TIS
TIS INTEC Group
Go Beyond

オープンソースカンファレンス2021 Online/Spring

Part1. 社外秘の情報のドキュメント管理どうしてますか？
Part2. 運用レコメンドPFについての紹介

TIS株式会社
IT基盤技術事業部
IT基盤エンジニアリング企画室　小針・小川

全体アジェンダ

1. 社外秘の情報のドキュメント管理どうしてますか？
   - 30分程度

2. 運用レコメンドプラットフォームについての紹介
   - 10分程度

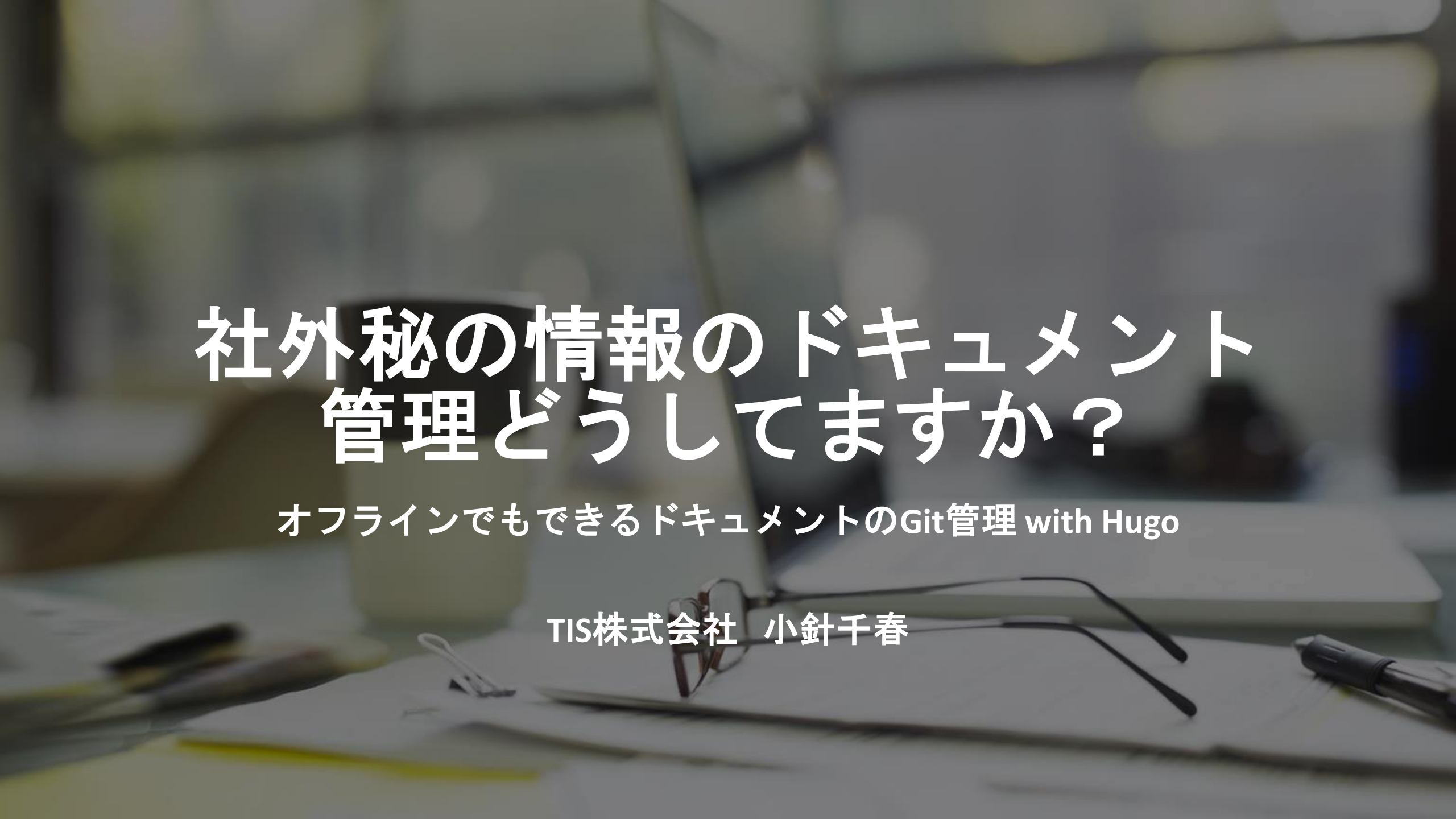
3. 質疑応答

発表者の紹介

- TIS株式会社
  - IT基盤技術事業部
  - IT基盤エンジニアリング企画室

- 主な業務内容
  - 「運用レコメンドプラットフォーム」というサービスの開発・運用

- 今日話す人（自己紹介は後ほど個別に）
  - 小針 千春（今話している人）
  - 小川 真人（あとで話す人）



社外秘の情報のドキュメント管理どうしてますか？

オフラインでもできるドキュメントのGit管理 with Hugo

TIS株式会社　小針千春

自己紹介　小針千春

- 社会人歴＝インフラエンジニア歴＝今月で満6年
  - ITは好きだけど、とくにインフラ好きではない
- ちょっとわかるかもしれない最近よく聞くOSS
  - Ansible, Terraform, Docker, Kubernetes
- 好きなキーワード
  - 自動化、効率化、一般化、可視化
- 得意な仕事、苦手な仕事
  - 1を10にするのが得意（改善）
  - 0を1にするのが苦手（新規創造）
  - 99を100にするのも苦手（仕上げ）

アジェンダ

1. なぜドキュメントを作るのか？
2. 自分たちの実施例の紹介
3. 実施して思ったこと

今日話さない内容

割愛しますが、必要なノウハウやスキルです。

- **Git**の使い方
- **Markdown**の書き方
- **Hugo**の使い方
- **Draw.io**の使い方
- Excelドキュメントを無くす方法
  - 個人レベルでは難しい
- 文書の書き方や構成の考え方
  - 本質的にはもっとも大事なスキル

最初に、Excelドキュメントへのスタンス

ドキュメントの話というと、Excelドキュメントの話が付き物
よく悪く言われるが、Excelドキュメントでも大抵のことはできる。

Excelの問題点？

- 運用・管理の難易度が高い
  - ⇒ ルールをしっかり作り込めば軽減できる
- 自動化ができない
  - ⇒ マクロでできるし、PowerShellなどでもできる
- バージョン管理できない
  - ⇒ SharePointなどでできるし、Gitでも一応バイナリごとできる

意外と反論できる。
しかし、なんとなく感覚的にExcelは悪いイメージがある。なぜ？

Excelの本当の問題点（だと思ってること）

- 編集が簡単で便利すぎる
  - とりあえずExcelで作った落書きがそのまま残り続けてしまう
  - 気軽にサイレント修正できてしまう
  - 直感的に『映え』を意識した見た目にできてしまう

- 多くの情報を包含できる
  - 「シート」という概念で、1ファイルの情報量が多くなる
  - 図や関数をいろいろな場所に自由に埋め込める
  - マクロまで作って埋め込める

自由度が高すぎて誘惑が多く、
網羅的なルール作りとルールの徹底が難しい。

結局、無くせるなら無くした方が楽。

今日の内容で伝えたいこと

- ドキュメントを作る目的を考えよう
- 「管理の難易度」を意識しよう
- 図を作ろう

今日話す内容はあくまで「例」

自分たちの場合に置き換えて考えることが大事

1. なぜドキュメントを作るのか？

Why make documents?

ドキュメント管理してますか？

いくつ当てはまりますか？

- ☑ 文書化されていない情報がたくさんある
- ☑ 図が全然無い（または一度作ったら全然更新していない）
- ☑ 関連情報がいろいろな場所に分散している
- ☑ Excel, Wordなどでしか管理されていない
- ☑ 更新する際のルールがやたら多い
- ☑ 更新するタスクだけ積まれて実際に更新されるのが滞りがち
- ☑ レビューが形骸化している（またはそもそも誰もレビューしていない）

なんのためにドキュメントが必要？

【備忘録】
情報を
忘れないようにする

【効率化】
スムーズに
情報共有や計画を行う

【属人化排除】
特定の人しか
知らない情報を減らす

【自己学習】
無駄な
会話やコストを減らす

他にもいろいろあると思う

こんなドキュメントはダメだ

【ごちゃごちゃ】

スムーズに
アクセスできない

【レビュー不足】

間違いや記載漏れが
多い

【更新されない】

誰も見なくなる
⇒誰も更新しなくなる
⇒属人化の原因

【差分がわからない】

レビューや更新の
ハードルがあがる

他にもいろいろあると思う

どうしたら改善できる？

- 【Automation】本質的でない部分は自動化しよう
  - 文字の装飾や配置を試行錯誤して頑張っていませんか？
  - スペルミスや誤変換などのしょうもないミスが多くないですか？
- 【Usability】見栄えや使いやすさにはこだわろう
  - 欲しい情報へアクセスするのに苦労していませんか？
  - 一度作成した画像の更新や修正で苦労していませんか？
- 【Acceleration】レビューや更新にスピード感を持たせよう
  - 細かい口頭説明が必要になっていませんか？
  - マスターファイルへの反映を直接手動で書き換えていませんか？

具体的な改善例

- 【Automation】自動化
  - Linterや校正ツールを使う
  - CIでチェックやツール実行を自動化する
- 【Usability】使いやすさ
  - 静的サイトジェネレーターを使う（後述）
  - わかりにくい文章は図で表現する（後述）
- 【Acceleration】スピード感
  - 差分管理する（レビュー難易度を減少）
  - 細かくチケット管理する

Excelドキュメントでも、
静的サイト以外は実施できること
（差分管理は微妙だが）

オススメしない改善例

管理や運用が難しくなると予想できることは危ない。破綻する。
ただし、目的や条件にもよる。

- Excelや独自ツールをめちゃくちゃ作り込む
  - 誰が保守するの？バグったらどうするの？
  - 簡単な作り込み程度なら「あり」
- ルールをめちゃくちゃ厳密にする
  - 更新のハードルあがって反映が遅くならない？
  - 間違ったら社会問題になるとかの重大なものであれば仕方ない
- WIKIで管理する
  - レビューが形骸化しない？
  - 簡易的な管理で十分なら「あり」

『適度な』手間が大事

2. 自分たちの実施例の紹介

Our Example

どんなものができるかの簡単なデモ動画

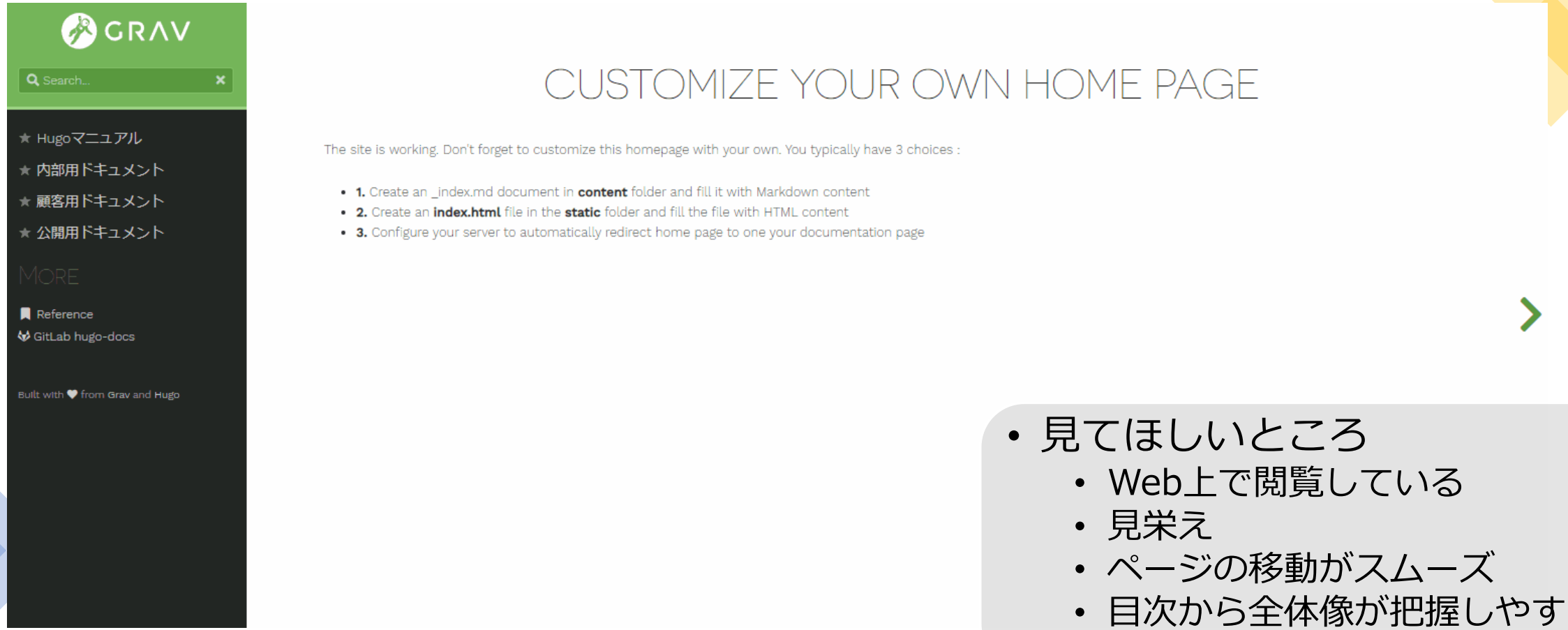
- 口頭では説明しづらいので、見て察してください

- 見てほしいところ
  - Web上で閲覧している
  - 見栄え
  - ページの移動がスムーズ
  - 目次から全体像が把握しやすい

どんなものができるかの簡単なデモ動画

実際に使用しているドキュメントと同じ構成のデモ動画

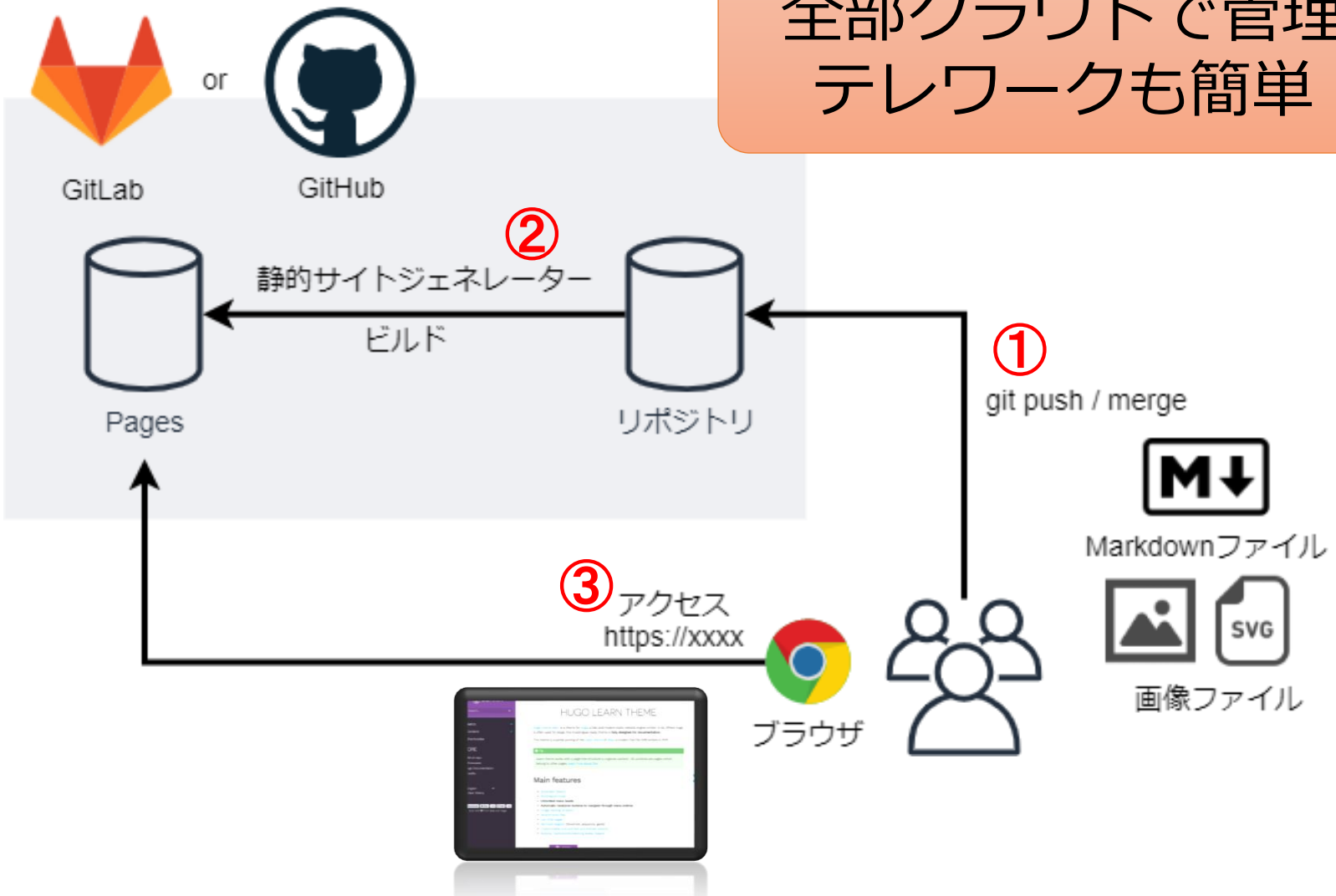
再生開始 再生終了



- 見てほしいところ
  - Web上で閲覧している
  - 見栄え
  - ページの移動がスムーズ
  - 目次から全体像が把握しやすい

クラウドが使える場合の理想構成

全部クラウドで管理！
テレワークも簡単！

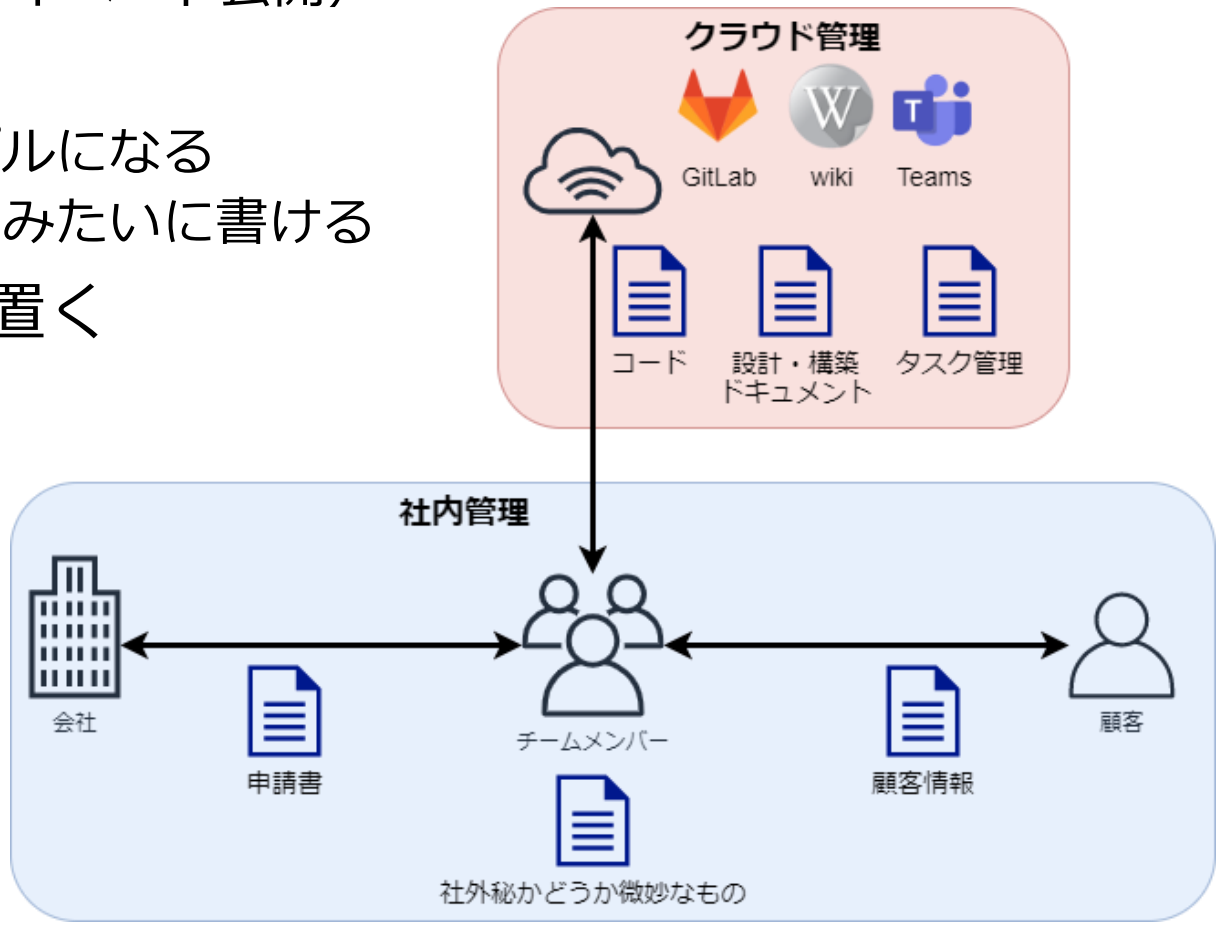


問題になったこと

- 会社のルールや報告書類が多い問題
  - わかりやすく整理しておきたい
  - 社外秘になるのでクラウドに置けない
- 顧客情報が膨らんでいく問題
  - わかりやすく整理しておきたい
  - 社外秘になるのでクラウドに置けない
- ローカルでドキュメント管理するの難しい問題
  - ファイルサーバ管理だと、どこに何があるのかわからなくなる
  - 画像や表などを埋め込みたいとExcelで作ることになる
  - レビューの難易度が高い

大まかな方針

- クラウド上に置いても問題ないものはクラウドに置く
  - 基本的にGitLab（現状はプライベート公開）
- なるべくコード化する
  - ドキュメントが格段にシンプルになる
  - 「このコードのココを参照」みたいに書ける
- 社外秘を含むものは社内に置く
  - ここをどうにかしたい

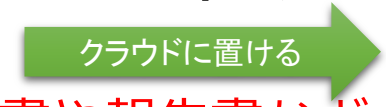


とりあえず社外秘の情報を明確化しておく

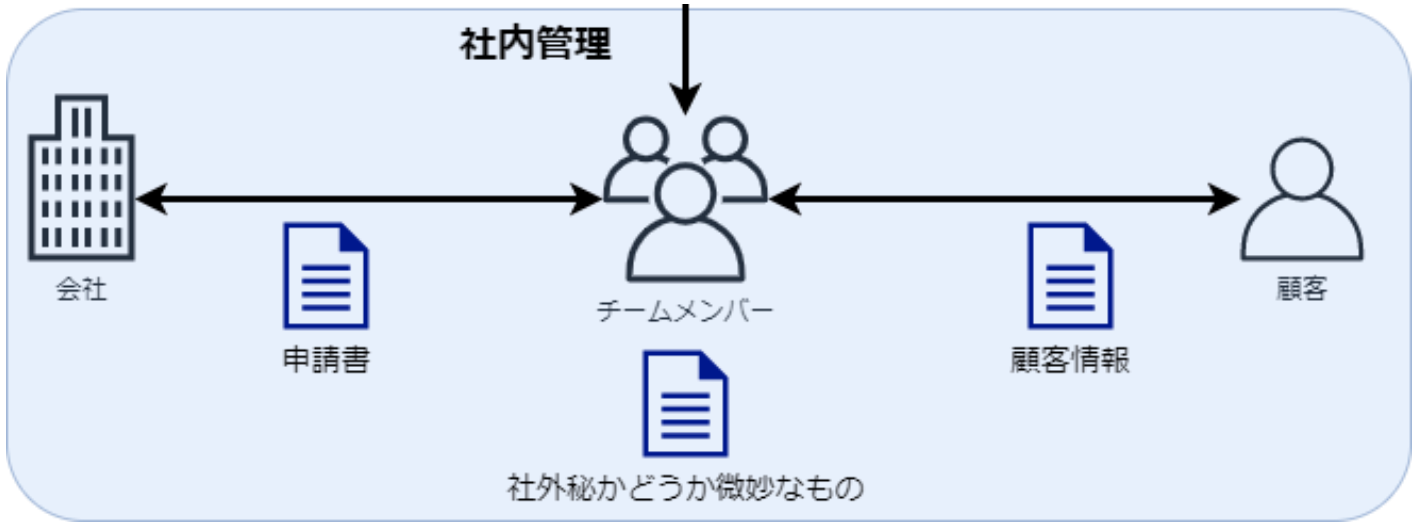
- コード（OSS化予定のため社外秘ではない）
  - アプリコード
  - IaCコード（Terraform, Kubernetes）

- ドキュメント（部分的に社外秘）
  - 開発用のメモ的なドキュメント
  - 設計や構築に関するドキュメント
  - 顧客情報などの社外秘内容を含むドキュメント

- プロジェクト管理関係のもの（部分的に社外秘）
  - タスク管理（Issue管理）
  - 会社から求められる申請書や報告書など



社内でどうやって管理する？



とりあえずGit管理したいな・・・

主なメンバー紹介

説明不要の
バージョン管理ソフト

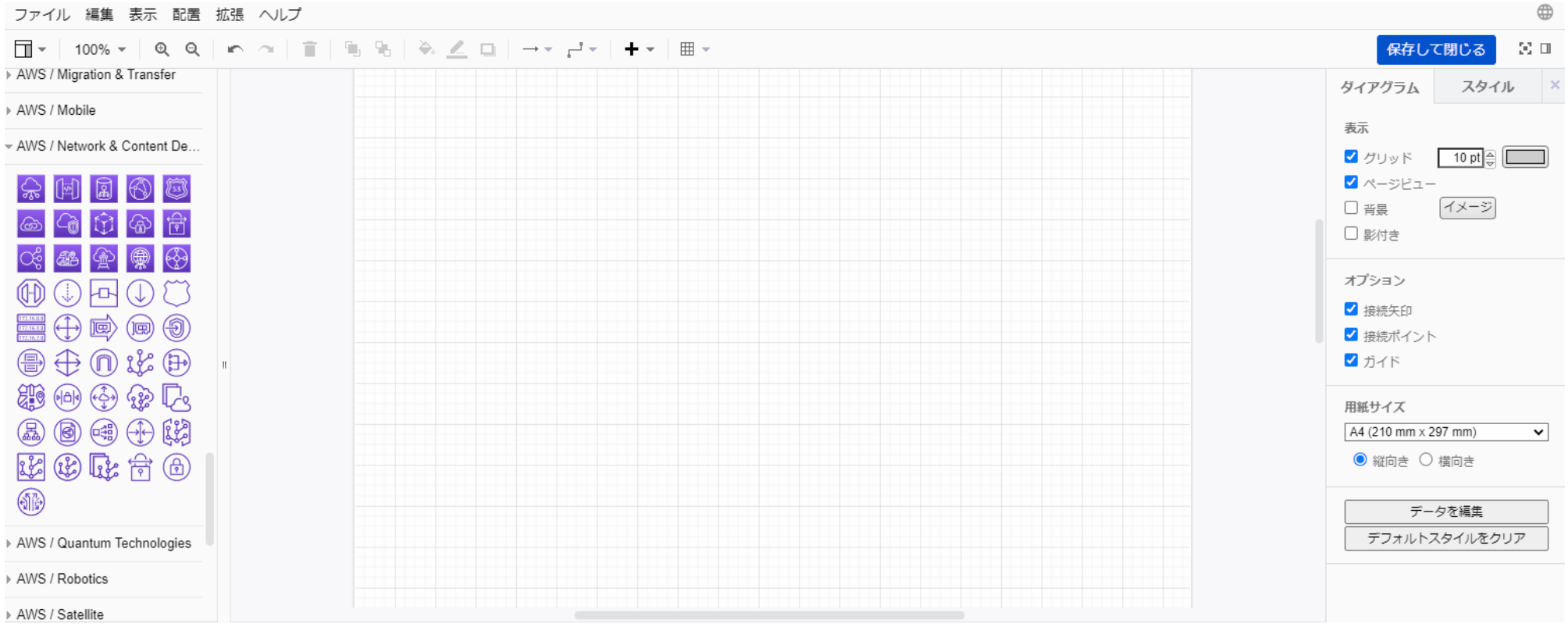
diagrams.net（draw.io）
オンライン作図ツール
オフライン版もある（未使用）

VSCode
説明不要のエディタ

Hugo
静的サイトジェネレーター

diagrams.net

- 作図ツール
  - 操作が快適（↓の動画の通り）
  - データはローカルに置かれる（クラウド上に置くことも可能）
  - SVG形式でレイヤー情報なども含めて保存できる ⇒ 後から編集しやすい



diagrams.net on

- VSCodeの「Draw.io Integration」アドオンを入れると、
  なんと**VSCode内で図の編集が可能になる！**
  - 図の編集・管理がとても快適になる
  - 実際の画面を見てみないと、おそらく意味がわからない
  - ⇒ デモ動画見せます

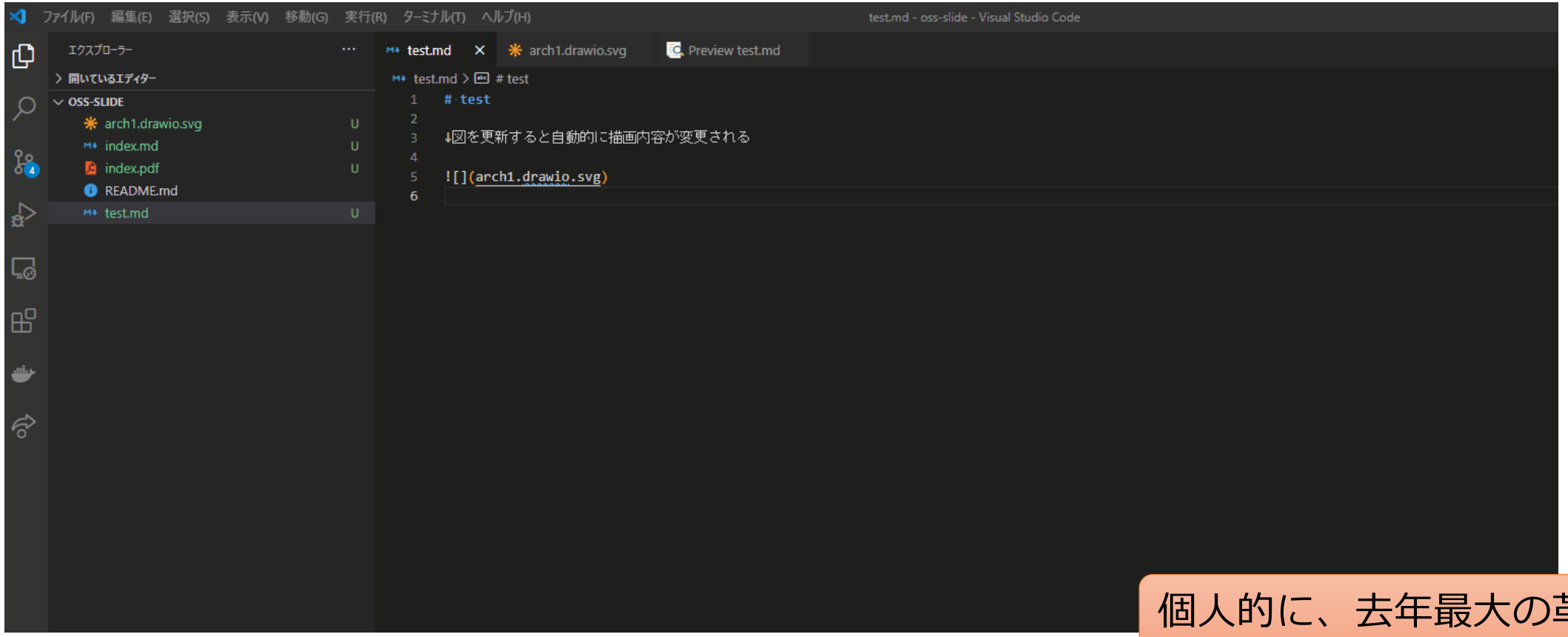
- 見てほしいところ
  - VSCodeの中に図の編集画面が開かれている
  - ファイルがローカルに置かれている
  - 画像を変更すると、即座にプレビューの描画内容が変更される

diagrams.net on

- 見てほしいところ
  - VSCodeの中に編集画面
  - ローカルでファイル管理
  - 画像変更が即座に反映

- デモ動画

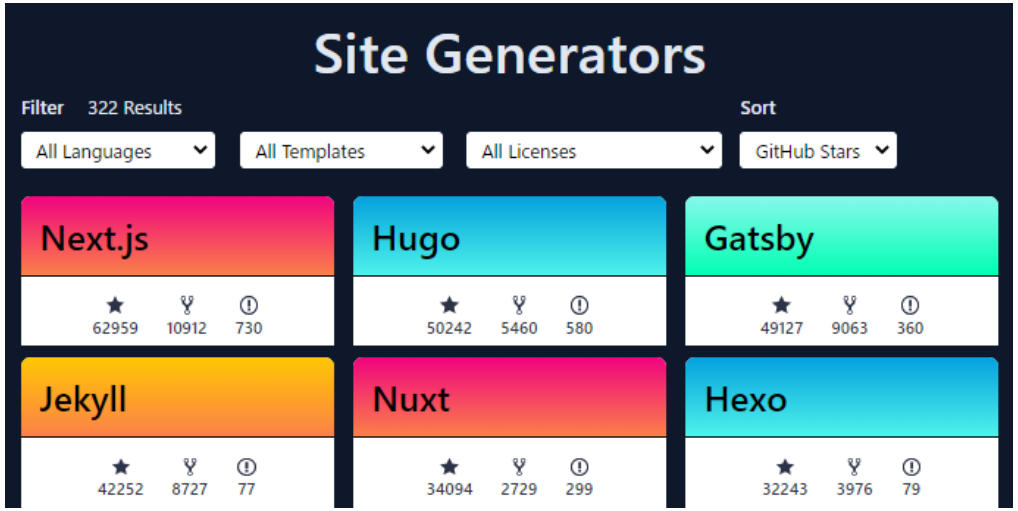
再生開始 ——— 再生終了



個人的に、去年最大の革命

- いわゆる、静的サイトジェネレーター
  - MarkdownからHTMLを生成する
  - デザインはカスタマイズ可能
  - 人気高めで、ノウハウがそこそこ充実している利点もある
- **バイナリファイルを1個ローカルに置くだけでビルドできる**
  - **サーバ不要！助かる！**
  - Webサーバある方がよいが、無くてもよい
- OSに依存しない
  - Windows, Linux, MacどれでもOK
- 最初の学習コストは高め

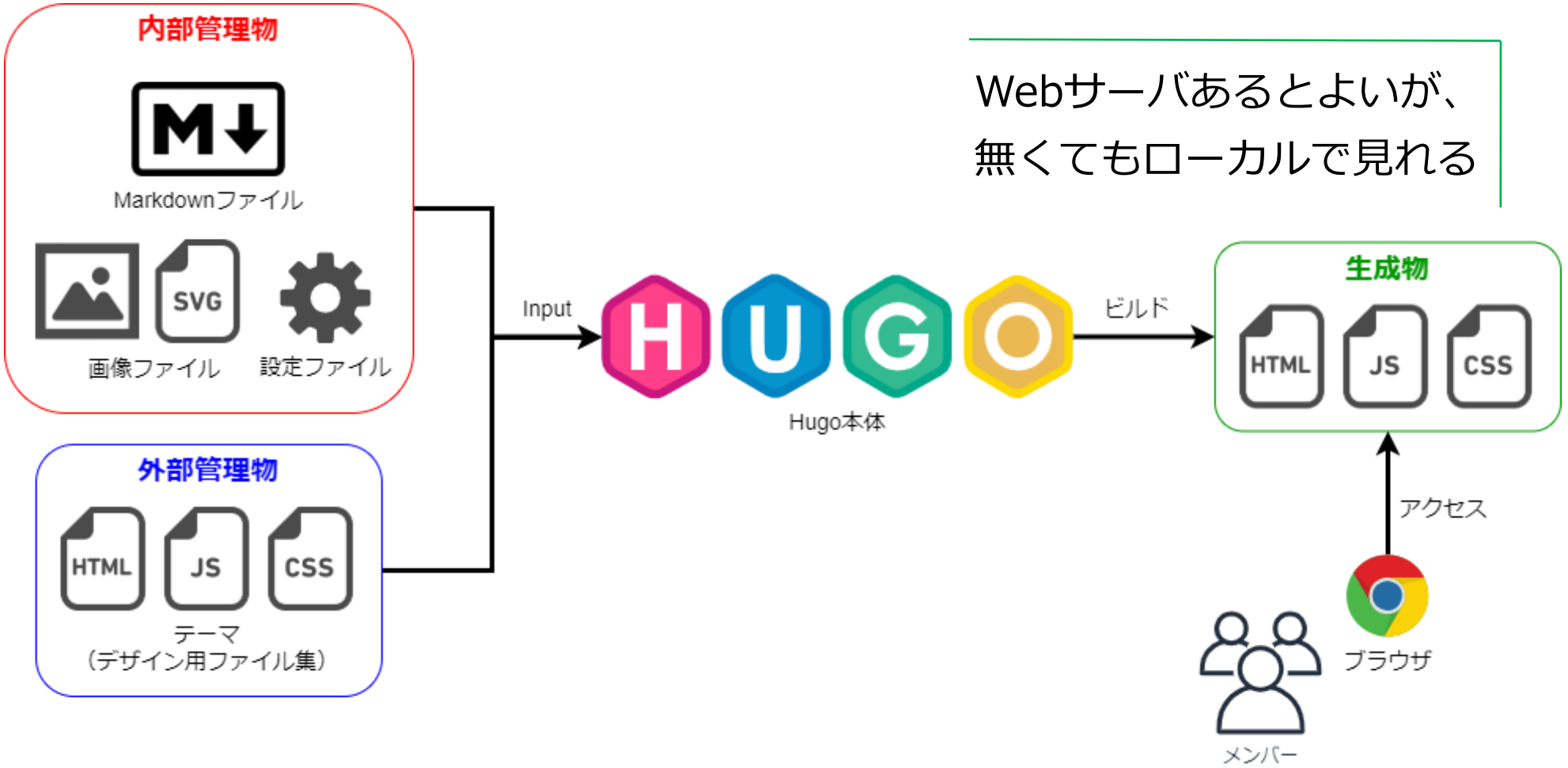
今回の内容では最重要



参考：Static Site Generators - Top Open Source SSGs | Jamstack

https://jamstack.org/generators/

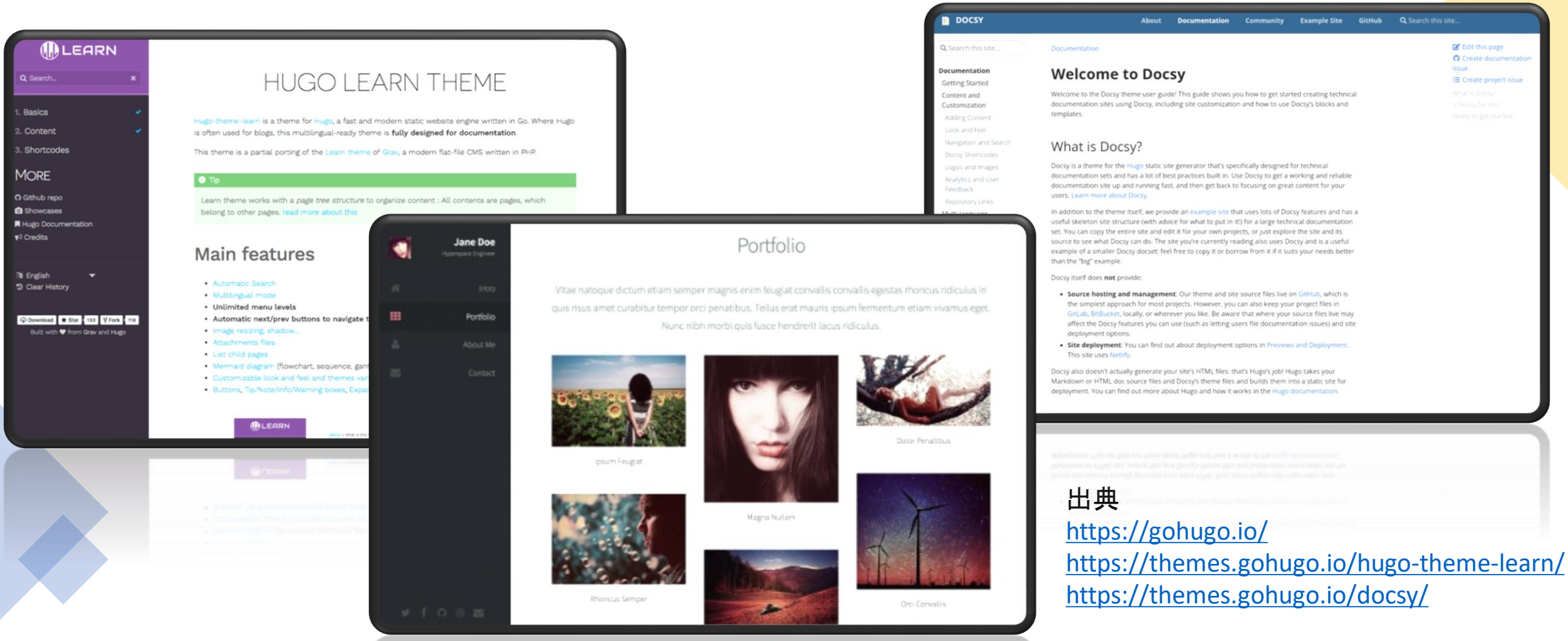
の概念図



Webサーバあるとよいが、
無くてもローカルで見れる

HUGO の生成物（見た目）の例

テーマを変えるとガラッと変わる



出典

https://gohugo.io/

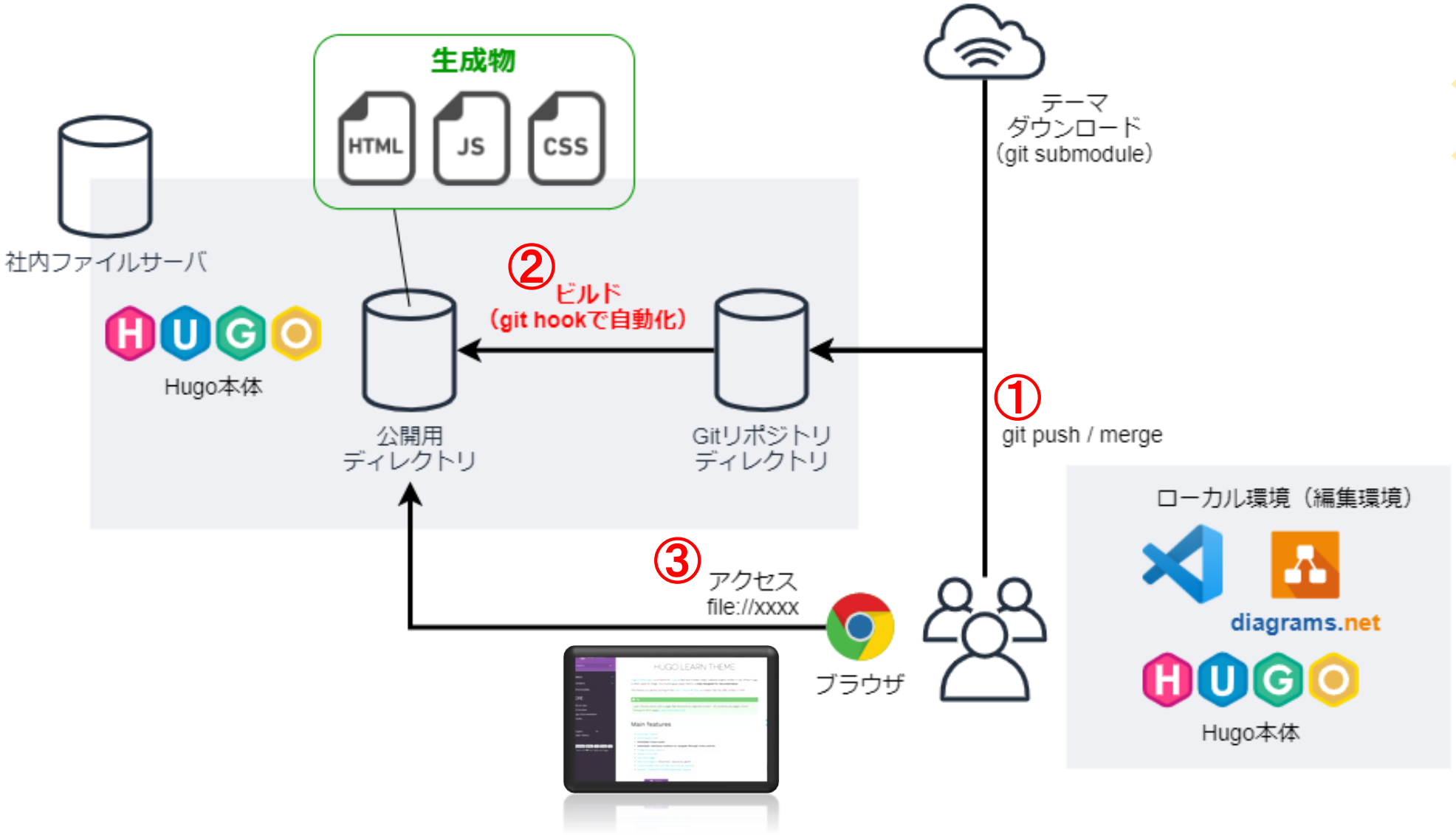
https://themes.gohugo.io/hugo-theme-learn/

https://themes.gohugo.io/docsy/

HUGO のメリット

- 低コストな管理で見栄えの良いドキュメント
  - テーマは気に入ったものをインターネット上からDLするだけ
  - オーバーライドする形式でテーマのカスタマイズ可能
- 編集難易度が低い
  - 中身のメインは一般的な書き方のMarkdownファイル
  - Markdownの校正ツールが使える
- 若干、動的な値を埋め込める
  - ファイル更新日、ページ内目次、文字数カウント、etc
  - 自作も可能（HUGO API, Javascript）
  - やりすぎは危険

社内で管理する構成


生成物
HTML
JS
CSS
社内ファイルサーバ
Hugo本体
②
ビルド
（git hookで自動化）
公開用
ディレクトリ
Gitリポジトリ
ディレクトリ
テーマ
ダウンロード
（git submodule）
①
git push / merge
③
アクセス
file://xxxx
ブラウザ
ローカル環境（編集環境）
diagrams.net
Hugo本体

その他：開発ノウハウはWikiで管理

- AWSにGrowiサーバ立てて運用

- 頻繁に変更入る
- レビューするほど重要でない

3. 実施して思ったこと

Impressions

実施して思ったこと

1. Hugoに詳しい人を1人作ると便利

2. GitHub Flowが管理しやすい
   - ※GitHub Flow: masterからissueブランチ切ってmasterブランチにプルリク

3. Excelドキュメントは例外的なルールを考える方が良い
   - WordやPDFなども同様

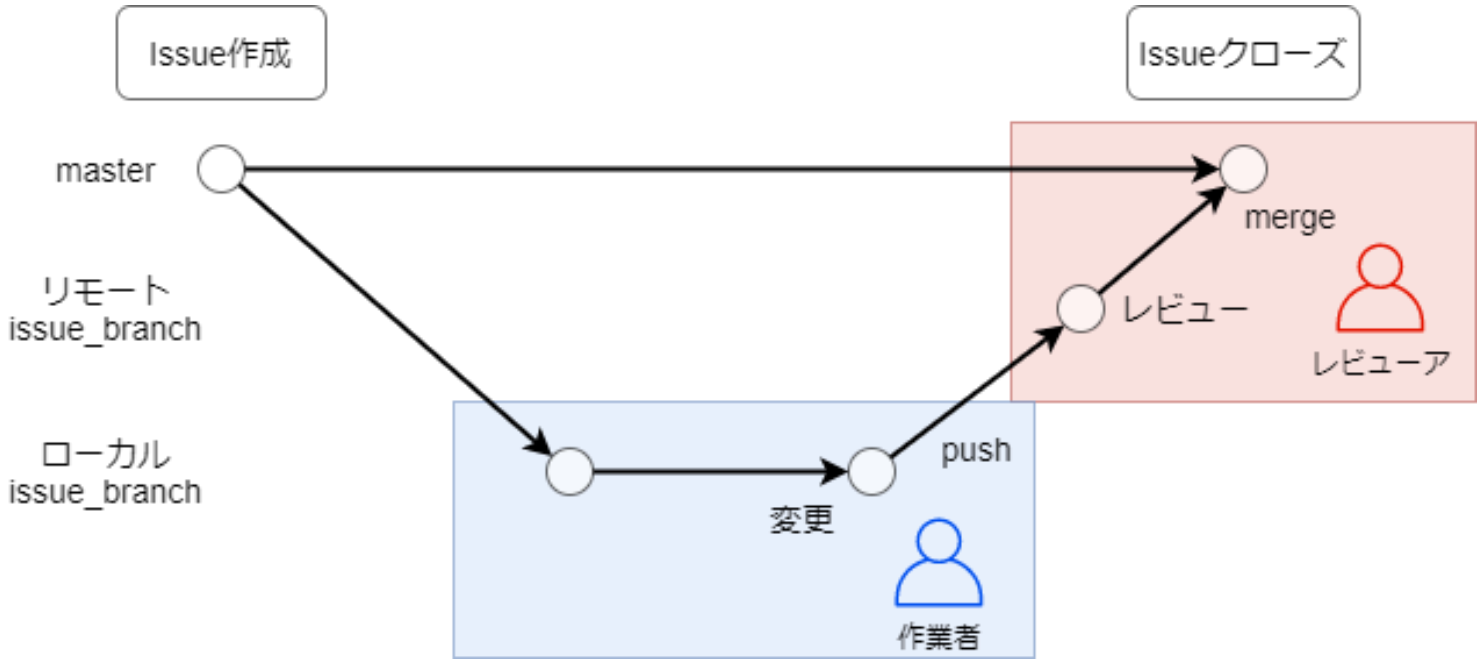
1. Hugoに詳しい人を1人作ると便利

Hugoを使っていると、必ずちょっとした不満点は出てくる。
Hugoに詳しい人がいると、ちょっとした手間で改善できたりする。
モチベーションの向上に繋がる。

自動でページ内の目次を
生成するようにした

2. GitHub Flowが管理しやすい

- 細かい単位でドキュメントを修正できる
- コンフリクト起こしにくい
- 更新が滞りにくい
- Speedy



3. Excelドキュメントは例外的なルールを考える

基本的に、ExcelやWordをGitで管理するのは難しい。
ファイルごとに目的と運用ルールを事前に決めておく方がよい。

- Markdownに移行する
  - 可能ならこれが最適（だいたい難しい）
- ファイルサーバで管理する
  - 個人的にはこれでよいかと思う
- 内部管理用のMarkdown、外部提出用のExcelで分けて管理する
  - ものによってはあり
  - ただし、必ずMarkdown更新→Excel更新の順番は守らないと破綻する
- バイナリのままGit管理する
  - あまりオススメしない
  - レビューがしづらく、変更に手間がかかるだけでメリットを感じられない

課題感

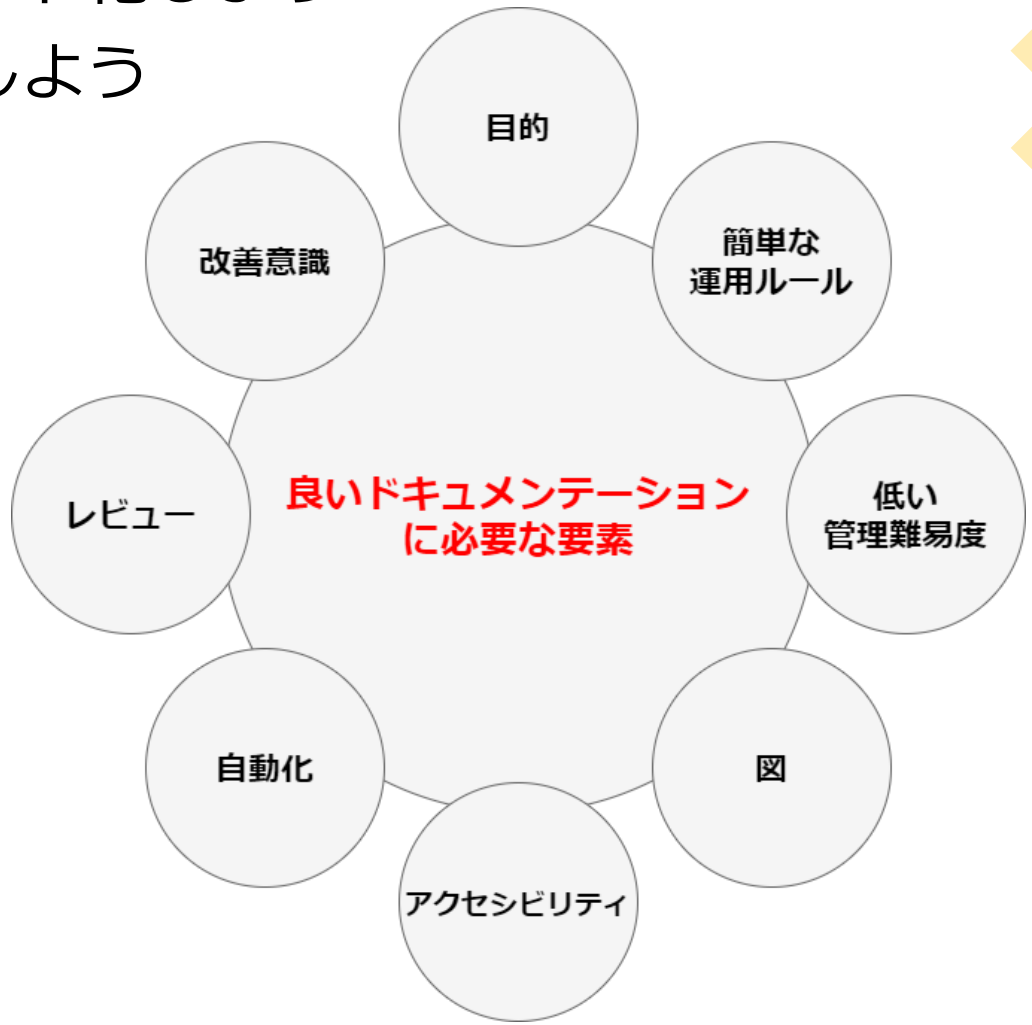
- GitLabやGitHubの差分確認が便利すぎた
  - ローカルだとそんなにキレイに差分を見れない
  - 現状は、VSCodeでGit系のアドオンを活用して実施している
  - そのうち良さ気な方法を検討したい

- テーマのカスタマイズは奥が深い
  - できたら嬉しいレベルなので、暇なときに試行錯誤している
  - 実際に行ったカスタマイズについての紹介を以下に書いてます
    - https://qiita.com/S_SenSq/items/39027868524e7970c12d

まとめ

- 社外秘の情報もしっかりドキュメント化しよう
- 良いドキュメンテーションを意識しよう
- 管理の難易度を意識しよう

- 当然、文章力や目次構成も大事
  - 今回は触れていない



運用レコメンドPFについての紹介

自己紹介　小川 真人

- 所属: TIS株式会社
  IT基盤エンジニアリング企画室
- 役割: 運用レコメンドPF サービス展開チーム
- 社会人歴8年目（TISは3社目で1年ちょっと）

- 興味のある分野・趣味
  - 物理層を含む低レイヤ(自宅にNW機器など買って遊んでるくらいには)
  - セキュリティ（自分では発想出来ない手法によってハックする手順を見るのが好き）
  - PC自作、写真（聖地巡礼含む）、オーディオ（特にポータブル）、アニメ

最終ゴールは、AIによる自律・自動型のIT運用

【保守・運用の効率化・品質向上の限界】

IT運用		効率化・品質向上への取組み
定常業務	定型業務	⇒旗印は「標準化」
非定常業務	非定型業務	⇒**標準化できない** **・・・> だからこそ、非定常・非定型**

【”運用レコメンドプラットフォーム”の目指す世界】

障害の発生予防（予兆検知）
障害の早期回復（ダウンタイムの短縮）

最終ゴールは、**「自律型自動運用」**

保守・運用作業の
Nextアクションをレコメンド

運用保守に関する
データの集約と共有

イベントデータ
(ログ、障害、構成変更、
システム操作、脆弱性発覚etc.)

構成情報データ
(サーバ、SW、
パラメータetc.)

集約データの**分析**

統計分析・機械学習 etc
(テキスト分類、類似度分析、共起分析 etc.)

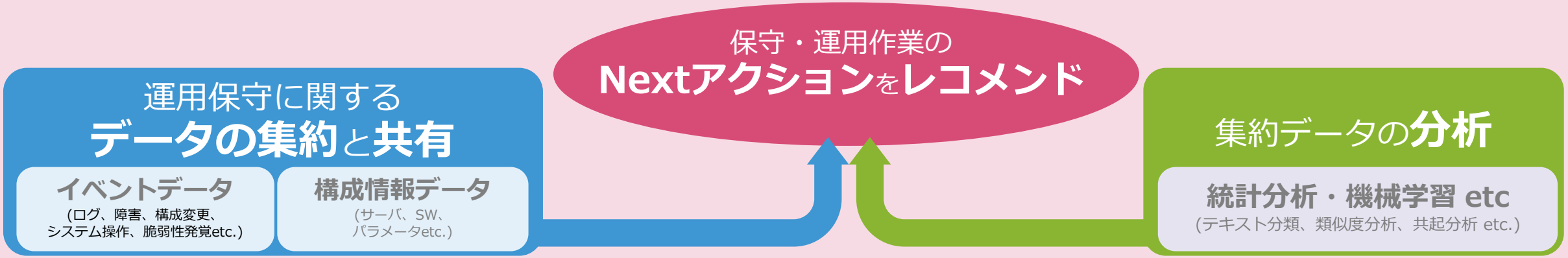
既存の分析ツール、可視化ツールとの違い

TIS
TIS INTEC Group

【運用レコメンドPFのコンセプト】

- 既存の監視系ツール、ログ管理系ツールの置き換えというわけではなく、
  各運用系ツールを束ねて、運用の効率化を促進するという位置づけの製品となります。
  - (メトリクス監視をしている既存の監視データを活用し、状態の異常を検知したり、
    構成情報やその他作業イベントの情報と関連付けて可視化したり)

障害の発生予防（予兆検知）と、障害の早期回復（ダウンタイムの短縮）



運用レコメンドPFの機能紹介

運用レコメンドプラットフォームサービス概要

2パターンの提供形態

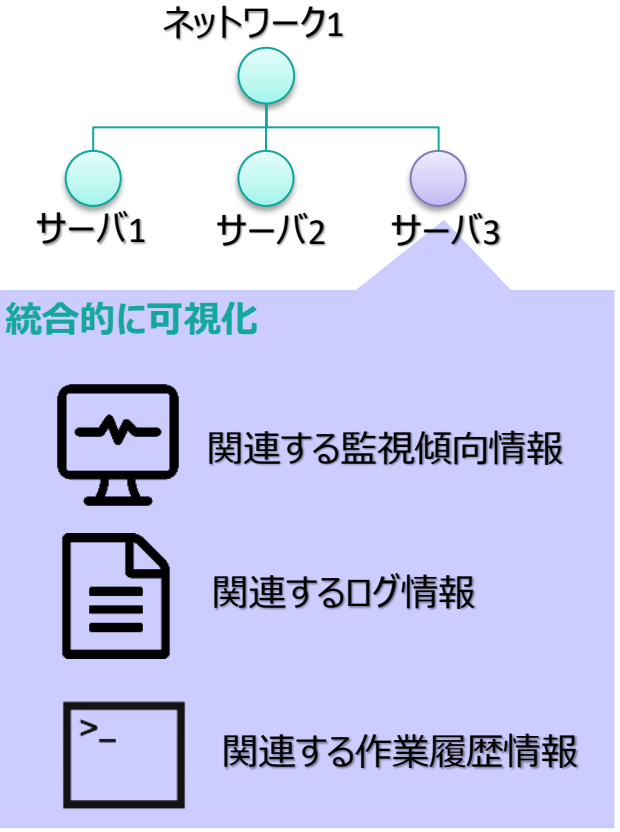
手軽に使える..

SaaS型

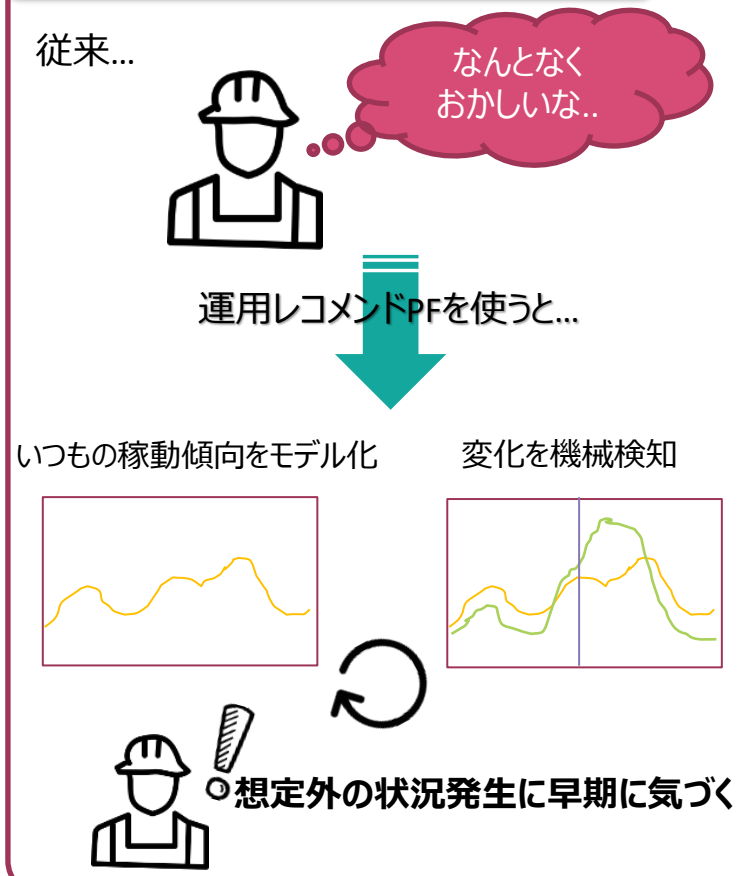
クローズド環境でも使える..

パッケージ型

運用情報の統合表示



いつもと違うを機械的に捉える



提供予定の機能概要

1 ダッシュボード Operation Dashboard

以下を確認可能に
・システムで異常が起こっていないか
・異常状態になっているのはどこか
・異常箇所に関連する状態がどうなっているか

- 監視データ、ログデータ、設定情報、操作履歴等統合的に確認可能

運用レコメンドプラットフォーム

2 データ収集基盤 Data Collector

- 根本的なデータ収集は既存監視ツール等と連携し、分析できる形に変換して保有することに注力

以下を実現
・保守運用に必要なデータを自動的に集約
・構成情報を軸にし、各データを関連付けて管理

3 データ分析基盤 Data Analyzer

- システムの状態変化の検出と検知結果と運用作業との関係性分析に注力

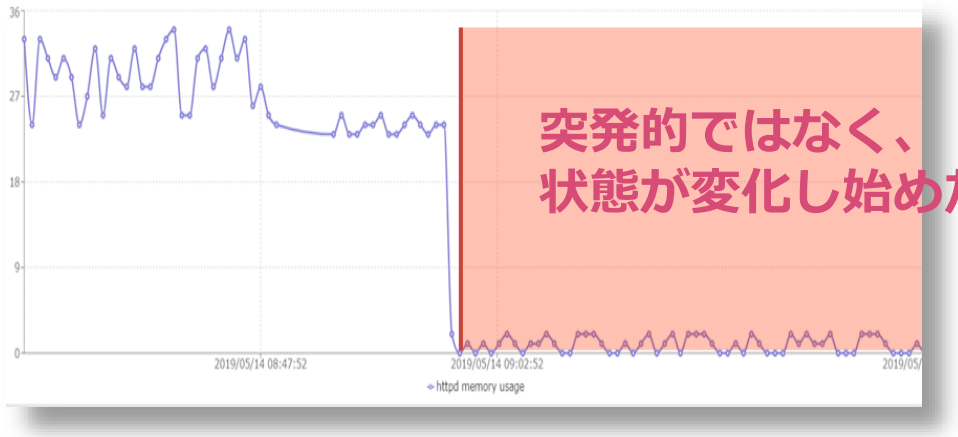
※データ分析機能詳細は参考情報ページを参照

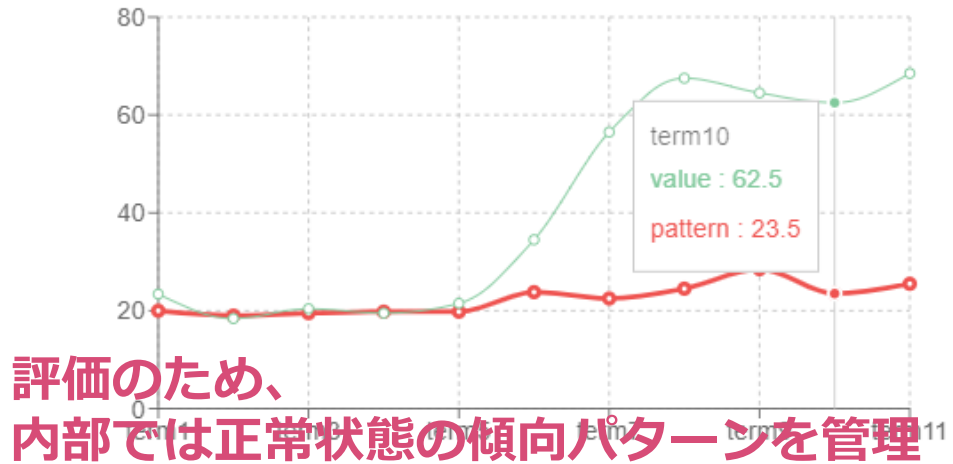
以下を実現
・いつもと状態が違うタイミングを早期に検知して伝える

分析機能仕組み １ -監視データ変化点検知分析-

- 数値系監視結果データに対して、変化点の検出処理を実施する機能
  - ① 指定した監視アイテムのデータに対し、**正常状態時の分析パターンを生成**
    (周期性も考慮した変動パターンの生成。デフォルトは**1週間の変動周期を考慮**)
  - ② 直近の監視データを用いて、①の正常パターンと比較して**変化点が発生していないかを定期分析**
    (実行間隔はデフォルト30分に1回)

分析評価イメージ



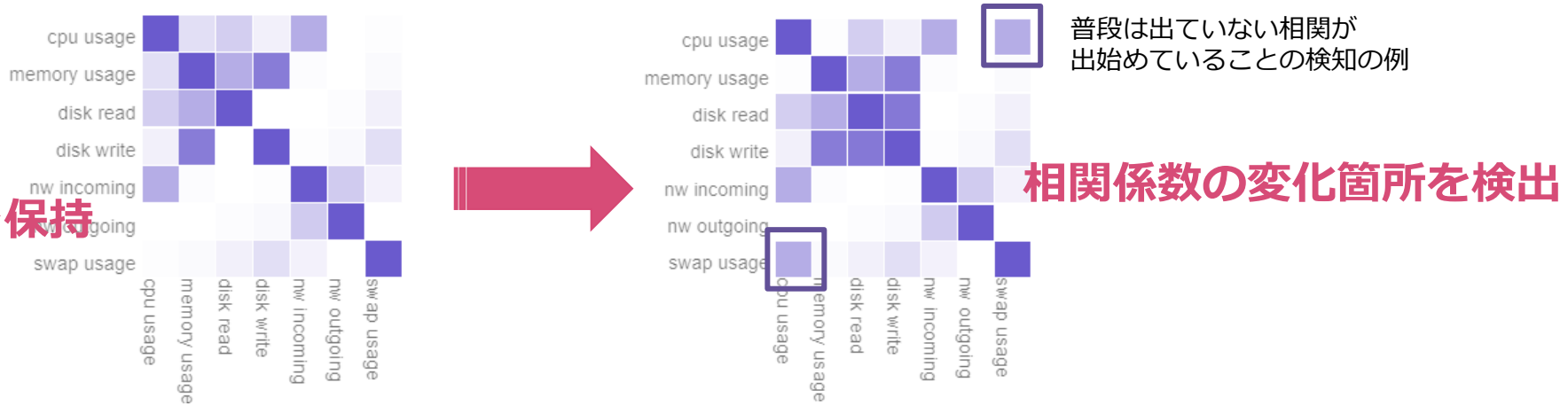


分析機能仕組み 2 -複数監視データ同士の相関変化分析-

- 複数の監視データ同士の変動の相関変化を分析する機能を提供
  - ① 任意の複数の監視項目のデータを用いてそれぞれの変動の相関関係の正常時パターンを算出
  - ② 定期的に、直近データを用いて相関関係を算出、①正常時パターンと比較し変動が大きい箇所を検知
    (実行間隔は調整可能。デフォルトは30分に1回)
  - 複数の監視項目の選択のパターン例
    - i. 同一のサーバの監視項目同士の相関を見るケース
    - ii. 同一の監視項目(CPU使用率)のサーバ間の相関を見るケース

分析評価イメージ

正常状態時の相関係数を内部で保持



相関係数の変化箇所を検出

分析機能仕組み 3 -ログデータ出力傾向変化分析-

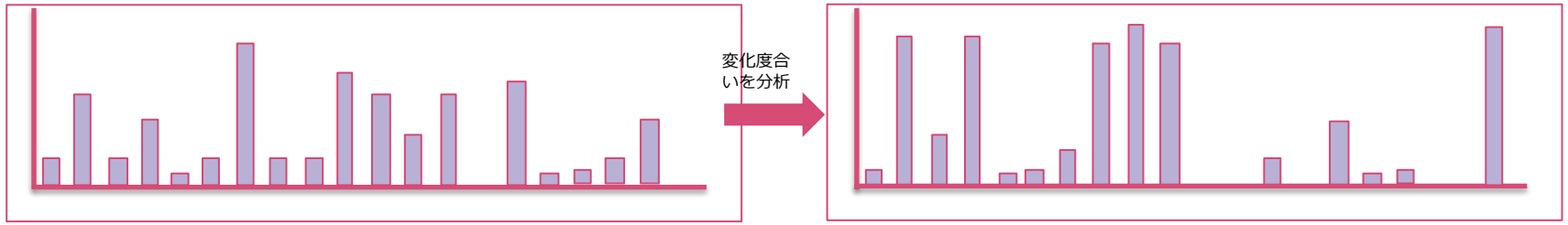
- ログに含まれるキーワードの出力傾向情報を分析し、前日の同時間帯と比較して傾向が変わってきたタイミングで検知

比較対象期間(前日の同時間帯データ)

2019/09/18 12:01:20 +09:00 warning Starting app server ...
2019/09/18 12:03:21 +09:00 info check test ok
2019/09/18 12:07:12 +09:00 info check test ok
2019/09/18 12:08:21 +09:00 info check test ok
2019/09/18 12:12:43 +09:00 warning Starting app server ...
2019/09/18 12:15:14 +09:00 warning lotate debug log
2019/09/18 12:16:45 +09:00 warning invalid type integer
2019/09/18 12:29:11 +09:00 info check test ok
2019/09/18 12:31:01 +09:00 warning Starting app server ...
2019/09/18 12:32:32 +09:00 info check test ok

評価対象期間(直近のデータ)

2019/09/19 12:05:23 +09:00 warning Starting app server ...
2019/09/19 12:06:22 +09:00 warning old test debug log
2019/09/19 12:07:21 +09:00 warning invalid type integer
2019/09/19 12:08:20 +09:00 warning invalid max connections parameters
2019/09/19 12:12:19 +09:00 warning Starting app server ...
2019/09/19 12:15:18 +09:00 warning lotate debug log
2019/09/19 12:16:17 +09:00 warning invalid type integer
2019/09/19 12:29:15 +09:00 warning less max connections parameters 200
2019/09/19 12:31:14 +09:00 warning Starting app server ...
2019/09/19 12:32:13 +09:00 info change debug level 3 to 4



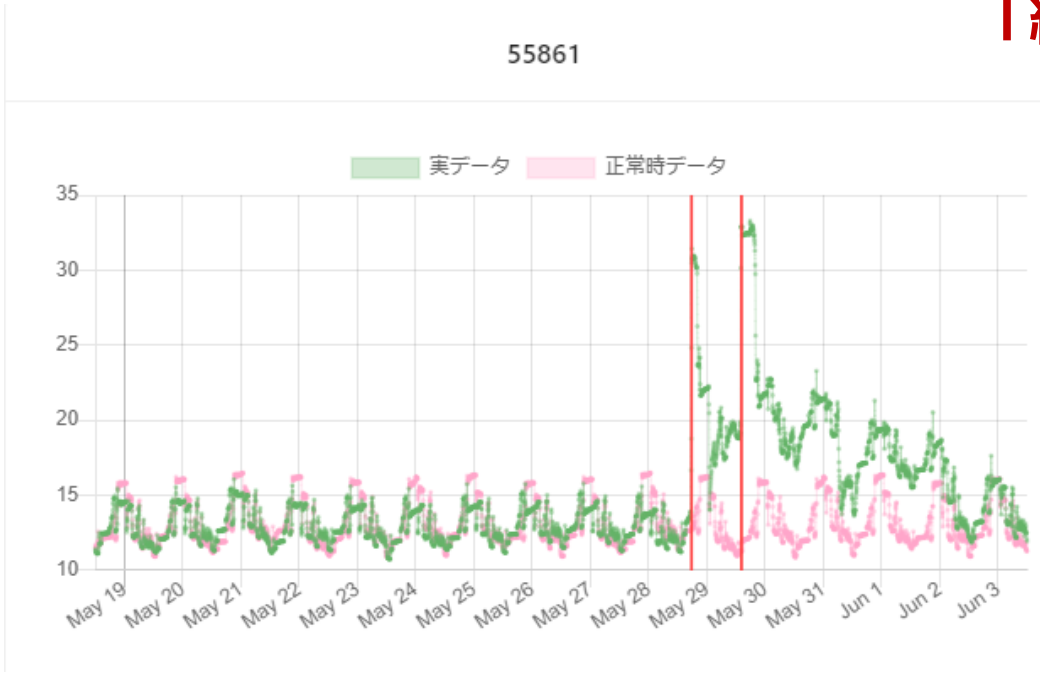
期待する効果

- キーワードマッチベースの検出では気づけない傾向変化を早期検知できるようになる

分析結果の確認イメージ

- 検知結果イメージ

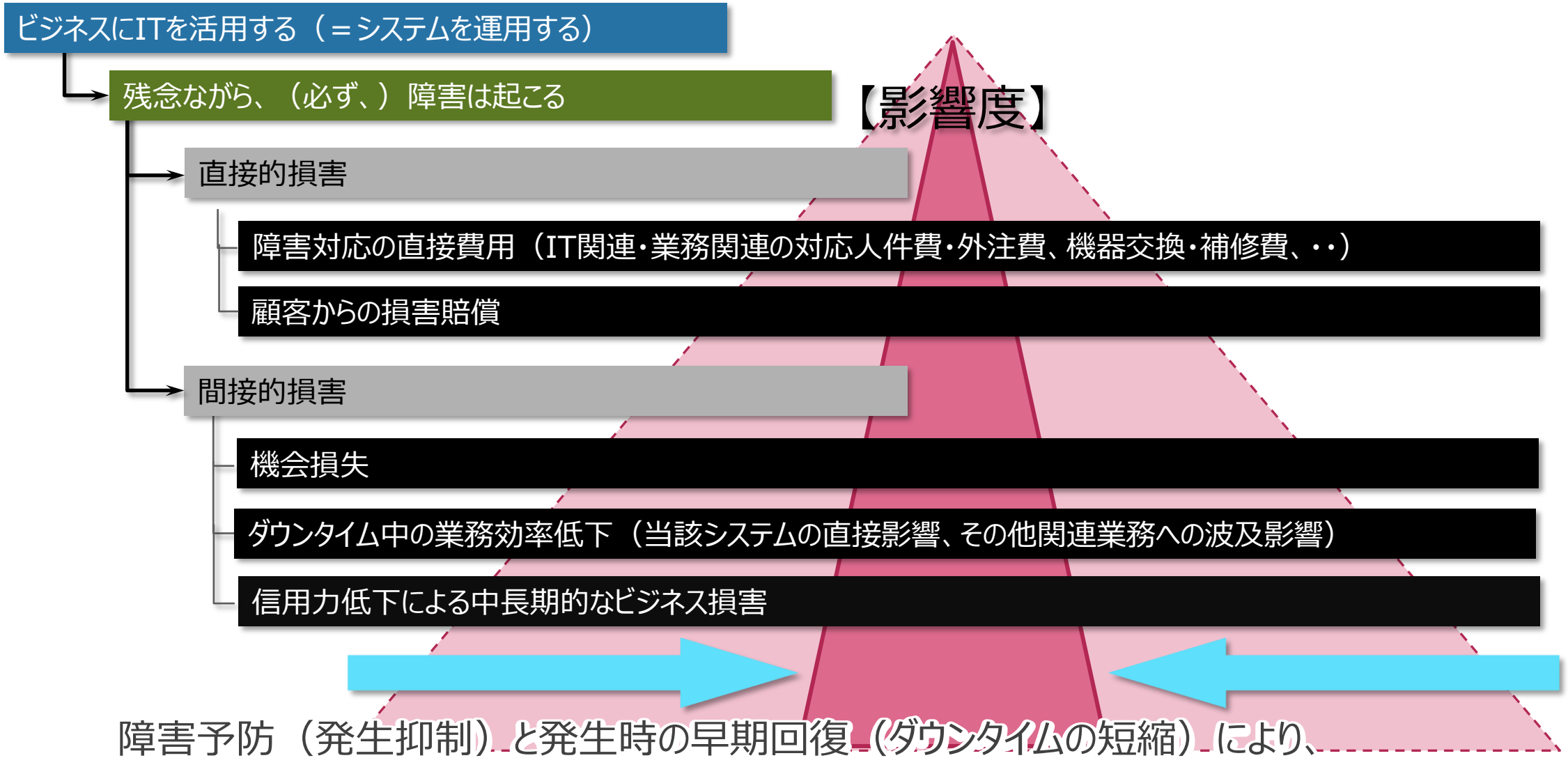
監視項目同士の相関度合いを算出し
「総合的にいつもとなんとなく異なるを早期検知」



正常時の状態と比較した変動箇所を検知
しきい値ベースの監視設定で気づきにくい
「いつものとなんとなく異なるを早期検知」

運用レコメンドPFに期待する効果

運用レコメンドPF活用によるビジネス損害の極小化



障害予防（発生抑制）と発生時の早期回復（ダウンタイムの短縮）により、
ビジネス損害を抑制

⇒見えない損害ほど、その影響度の抑制効果が大

システムの異常に”**いち早く**”・”**的確に**” 気付けるように

TIS
TIS INTEC Group

【現行のよくある運用シーン】

1 監視の仕組みは入っていて障害には気付けるようになっているが・・・

課題

- 閾値ベースの設定が中心で調整が複雑になりがち(**属人的**)
- 閾値を厳しくしすぎると大量のアラート発生
- 閾値をゆるくしすぎると障害発生の検知が遅れる

2 おかしくなってきていないかの確認を
週次や月次で目検チェックしているが・・・

課題

- チェックする対象が多く時間がかかる
- おかしそうかなという基準が見る人によってブレる(**属人的**)
- チェックとチェックの間隔が長くなりがちなので検知のタイミングは遅れる

システムの異常に”**いち早く**”・”**的確に**” 気付けるように

【現行のよくある運用シーン】

1 運用レコメンドPF

いつもの稼働状態を
データとして記録

正常時
パターン

直近稼働
パターン

機械的に違いをチェックし続け
変化の発生を検知

データ収集

運用システム

いつもと違うおかしくなってきたタイミングを**機械的に判断**できる！
将来的には・・・
属人的になりがちな閾値ベースの監視設定を排除していける

”なんでだっけ？”をよりスピーディに解決できるように

TIS
TIS INTEC Group

【現行のよくある運用シーン】

1 監視の仕組み、作業記録管理、インシデント管理の仕組みは入っているが・・・

課題

- それぞれ管理しているツールが別々になっていて**確認に手間**がかかる
- **各情報の関連付け**もできておらず効果的に活動できる状態で管理できていない

2 監視の仕組みは調整されていてアラートは的確にあがってくるが・・・

課題

- 監視アラートが何をきっかけにその状態になったのかを**確認するのに時間がかかる**
- どこを確認すべきかについても**属人的**になりがち

”なんでだっけ？”をよりスピーディに解決できるように

【現行のよくある運用シーン】

1 運用レコメンドPF

運用情報を時間軸に沿って統合的に管理

時間

- サーバ内の設定パラメータ変更
- サーバの負荷上昇警告アラート
- サーバ内の設定パラメータ変更
- サーバの負荷復旧アラート

運用システム

監視データ
監視アラート
ログデータ
操作履歴
設定変更履歴

2

何か起こった時に時間に沿って実施された事象を**即座に確認**できる！
将来的には・・・
機械的に関係性を判断し作業実施前の未然予防や
アラート発生時のリカバリプランの判断を機械的にできようになる

運用レコメンドPFの提供状況と今後の方針

運用レコメンドPFの状況

TIS
TIS INTEC Group

提供状況

2020/10/30に**正式版の提供を開始**しました。

今後の方針

AI技術を益々活用した**保守運用者を補助**できる仕組みを開発中です。

～開発中や検討中の機能～

- 分析モデルの自動管理強化 (現状いつもの状態の期間指定が運用者による指示ベース)
- 構成情報の表現のバリエーション強化
- ゆくゆくはAI技術を駆使して、
  - 異常の原因となった行動の分析(原因分析)
  - 異常により影響を与える箇所の分析(影響分析)

おわりに

- 運用レコメンドPFに興味を持った方
  - 機能的に使ってみたい　→　トライアル導入のご相談ください
  - 中の仕組みを知りたい　→　気軽にお声がけください

- TIS(我々の部隊)では、
  - CloudNativeなシステムに取り組んでみたい
  - 運用保守改善に貢献できる仕組みづくりをしてみたい

  といった方と一緒にチャレンジしたいと思っています。

THANK YOU

TIS INTEC GROUP Go Beyond

本資料に関するお問い合わせ

TIS株式会社
IT基盤エンジニアリング企画室

運用レコメンドPFサービス運営窓口
opsbear.service@ml.tis.co.jp