Red Hat

クラウドネイティブ・アプリケーションへの道

8 つのステップ

Red Hat
Application Services

目次

Red Hat
Application Services

はじめに

アプリケーションは、大多数の組織が顧客、パートナー、従業員と交流するための手段となりました。このような新しいデジタルネイティブ機能の急速な台頭は、従来のビジネスモデルに破壊的な革新をもたらしており、従来型の企業や業界は、適応とモダナイズを余儀なくされています。

多くの組織にとって、革新的なデジタル体験を生み出すということは、アジャイルな組織文化への転換を意味します。このような環境において、目まぐるしいスピードで発生する要求を満たすには、より迅速で柔軟な開発およびデリバリーモデルが必須となります。しかし、ほとんどの組織では、テクノロジー基盤を完全に再構築したり、新しいプラクティスや考え方をすぐに取り入れたりする余裕はありません。そのため、文化、プロセスおよびテクノロジーを段階的に、かつ根本的に変化させ、スピードとアジリティに対応しようと試みています。この e ブックでは、アプリケーションへのクラウドネイティブ・アプローチの導入を検討する際に考慮すべき 8 つのステップをご紹介します。

「クラウドネイティブ開発の核心にあるのは、テクノロジーだけでなく、チーム、人、コラボレーションです...。コラボレーションは、反復的かつ柔軟な方法でアプリケーションを構築するための鍵であり、ステークホルダーと作成側の全員が、製品の作成、コード化、テスト、デプロイメントに貢献する必要があります」

— Red Hat リサーチレポート：クラウドネイティブ開発に関する Red Hat の見解、2021 年 6 月

ステップ 1：クラウドの文化とプラクティスを進化させる

クラウドネイティブ・アプリケーションを実現する上で、アプリケーションをより迅速かつ効率的に構築し、導入するために、開発、基幹業務および IT 業務チームにはさまざまな面での転換が必要です。業種や規模にかかわらず、クラウド・アプリケーションの展開を成功させるためには、連携や調整を必要とする多岐にわたる作業、技術、チーム、プロセスについて考慮する必要があります。パブリッククラウドやハイブリッドクラウドのリソースを利用する従来のアプローチは、チームが独立した意思決定を行い、迅速に行動することを可能にします。しかしこの戦略では、データや環境の各種プロセスが孤立しがちで、イノベーションが生まれにくくなります。

急速にイノベーションが進む時代にあって、複数の分散環境、高度にカスタマイズされたレガシー・アプリケーション、また新たなアプリケーション・ワークロードの管理における複雑さは、アプリケーション向けに統合クラウド戦略を策定する組織にとって大きな課題となっています。全社的なクラウド戦略がなければ、企業はアプリケーション・ポートフォリオの可能性を十分に引き出せないままになってしまいます。

コラボレーティブなクラウド文化の導入で重要なのは、新しいツールとテクノロジーの使用だけではありません。アプリケーションの開発と提供のための、より統合された連携アプローチを受け入れることに、皆が意欲と信頼を持つよう促すことが重要です。オープンソースソフトウェア・プロジェクトの文化は、アプリケーションの統合的で接続性に優れたクラウド戦略を構築するための指針となります。

ステップ 2：マイクロサービスを使用して既存アプリケーションを高速化する

アプリケーションのクラウドネイティブ化に向けた取り組みを開始するにあたって、組織は新規開発にのみ重点を置くべきではありません。ビジネスを運営し、収益を創出する上で、多くのレガシー・アプリケーションは必要不可欠であり、単純に置き換えることができないためです。これらの既存アプリケーションは、新たなクラウドネイティブ・アプリケーションと統合し、一緒に動作させる必要があります。では、既存のモノリスをどのように高速化できるのでしょうか。その答えは、既存のモノリシックなアーキテクチャを、モジュール式のマイクロサービスベースのアーキテクチャと、 API (アプリケーション・プログラミング・インタフェース) ベースの通信に移行することにあります。

モノリシックなアプリケーションをマイクロサービスにリファクタリングするという負担の大きい作業に取りかかる前に、組織はまず、マイクロサービス・アーキテクチャの安定した基盤を構築する必要があります。

マイクロサービス手法への移行は、すべてを一度に移行しようと急ぐことではありません。段階的なアプローチによって、無理のないペースで、既存のモノリスをより細かいコンポーネントに分割する作業から始めましょう。段階的なアプローチを使用することで、アプリケーションは、確かな設計方針と適切に定義されたドメイン境界に従って構築されることになります。このアプローチにより、必要に応じて、さらに段階的かつリスクの少ない方法でマイクロサービス・アーキテクチャに移行することもできます。また、マイクロサービス・アーキテクチャを成功させるための基盤も構築できます。

レガシー・プラットフォームへの依存度が高いアプリケーションでも、既存のモノリスをコンテナベースのプラットフォームに移行することで、アップデートやデプロイを高速化することができます。この移行により、導入と提供にかかる時間が短縮され、投資対効果 (ROI) が向上します。移行後にモノリスのインテグレーションや機能を構築する場合は、クラウドネイティブな技術や手法を利用できます。

ステップ 3：開発の迅速化に向けてアプリケーション・サービスを活用する

ソフトウェア開発の迅速化においては、再利用性は常に成功の鍵となってきました。クラウドネイティブ・アプリケーションの場合も例外ではありません。しかし、再利用可能なクラウドネイティブ・アプリケーションのコンポーネントが、クラウドが提供するスピードとスケーラビリティを最大限に引き出せるようにするには、基盤となるインフラストラクチャに合わせて最適化し、これに統合する必要があります。

キャッシュサービス、ルールまたはワークフローエンジン、インテグレーション・コネクター、モバイルおよび API 管理機能、データ仮想化サービス、メッセージングブローカー、サーバーレス・フレームワークを作り直す必要はありません。基盤となるコンテナベースのインフラストラクチャに統合され、最適化された既存のコンポーネントを使用することができるためです。SaaS (Software-as-a-Service) であっても、PaaS (Platform-as-a-Service)、iPaaS (integration Platform-as-a-Service) であっても、このアプリケーション・サービスはすぐに使える開発者用ツールです。

DevOps プロセスとコンテナ化はアプリケーションのデリバリーとデプロイメントを加速させますが、開発を迅速化し、新しいアプリケーションをより短期間で市場に投入できるようにするには、クラウドネイティブ・アプリケーションで、こうした種類のサービスを 1 つまたは複数利用することが必要になる場合があります。たとえば、開発者は、とくにコンテナベースのインフラストラクチャで効果的に動作するように構築されたアプリケーション・サービスを利用できます。これらのサービスは、継続的インテグレーション/継続的デリバリー (CI/CD) パイプライン、ローリングおよびブルーグリーン・デプロイメント、自動スケーラビリティ、フォールトトレランスなどのプラットフォーム機能を活用できるように設計されています。

ステップ 4：作業に適したツールを選択する

クラウドネイティブ・アプリケーションでは、選択する開発言語やフレームワークが、特定のビジネスアプリケーションのニーズに合わせて調整されることが多くなっています。この複雑性とアプリケーションの多様性を管理するには、クラウドネイティブな開発用のフレームワーク、言語、アーキテクチャの適切な組み合わせをサポートする、コンテナベースのアプリケーション・プラットフォームが必要です。

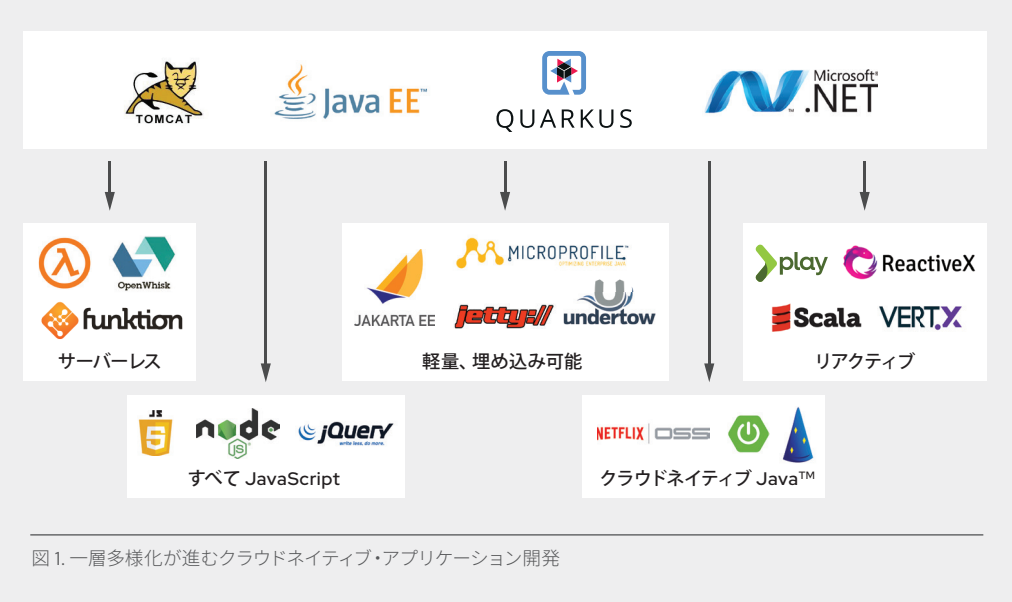

図 1. 一層多様化が進むクラウドネイティブ・アプリケーション開発

クラウドネイティブな開発では、作業に適したツールの選択も求められます。クラウドネイティブ・アプリケーションは、12 ファクター手法、ドメインベースの設計、テストベースの設計と開発、モノリスファーストまたは高速モノリス戦略、ミニサービス、またはマイクロサービスを使用して実装できます。どのようなアプローチを取るにせよ、組織のクラウドネイティブ・プラットフォームは、適切な開発要件をサポートするために、フレームワーク、言語、アーキテクチャを適切に組み合わせて提供できる必要があります。さらに、基盤となるコンテナベースのプラットフォームについては、テクノロジーの変化に合わせて継続的にアップデートされる、一連の厳選されたランタイムおよびフレームワークをサポートすることが条件になります。

ステップ 5：セルフサービス型オンデマンド・インフラストラクチャを実現する

アジャイル方式は、開発者によるソフトウェアの迅速な作成とアップデートを可能にしてきましたが、必要なときに必要なところでタイムリーなインフラストラクチャ・アクセスを可能にする効率的なメカニズムが欠けていることがよくあります。これにより、アプリケーションをプロダクション環境にリリースする際に、市場投入までのスピード全体に影響が出ます。インフラストラクチャが安価で、エンジニアの人件費が高額なこの時代にあって、チケットを申請してから IT 運用部門がリソースをリリースするまで何週間も待つというのは、もはや持続可能なモデルではありません。

セルフサービス型でオンデマンドのインフラストラクチャのプロビジョニングは、開発者が必要なときにインフラストラクチャにアクセスできるため、不正なシャドー IT に代わる強力なオプションとなります。しかし、このモデルが効果を発揮するのは、動的で複雑になりがちな環境全体を、IT 運用チームが制御し、可視化できる場合に限られます。

コンテナおよびコンテナ・オーケストレーション・テクノロジーは、基盤のインフラストラクチャへのアクセスを抽象化し、単純化して、データセンター、プライベートクラウド、パブリッククラウドなど、さまざまなインフラストラクチャ環境の枠を超えた堅牢なアプリケーション・ライフサイクル管理を実現します。さらに、コンテナ・プラットフォームは追加のセルフサービス、自動化、アプリケーション・ライフサイクル管理の機能を提供します。このモデルでは、開発者および運用チームが一貫性のある環境を迅速にスピンアップできるため、開発者はインフラストラクチャのプロビジョニングに関連する障害や遅延に手間取ることなく、アプリケーションの構築に集中できます。

また、コンテナによってアプリケーションの可搬性がもたらされ、どのクラウドプロバイダーにも導入し、実行できるクラウドネイティブ・アプリケーションの作成などが可能になります。可搬性により、いつでも、どのクラウドプロバイダーでも選択することができ、あるクラウドプロバイダーから別のプロバイダーへの移行が容易になります。また、関連コストを最適化でき、特定のクラウドプロバイダー API に合わせたコーディングを行わずにマルチクラウド・アプリケーションを開発できるようになります。

ステップ 6：IT 自動化によりアプリケーション提供を迅速化する

IT やインフラストラクチャの自動化は、手動による IT 作業を排除し、クラウドネイティブ・アプリケーションの提供を迅速化する上で必要不可欠です。自動化は、ネットワークとインフラストラクチャのプロビジョニングから、アプリケーションの導入と構成管理に至るまで、あらゆる作業またはコンポーネントに統合し、適用できます。

IT 管理ツールと自動化ツールが作成する反復可能なプロセス、ルール、フレームワークにより、市場投入時間の遅延につながる労力のかかる、人間による作業を置き換えるか、または低減することができます。さらに、これらのツールは、コンテナのようなテクノロジーや DevOps などの手法、クラウド・コンピューティング、 セキュリティ、テスト、監視や警告などの幅広い分野へと適用することができます。つまり、自動化は価値実現までの時間を短縮し、IT の最適化とデジタル・トランスフォーメーションを実現する上での鍵となります。

e ブック『組織を自動化する』で、IT 自動化が果たす重要な役割に関する詳細をご覧ください。

e ブックをダウンロードする

IT 自動化の指針

1. IT 運用の自動化へ向けて、プログラムによる組織規模の自動化手法を導入する上で、サービス要件の設定を目的とした、組織全体でのコラボレーションに基づく対話を取り入れる。
2. 自動化言語やプロセスの学習支援の基盤として、自動化サンドボックスを検討する。
3. 自動化について真剣に検討する。手動制御による安心感を残したい気持ちを抑えて、不要なマニュアル作業をすべて排除する。
4. 体系的な方法により、小さく達成可能なステップに分けた段階的な自動化の導入を検討する。着手した後は、各段階が次の段階への足がかりとなり、自動化の体制が組織に浸透していく。
5. まず、1 つのタスクやサービス (コンピュート、ネットワーク、ストレージ、プロビジョニングなど) の自動化から始める。次に、その自動化を他者と共有し、体系的な方法で発展させる。
6. セルフサービス・カタログ (ユーザーが必要とし、デリバリーを迅速化するもの) を導入する。
7. 計測、監視、チャージバックのポリシーとプロセスを導入する。

長期的には全面的な自動化が統合された形で実現するほか、優れた効率性や DevOps パイプラインとイノベーションのスピードアップをもたらします。

継続的デリバリー (CD) は、自動化を使用して新しいコードのリリースを迅速化するソフトウェア開発手法です。開発者によるアプリケーションへの変更を、自動化によってコードリポジトリやコンテナレジストリにプッシュできるプロセスを確立します。

ステップ 7：継続的デリバリーと高度なデプロイメント技術を導入する

リリースサイクルが長いということは、ソフトウェアのバグの検出から解決までに長い時間を要するだけでなく、顧客や市場のニーズに対するタイムリーな対応が本質的に困難であることを意味します。モバイル、Web、IoT (モノのインターネット)、 エッジコンピューティング・アプリケーションなど、トラフィック量の多いアプリケーションの場合、未解決のバグがあると多数のユーザーに影響が出ます。その結果、カスタマー・エクスペリエンスの低下、セキュリティや安全性の問題、生産性の低下や収益の減少が生じます。社内で利用するビジネス・アプリケーションであっても、アプリケーションの停止やソフトウェアのバグ対応の遅れにより、多大なビジネスコストが発生することがあります。

「高度なデプロイメント技術は、イノベーションに構造と明確性をもたらします。デプロイメント方法論が成熟することで、本当の意味での実験、フィードバック、分析を行える環境が生まれます。実験環境の改善が、優れたイノベーションにつながります」

Red Hat 開発者エクスペリエンス
ディレクター
Burr Sutter (バー・サッター)

アジャイル開発手法の発展は、「release early, release often (短いサイクルで頻繁にリリースする) 」モデルを生み出しました。DevOps と継続的デリバリーのアプローチにより、この手法の適用範囲がさらに拡大しました。開発、運用、品質保証、セキュリティ担当チームを密接に連動させることで、ソフトウェアの提供プロセスを改善できます。これにより、変更されたコードを短期間で確実にプロダクション環境にリリースでき、開発者に迅速にフィードバックを提供できます。この反復的かつ迅速なフィードバックループは、CI/CD によって可能となります。このループによってインフラストラクチャの自動化が、自動化されたテストや脆弱性スキャン、セキュリティ・コンプライアンス、規制チェックなどアプリケーション提供のあらゆる側面をカバーする、自動化されたエンド・ツー・エンドのデリバリー体制へと拡張します。デリバリーパイプラインの自動化の目的は、業務能力に影響を及ぼすことなくアップデートを提供し、デリバリーのリスクを低減することにあります。

継続的デリバリー (CD) を実現するための最初のステップは、継続的インテグレーション (CI) の実現です。CI システムは、さまざまなソース・コントロール・リポジトリの変更を監視し、適用可能なすべてのテストを実行し、ソースコントロールの変更に基づいてアプリケーションの最新バージョンを自動的に構築するビルドシステムです。

ソフトウェアリリースに伴うリスクを低減し、顧客に予期せぬ悪影響を与えずに制御された実験の環境を構築することを目標とする高度なデプロイメント・パターンを利用できます。この目標は、組織全体でイノベーションを増進する上で不可欠です。

高度なデプロイメント手法は、本質的にデリバリーのあり方を変えるものです。これまでは週末の営業時間外に行われるメンテナンス期間やダウンタイムを伴う作業であったものが、通常の営業時間内に行うルーティン作業となります。プロダクション環境にダウンタイムが一切生じないばかりか、作業中も顧客はアプリケーションを利用し続けることができます。

この手法により、新しいアプリケーションの導入に伴って顧客が経験する不便さを解消できるため、組織はビジネス上求められる頻度でアップデートやリリースを提供できます。以下に一般的なデプロイメント手法の例を挙げます。アプリケーションのユースケースによっては、ゼロダウンタイムの導入が可能です。

ローリング・デプロイメントは、1 つのアプリケーションのすべてのインスタンスを一度にアップデートするのではなく、各インスタンスをトラフィックを受信しないようにロードバランサーから除外し、個別にアップデートする導入形態です。インスタンスのアップデートが完了すると、再度ロードバランサーに追加されます。すべてのインスタンスがアップデートされるまで、このプロセスが続きます。

ブルーグリーン・デプロイメントは、2 つの同じ環境を、一方はアクティブに、もう一方はアイドル状態で運用するモデルを指します。変更内容はまずアイドル環境に展開され、プロダクション環境で変更が検証されると、ライブトラフィックを更新済みの環境に切り替えます。直前のバージョンへのロールバックは、単純にトラフィックを元の環境に戻すことで可能ですが、データの移行も考慮に入れられていることが条件となります。

カナリア・デプロイメントは、2 つの同じ環境を使用するという点ではブルーグリーン・デプロイメントに類似していますが、導入の制御方法が異なります。新しいリリースをデプロイした後、一部の小数ユーザーがプロダクション環境でリリースのテストを行います。新しいリリースの検証が成功すると、トラフィックを段階的に新バージョンへと移行します。この間、すべてのユーザーが新しいリリースに移行するまでは結果の監視と検証が続きます。

ステップ 8：モジュール式アーキテクチャを進化させる

ソフトウェア作成の手法の 1 つであるマイクロサービスベースのアーキテクチャでは、アプリケーションを互いに独立した最小コンポーネントに分割します。すべての機能を 1 カ所にまとめるという、従来のモノシリックな手法とは異なり、マイクロサービスは個々が独立したコンポーネントであり、これらのコンポーネントは同じタスクを達成するために連動します。このソフトウェア開発手法は、細分化でき、軽量で、同様のプロセスを複数のアプリ間で共有できるというメリットがあります。マイクロサービス・アーキテクチャは特定の基盤となるインフラストラクチャを要求するものではありませんが、コンテナベースのプラットフォームが最適な基盤となります。

極めて大規模なチームやプロダクション環境へのデプロイを 1 日に複数回行う組織には、マイクロサービスベースのアーキテクチャを展開することでさらなるメリットが得られる可能性があります。アーキテクチャの観点からは、マイクロサービスを利用するには各サービスを独自のデプロイメント単位に分ける必要があります。その後、各マイクロサービスは 個別に管理され、デプロイされ、それぞれのライフサイクルを担当するチームも異なる場合があります。

マイクロサービスに代わる選択肢として、ミニサービスがあります。ミニサービスとは、ドメイン単位で分割されたサービス群であり、通常アプリケーションサーバーで実行されます。ミニサービスは、マイクロサービスベースの設計やインフラストラクチャのような複雑さを回避しながら、アジリティと拡張を強化します。ミニサービスでも、アジャイル、DevOps、CI/CD 手法への投資が必要であり、先進的なアプリケーションサーバーや、複数のフレームワーク、アーキテクチャ、言語サポートをコンテナベースのインフラストラクチャと組み合わせて使用することが理想的です。

マイクロサービスやミニサービス、モノリスファーストを含め、クラウドネイティブ・アプリケーション開発に向けたさまざまフレームワーク、言語、アプローチをサポートするプラットフォームは、アプリケーションのクラウドネイティブ化を成功させるための鍵となります。

Red Hat でできること

組織のクラウドネイティブ化における現状や優先事項を問わず、Red Hat はお客様をサポートするテクノロジーとサービスを提供して、あらゆるステップでお客様をお手伝いします。

図 2. Red Hat を利用したクラウドネイティブ化

クラウドネイティブの 1 つのユースケースに集中的に取り組む組織もあれば、同時に複数のユースケースを優先させなければならない組織もあります。アプローチが漸進的であるか革新的であるかを問わず、クラウドネイティブ化への道はそれぞれ異なり、必ずしも直線的ではありません。道のりがどのようなものであれ、アプリケーションをより迅速に市場に投入するには、適切なテクノロジー、DevOps 体制、および文化が必要です。

Red Hat は、クラウドネイティブ・コンテナ開発プラットフォームおよびオーケストレーション・プラットフォームである Red Hat® OpenShift® を提供してお客様の取り組みをサポートします。Red Hat Application Services は、Red Hat OpenShift 上で動作するアプリケーション開発用の包括的なポートフォリオです。Red Hat Application Services ポートフォリオには、クラウドネイティブ・アプリケーションを開発し、デプロイし、スケーリングするためのフレームワーク、ツールおよびソリューションが含まれています。市場参入の迅速化を目指している組織には、Red Hat OpenShift Application Services (Red Hat Cloud Servicesの一部) の利用をお勧めします。これは、クラウドネイティブ・アプリケーションの開発、デプロイ、スケーリングの最適化された開発者体験を提供する Red Hat OpenShift のホスト型およびマネージド型のクラウドサービスです。

Red Hat コンサルティングは、複雑なクラウドネイティブ開発の道のりを迷うことなく進めるよう、戦略的なアドバイスや深い専門知識でサポートします。Red Hat のコンサルタントが、Red Hat Open Innovation Labs からディスカバリーセッション、そしてプロジェクト実施計画に至るまで、クラウドネイティブ化の取り組みにおけるあらゆるステップでお手伝いします。

Red Hat について

エンタープライズ・オープンソース・ソフトウェア・ソリューションのプロバイダーとして世界をリードする Red Hat は、コミュニティとの協業により高い信頼性と性能を備える Linux、ハイブリッドクラウド、コンテナ、および Kubernetes テクノロジーを提供しています。Red Hat は、クラウドネイティブ・アプリケーションの開発、既存および新規 IT アプリケーションの統合、複雑な環境の自動化および運用管理を支援します。受賞歴のあるサポート、トレーニング、コンサルティングサービスを提供する Red Hat は、フォーチュン 500 企業に信頼されるアドバイザーであり、オープンな技術革新によるメリットをあらゆる業界に提供します。Red Hat は企業、パートナー、およびコミュニティのグローバルネットワークの中核として、企業の成長と変革を支え、デジタル化が進む将来に備える支援を提供しています。

アジア太平洋
+65 6490 4200
apac@redhat.com

オーストラリア
1 800 733 428

インド
+91 22 3987 8888

インドネシア
001 803 440 224

日本
03 4590 7472

韓国
080 708 0880

マレーシア
1 800 812 678

ニュージーランド
0800 450 503

シンガポール
800 448 1430

中国
800 810 2100

香港
800 901 222

台湾
0800 666 052

fb.com/RedHatJapan
twitter.com/RedHatJapan
linkedin.com/company/red-hat

jp.redhat.com
#F30307_1020