

モダン・ソフトウェアエンジニアリング のエッセンス

2020年7月21日

ワイクル株式会社

角 征典 (@kdmsnr)





Martin Fowler says (2010)

- 厳格すぎて、価値が限られる
- Alistair Cockburnが、ソフトウェア開発では人が中心的な要素であり、人は本質的に非線形的で予測不能なものであると説明してくれた

<https://bliki-ja.github.io/Semat/>

Martin Fowler says (2010)

- 厳格すぎて、価値が限られる
- Alistair Cockburnが、ソフトウェア開発では人が中心的な要素であり、人は本質的に非線形的で予測不能なものであると説明してくれた
- 人が扱いやすい計算式で記述できる予測可能なエージェントになれば可能性はあるかもしれない

<https://bliko-ja.github.io/Semat/>

7.2 理論の使用

- 理論とは？
 - i) 現象を「記述」するもの
 - ii) 現象を「予測」するもの
- 「予測」するためには「記述」が必要であり、
「記述」するためには「言語」が必要である

7.2 理論の使用

- 理論とは？
 - i) 現象を「記述」するもの
 - ii) 現象を「予測」するもの
- 「予測」するためには「記述」が必要であり、
「記述」するためには「言語」が必要である
- 将来を予測したいが、まずはそのための言語が必要

Essenceのアーキテクチャ



図3-3 : Essence のアーキテクチャ

Essenceのアーキテクチャ

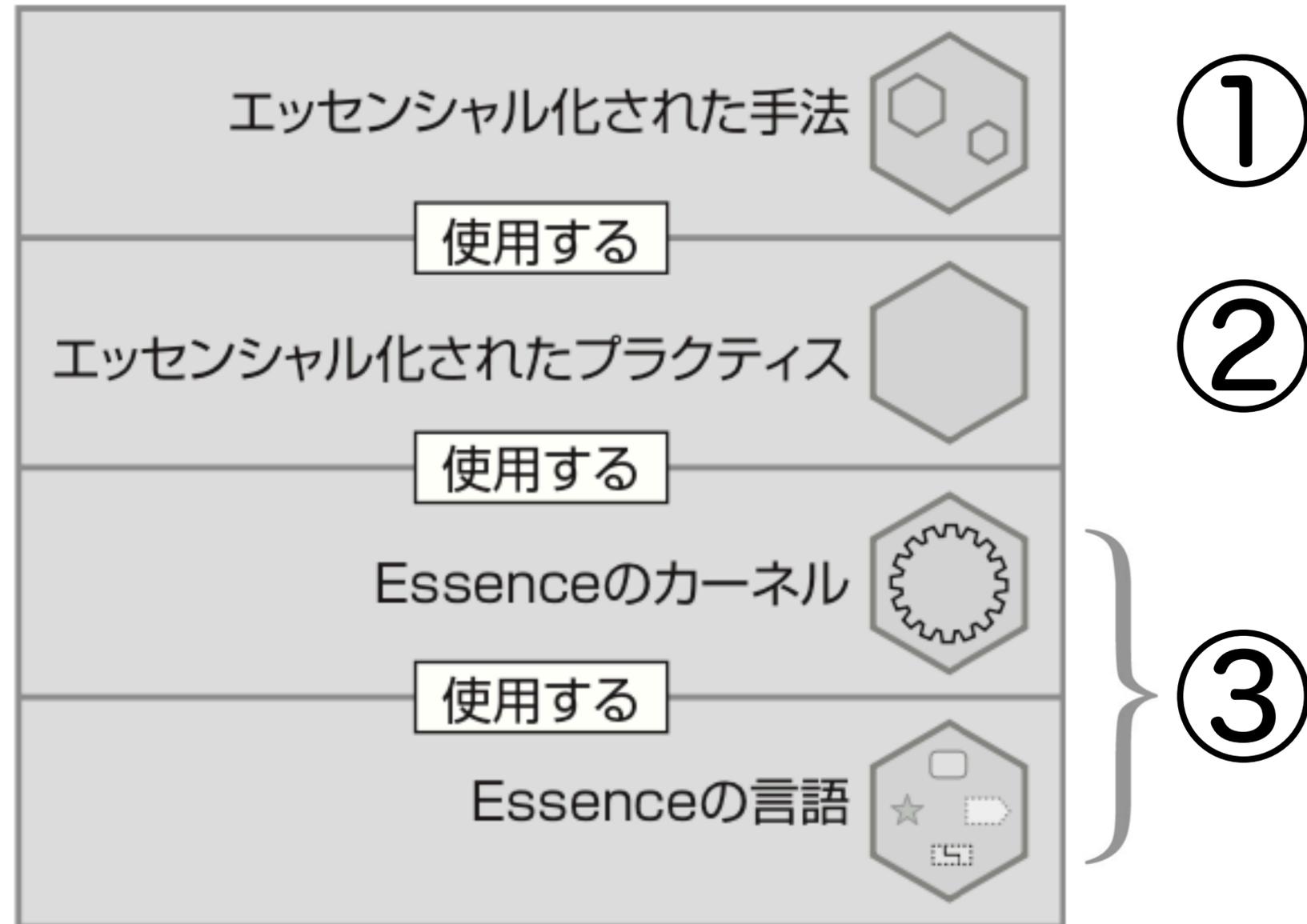


図3-3 : Essence のアーキテクチャ

①エッセンシャル化された手法

ざっくりと「手法」とは何か

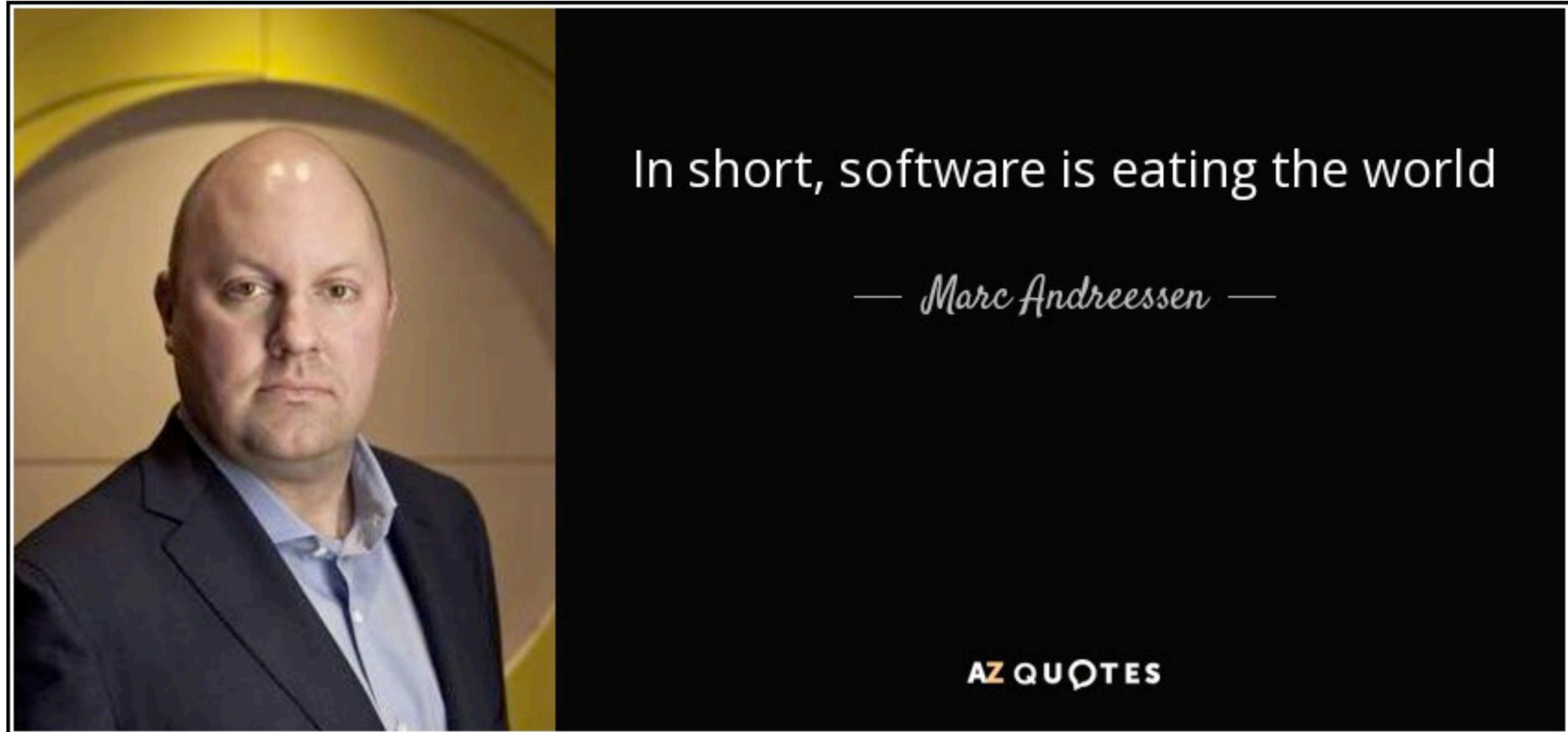
- 「ソフトウェアを開発・維持するときに必要となるすべてのことに対して、アドバイスを提供するもの」
- 「すべてのこと」はウォーターフォール手法が理解しやすい
 - うまくいかないことを除けば（！）そんなに悪くはない
- 現実的には、アジャイルを含む反復的手法でやるべき
 - とはいえ、自己組織化が前提だと「すべてのこと」を見逃しやすい
 - ∴ 全体の見取り図を別途用意しておくとうい

手法の問題は大きく2つある

- 乱立による足の引っ張り合い（**手法の戦争**）
 - 似たようなことをやってるのに名前が違う
 - 部品（プラクティス）がモジュール化されていたら再利用可能なのに！
- 手法の作者が決める絶対的なルールがある（**手法の監獄**）
 - 「 そんなものでは止めを刺せん」
- 書籍では触れていないが、認定資格制度の問題もありそう💧

「ソフトウェアが世界を食べる」時代（2011）

ソフトウェアの手法だけを語ってる場合じゃねえ!!



食後の手法はこうありたい

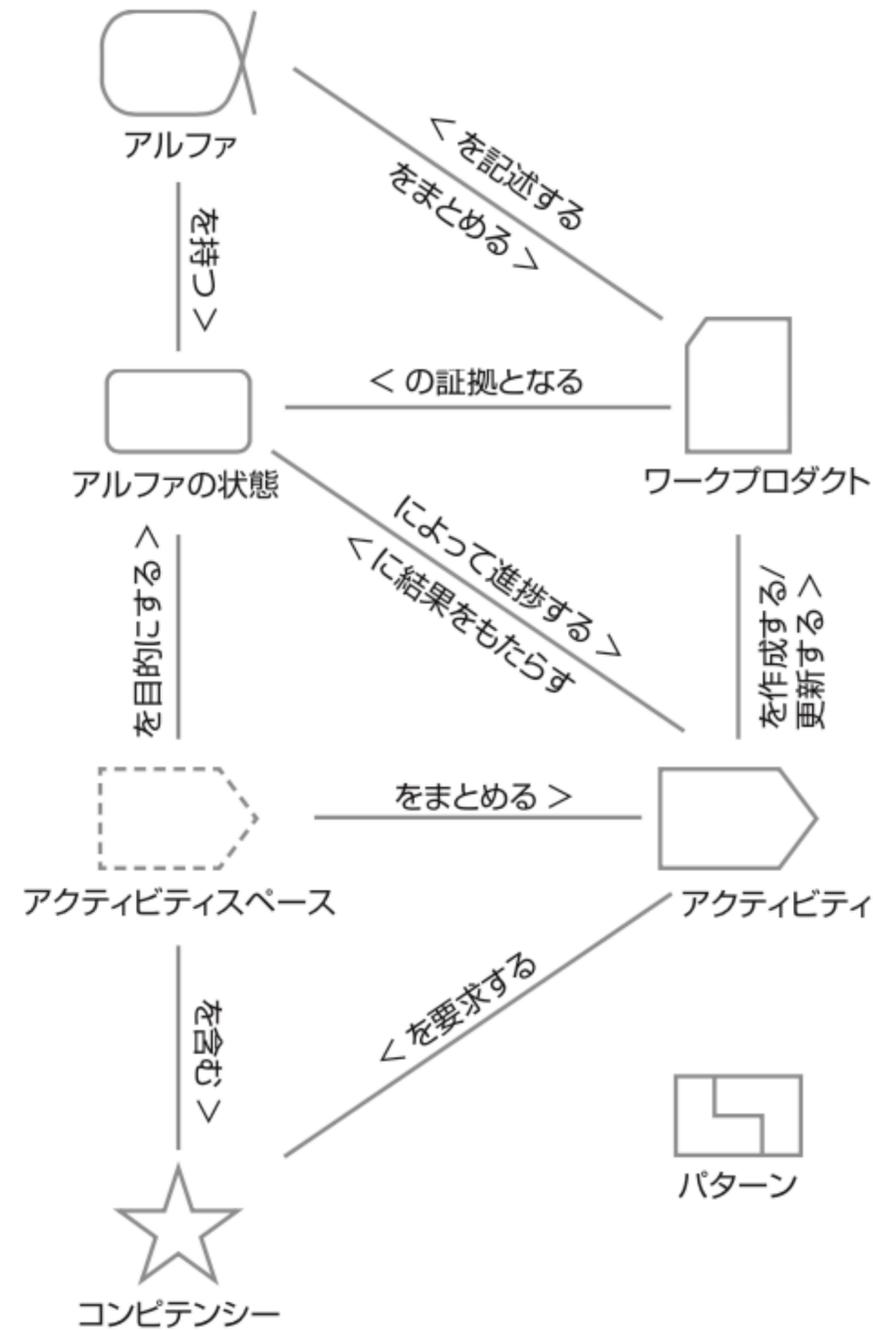
- 既製品を使うのではなく、状況にあわせて自分たちで手法を作りたい
- ただし、ゼロから手法を作るのは大変すぎる💧
 - ∴ 既存のプラクティスを「合成」すればいい (3.2 「手法はプラクティスの合成」)
- それに、自分たちで手法を作っても誰も理解してくれない💧
 - ∴ 「記述する言語」を統一して誰でも読めるようにすればいい
 - プロダクトのUMLに対するプロセスのEssenceという位置づけっぽい？
(UMLと同じくらいの希望と絶望を持つといいと思う……)

②エッセンシャル化された プラクティス

プラクティスとは何か

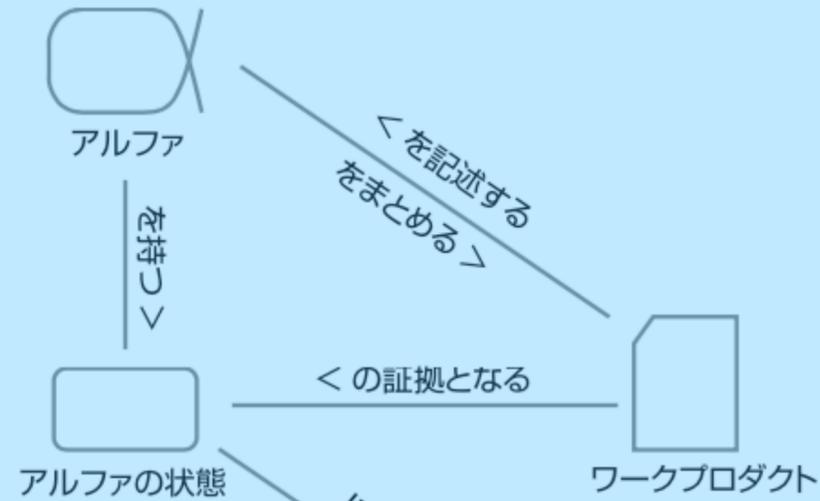
- 手法を構成する具体的な作業方法のこと
 - 手法と違って「こんななんなんぼあってもいいですからね 🍲」
- 3つの領域に影響を与える（アルファの状態を変化させる）：
 - **顧客**
 - **ソリューション**（技術）
 - **活動**（プロジェクト）

プラクティスを記述する言語

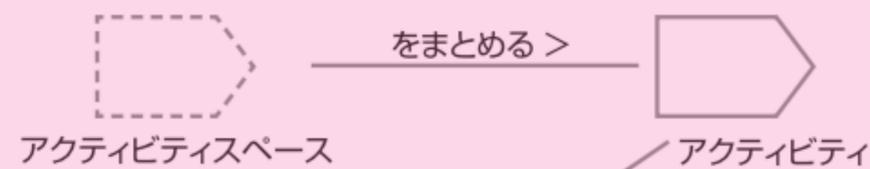


プラクティスを記述する言語

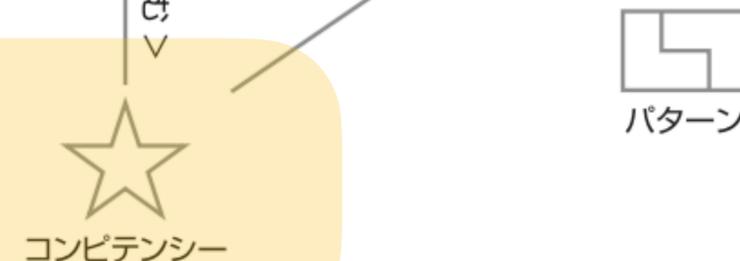
使うべき「もの」
(扱うのほうがよかった?)



やるべき「こと」



必要な「能力」



「ペアプログラミング」を記述してみた

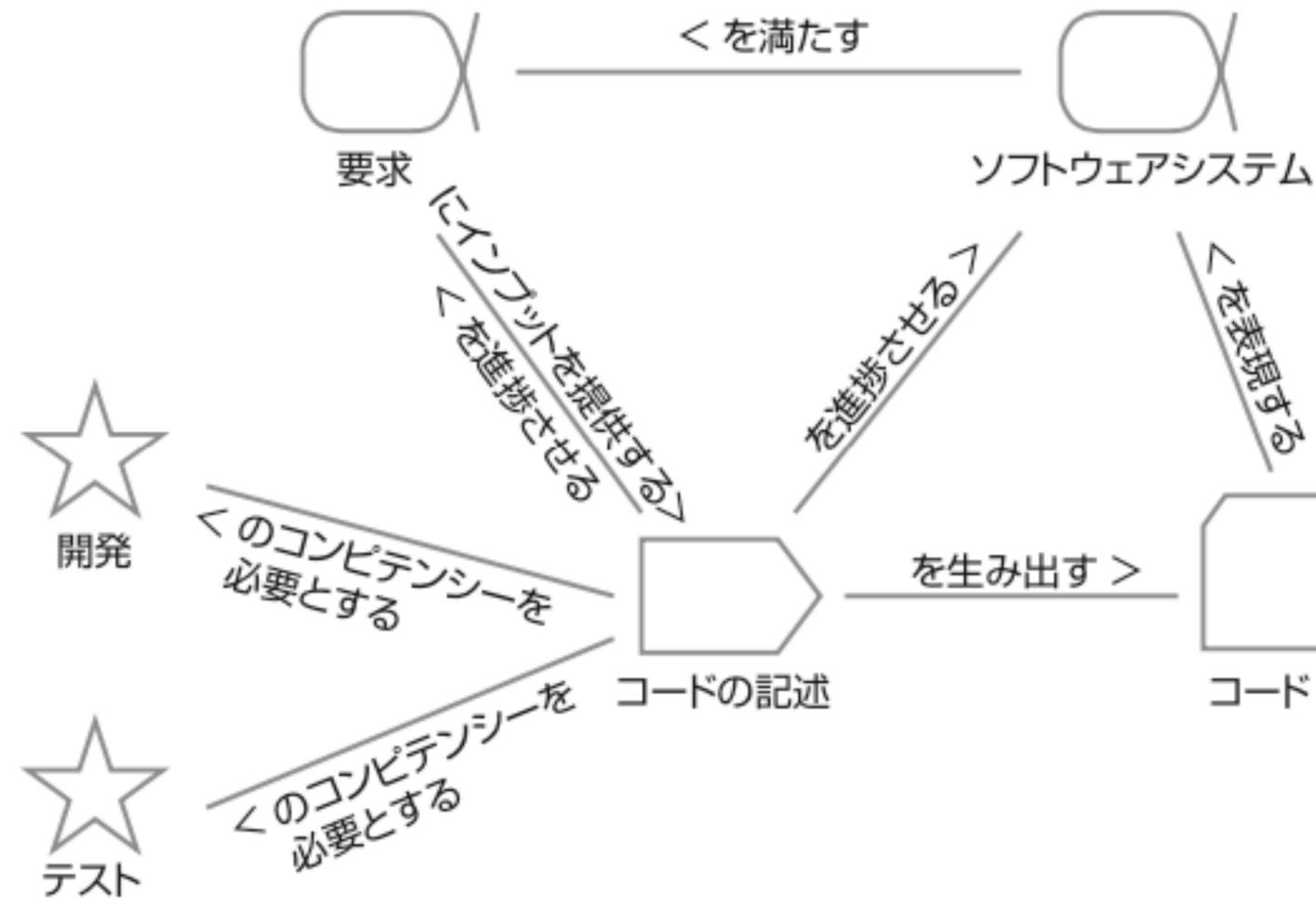


図5-1：Essence の言語で表現したペアプログラミング

「ペアプログラミング」を記述してみた

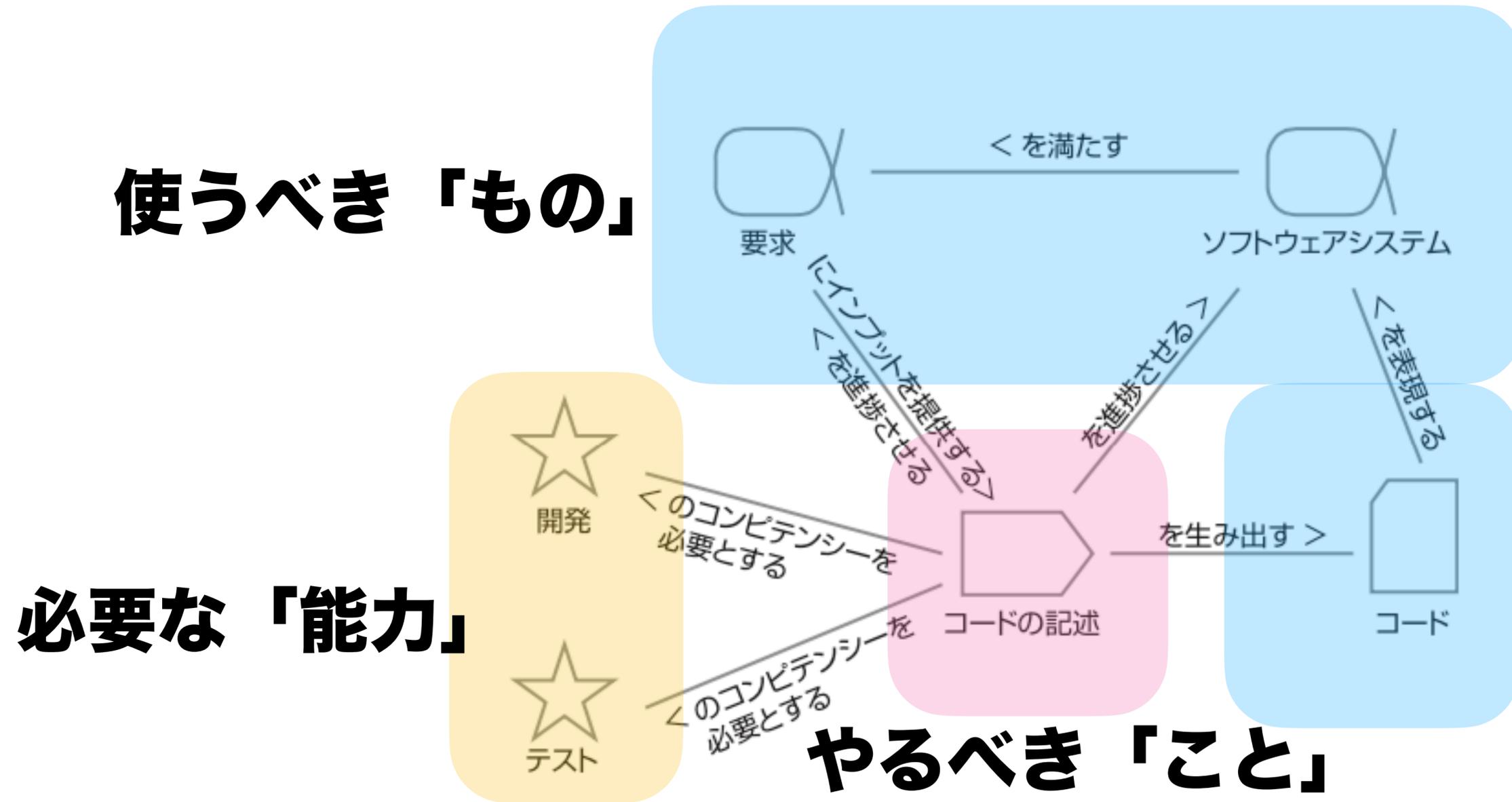


図5-1：Essence の言語で表現したペアプログラミング

素直な感想

- これって、本当にわかりやすい……のか？ 🤔
- UMLと同じくらいの希望と絶望を持つといいと思う (2回目)
- もうちょっと説明を続けます

やるべき「こと」がプラクティスの中心

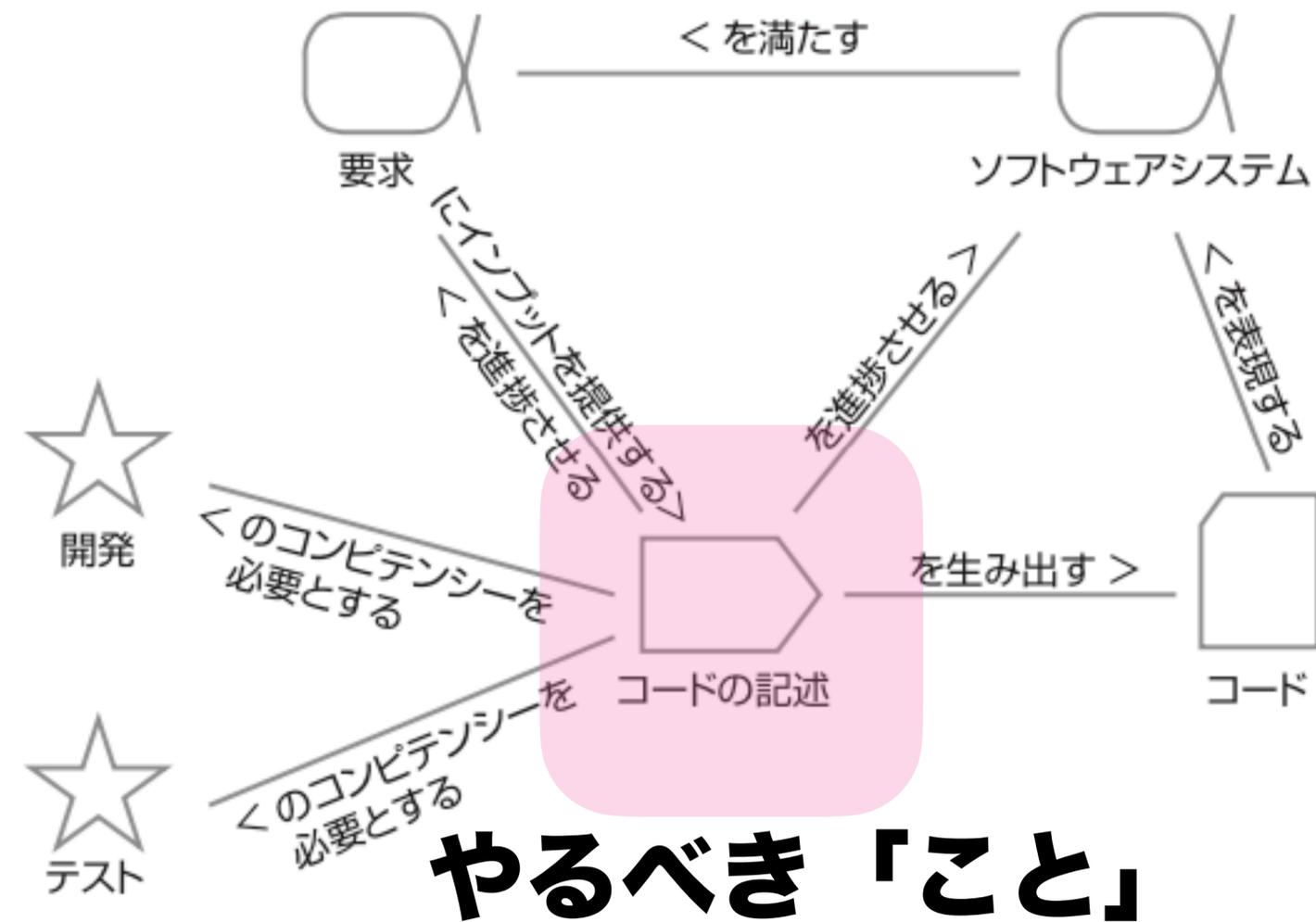
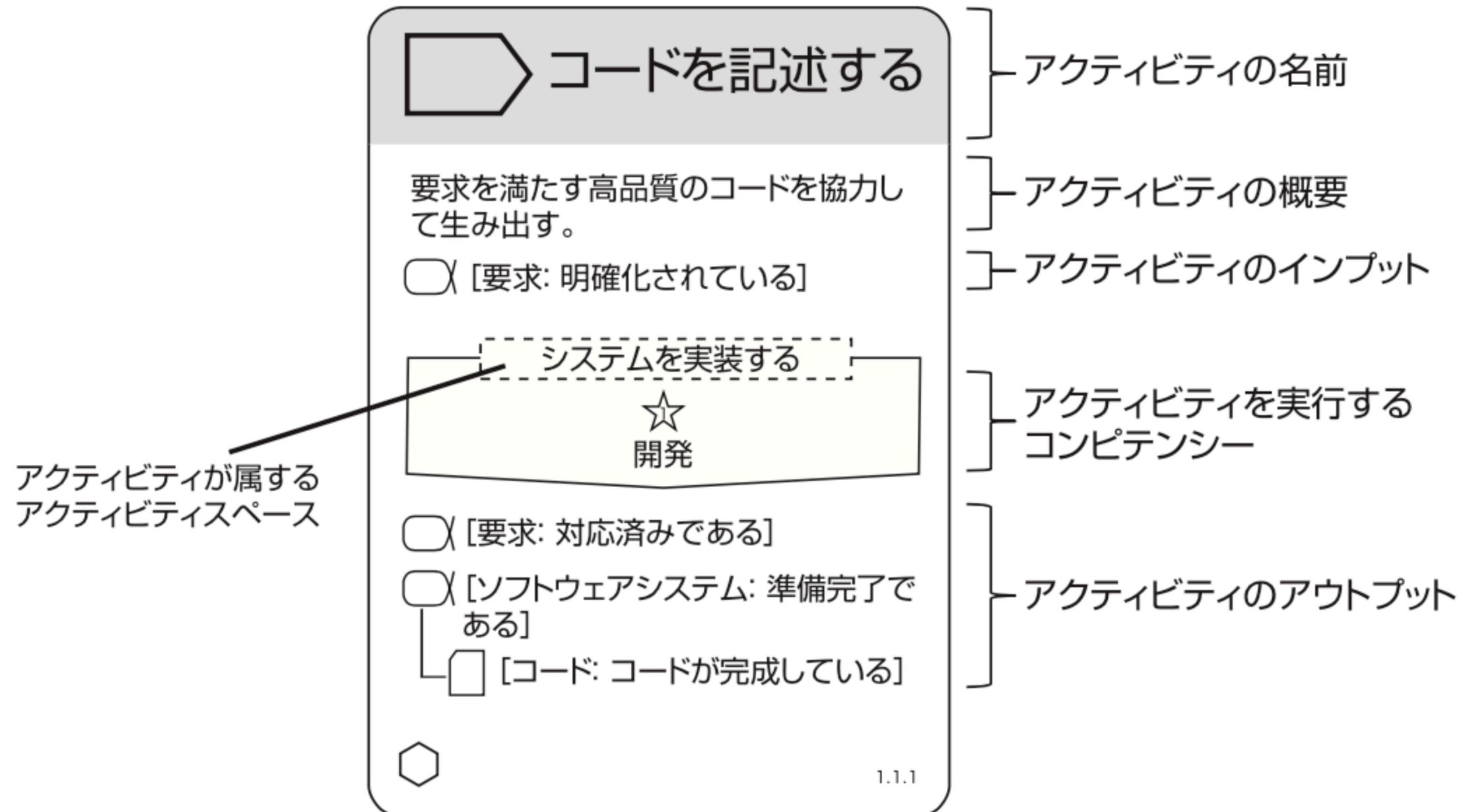
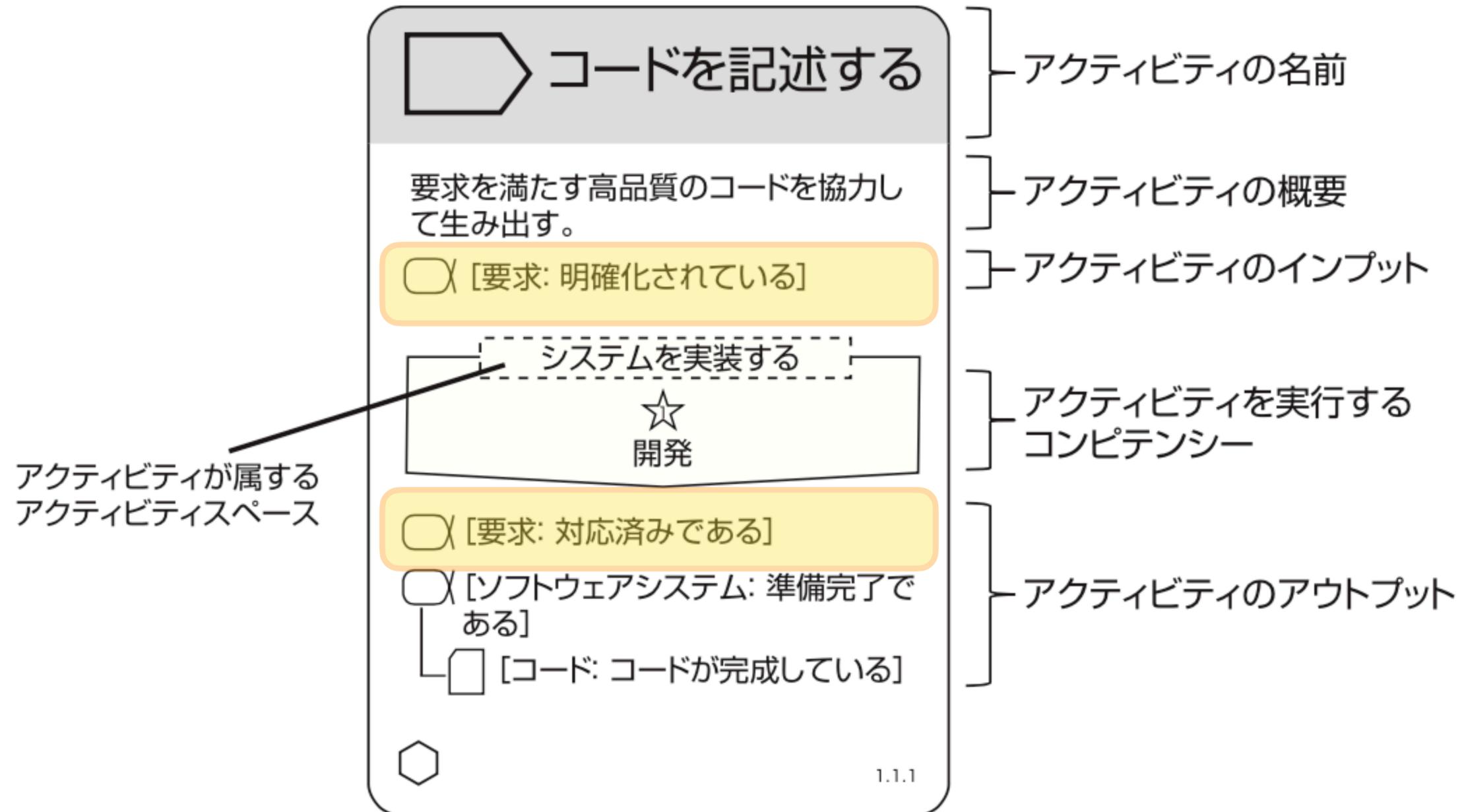


図5-1：Essence の言語で表現したペアプログラミング

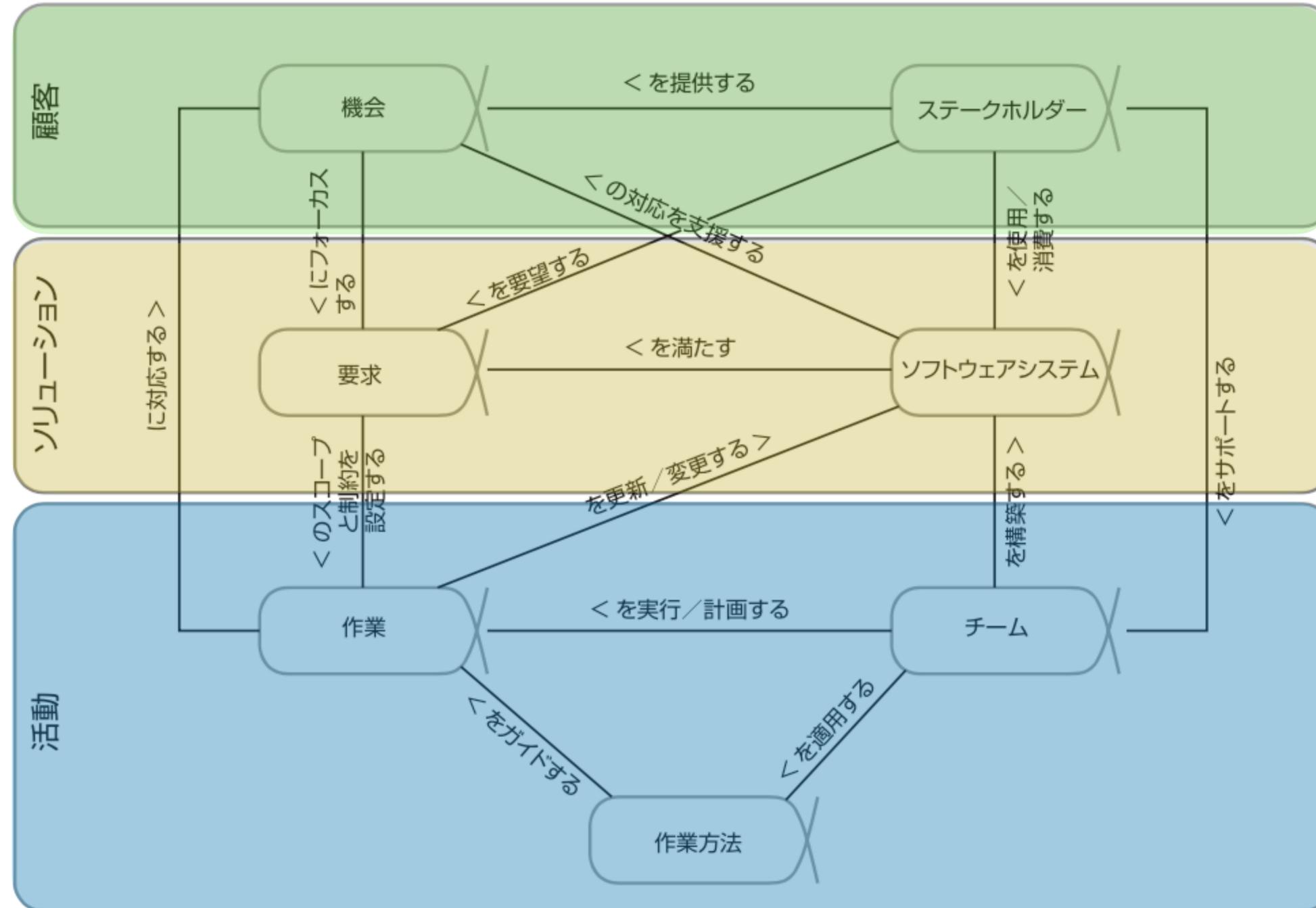
「こと」の詳細をカードに記述する



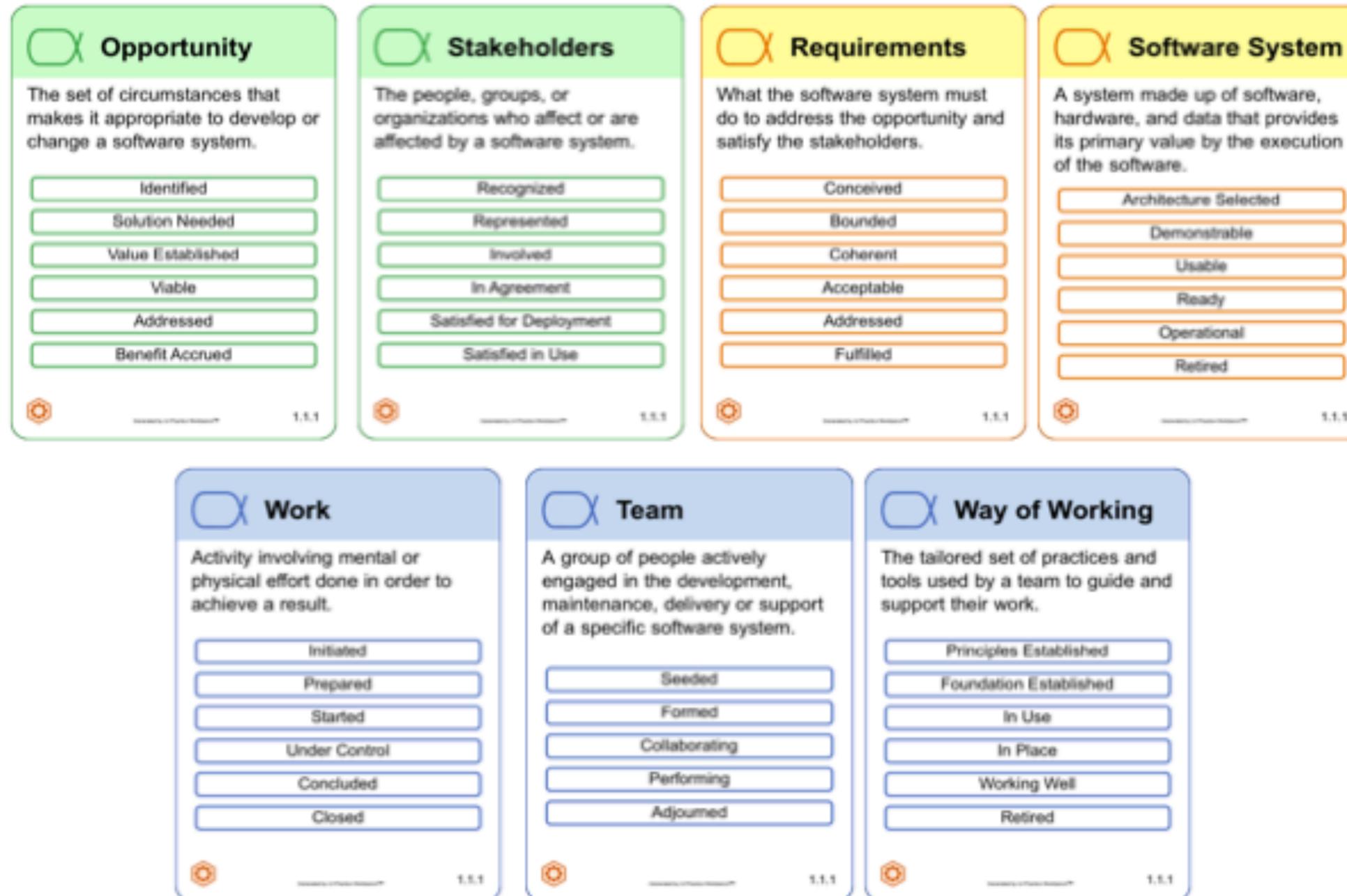
「こと」は「もの」の状態を変える



用意された7種類のアルファ (もの)



アルファはカード化されている



アルファは複数の状態を持つ

Requirements

What the software system must do to address the opportunity and satisfy the stakeholders.

Conceived

Bounded

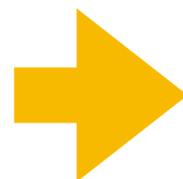
Coherent

Acceptable

Addressed

Fulfilled

Generated by U1 Proctor Workbench™ 1.1.1



Requirements

Conceived

- Stakeholders agree system is to be produced
- Users identified
- Funding stakeholders identified
- Opportunity clear

1 / 6

Generated by U1 Proctor Workbench™ 1.1.1

Requirements

Bounded

- Development stakeholders identified
- System purpose agreed
- System success clear
- Shared solution understanding exists
- Requirement's format agreed
- Requirements management in place
- Prioritization scheme clear
- Constraints identified & considered
- Assumptions clear

2 / 6

Generated by U1 Proctor Workbench™ 1.1.1

Requirements

Coherent

- Requirements shared
- Requirements' origin clear
- Rationale clear
- Conflicts addressed
- Essential characteristics clear
- Key usage scenarios explained
- Priorities clear
- Impact understood
- Team knows & agrees on what to deliver

3 / 6

Generated by U1 Proctor Workbench™ 1.1.1

Requirements

Acceptable

- Acceptable solution described
- Change under control
- Value to be realized clear
- Clear how opportunity addressed
- Testable

4 / 6

Generated by U1 Proctor Workbench™ 1.1.1

Requirements

Addressed

- Enough addressed to be acceptable
- Requirements and system match
- Value realized clear
- System worth making operational

5 / 6

Generated by U1 Proctor Workbench™ 1.1.2

Requirements

Fulfilled

- Stakeholders accept requirements
- No hindering requirements
- Requirements fully satisfied

6 / 6

Generated by U1 Proctor Workbench™ 1.1.1

素直な感想

- プラクティスによって状態を変化させるのはわかりやすい
- 状態などをカードにするのはいいアイデアだと思う！ 
- 途中からEssenceの説明に踏み込んでしまったた……。

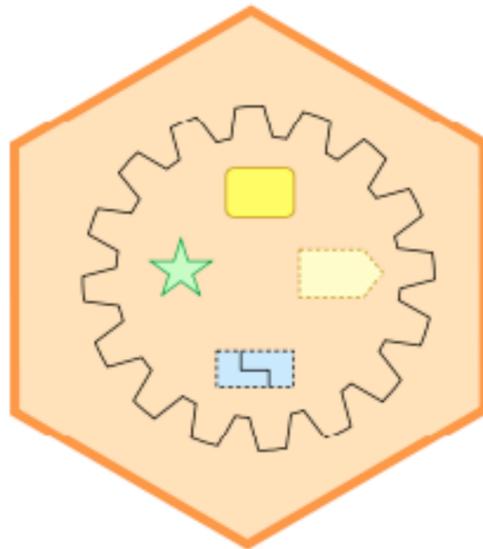
③ Essence

(カーネル + 言語)

Essenceとは？

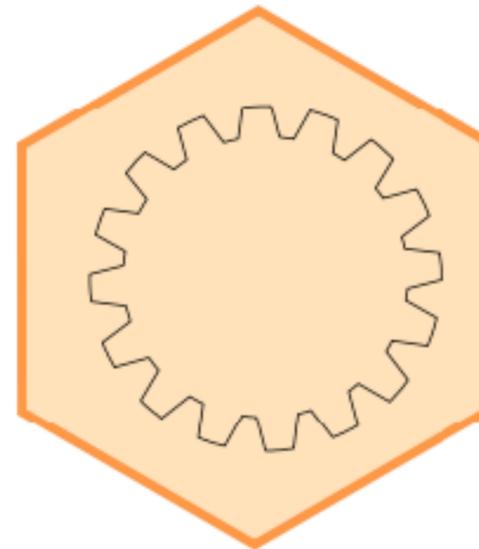
「手法の基盤」となるもの

Essence



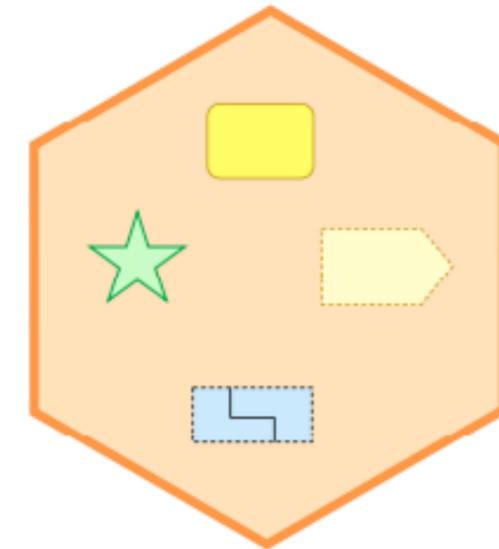
=

カーネル

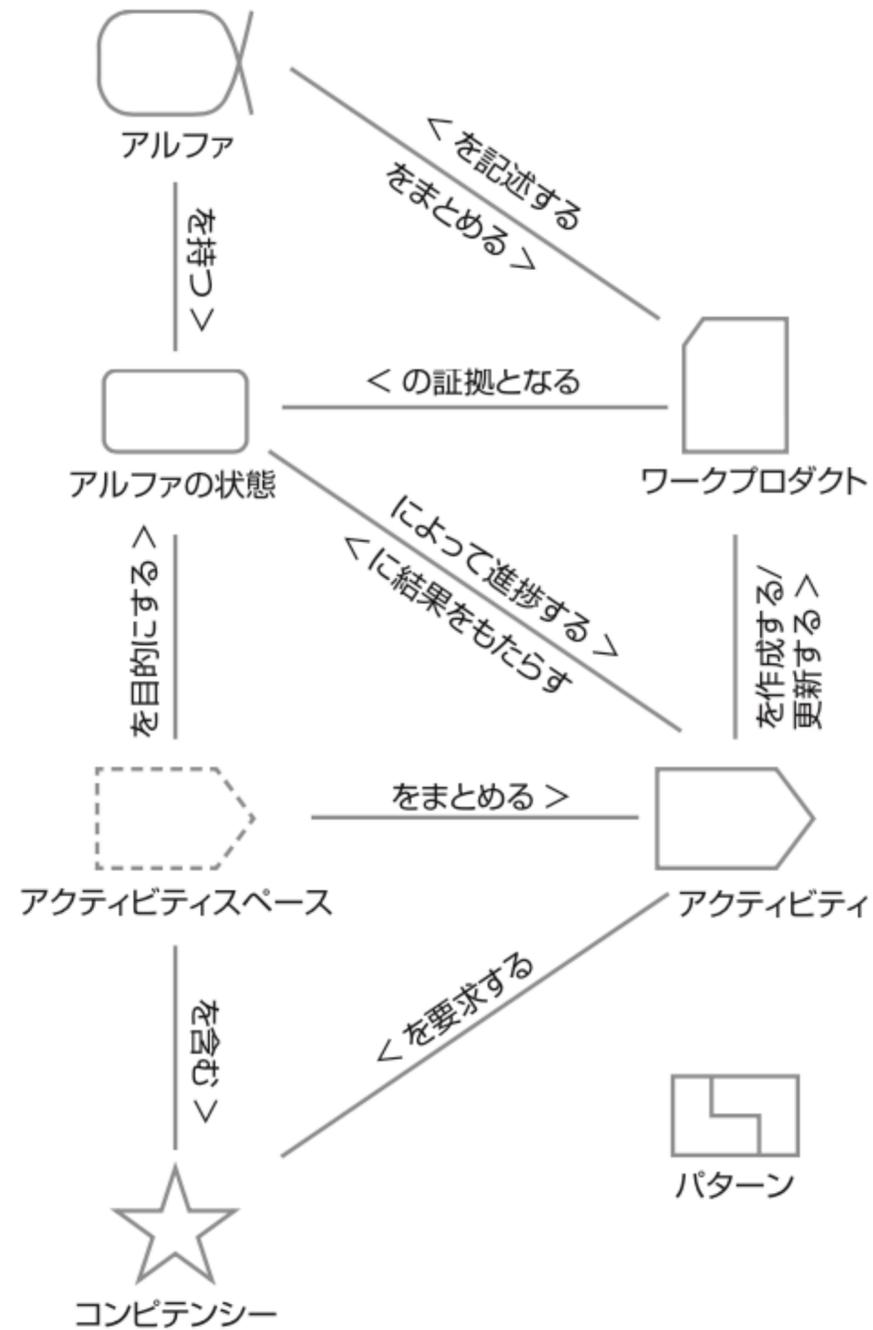


+

言語



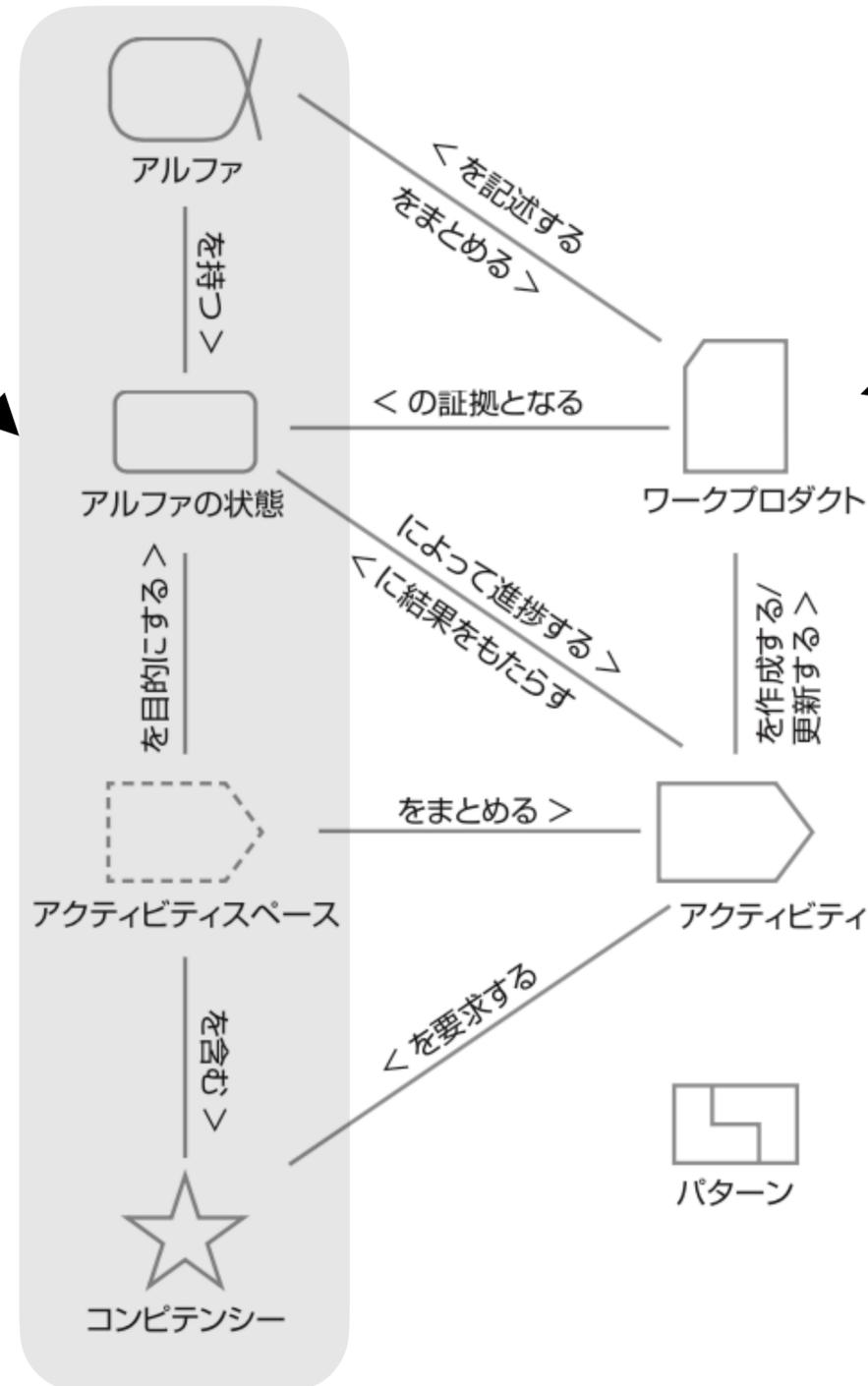
Essence言語



Essenceカーネル

ここがカーネル

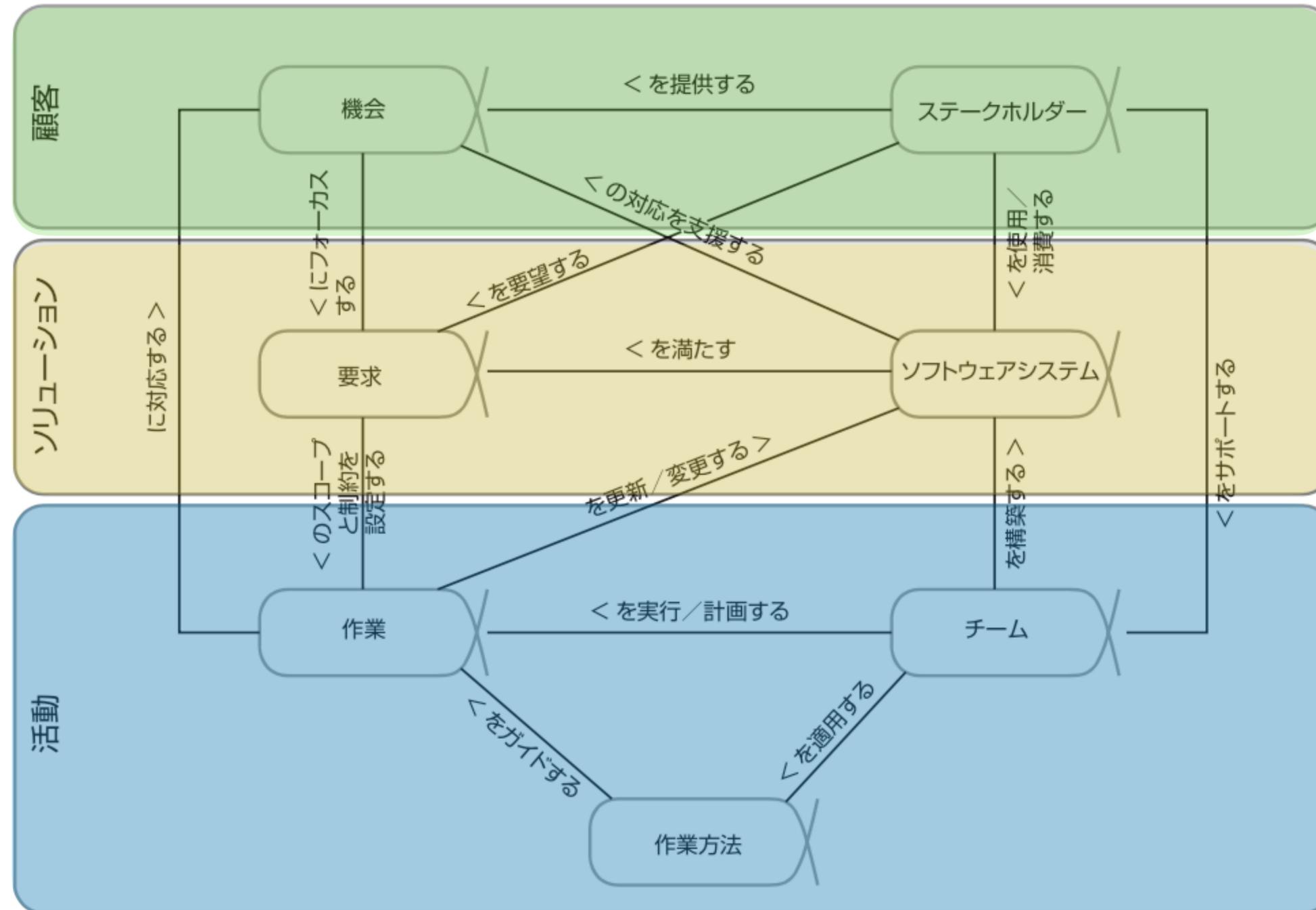
- もの
- こと
- 能力



こいつらは具体的すぎるので
カーネルに入っていない
(表現型だけ定義されている)

カーネルの アルファ (もの)

カーネルのアルファ (もの)



アルファは複数の状態を持つ

Requirements

What the software system must do to address the opportunity and satisfy the stakeholders.

Conceived

Bounded

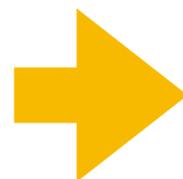
Coherent

Acceptable

Addressed

Fulfilled

Generated by U1 Proctor Workbench™ 1.1.1



Requirements

Conceived

- Stakeholders agree system is to be produced
- Users identified
- Funding stakeholders identified
- Opportunity clear

1 / 6

Generated by U1 Proctor Workbench™ 1.1.1

Requirements

Bounded

- Development stakeholders identified
- System purpose agreed
- System success clear
- Shared solution understanding exists
- Requirement's format agreed
- Requirements management in place
- Prioritization scheme clear
- Constraints identified & considered
- Assumptions clear

2 / 6

Generated by U1 Proctor Workbench™ 1.1.1

Requirements

Coherent

- Requirements shared
- Requirements' origin clear
- Rationale clear
- Conflicts addressed
- Essential characteristics clear
- Key usage scenarios explained
- Priorities clear
- Impact understood
- Team knows & agrees on what to deliver

3 / 6

Generated by U1 Proctor Workbench™ 1.1.1

Requirements

Acceptable

- Acceptable solution described
- Change under control
- Value to be realized clear
- Clear how opportunity addressed
- Testable

4 / 6

Generated by U1 Proctor Workbench™ 1.1.1

Requirements

Addressed

- Enough addressed to be acceptable
- Requirements and system match
- Value realized clear
- System worth making operational

5 / 6

Generated by U1 Proctor Workbench™ 1.1.2

Requirements

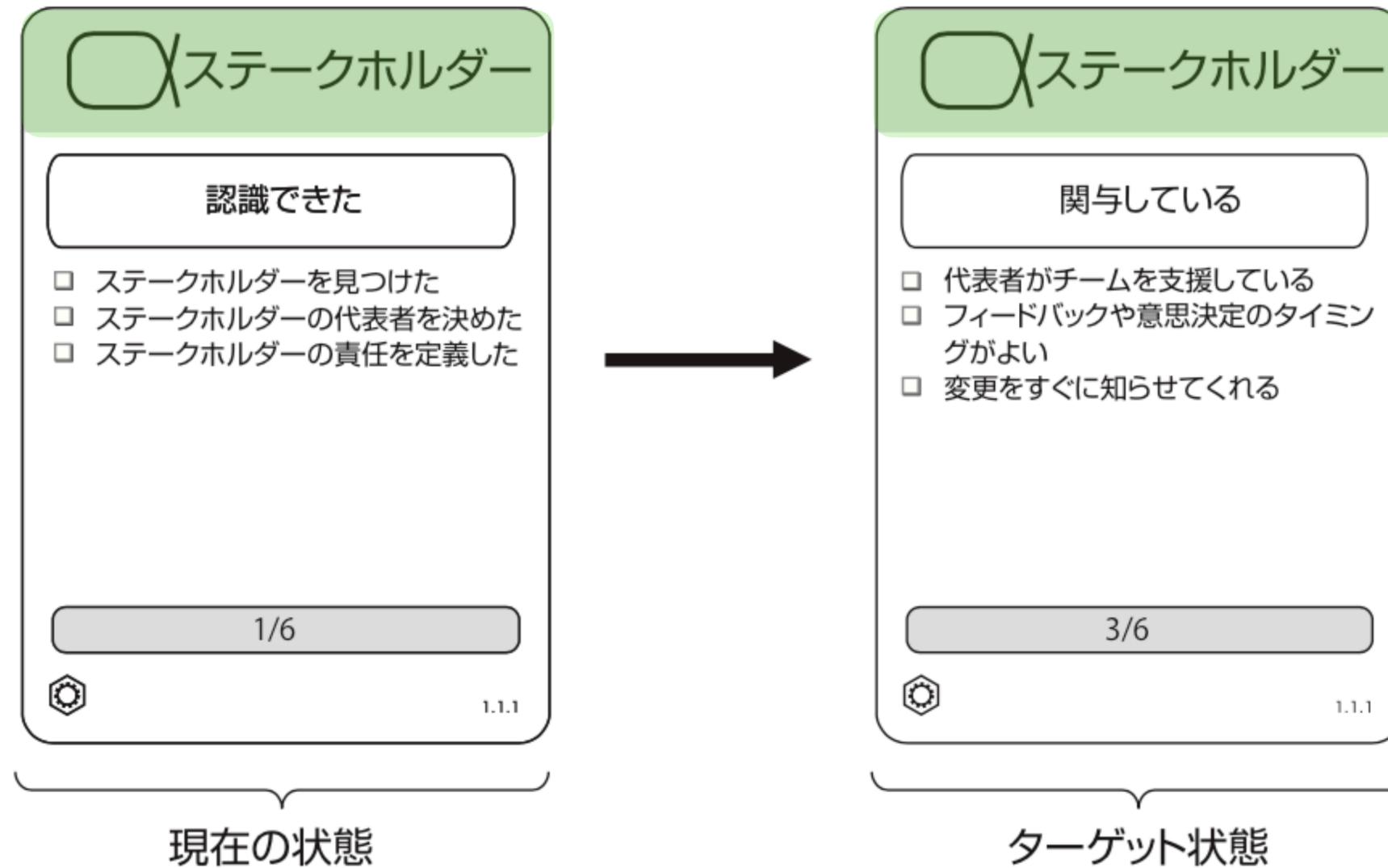
Fulfilled

- Stakeholders accept requirements
- No hindering requirements
- Requirements fully satisfied

6 / 6

Generated by U1 Proctor Workbench™ 1.1.1

状態を進捗させるために考える🧠

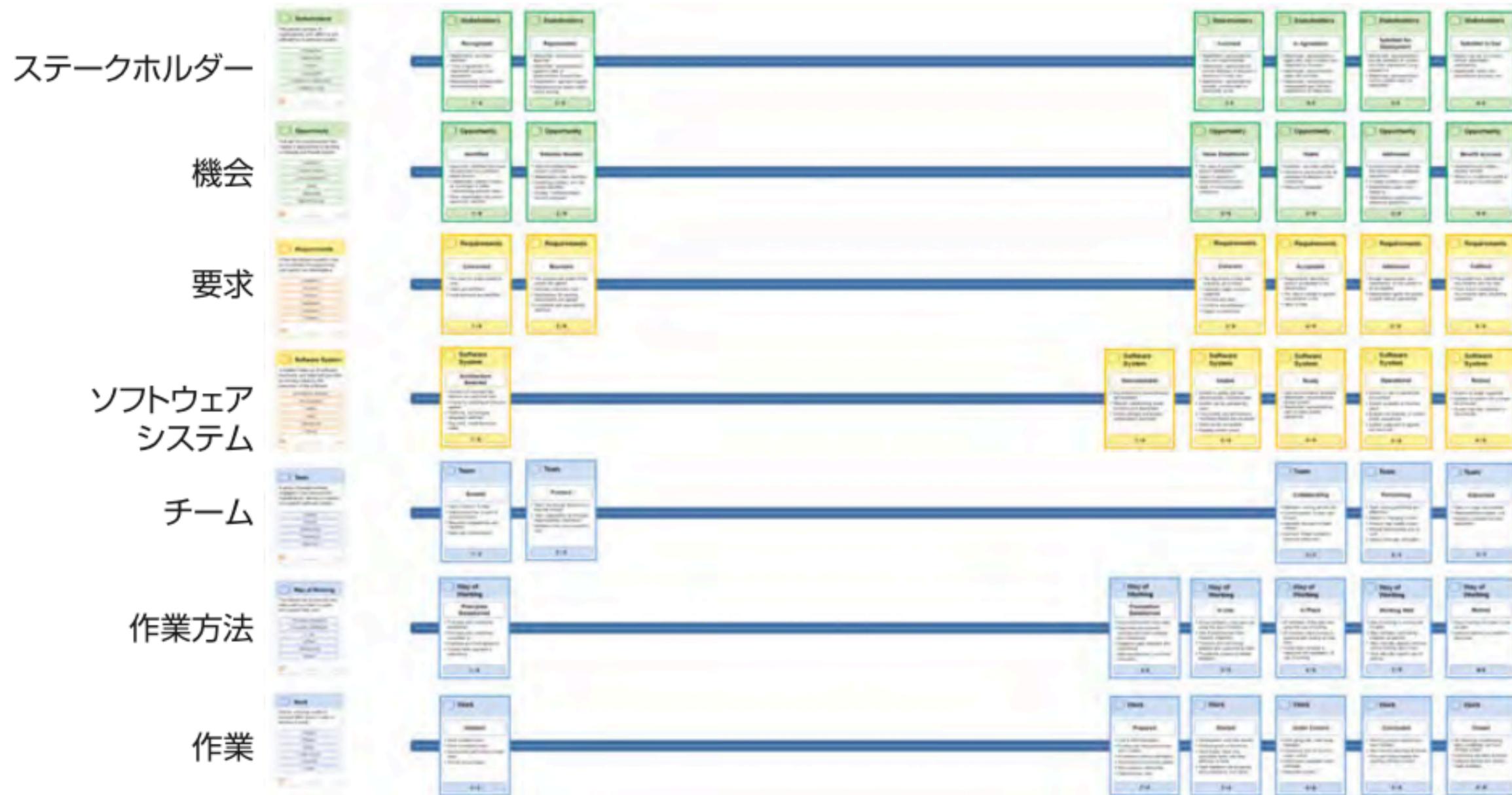


🧠 「ステークホルダーを巻き込むためのミーティングを開催してみる？」

「状態を追え」ゲーム



「状態を追え」ゲーム



状態の進捗の可視化

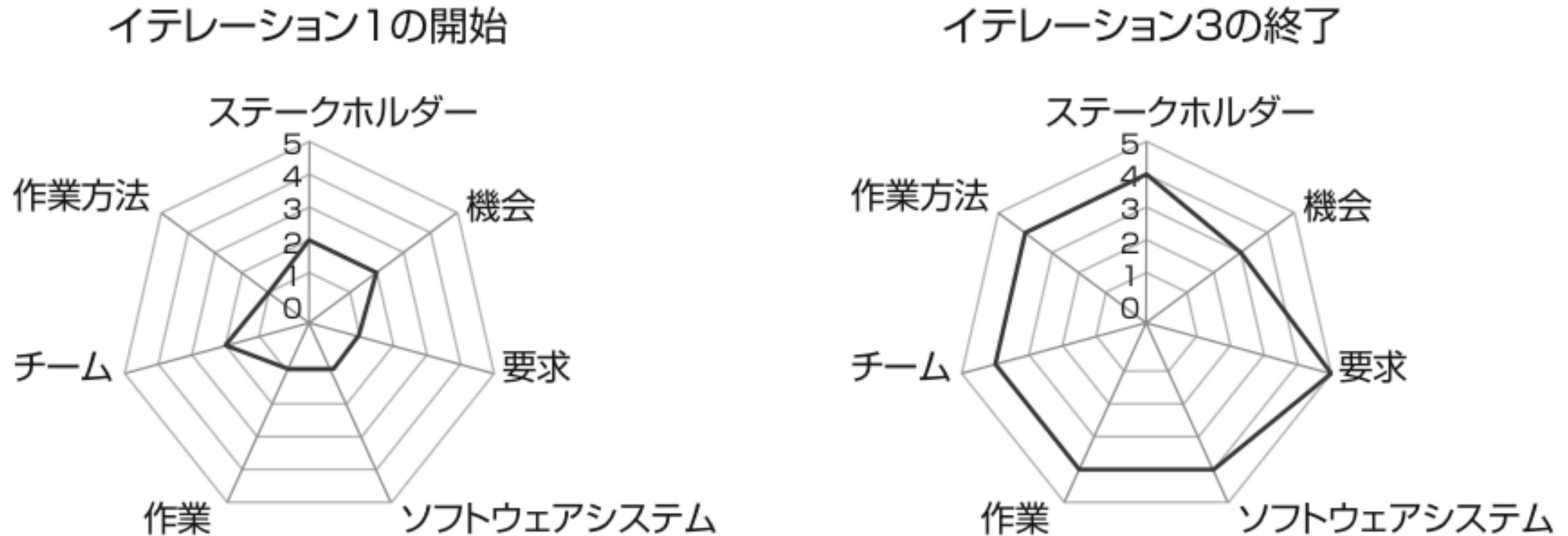


図11-2：「イテレーション1開始」と「イテレーション3終了」のレーダーチャート

(補足) 日本語のアルファ状態カードつくりました

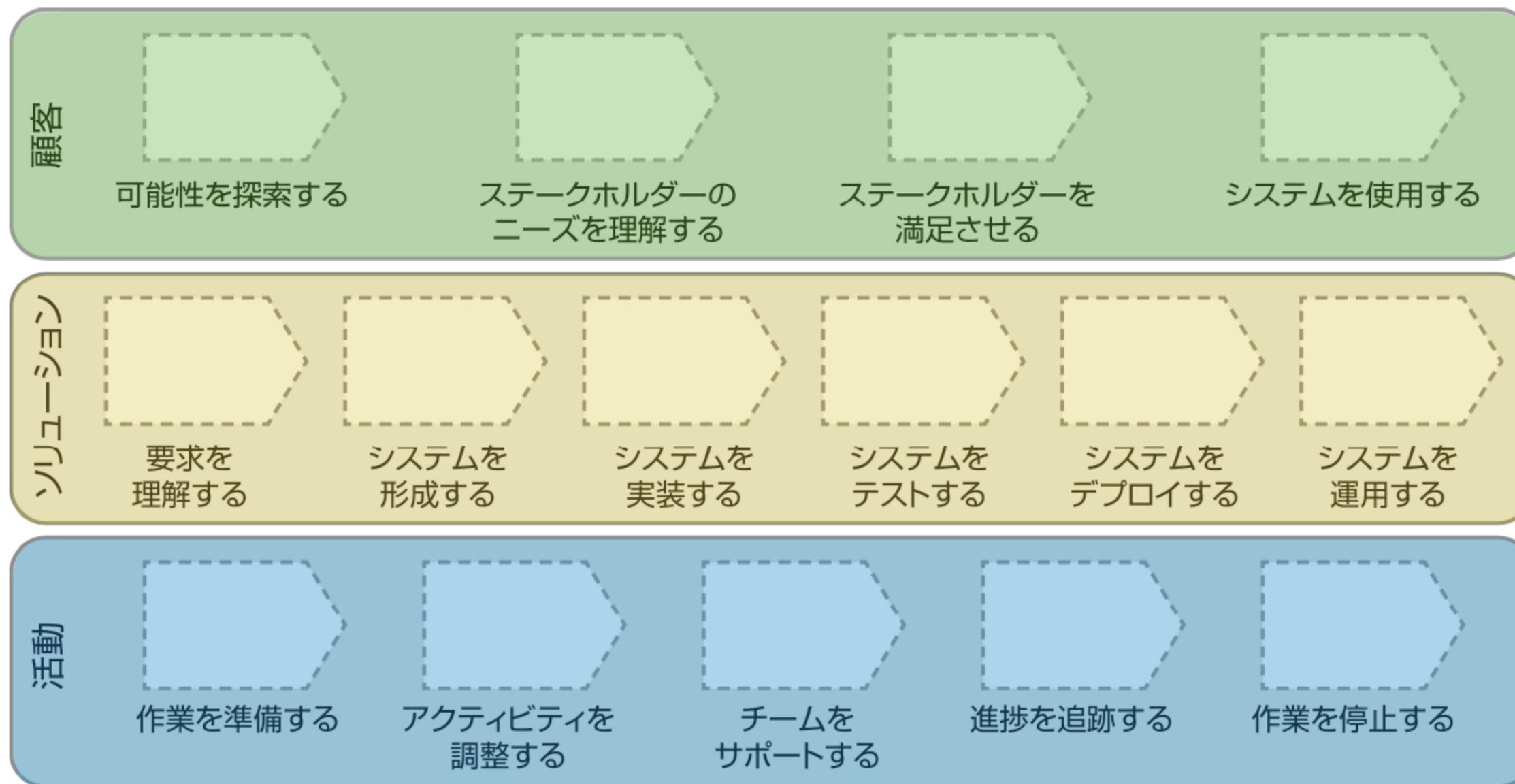
C 機会(1) 特定された <ul style="list-style-type: none"> 機会のアイデアが特定した 予算を出してくれる1人以上のステークホルダーが興味を持った 他のステークホルダーを特定した 	C 機会(2) ソリューションが必要になった <ul style="list-style-type: none"> ソリューションを特定した ステークホルダーのニーズを確定した 問題と根本原因を特定した ソリューションのニーズを確認した 少なくとも1つのソリューションが提案された 	C 機会(3) 価値が確立された <ul style="list-style-type: none"> 機会の価値が定量化された ソリューションのインパクトが理解された システムの価値が理解された 成功基準が明確になった アウトカムが明確になり定量化された 	C 機会(4) 実行可能になった <ul style="list-style-type: none"> ソリューションの概要が決まった 制約を前提としたソリューションがある リスクが受容および管理可能である ソリューションが利益につながる ソリューションの開発理由が理解された 深掘りが実行可能である 	C 機会(5) 対応済みである <ul style="list-style-type: none"> 機会に対応した ソリューションを展開する価値がある ステークホルダーが満足した 	C 機会(6) メリットが発生した <ul style="list-style-type: none"> ソリューションからメリットが生じる ROIが受け入れ可能である 	C ステークホルダー(1) 認識できた <ul style="list-style-type: none"> ステークホルダーを見つけた ステークホルダーの代表者を決めた ステークホルダーの責任を定義した
C ステークホルダー(2) 代表者がいる <ul style="list-style-type: none"> ステークホルダーの責任に同意してくれた 代表者に権限が与えられた 協力的なアプローチに同意してくれた 作業方法をサポート&尊重してくれた 	C ステークホルダー(3) 関与している <ul style="list-style-type: none"> 代表者がチームを支援している フィードバックや意思決定のタイミングがよい 変更をすぐに知らせてくれる 	C ステークホルダー(4) 合意できた <ul style="list-style-type: none"> 期待を最小限にしてくれた 代表者が関与に満足している 代表者からのインプットが重視されている チームからのインプットが重視されている 優先順位が明確&視点が安定している 	C ステークホルダー(5) デプロイに満足している <ul style="list-style-type: none"> ステークホルダーからフィードバックがある システムがデプロイ可能である 	C ステークホルダー(6) 利用に満足している <ul style="list-style-type: none"> システムの利用に関するフィードバックが利用可能である システムが期待と合致している 	S 要求(1) 企画されている <ul style="list-style-type: none"> ステークホルダーがシステムの開発に合意している ユーザーが特定されている 資金を提供するステークホルダーが特定されている 機会が明確である 	S 要求(2) 明確化されている <ul style="list-style-type: none"> 開発のステークホルダーが特定されている システムの目的が合意されている システムの成功が明確である ソリューションの共通理解が存在する ユーザーが特定されている 要求のフォーマットが合意されている 要求管理が整っている 優先順位付けの仕組みが明確である 制約が特定および検討されている 想定が明確である
S 要求(3) 論理的である <ul style="list-style-type: none"> 要求が共有されている 要求の源が明確である 論理的根拠が明確である 衝突が解消されている 必要不可欠な特徴が明確である 重要な利用シナリオが説明されている 優先順位が明確である 影響が理解されている 提供するものをチームが理解・合意している 	S 要求(4) 受け入れ可能である <ul style="list-style-type: none"> 受け入れ可能なソリューションが記述されている 変更が管理されている 実現される価値が明確である 機会の対応が明確である テスト可能である 	S 要求(5) 対応済みである <ul style="list-style-type: none"> 受け入れ可能になるまで十分に対応されている 要求とシステムが合致している 実現された価値が明確である 運用に値するシステムである 	S 要求(6) 満たされている <ul style="list-style-type: none"> ステークホルダーが要求を受け入れている 邪魔になる要求がない 要求が完全に満たされている 	S ソフトウェアシステム(1) アーキテクチャが選択済みである <ul style="list-style-type: none"> アーキテクチャの選択基準に合意した ハードウェアプラットフォームを特定した テクノロジを選択した システムの境界がわかっている システム構成を決定した 購入・構築・再利用を決定した 重要な技術的リスクに合意した 	S ソフトウェアシステム(2) デモ可能である <ul style="list-style-type: none"> 重要なアーキテクチャの特性をデモした システムを実行してパフォーマンスを測定した 重要なハードウェア構成をデモした 重要なインターフェイスをデモした 環境との統合をデモした アーキテクチャが目的に合っていると受け入れられた 	S ソフトウェアシステム(3) 使用可能である <ul style="list-style-type: none"> システムは運用可能である システムの機能はテスト済みである システムのパフォーマンスは受け入れ可能である 欠陥レベルは受け入れ可能である システムが完全に文書化されている リリース内容が周知されている 追加された価値が明確である
S ソフトウェアシステム(4) 準備完了である <ul style="list-style-type: none"> ユーザー文書が利用可能である システムが目的に合致している ステークホルダーが必要としている 運用サポートの準備できている 	S ソフトウェアシステム(5) 運用可能である <ul style="list-style-type: none"> システムが稼働可能である システムが稼働している 合意されたサービスレベルをサポートしている 	S ソフトウェアシステム(6) 廃止済みである <ul style="list-style-type: none"> リプレイスまたは破壊された システムが終了した 権限のあるユーザーがいない 更新を停止した 	E 作業(1) 開始された <ul style="list-style-type: none"> 求められる結果が明確である 制約が明確である 予算のステークホルダーがわかっている 起業者が特定された 受け入れるステークホルダーがわかっている 予算の源が明確である 優先順位が明確である 	E 作業(2) 準備できた <ul style="list-style-type: none"> コミットされている コストと努力が見積もられている 利用可能なリソースがわかっている リスクの影響度が理解されている 受け入れ基準が設定されている 着手できるほど十分に分割されている タスクが特定されて優先順位がつけられている 依頼できる計画がある 予算が用意されている 少なくとも1人のチームメンバーが確保できている 統合地点が定義されている 	E 作業(3) 着手した <ul style="list-style-type: none"> 開発が開始された 進捗を測定している 完成の定義が設定されている タスクが進行中である 	E 作業(4) 制御中である <ul style="list-style-type: none"> タスクが完了した 計画外の作業が制御されている リスクが制御されている パフォーマンスを反映して再見積もりした 進捗が計画されている 再作業が制御されている コミットメントが一貫して満たされている
E 作業(5) 完了した <ul style="list-style-type: none"> 管理者の作業のみが残っている 結果が達成された 最終的なシステムが受け入れられた 	E 作業(6) 終了した <ul style="list-style-type: none"> 学びがあった 指標が利用可能である すべてが達成された 予算を調整して終了した チームが解散された uncompleted tasks 	E チーム(1) 集められた <ul style="list-style-type: none"> ミッションが定義された 制約が判明および定義されている 成長の仕組みが整っている 編成が定義された 責任の概要が決まった 必要となるコミットメントのレベルが明確である 必要となるコンピテンシーが特定された 候補が決まった ガバナンスルールが定義された リーダーシップモデルが選択された 	E チーム(2) 形成された <ul style="list-style-type: none"> 十分なメンバーをリクルートした 役割を理解した 仕事のやり方を理解した メンバーを紹介した 個人の能力に合った責任を受け入れた メンバーが作業を受け入れた 外部のコラボレーターが特定された コミュニケーションの仕組みが定義されている メンバーがチームにコミットしている 	E チーム(3) コラボレーションしている <ul style="list-style-type: none"> チームとして作業している オープンで正直にコミュニケーションしている ミッションにフォーカスしている メンバーはお互いによく知っている 	E チーム(4) 機能している <ul style="list-style-type: none"> 一貫してコミットメントを満たしている 継続的に変化に适应している 問題に対処している 得意や進捗が最小限である 継続的にムダを排除している 	E チーム(5) 解散した <ul style="list-style-type: none"> 責任が果たされた メンバーが他のチームで利用可能である ミッションが終了した
E 作業方法(1) 原則を作った <ul style="list-style-type: none"> チームが積極的に原則をサポートしている ステークホルダーが原則に合意している ツールの合意が必要である アプローチが推奨されている 運用の背景が理解されている プラクティスとツールの制約がわかっている 	E 作業方法(2) 基盤を作った <ul style="list-style-type: none"> プラクティスとツールを選択した 作業開始に必要なプラクティスが合意された 必須のプラクティスとツールが特定された 利用可能な作業方法と必要な作業方法のギャップが理解された 能力のギャップが理解された 作業方法の統合が可能である 	E 作業方法(3) 利用中である <ul style="list-style-type: none"> プラクティスとツールを使用中である 定期的な検査している 状況に适应している チームが賛同している フィードバックの仕組みがある プラクティスとツールがコラボレーションをサポートしている 	E 作業方法(4) 実施中である <ul style="list-style-type: none"> チーム全体で使用している チーム全体で利用可能である チーム全体で検査と適応している 	E 作業方法(5) 順調である <ul style="list-style-type: none"> 進捗が予測可能である 自然にプラクティスを適用している 自然にツールに支えられている 継続的に調整している 	E 作業方法(6) 廃止済みである <ul style="list-style-type: none"> すでに使用していない 教訓が共有されている 	

<https://github.com/kdmsnr/essence-alpha-state-cards-ja>

カーネルの アクティビティスペース (こと)

アクティビティスペース（こと）

正確にはプラクティスのアクティビティ（こと）を入れる「ことの入れ物」



プラクティスが足りないところがある

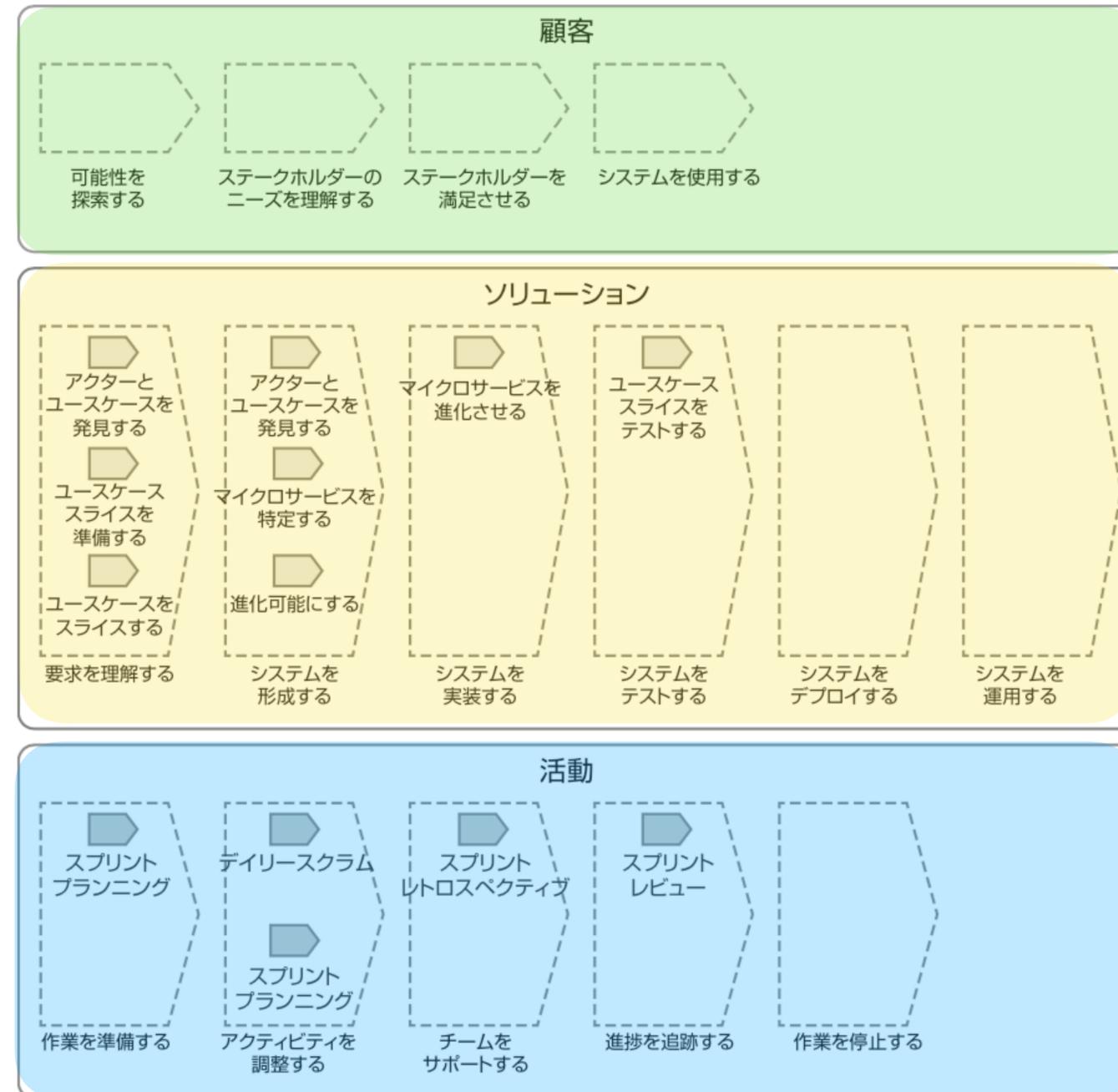


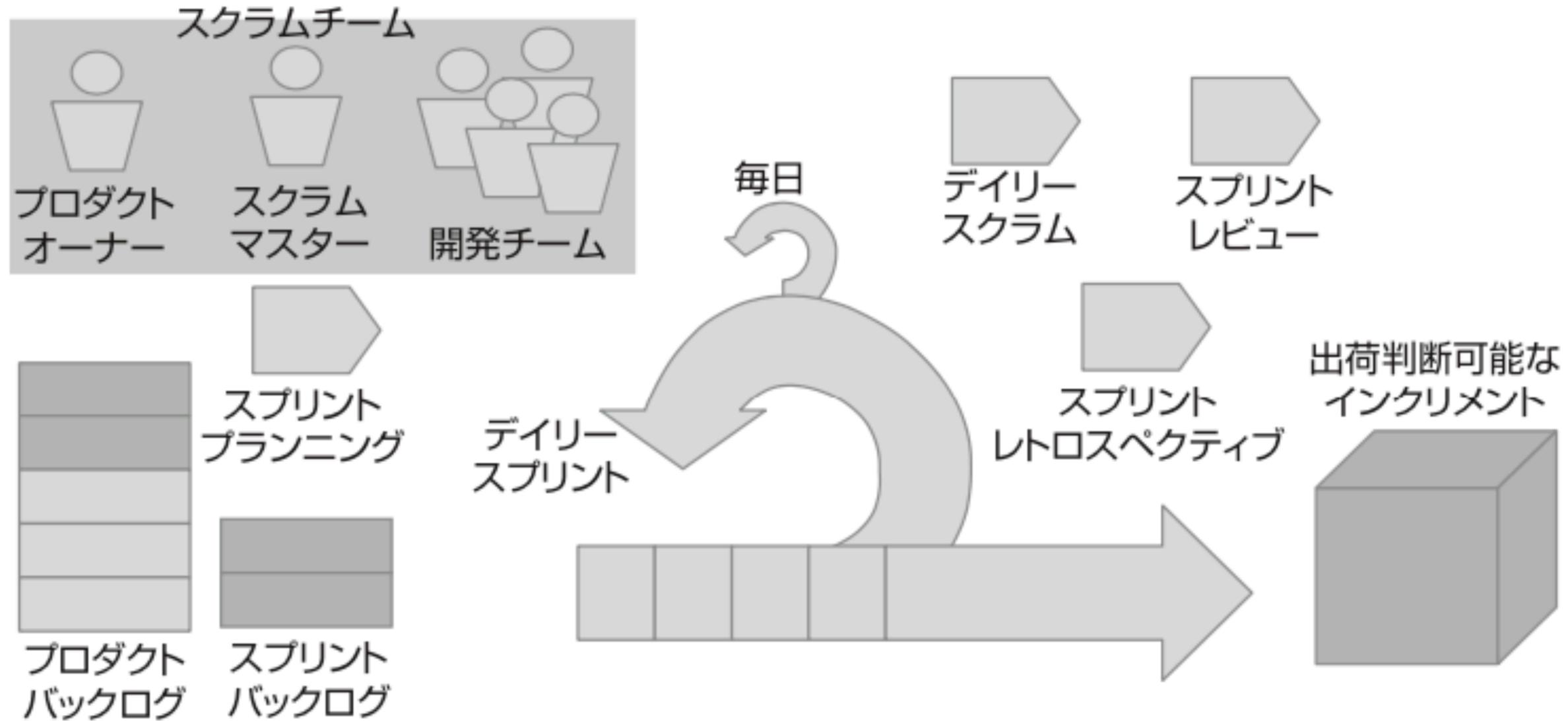
図18-4：スミスのプラクティスのアクティビティで埋められたアクティビティスペース

(補足) カーネルでは足りないとき

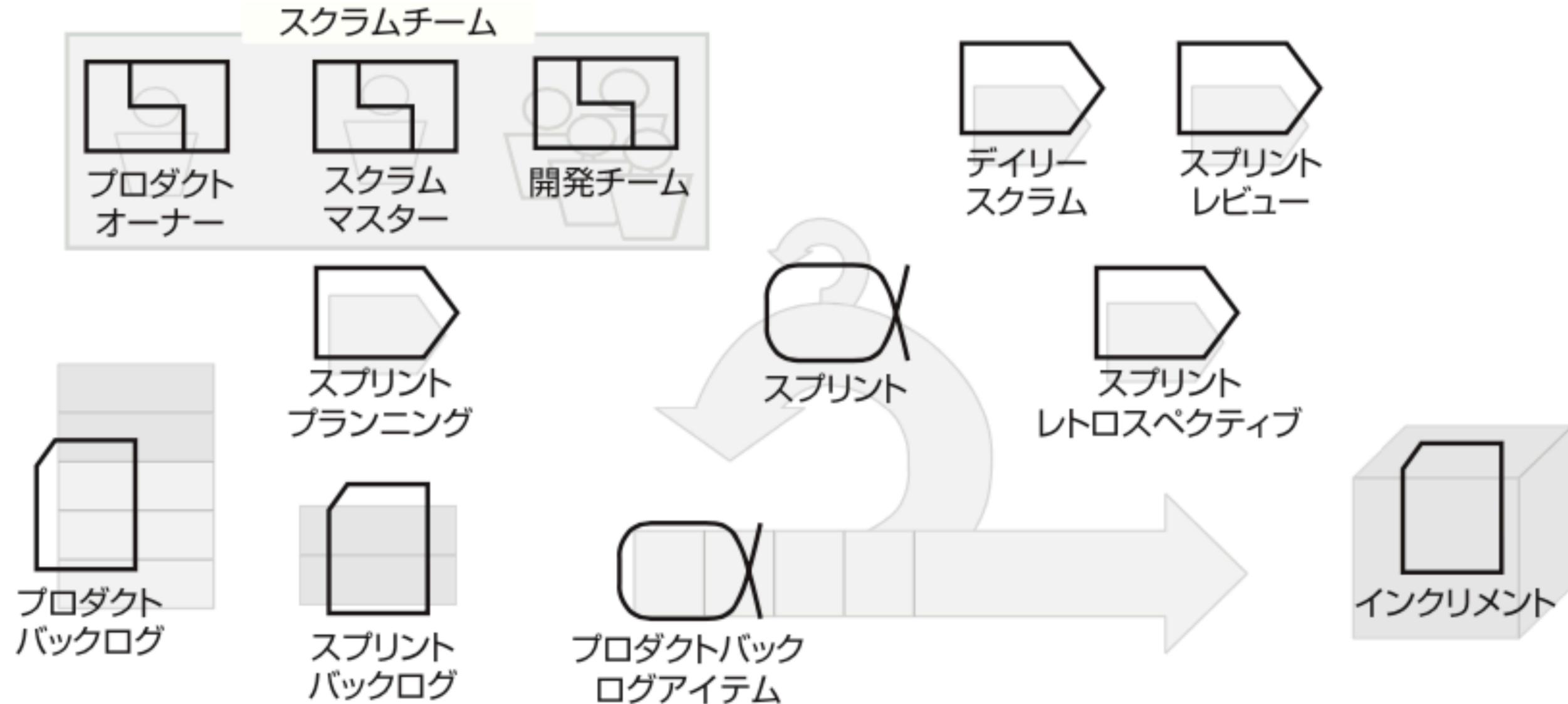
- カーネルにあるアルファ (もの) やアクティビティスペース (こと) では不十分なことがある
- プラクティスの作成時にカーネルを継承して拡張すればいい
 - e.g. [要求] を継承した [プロダクトバックログアイテム]
 - e.g. [作業を準備する] を継承した [チームのキックオフ]

スクラムとEssence

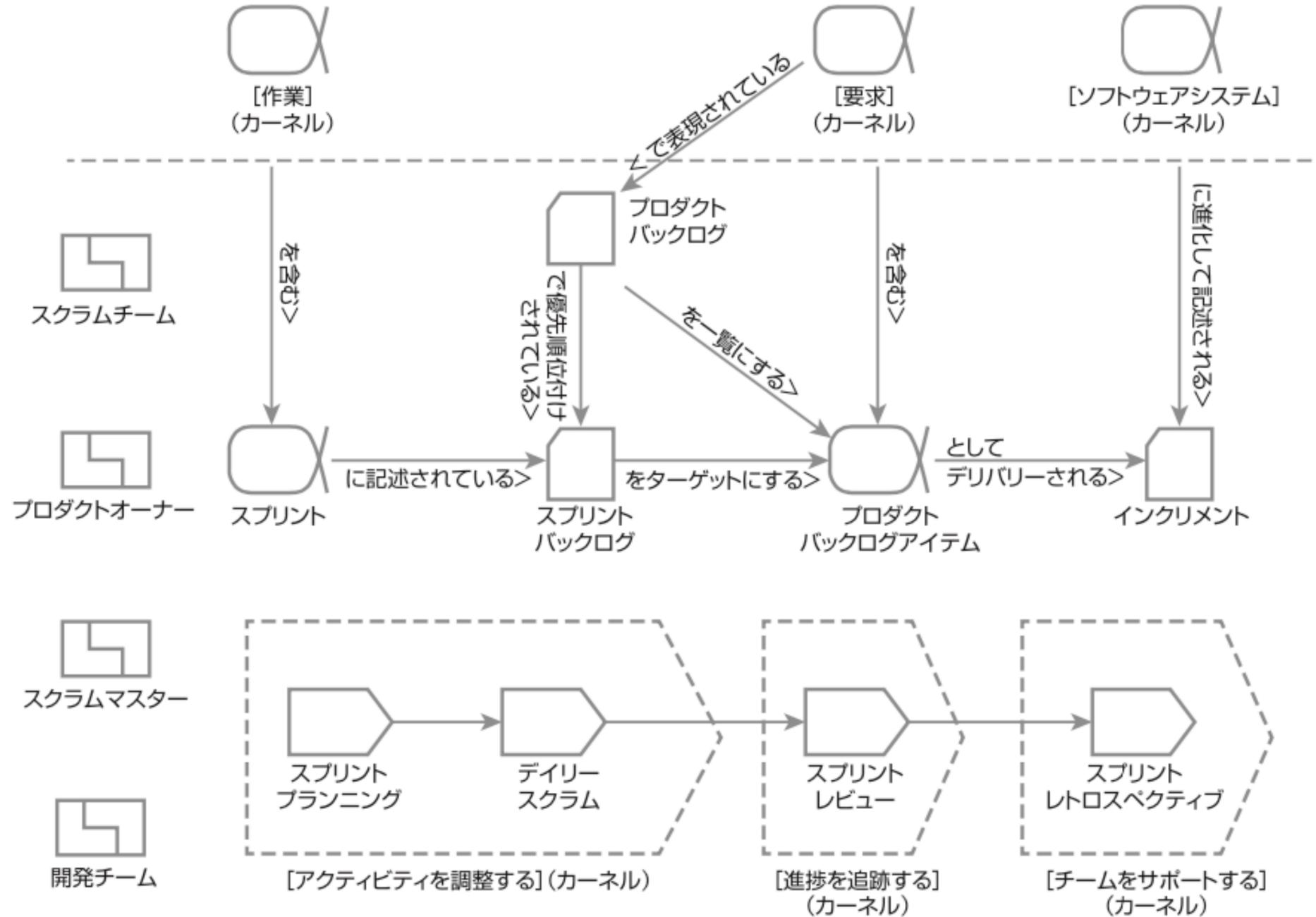
スクラムの全体像



Essence言語にマッピング



Essence言語で表現したもの



素直な感想

- えっ、めっちゃわかりにくくなって……ない？ 🤔
- UMLと同じくらいの希望と絶望を持つといいと思う (3回目)
- もうちょっと説明を続けます

カード化されたのは👍

スプリント

1か月以下のタイムボックス。その期間内に、「完成」した、利用可能な、出荷判断可能なインクリメントを作成する。スプリントが終わったら、すぐに新しいスプリントを開始する。

スケジュール済み

計画済み

レビュー済み

関連する: [作業]

03.2020

状態が明示的になったのは👍

📌 スプリント

1か月以下のタイムボックス。その期間内に、「完成」した、利用可能な、出荷判断可能なインクリメントを作成する。スプリントが終わったら、すぐに新しいスプリントを開始する。

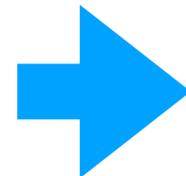
スケジュール済み

計画済み

レビュー済み

関連する: 📌 [作業]

📅 03.2020



📌 スプリント

スケジュール済み

- ❑ 次のイテレーションが予定されている
- ❑ バックログアイテムが優先順位づけされている
- ❑ 十分なバックログアイテムが計画準備完了している

1/3

📅 03.2020

📌 スプリント

計画済み

- ❑ イテレーションのゴールが合意されている
- ❑ バックログアイテムが合意されている
- ❑ 主要なリスクが特定されている
- ❑ 十分なバックログアイテムが開発準備完了である

2/3

📅 03.2020

📌 スプリント

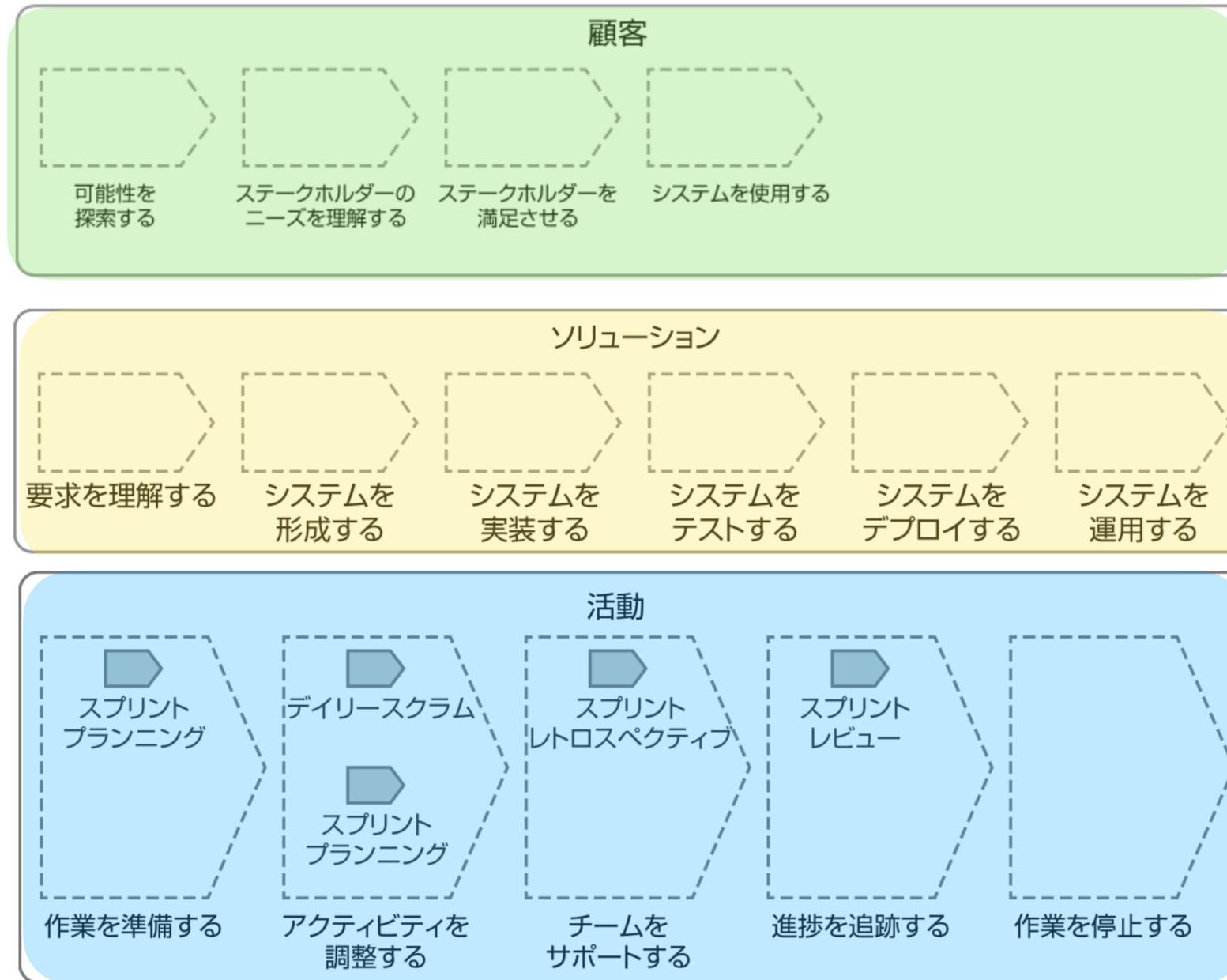
レビュー済み

- ❑ 完成したバックログアイテムがレビュー済みである
- ❑ 未完了のバックログアイテムが対応済みである
- ❑ 改善アクションが計画されている

3/3

📅 03.2020

スクラムだけじゃ不十分なこともわかる👍



([活動] 領域だけじゃない気もするけど、不十分なのは同意)

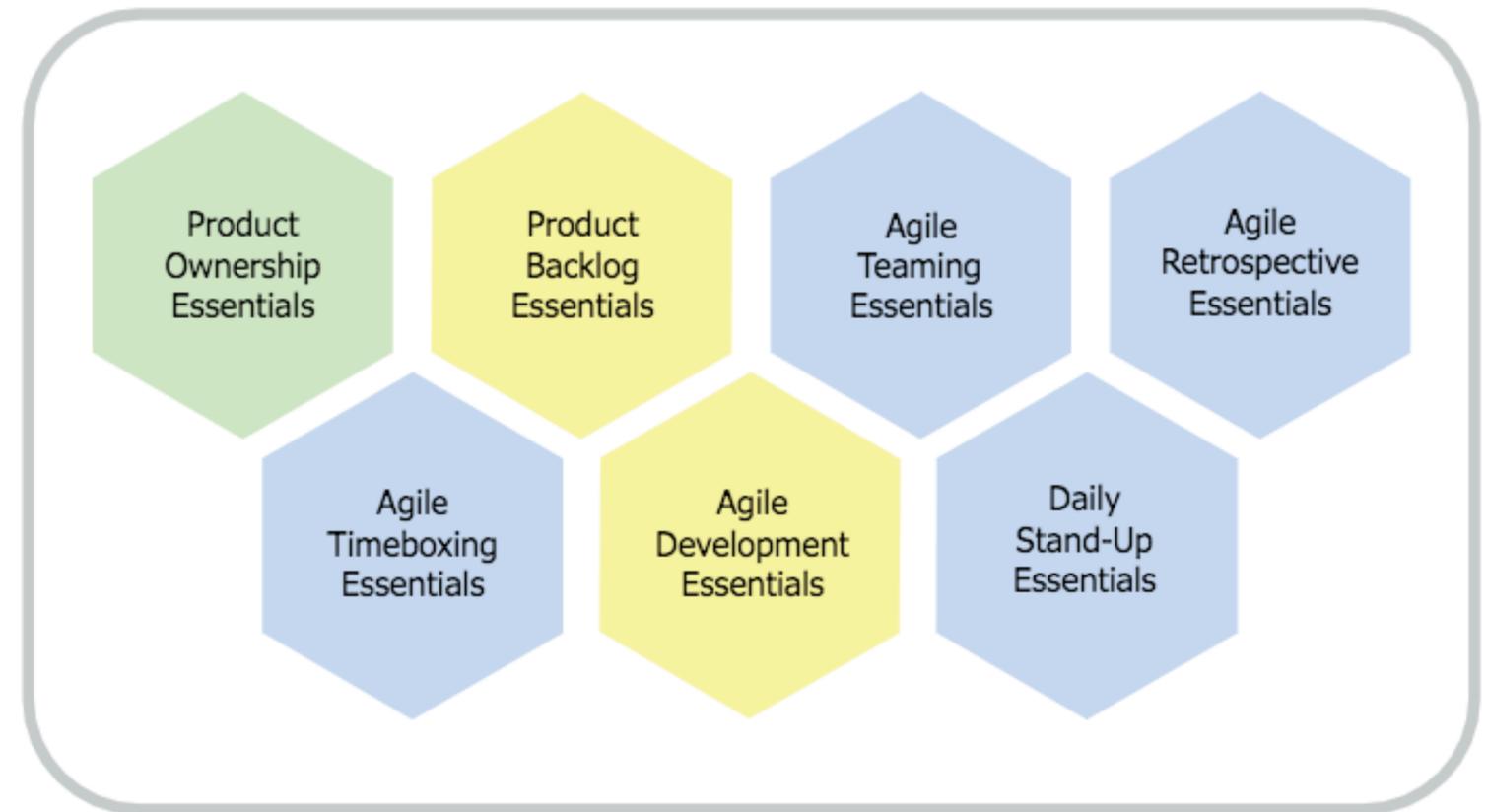
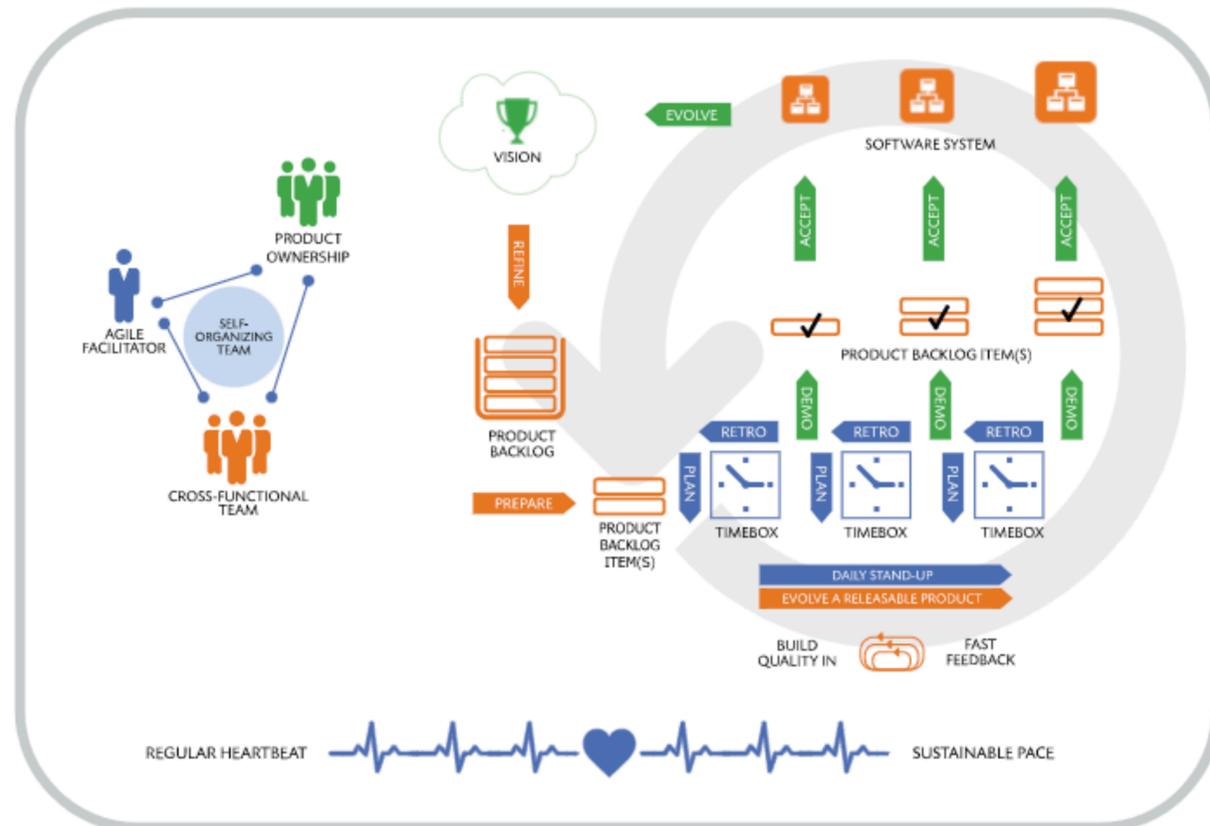
スクラムの58%は失敗してる😱

- Essenceのカードを使えば、どこで失敗しているかがわかる
- 次の「状態」に進むためには何をすべきか？ を考えられる



<https://pages.services/ss.ivarjacobson.com/essential-scrum>

(補足) Agile Essentials もある



<https://practicelibrary.ivarjacobson.com/>

全体的な印象

- アジャイル開発の「楽しさ」は大幅ダウンしてる気がするね🙄
- 「すでに流行ってる」そうだが、全然そんな気がしない...🙄
- あれほど避けていた「手法の監獄」感が出てない！？🙄
- UMLと同じくらいの希望と絶望を持つといいと思う（4回目）🙄
- でも、開発がうまくいってないときに使うのはよさそう😊
- 「手法の基盤」となる安定感はなんとなく感じられる😊
- ヤコブソンが元気そうでよかったです😊

開発がうまくいってない方は是非どうぞ！

