（株）SHIFT資料

若手が自律的に育つ環境とは

- 1年半で自動化アーキテクトになれた訳

2017.02.04

株式会社　SHIFT

目次

自己紹介

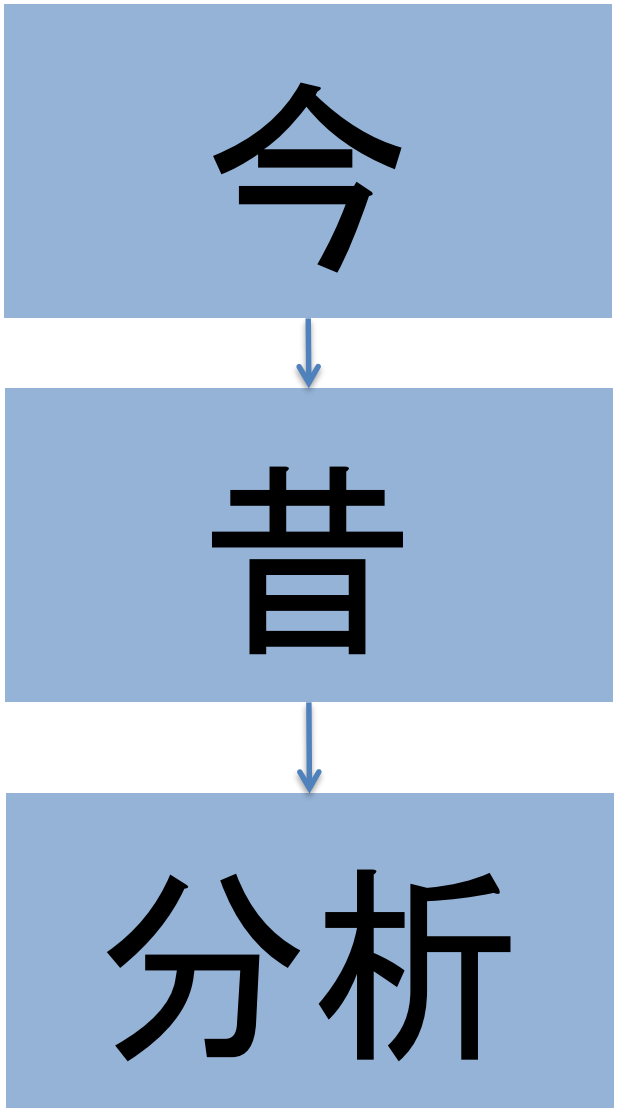
- 名前：山下 裕晃
- 所属：株式会社SHIFT 技術開発部
- 入社：2015年新卒入社（２年目）
- 出身：山口県（本籍：鹿児島）
- 大学：慶應義塾大学商学部
  - 専門は商品開発論（文化論、社会学的）
  - 大学時代は他大学の先生のカバン持ち

本公演の
ポイントと道筋

本公演のポイント

- 成長の流れ
- モチベーションの維持
- 具体的な勉強法
- 助けられた経験・良かった環境

本公演の道筋



自動化アーキテクトとしての仕事

未経験からの成長の歩み

自律的に成長出来た要因

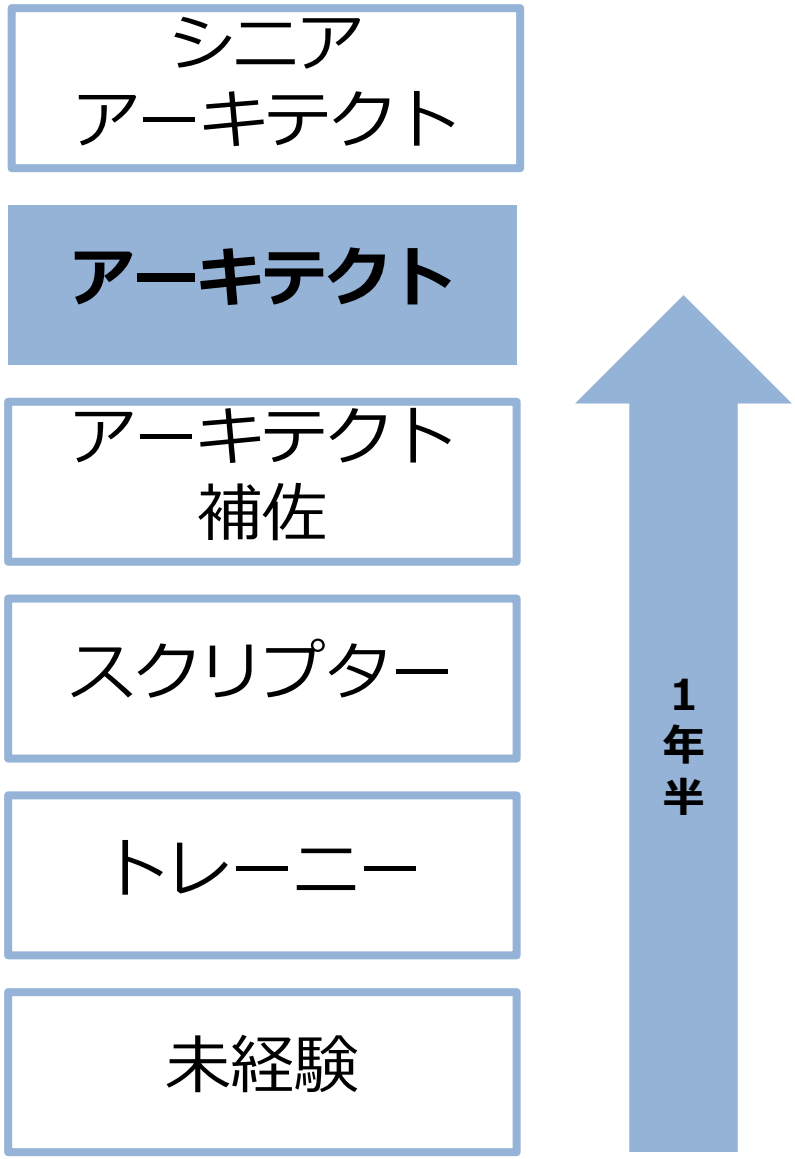
自動化アーキテクト
としての仕事

まず自動化アーキテクトって

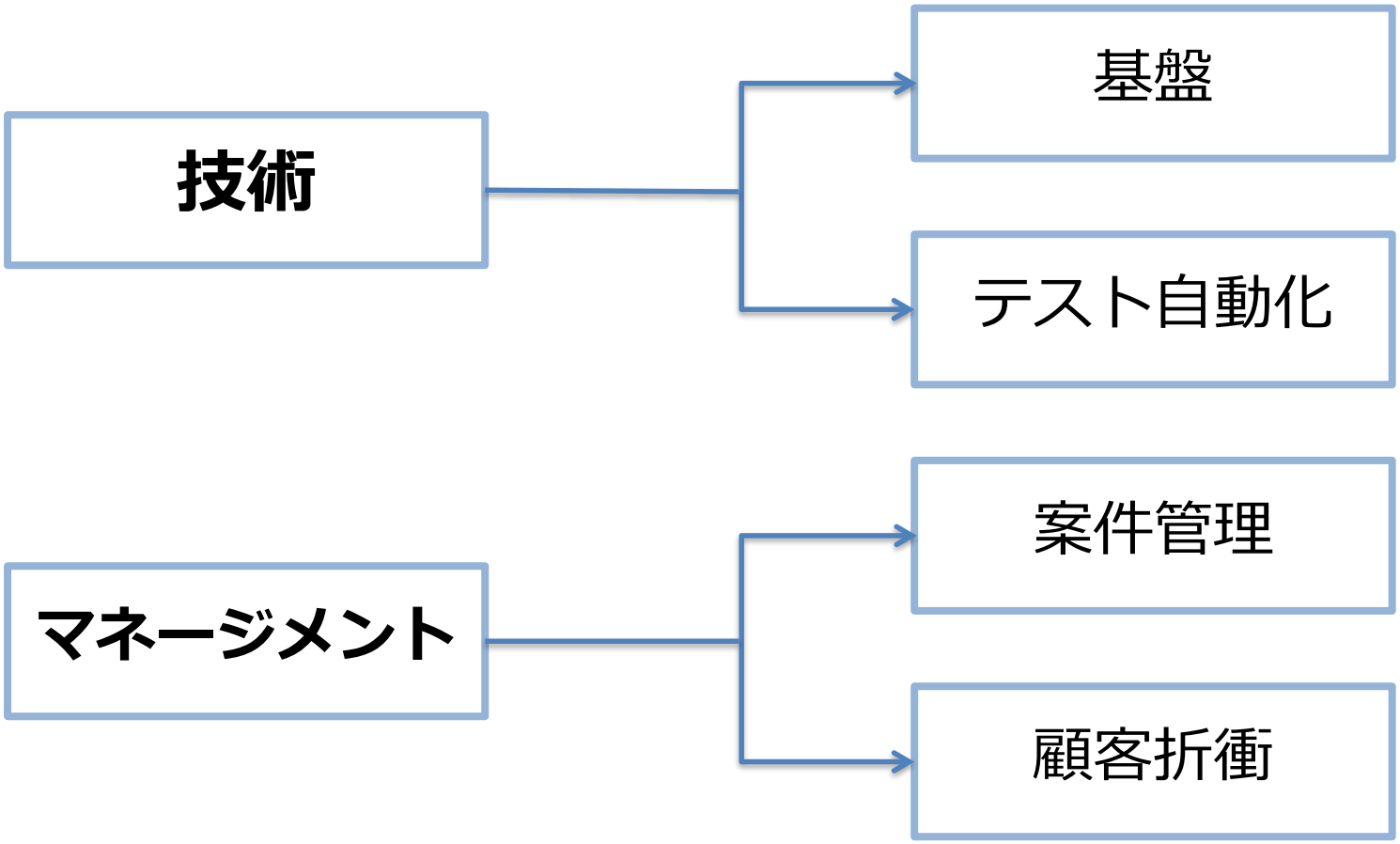
何を指しているの？

君は名乗れるの？

成長段階



自動化アーキテクトの定義



顧客の課題を洗い出し
技術によって解決出来ること

顧客と課題

顧客

- 旅行予約サービスを運営
- 毎年着実に成長。合わせて開発チームも拡大
- 一方で、レガシーな環境や仕組みで開発を継続

課題

- B to C サービスのため止まった際の**機会損失が大きい**
- 手動かつ属人化した作業により**品質維持に懸念**
- レガシー環境や手動作業により**開発効率が低下**

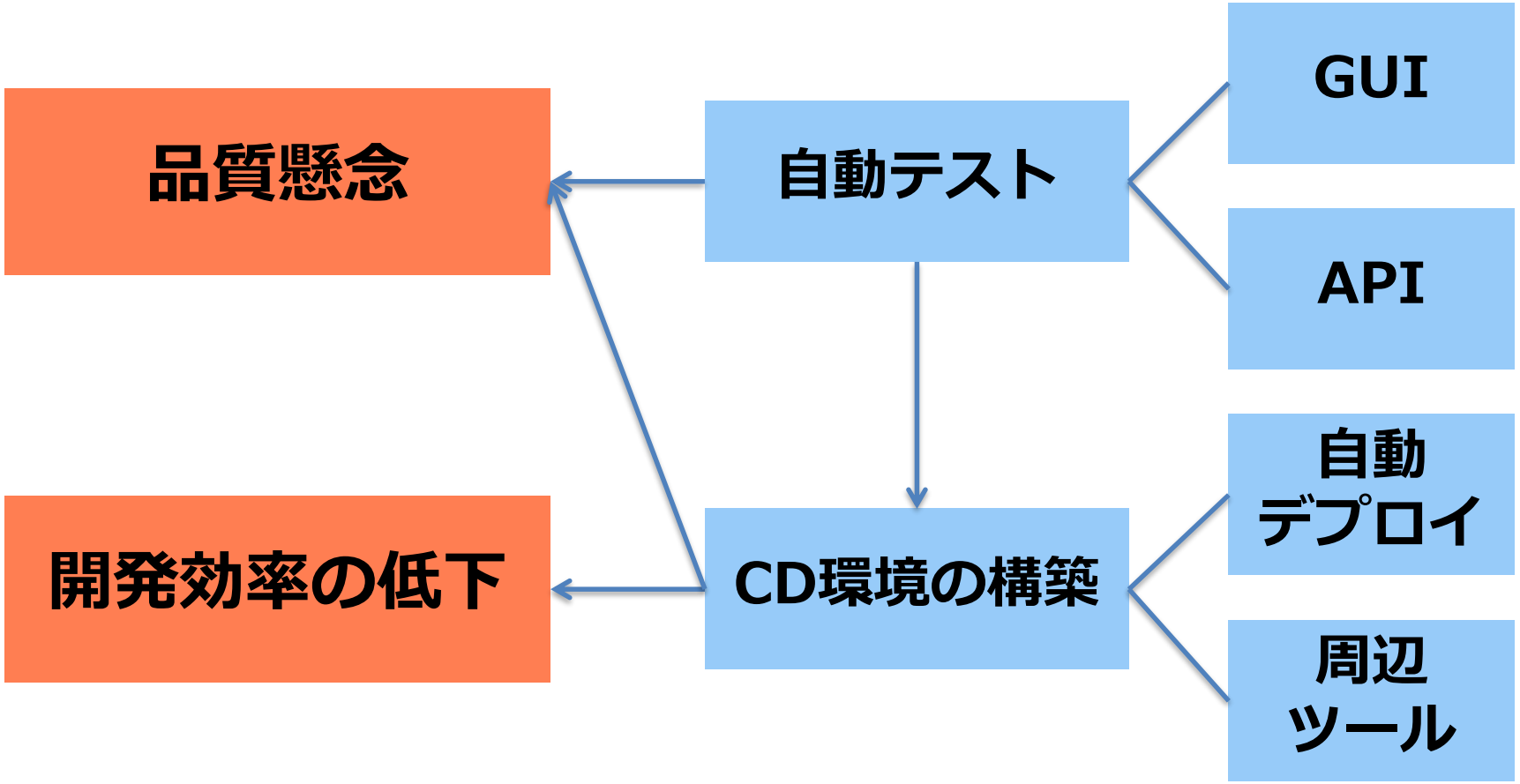
複雑に絡み合った問題が存在

品質懸念

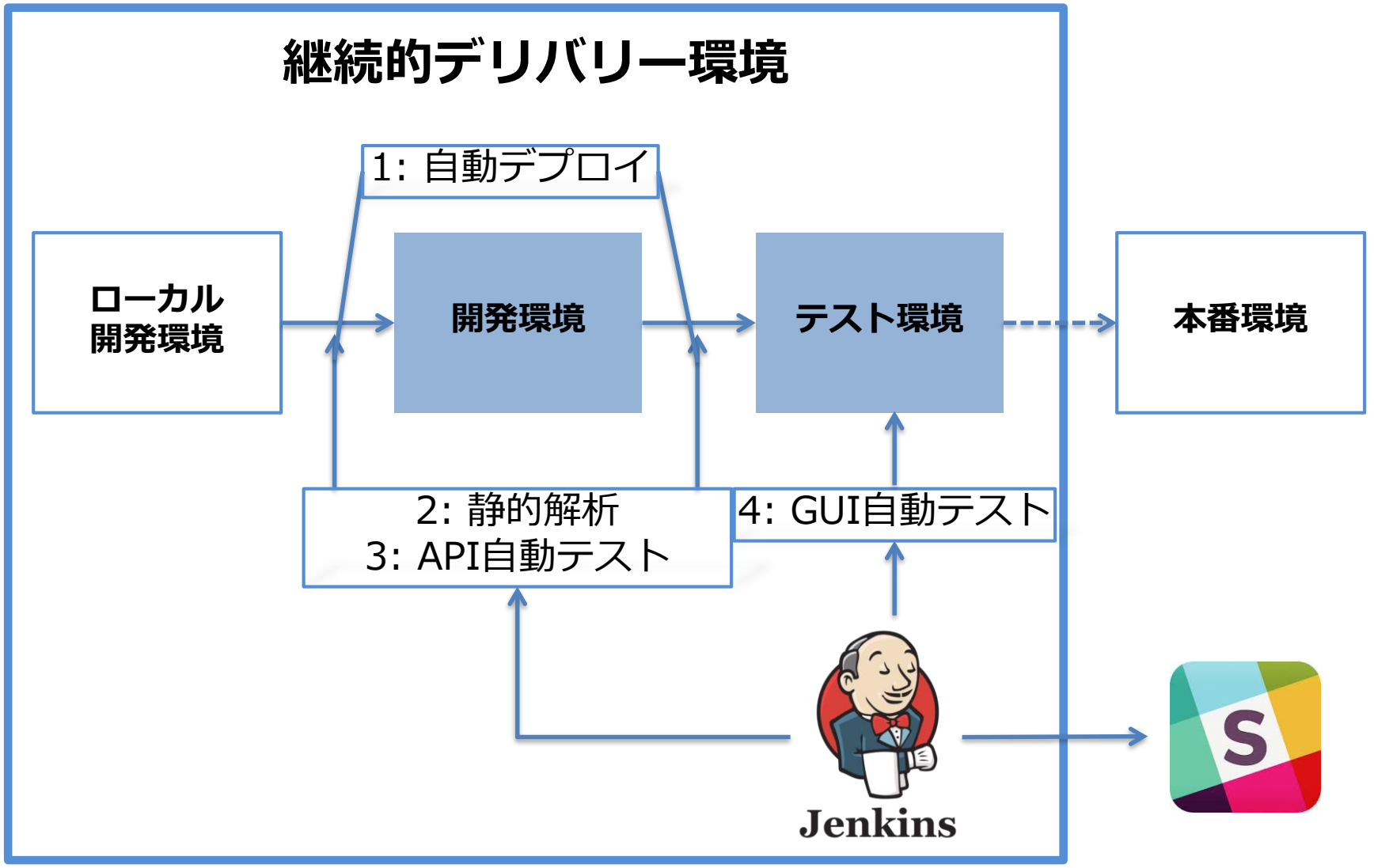
開発効率の低下

- 一つの開発環境で全員が作業
- ブランチが切られていない
- テストは開発環境に対して実施
- 回帰テストまで手が回らない
- テスト環境へのデプロイは手動
- テスト環境でのテストは
  開発側で簡単な動作確認のみ

継続的デリバリー環境の構築による抜本的改善



構築内容の概念図


継続的デリバリー環境
1: 自動デプロイ
ローカル
開発環境
開発環境
テスト環境
本番環境
2: 静的解析
3: API自動テスト
4: GUI自動テスト
Jenkins

自動デプロイにより工数圧縮と脱属人化

具体策：1

- Gitの導入に合わせてブランチ戦略を導入
- EC２やLambdaへのデプロイをAWS CLIで自動化
- 上記の処理をJenkinsを構築し、テストと合わせて実行、集計、Slackへの通知を実施

ミスが混入せず、誰でも高速なデプロイが出来、

常に最新のテスト環境が用意可能に

不具合の作り込みを未然に防ぐ

具体策：2

- 低コストかつ早期の不具合原因の発見するため SonarQubeを利用した静的解析の仕組みを導入
- Jenkinsを利用し、CD環境に組み込み

不具合発生率の低減とコード品質への意識を浸透

継続的デリバリーを支えるAPIテスト

具体策：3

- GUIテストよりも**短時間でかつ網羅的**なテストが出来るAPIテストを自動化（Frisbyを使用）
- 自動化スクリプト自体を書くのではなく、yamlファイルで記述すると自動生成する仕組みを構築

サービスの**基盤であるAPIの品質を幅広くカバー**

売上に直結する機能の品質を担保

具体策：4

- SeleniumやAppiumを使って回帰テストを自動化
- 毎日実行し、バグの早期発見を行える体制を構築
- クロスブラウザ・プラットフォームに対応

重要機能（予約）の**品質を担保**

手戻り工数とリードタイムの短縮

複雑に絡み合った問題にメスを入れた

品質懸念

開発効率の低下

- 個人の開発環境が整備
- ブランチ戦略が導入
- テスト環境へのデプロイは自動化
- テスト環境でテストを実施
- 毎日、回帰テストが実施

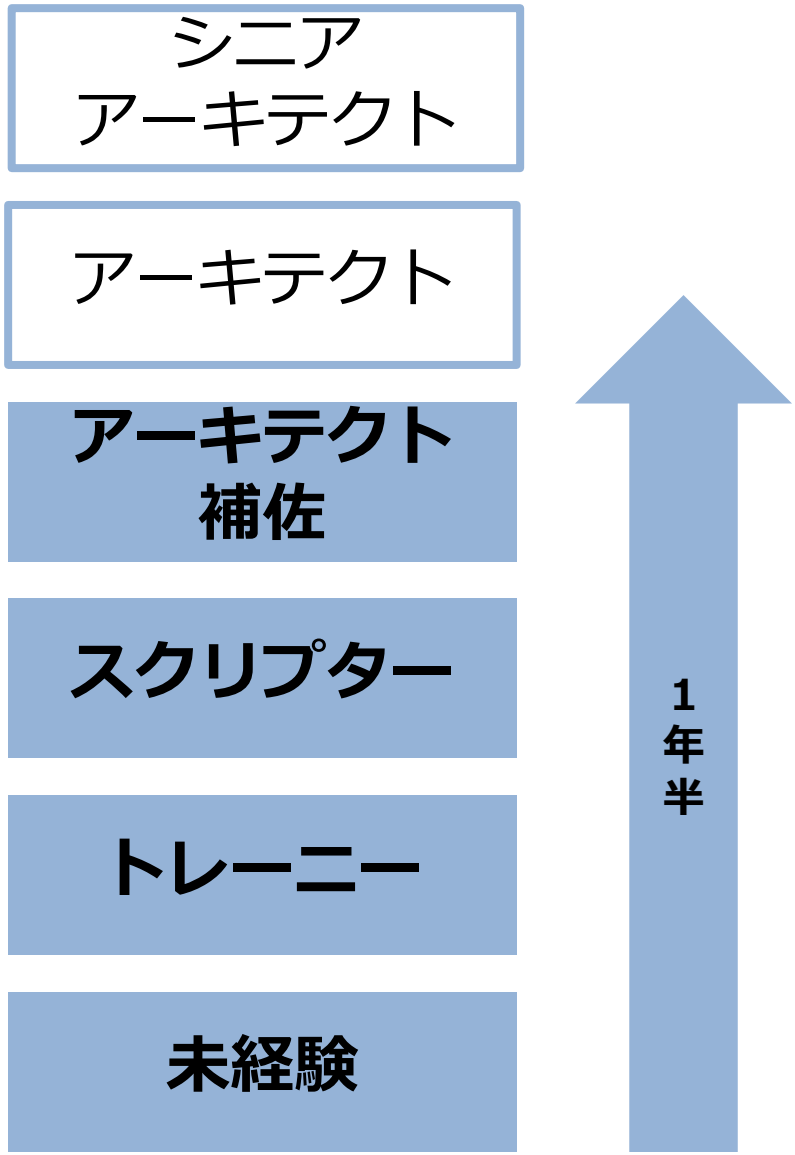
今後の展開

DevOpsの実現

- **IaC（インフラのコード化）の導入**
- **本番環境までのCD環境を拡張**
- **Blue/Greenデプロイメントの実現**
  - A/Bテストの実施やFBスピードの向上により
    サービス価値の底上げに貢献
- **技術的なプラクティス以外の導入**
  - Scrumやかんばんなど

成長の歩み

成長段階



配属前の状態

サービスってどうやって
動いているの？

プログラミングすれば
動くの？

そもそもプログラミング
って何？

先輩の言っていることが
日本語なのに聞き取れない

一先ず、テスト設計・実行
から一歩ずつ頑張ろう

突然の配属

配属メール見た？

山下君、宜しく

なんのこと？

技術開発部！？

驚くほど分からず、進む場所すら見えない

業務内容

- スクリプトエンジニアとして働ける様に研修
- Javaやテストフレームワークの勉強
- IDEやGitなど基本的なツールに慣れる

ぶつかった壁

- **言っていることすら分からない**状態
- あまりに**膨大な前提知識**に心が折れそうになる
- **どうやったら出来るようになるか**が分からない

帆を張る

解決策

- まず一冊仕上げて、前提・基礎を理解する
- 教わったことは項目別にメモし、分類

考えていたこと

- **受け取れる情報を増やす**ために、浅く広く勉強
- どんなに分厚くても、必ず最後は終わらせられる。

作業者として参画

業務内容

- アーキテクト共に案件に入りテストスクリプト作成
- 準閉鎖環境で前任者が書いたコードを読みとく
- 自動化用のテスト設計書の作成

ぶつかった壁

- 上司が解決する際の**考えているフローが分からない**
- パーツを積み上げても**全体が理解出来る気がしない**
- 頼りのコードが読めない＋**単純な量産活動**

限られた状況でもやれることはある

解決策

- 上司の仕事を分解し、影響範囲が大きい順から勉強
- 営業や常駐業務に同行し実施内容から全体像を理解
- 生産性を上げるためにツールに習熟
- 他人のコードは読んだ後に、自分なりに見ずに書く

考えていたこと

- 必要なのはアーキテクトであって作業者ではない
- **限られた環境でもやれることはある。**

初めてのマネージャー業務に大混乱

業務内容

- 手動テストチームを担当し、総合・受入テストを実施
- 案件管理や顧客折衝を実施
- 同時に設計も実施

ぶつかった壁

- 短納期かつチームでの業務のため、前行程が止まることで全体が一気に混乱し、**管理者にしわ寄せ**
- 考える仕事と管理する仕事を**並行出来ない**
- 上司も先輩もいなくなり**自分しかいない**という状況

裁量＋責任で自分事として捉える

解決策

- どのタスクが今クリティカルかを考える
- きちんと情報を共有し最初から関係者を巻き込む
- どんどん他の人が出来ることは任せる

考えていたこと

- 面倒でも**前提を共有**した方が楽
- **段取りの重要性**を身を持って知る。
- 俺が**最後の砦**だという意識

責任者になった興奮から一転して転落

業務内容

- 上司の支援を得ながら、自動テストを一から組む
- 基盤構築なども少しずつ着手
- 新しい技術を自分で調べて導入

ぶつかった壁

- 労力に対し、自動テストの効果が**部分的な事に失望**
- 同時に**キャリア的にも悩み**始め、**モチベーションが↓**

自分の手で道を切り開く

解決策

- 勉強会＋本で新しい考えを取り入れる（DevOps等）
- アメリカの求人票で世界のトレンドを把握する
- **お客様に提案**、勉強会に行ってCTOに提案

考えていたこと

- 部分最適を幾らやっても**全体最適にはならない**
- 本質的な価値は、顧客に届く価値 = スループット
- **スループットを最大化**するものに注力しよう

点からフローへの挑戦

業務内容

- 開発環境の改善に着手
- 継続的デリバリー環境の構築

ぶつかった壁

- **見積もれない**タスクの数々
- 度重なる予見外のトラブル（技術的、PJ的）
- 関係者の拡大による**調整業務の増大**
- 期待値を勝手に高くし、**1人で頑張り潰れそう**になる。

思いを形にするのは自分。でも抱え込まない

解決策

- **素直に事情を伝え**、調整してもらう。
- 自分が**やるべきところに集中**する
- ベース部分は**先取りして準備**する

考えていたこと

- 提案を実現するのは自分。責任を持ってやりきる
- ただ、**一人相撲では解決しない**
- 目の前の問題解決だけにのめり込むと悪循環に陥る

成長要因の分析

内的要因

外的要因

内的要因

1. 時期にあった適切な自己目標の設定
2. 自分にあった勉強方法の確立
3. ブレない、曲げない、諦めない

外的要因

1. 必要なのはスポンサー
2. 見習う人達がいるということ
3. 挑戦してもいいという安心感

共通項

内発的**動機**

よくある思い込み

高い給料やポジションがあれば、人はやる気を出す

失敗した時には**責任を明確化**することで、当事者意識を持たせる

仕事には緊張感が大事。**プレッシャー**を掛けて緊張感を持たせる

短期的な効果

業務中のみ

22%

業務時間は実は僅かしかない

$$\frac{240d}{360d} \cdot \frac{8h}{24h} = 22\%$$

内発的動機こそが自律性と持続力の源泉

- 自分の**興味関心**や、自分自身が**楽しい嬉しい**と思う事
  - 感謝されるのが嬉しい
  - 新しいこと考えたり学んだりするのが楽しい
  - 自分で仕事をグリップ出来ていると実感すると楽しい

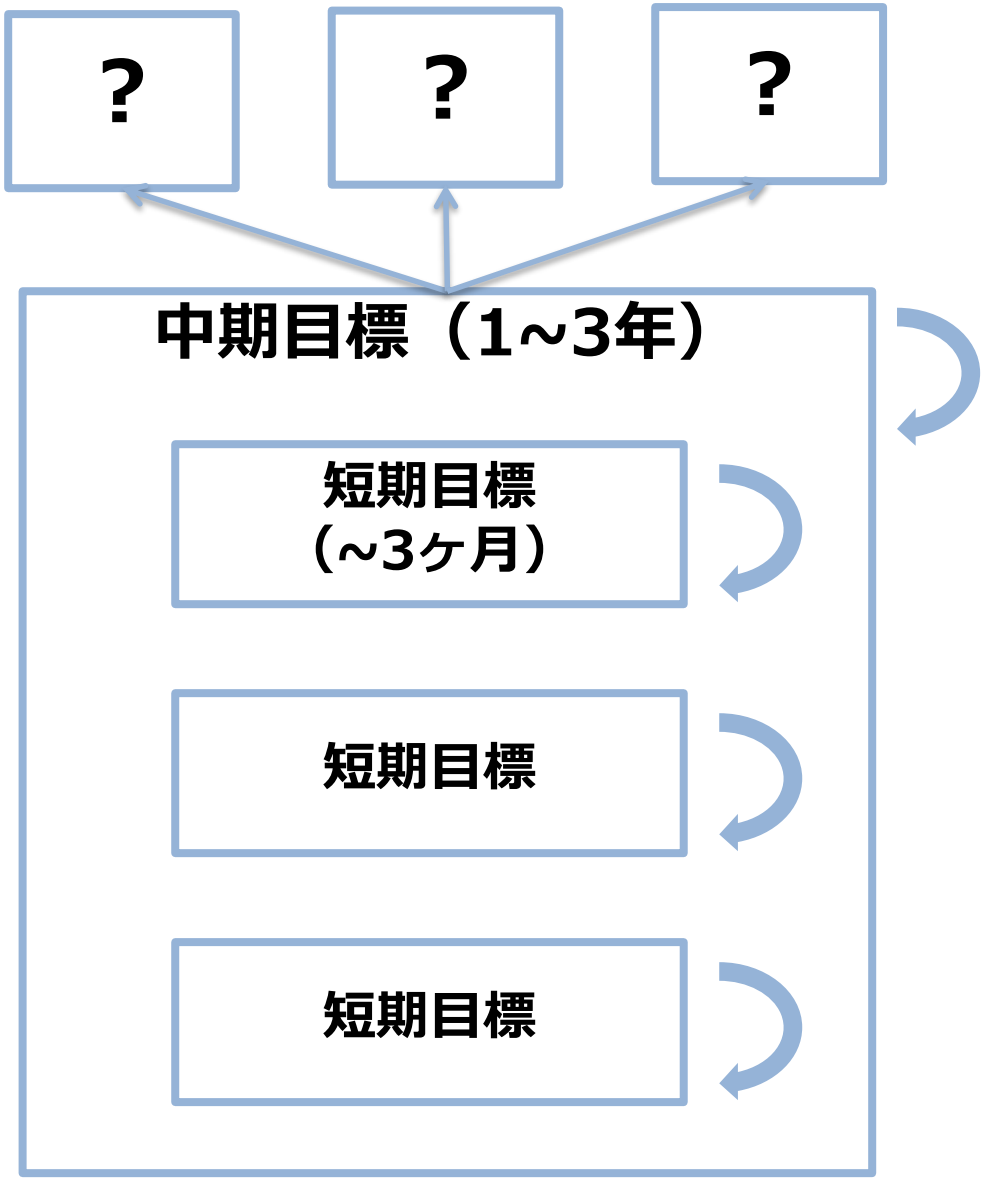
自ら長期的に業務以外でも取り組む

内的要因の分析

内的要因

1. **時期にあった適切な自己目標の設定**
2. 自分にあった勉強方法の確立
3. ブレない、曲げない、諦めない

目標設定の概念図



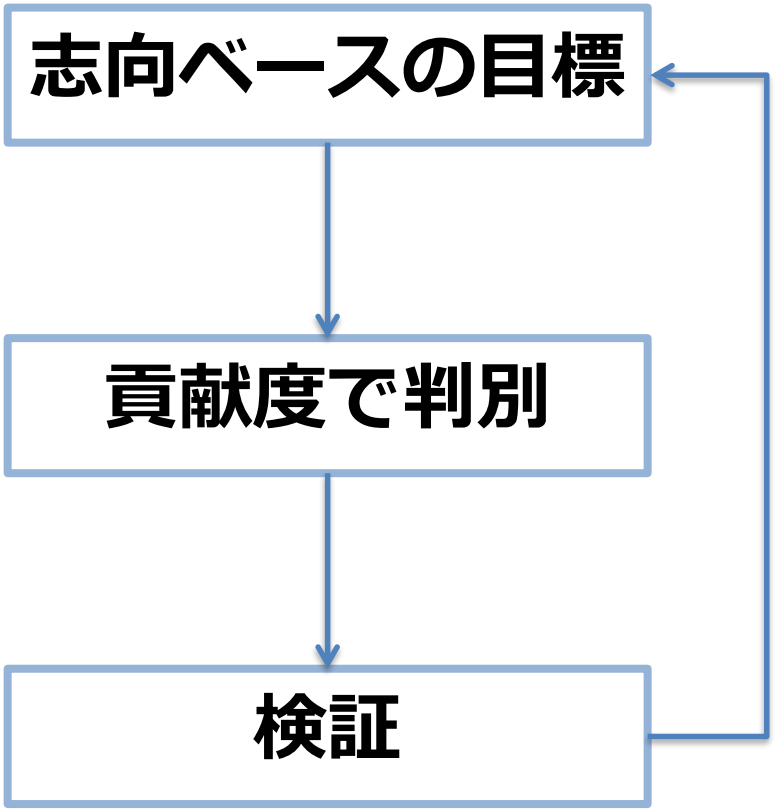
- 長期目標は決めない
- 中期目標のを軸に
  複数の短期目標の設定
- 目標は新しく設定するの
  ではなく「改善」する

長期目標は敢えて明確化しない

- **Planned Happen Stance Theory**
  - 計画された偶然性理論
  - キャリア的に成功する人の多くはキャリア的跳躍がある
- **前提の崩壊**
  - 産業構造の変化による業界自体の消滅
  - 技術革新によるスキルの陳腐化
  - 自らの嗜好の変化

積み上げ式の崩壊

自らの志向と貢献度をリンクさせる



ワクワクする目標を立てる

顧客や会社に
本質的に貢献するのか？

事実ベースの情報を集める
顧客や上司に提案、**FB**を得る

自分がどうなっていたいか

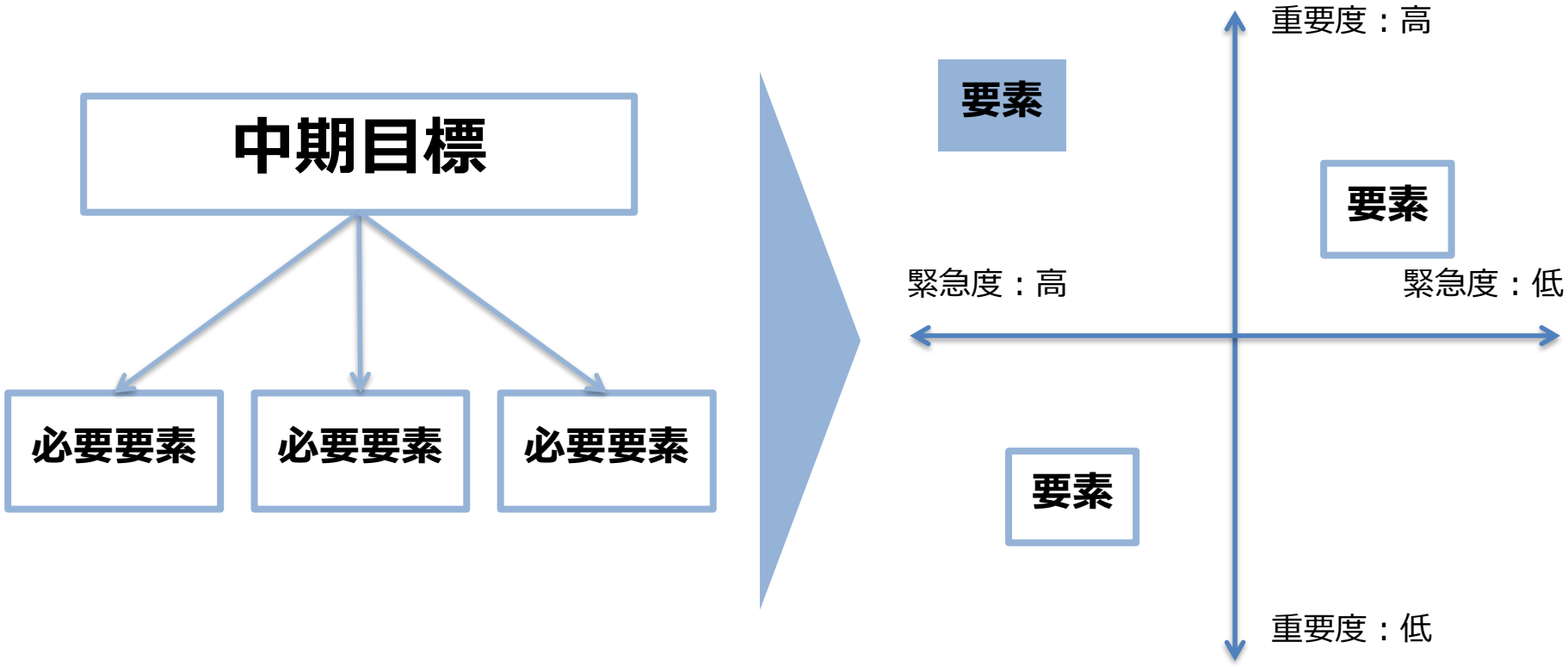
1. テスト自動化アーキテクトになる（今まで）

自動化するために必要な業務を
一気通貫で出来る様になり、自分の担当を持つ

2. DevOpsエンジニアになる（これから）

引き続き、コアとなるテスト自動化のスキルを高めながら
開発フロー全体を改善し、サービス価値向上に貢献する

中期目標を分解し、二軸で整理



終了条件が明確にして達成感を得る

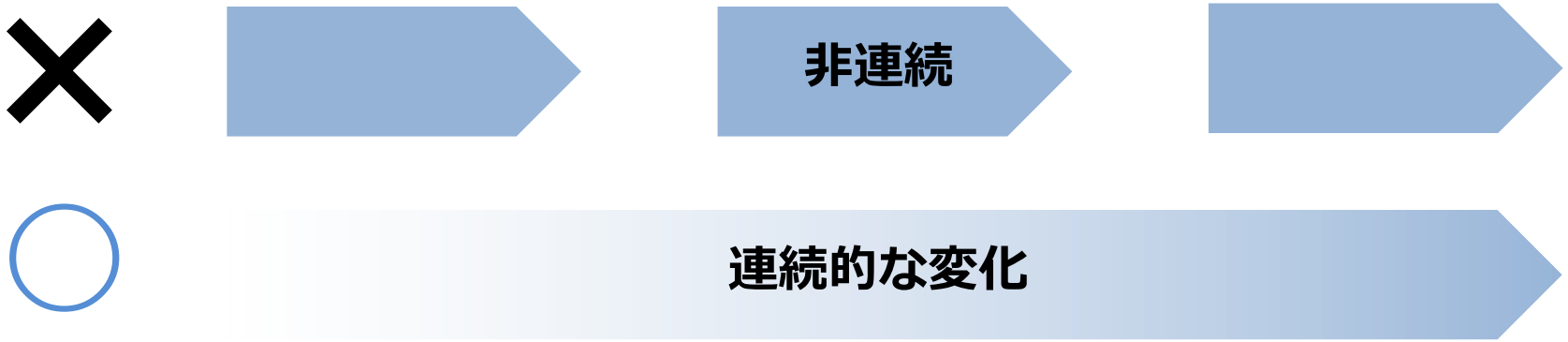
1. テスト自動化アーキテクト

- ProgateやDotinstallの○○の講座を終わらせる
- LinuxでWebサーバーを立て、WordPressを動かす
- テストの概念を根拠を持って説明出来る様にJSTQBを取る

2. DevOpsエンジニア

- DevOpsやAgileの概念を理解するために1冊ずつ読む
- Ansibleで簡単な開発環境（WEB+DB）を作成する
- 開発側を理解するためにチュートリアルでサービスを作る

状況の変化に合わせて随時目標を変更



「改善」

内的要因

1. 時期にあった適切な自己目標の設定

2. 自分にあった勉強方法の確立

3. ブレない、曲げない、諦めない

自分のことを理解する

モチベーション

- 目標とその理由が必要
- やらされ感もNG

→ **アレンジ**

理解の仕方

- 全体像が見えないと理解度⬇
- 並列しての勉強は集中度⬇

→ **全体理解**

達成感

段階別に分類し、個別に対策

概要

- ネット情報を中心に勉強
- どういった技術・スキルか
  他の技術や思想との関連を把握

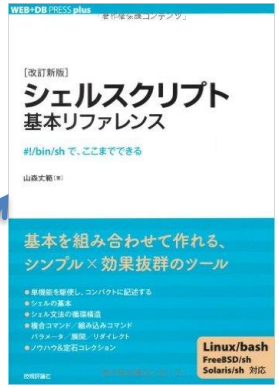
基礎

- 薄いが**全体を理解**出来る本を一冊
- dotinstallやprogateなど、サイトで学習

応用

- 本格的な本を**実際に手を動かして**勉強
- 関連技術もさらう感じで勉強

Linuxの場合



ドットインストール

sed & awk

内的要因

1. 時期にあった適切な自己目標の設定

2. 自分にあった勉強方法の確立

3. ブレない、曲げない、諦めない

無理をしない、でも粘り強く愚直にやる

- 自分と合わない目標・考えや勉強法は
  根性では乗り越えられないと知る
- **完璧な環境なんてない**
  諦めずに少しでも近づけていく
- 凄そうな人も自分も、**どちらも人間**

決めたら、後は愚直にやるのみ


Linux サーバー
Web API: The Good Parts
Selenium
TCP/IP
基本情報
過去問題集
Web を支える技術
Selenium 実践入門
DevOps
Slack
JavaScript本格入門 山田祥寸
Docker
Selenium WebDriver
Java
Java
Linux
Amazon Web Services
PHP + MySQL
HHKB Professional BT

55 + 20 =

75 冊

まとめ

ワクワク＋貢献出来る**目標**を

自分にあった**勉強法**で

達成に**邁進**する

外的要因の分析

1. 必要なのはスポンサー

2. 見習う人達がいるということ

3. 挑戦してもいいという安心感

理解し、チャンスも与えられる人が必要

○ 悩みや希望を聞いてあげる

◎ 悩みの解決や希望に繋がる**環境**を与える

1．自律的に動ける環境

1．多様な環境

1. 自律的に動ける環境

1. 多様な環境

都合よくはいかない

やらされ感

なんでやるの？

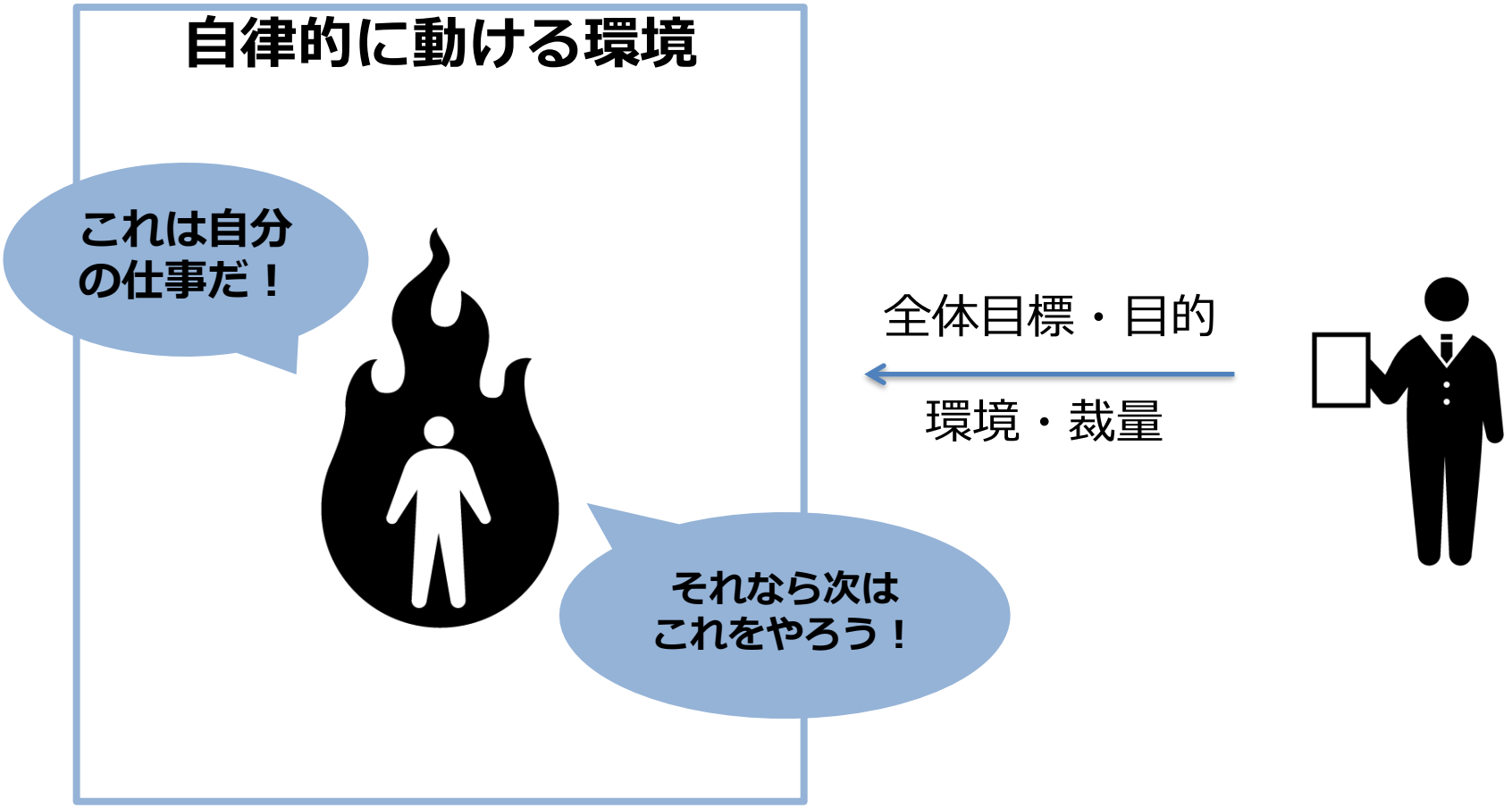
どこまで
やっていいの？

あれやれ
これやれ

つべこべ
言うな

頭使って
自分から動け

判断軸と試行錯誤の場を与える



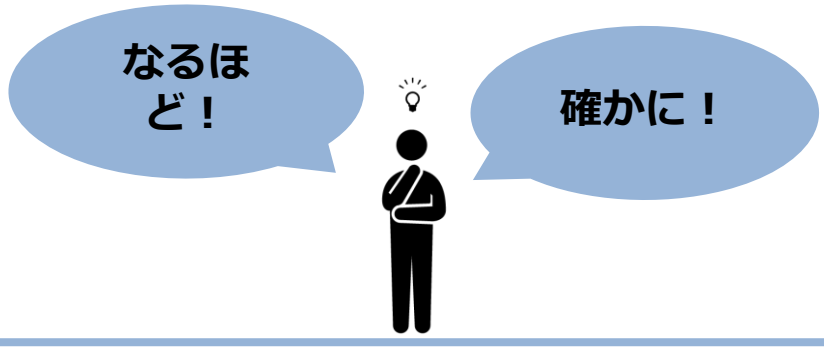
1. 自律的に動ける環境

1. 多様な環境

狙った様に成長するわけではない

体験・暗黙知

多様な環境 → 暗黙知の共有



今まで経験した環境

1. **自動化案件**
   - 大手 ★
   - WEB
2. **QA業務 ★**
   - 設計・実行
   - 案件管理
3. **プリセールス ★**
4. **全社イベント**
5. **採用 ★**

★： 他部署と関連

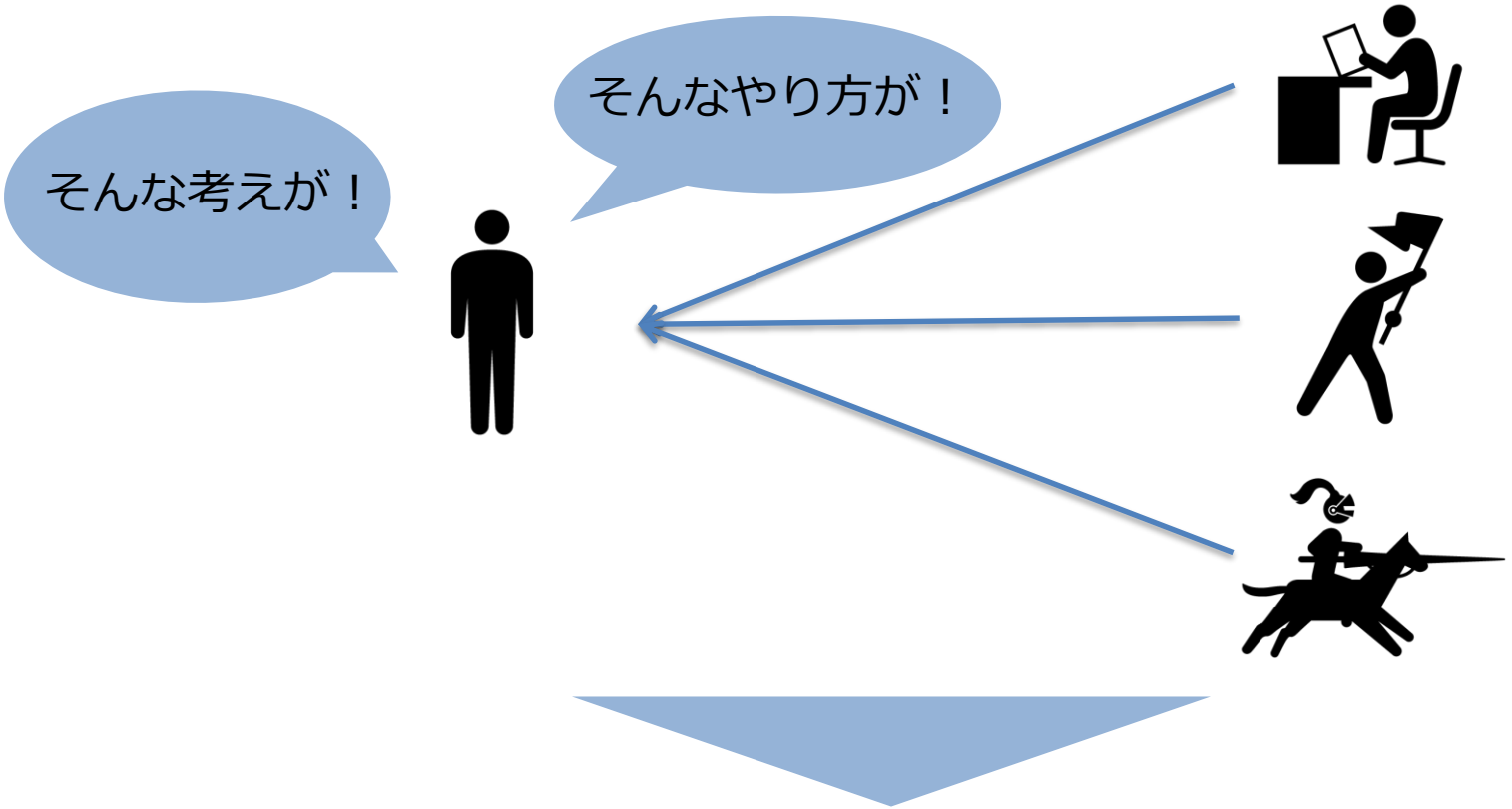
外的要因

1. 必要なのはスポンサー

2. 見習う人達がいるということ

3. 挑戦してもいいという安心感

若手は先輩達の背中を見て育つ



異質な現場を複数体験させる

今までの出会い

Seleniumデザインパターン 監修など

営業・案件管理・アーキテクト何でもこなす

直接の上司で仕事の根本から教わる

システムテスト自動化 標準ガイド 監修

実践Appium 監訳など 他多数

強いプロ意識＋勉強の鬼

コアテクノロジーBU G2 グループ長

理論的かつ情緒的。強いエンパワーメント

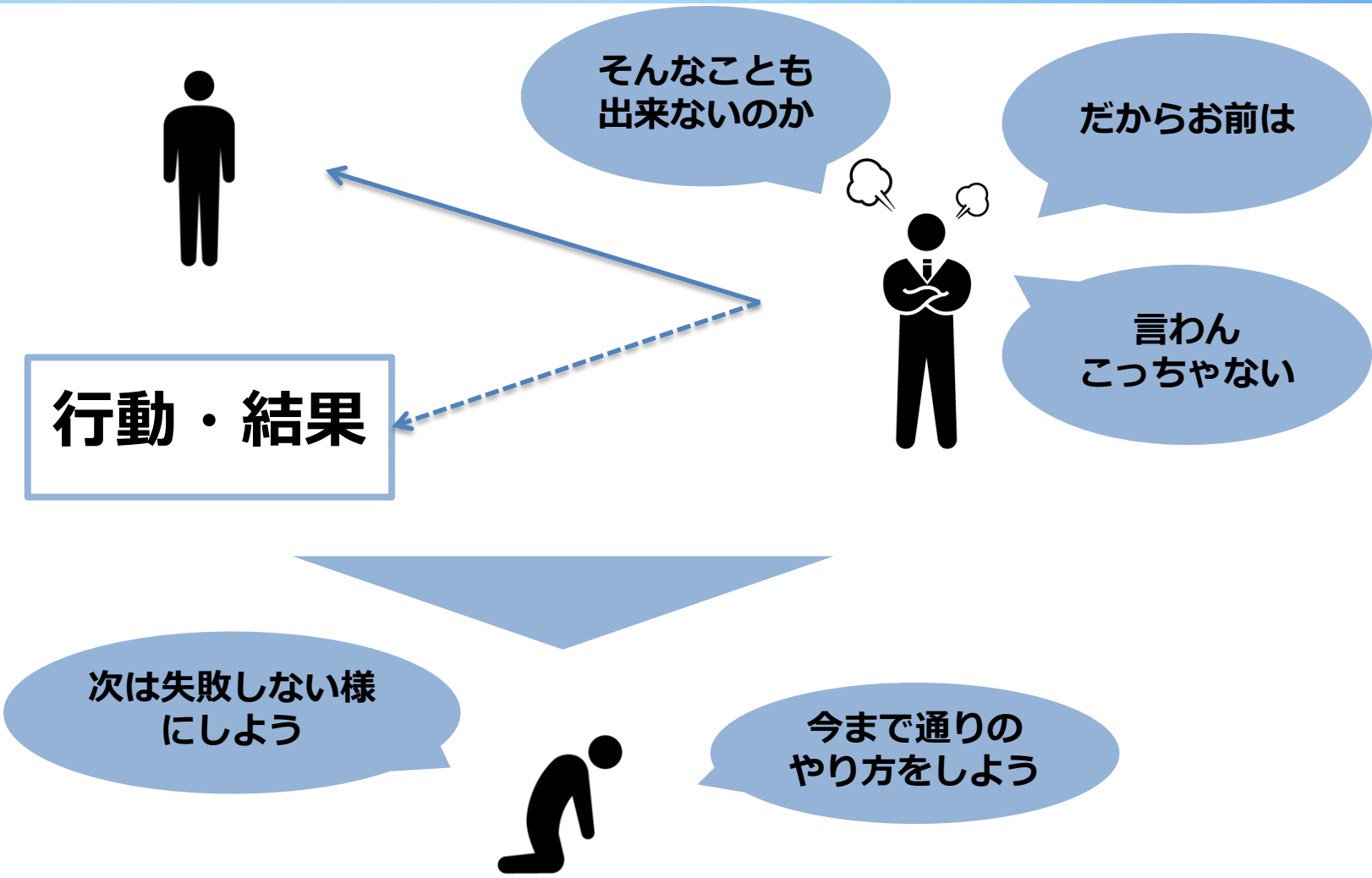
外的要因

1. 必要なのはスポンサー

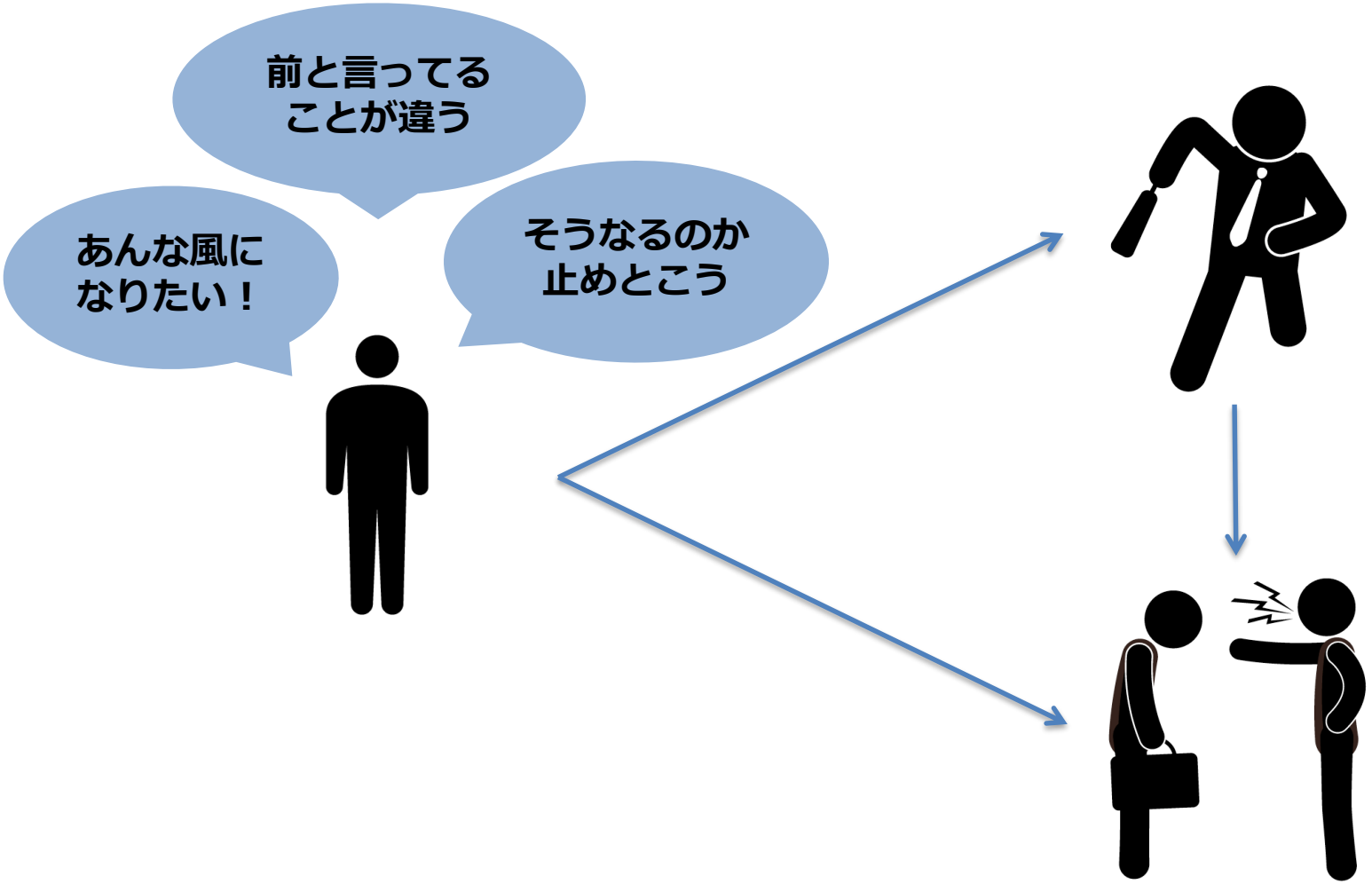
2. 見習う人達がいるということ

3. 挑戦してもいいという安心感

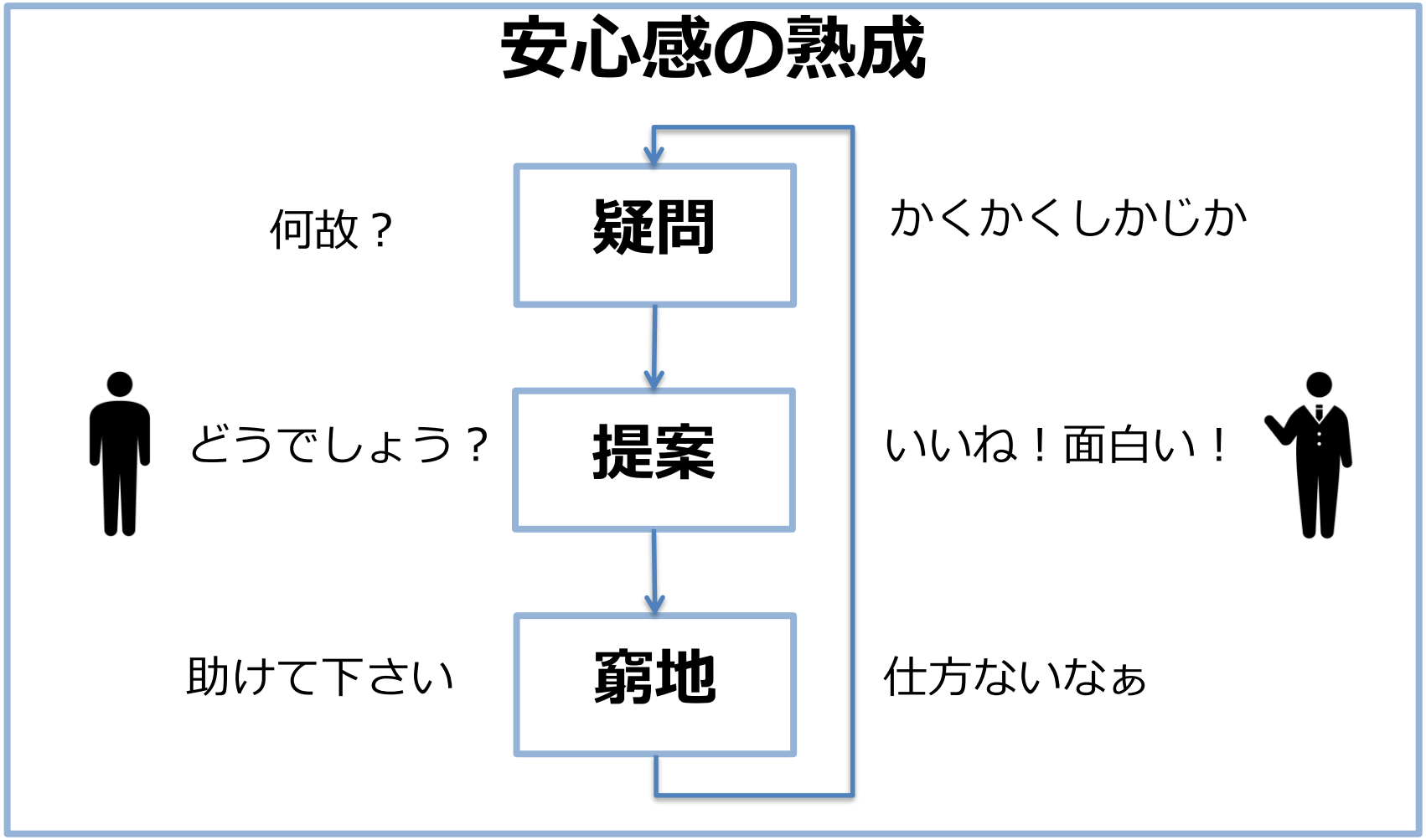
若手はいつも不安


そんなことも
出来ないのか
だからお前は
言わん
こっちゃない
行動・結果
次は失敗しない様
にしよう
今まで通りの
やり方をしよう

周りのこともよく見ています。



安心感こそ最高のガソリン



まとめ

判断軸となる

全体目標と目的を与えた上で、

自律的に動ける**多様**な環境を

用意し、**見守る**

最後に

本公演のポイント

- 成長の流れ
- モチベーションの維持
- 具体的な勉強法
- 助けられた経験・良かった環境

新しい挑戦をされる皆様

ご自身の中で**ワクワク**することは？

マネージャーの皆様

部下は、**ふとした言葉**に影響を受けています

ご清聴ありがとうございました