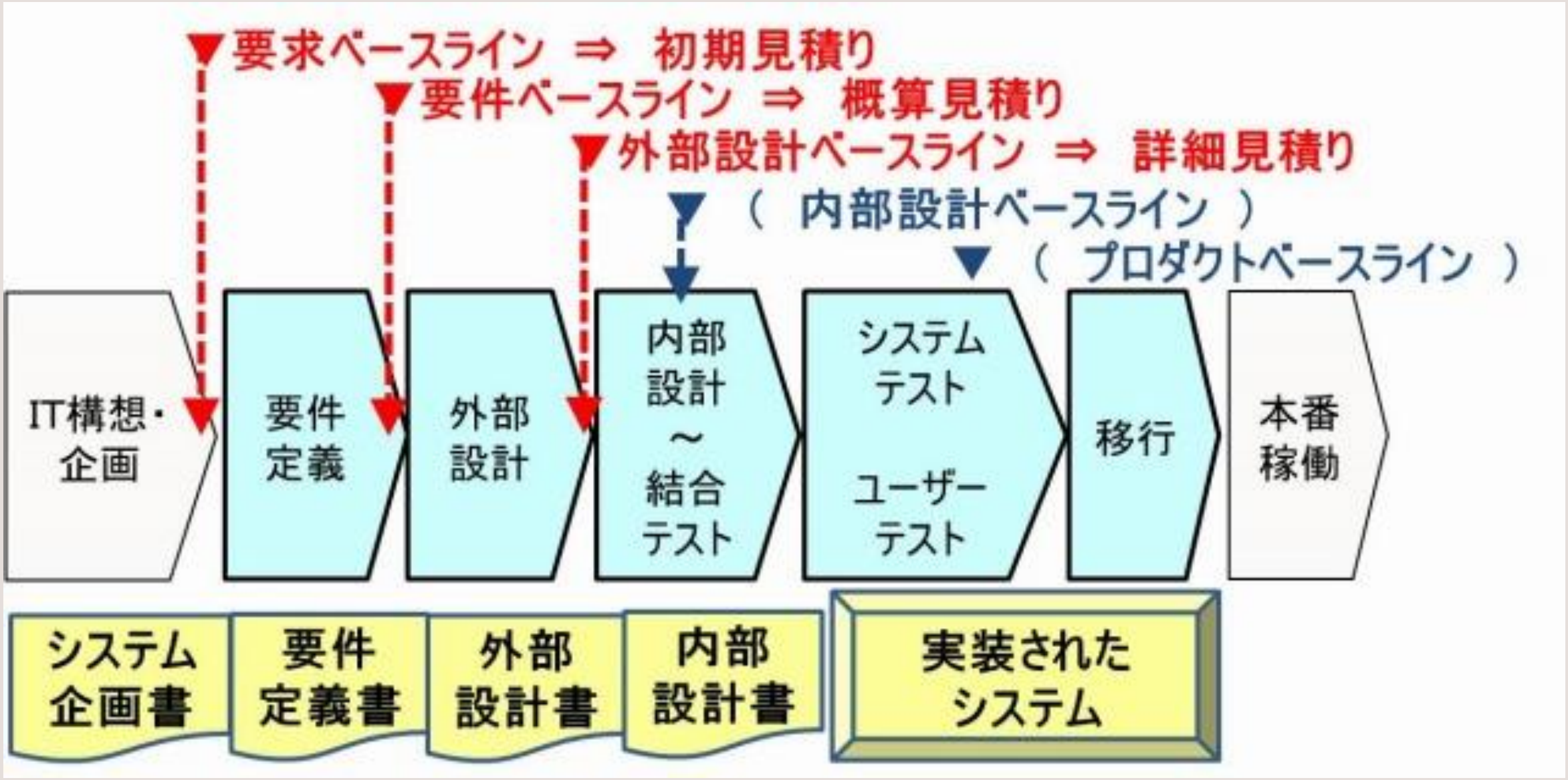
システム設計実践演習
第１１回

仕事の流れについて


要求ベースライン ⇒ 初期見積り
要件ベースライン ⇒ 概算見積り
外部設計ベースライン ⇒ 詳細見積り
（ 内部設計ベースライン ）
（ プロダクトベースライン ）
IT構想・企画
要件定義
外部設計
内部設計～結合テスト
システムテスト
ユーザーテスト
移行
本番稼働
システム企画書
要件定義書
外部設計書
内部設計書
実装されたシステム

システムリリース

納品

受入テストで問題が無ければシステムの開発は完了です。

ベンダーは発注者にシステムを納めます。これを**納品**といいます。納品する内容は、一般的に契約時の資料に記載します。

多くの場合、構成するプログラム(実行ファイル)だけでなく、そのソースコードやテスト資料、マニュアルなどを合わせて、納品書とともに納品します。

納品

ただし、開発したシステムがWebシステムなどで、開発後の運用もそのベンダーに任せる場合は、プログラムやソースコードがあっても発注者が使うことがありません。

この場合などは、テスト資料やマニュアルなど確認に必要なものだけ受け取ることが一般的で、**システムがWebサーバ上に設置されて公開されたことをもって納品とすることもあります**。

検収

システムがベンダーから納品されると、発注者は**検収**を行います。検収は、納品されたもの（ソースコードやテスト資料、マニュアルなど）を見て、発注したときの仕様や契約内容にあっているかを確認する作業です。これらに問題が無ければ、検収書を作成・押印し、ベンダーに渡します。

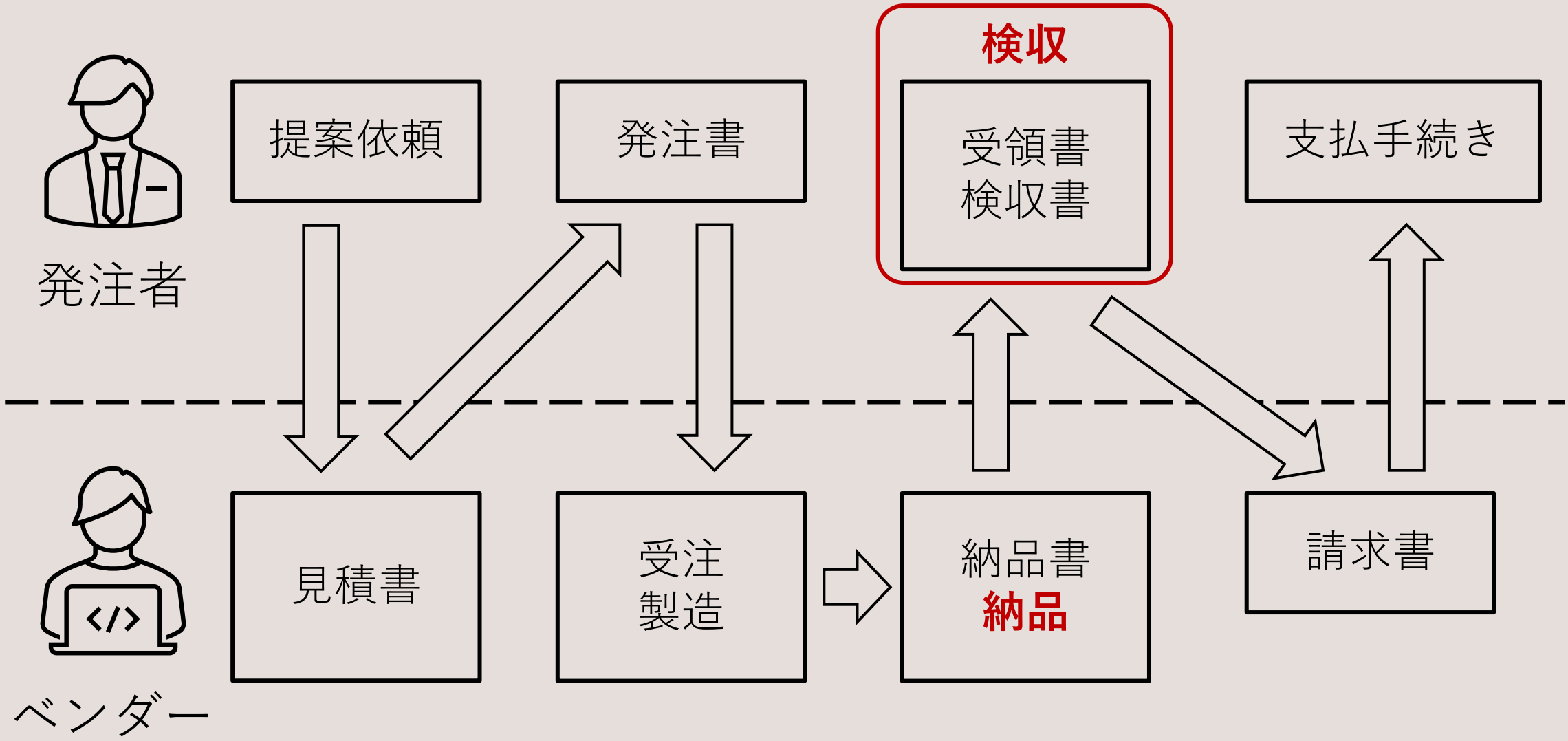
検収が完了すると、正式に受け取ったものとして扱われ、ベンダーからの請求に対する支払いに進みます。

検収

検収前の段階で不具合が見つかった場合はベンダー側のミスとみなされますが、検収が終わったあとに不具合が見つかった場合は発注者側のミスと見なされます。

ただしシステムは使ってみないと中身がどのようになっているのか判断できない部分もあります。例えば、半年に１度しか使わない機能であれば、そのときになって初めて判明する不具合も存在します。

システム 開発依頼の大まかな流れ



契約不適合責任

検収が終わってシステムの運用が始まっても１年程度の間に不具合が見つかった場合には、ベンダーが無償で対応することが一般的です。これを**契約不適合責任**(旧：**瑕疵担保責任**)といいます。

契約不適合責任の期間は民法で定められています。

２０２０年までは瑕疵担保責任でしたが、民法改正によりその施行より後の契約では契約不適合責任に変更となり、権利を主張できる期間などが変更されました。

契約不適合責任(旧：瑕疵担保責任)

名称	内容	権利主張の期間	権利行使の時効
瑕疵担保責任	・契約解除 ・損害賠償請求	納品後１年	納品後１０年
契約不適合責任	・契約解除 ・損害賠償請求 ・追完請求 （不具合の修正） ・代金減額請求	契約不適合を 知ってから１年	不具合を発見してから５年 納品後１０年

デプロイとリリースの違い

制作したWebシステム、設定ファイルなどを他の環境に移行することをデプロイと以前説明しました。デプロイは本番環境に移行することだけでなく、開発環境から検証環境に移行することも含みます。

また、システムを本番環境にデプロイしてもユーザがその事実を知らなければ利用されないため告知をする必要があります。

「**開発したシステムを利用者に知ってもらい、実際に使ってもらう状態**」にすることを**リリース**と言います。

デプロイの手法

Webシステムをデプロイするときに大きな改修の場合、その配置に時間がかかります。移行途中にWebサーバーにアクセスしてきた利用者の端末では、古いバージョンと新しいバージョンが混在する状態になり、挙動がおかしかったりエラーが発生します。

これを防ぐために、**デプロイするときにはシステムの一時的な停止やWebサーバーの再起動が必要**になることもあります。

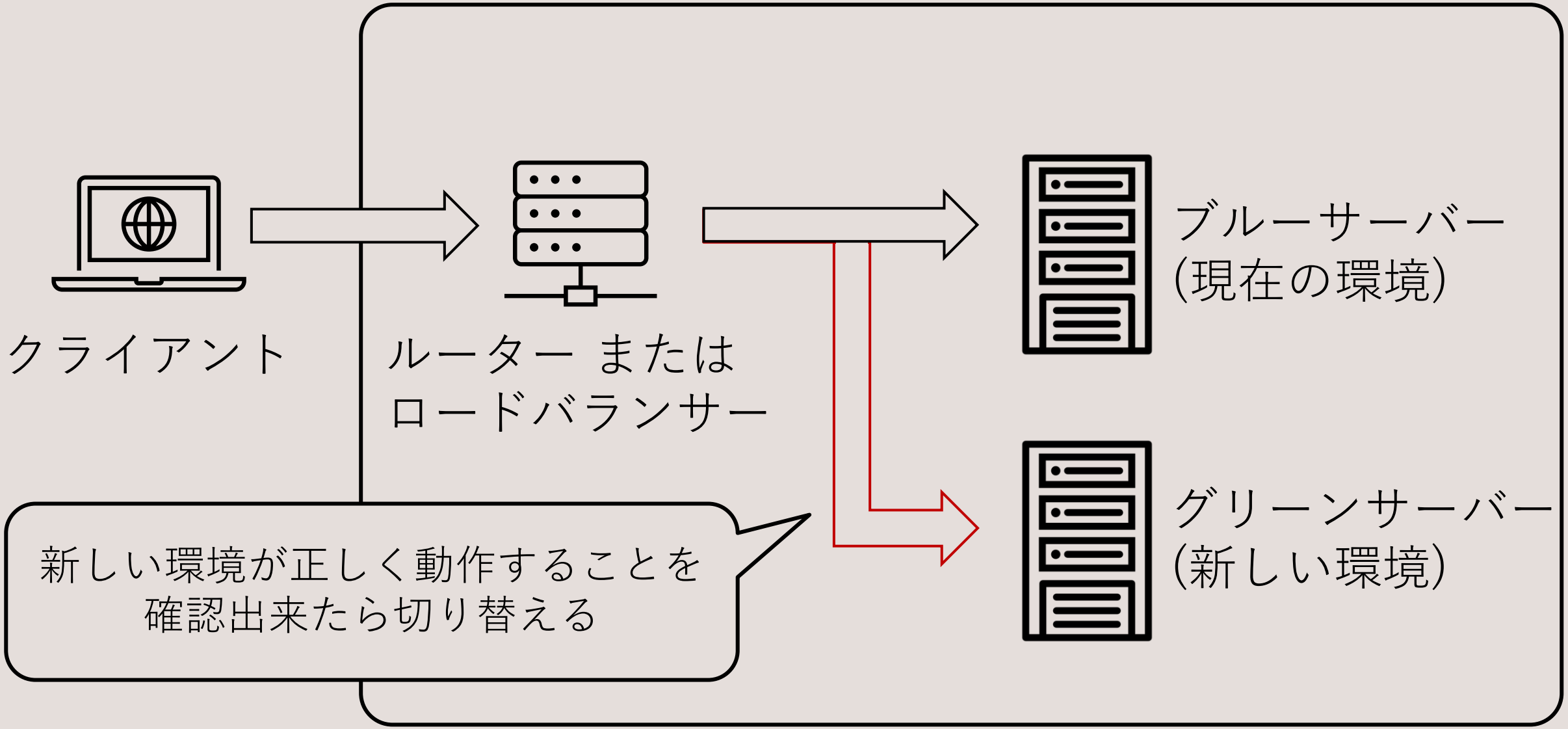
数時間の停止が許されるなら、事前告知してメンテナンス中にすることを検討します。

ブルーグリーンデプロイ

ECサイトのようなシステムでは２４時間３６５日いつでも使えることが当たり前になっています。このようなシステムでは、短時間の停止も許されない状況があります。そこで、サービスを停止せずにデプロイする方法として、**ブルーグリーンデプロイ**があります。

「旧システム(ブルー)」と「新システム(グリーン)」の２つのサーバーを用意して、**新システムが正しく動作することを確認してから、ネットワークの接続を切替えることで、停止時間を最小限に抑えることが出来ます**。

ブルーグリーンデプロイ

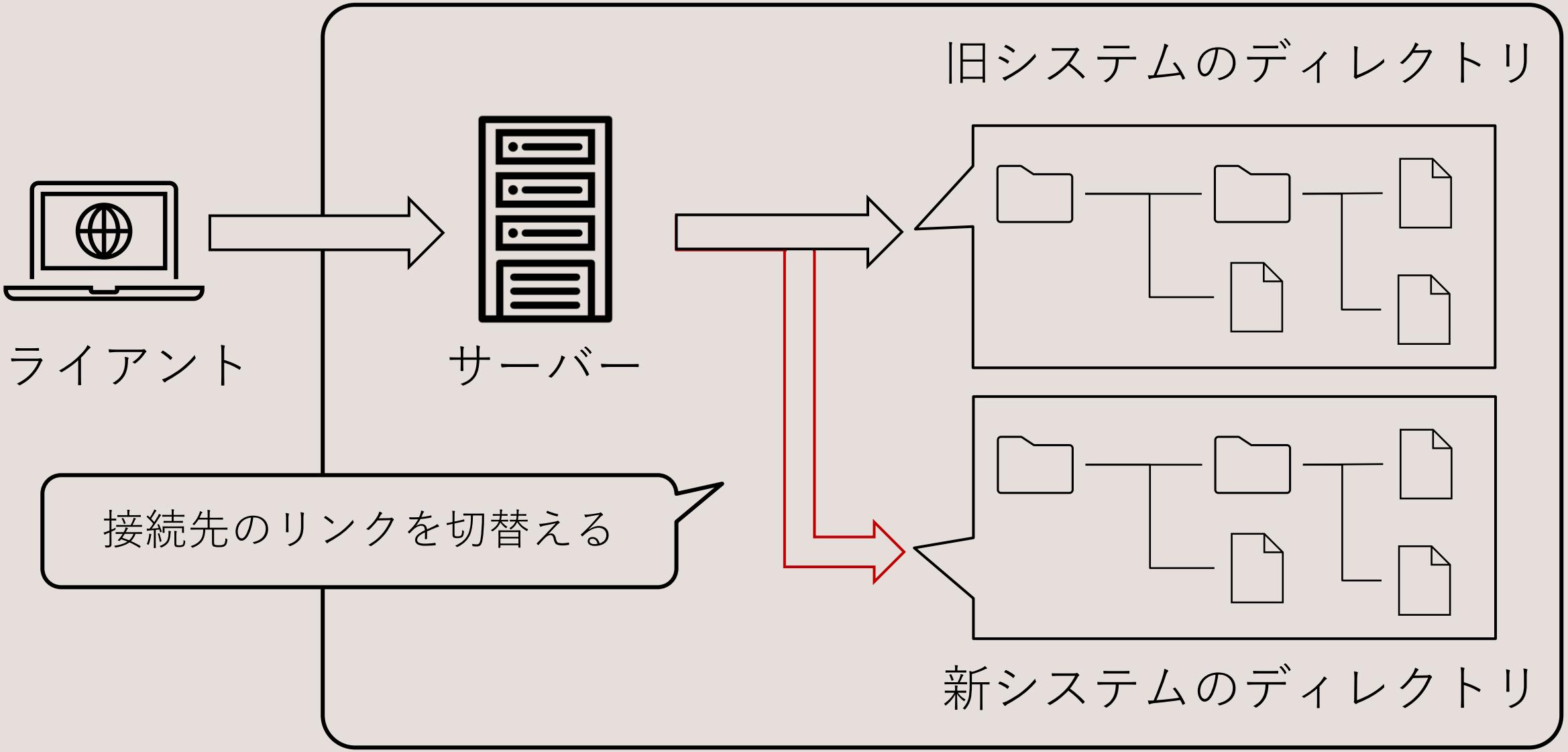


ブルーグリーンデプロイ

この切替には、ルーターやロードバランサーを使用します。

１つのサーバーしか契約していない場合は、複数ディレクトリを用意してリンクを切替える方法が良く使われます。

ブルーグリーンデプロイ



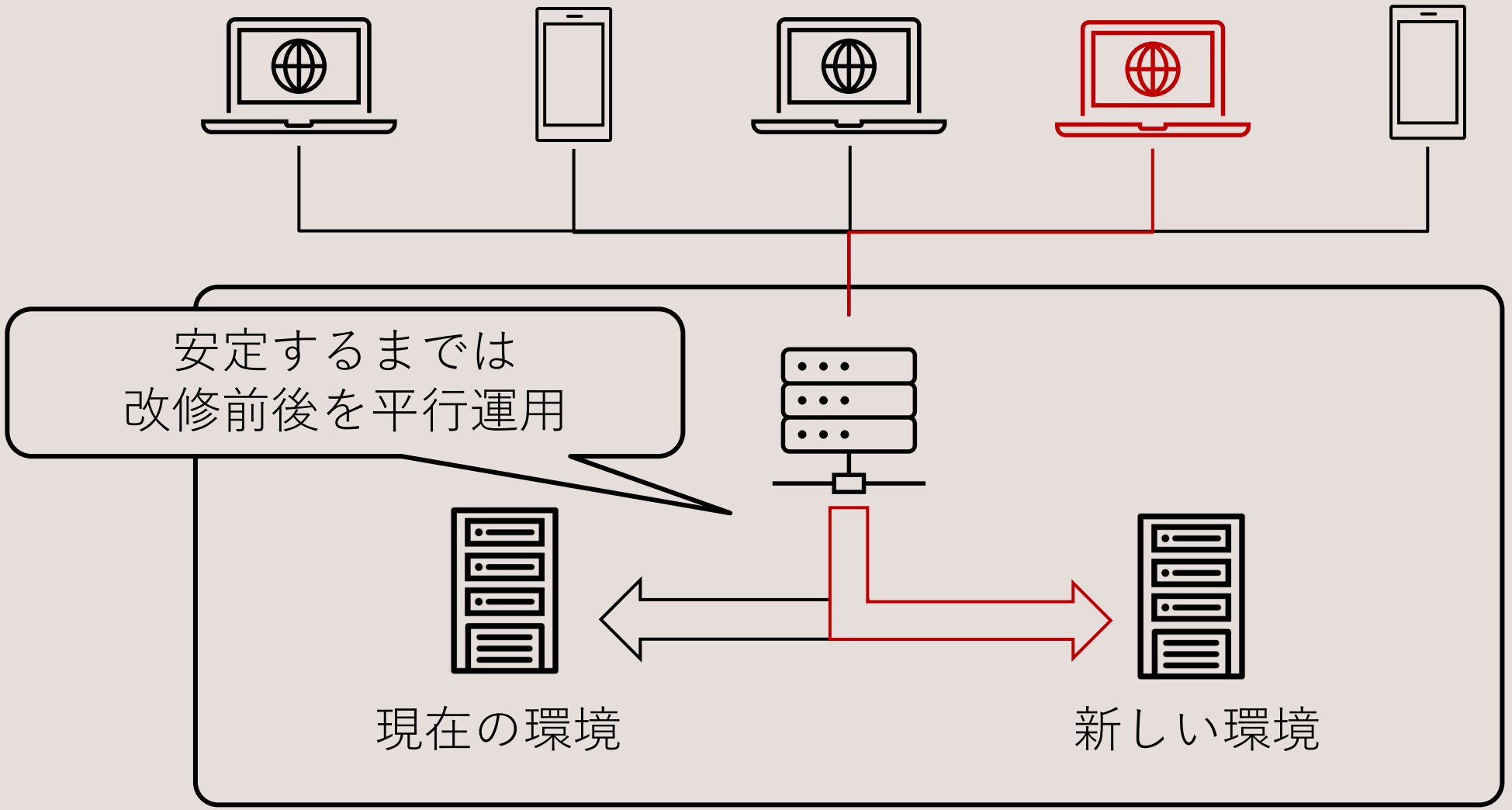
ローリングデプロイとカナリアリリース

すべての利用者をまとめて新システムに移行するのではなく、一部の利用者だけを新システムに移行して様子を見る方法もあります。これを、**ローリングデプロイ**と呼びます。

複数サーバーを用意しておき、ロードバランサーを用いて一部のユーザだけ新しいサーバーに振り分ける方法です。少しずつ利用者を新しいサーバーに振り分けることで、問題の発生する利用者を最低限に抑えることが出来ます。

このような方法は**カナリアリリース**とも呼ばれます。

ローリングデプロイとカナリアリリース



運用と保守

障害とは

システムの運用を開始した後で、何らかの原因でシステムが停止したり、応答するまでにいつもより時間がかかったりすることがあります。このようにシステムが使えなかったり、使い物にならなかったりする状態を**障害**と言います。

利用者側の問題で起きる障害もありますが、**サーバー側で障害が発生すると影響が大きいため、早期に対処する必要があります**。

障害の原因

障害が発生する原因はさまざまです。

例えば、Webサービスがメディアなどで取り上げられたことで、短期間にアクセスが集中したため、サーバーがつながりにくい状態になったり、サーバーのメモリやハードディスクなどが何らかの原因で壊れるハードウェアの障害もあります。

また、地震や火事、水害などの災害、落雷による停電で一時的にシステムが使用できなくなることもあります。

サーバ障害の対策

サーバー側の障害には、複数サーバーを用意する対策が一般的です。障害が発生したサーバーをほかのサーバーに切り替えてサービスを継続する方法で、これを**冗長化**と言います。

冗長化については、障害が発生したときに備えるだけでなく、普段からアクセスを分散させておくことでアクセスの急増に対応する**負荷分散にもつながります**。

障害への対応

障害が発生した時には、それを最短で復旧させる必要があります。速やかに対応するには、障害の原因を調べることから始めます。このときに役立つのが**ログ**です。ログは**システムの利用状況などを記録したもの**で、たいていのシステムはログを出力できるようになっています。

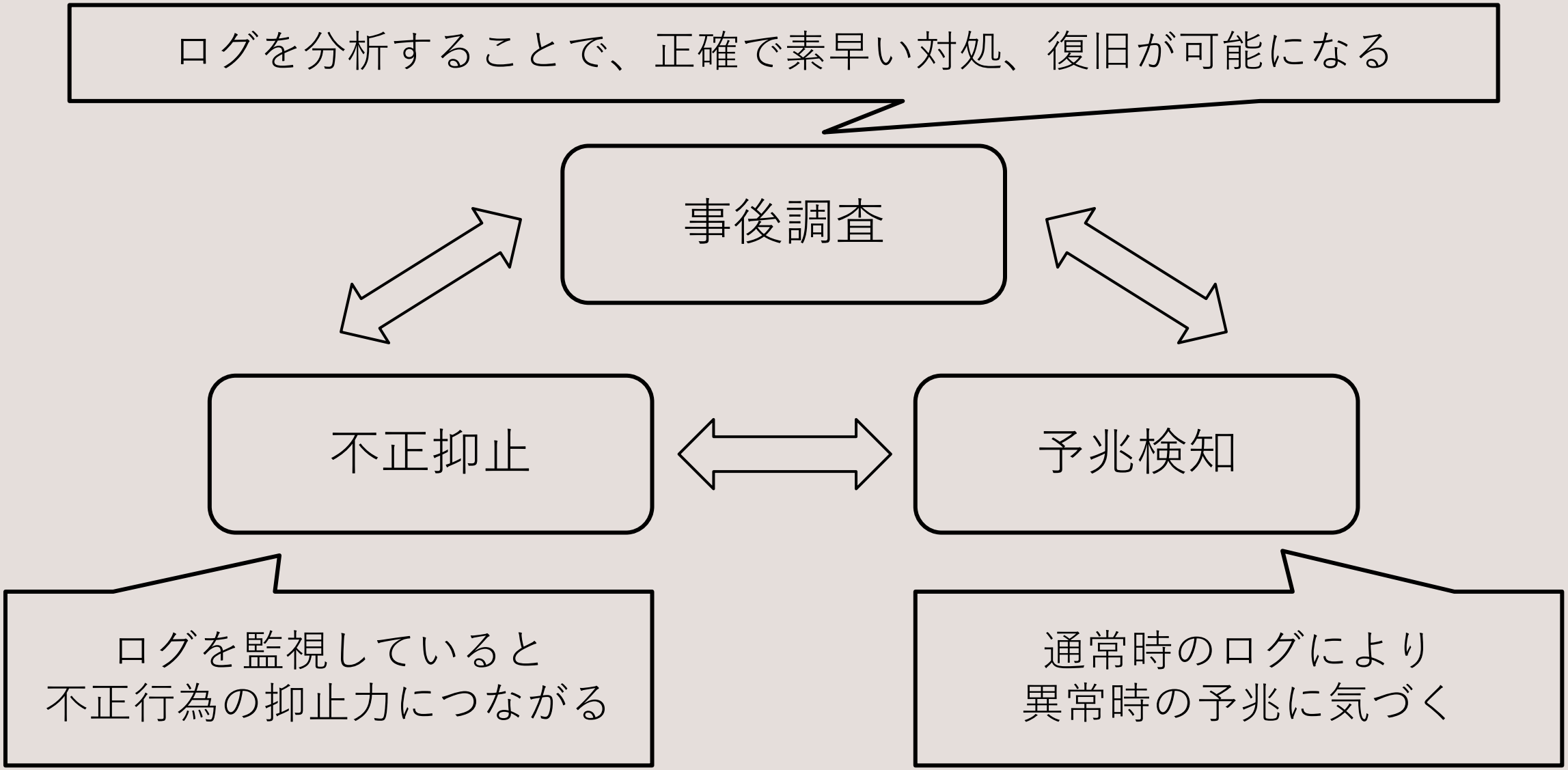
正常時の状態は把握したり、異常時の原因を追究したりするために使われます。

ログの役割

ログを正しく活用するためには、普段からログを確認しておかないと、そのログの内容が通常の状態なのか、異常な状態なのか判断できません。

「アクセス数が急増している」と判断するには、普段のアクセス数を把握しておく必要があります。また、普段からログを確認することで、攻撃の予兆を検知すれば、その通信を遮断するなどの対策を実施することができます。それがサーバーに対する攻撃などを予防することにもつながります。

ログの役割


ログを分析することで、正確で素早い対処、復旧が可能になる
事後調査
不正抑止
予兆検知
ログを監視していると
不正行為の抑止力につながる
通常時のログにより
異常時の予兆に気づく

復旧方法について

ログによって障害の原因が判明したら、復旧方法を検討します。

「アクセス集中」であれば負荷分散を実施などの対処があります。「メモリが不足」していれば、不要なプログラムを停止、再起動といった対応があげられます。「ハードディスクの容量不足」であれば、不要なファイルを削除したり、ディスクを追加したりといった対応があります。

監視

トラブルを未然に防ぐとともにトラブルの発生に速やかに対応するために必要なのが、ログなどの**監視**です。

監視の考え方は、大きく２つに分けれらます。

種類	目的	例	運用
異常監視	異常が発生した時に管理者に通知するため	・CPUの使用率が一定の割合を超えたら管理者にメール ・攻撃を検知したら運用ルームのランプを点灯させる	システムから連絡があるので受動的な対応
正常監視	異常につながる変化の兆しに気づくため	・CPUの負荷やメモリの使用量などをグラフにして傾向を把握	担当者が能動的に監視する必要あり

運用を考慮したシステム開発

自社で開発したシステムの場合、**監視や障害対応を考慮してログを出力できるようにしておく**必要があります。

役割としては発注者がログを見ることは無く、ベンダーの担当者が確認します。そのため、障害が発生した時に出力するログの内容を整理しておきましょう。

ログがわからないため、発注者に障害の原因を報告できないとお互い困ることになります。

保守契約

システム開発では開発費用を支払うだけでなく、**保守契約**を締結することが一般的です。開発したシステムに使われているフレームワークやライブラリに見つかった脆弱性の対応や、技術的な質問への対応、定期的なメンテナンス作業などが保守契約に含まれています。

保守費用は、システム開発費用の15%程度が年間保守費用の目安といわれています。５００万をかけてシステム開発した場合、７５万が年間にかかる保守費用という具合です。

保守契約

問題が起きなくても支払いが発生しますが、何かトラブルが発生した時に保守契約がないと対応できません。

問題が発生してから、対応する会社を探すと時間がかかります。
さらに対応費用が高額になる可能性があります。

改修案件（追加開発）

システムの開発が完了して運用を開始すると、利用者から機能追加の要望が出てきます。このような要望に対応するには、**追加開発**が必要です。

保守契約を締結していても、機能の追加はその範囲外であり、追加の費用が発生します。

一般的にはそのシステムを開発したベンダーに追加開発を依頼するとになります。

※信頼を失っていなければ・・・

改修案件（追加開発）

システム開発をするときは、今の開発だけを考えるのではなく、将来的な追加や修正、運用といった部分も検討しておくことが重要です。

また、開発体制にも注意が必要です。高いスキルを持った開発者が１人の場合、転職や契約期間の終了などで開発したシステムを保守できる人がいなくなる可能性があります。