Engineer

ソフトウェア開発のベストプラクティス

目的と適用条件が明確で、効果を確かめられる開発手法

どのようなものか

ソフトウェア開発のベストプラクティスとは、特定の条件で効果が確かめられ、通常の選択肢として使える方法です。たとえば復旧時間を知る必要があるなら、別の環境でバックアップを復元する方法があります。バックアップの処理が完了した記録からは、コピーが実行されたことが分かります。復元まで試すと、データを実際に開けるか、どれだけ時間がかかるか、担当者のその場の工夫に頼る箇所があるかが分かります。

研究組織やプロジェクト、標準化団体が方法を推奨することがあります。推奨には、チームが独立して変更を展開できる、または変更を確認できる人がいる、といった前提が伴います。チームは参考として採用する場合も、共通の規則にする場合もあります。契約などで別途義務が定められていることもあります。誰が何のために決めた規則かが分かれば、適用範囲や例外を認められる人を確認できます。

NISTのセキュアソフトウェア開発フレームワークのバージョン1.1は、さまざまな開発工程に組み込める高水準の実践項目をまとめ、各項目に作業や実施例を示しています。この構成は、たどり着きたい結果と、そこに至る一つの方法を分ける参考になります。「変更をレビューした証拠を残せる」は結果で、あるサービスの特定の設定を有効にするのは実施方法の一つです。別のチームは、手元の工程で同じ証拠が得られ、必要な保護が保たれるかを確認できます。

正式な名称としての使い方もあります。IETFはRFC 2026の第5節に記載された手続きでBest Current Practiceの文書を発行します。RFC 8174は、関連する規約が定義する用語を大文字で使った場合に、どのような条件で特別な意味を持つかを明らかにしています。一方、社内文書でベストプラクティスと呼ぶものは、一般的な推奨事項である場合があります。推奨の出典と適用範囲を確かめてから、どこまで拘束力があるかを判断します。

概念を判断に生かす

「ソフトウェア開発のベストプラクティスとは？」を実際の判断に生かしましょう。その概念を、開発または購入するシステムに結び付けて考えるお手伝いをします。 変更を検証し、リリースし、元に戻せる仕組みを整え、金曜の深夜ではなく火曜の午後にお客様へ届けられるようにします。

サービスを見る お問い合わせ →

用語の背後にある前提を検証する

解説とリンク先の実績を参考に、プロジェクトについての質問を整理してください。ツールを選ぶ前に、前提を一緒に検討できます。 GitHub ActionsでGIS SaaSのデプロイを自動化 — デプロイ、データ、報告を手作業から解放

GitHub ActionsでGIS SaaSのデプロイを自動化 お問い合わせ →

なぜ重要なのか

方法が明確になると、変更のたびに同じ判断を繰り返す時間が減ります。新しく加わった人は、レビューを頼む前に何を用意するかが分かります。運用担当者は複数回の展開について比較できる情報を受け取れます。サービスを共有する複数のチームは、どんな変更が互いへの連絡を必要とするかを決められます。「よく確認する」では判断が人によって変わります。「同じ注文で二回請求されないことを示す」なら、実際に検証できる結果について話せます。

通常の方法を決めておくと、例外も見つけやすくなります。小さな変更を頻繁に出せるサービスもあります。機器のファームウェアを更新する場合、現地で担当者が機器を隔離し、更新し、動作を確かめる必要があるかもしれません。変更を小さくすることは、設計、試験、機器を更新する順序に役立ちます。実際に機器を触る頻度は、復元にかかる時間や現地の体制を考えて決めます。

DORAは、共通のメインブランチで開発する方法を、小さな変更を頻繁に統合するものとして説明しています。長く残るブランチをまとめるのに時間がかかるチームにとって、検討する価値のある方法です。導入するチームには、手元の結果を確認する仕事が残ります。変更を統合するまで何日かかるか、何が差し戻されるか、展開後に何が起こったかを見ます。ブランチが少なくなっても、一つにためる作業が増えたのであれば、統合が楽になったかは分かりません。

実際の活用例

社内の集計ツールを担当するチームを考えます。変更は別の担当者が確認し、共通のテストを通るのが通常です。ある月、抽出条件の誤りで月次報告が遅れました。全テストに数時間かかるのは、一部が別の出力形式の確認も行うからです。今回変更するのは抽出条件だけで、報告書を配る前なら元に戻せます。チームは、どの検証があれば今回の変更に限って十分かを考えます。

まず、報告書の配布前に承認が義務付けられていないかを調べます。必要な承認は、急ぎの場合も得ます。次に、具体的なリスクを示します。正しい行を含め、従来の集計単位を保ち、他部門のデータが含まれないようにする必要があります。開発者が結果の分かっているデータを使って試験し、実際の行を期待値と比べます。別の担当者が変更と比較結果を確認します。前回の報告書を手元に残し、配布を中止できる人も決めます。全テストは後で実行し、より速い標準の手順が必要かを検討する材料にします。

例外の記録には普段の規則、例外を認める理由、維持する保護策、例外の期限を書きます。実際にかかった時間と、結果も残します。同じことが翌月も起きるなら、この報告書専用の高速な検証を普段の手順に組み込めます。組み込む前に、わざと月を誤らせた入力や、除外すべき部門の行を与えます。両方の誤りを検出できれば、単に所要時間を測るだけでは得られない根拠になります。

二十二本のリンターを使ったコミット時の検査の事例は、プロジェクトで合意した確認項目を実際に守る仕組みを示しています。自動検査から漏れたアクセシビリティーの問題の事例は、毎回同じ検査を実行できても、利用者が体験する問題すべてをその検査で確認できるわけではないことを示しています。

問題が起きるところ

チェック欄を埋めることが目標になる。 用紙が埋まっていても、防ぎたかった不具合は起こり得ます。方法の違いが結果に表れる例を選び、実物を確認します。復元されたデータを開けるか、意図した利用者に新しい版が届いたかを確かめます。

道具を導入して作業が変わったと思う。 レビューのサービスを契約しても、変更に異議を唱える時間や背景知識までは得られません。何を実現するための道具かを書き、実際の変更で、必要な情報が間に合うかを確かめます。

例外がいつの間にか既定の方法になる。 急ぎの仕事に使えた方法は繰り返し使われがちです。範囲、責任を持つ人、残るリスクを減らす措置、見直す時期を決めます。同じ例外が続くなら、本来の方法を公開の場で見直すきっかけになります。

前提を確かめずに手法をまねる。 頻繁に公開できるネット上のサービスと、現地作業が必要な設備では条件が違います。元の手法が解決する問題と手元の問題を比べ、小さく試します。何が改善で、どの結果が中止の目安になるかをあらかじめ決めておけば、観察に基づいて採否を判断できます。

参考資料

- NISTのセキュアソフトウェア開発フレームワーク、バージョン1.1 — 実践項目、作業、実施例をまとめています。
- RFC 2026の第5節 — IETFがBest Current Practiceを扱う手続きを記述しています。
- RFC 8174 — 仕様文書で大文字の用語が規範的な意味を持つ条件を明確にします。
- DORAによるメインブランチを使った開発の解説 — 小さな変更を頻繁に統合する方法と、その影響を調べる手がかりです。

用語集

実際の制約に合わせて選ぶ

デプロイの流れ、運用上のリスク、緊急対応の担当者をお知らせください。検査、復旧手段、明確な責任の所在を伴う変更の範囲を定められます。

料金を見る お問い合わせ →

用語を実務に結び付ける

「ソフトウェア開発のベストプラクティスとは？」がお客様の組織にどう当てはまるかをご相談ください。現在のシステムの具体例が一つあれば、話を始められます。

サービスを見る

welcome@engineer.company +45 50 10 61 84 デンマーク、コペンハーゲン

このページを持ち歩く 連絡先カード

心を込めて丁寧に制作： D.

CVR 45969703 · Tagensvej 105C, 3. th, 2200 København N

著作権 © 2025 – 現在 · Engineer ApS · すべての権利を留保します。

印刷元： https://engineer.company/ja/glossary/engineering-best-practices/