データ統合パターン

Red Hat インテグレーションドメインアーキテクト **Monica Hockelberg (モニカ・ホッケルバーグ)**
Red Hat スペシャリストソリューションアーキテクト **Gbenga Taylor (グベンガ・テイラー)**

Red Hat

目次

背景

データ統合パターンを使用することにより、組織内のエンタープライズデータを、統一された、正確かつ一貫性のある方法で使用できるようになります。通常、こうしたデータは多種多様なものがさまざまな場所に存在しており、形式 (バイナリ、JSON、XML、レガシーシステム、RDBMS、NoSQL など)[1] も混在しています。

データ統合の目標を達成するために使用されるアプローチは、主に、さまざまなデータセットを取り巻くサービスの品質 (QoS) と使用特性によって決まります。データ統合戦略は、エンタープライズ・データソースのさまざまなセットを論理的に (場合によっては物理的にも) 組み合わせて、組織に必要なデータサービスを公開するのに役立ちます。

データ統合パターンを理解し、それらを効果的に使用することによって、効果的なデータ統合戦略を構築できます。次の各セクションでは、これらのパターンについて詳しく説明します。

パターン1：データ集約

- 抽出、変換、ロード (ETL)
- ELT またはデータレイク

このパターンは、複数のさまざまなソースシステムからデータを抽出し、別個に用意した単一の永続的保管場所に格納できるよう処理します。ETL プロセスは 3 つのステップで構成されています。

1. さまざまなデータソースからデータを抽出します。
2. ストレージと使用要件に合わせてデータを変換します。
3. 変換されたデータをターゲットシステム (データウェアハウス、運用データストア (ODS) など) にロードします。

注：ELT またはデータレイクの場合は、上記のステップ 2 と 3 が逆になります。

データ集約では、データ統合プロセスの設計と実装を行い、強化された完全なデータをデータストアに供給します。分析の全段階として大量のデータを処理する必要がある場合に適しており、全体的なデータ品質 (完全性、一貫性、正確性など) を向上させます。

このアプローチを採用すると、データの再構築、調整プロセス、綿密なクレンジング、およびアグリゲーションとさらなる強化を目的とした追加のステップを実行できるようになります。データ集約は、すべてのデータ変更が経時的にキャプチャされ、変化するディメンションとして保持されるため、ほぼリアルタイムのデータや履歴データを必要とするビジネス要件をサポートできます。

データ集約によってある程度のデータレイテンシーが生じ、データがわずかに古くなるため、データの一貫性が重要な意味を持つディメンションには最適と言えない場合があります。データ集約には物理的なストアが必要なため、全体的な戦略を立てる際には、ストレージ容量の追加も視野に入れた計画が必要です。

1 JavaScript Object Notation (JSON)、Extensible Markup Language (XML)、リレーショナル・データベース管理システム (RDBMS)

データ集約：ETL

もともと、ETL はサーバーの処理能力に伴う課題に対処し、生データをターゲットデータストアの使用可能なデータに変換する負荷を軽減するためのものでした。しかし時間の経過とともにサーバーテクノロジーが発達し、この方法で変換処理をオフロードする必要はなくなりました。現在ではむしろ、変換プロセスそのものが障害になる可能性があります。特に、この処理のためだけに大量のデータを移行し、基本的に他にはメリット (ターゲットサーバーの負荷軽減など) がないクラウドデータウェアハウスの場合はそうです。

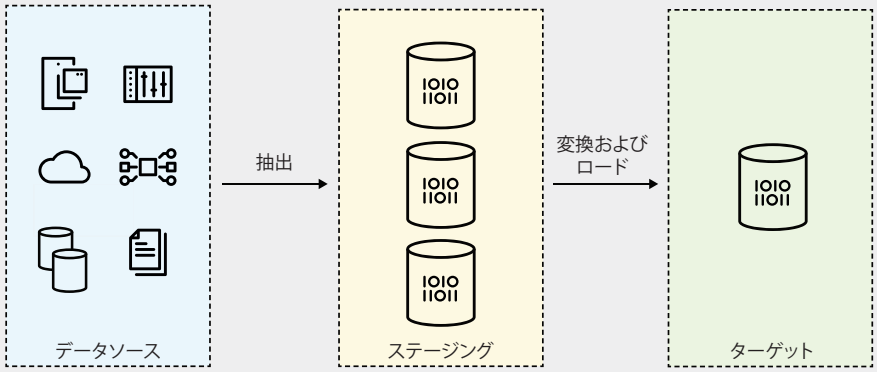

図 1. 抽出、変換、ロード (ETL)

データ集約：ELT またはデータレイク

ELT ではデータの変換を宛先で行うため、この変換による障害は解消されます。このアプローチはきわめてスケーラブルで、必要な量のデータを生のフォーマットで格納し、迅速に宛先へと転送します。このアプローチにより、宛先に送る前にデータの変換処理を行うことに特化したインフラストラクチャを立ち上げる必要がなくなります。

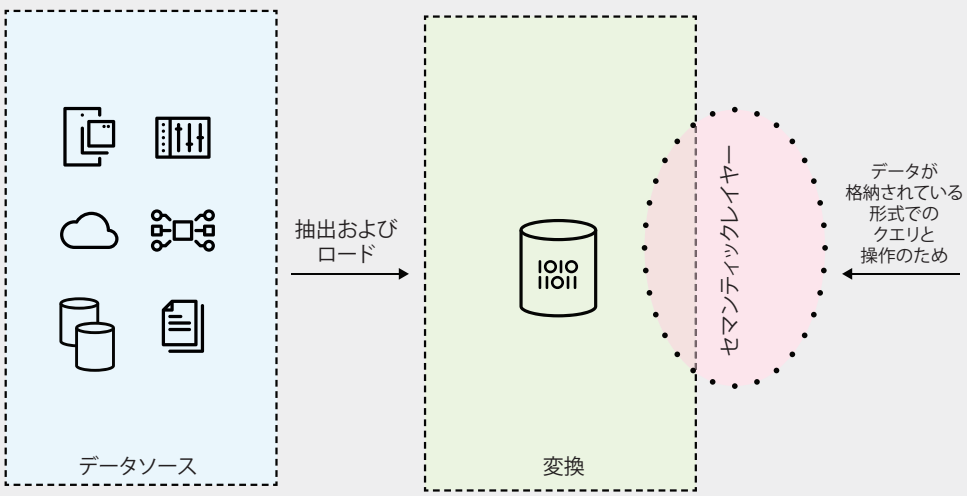

図 2. ELT またはデータレイク

パターン 2：データフェデレーション

- コンポジットサービス
- データ仮想化またはエンタープライズ情報統合 (EII)

データフェデレーションでは、基礎となるソースシステムからオンデマンドでデータを取得するプル型のアプローチを使用します。基礎となるソースデータはそのままで、仮想的にデータを処理して表示できるようにします。ソースシステムデータのレプリケーションや移行は必要ありません。

このパターンでは、データへのリアルタイムアクセスが可能です。基礎となるソースデータがそのままなので、追加のストレージ要件はありません。このパターンにおいて考慮すべき重要な事項は、パフォーマンスです。フェデレートされたデータビューに対してクエリを実行すると、最終的なデータ表示のために必要な変換処理によるオーバーヘッドが発生するからです。

データフェデレーションの実装形態には、複数のデータソースからデータを返すコンポジットサービスから、すべてのエンタープライズデータ向けの完全な仮想化データアクセスレイヤーまで、大きな幅があります。

データフェデレーション：コンポジットサービス

コンポジットサービスは、アグリゲーターパターンを実装します。さまざまな異なるサービスからのデータを有意義な方法で組み合わせ、消費するアプリケーションにこの応答を提供します。

このパターンは、サービス指向アーキテクチャ (SOA) とマイクロサービス・アーキテクチャの両方で使用されます。これら 2 つのアーキテクチャスタイルの主な違いは次のとおりです。

1. サービスインタフェースの粒度。通常は、SOA サービスのほうが粗くなります。
2. マイクロサービス・アーキテクチャでは、各マイクロサービスは独立しており、個別にデプロイできます。

マイクロサービスはソースシステムのデータストアでの依存関係を維持しません。マイクロサービスでは処理に使用するデータコピーを用意し、それをランタイムで使用します。マイクロサービスのレプリカデータストアには、結果整合性のあるデータが格納されています。このデータは、ソースシステムのデータストアが収容するデータについて、ある程度の一貫性を備えています。

多くの組織が成功しているモダナイゼーション・アプローチの 1 つが、レガシーアプリケーションの API 化です。このアプローチでは、先進的なアプリケーション・プログラミング・インタフェース (API) (通常は REpresentational State Transfer (RESTful)) を既存のレガシーアプリケーションまたはサービスの前に配置して、既存のレガシーアプリケーションまたはサービスへの影響を最小限に抑えながら、モバイル、Web、およびモノのインターネット (IoT) アプリケーションに必要な、先進的なインタフェースを提供します。

API 管理ソリューションは、実行する必要のある革新的な作業の実行と、組織が環境の内外に公開しているデータに関する制御の取得および適用を可能にするため、モダナイゼーションの取り組みにシームレスに適合します。API 管理ソリューションを使用するとセキュリティ、レート制限、分析収集、およびその他の機能を実装する構成可能なランタイム実施ポリシーを使用できるので、API ポートフォリオのより効果的な管理が可能になります。

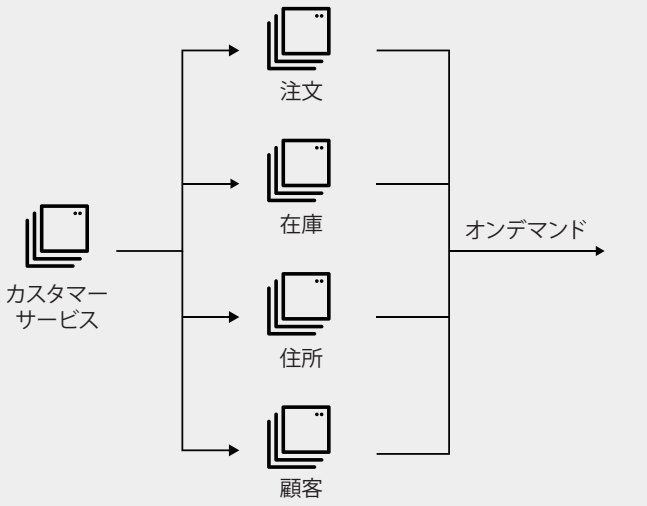

図 3. コンポジットサービス

データフェデレーション：データ仮想化またはエンタープライズ情報統合 (EII)

EII (Enterprise Information Integration) の目的は、多様なデータソースの大規模なセットを複数組み合わせて、単一の統一されたデータソースとしてデータコンシューマーから利用できるようにすることです。

EII は、組織全体のデータと情報のビューを統一する機能であるため、しばしばデータ仮想化と互換的に使用されます。データ仮想化は、データ抽象化を使用してエンタープライズデータへの共通データアクセスレイヤーを提供する実装であり、エンタープライズが承認した命名規則を使用する統一されたデータ構造を公開します。

データ仮想化のもう 1 つの主な機能は、安全なデータゲートウェイとして機能し、きめ細かなアクセス制御を適用することです。データ仮想化のセキュリティモデルは、行および列データを処理し、必要に応じて個々のデータ要素のフィルタリング、難読化、サニタイズを行うことができます。

データ仮想化の主なメリットの 1 つは、部門に関係なくすべてのデータを 1 つの場所に配置するので、データ使用者が容易にデータにアクセスできることです。この仮想化されたデータはライブで提供され、コピーではなく、読み取りと書き込みが可能です。クラウドプラットフォーム、オンプレミスでホストされているデータ、エンタープライズシステム、およびビッグデータ処理システムで機能するエンタープライズ環境では、このアプローチにより、これらの情報をすべて組み合わせて、より容易に使用できるようになります。

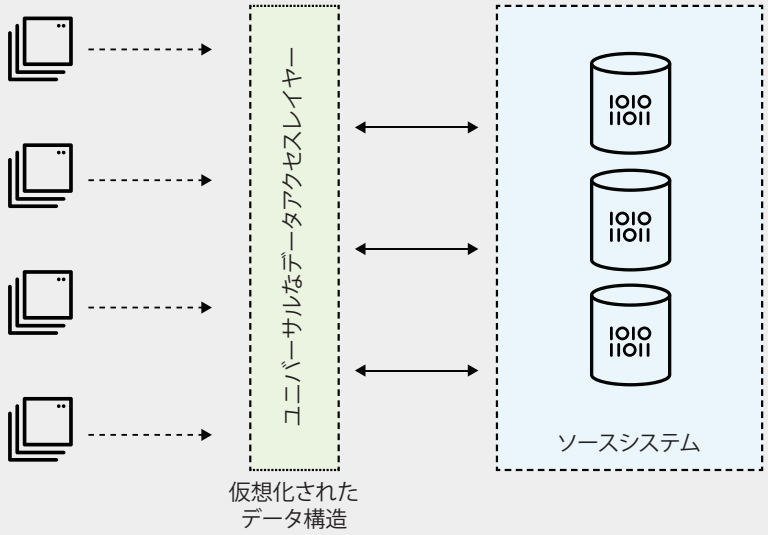

図 4. データ仮想化またはエンタープライズ情報統合 (EII)

パターン 3：データプロパゲーション

- エンタープライズ・アプリケーション統合 (EAI)
- エンタープライズ・データ・レプリケーション (EDR)

データプロパゲーションでは、データ更新 (例：ソースシステムから 1 つ以上のターゲットシステムにイベントをパブリッシュする) の同期または非同期のプロモーションを行います。このパターンは、アプリケーションレベルとデータストアレベルの両方で適用できます。

アプリケーションレベルでは、ソースアプリケーションのイベントが 1 つ以上のターゲット・アプリケーションの処理をトリガーします。このタイプのイベントには、ソースアプリケーションからターゲット・アプリケーションにデータを伝播するための小さなメッセージペイロードがあります。

データストアレベルでは、ソースシステムのイベントがソースデータストアの更新をトリガーします。これらの変更イベントは、ほぼリアルタイムで 1 つ以上のターゲットデータストアにレプリケートされます。

データプロパゲーション：エンタープライズ・アプリケーション統合 (EAI)

従来型の EAI では、アプリケーション間の統合を一元的に管理でき、多くの場合、既存のレガシー・エンタープライズ・サービス・バス (ESB) インフラストラクチャを使用します。

先進的な EAI は分散型かつ軽量で、弾力性に優れた運用環境に合わせて拡張可能です。統合自体は、コンテナ化アプリケーションとしてデプロイできます。

選択するアプローチに関係なく、ビジネスロジックと統合ロジックを分離し、独立して構成可能な (コードではない) 運用ポリシーを維持することが重要です。ランタイムポリシーの適用と統合ロジックは、そのタイプのトラフィックが発生する環境内の場所にのみデプロイすることを推奨します。

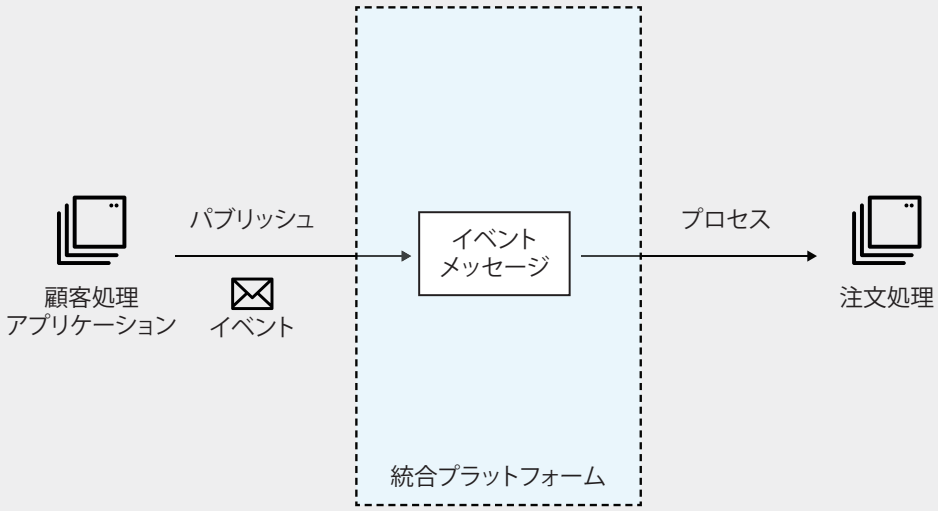

図 5. エンタープライズ・アプリケーション統合 (EAI)

データプロパゲーション：エンタープライズ・データ・レプリケーション (EDR)

分散型アーキテクチャやマイクロサービス・アーキテクチャでは、レプリケーションにより、アプリケーションの信頼性を高めることができます。分離や独立性のレベルは固有のものなので、アップストリームのシステムで障害が発生しても、必ずしもダウンストリームの分散されたサービスに影響を与えません。さらに、この設計によりアプリケーションやサービス・ポートフォリオのパフォーマンスが向上する場合もあります。

サービスが必要とするデータはレプリケートされ、同じ場所に配置され、特定のサービスによって、より使いやすい方法で格納されます。その後、サービスはより効率的に運用され (オーバーヘッドとレイテンシーが低減)、意図されたリクエストに対応できるようになります。これは、サービスがアップストリームのデータストアまたはシステムで発生しているアクティビティに緊密に縛られていないためです。

オンプレミスシステム、プライベートクラウド、パブリッククラウドを横断する場合もある現代の多様な運用環境では、分散型アプリケーションを、最終的には整合性を備えることとなる独自のデータストアとともにデプロイすることで、柔軟なデプロイと弾力性に優れたスケーリングが可能となります。これにより、運用環境内で発生するワークロードのピークへも、発生した場所で即時に対応しやすくなります。

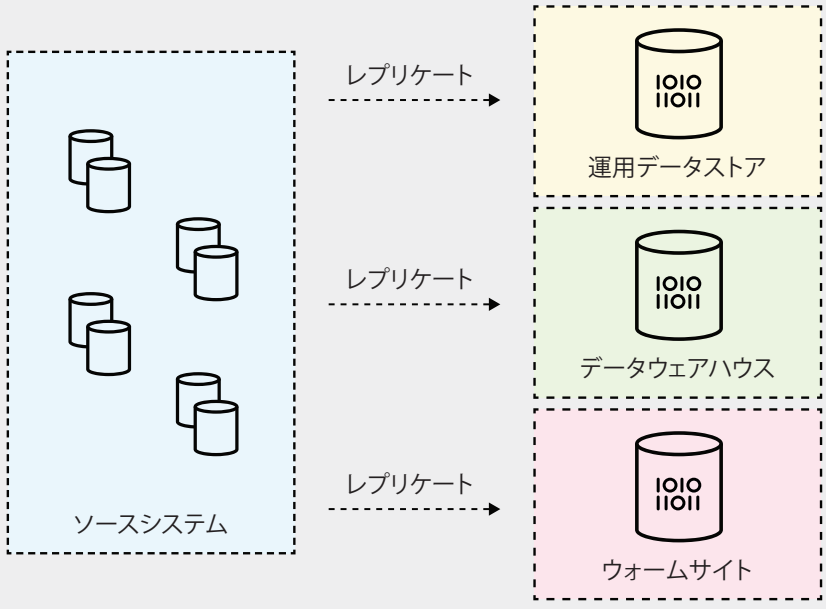

図 6. エンタープライズ・データ・レプリケーション (EDR)

補完的なパターンおよびプラクティス

- 変更データキャプチャ (CDC)
- イベントソーシング
- ストリーミングデータとイベントストリーム処理 (ESP)
- 従来型のメッセージング
- 分散キャッシングとインメモリデータグリッド

変更データキャプチャ (CDC)

変更データキャプチャは、ソースデータストア内のデータ変更イベントを検出し、別のデータストアまたはシステムで更新プロセスをトリガーします。CDC は通常、トリガーベースまたはログベースで実装されます。トリガーベースのアプローチでは、トランザクションイベントは別のシャドウテーブルにログされます。このシャドウテーブルは、リプレイしてイベントを定期的にターゲットシステムにコピーできます。ログベースの CDC (トランザクションログ・テーリングとも呼ばれる) は、トランザクションログをスキャンして、データ変更イベントを識別します。このアプローチは、多くのデータ変更シナリオに適用でき、必要なオーバーヘッドが最小限で済むため、トランザクション量が非常に多いシステムをサポートできることから、よく使用されます。

イベントソーシング

イベントソーシングは、アプリケーションの状態に対するすべての変更がイベントのシーケンスとして確実に格納されるようにするパターンです。これらのイベントはテンポラルクエリで使用できるため、過去の状態の再構築やアクティビティのリプレイが可能になります。このパターンは、監査ログの作成、デバッグ、特定の時点での状態の再構築が必要なユースケースに役立ちます。

ストリーミングデータとイベントストリーム処理 (ESP)

ESP は、継続的にデータを作成するシステムから生じる一連のデータポイントでアクションを実行します。ここでは、「イベント」はシステム内のデータポイント、「ストリーム」はそれらのイベントの継続的な提供を指します。この一連のイベントは、ストリーミングデータとも呼ばれます。これらのイベントの結果として実行されるアクションのタイプには、アグリゲーション、分析、変換、補完、および別のデータストアへの取り込みがあります。

イベントストリーミング・プラットフォームには、主に 3 つの機能があります。

1. 特定のイベントストリームのパブリッシュとサブスクライブ。メッセージキューを使用する従来型のメッセージングで行う方法と同様です。
2. イベントストリームの、フォールトトレラントで永続的な方法による格納。
3. イベントストリームのリアルタイム処理。たとえば、Apache Kafka は広く使用されているイベントストリーミング・テクノロジーであり、このタイプの機能を提供します。

ストリーミングデータのアプローチは、ビッグデータ、モバイルアプリケーション、プライベートクラウド、パブリッククラウド、およびマイクロサービス・アーキテクチャを採用するエンタープライズの増加に伴い、今日ますます普及しているクラスタ化されたコンピューティング環境での実行に非常に適しています。

従来型のメッセージング

従来型のメッセージングでは、信頼できるメッセージング特性を実装するという意味において、メッセージコンシューマーは極めて単純です。メッセージブローカーは、コンシューマーによって確認されたメッセージを追跡し、メッセージ順序の処理を確実に行い、メッセージの見落としや見逃しがないようにします。従来型のメッセージブローカーは、さまざまなプロトコルを使用するコンポーネントのブリッジとして使用できる複数のプロトコル (AMQP、MQTT、STOMP など) [2] を実装しています。また、メッセージの存続時間 (TTL) や有効期限、非永続的なメッセージング、要求と応答のメッセージング、相関 ID セレクターなどの追加機能も提供します。これらは、Kafka のようなストリーミング・プラットフォームには適さないユースケースです。

分散キャッシングとインメモリデータグリッド

キャッシングとは、将来の要求をより迅速に処理するためのシステムでデータを保持するためのストレージ容量を用意することです。キャッシュには、頻繁にアクセスされるデータや、別のデータストアに格納されているデータを複製したコピーを含むデータが格納されます。キャッシングの最大の目的は、パフォーマンスの向上です。

テクノロジーの急速な進化と、コスト効率に優れたデータストレージの普及により、分散キャッシング・アーキテクチャが一般的になりつつあります。分散キャッシュは複数のノードにデプロイされ、手頃な価格のメモリとより高速なネットワークカードを備えた低コストのハードウェアを使用して水平方向に拡張できます。

インメモリデータグリッドは、いくつかの特徴的な追加機能を提供する分散キャッシング・ソリューションです。まず 1 つ目の機能は、分散キャッシュデータをデータの存在する場所で計算処理する能力です。これは、計算処理能力をデータの存在場所に配置することで実現されます。もう 1 つの特徴的な機能は、標準 SQL および MapReduce をベースとする分散型大規模並列処理 (MPP) のサポートです。これにより、多くのノード間でインメモリで格納されたデータ全体を効率的に計算できます。そして、インメモリデータグリッドはクロスサイト・レプリケーションをサポートし、さまざまなデータセンターやプライベートクラウドまたはパブリッククラウドでデータを利用できるようにします。

分散キャッシングはデータの高可用性へのニーズの高まりを受けて開発されました。一方、インメモリデータグリッドは、データ処理の複雑化傾向により幅広く対処するものです。先進的なデータ処理ワークロードには、キーベースのアクセスではもはや十分ではありません。データ処理には、分散 SQL、複雑なインデクシング、大規模なインメモリデータセット全体での MapReduce 処理などのツールが必要になっています。

このようにキャッシング・アーキテクチャが進化を続け、データ管理機能のシンプルさよりもハイブリッドデータやコンピューティングを管理できる機能が重視されるようになっていく中、組織はこの変化に対応するため分散キャッシングをどのように活用すべきなのか、再考する必要があります。ハイブリッドデータ管理の目的は、あらゆるタイプのデータ (構造化、非構造化、およびその間のものすべて) を、データがどこに存在するか (オンプレミス、パブリッククラウドまたはプライベートクラウド、あるいはこれらのデプロイメントの組み合わせなど) にかかわらず、簡単にアクセスおよび分析できるようにサポートすることです。

---

2 Advanced Message Queuing Protocol (AMQP)、MQ Telemetry Transport (MQTT)、Simple Text Oriented Messaging Protocol (STOMP)

Red Hat ソリューション

Red Hat は、Red Hat® Integration を通じてさまざまなコンポーネントを提供します。これらは、これまで述べてきたパターンの実装に活用することができます。

Red Hat Fuse

Red Hat Fuse は、スタンドアローン方式またはクラウドでデプロイ可能な、分散型のクラウドネイティブ統合プラットフォームです。統合に関するエキスパート、アプリケーション開発者、およびビジネスユーザーは、それぞれが選んだ環境で個別にコネクテッド・ソリューションを開発できます。Red Hat Fuse は、レガシーシステムから API、マイクロサービス、パートナーネットワーク、IoT デバイスに至るまであらゆるものを接続し、エンタープライズのエッジにアジャイル・インテグレーションのメリットをもたらす、一元化されたオープンソース・ソリューションを提供します。オープンソースとオープンスタンダードに基づいて構築された Red Hat Fuse には、Apache Camel が含まれます。これは、最も一般的に使用されているエンタープライズ統合パターン (200 以上のコネクタとデータ形式を網羅) の実装であり、レガシーシステムから SaaS (Software-as-a-Service) ソリューション、データベース、デバイスに至るまで、ほぼあらゆるものに統合可能です。

データ仮想化

Red Hat Integration にはデータ仮想化機能が含まれます。この機能は、オープンソースの Teiid プロジェクトをベースとした無駄のない仮想データ統合ソリューションであり、従来は活用しにくいかたちで存在していたデータの価値を引き出し、利用しやすい、一元化された、次のアクションへつながる情報として提供します。物理的に異なる、または統合されていないシステムやデータソース上に分散したデータを簡単に組み合わせ、単一のソースとして扱えるようにすることにより、適切なデータを必要な形式かつ適切なタイミングで、任意のアプリケーションまたはユーザーに配信できます。

Red Hat AMQ

Red Hat AMQ は柔軟なメッセージング・プラットフォームで、情報を確実に提供し、さまざまなユースケースのリアルタイム統合を実現します。

AMQ ブローカー

Red Hat AMQ のコンポーネントである AMQ ブローカーは、Apache ActiveMQ Artemis をベースとする、オープンソースで高性能な従来型のメッセージブローカーです。これは、JMS、AMQP、MQTT、STOMP、およびさまざまな言語間クライアントなどの複数の基準をサポートしています。

AMQ Streams

Red Hat AMQ のコンポーネントである AMQ Streams は、サポートされた Apache Kafka のオープンソース・ディストリビューションです。このディストリビューションは、Red Hat OpenShift® などのクラウドネイティブな Kubernetes プラットフォームでの Apache Kafka のデプロイ、構成、管理を単純化します。AMQ Streams はイベントストリーミングのバックボーンを提供し、高スループットと低レイテンシーでのデータ交換を可能にします。また、ストリーミングデータとイベントストリーム処理のバックボーンになることができます。

Debezium

Debezium は Red Hat Integration のもう 1 つの重要なコンポーネントであり、ログベースの変更データキャプチャを実装します。データベース・トランザクション・ログの変更を監視し、これらの変更を、他の外部システムで使用される Apache Kafka トピックへのイベントとして外部化できます。一般的なリレーショナル・データベースと NoSQL データベースのサポートを提供します。

Red Hat 3scale API Management

Red Hat 3scale API Management は、拡張性の高い API 管理プラットフォームで、API プログラムの作成およびデプロイを目指す組織に制御性、可視性、柔軟性を提供します。包括的なセキュリティ、収益化、レート制限、分析、および成功する API 戦略を構築するために必要な開発者コミュニティ機能を提供します。

Red Hat 3scale API Management は、Red Hat Fuse またはその他のあらゆるソースによって作成された HTTP ベースの API サービスを管理できます。API プロバイダーは、さまざまな認証プロトコルと承認プロトコルを使用したセキュリティ構成、提供した API の使用状況分析、トラフィック制御とアクセスポリシー適用の構成、カスタマイズ可能な開発者ポータルを介した API ユーザーのコミュニティ作成、および API サービスの収益化ポリシーの設定を行えます。Red Hat 3scale API Management の構成タスクは、管理コンソールのユーザーインタフェース (UI) 経由で実行することも、DevOps パイプラインの一部として自動化することもできます。

Red Hat Data Grid

Red Hat Data Grid は、高性能な分散キャッシュであり、従来型のクラウドネイティブ環境にデプロイできるインメモリデータグリッドでもあります。アプリケーションの応答時間が短縮され、開発者はアプリケーションのパフォーマンスを劇的に向上させると同時に、弾力性に優れたスケーリングに加え、可用性と信頼性を実現できます。アプリケーションと既存のデータアクセスレイヤーおよびソースの間のシームレスなキャッシュとして機能し、オプションでニアキャッシュを使用して構成することができます。つまり、クライアント・アプリケーションは、最近使用したデータを格納するローカルキャッシュを保持できます。Red Hat Data Grid は、プラットフォームに依存しないアクセスを可能にし、複数の言語で開発されたアプリケーションをサポートします。また、MapReduce と似た分散ストリーム実行機能を持ち、データセンター間のレプリケーションを可能にします。

まとめ

膨大な量の新しいデータが毎日作成され、エンタープライズの運用環境は、クラウド、コンテナ、弾力性、分散など、複雑さを増しています。このような状況においては、既存のデータ統合戦略を見直し、戦略が組織の現在のビジネス目標と戦略的方向性に沿ったものであるかどうかを確認することが重要です。実際、データ統合の包括的なパターンはそれほど変わっていない一方で、これらのパターンを適用するために用いられる実装やテクノロジーは、先進的な分散型アーキテクチャとその課題に対処するために進化しています。適切なアプローチは、システムの品質や特性に応じて異なる場合があるため、データ統合戦略に取り組む組織は、複数のパターンを使用することも珍しくありません。

今すぐ始める

まずは、次の質問に答えることから始めましょう。これらは、組織のデータ統合戦略の作成と再設計に役立つ重要な質問です。

- まず、データ統合を必要とするビジネス上の理由は何ですか？
  - ビジネスケースについて検討し、投資対効果 (ROI) を計算しましょう。
  - 解決しようとしているのはどのような問題ですか？
  - 何をもって成功としますか？それをどのように測定しますか？

- 組織内の誰が関与しますか?
  - 既存および将来的なデータ統合の課題に対処するために必要なスキルを持つ人は、組織内にいますか?
  - 価値提供を強化し、データ統合の取り組みを組織全体へと拡張するために、どのような方法をとりますか?
  - このプログラムを主導することになるのは誰ですか?
- 統合の対象とするのはどのようなデータですか?
  - データの棚卸しをしましょう。データはどのような形式で、どこにありますか?
  - データの分類法は確立されていますか?
  - 扱っているデータは機密データですか?(制限付き、プライベート、パブリックなど)
  - 組織では将来、新しいタイプのデータを扱う必要がありますか?
- データパイプラインのどの時点でデータを統合する必要がありますか?
  - データウェアハウスに到達する前の時点、生の形式 (データレイクなど) で格納しても問題ない、あるいはその両方ですか?
- データを統合する必要がある場所はオンプレミスとクラウドのどちら、あるいはその両方ですか?
- 最後に、データ統合をどのように行う予定ですか?
  - データ統合に関する目標を達成するには、どのようなポリシー、手順、ツールが必要ですか?
  - データの管理、品質保証、保護はどのように行いますか?
  - どのようなテクノロジーを使用することになりますか?
  - プロジェクトを始動するために、エキスパートの支援が必要ですか?

アプリケーションおよびデータ統合について詳しくは、こちらのデータシートをご覧になり、Red Hat のコンサルタントまでお問い合わせください。

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、新規および既存 IT アプリケーションの統合、クラウドネイティブ・アプリケーションの開発、Red Hat が提供する業界トップレベルのオペレーティングシステムへの標準化、複雑な環境の自動化、セキュリティ保護、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、Fortune 500 企業に信頼されるアドバイザーです。クラウドプロバイダー、システムインテグレーター、アプリケーションベンダー、お客様、オープンソース・コミュニティの戦略的パートナーとして、Red Hat はデジタル化が進む将来に備える企業を支援します。

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
0120 266 086
03 5798 8510

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
#0000000_0020