Corda NetworkをKubernetesで運用する

商用環境へ向けたCorda Enterprise構築

2021/05/14　株式会社トリクトラック　澁谷　美緒

自己紹介

しぶや　みお

澁谷　美緒

株式会社トリクトラック代表取締役
コンサルタント/PM

2014年～　アジャイル開発
を実践しています。

2019年～　Corda開発

トリクトラック

株式会社トリクトラック

- 2014年創業、所在地は愛知県半田市

フルリモート・アジャイル開発

- コロナ以前から基本メンバーはフルリモートでアジャイル開発を実践
- 事務所は半田市にありますが、プロジェクトメンバーは全国にいます
- 会社の徒歩圏内に住んでいても出社は自由
- 採用もすべてリモートで完結
- Slackに出社する感覚／ビデオ会議で各種アジャイルミーティングを実施
- 事務所には社員猫が出社しています

本日の概要

弊社（トリクトラック）が Corda 商用案件開発で採用した
アーキテクチャと開発手法を紹介します。

Cordaを使った商用案件開発の難しい点

- 新規技術の**理解・習得のコスト**
- Corda本体の**進化に追随していく柔軟さ**が必要
- 分散台帳の**本質的な複雑さ**に起因する開発難度の高さ
- 新技術に対する**ビジネス側の理解度と期待度**のアンバランス

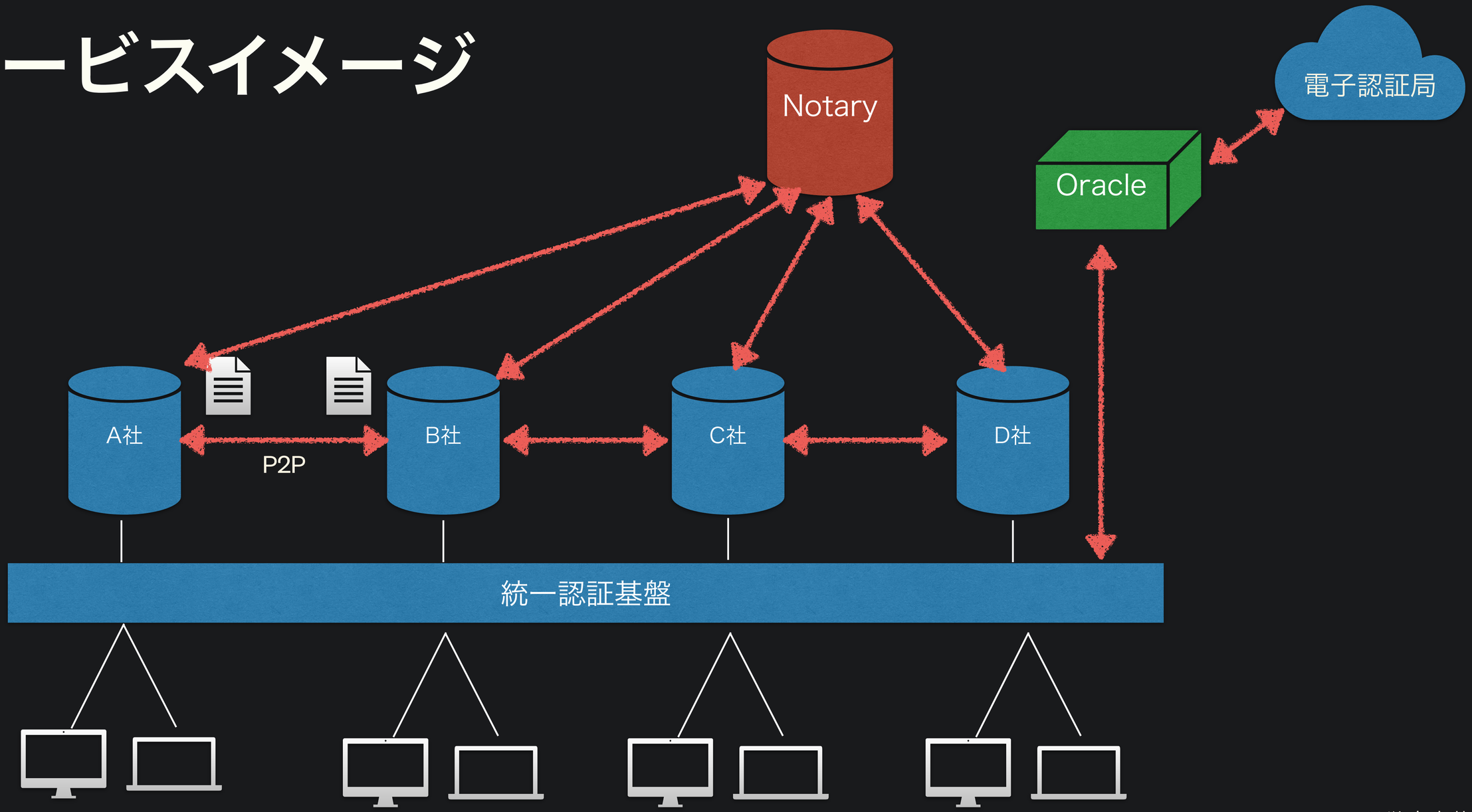
トリクトラックのアプローチ

Agile
devOps
Kubernetes

開発要件概要

- BtoBの取引基盤
- 取引参加企業　1社ごとに1ノード（サーバー）を立てる
- 取引の実在性・非改ざん性はサービス基盤の運用するNotaryが保証
- 取引ごとに認定認証局のタイムスタンプを押す
- （将来的には）海外に現地法規に基づいてノードクラスタを構築可能

サービスイメージ



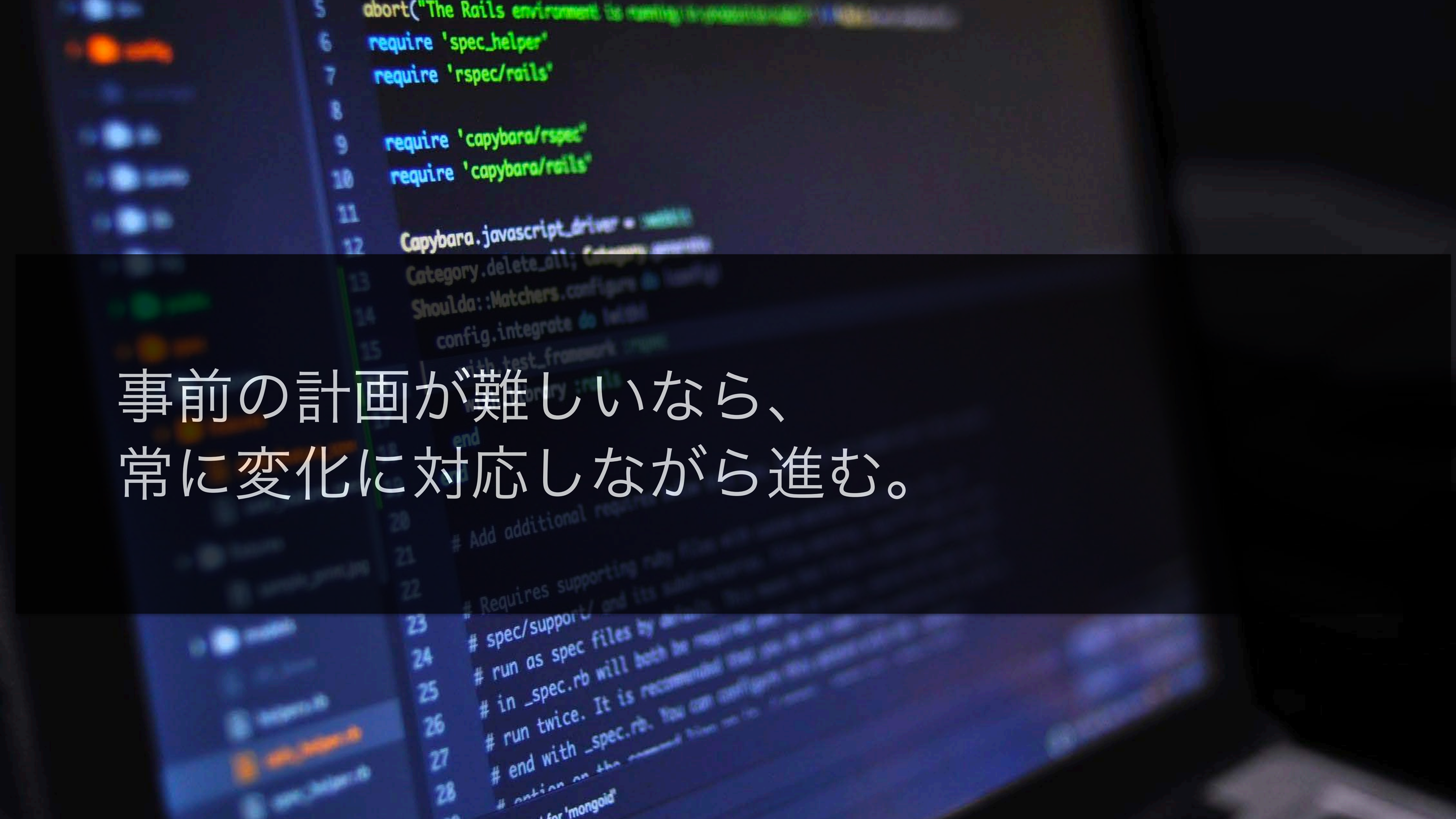
※Oracleは独自実装

ビジネスレベルのCordaサービスが解決すべき課題

- **要件とリスクのぶれ**をどうやって吸収し、開発を収束させるか？
- ユーザーからCordaアーキテクチャを**いかに自然に見せるか**？
- 参加企業ごとに存在するノードを**誰**が**どのように**管理するか？
- **複雑なアーキテクチャ**をどのように統制するか？

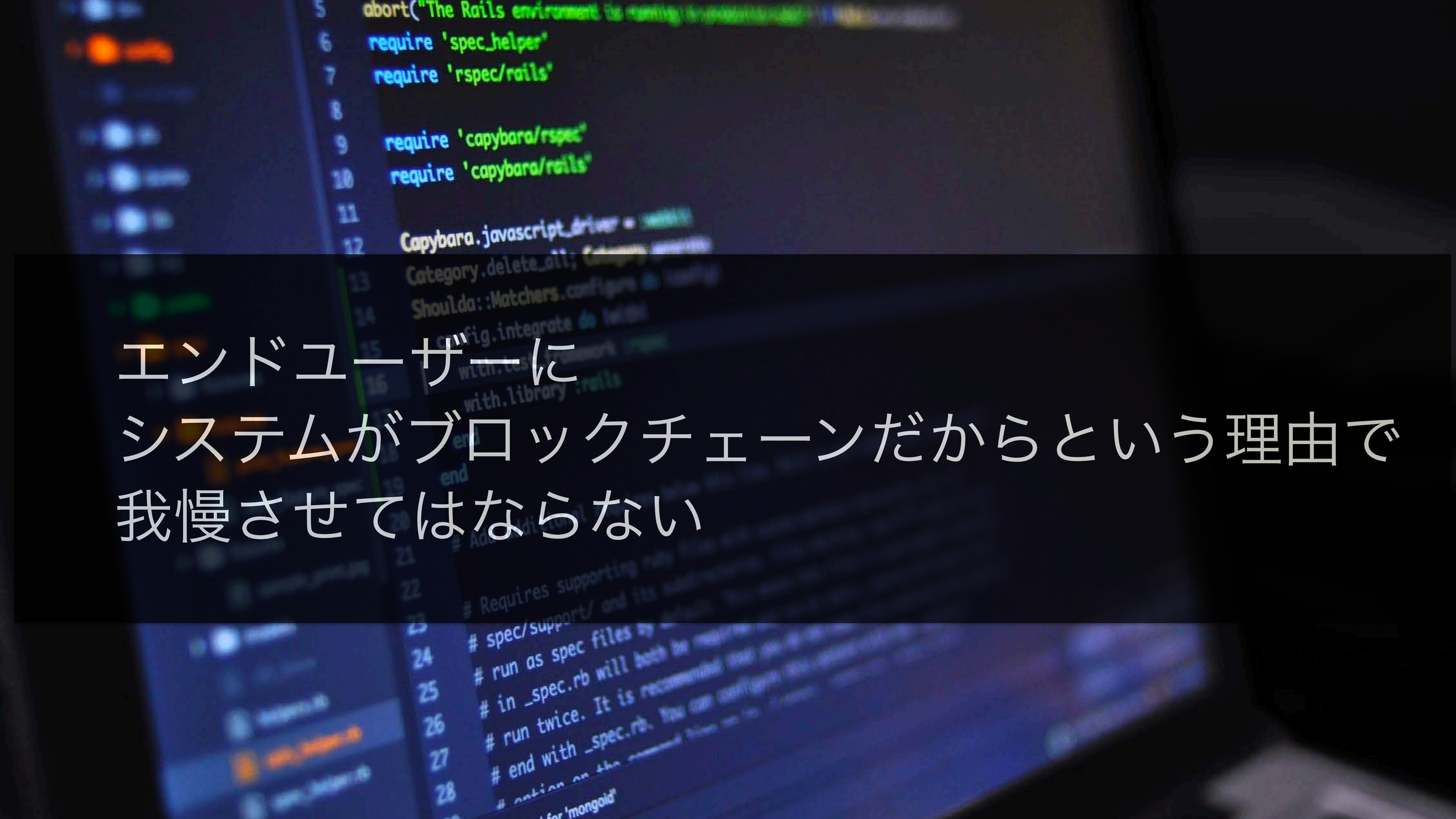
要求とリスクのぶれにどう対処するか？

- Cordaを使った商用案件は**未踏領域**
  - 未知のリスク、予想不可能なリスク
  - 未知の技術で工数予測が難しい
- ブロックチェーンの**ビジネス利用モデルは発展途上**
  - ユーザー要望の予測が難しい
  - 期待のコントロール


事前の計画が難しいなら、
常に変化に対応しながら進む。

ユーザーからCorda アーキテクチャを
いかに自然に見せるか？

- Blockchainの速度によるユーザー体験の悪化
- プラットフォーム共通のアクセスコントロールの提供が必要
- 複数の独立したノードに共通のUIを提供したい


エンドユーザーに
システムがブロックチェーンだからという理由で
我慢させてはならない

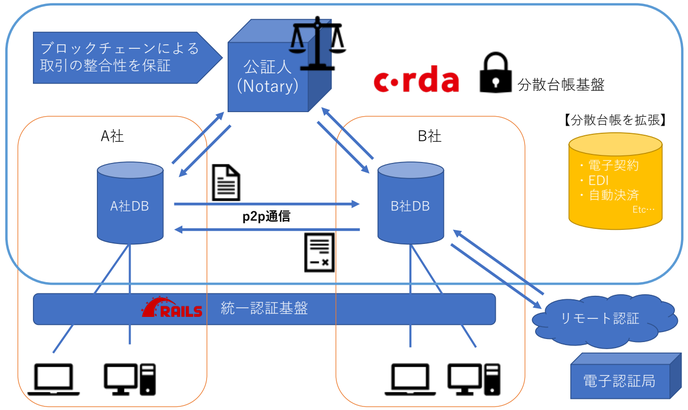
多数のノードを誰がどのように管理するか？

- プラットフォーム規模が増大するほど**管理すべきノードの数**は増える
  - 大規模PFでは**数千ノード**にのぼる予測
  - 本質的な分散台帳の特性上**各ノードの独立性**は可能な限り確保する
- 多数ノードを同時に**アップデート**・CorDAppを**デプロイ**する必要性
- BtoBに参加する企業の全てが**サーバー管理者**を用意できる訳ではない

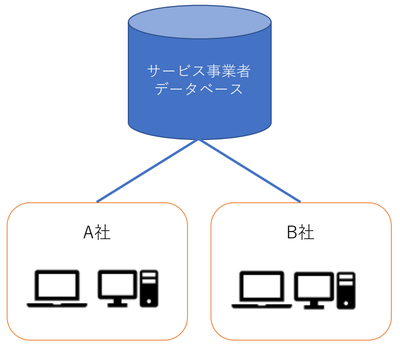
複雑なアーキテクチャの統制

中央集権的アーキテクチャとの比較

Corda



従来のシステム

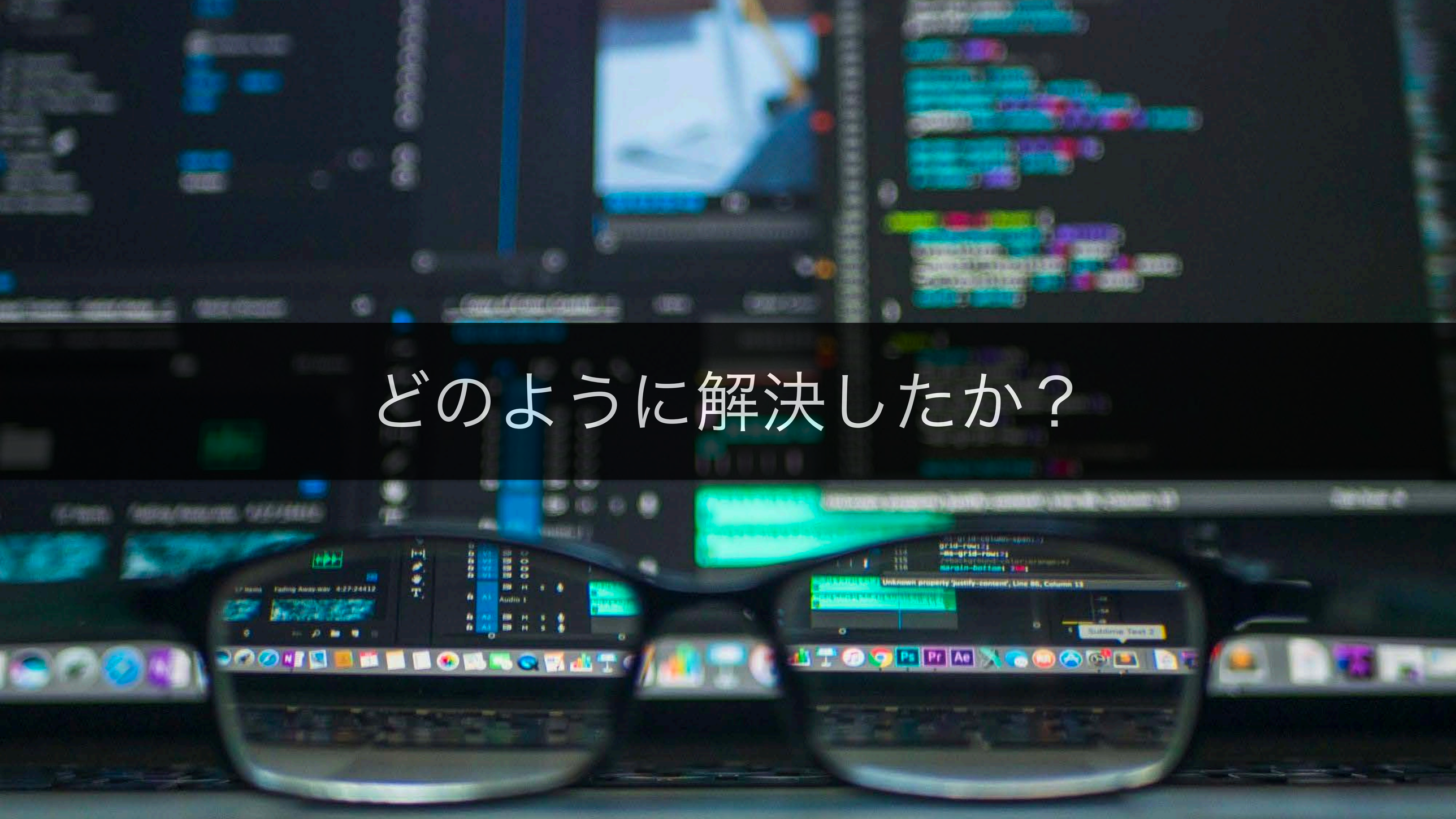


分散台帳は**本質的に複雑**

バックアップ・バージョンアップの困難

- 多数ノードのデータベースを**一つずつ独立にバックアップ**しなければならない
- ノードの一つに**障害が生じた場合のリストア**は誰が・どのように行うのか？
- CorDappのバージョンアップを**数千ノードに対し同時**に行う必要性
- **OS・ミドルウェアのバージョンアップ**は誰が責任を負うのか？

アーキテクチャレベルで
運用を考慮した設計が必須



どのように解決したか？

Agile
devOps
Kubernetes

クラウドネイティブアーキテクチャ

採用アーキテクチャ

- フロントエンド- Single Page Application
  - React + Redux
- バックエンド - API Gateway Model
  - AWS Cognito + API Gateway
  - Ruby on Rails
  - AWS Lambda
- Cordaプラットフォーム
  - Corda Enterprise
  - CENM(Corda Enterprise Network Manager) = Private Network

すべてのコンポーネントを

Kubernetes / Managed Serviceで管理

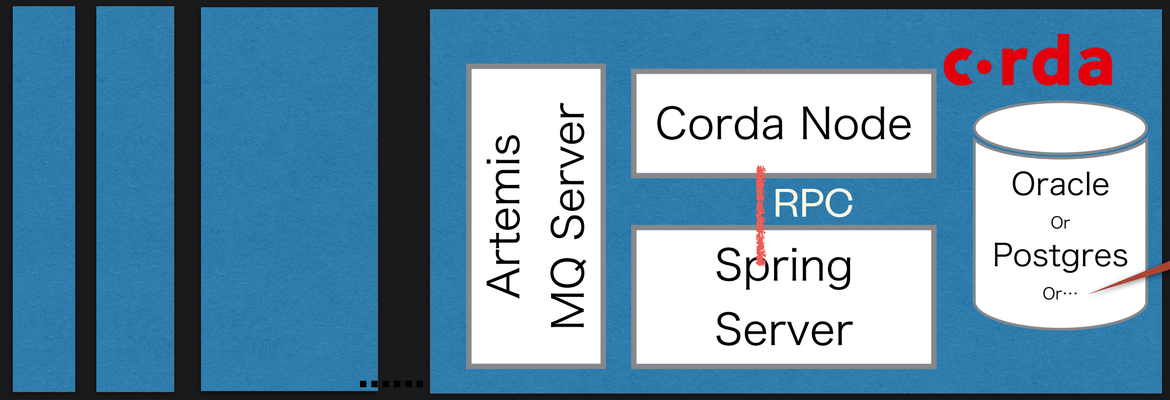
アーキテクチャ構成

Kubernetes

ノード管理
・新規ノードデプロイ
・既存ノード停止・更改

監視
・ロギング/モニタリング

独自オペレーター **CNAT**



Corda Ledgerで取引実行
取引データは全て各拠点ノードが独立して保持

Rails ~~API~~ Server

API Gateway

React - Single Page Application

取引データは持たない。
キューイング、バリデーション、ロールベース認可

Cognito

認証、
ユーザープール

アジャイルアプローチ

アジャイルアプローチ

機能ごとのデリバリー

- 要件定義・設計・実装と進むのではなく、アーキテクチャを横断する機能をとにかく作ってユーザーに見てもらう

ビジネス要求に合わせた素早いピボット

- 開発状況の詳細をクライアントと全て共有
- 発見したリスクやビジネス状況の変化を二週間ごとに反映

開発サイクル（例）

【開発内容】

- 取引申込み
- 申込み履歴一覧

【開発内容】

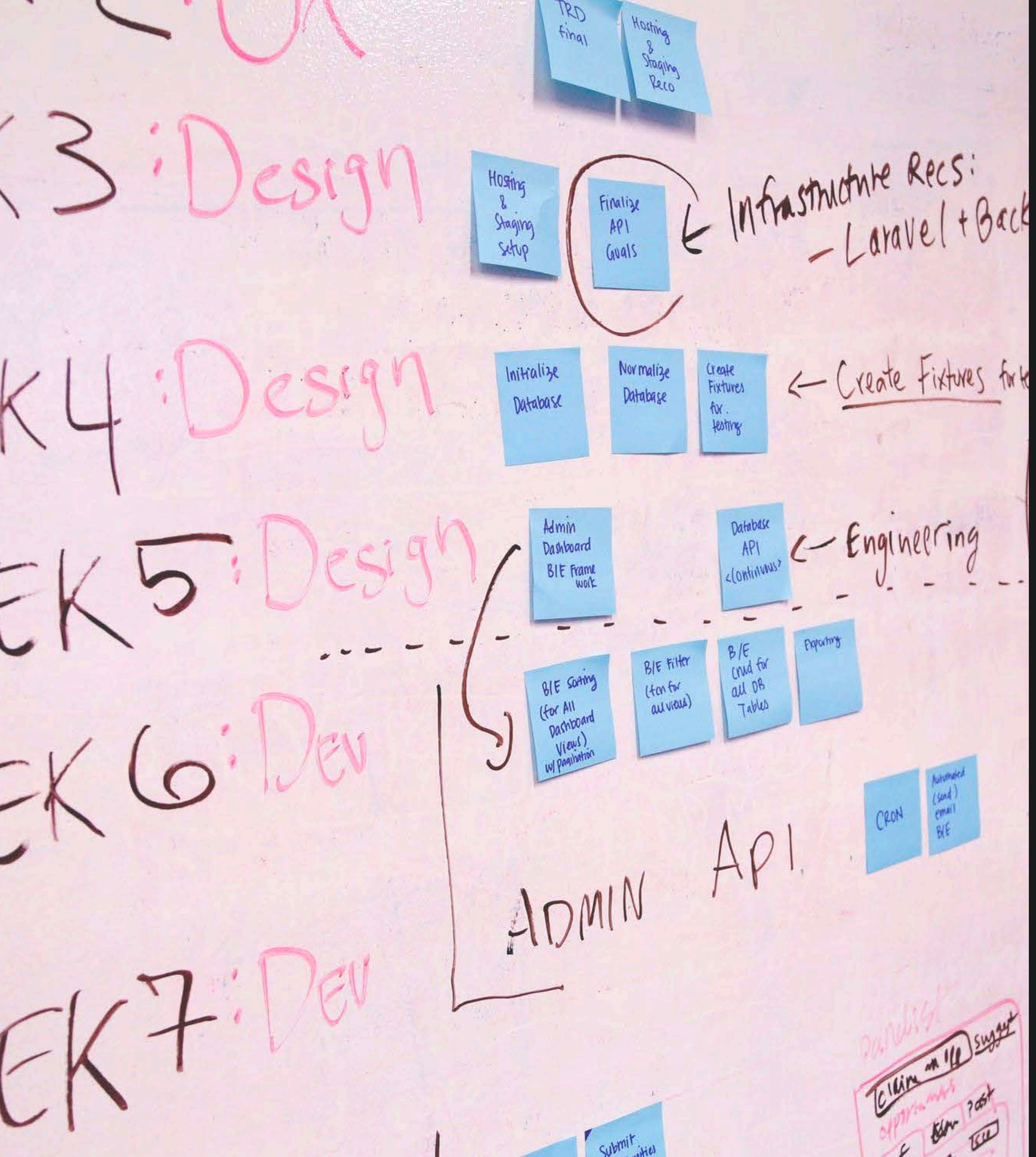
- 申込み受諾
- 受諾履歴一覧

【開発内容】

- 申込み条件要件変更
- 申込み否認
- バックアップリストア

. . . つづく

アジャイルアプローチにより、
ビジネスからフィードバックを得ながら
未知のリスクに対応できるようになった



アジャイルアプローチには
変化に強い設計とコーディング、
devOpsとデリバリ手段が必須

devOpsの実現・徹底

devOpsのすすめ

- Auto Testing
- Infrastructure as Code
- Single Source of Truth

Auto Testing

- 自動化された単体テストに**プログラマーが責任を持つ**
- Pull Requestごとに**CIがUnit/Integration Testを実行**する
- 全ての**重要な機能はE2Eテスト**を用意し、定期的に自動実行する

Infrastructure as Code

- 詳細設計書・手作業を前提とした手順書は**作成しない**
- 全ての設定と作業は**コードとして記述**し、リポジトリにコミットする
- 最小限の手順書と**コードベースでデプロイを実行**する

Single Source of Truth

- Infrastructure as Codeが前提
- Githubのmaster branchから**自動的にデプロイ**する
- 環境の設定・変更はすべてGithub上の**コードとして存在**する
- 手作業によるデプロイや変更作業は**一切行わない**

devOpsによって
要件の変更や未知の技術への
迅速な対応が可能になった。

※準備なく闇雲な変更をしてはならない

Kubernetesの採用

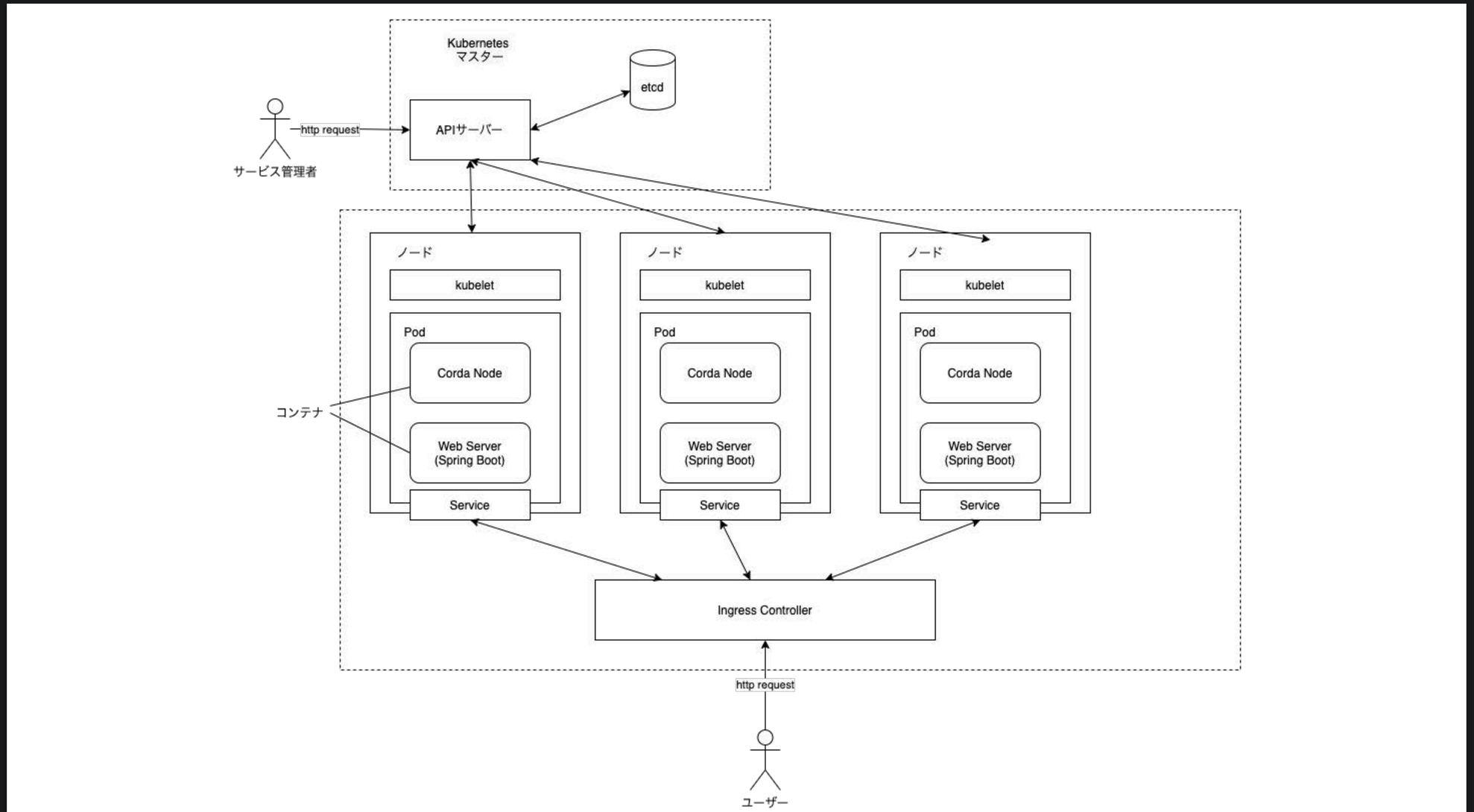
Kubernetesとは

- 自動デプロイ
- 構成管理
- 自動運用管理/セルフヒーリング
- オートスケーリング

Kubernetesで実現したこと

- ブラウザの管理画面からCorda NodeをAzure上にデプロイ/停止
- 全てのノードを対象に定時自動バックアップ
- http endpointへのリクエストでリストア
- gitOpsでCorda Enterpriseバージョンアップ
- 全ノードを対象にCorDAppのデプロイ/ロールバック

Kubernetes 構成概要



Kubernetes採用技術

- マネージドサービス
  - Azure Kubernetes Service
  - Amazon EKS
- デプロイ
  - Terraform
  - CircleCI（ビルド）
  - Azure Container Registory
  - Argo CD
- 永続ボリューム管理
  - Velero
- 監視
  - Prometheus
  - Loki
  - Grafana

Corda Nodeの制御を行う独自オペレーターを開発

CNAT(corda node administration tool)

- 独自オペレーター
- Kubernetesプラットフォーム上でCorda Nodeを管理
  - 新規ノードデプロイ
  - 既存ノード変更・停止
  - バックアップ
  - リストア

CNAT/Kubernetesを利用したノード管理

- 本番ノードをブラウザ管理画面からデプロイ
- ワンクリックでノードデプロイ/停止
- 管理画面からノードリソース（メモリ、CPU）管理
- 全ノードを対象にメトリクス/エラーログ監視
- GithubのリリースCorDAppのデプロイ

Kubernetesを採用したことで
独立ノードの管理・変更が可能になった。

Agile
devOps
Kubernetes

ご清聴ありがとうございました