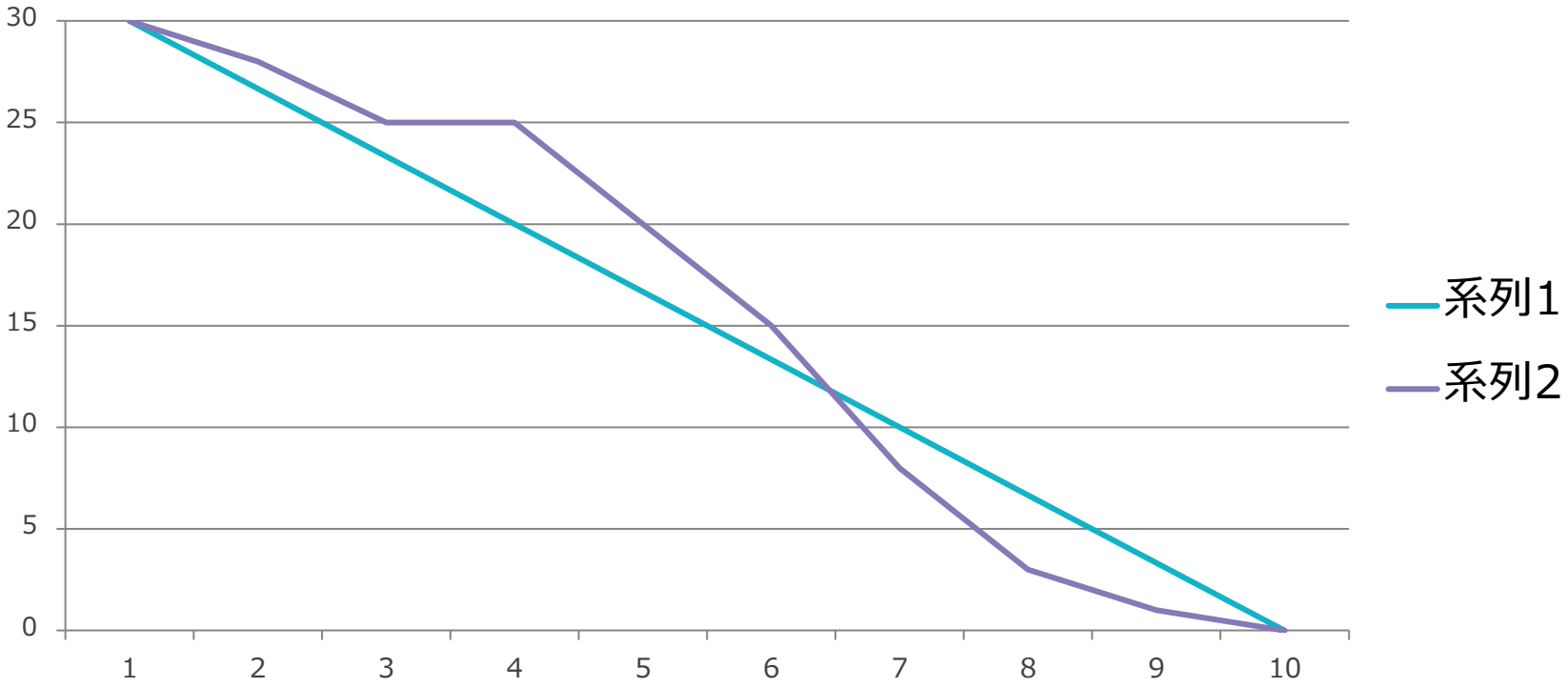
- この場でスプリントバックログの**作業進捗を検査**する。
- デイリースクラムは**毎回同じ時間**、**場所**で開催する。

開発者が計画を調整できるのは、デイリースクラムのときだけではない。スプリントの残りの作業を適応または再計画することについて、より詳細な議論をするために、開発者は一日を通じて頻繁に話し合う。



スプリントの進捗の確認

スプリントバックログの残作業量を追跡し、進捗を確認する。
デイリースクラムでバーンダウンチャートを確認することが多い。



スプリントレビュー

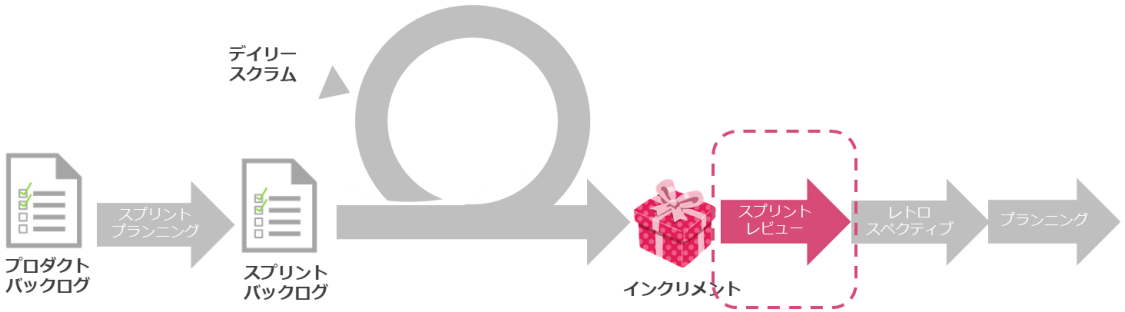
タイムボックス:　2時間 / 2週間

スプリントの**成果を検査**し、**今後の適応**を決定する。
スクラムチームは、主要なステークホルダーに作業の結果を提示し、**プロダクトゴールに対する進捗**について話し合う。

何が達成された？

環境の変化は？

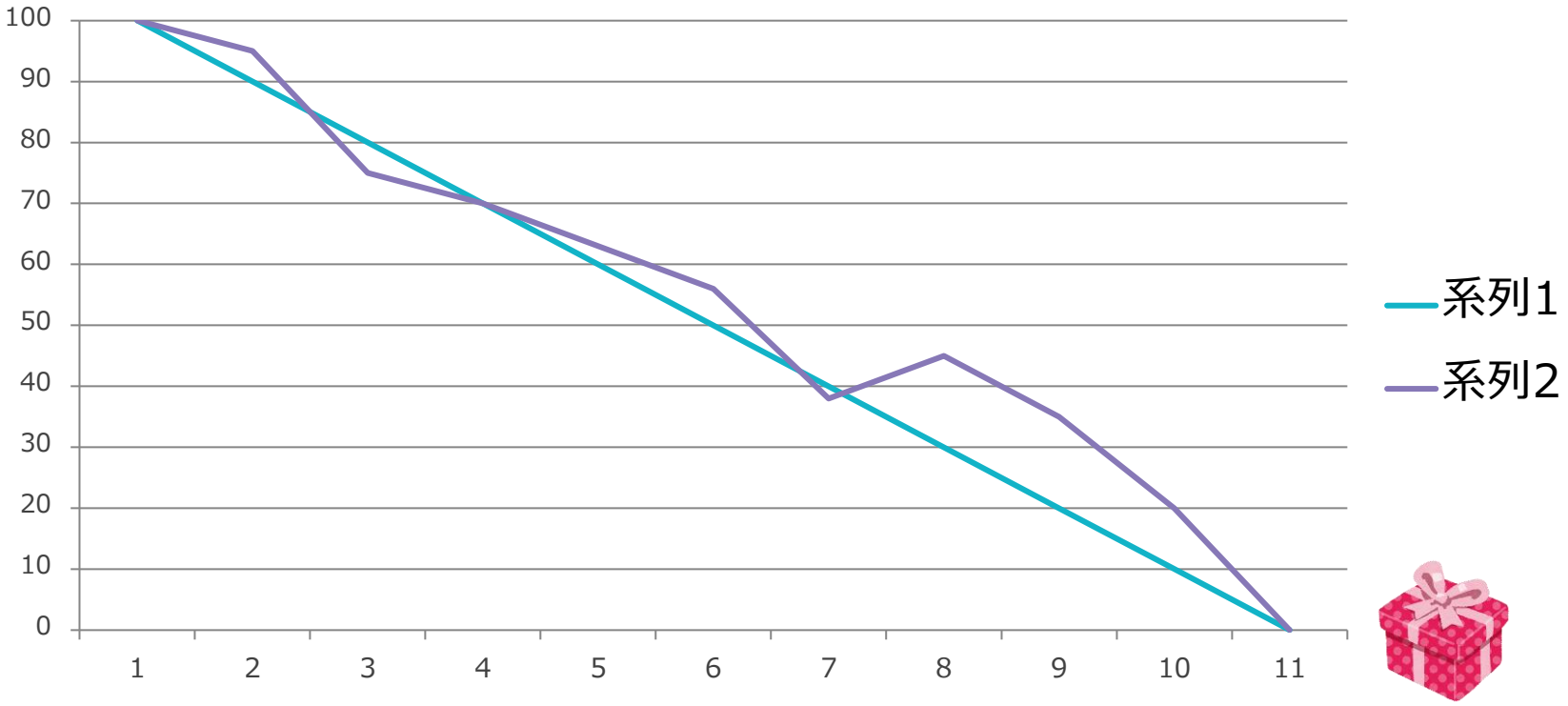
- ✓ スプリントレビューの情報に基づいて、参加者は次にやるべきことに協力して取り組む。
- ✓ プロダクトバックログを調整することもある。
- ✓ スプリントレビューはワーキングセッションであり、スクラムチームはスプリントレビューを**プレゼンテーションだけに限定しない**ようにする。



ゴールへの進捗の確認

リリースバーンダウンチャートなどを用いて、
希望している内容が希望しているタイミングでリリースできるかを追跡、確認する。

スプリントレビュー時に確認するとプロダクトオーナーにスコープの調整の材料を与えることができる。リリースに必要なポイント数とタイミングが分かれば、１スプリント実施するだけで遅れの有無がわかる。

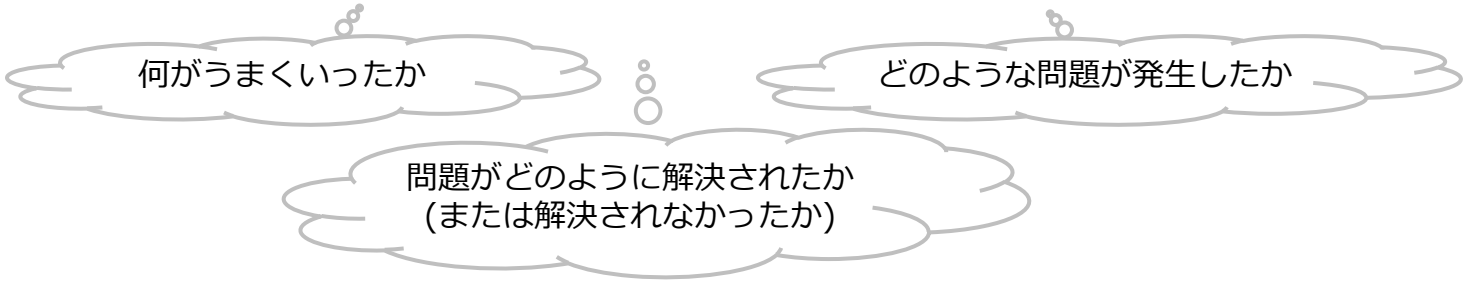


スプリントレトロスペクティブ

タイムボックス:　**1.5**時間 / 2週間

スクラムチームの検査と**次のスプリントの改善計画**を作成する。
スクラムチームは、個人、相互作用、プロセス、ツール、完成の定義に関して、今回のスプリントがどのように進んだかを検査する。スクラムチームを迷わせた仮説があれば特定し、その真因を探求する。

- スクラムチームは、自分たちの効果を改善するために最も役立つ変更を特定する
- 最も影響の大きな改善は、できるだけ早く対処する



会議体及び出席者

ミーティング出席者 ガイドライン	スプリント プランニング 第一部	スプリント プランニング 第二部	デイリー スクラム	スプリント レビュー	レトロ スペクティブ
プロダクト オーナー	○	○ [1]	△ [2]	○	△ [4]
スクラム マスター	○	○	△ [3]	○	○
開発者	○	○	○	○	○
ステーク ホルダー	⊘	✕	⊘	⊘	✕

○：参加すべき
✕：参加不可
△：コンテキストにより参加可能
⊘：参加可能だが決定権なし

[1] スプリントプランニング第二部へのプロダクトオーナーの参加
スプリントプランニング第二部では実現方法及び実現するための計画について話し合うため、常に場にいる必要はない。
ただし、必要なタイミングで質問に答えられるようにしておくこと。
また、開発者はプロダクトオーナーに対し、どのようにスプリントゴールを達成するのかを説明できる必要がある。
[2] デイリースクラムへのプロダクトオーナーの参加
スプリントゴールを達成するにあたっての障害事項が頻繁に表れ、プロダクトオーナーの支援が必要な場合などに置いて有効。
[3] デイリースクラムへのスクラムマスターの参加
スクラムマスターは開発者にデイリースクラムを開催してもらうようにするが、開催する責任は開発者にある。
初期のチームにおいては、タイムボックスの順守やルールを守らせる役割として入ることもある。
開発者だけでうまくデイリースクラムを行えているのであれば、参加する必要はない。
[4] スプリントレトロスペクティブへのプロダクトオーナーの参加
プロセスの検査と改善が目的のため、POがいることで率直な問題点を明らかにできない場合はPO抜きで行う。

まとめ

3つの役割


プロダクトオーナー


スクラムマスター


開発者(複数人)

その他


ステークホルダー

5つのイベント


スプリント
（1~4週間）


スプリントプランニング


デイリースクラム


スプリントレビュー


レトロスペクティブ

３つの成果物


プロダクトバックログ


スプリントバックログ


インクリメント（作成物）

スプリントの流れ

「スクラムガイド」 2020年11月版

https://scrumguides.org/docs/scrumguide/v2020/2020-Scrum-Guide-Japanese.pdf

スクラムと品質

ウォーターフォールでの品質課題

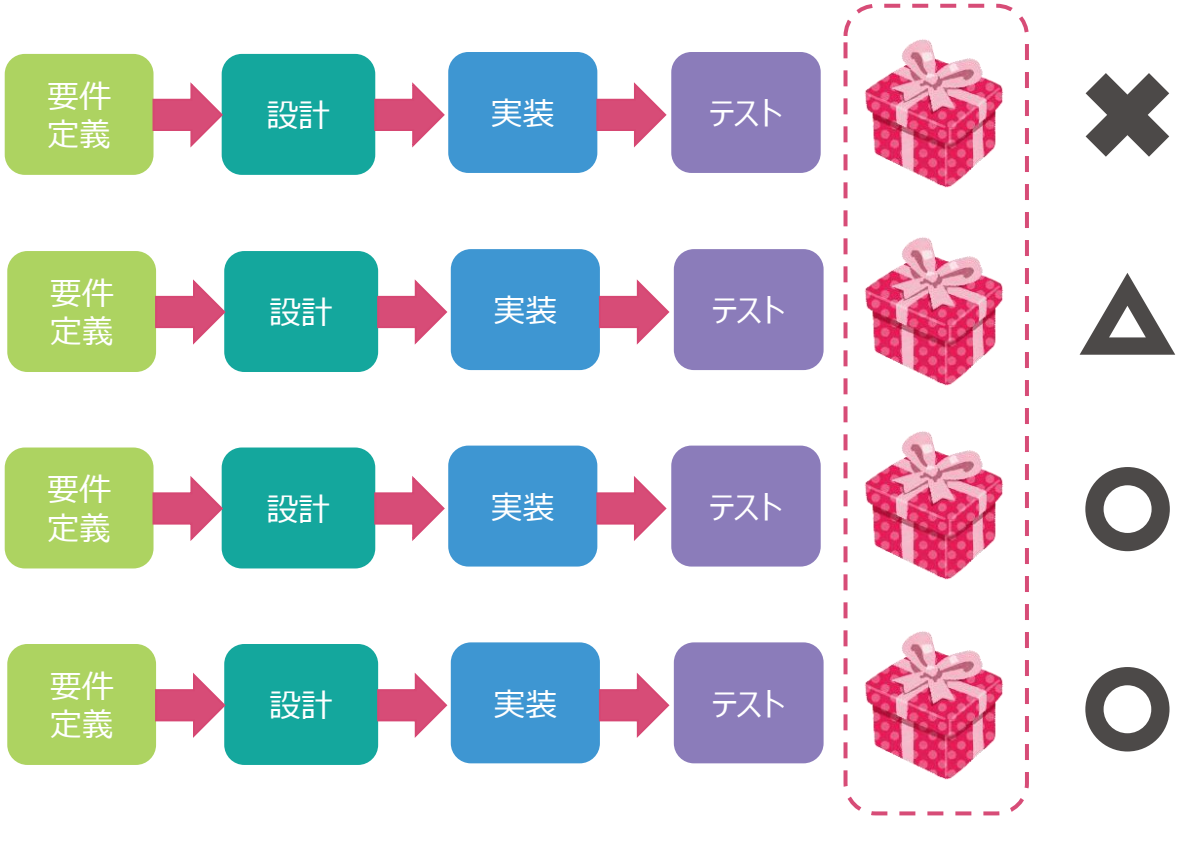
フェーズゲートにより内部品質は担保されるが、
作成した製品が本当に欲しいモノだったのかわかるのは
受入テストのタイミング。

大きなリスクが後半に待っている。

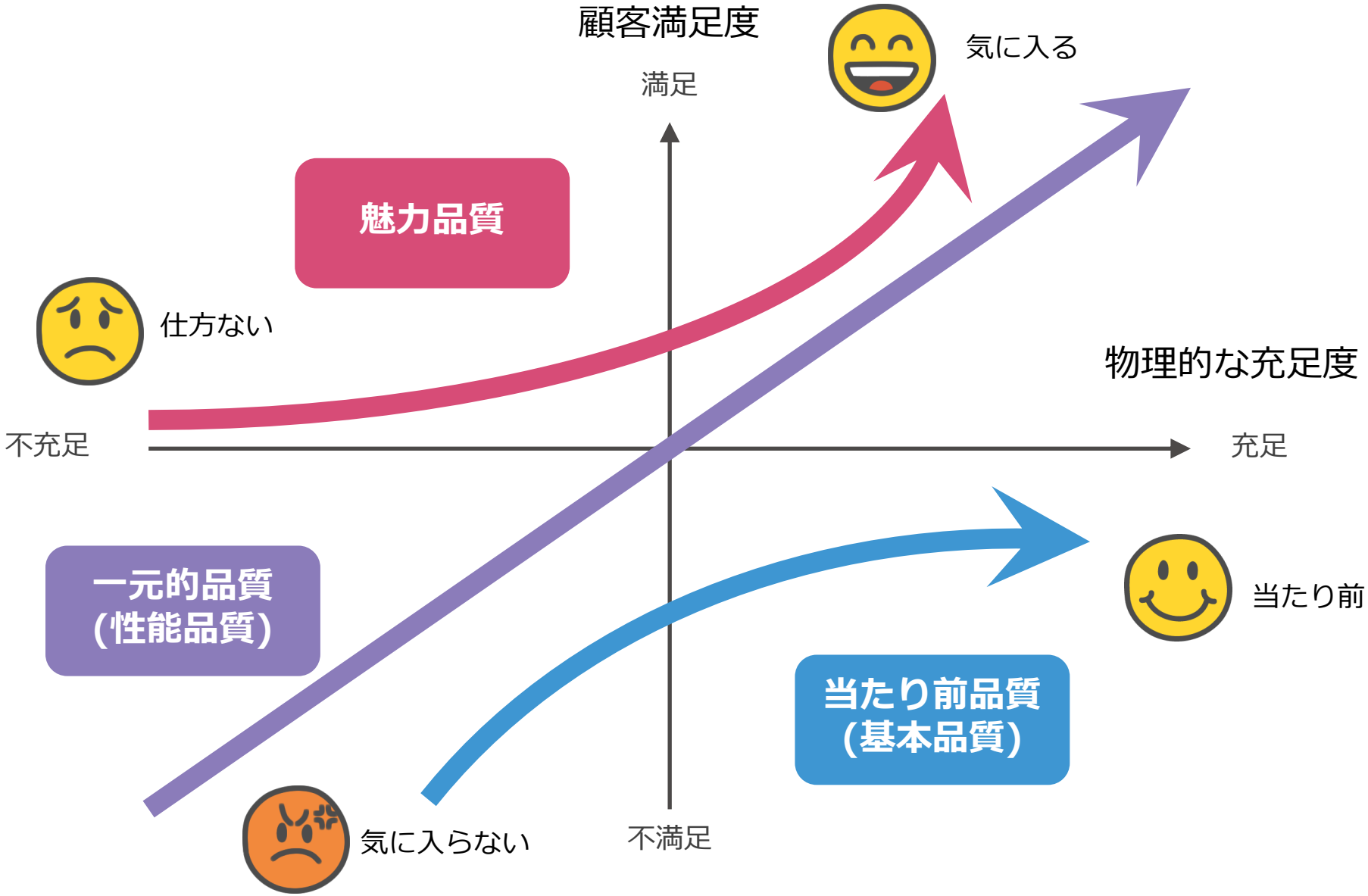
スクラムでは

スプリントレビューで**ビジネス要件との適合性**を確認

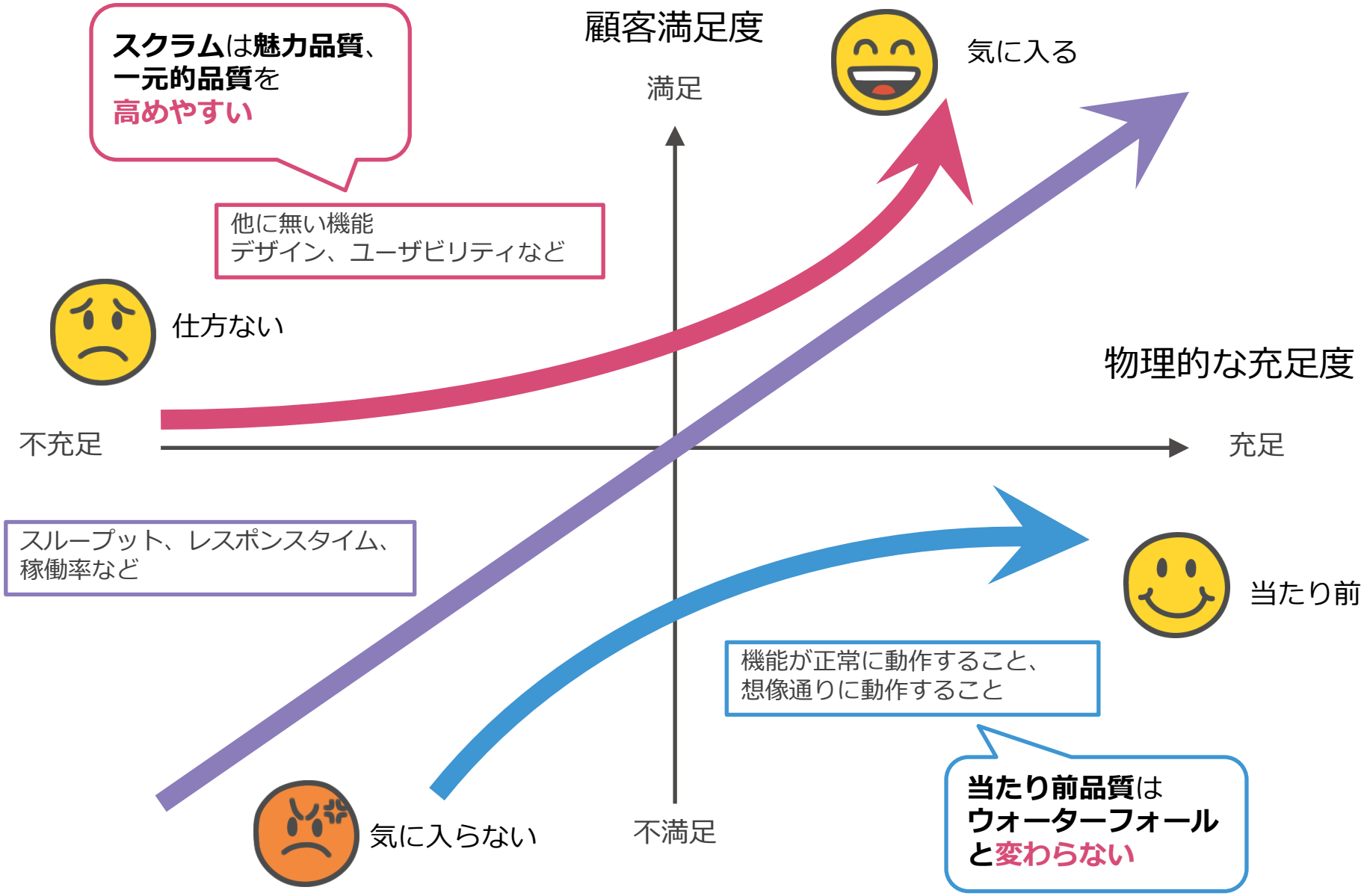
時間



狩野モデル



狩野モデル

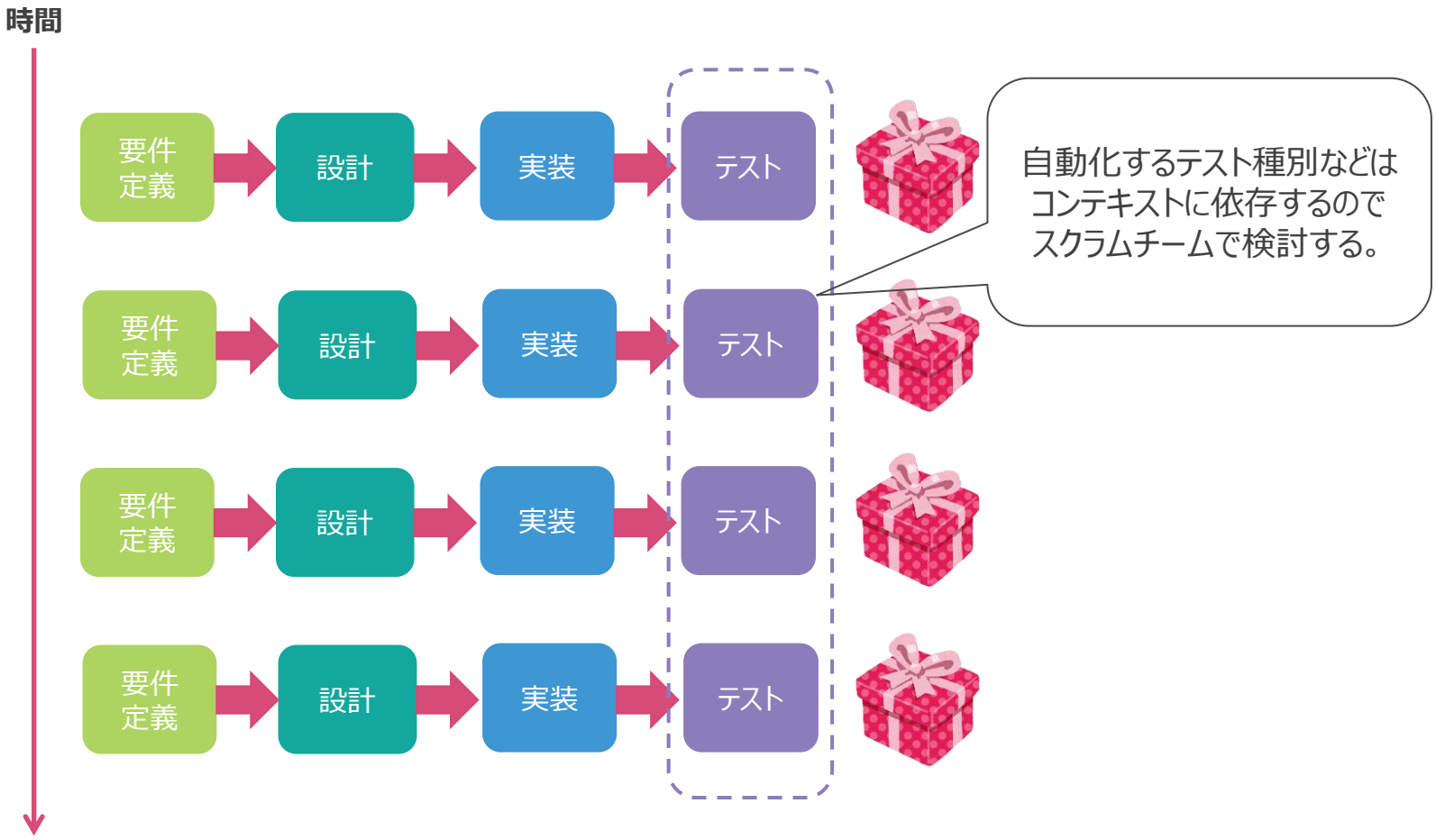


QCD + S

- Quality(品質) ： 固定
- Cost(コスト・体制) ： 固定
- Delivery(価値提供までの時間) ： 固定
- Scope(スコープ) ： **調整可能**

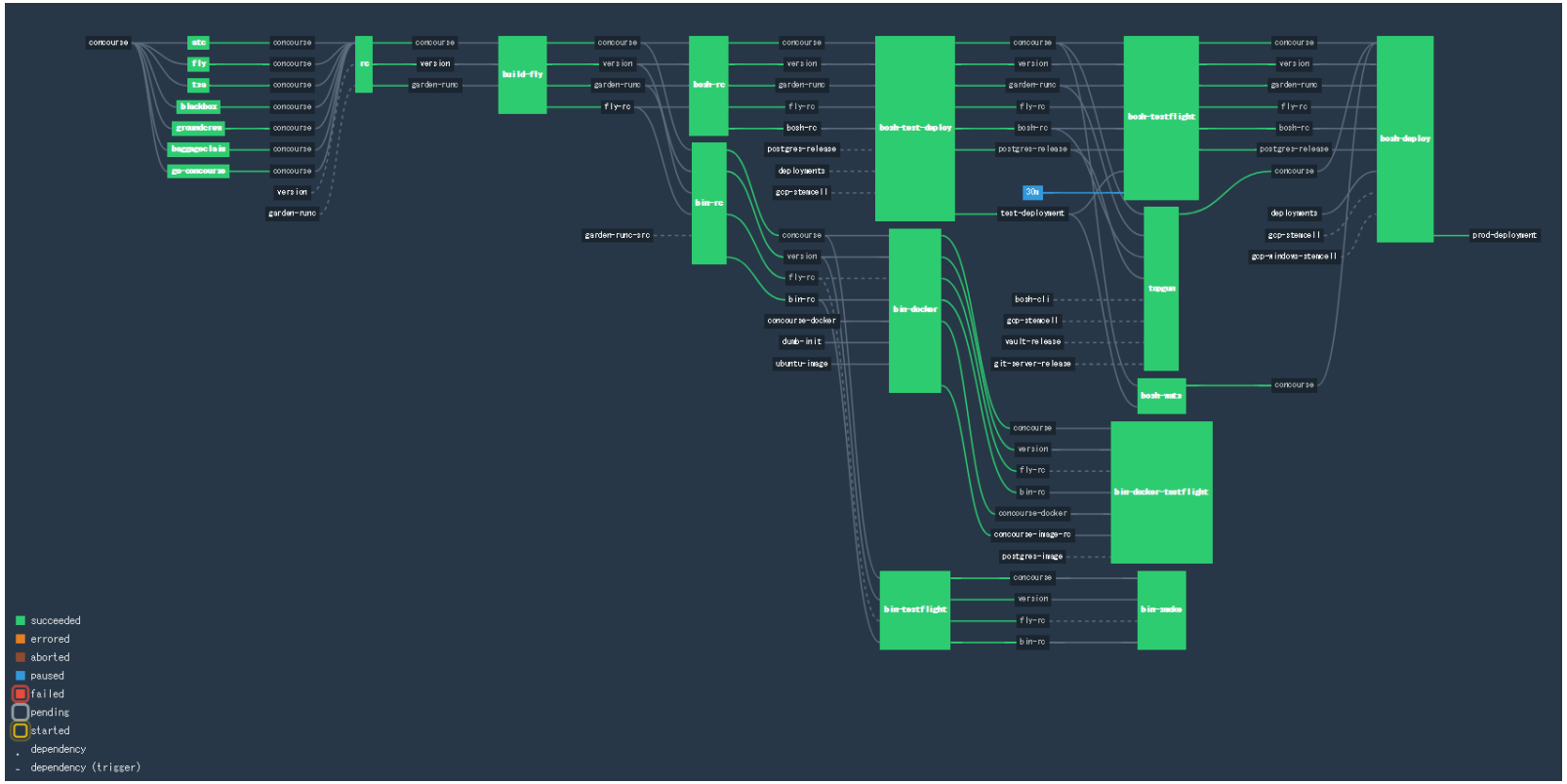
テスト自動化

繰り返し開発していくことになるので、テストを行う回数がウォーターフォールより多い。
前のスプリントで追加した機能のリグレッションテストなどを考えると、**自動化は必須**。



継続的インテグレーション/デリバリー

テスト自動化と同様に、頻繁にビルド・デプロイを行うことになるため、自動化を推奨する。



スクラム概論
(https://fintan.jp/wp-content/uploads/2019/09/introduction-to-scrum.pdf)
2022年1月17日15時の最新版を取得

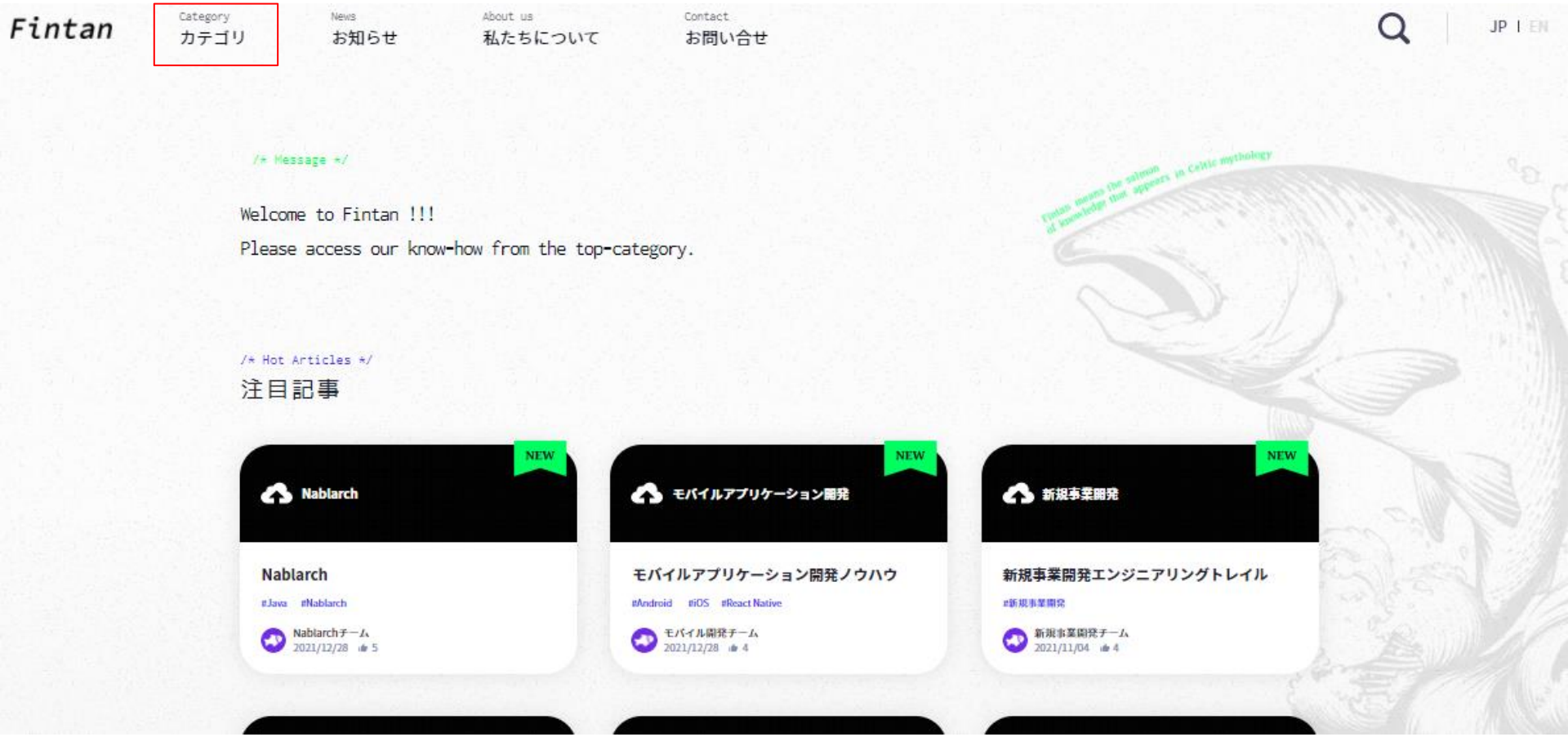
本日お伝えしたこと

1. なぜアジャイルを学ぶのか
   - 価値ある機能をどう提供するか
   - 受託開発とサービス開発の違い
   - ウォーターフォールとアジャイルの違い
2. アジャイルとは何か
   - アジャイルソフトウェア開発宣言とその背後にある原則
3. スクラムを知る
   - スクラムとは何か
   - スクラムの３・５・３
   - スクラムの品質

Don't do agile, be agile

おすすめの参考情報

Fintan（https://fintan.jp）では、TISインテックグループの様々な組織の技術ノウハウを公開しています。
アジャイル・スクラムに関する記事は『アジャイル・スクラム』のカテゴリにまとめられています。画面上部「カテゴリ」からたどれます。一見の価値アリです。



Fintan(https://fintan.jp)
2022年1月19日11時の最新版を取得

ITで、社会の願い叶えよう。

TIS INTEC
Group