行き詰まらないための運用自動化

～全自動化する体力がない僕らの戦い方～

いらにか（@happy_packet)

自己紹介

~iranika~

いらにか

#Contact

Twitter : @happy_packet
Discord : iranika109#1724
E-Mail : iranica109@gmail.com

注記

- 自動化と全自動化を区別して表記しています。
  ✓自動化≒オペレーションにまだ人が介在する。一部自動化。
  ✓全自動化≒オペレーションに人が介在しない。
  ✓ニュアンス的には、以下のようなイメージで使い分けています
    - ◆ 自動化は「フローチャート上のプロセスに当たる処理だけの実装」
    - ◆ 全自動化は「フローチャートのすべての要素を実装」
    - ◆ 全自動化 ＝ (自動化 × プロセス数) + フロー制御

行き詰まらないための運用自動化

PHASE.1 考える

事の発端

偉い人「自動化して工数削減！！」
上長「わかりました！！」
ぼく「人員とか予算とかは？」
偉い人・上長「そんなモノはない」
ぼく「！？！？」

そして僕の自動化が始まった。
以上。

無いものがいっぱい

- プログラマ(Coder)←重要
- 自動化に関するノウハウ←重要
- 時間
- 予算
- 度胸と責任者(変更は運用にとって脅威)

無いなら、有るものでカバーするしかない！！
そういうときは、そもそも論で要件を見つめ直してみる

そもそも論で要件を見つめ直す

- 自動化って言うけどやるべきなのは全自動化？一部自動化？
- そもそも僕らがやりたいのは本当に自動化なのか？
- 本当に欲しいのは自動化の先に得られる運用負荷軽減では？

- 多角的に状況を観察して、コストパフォマンス重視で取り組まないと、無い無い尽くしの僕らでは体力が持たない。
- というか、今の僕らでは全自動化は無理。
- 一部自動化ならできそう（以後、自動化≒一部自動化）

運用負荷軽減につながること

- オペレーションダイエット（業務整理）
  - ✓作業を見直して手順修正など簡易な変更で削減できる部分は徹底的に削る
  - ✓人がやるべきではない作業（大量の情報処理等）はツールを作って対処
- 自動化
  - ✓人がやるべきではない作業をソフトウェア（広義）にやらせる。
  - ✓ただしソフトウェアは保守が必要という事を忘れてはいけない。
  - ✓「自動化後の運用工数 + ソフトウェア保守工数 = 総工数」
  - ✓自動化して総工数が増えることもある（安易な自動化はダメ絶対）
- 生産性の高い優秀な人材
  - ✓生産性（処理能力）が向上すれば相対的に負荷が軽減する
  - ✓新規雇用か人材育成で手に入れる

たぶんこんな感じ？

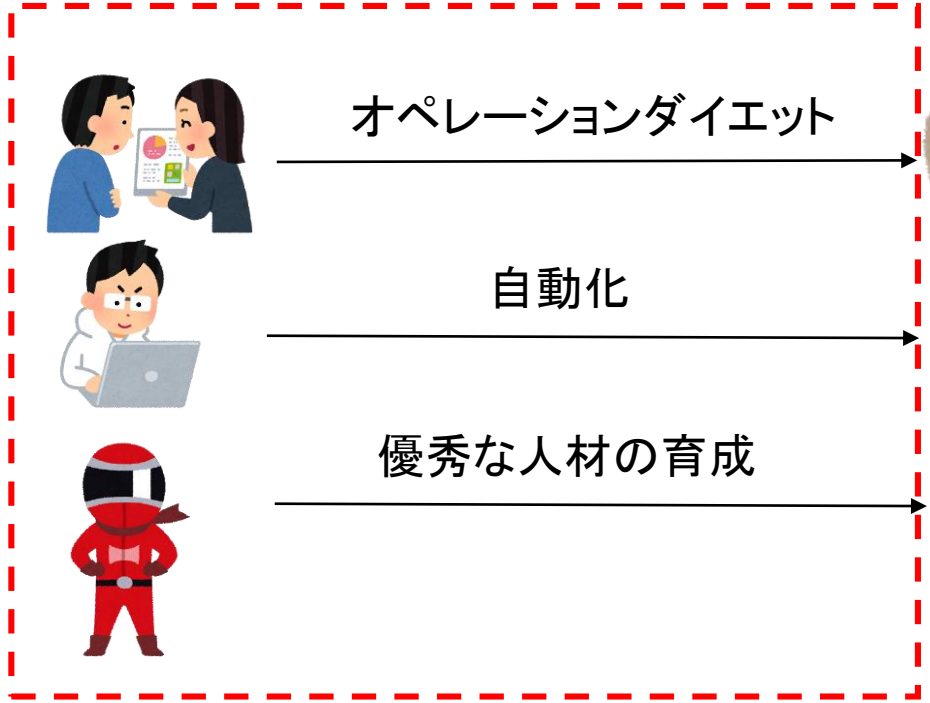
オペレーションダイエット

自動化

優秀な人材の育成

肥え太った運用の怪物

たぶんこんな感じ？



肥え太った運用の怪物

今日はここの話

行き詰まらないための運用自動化

PHASE.2 オペレーションダイエット

オペレーションダイエットのイメージ

余分な脂肪

重く鈍い筋肉

肥え太った運用の怪物

低脂肪

軽く俊敏な筋肉

スリムで俊敏な運用の怪物

オペレーションダイエットの狙い

- 不要な作業・要素を排除して、余計な脂肪を減らす。
- オペレーションが体重の何%くらい占めているのか把握する。
  - ✓ 絞りたい部位（業務）に優先順位がつけられる。効果的な部位から優先してダイエット。

- 作業にツール等を取り入れ、軽くて俊敏な筋肉を手に入れる。
  - ✓ ツールによる負荷軽減は、体重が減るわけでない。筋力が上がって体が軽く感じるだけ。
  - ✓ 厳密に言うとツールは強化外骨格のようなサポーター。
  - ✓ 自動化では最終的に強化外骨格も体の一部として扱うので広義の意味で筋肉と呼称。
  - ✓ ツールを失うと軽減されていた負荷ものしかかってくる辛さがある。
  - ✓ 当然ツールの重量（保守工数等）だけ体重も増える。下手すると脂肪より厄介になる。
  - ✓ ちなみに狭義の筋肉≒オペレーターの処理能力。

オペレーションダイエットの教訓

- 脂肪を減らすことが一番良いダイエット。
- ツール導入で体重は減らない。筋力アップによって体が軽く感じるだけ。
- ツールに頼りすぎると失ったときに自重を支えられずに歩けなくなる。

行き詰まらないための運用自動化

PHASE.3 とりあえず決めて、とりあえずやってみる

とりあえず方角と第一目標地点を決める

- 自動化した未来が具体的に想像できるなら自動化は出来る
- 具体的に想像できないなら、トライアンドエラーを繰り返すしか無い
- とりあえず方角（方向性）と第一目標地点を決める。

今回のケースの方向性

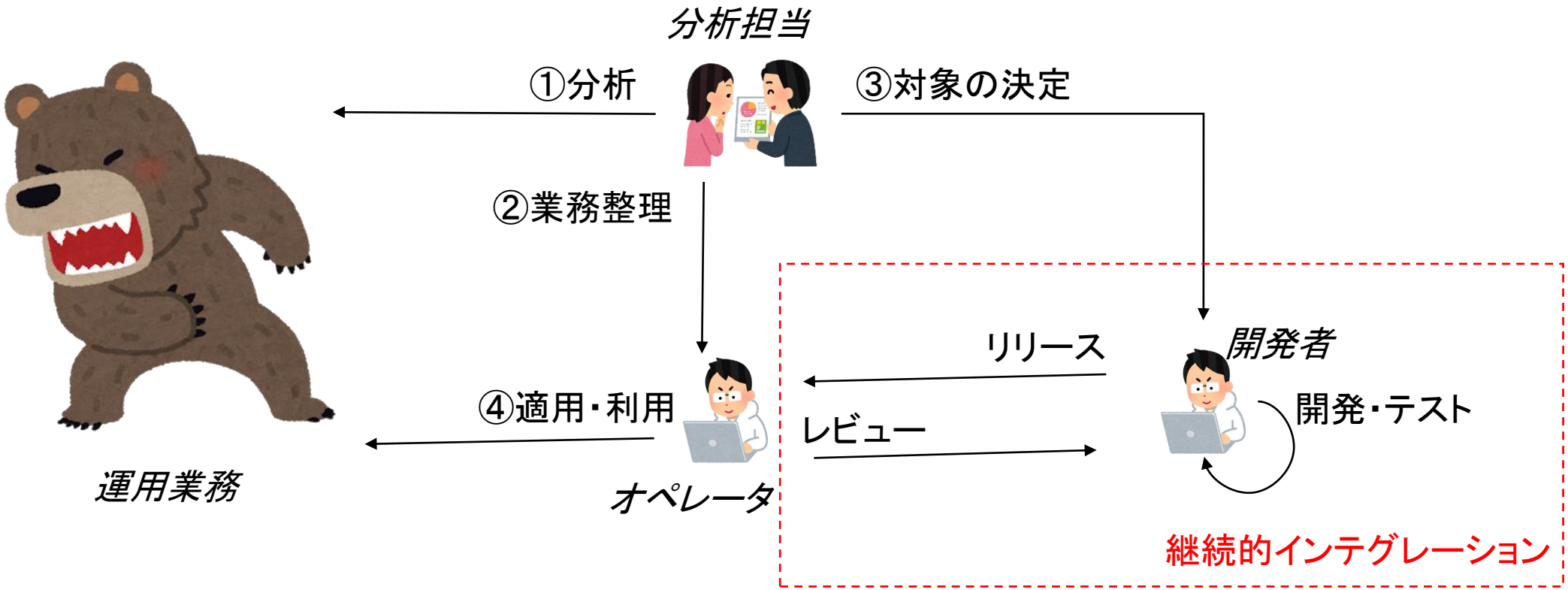
- 全自動化する体力がそもそもないので、業務整理しつつ一部だけ自動化。
  - ✓イベント＆トリガーが煩雑であれば、管理を人に任せる
  - ✓アクション部分だけを自動化（スクリプト化）する。
  - ✓ゆくゆくはイベント＆トリガー管理もジョブ管理ツールなどに任せることを想定する
- いつでもやめられるように自動→手動に切り替えられるようにする
- プランより先にプロトタイプを作ってみる
  - ✓プロトタイプをレビューするとどのようなデザインがいいか具体的に見えてくる
  - ✓どのように使わせるかデザインを意識してプロトタイプを作る
  - ✓プロトタイプは小さく作り、開発速度を重視
- プレビューリリースで早めに使い始めて成果を出すようにする
  - ✓機能拡張や品質向上は後から。

今回のケースの第一目標地点

- 業務整理しつつ自動化対象の選定をする（最低3案件を3ヶ月以内にダイエット）
- 自動化対象の早期プレビューリリース（選定から1ヶ月以内を目標）

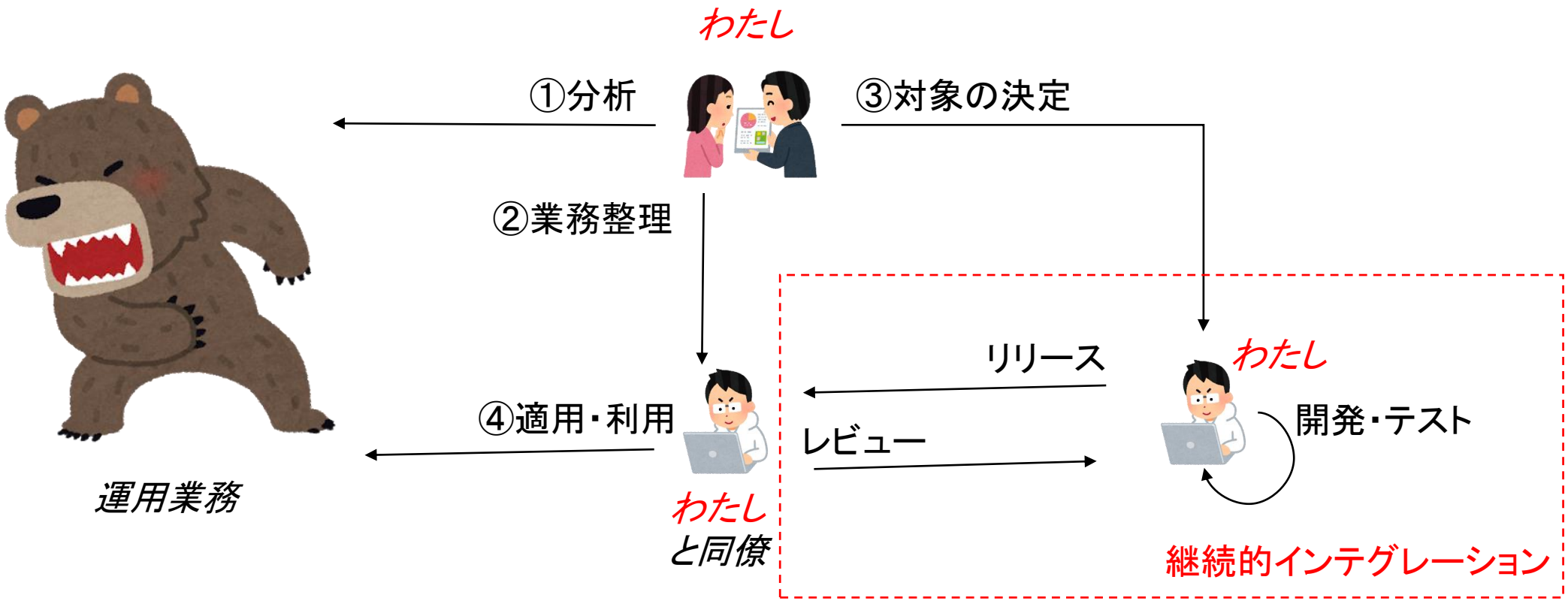
今回のイメージ

これ↓を並列稼働させる（とりあえず3ライン稼働）



今回のイメージ（配役）

これ↓を並列稼働させる（とりあえず3ライン稼働）



やってみた感想

■良かった点

- 並列稼働で業務ごとに分散して進めることが出来た。
  - ✓複数チーム作れるなら稼働ラインも増やせる。
- 小出しで早めに新しい手順やリリースを使い始めることで、成果も早く出る。
  - ✓成果として体感できると、モチベーションにもつながる。
  - ✓要望や課題などがどんどん見つかって、倒すのがだんだん楽しくなってくる。

■大変だった点

- 分析担当は自動化後のビジョンが見える人じゃないと難しい。（設計・要件定義）
- 頻繁に運用変更（適応）が発生するので展開に少し苦労した。（反映コスト）
  - ✓ツールの配布方法、手順の展開等…
  - ✓この点に関しては、どう使わせるのかデザインを工夫することで少し楽にできた。

行き詰まらないための運用自動化

PHASE.4 壊して新しく作る

さらに見えてきたビジョン

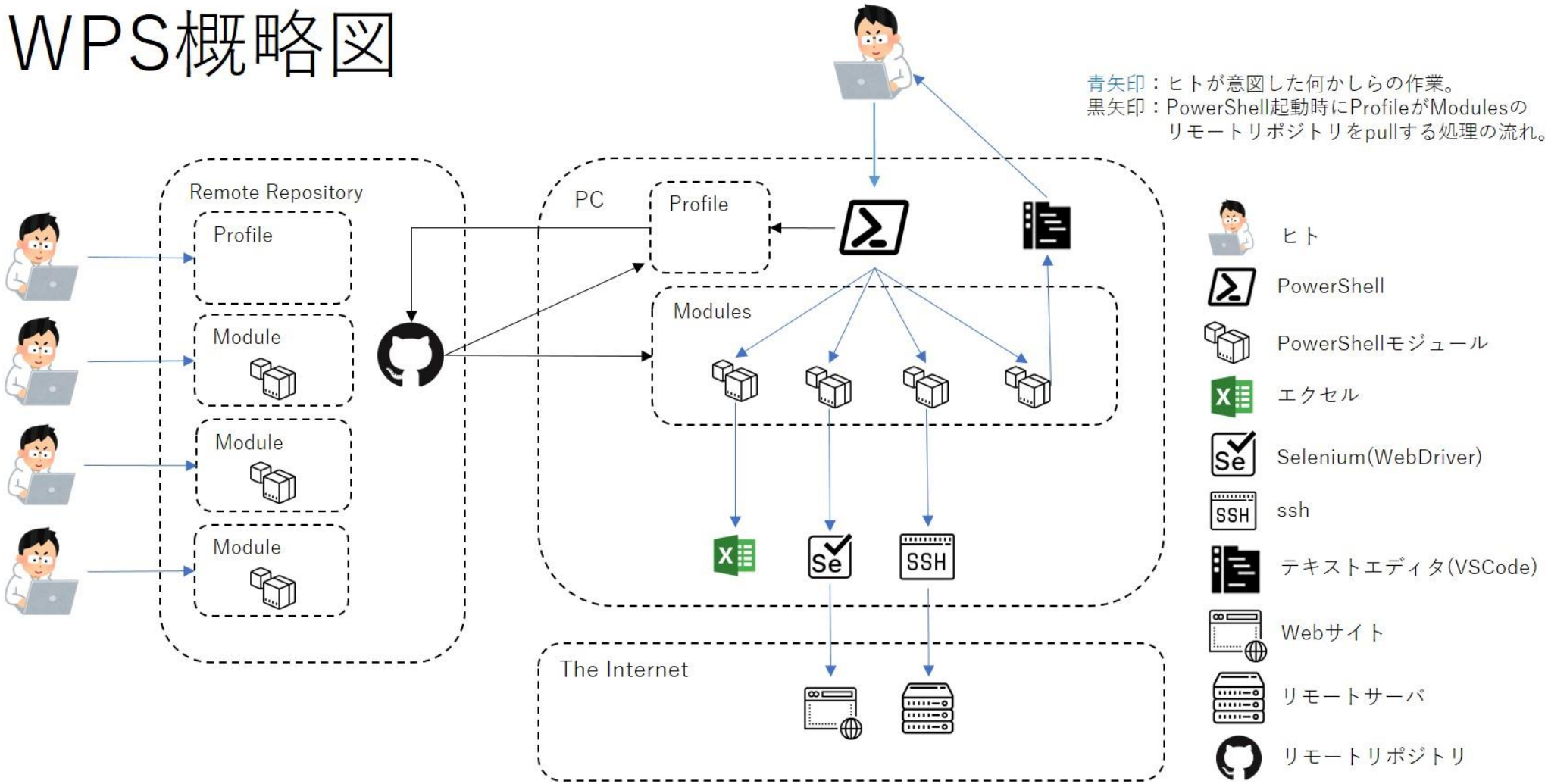
- とりあえず決めた方向性は間違っていないと感じた。
  - ✓全自動化する体力がそもそもないので、業務整理しつつ一部だけ自動化。
  - ✓いつでもやめられるように自動→手動に切り替えられるようにする
  - ✓プランより先にプロトタイプを作ってみる
  - ✓プレビューリリースで早めに使い始めて成果を出すようにする
- 以下の方向性も重要なので追加することに
  - ✓だれでも使い始めやすいようにする（導入コスト）
  - ✓だれでも使いやすいようにする（UI/UXの向上）
  - ✓壊して作ることに抵抗が生まれないような何かがほしい（変更のフットワークを軽く）
- 新たな要望や課題
  - ✓プロトタイプ作成とプレビューリリース、CIを迅速かつ楽にやりたい
  - ✓リリースの配布を簡単かつ迅速にできる仕組みがほしい
  - ✓使い方とかの口頭説明をなるべく減らしたい
  - ✓自動でオペレーション遂行不可な場合に、誰でも手動でオペ可能な状態にしたい

さらに見えてきたビジョン

- とりあえず決めた方向性は間違っていないと感じた。
  - ✓全自動化する体力がそもそもないので、業務整理しつつ一部だけ自動化。
  - ✓いつでもやめられるように自動→手動に切り替えられるようにする
  - ✓プランより先にプロトタイプを作ってみる
  - ✓プレビューリリースで早めに使い始めて成果を出すようにする
- 以下の方向性も重要なので追加することに
  - ✓だれでも使い始めやすいようにする（導入コスト）
  - ✓だれでも使いやすいようにする（UI/UXの向上）
  - ✓壊して作ることに抵抗が生まれないような何かがほしい（変更のフットワークを軽く）
- 新たな要望や課題
  - ✓プロトタイプ作成とプレビューリリース、CIを迅速かつ楽にやりたい
  - ✓リリースの配布を簡単かつ迅速にできる仕組みがほしい
  - ✓使い方とかの口頭説明をなるべく減らしたい
  - ✓自動でオペレーション遂行不可な場合に、誰でも手動でオペ可能な状態にしたい

とりあえず思いついた仕組み

WPS概略図



なぜPowerShell?

- 単なる思いつき。
  - ✓簡単に実装できそうだったし…
  - ✓どうせ職場にはWindowsユーザしかいないし…
  - ✓WindowsならPowerShell標準搭載してるし…
  - ✓どうせみんなPowerShell活用してないだろうし…
  - ✓プロファイルとか書き換えても致命的な影響を被る人いなさそうだし…
  - ✓マルチプラットフォーム対応したくなったらPowerShell Core使えば…
- 別にBashでも似たようなことは出来る。
- いつでもやめる気でいるし好きにやらせろ。

WPSの嬉しいところ（一部抜粋）

- PowerShellモジュール(関数をコマンド)として配布できる
  - ✓ヘルプトピックが書きやすい
  - ✓Man(Get-Help)でツールの説明や引数、実行例などを参照させられる
  - ✓引数のバリデーションが簡単に記述できる（param文）
  - ✓UI部分だけPowerShellで書いて、コア処理を他言語で書いた実行バイナリに任せたり、設計の自由度が高い。
- Gitの導入を強制
  - ✓ユーザが開発に参加できる！！（するとは言っていない）

新しい発見～便利なヘルプトピック～

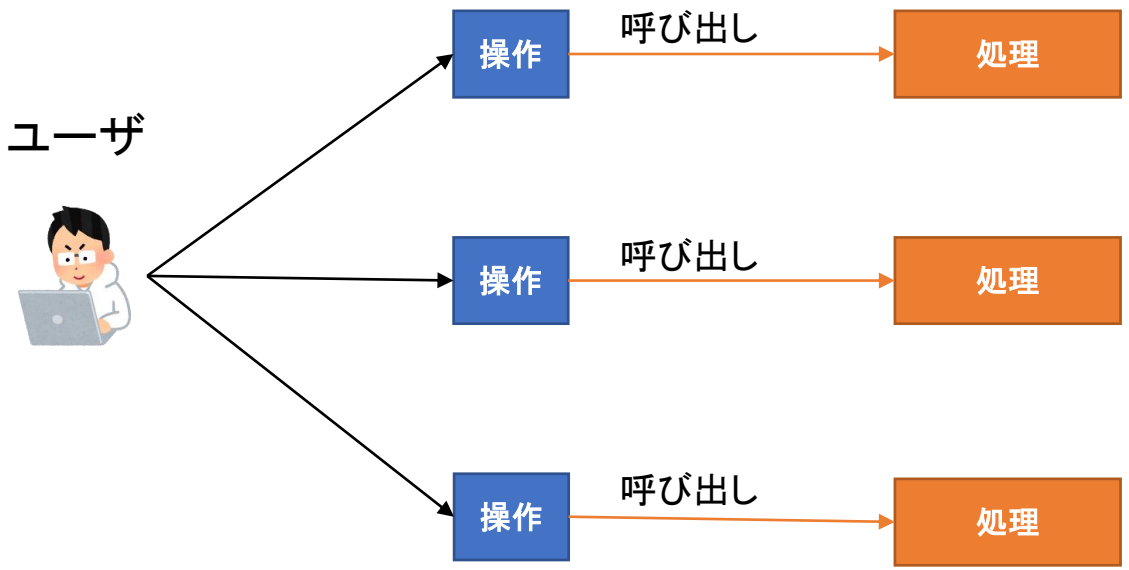
- man(Get-Help)でツールの情報が参照できる（Unix文化の導入）
- 関数にヘルプトピックをコメントとして記述できる。
  - ✓コードと一緒に管理ができるのは開発者として嬉しい
  - ✓コメントエリアなので開発者以外に編集させるのも抵抗感が少ない
- 引数の型など一部情報は勝手に解析してくれる。

新しい発見～オプション引数～

- PowerShellのおかげでオプション引数の実装がしやすくなった
- 共通オプションによるUXの向上（例）
  - ✓OpenDoc：使用方法の手順書を開く
  - ✓OpenSyudoc: ツールと同様のことを手動でやる場合の手順書を開く
  - ✓ContactUs: ツールに関するお問い合わせ先を表示する
  - ✓UseConsole: 対話形式でコンソールから情報を入力する
- 安易な気持ち（ノリ）で試験導入しやすくなった。
- あまり手間を掛けてないので廃止しても心へのダメージが少ない。
- よりUI処理とコア処理を分離して考えるようになった。

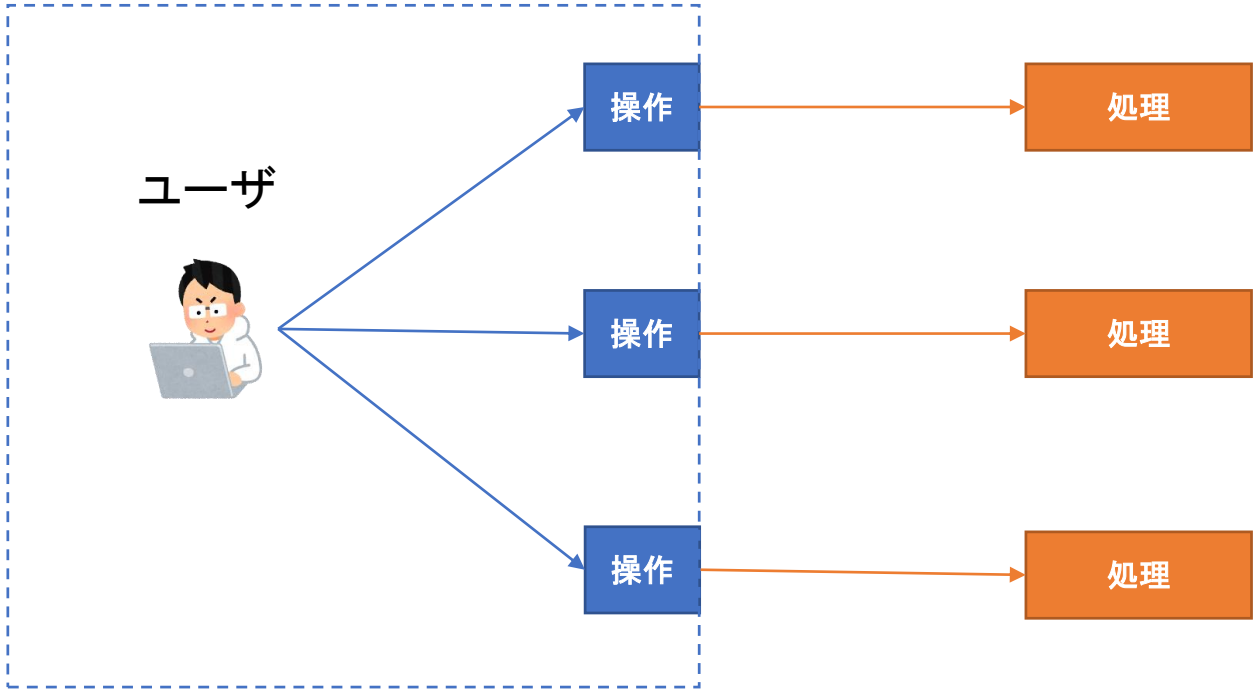
新しい発見～操作のデザイン～

操作を経由して処理が呼び出される！！（当たり前）



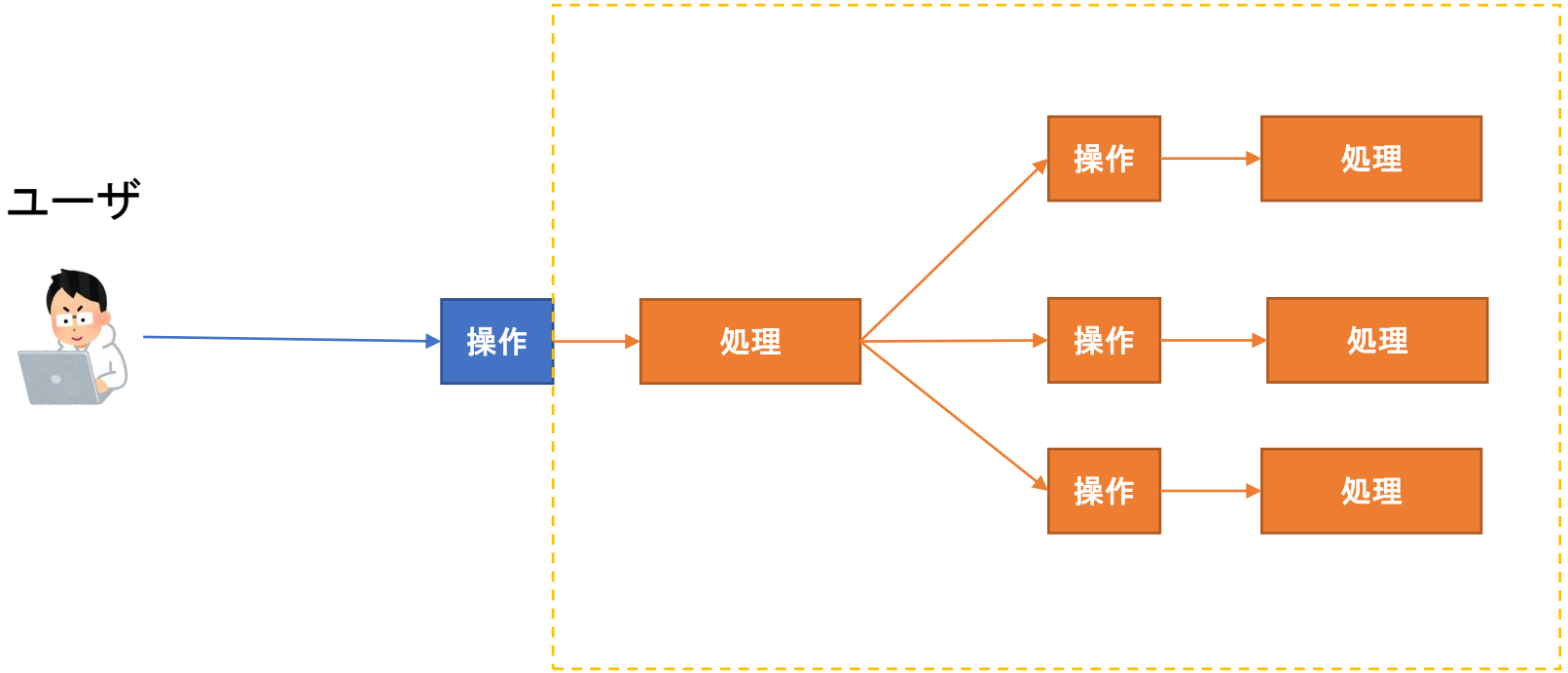
新しい発見～操作のデザイン～

ユーザが意識するのは操作部分まで。
どのように処理が呼び出されるかは意識する必要はない。



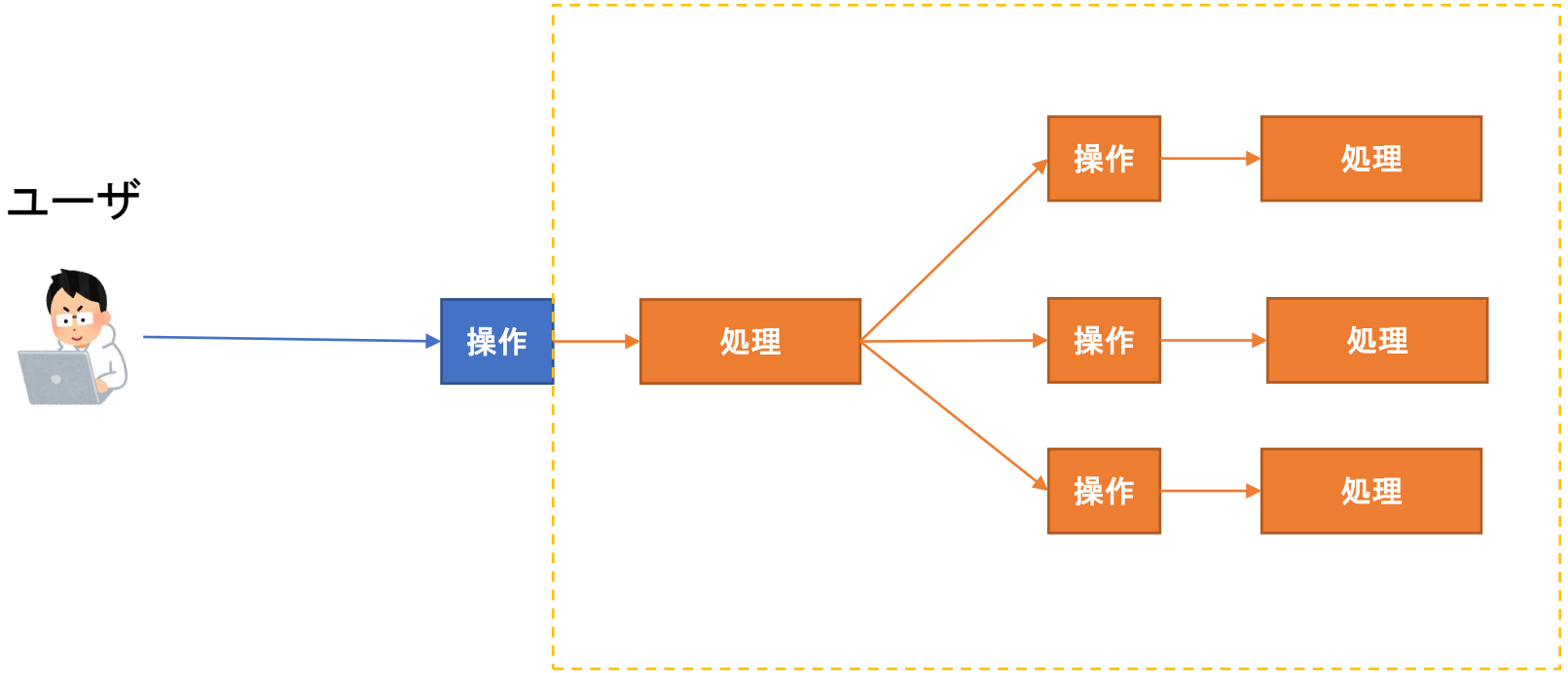
新しい発見～操作のデザイン～

なので操作の裏で操作→処理をやってもユーザは気にならない。
ユーザが操作で手に入れたいのは処理が生み出す成果物。



新しい発見～操作のデザイン～

Unix哲学が活かせる！！



さらに見えてきたビジョン（再掲）

- とりあえず決めた方向性は間違っていないと感じた。
  - ✓全自動化する体力がそもそもないので、業務整理しつつ一部だけ自動化。
  - ✓いつでもやめられるように自動→手動に切り替えられるようにする
  - ✓プランより先にプロトタイプを作ってみる
  - ✓プレビューリリースで早めに使い始めて成果を出すようにする
- 以下の方向性も重要なので追加することに
  - ✓だれでも使い始めやすいようにする（導入コスト）
  - ✓だれでも使いやすいようにする（UI/UXの向上）
  - ✓壊して作ることに抵抗が生まれないような何かがほしい（変更のフットワークを軽く）
- 新たな要望や課題
  - ✓プロトタイプ作成とプレビューリリース、CIを迅速かつ楽にやりたい
  - ✓リリースの配布を簡単かつ迅速にできる仕組みがほしい
  - ✓使い方とかの口頭説明をなるべく減らしたい
  - ✓自動でオペレーション遂行不可な場合に、誰でも手動でオペ可能な状態にしたい

さらに見えてきたビジョン（再掲）

- とりあえず決めた方向性は間違っていないと感じた。
  - ✓全自動化する体力がそもそもないので、業務整理しつつ一部だけ自動化。
  - ✓いつでもやめられるように自動→手動に切り替えられるようにする
  - ✓プランより先にプロトタイプを作ってみる
  - ✓プレビューリリースで早めに使い始めて成果を出すようにする
- 以下の方向性も重要なので追加することに
  - ✓だれでも使い始めやすいようにする（導入コスト）
  - ✓だれでも使いやすいようにする（UI/UXの向上）
  - ✓壊して作ることに抵抗が生まれないような何かがほしい（変更のフットワークを軽く）
- 新たな要望や課題
  - ✓プロトタイプ作成とプレビューリリース、CIを迅速かつ楽にやりたい
  - ✓リリースの配布を簡単かつ迅速にできる仕組みがほしい
  - ✓使い方とかの口頭説明をなるべく減らしたい
  - ✓自動でオペレーション遂行不可な場合に、誰でも手動でオペ可能な状態にしたい

さらに見えてきたビジョン（再掲）

- とりあえず決めた方向性は間違っていないと感じた。
  - ✓全自動化する体力がそもそもないので、業務整理しつつ一部だけ自動化。
  - ✓いつでもやめられるように自動
  - ✓プランより
  - ✓プ
  - ✓リリース
  - ✓使い方とかの口頭説明
  - ✓自動でオペレーション遂行不可な場合に、誰でも手動でオペ可能な状態にしたい

こんな感じで進めていけば何とかなりそう！！

ちなみに…

ちなみに…

これってアジャイル…
これってスクラム…

…

ちなみに…

これってアジャイル

DevOps!!

（ひとり）

行き詰まらないための運用自動化

PHASE.5 DevOps文化をつくる