伊藤忠テクノソリューションズ株式会社様

運用設計導入コース (第2回)

運用課題の分析

本編

運用設計ラーニング

2023-01-11

本編 運用課題の分析

運用課題の分析に必要なこと

講師が「運用あるある」で気付いたこと (第1回)

問題点1: **「課題」への対応を「点」に対して行なっていた**

問題点2: **過去の延長線上で課題を解決しようとしていた**

問題点3: **全ての「課題」が発生後の対応になっていた**

分析ポイント1:「課題」を「面」で捉えて分析

問題点1:**「課題」への対応を「点」に対して行なっていた**

「課題」を**「面」で捉えて**分析する。

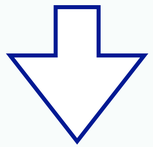
過去の解決事例を活用し、**隣接業務や類似業務も含めて分析**する。

課題の**収集**と**蓄積**

分析ポイント2: 「課題」を設計で解決することを前提に分析

問題点2: **過去の延長線上で課題を解決しようとしていた**

「課題」は**「設計で解決」**(根本解決)**することを前提に**分析する。



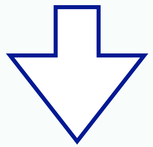
「課題」は原則として**設計にフィードバック**し、**設計側で解決**する。

課題を設計に**共有**

分析ポイント3: 「課題」を事後的・予防的双方の観点から分析

問題点3: **全ての「課題」が発生後の対応になっていた**

「課題」を**事後的・予防的双方の観点**から分析する。



過去の解決課題の事例を活用して**予測・予防**する。

課題の**継続的な分析**

まとめ: 課題分析のポイント

課題の**収集**と**蓄積**

過去の解決事例を活用し、**隣接業務や類似業務も含めて分析**する。

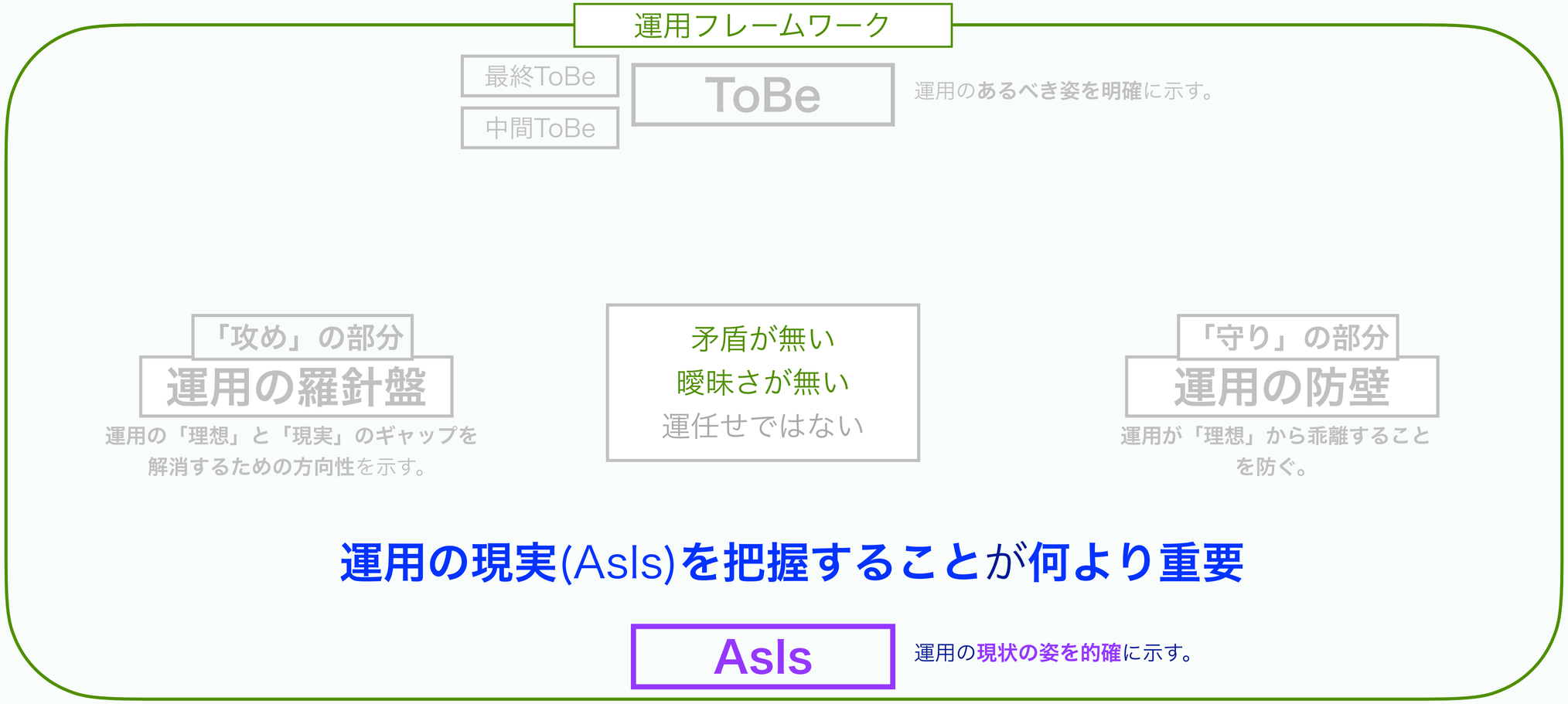
課題を設計に**共有**

「課題」は原則として**設計にフィードバック**し、**設計側で解決**する。

課題の**継続的な分析**

過去の解決課題の事例を活用して**予測・予防**する。

運用課題分析の前提: 運用現場の「現実」の把握



既に運用が動いている場合は、一番最初にやるべきこと。

運用課題の分析に必要なこと = 運用現場のAsIsの把握

運用の現実(AsIs)を把握することが何より重要

「運用課題」というネガティブなAsIsだけではなく、運用のポジティブなAsIsも含めて把握すべき

運用現場のAsIsの**収集**と**蓄積**

運用現場のAsIsを**運用現場と設計側で共有**

運用現場のAsIsの**継続的な分析**

本編 運用課題の分析

AsIsを知るためのサブフレームワーク

運用現場のAsIsの把握

運用の現実(AsIs)を把握することが何より重要

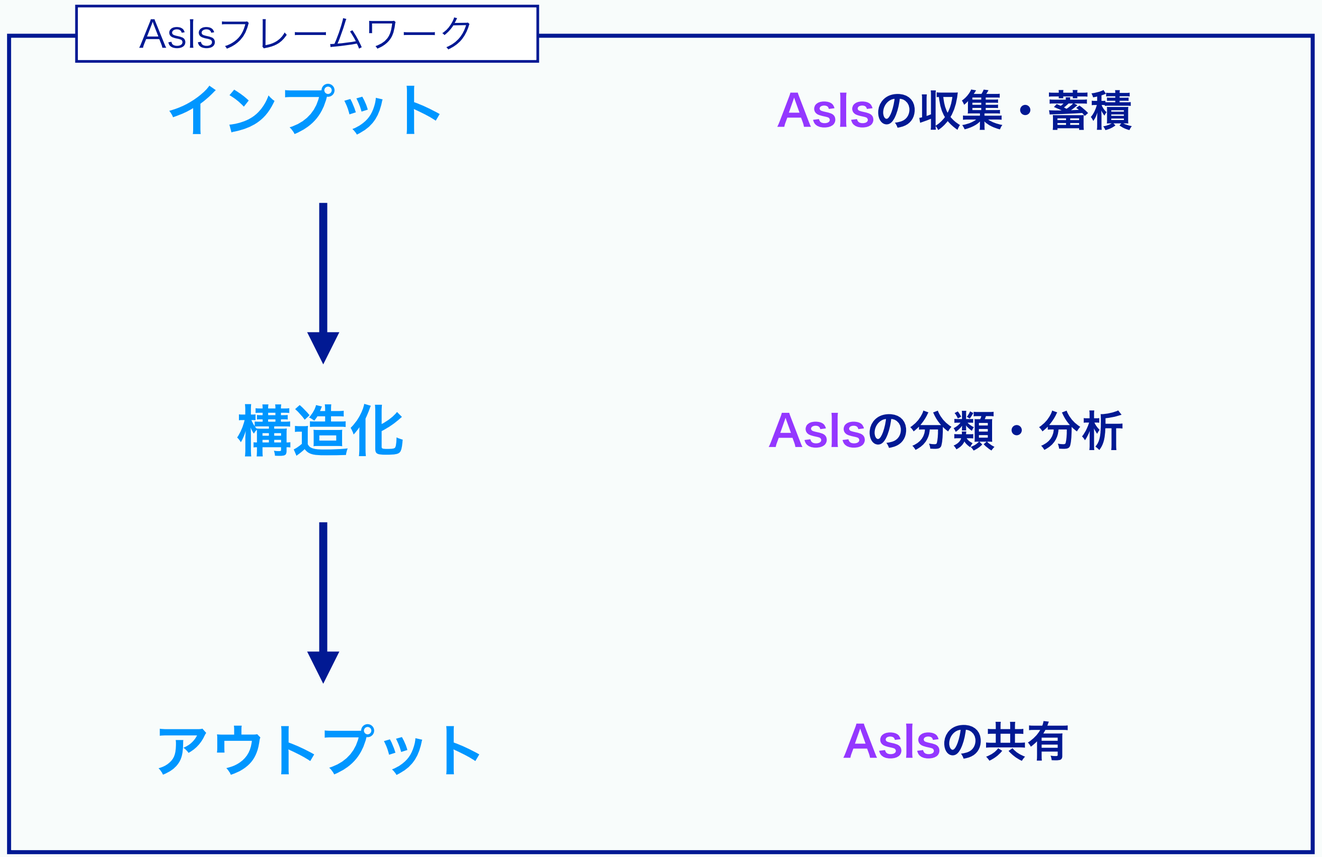
「運用課題」というネガティブなAsIsだけではなく、運用のポジティブなAsIsも把握すべき

運用現場の**AsIsの収集と蓄積**　AsIsのインプット　AsIsの構造化 (分析)

運用現場の**AsIsを運用現場と設計側で共有**　AsIsのアウトプット

運用現場の**AsIsの継続的な分析**　AsIsの構造化 (分析)

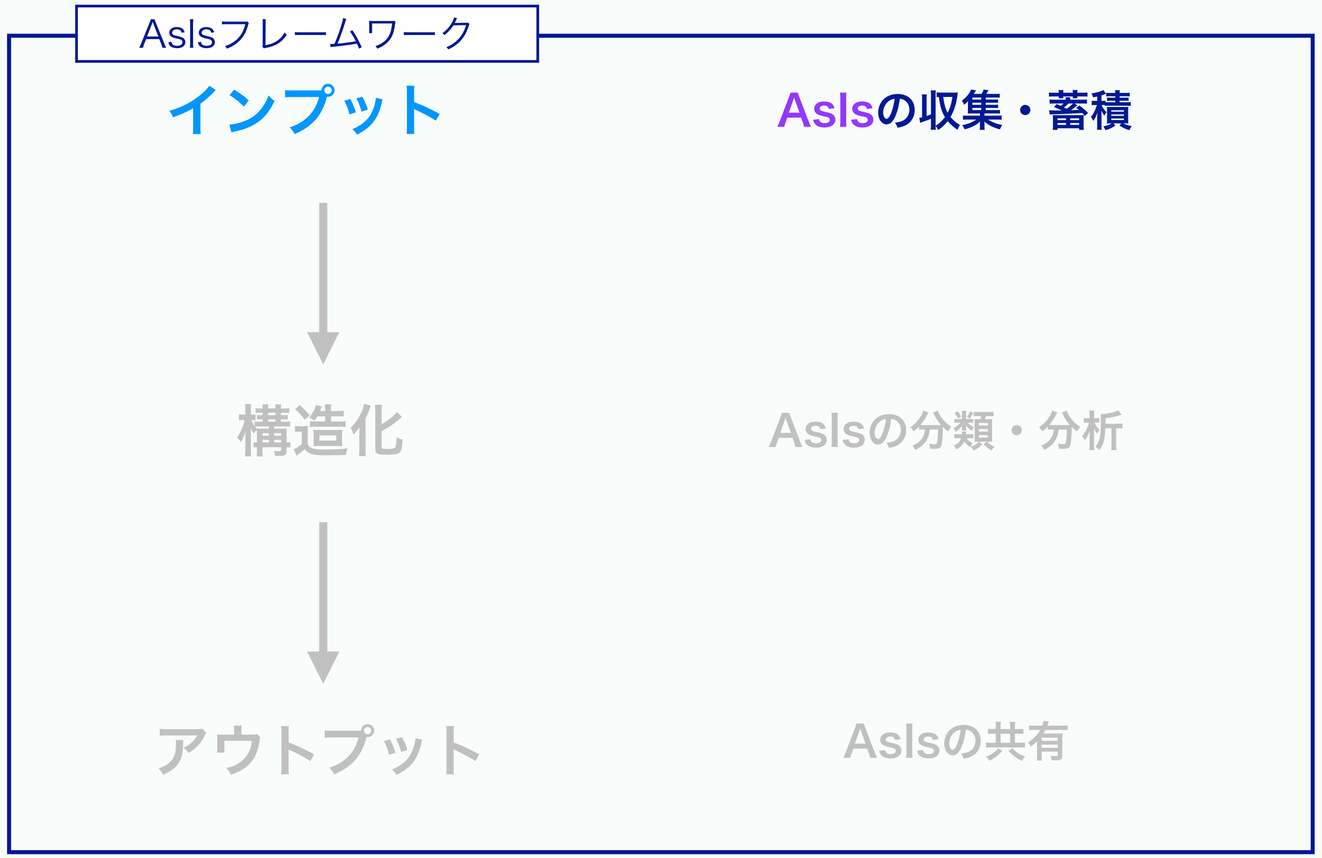
AsIsを知るためのサブフレームワーク



AsIsを知るためのサブフレームワーク

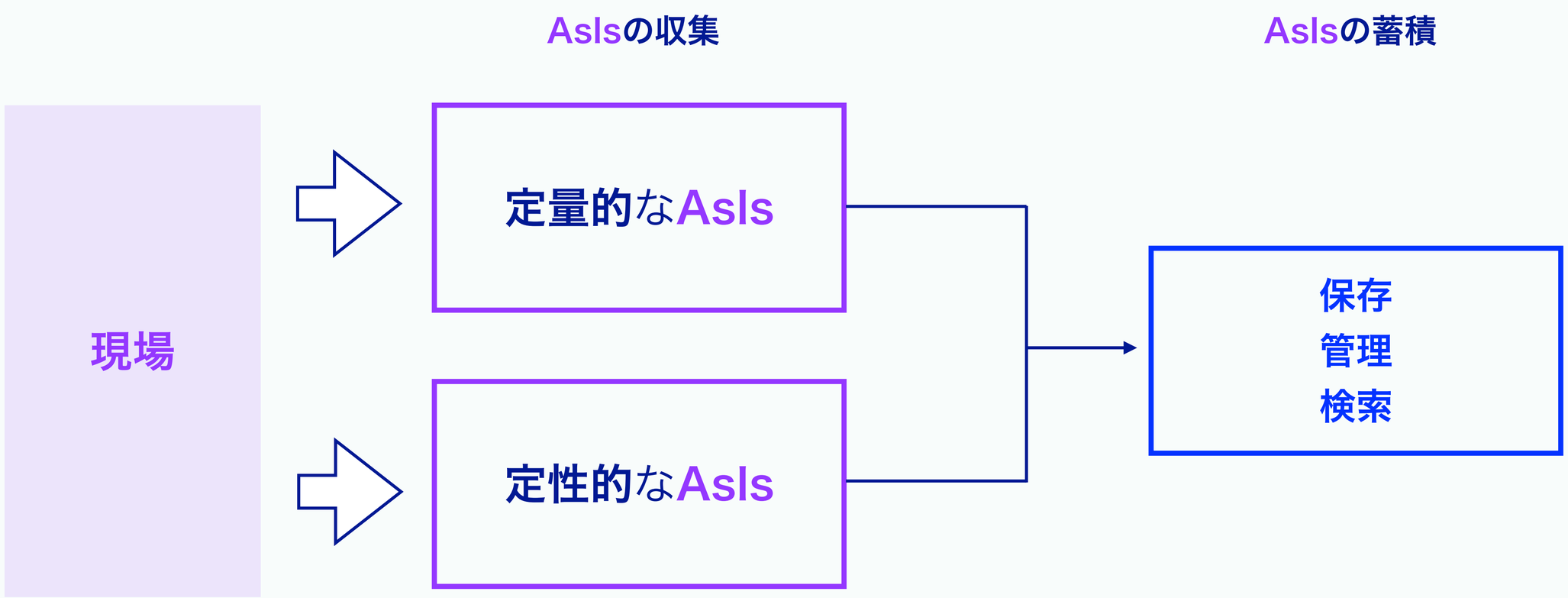
1. インプット

Aslsを知るためのサブフレームワーク



AsIsのインプットの全体像

インプット



AsIsのインプット (収集: 定量的AsIs)

インプット

AsIsのインプット (収集: 定量的AsIs)

インプット　AsIsの収集

現場の定量的な情報を収集する仕組みが必要。

Step1	収集対象の選定	収集する定量的情報を選定する。
Step2	一次情報の決定	収集する定量的情報に必要な一次情報を決定する。
Step3	データフローの整備	一次情報を定量的情報に変換するためのデータフローを整備する。

データフローの整備により、定量的情報の自動集計への下地を整える。

AsIsのインプット (収集: 定性的AsIs)

インプット

AsIsのインプット (収集: 定性的AsIs)

インプット　AsIsの収集

現場の定性的な情報が上がってくる、上がってきやすい仕組みが必要。

Step1	三現主義	マネージャやアーキテクトが定期的に現場に話を聞きにいく。
Step2	場の整備	気軽にヒヤリハットの存在を共有する場を整備する。
Step3	文化の醸成	最初にミスした人やネガティブ情報を上げた人を評価する。

「現場に聞きに行くこと」から初めて、仕組み化、文化の形成をしていく。

Aslsのインプット（蓄積)

インプット

Aslsのインプット（蓄積)

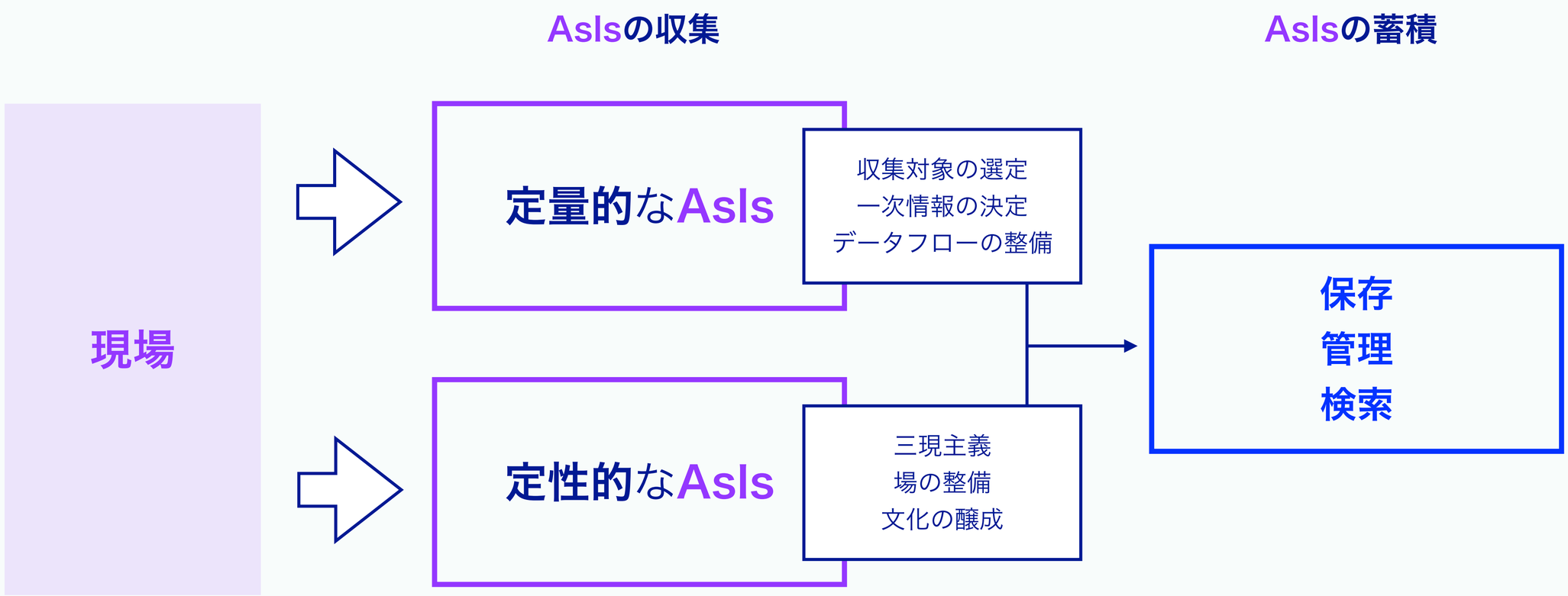
インプット　Aslsの蓄積

収集したAslsを蓄積する仕組みが必要。

保存	収集したAslsを永続的に保存する。(内容によっては数年で破棄)
管理	メタ情報(情報の管理情報)の管理をする。
検索	蓄積されたAslsを効率的に検索・参照する。

まとめ: AsIsのインプット

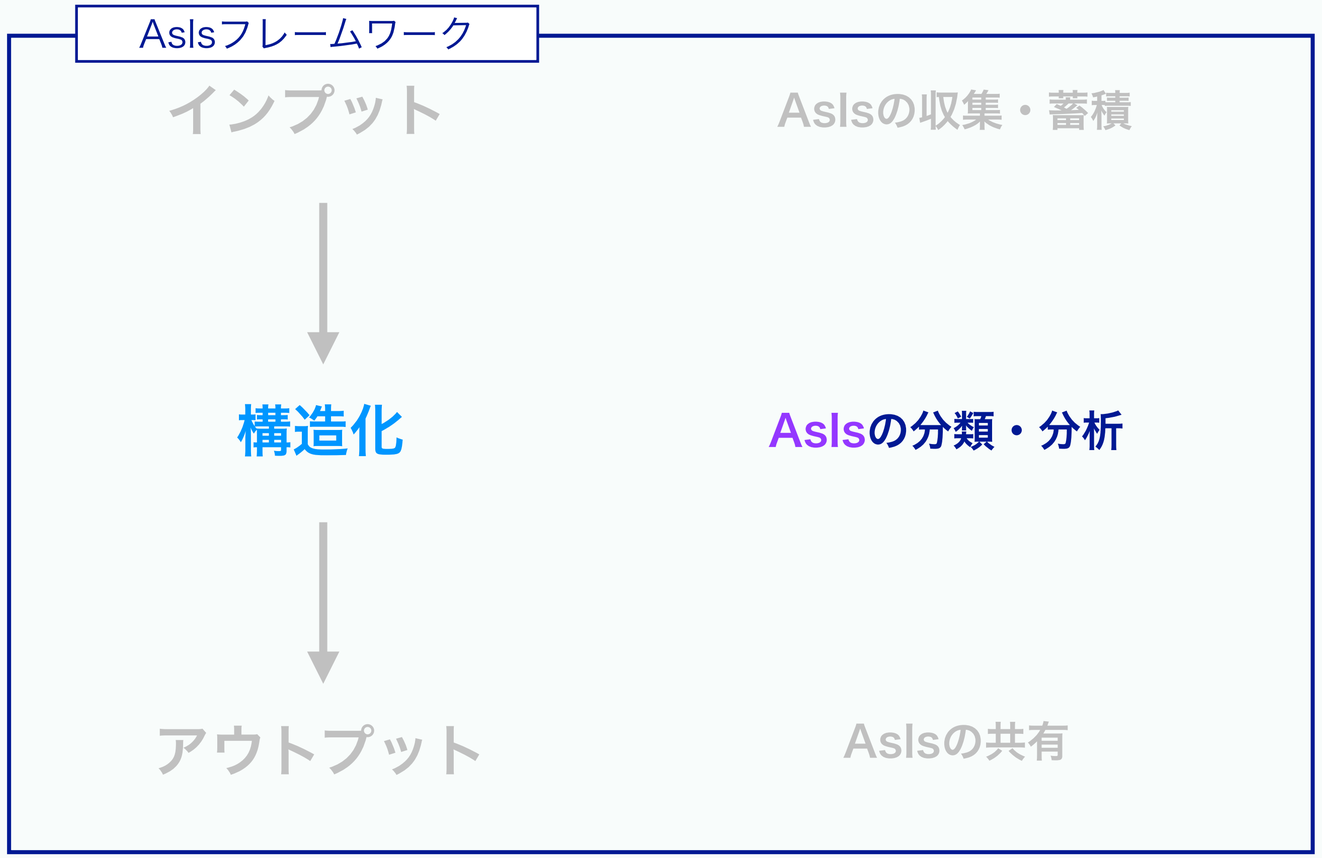
インプット



AsIsを知るためのサブフレームワーク

2. 構造化

AsIsを知るためのサブフレームワーク

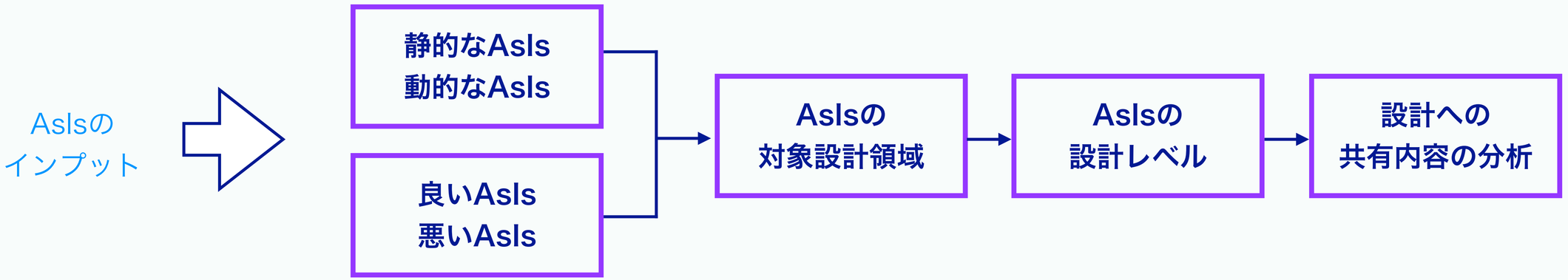


AsIsの構造化の全体像

構造化

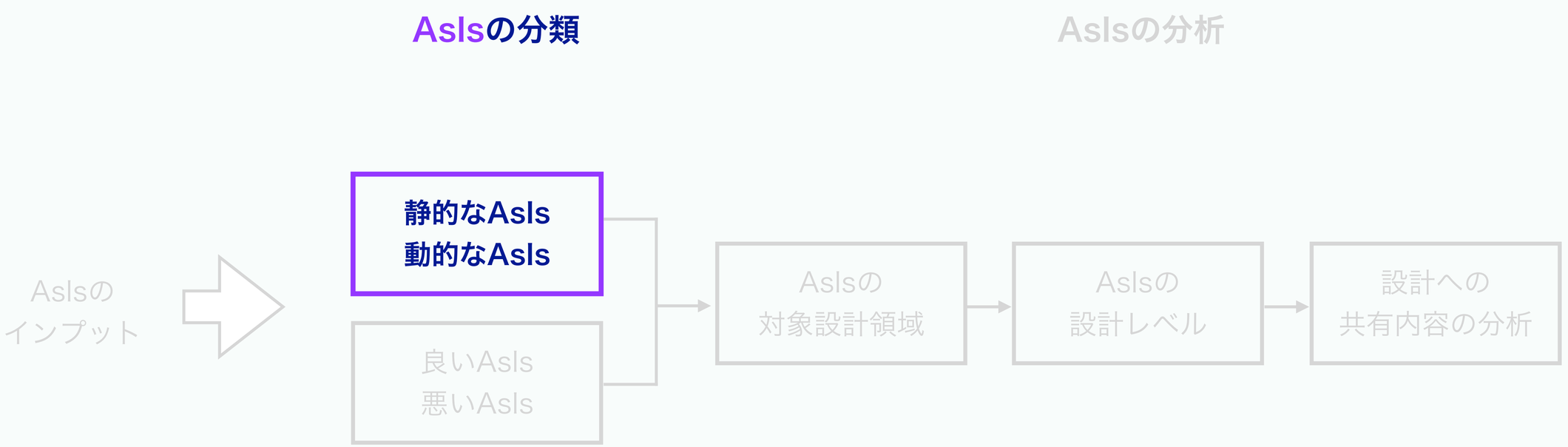
AsIsの分類

AsIsの分析



AsIsの構造化 (分類: 静的なAsIsと動的なAsIs)

構造化



AsIsの構造化 (分類: 静的なAsIsと動的なAsIs)

構造化　　AsIsの分類

AsIsに対する**説明責任**がどちらにあるかを分類する。

静的なAsIs　現在の設計、実装

「であるはず。」**上位職や設計者の認識**

現在の設計や実装の説明責任を担う。
設計や実装に曖昧さ、矛盾、運任せがあった場合は修正する。

動的なAsIs　現在、運用現場で起きていること、起きている変化

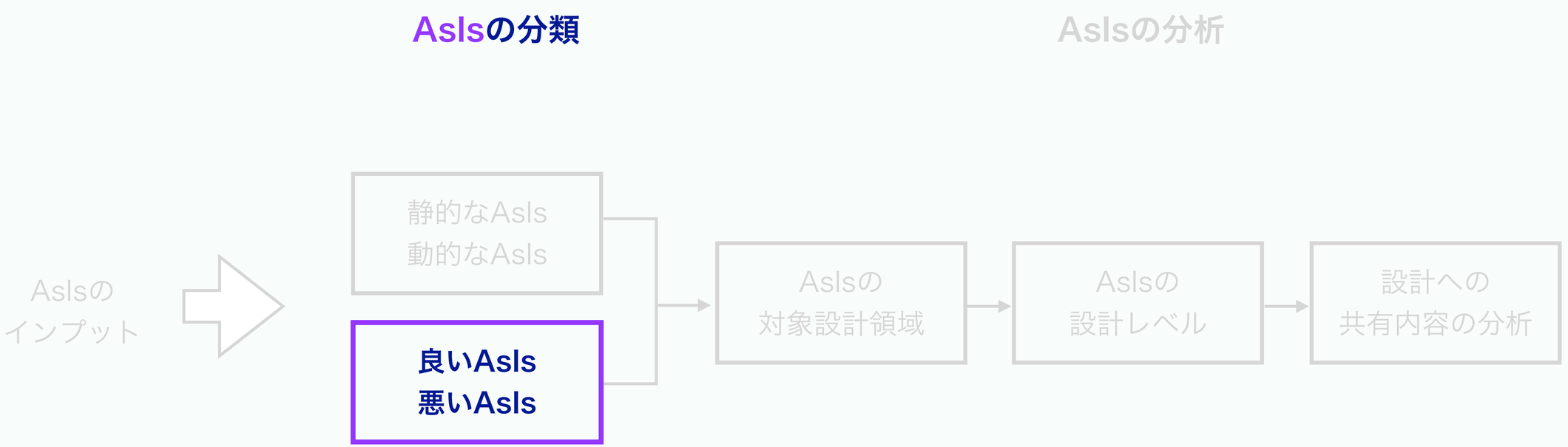
「である。」「となりつつある。」**現場職の認識**

現場における現実や変化の説明責任を担う。
設計と現実の差異があった場合は情報を設計側に共有する。

自己の**認識を言語化**して、**他者に伝えることができる「説明能力」**が全員に求められる。

Aslsの構造化 (分類: 良いAslsと悪いAsls)

構造化



AsIsの構造化 (分類: 良いAsIsと悪いAsIs)

構造化　　AsIsの分類

AsIsの善し悪しを判断する基準を、運用全体で明確にして分類する。

良いAsIs　ToBeに近い状態にある。

- **静的に設計通り。かつ、動的に現実に合致。**

仮にToBeに近くなくても「現段階の設計」には沿っている。

課題が発生しにくい。
課題が発生しても対処しやすい。

悪いAsIs　Tobeから遠い状態にある。

- **設計通りに実装されていない。(違反実装)**
- **設計が現実から乖離。(乖離設計)**
- **そもそも設計がされていない。(未設計)**

矛盾、曖昧、運任せが存在している。

これが自体が「課題」である。
更に、新たな課題の要因になっている。

多くの場合、悪いAsIsが課題の発生源。

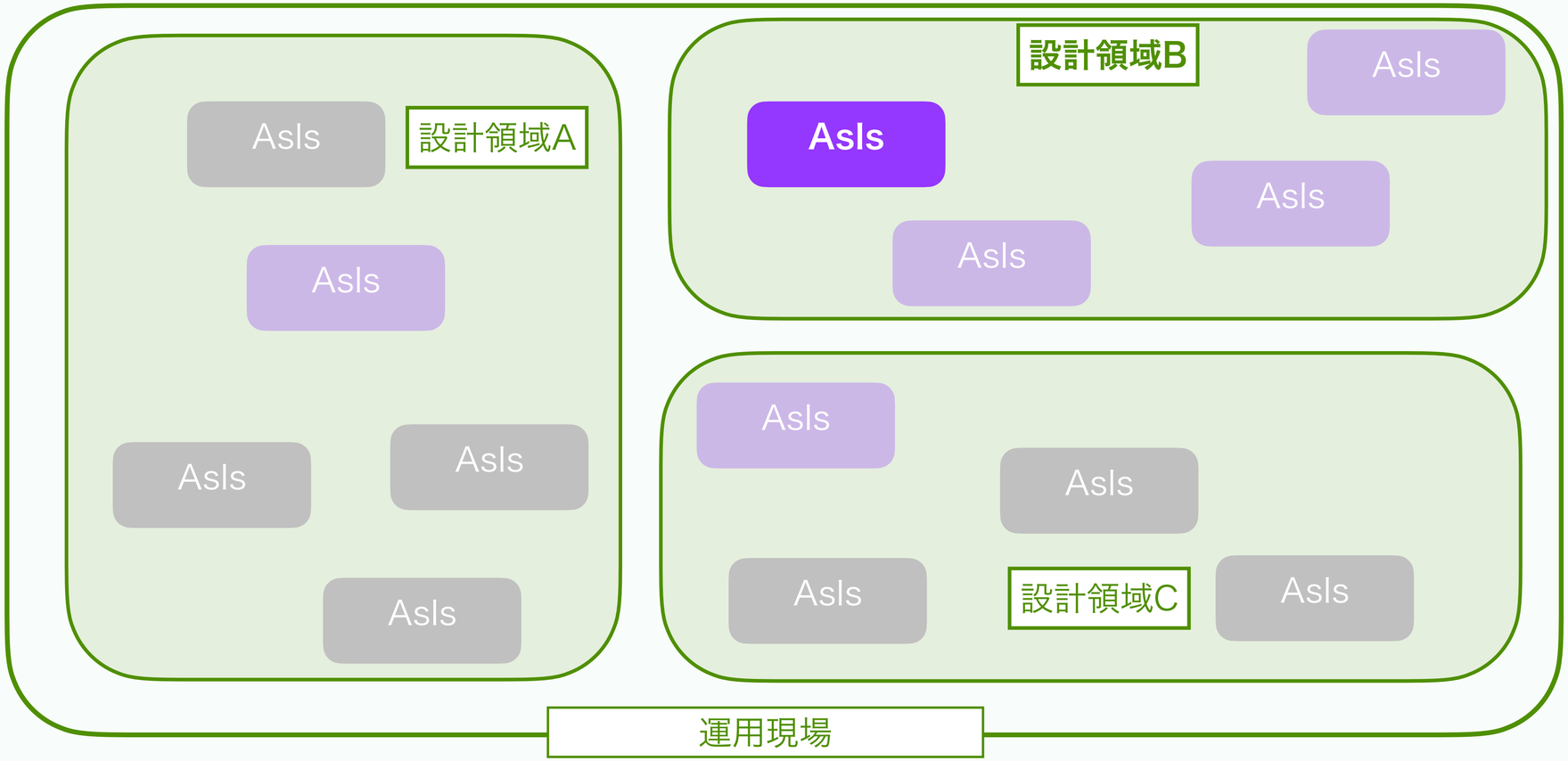
Aslsの構造化 (分析: どの設計領域のAslsか)

構造化

AsIsの構造化 (分析: どの設計領域のAsIsか)

構造化　　AsIsの分析

AsIsが関連する設計領域や影響する範囲を分析する



Aslsの構造化 (分析: Aslsレベル)

構造化

AsIsの構造化 (分析: AsIsレベル)

構造化　　AsIsの分析

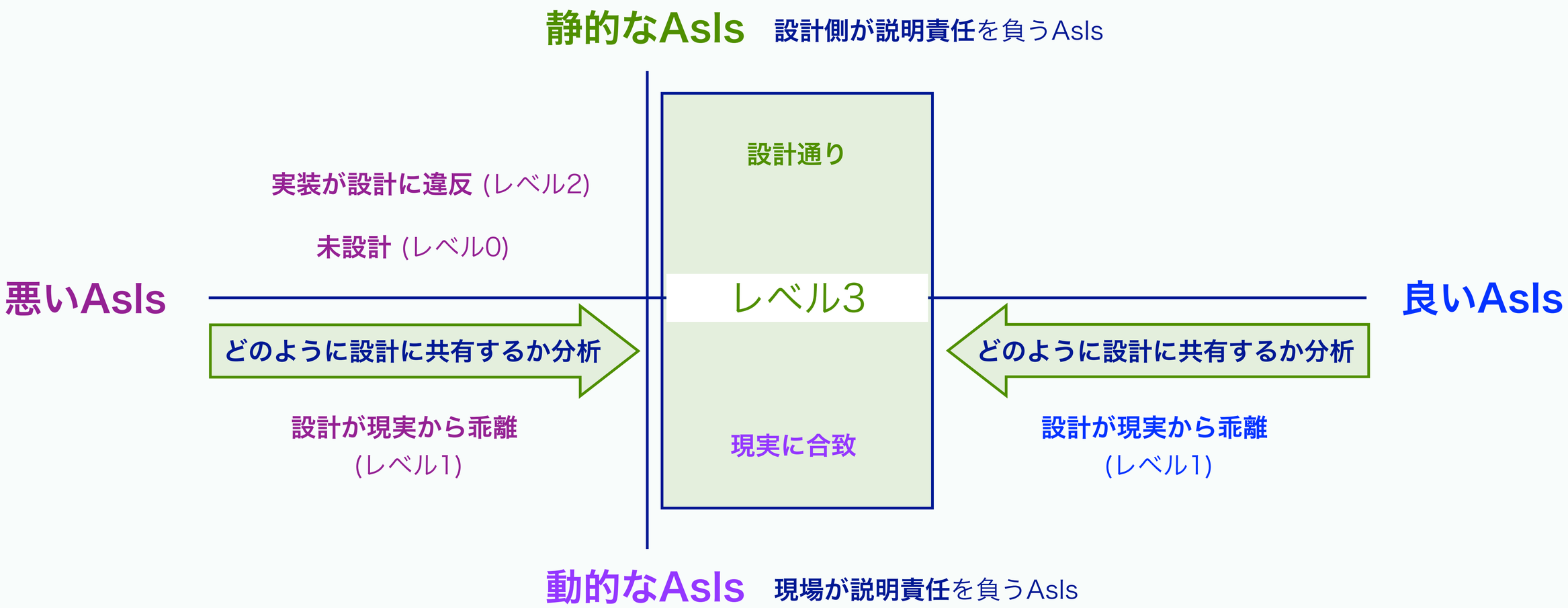
AsIsにより判明した設計の状態(レベル)**を分析する**

レベル0	**未設計**	**設計がされていない状態や曖昧な状態が判明** 環境や前提条件の変化により、いつでも発生し得る。
レベル1	**設計が現実から乖離**	**設計が現実から乖離している状態が判明** 現実の変化により、いつでも発生し得る。
レベル2	**実装が設計に違反**	**設計が現実に即していても、実装が設計から乖離している状態が判明** 設計の変化により、いつでも発生し得る。
レベル3	**設計と現実が合致**	**設計が現実に即し、実装も設計に即している状態が判明**

現実の変化により、AsIsレベルは常に下がっていく

参考: Aslsの分類とAslsレベル

構造化　　**Aslsの分類・分析**



Aslsの構造化 (分析: 共有内容の分析)

構造化

AsIsの構造化 (分析: 共有内容の分析)

構造化　　AsIsの分析

「課題」を「設計で解決」(根本解決)**するための分析**

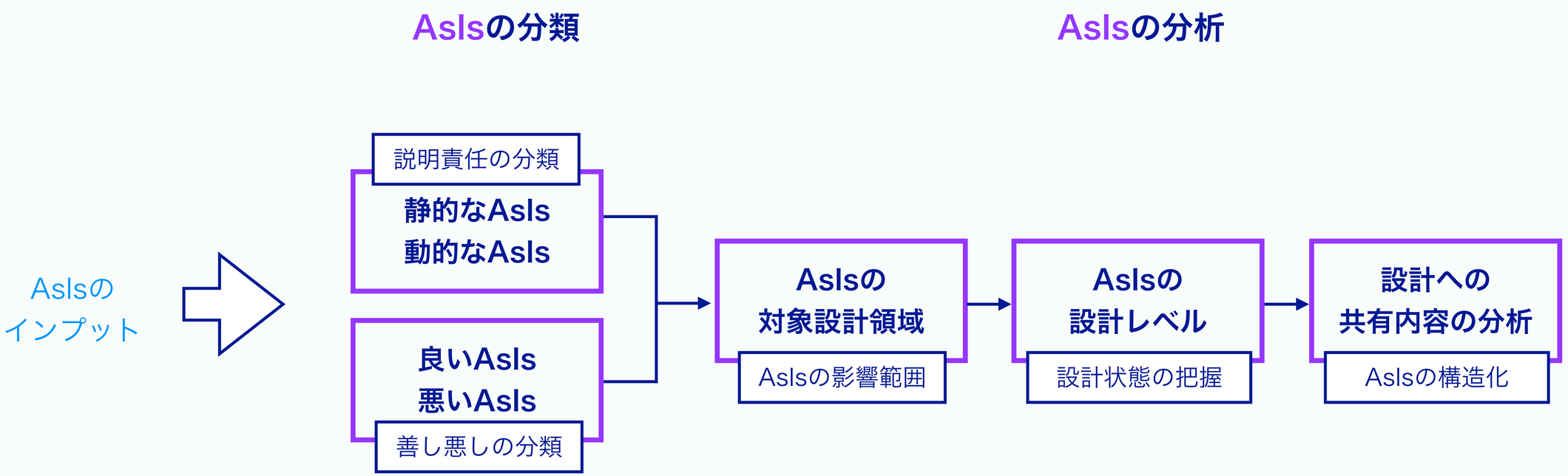
良いAsIsの場合は、「どのように設計に取り込むべきか」を分析する。

- **AsIsは常に変わる。**インプットで上がってきた新鮮な情報をすぐに活かす。
- **客観的に情報を見る。**定性的、定量的のバランス良く見る。
- **数字で表現する。**やはり定量的に評価できるとより良い。

現場と設計側の双方が理解しやすい形で、分析結果を示すことが大切。

まとめ: AsIsの構造化

構造化



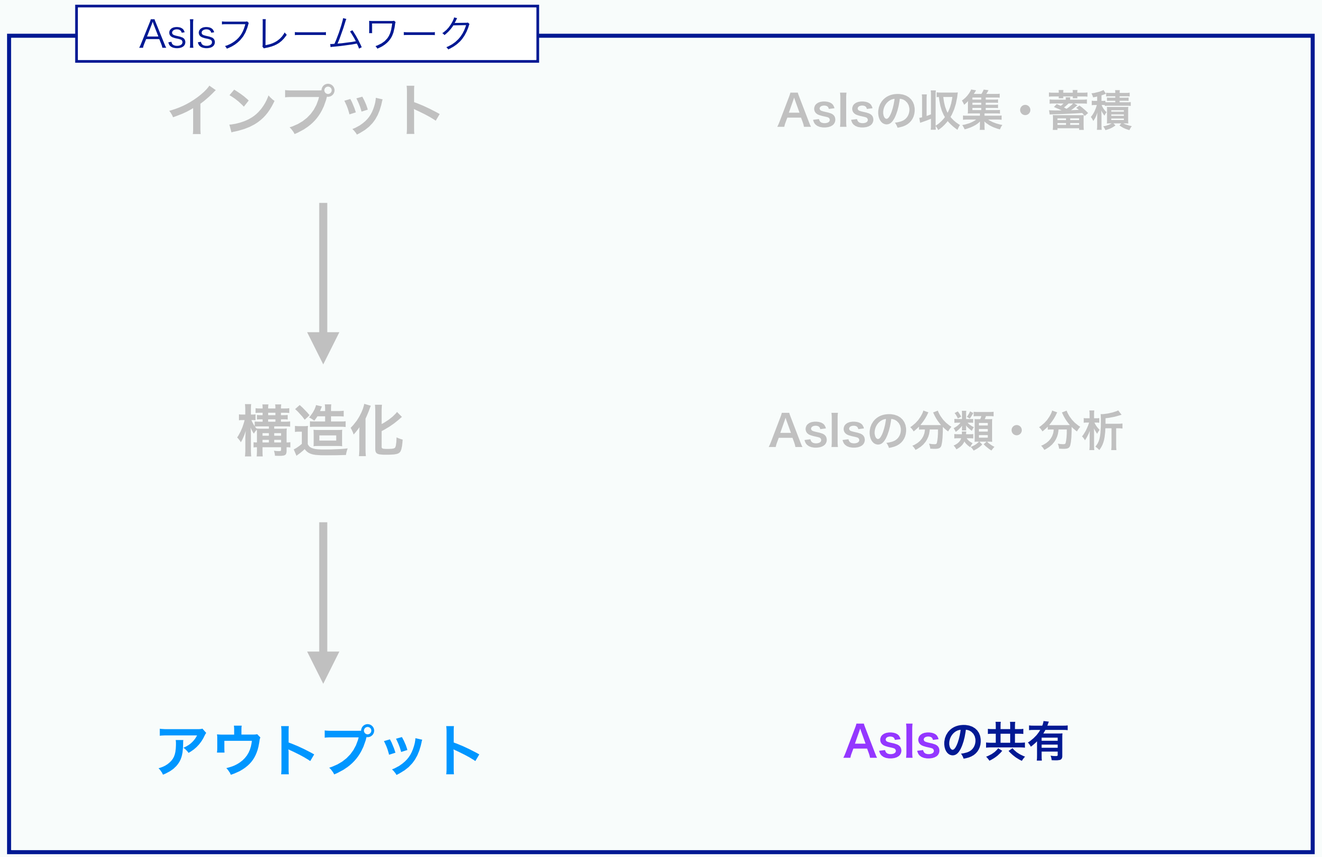
説明責任の分担とAsIsの善し悪しの
判断基準を明確にする。

現場と設計側の双方がわかりやすい形で
分析結果を共有する。

AsIsを知るためのサブフレームワーク

3. アウトプット

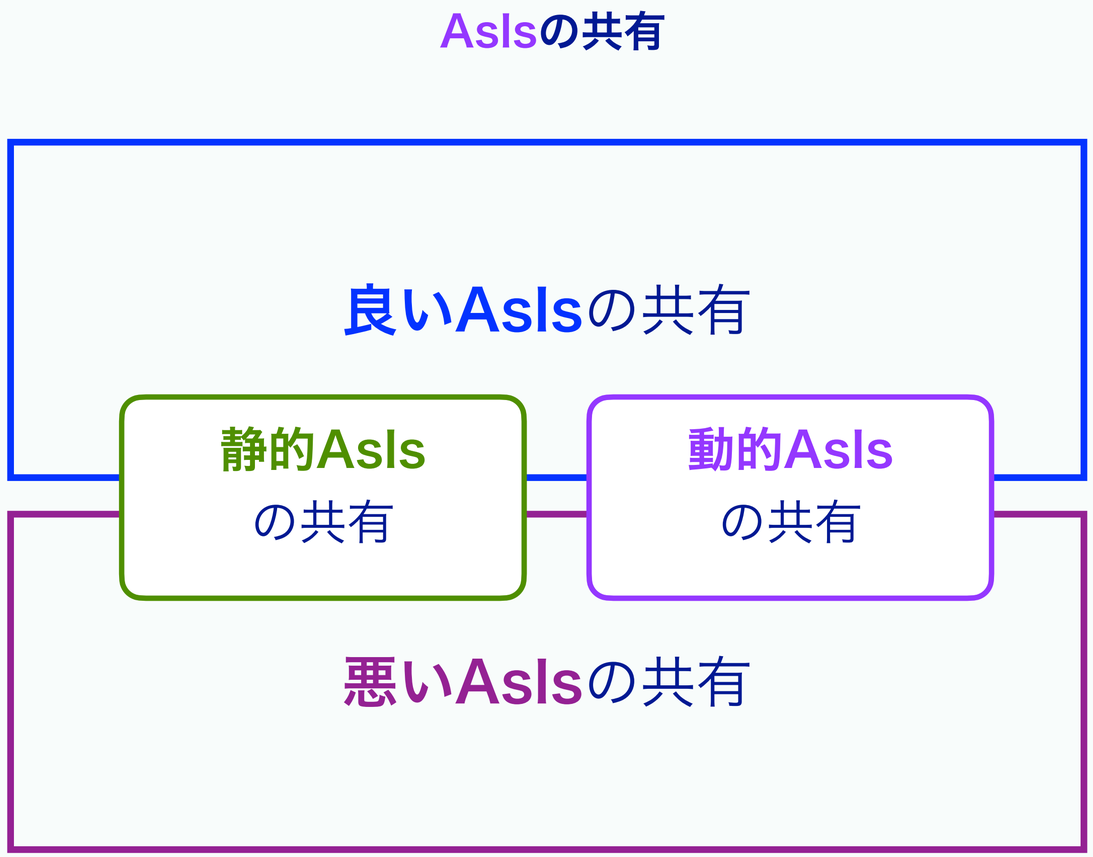
AsIsを知るためのサブフレームワーク



Aslsのアウトプットの全体像

アウトプット

Aslsの
構造化



Aslsのアウトプットの全体像

アウトプット

Aslsの
構造化

Aslsのアウトプット (良いAsls vs. 悪いAsls)

アウトプット　　Aslsの共有

運用に関与する人なら誰でも見えるところに、Aslsを公開する。

良いAsls	設計ドキュメント・実装ドキュメントで共有する。
悪いAsls	問題管理システムで共有する。

正常系Aslsの共有

異常系Aslsの共有

容易にアクセスでき、内容がわかりやすいことが大事

Aslsのアウトプットの全体像

アウトプット

Aslsの
構造化

AsIsのアウトプット (静的AsIs vs. 動的AsIs)

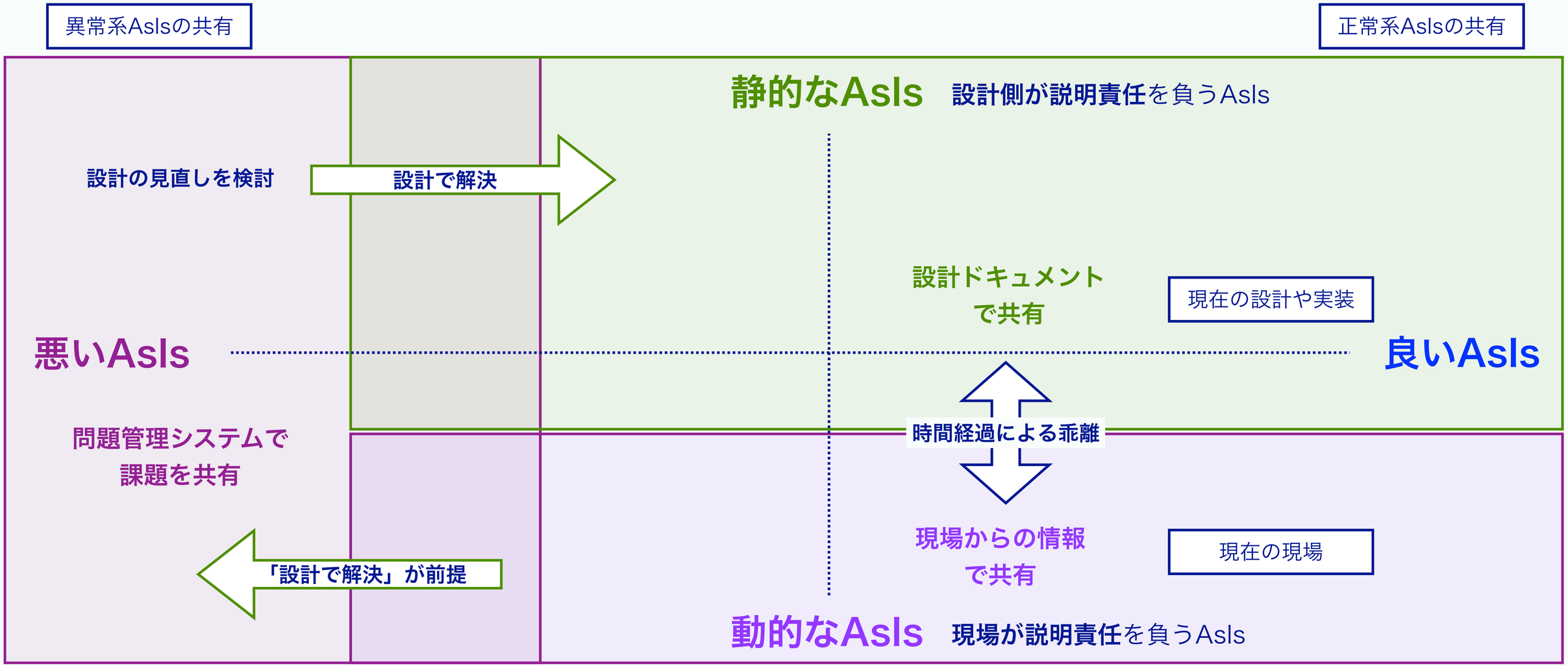
アウトプット　　AsIsの共有

運用に関与する人なら誰でも見えるところに、AsIsを公開する。

静的なAsIs	設計や実装のドキュメントを作成・更新し、正確に共有する。	現在の設計や実装
動的なAsIs	現場の声をレポートし、的確に共有する。	現在の現場

良いAsIsは正常系、悪いAsIsは異常系として扱う。

AsIsのアウトプット: 仕組み化のイメージ



留意点: 悪いAslsの扱い

アウトプット　　Aslsの共有

設計変更せずに、**いきなり対応しよう**としない。

良いAsls　　設計ドキュメント・実装ドキュメントで共有する。

悪いAsls　　**問題管理システムで共有する。**

いきなり対応しようとしない。

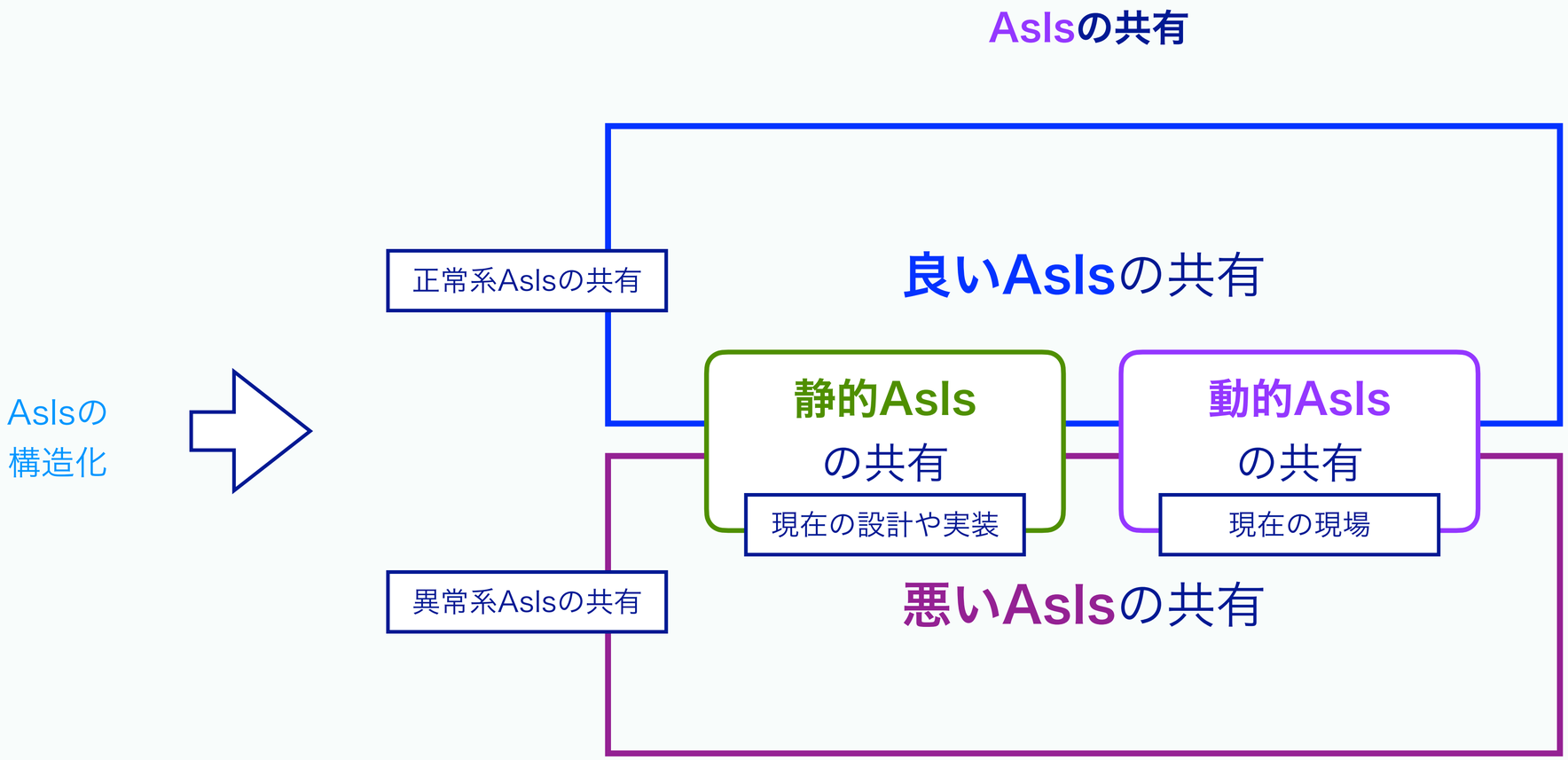
正常系Aslsの共有

異常系Aslsの共有

現状の設計やToBeとの整合性を考え、必ず再設計する必要がある。

まとめ: Aslsのアウトプット

アウトプット



まとめ

講義のふりかえり

運用課題の分析に必要なこと

運用の現実(AsIs)を把握することが何より重要

「運用課題」というネガティブなAsIsだけではなく、運用のポジティブなAsIsも含めて把握すべき

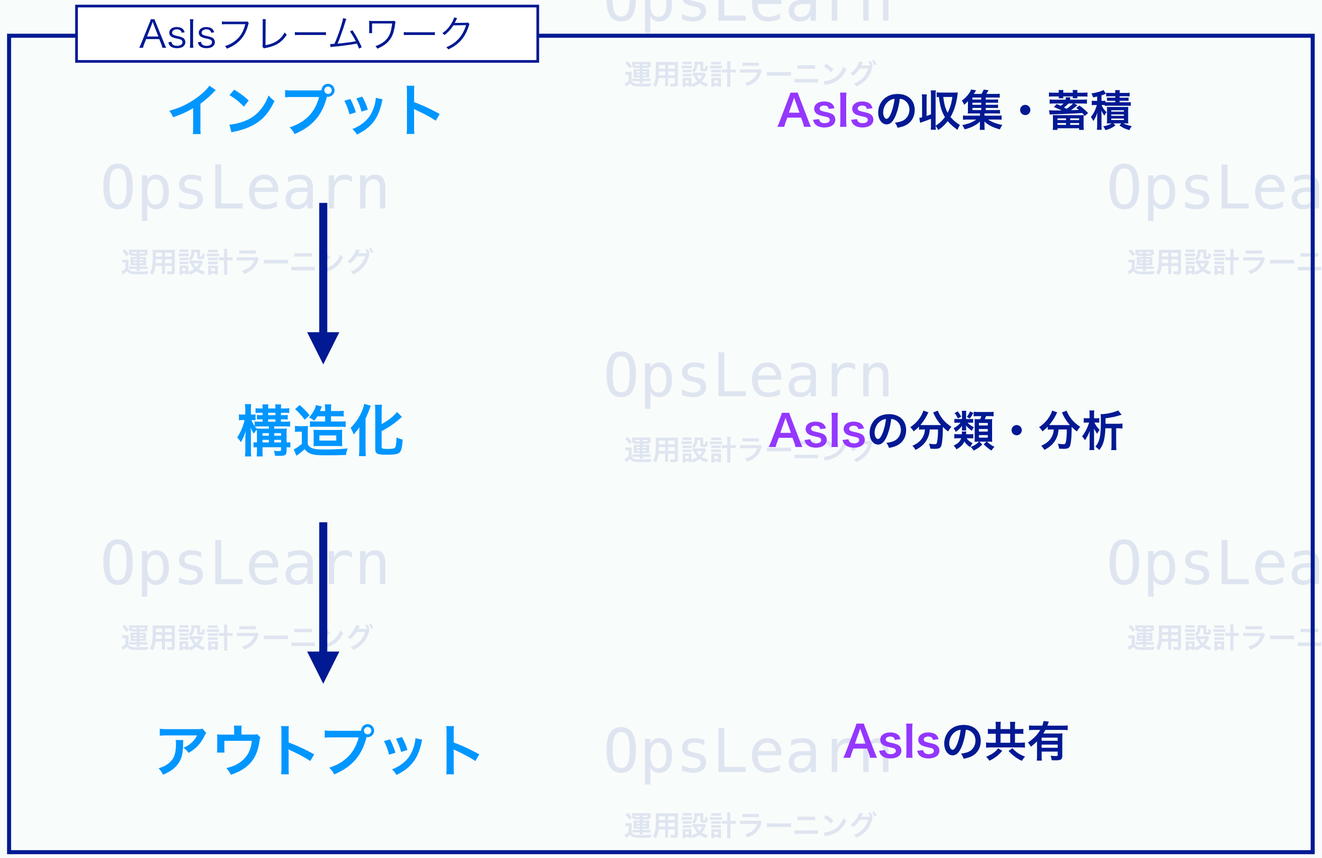
運用現場の**AsIsの収集と蓄積**

運用現場の**AsIsを運用現場と設計側で共有**

運用現場の**AsIsの継続的な分析**

既に運用が動いている場合は、一番最初にやるべきこと。

AsIsを知るためのサブフレームワーク



重要! 課題を見つけても、いきなり対応しようとしない。

今回の学習ポイント (再)

インプット

本講義で意識してほしいこと

- 運用現場に多くの課題があり、現場がなかなか楽になっていかないのはなぜなのか。
- 運用現場においてAslsの認識のギャップが起こりやすいのはなぜか。

一次アウトプット

本講義から持ち帰ってほしい事

- 「運用のAsls」を把握する上での考え方
- 「運用のAsls」を分析する上での考え方
- 「運用のAsls」を共有する上での考え方

- 運用の**Aslsを把握・分析・共有するための仕組み**作り
- 自分達の「運用のAsls」は**今現在どうなっている**か?

最終アウトプット

今後の予定

運用フレームワーク

ToBe

運用の**あるべき姿を明確**に示す。

講義3　運用設計とは

第一回の「運用フレームワーク」の考え方に基づき、第二回の「運用課題の分析」で見えてきた
運用課題の根本原因を解消するための、あるべき「運用設計」について解説します。

講義1　運用フレームワークの考え方

講義2　運用課題の分析

AsIs

運用の**現状の姿を的確**に示す。