RICOH
imagine. change.

開発者主導で回す GitHub Enterprise 運用

ルール可視化から始める改善サイクル

(株)リコー
リコーグラフィックコミュニケーションズBU
商用印刷事業本部 HW開発センター

山口 淳平

2026年3月24日 / OctoNihon Forum

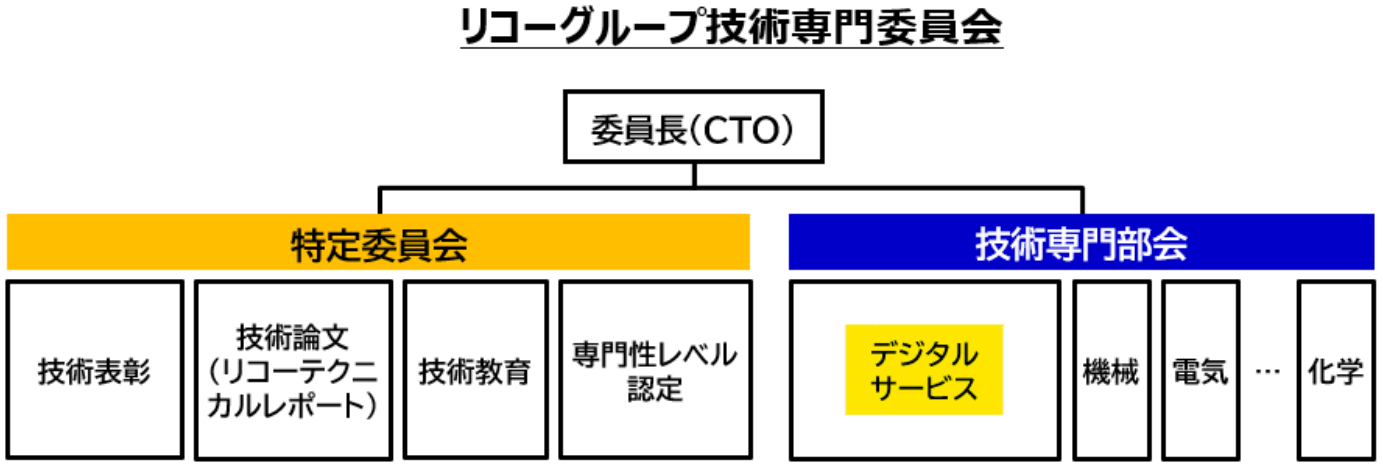
自己紹介

- 氏名：山口 淳平
- 所属：(株)リコー
  リコーグラフィックコミュニケーションズBU
  商用印刷事業本部 HW開発センター
  第三HW開発室

業務

- 本務は商用印刷プリンタの組込ソフト開発
- グループ横断のソフトウェア開発コミュニティ
  「デジタルサービス部会」のメンバーとして、
  GitHub Enterprise運用とInnerSource推進をリード

技術専門委員会 デジタルサービス部会



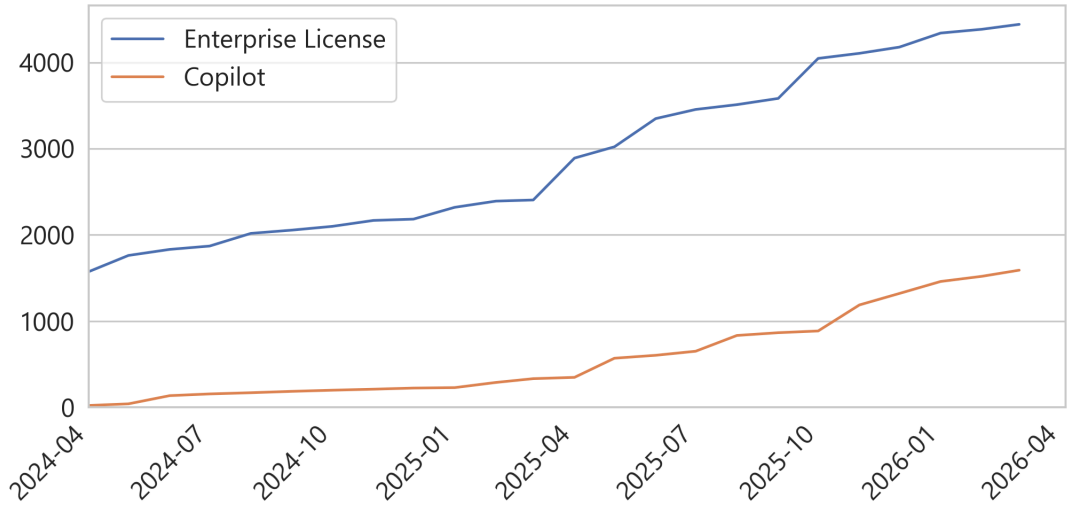
- **技術専門委員会**：リコーグループ内で、技術分野ごとの知見共有や連携を進める全社横断の技術コミュニティ
- **デジタルサービス部会**：デジタルサービス領域を担当し、知見共有・人材育成・部門横断の改善活動を進める

本日のアジェンダ

RICOH
imagine. change.

1. **背景**：GitHub Enterprise 利用の急拡大
2. **課題**：拡大に伴って生まれた運用上のギャップ
3. **取り組み**：開発者主導の運用自治会の発足と3つの取り組み
4. **まとめ**：見える運用、回る運用へ

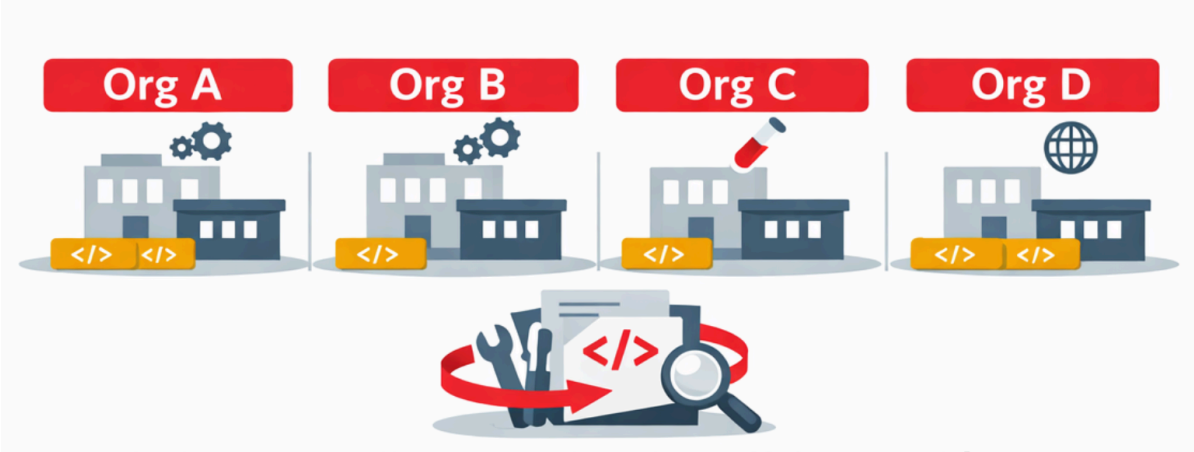
背景：GitHub Enterprise 利用の急拡大



- リコーグループ全体で GitHub Enterprise を活用
- リコーグループでは2023年末からGitHub Copilotが利用可能となり、**利用者・利用部門が急速に増加**
- **「ちゃんと使える環境」を整える必要性**が高まってきた

課題：拡大に伴って生まれた運用上のギャップ

課題①：Org の増加と部門間の共有のしにくさ



便利ツールを作ったけど、どこに共有すれば良い？

- 部門ごとにOrgを作成することが前提となっている運用だった
  - **100以上の Org** が作成される状態に
  - 部門ごとに Org が分かれ、**コードの発見・再利用がしにくい**
- 部門を越えてコードを共有したくても、
  共通の共有先となる Org がなかった

課題②：ポリシーの見えにくさ

- Enterprise 全体の設定・方針が**利用者から見えにくい**状態だった
- 「設定を変更してほしい」と思っても
  - **どこに相談すれば良いか**わかりにくい
  - **変更を依頼して良いのか**わかりにくい

課題②：具体的な困りごと

場面	何が起きていたか
Enterprise内限定でforkしたい	「Enterprise内のみ許可」という選択肢がない時期に禁止設定となり、その後も見直されていなかった
新しい Copilot モデルを使いたい	モデル追加サイクルが速く、IT部門だけでは現場ニーズへの迅速な対応が難しかった

→ **ルールが見えない・変えられない** ことで開発者体験が損なわれていた

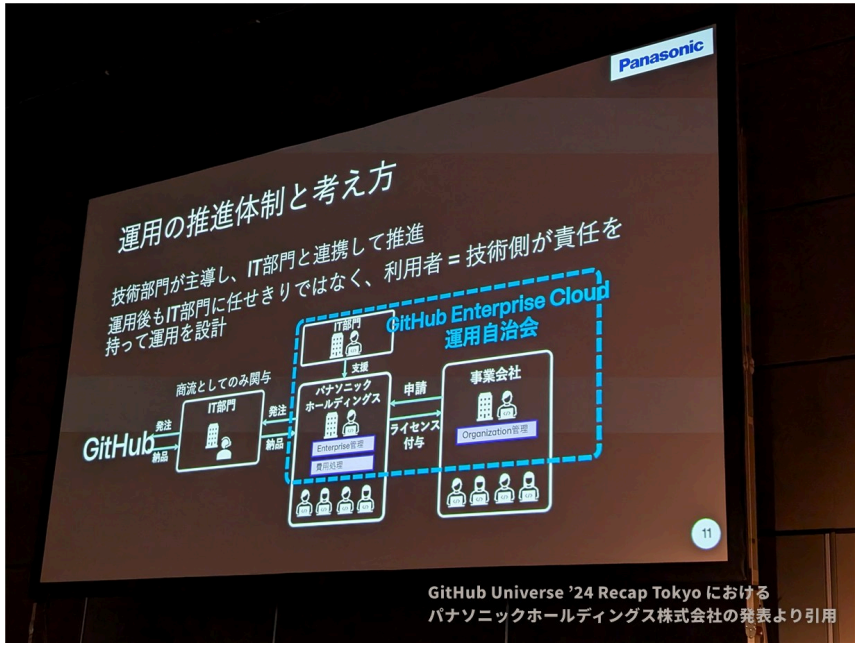
改善活動のスタート

改善活動のきっかけ

RICOH
imagine. change.

- 最初はGitHubを利用する一開発者だった
- 2023年12月、社内の GitHub LT 大会で登壇し、**課題提起**
- LT 直後、デジタルサービス部会のメンバーから声がかかる
  - 「**一緒に GitHub Enterprise の運用を改善しませんか？**」
- ここから、**運用改善活動**への参画がスタート

先行事例：パナソニックグループの「自治会」



- GitHub Universe '24 Recap Tokyo で紹介された事例
- 同様の課題を「**自治会**」という仕組みで解決
  - 技術者側が責任を持って運用を設計・運営
- パナソニックグループの事例を参考にし、**リコー式の自治会を設立**

GitHub Enterprise 運用自治会の発足

RICOH
imagine. change.

- 25年4月から活動開始
- **18名**のメンバーで構成
  - **10以上の部門・グループ会社**から集まった現場開発者
  - 上記に加えIT部門・セキュリティ部門など多様な立場から構成
  - **現場の声がそのまま運用に反映できる体制**を構築
- 25年10月から正式に**Enterprise管理者**として運用を開始

声を上げてから約2年、一開発者からEnterprise管理者へ

運用自治会の取り組み

1. 全社員が参加できる共有 Org の開設
2. ルールを GitHub で公開する
3. 検証用 Enterprise 環境（sandbox）の構築

① 全社員が参加できる共有 Org の開設

rfgricoh

リコーグループのメンバーが参加可能なOrganization

17 followers　Part of RICOH Company, Ltd.

README.md

🎉 リコーグループ GitHub Organization `rfgricoh` へようこそ

`rfgricoh` は、リコーグループの開発者のための Organization です。
部門やチームの壁を越えて、知識を共有し、一緒に開発を楽しみましょう！

📋 基本ルール

🗂️ リポジトリの作成について

リポジトリは自由に作成できますが、以下のルールを守ってください：

- **どんなリポジトリを作っても構いませんが、製品テーマ等の場合は、アクセス権の設定に注意し、適切なアクセス権を付与してください**
  - アクセス権管理の考え方については、アクセス権管理 - GitHub活用ガイドを参照してください。
- **ガイド、テンプレート、サンプルコードなどのナレッジ共有も大歓迎！**
- 迷ったら問い合わせ先に気軽に相談してください

リポジトリを作るときのポイント

- 必ず `README.md` を含めてください
- テンプレートリポジトリ rfgricoh/repo-template を用意してますので、ぜひ使ってください！
- `rfgricoh` 内の全メンバーにアクセス権限を付与したい場合、全メンバーが所属する ALL_MEMBERS チームが便利です

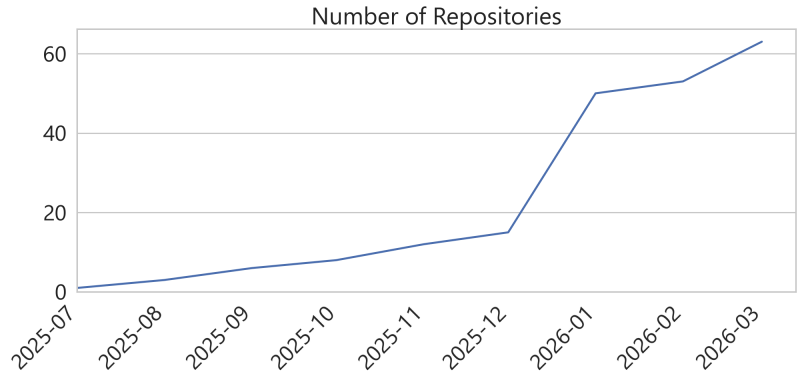
📛 リポジトリの命名規則

個人用のプロジェクトの場合は、以下のように `u-{username}` をprefix として使用してください：

- `u-{username}` ：個人プロジェクト一般
- `u-{username}-{project-name}` ：特定のプロジェクト用

- 部門ごとにOrgが分かれていたため、ライセンスを持つ**全グループ社員が参加できる** Org を作成
- コードやナレッジの部門横断的な共有により、**InnerSource を推進**
- 「**作ったものをグループ内で共有したい**」という声に応える場所

① 共有 Org を盛り上げる工夫



- リポジトリの命名規則により個人的なリポジトリも作成しやすく

> #### リポジトリの命名規則
>
> 個人用のプロジェクトの場合は、以下のように `u-{username}` をprefix として使用してください：
>
> - `u-{username}` ：個人プロジェクト一般
> - `u-{username}-{project-name}` ：特定のプロジェクト用

- リポジトリ数も徐々に増え、60を突破

② ルールを GitHub で公開する

RICOH
imagine. change.

概要

この文書では、GitHub Enterpriseの「Policies」タブで設定可能な各種ポリシーと、現状の設定について説明します。 これらのポリシーは、Enterprise内の全てのOrganizationに適用されるグローバルな設定です。

> ⓘ Note
>
> GitHubの機能追加に伴って設定項目が変わることが多いため、この文書ではユーザー側の利用に関わる主要な設定項目のみを記載しています。 全ての設定項目を網羅しているわけではないことにご留意ください。

Member privileges

Unaffiliated users

概要

あるメンバーが全てのOrganizationから削除されると、Unaffiliated users(未所属ユーザー)になります。この設定では、そのようなユーザーをEnterprise内に残すか、完全に削除するかを選択できます。

設定項目

1. Remain in the enterprise
   - Unaffiliated usersがEnterpriseに留まることを許可します。Organizationに所属していなくてもEnterprise内に存在できます。
2. Remove from the enterprise
   - ユーザーが全てのOrganizationから削除された場合、Enterpriseからも削除されます。

現状の設定

1. Remain in the enterprise

ポリシーの透明化

- Enterprise 全体の**設定・ポリシーを Markdown 化しリポジトリ管理**
- **共有 Org 内**で公開し、ライセンスを持つ全員が閲覧可能

参考： 信頼できる開発プラットフォームをどう作るか？- Governance as Codeと継続的監視／フィードバックが導くPlatform Engineeringの進め方

② 変更理由をプルリクエストに残す

フォークポリシーの有効化: Enterprise内組織間でのフォークを許可

Merged nmoa merged 2 commits into main from change-fork-policy on Nov 18, 2025

Conversation 1 Commits 2 Checks 0 Files changed 1

nmoa commented on Nov 12, 2025 • edited

設定の変更内容

Repository forking ポリシーの変更

- **変更前:** `All organizations: Disabled`
- **変更後:** `All organizations: Enabled`
  - フォークを利用可能な範囲: `Organizations within this enterprise`

設定変更の理由・経緯

プライベートおよびinternalリポジトリのフォークを有効化することで、以下のメリットを実現します:

- Enterprise内のOrganization間でのコラボレーションを促進
- 開発者が安全な環境でコードの実験や改善を行える
- プルリクエストベースの開発ワークフローをサポート

フォーク範囲を「Organizations within this enterprise」に限定することで、Enterprise外へのコード流出を防ぎつつ、内部コラボレーションを可能にします。

変更に伴う影響と注意点

ポジティブな影響

- Enterprise内のOrganization間でリポジトリをフォークできるようになる
- Organizationをまたいだコラボレーションが容易になる

注意すべき点

各 Organization ではデフォルトでフォークが有効化されるので、Organizationの管理者は必要に応じて、Organization レベルでのフォークポリシー設定を各自で行うこと。

1. `https://github.com/organizations/[自分のOrg名]/settings/member_privileges` のページにアクセスし、「Repository forking」の欄に移動する

意思決定の経緯を記録

- プルリクエストを作成することで **変更の背景や理由が GitHub 上に記録される**

② 「誰でも提案できる」仕組みへ

RICOH
imagine. change.

ポリシー変更のルール

このドキュメントでは、Enterprise設定やポリシーを変更する際の手順を説明します。

変更手順

1. プルリクエストの作成

このリポジトリで、変更したい設定を記載したプルリクエストを作成します。

- 対象ファイル: `enterprise-settings/Policies.md` または `enterprise-settings/AI Controls.md`
- ブランチ: `main` から新しいブランチを作成してください

2. Teams チャネルでの告知

作成したプルリクエストについて、Teams の GitHub Enterprise 運用自治会チャネル に投稿します。

投稿時には **@運用自治会** にメンションしてください。

3. レビューと承認

運用自治会のメンバーによるレビューを待ちます。

承認条件

変更内容によって、以下の承認条件が適用されます。

A. GitHub Copilot モデルの有効化のみの場合

- 利用可能な GitHub Copilot のモデルが増えて、該当モデルを "Let Organization Decide" に設定する変更については、規定数の承認を必要とせずにマージして良い
- 理由: モデルの有効化に関しては特段のリスクがなく、素早く有効化できる形にしたほうが GitHub Copilot の有効活用につながるため

B. その他の設定変更の場合

- 特に意見等が出ていない、もしくは意見等が解決している状態で、運用自治会の利用区メンバーの **過半数の承認** が得られること
  - 2025/12/24時点での利用区メンバーは13名。
- 起票者が自治会メンバーの場合は、起票者を含めて過半数になれば良い

ポリシー変更のルールも明文化

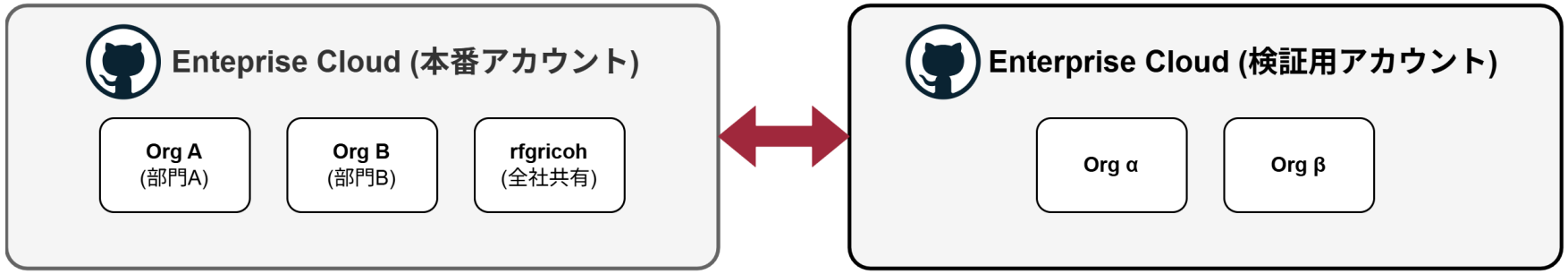
- 変更の提案から承認・反映までの**ルールもリポジトリ内で明文化**
- ルールが明確になることで、**誰でも安心して変更に参加できる**

② Copilot 新モデル、GA 当日に有効化

迅速な意思決定

- Copilot への新しい AI モデル追加時は即有効化できるようなポリシーとした
- Issue Formで追加リクエストを起票すると、自動でポリシー変更のPRが作成される仕組みを構築
- モデルがGAとなった日の
  朝には新しいモデルが有効化！

③ 検証用 Enterprise 環境（sandbox）



- 本番アカウントとは別に**検証用 Enterprise Cloudアカウント**を用意
- 目的
  - ポリシー変更の挙動確認
  - 自動化ワークフローの安全なテスト
- **本番環境を守りながら検証**できることで、変更や実装への心理的なハードルが下がった

まとめ：運用自治会の取り組み

RICOH
imagine. change.

1. 全社員が参加できる共有 Org の開設
2. ルールを GitHub で公開する
3. 検証用 Enterprise 環境（sandbox）の構築

まとめ：見える運用、回る運用へ

RICOH
imagine. change.

Before → After

Before	After
部門ごとにOrgが分かれ、 部門をまたいだ共有ができない	全社共通 Org で誰でもナレッジを共有
ルールが見えない、変更提案できるか分からない	ポリシーをGitHubで公開し、誰でも提案可能
Copilot モデル追加の対応が追いつかない	GA 当日の朝には新モデルが有効化

ー開発者の問題提起から、開発者主導の運用体制へ

みなさんの環境でも、**「運用をエンジニアリングする」** ヒントになれば幸いです。