Workday のテクノロジ プラットフォームと開発プロセスについて

Workday のテクノロジ プラットフォームと開発プロセスについて

エグゼクティブ サマリー

簡単に使えるのはもちろん、その使いやすさを変わらずにずっと運用・維持できること。エンタープライズ アプリケーションに求められる一番の条件です。Workday は、この基本方針を命題として、エンタープライズ アプリケーションの設計に取り組んでまいりました。

新しいエンタープライズ アプリケーションの開発とは、従来のプロセスを進化させるのではなく、既成の概念を超えた革新を果たすことでした。新たなエンタープライズ アプリケーションには、複数のレベルで、これまでにない斬新な発想が求められたのです。

- デリバリー モデル
- テクノロジ プラットフォーム
- 開発プロセス
- アプリケーション設計

Workday は、この 4 つの分野の全てに新風を巻き起こすエンタープライズ ソフトウェア企業として創業しました。エンタープライズ アプリケーションの最適なデリバリー モデルとして、SaaS を採用し、財務アプリケーションと人事アプリケーションに革新的な設計を施しました。

本ホワイトペーパーでは、Workday のテクノロジ プラットフォームとその開発プロセスに焦点を当てます。現代における全ての働く人々にとって、エンタープライズ アプリケーションに求められるものは何か。この問いにお応えしながら、何故、エンタープライズ アプリケーション プラットフォームに新しいアーキテクチャのアプローチが必要なのか、そして、何故、この分野の従来のアプリケーション開発プロセスを変えざるを得ないのかについてご説明します。その上で、Workday の製品づくりの根幹となるテクノロジ プラットフォームの構成要素と開発プロセスについてご説明します。

新たに求められるもの

Workday は、従来では考えられなかったほど簡単に、長期間にわたって運用・維持できるコア エンタープライズ アプリケーションを実現させたいとの発想から誕生しました。

もっと簡単に、使いやすく

これまで、エンタープライズ アプリケーションは、企業内で幅広く活用されているとは言えませんでした。間接部門の専門スタッフが使い方のトレーニングを受けて、データの更新や処理対応をしていました。残念ながら、他の社員は積極的にシステムを活用するまでには至らず、もっぱら、トレーニングを受けた専門スタッフに任せきりで、彼らが作成したレポートやスプレッドシートに頼っていました。

Workday は、タスクの遂行やレポート作成を IT 部門に任せるのではなく、企業の全社員がエンタープライズ アプリケーションを使いこなすようになるべきだと考えています。社員やマネージャのためのセルフサービス機能は、特別なトレーニングを必要とするべきではなく、アドオン型のモジュールとして提供されるものでもありません。Workday では、ほとんどの社員の方が日常的に使い慣れているコンシューマー向けインターネット アプリケーションのユーザー エクスペリエンス (UX) を、エンタープライズ アプリケーションの評価基準としています。

モバイル対応は、アドオン型のプラットフォームやモジュール、あるいは個別のアプリケーションとしてではなく、ユーザー エクスペリエンス (UX) に本来備わっているものでなければなりません。さらには、モバイル機器の使われ方に合わせて、UX も絶えず改善していかねばなりません。

図 1

トランザクションだけに着目したエンタープライズ アプリケーションでは、一般にビジネス ユーザーの期待に応えることはできないでしょう。ビジネス ユーザーが求めるのは、現在のトランザクション データを、前後の文脈を踏まえて分析し、その分析に基づいて即座に行動に移すことに他なりません。従来のシステムは、トランザクションの取りまとめを目的として構築されていたため、分析については、異なるセキュリティで管理された別システムで行わざるを得ず、ユーザーは分析結果に基づいて行動を起こすことができませんでした。

トランザクション、分析、行動という 3 つの要素の分断を解消するには、単一のデータを使用し、単一のセキュリティ モデルで、これら 3 つを一元的にサポートできるシステムが必要不可欠だと Workday は考えます。

運用・維持が、より自在に、ずっと便利に

日々刻々と変化を続けるビジネスの動向に合わせて、エンタープライズ アプリケーションもまた、変わっていく必要があります。しかし、ビジネスは進化しているというのに、エンタープライズ アプリケーションはその進化に追いついていません。これまでのシステム アーキテクチャには、アプリケーションのカスタマイズ用ツールが付属していました。このため、運用中のシステムの実装に変更を加えるのは、お客様の役割でした。残念ながら、昨今では、アプリケーションの変更は次第に難しさを増し、コストもかかるようになっています。大幅なカスタマイズを行った結果、システムが複雑化し、最新バージョンへのアップデートを放棄しなければならないケースも散見されます。お客様は、新しいイノベーションやベスト プラクティスを活かすことができず、市場のペースから取り残されることになります。

Workday は、大きなコストがかかるアップグレードを必要としないで、いつでもベンダーが提供する新機能が導入できるようにするべきだと考えています。お客様が、高いコストをかけてカスタマイズを再実装することなく、稼働停止期間もなしに新しいシステムに移行するためには、ベンダーが新しいリリースやアップデートへのデータ移行を担当する必要があります。毎日、もしくは毎週といった頻度で新機能をリリースするような継続的開発モデルをベンダーが採用している場合、アップデートに関するベンダーの責任は特に重大になります。

Workday は、システムの整合性を損なうことなく、お客様の手によってアプリケーションのカスタマイズが可能な新しいモデルを作り出しました。従来のカスタマイズでは、データ モデルの変更と、何百行にも及ぶコードのプログラミングがつきものでした。しかし、Workday は、ガイド付きインターフェイスだけで、アプリケーションの設定を可能にしたのです。このモデルでは、基盤となるデータ モデルとアプリケーション コードを変更することがないため、新しいリリースが提供されても、お客様は機能を継続的にお使いいただくことができます。

エンタープライズ アプリケーションのインフラストラクチャの管理には、高度な作業が必要となります。これらの作業には、セキュリティ、ハードウェアとソフトウェアの拡張、災害復旧、パフォーマンス管理、システム管理、ネットワークなどの分野に精通した専門家チームが不可欠となります。こうした専門チームの選定、育成、管理は、本来、エンタープライズ アプリケーションを導入するお客様が行うべきものではないはずです。

せっかくエンタープライズ アプリケーションを導入しても、他のアプリケーションとのインテグレーション構築や、その管理をするために専門チーム維持することが、総所有コストの大部分を占めるのでは意味がありません。インテグレーション ツールをアプリケーションの一部として、柔軟性の高いアプリケーション プログラミング インターフェイス (API) を提供することにより、コスト効率が向上すると Workday は考えます。インテグレーション プラットフォームが、その基盤となるアプリケーション プラットフォームにはじめから組み込まれていれば、お客様はインテグレーションを利用しやすくなり、アプリケーションとの緊密性もさらに高まることになります。

新たなアプローチの必要性

新たな要件を満たすために、エンタープライズ アプリケーションにおけるアーキテクチャ開発プロセスについての考え方を、抜本的に変える必要があります。

これまでのアーキテクチャの概要について、簡単にご説明します。従来のエンタープライズ アプリケーションは、「リレーショナル クライアントサーバー」という構築的なアプローチを採用していました。これは、データベース層においてアプリケーションの構造を表すためにリレーショナル モデルを作成するものです。次に、ビジネスロジック層でデータベースのデータを保存、取得するためのコードを記述します。プレゼンテーション層 (通常はインターネット ブラウザ) では、データとトランザクション ページを表示するためのコードを記述します。

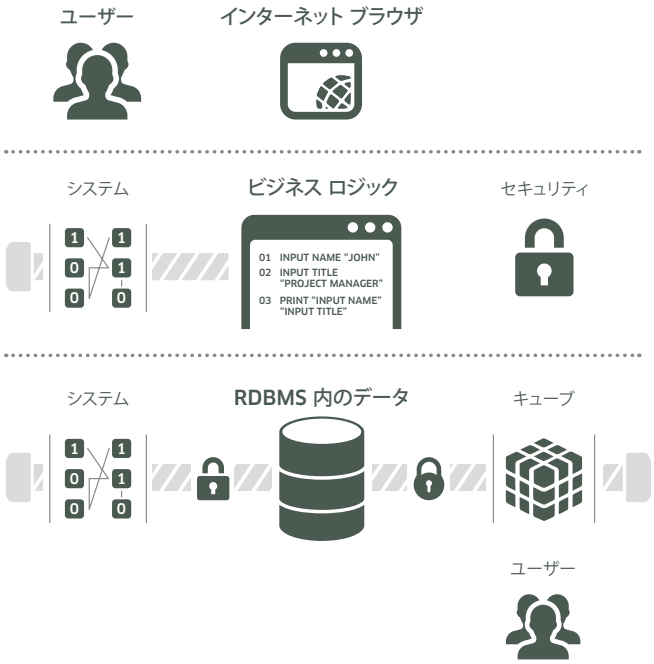

図 2

このタイプの従来型アーキテクチャでは、トランザクションを確実かつ広範囲に把握でき、トランザクションの履歴レポートを作成することができるとされました。

しかしながら、Workday は設立当初、この構築的なアプローチに対し次の 3 つの懸念を抱いていました。

1. **複雑性**: 複雑なアプリケーションには、テーブルを備えた複雑なデータ スキーマや、動作を表す大量のコードが必要になります。通常、エンタープライズ アプリケーションは、何千ものテーブルのデータを操作する、何百万行にも及ぶコードによって構成されます。このため、アプリケーションに大幅な変更を加える場合、これらのコードやテーブルについての膨大な変更や、調整が必要となります。

2. **インテグレーション**: 他のシステムとのインテグレーションを行うには、トランザクション データ (通常はファイル形式) をエクスポートしてからインポートするか、もしくは、ビジネス ロジック レベルで API を操作します。こうすることで、データはリレーショナル データベースから簡単に取得できますが、アプリケーションに組み込まれたセキュリティとビジネス ロジック レベルをそのまま迂回することになります。API は通常、後付けである場合が多く、また、別のシステムではなくユーザーとの対話を想定したビジネス ロジックとやり取りをしなければならないため、API を使用したアプリケーションとのやり取りは、複雑になることがあります。

3. **ビジネス インテリジェンス (BI)**: 従来型のアーキテクチャにも、詳細なトランザクション レポートの作成機能があります。しかしながら、ビジネス ユーザーの求める形でレポートが作成されるわけではありません。従来のアーキテクチャで開発されたアプリケーションでは、そのほとんどが、別の BI ツールにデータを転送してビジネス ユーザーの要件に合ったレポートと分析を提供します。データをコピーした場合、コピーへのアクセスの安全性を確保したり、コピーとアプリケーション内の元のデータとの整合性を考慮する必要が生じます。

Workday は、アーキテクチャと同様に従来型の開発プロセスについても懸念を抱いていました。従来型のプロセスは、数年に一度の頻度で大規模なリリースをサポートすることを前提に設計されていました。しかも、その開発は、現在稼働しているプログラムとは別のコードラインで行われ、既存顧客を新規リリースに切り換えるプログラムは、通常はリリースが構築された後に作成されます。さらに、アプリケーションのインフラストラクチャは、(オンプレミス型の導入の場合) お客様が管理するか、開発チームとは別のインフラストラクチャ チームが管理するものでした。

これらの懸念を踏まえ、Workday は開発と運用のプラットフォームに、これまでの考え方をまったく変えた新しいアーキテクチャによるアプローチを採用することに決めました。Workday が目指すのは、頻繁なアップデートと持続的な変更が可能な開発プロセスです。

新しいプラットフォームと開発プロセス

Workday は、従来のリレーショナル クライアント サーバー アーキテクチャが抱える様々な問題を解決するために、アーキテクチャのすべてのレベルにおいて、新たなアプローチの採用に踏み切りました。それが、SaaS デプロイメント モデルの完全サポートです。

Workday に課せられた命題は、エンタープライズ アプリケーションに優れたユーザー エクスペリエンスをもたらすこと。そして、当社のプラットフォームとアーキテクチャによって実現されるすべてのアクティビティの中心にあるのが、Workday のアプリケーションです。

次の図 3 は、Workday のプラットフォーム アーキテクチャの概要を示したものです。

Workday のアプリケーション

Workday の基盤は、Java によって構築されています。また、メタデータ抽象化層を設けることにより、開発をシンプルにしました。この抽象化層で使用する言語が XpressO です。アプリケーション開発においては、この XpressO を使用することで、実装に関連した細かな問題を意識する必要がなくなり、固有のテクノロジに依存することがなくなりました。これによって、Workday のアーキテクチャの進化は加速しました。

当社のアプリケーション開発陣は、その機能面に専念でき、永続化、トランザクション、スケーラビリティ、データセンターの位置といった、技術や実装における細目に思いを煩わせることもありません。卓越したユーザー エクスペリエンスを備えた実用的なアプリケーションの開発に集中できるようにすることこそ、Workday が目指すものです。

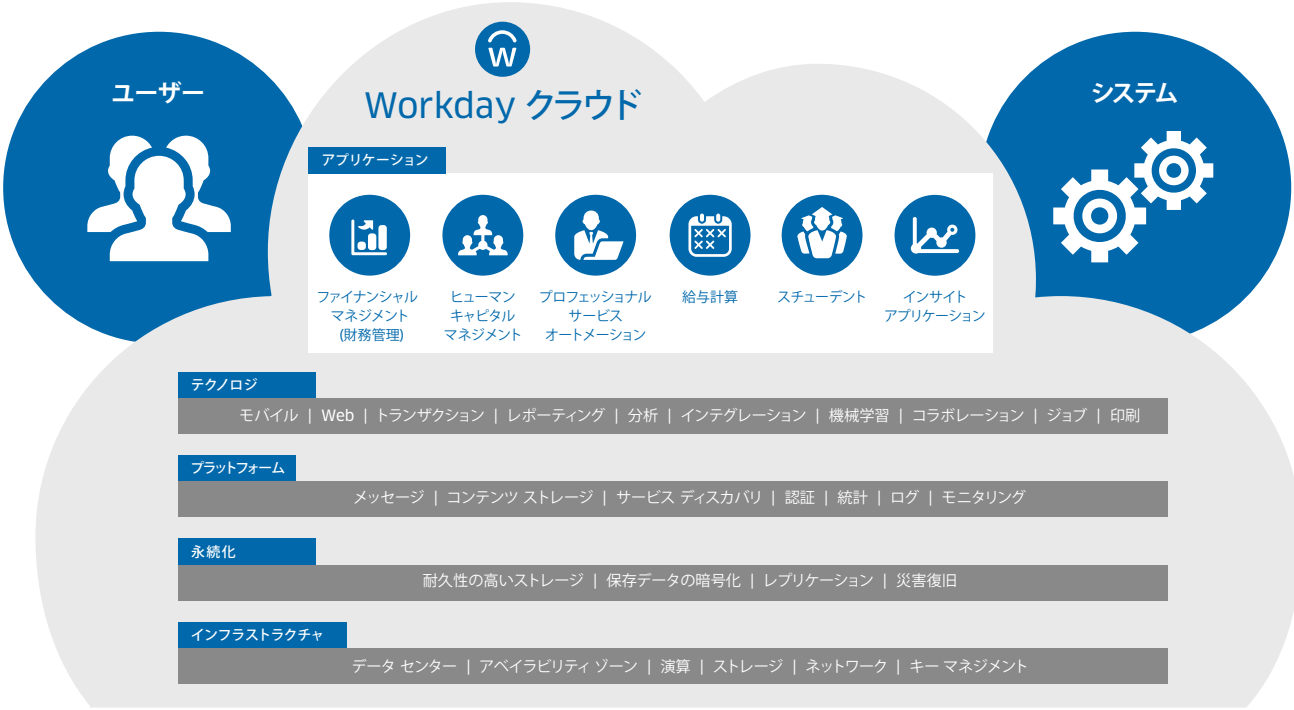

図 3

図 4

Workday のアーキテクチャとプラットフォームでは様々なサービスが連携し、アプリケーションのメタデータを解釈します。このメタデータは、XpressO がアプリケーションのトランザクションや、リクエストを処理するときに生成されるものです。アプリケーションの定義を実装プラットフォームの詳細と明確に分離することで、新しいアプリケーションを速やかにオンラインで稼動させることができます。これは、Workday が創業当初から立証してきたことです。

アプリケーションの作成と保守を容易にするために、前述の懸念事項を解決することを主眼とした設計理念を核として、Workday は、アーキテクチャを構築しています。この設計理念が革新的なプラス効果をもたらし、先進のアプリケーションを提供し、絶え間のない進化を生み出す能力につながっています。

重要な設計理念の 1 つが、データ ストレージに対する Workday のアプローチです。当社は、トランザクション データの保存にはリレーショナル データベースを使用しますが、アプリケーションのコア部分のモデル化についてはリレーショナル データベースを使用しません。アプリケーションのすべてのトランザクションは MySQL データベースに保存されますが、そのデータベースのスキーマにはアプリケーションとの類似性はまったくありません (たとえば、顧客テーブルや従業員テーブルは存在しません)。そのスキーマは、アプリケーション データとメタデータを保存するために最適化された、少数のテーブルのみで構成されます。

最大限のセキュリティを確保するために、このデータベースへのアクセスはすべて Workday のアプリケーション サービスを経由します。つまり、お客様企業の社員 (IT 部門も含む) はデータベースに直接アクセスすることができません。最適化したスキーマを使用してトランザクション データを保存しているため、新機能のリリース時や、毎週のアップグレード時も、お客様がこのスキーマの変更に頭を悩ますこともありません。

もう 1 つの、特筆すべき重要な設計理念は、アプリケーションの構造と動作の記述にメタデータ オブジェクト モデルを採用していることです。ここにおいて、XpressO が重要な役割を果たします。Workday アプリケーションには、コードベースの手続き型ロジックが存在しません。代わりに当社の開発陣は、鍵となるビジネス オブジェクトに対してクラスを定義することでアプリケーションの構造を定義しました。クラスには、他のクラス、属性、メソッドとの関係を設定できます。メソッドがアプリケーションの動作を定義するため、ビジネス ロジックは手続き型のコードで一切記述されることなく、宣言型の関係式によって定義されます。

オブジェクト モデルのパーツ (クラス、関係、属性、メソッド) はすべて、フォームベースのタスクを使って作成します。この方法で作成したアプリケーションは、アプリケーション オブジェクト モデルの各パーツのメタデータ定義の集合体となります。これらのメタデータ定義がシンプルな Java オブジェクトの集合体として、Java 仮想マシン (VM) のメモリに保存されます。Java VM は、Workday のすべてのアプリケーションのランタイムとなり、この Java ランタイムがメタデータ定義を、アプリケーションを構成するトランザクション (タスクやレポート) に解釈します。

Workday は、この 2 つの設計理念に基づき、コア アプリケーション データと、アプリケーションのすべてのメタデータ定義を、インメモリに保持することにしました。データをインメモリに保持することで、Workday のレポートの大半はデータベースにアクセスすることなく、作成することができます。また、インメモリ データは各クラスの関係を充実したネットワークで表現するオブジェクトモデルで構築されているため、Workday のレポートでは、単にデータを表示するだけでなく、マルチディメンション分析も可能になります。

この機能の簡単な例として挙げられるのが、社員数レポートです。ある企業の社員数を表示するには、アプリケーションで全社員の全インスタンスを収集、合計します。その結果、全社員のオブジェクトが収集され、社員オブジェクトに関連するデータのすべてのディメンションを精査できるインメモリ キューブが生成されます。

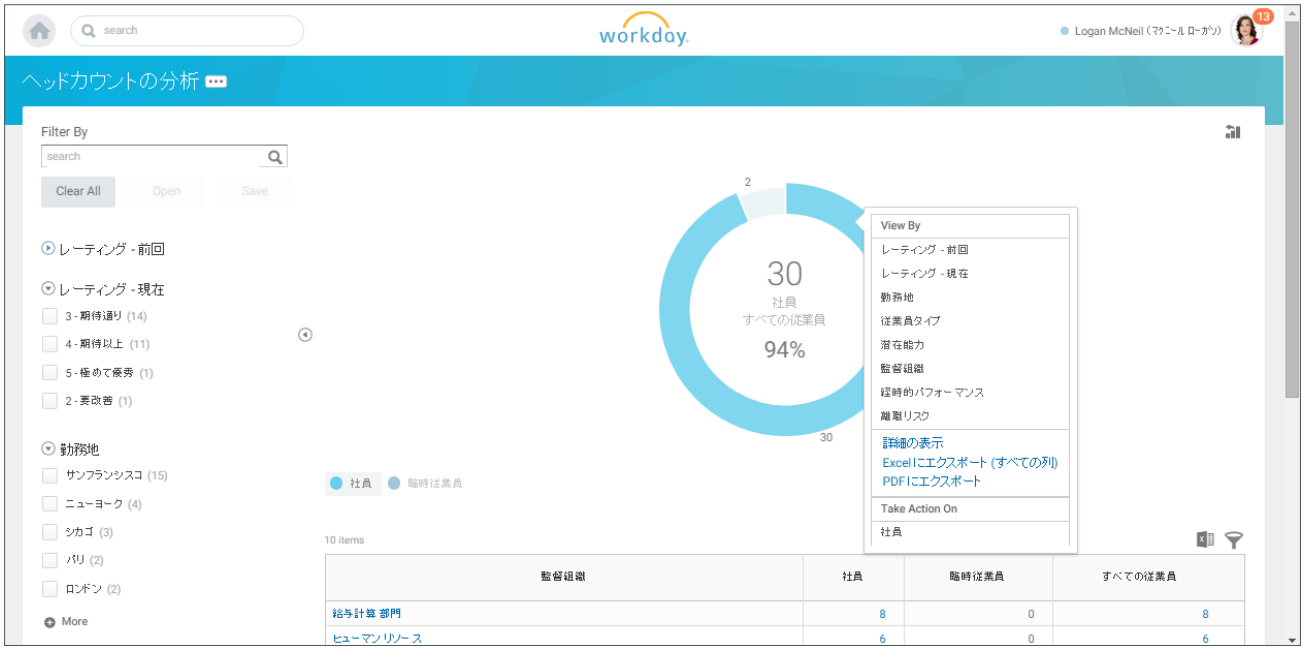

図 5

アプリケーションをこのように実行することで、多くのメリットが得られます。

- **開発者の生産性**: リレーショナル クライアント サーバー アプリケーションでは、何百万行にも及ぶコードによって、何千個ものリレーショナル テーブルを操作します。Workday が採用する定義型アプローチでは、相互参照される何百万ものメタデータ定義でアプリケーションが作成されます。膨大な数のメタデータ定義を扱うためシンプルとは言い難いですが、この方法でアプリケーションを編成することで、開発者は継続的にアプリケーションの拡張やリファクタリングを実行できることになります。お客様に実装した後でも、アップグレードに支障をきたすことなく、変更が可能です。
- **ユーザーとシステムからの要求に対応する単一のトランザクション処理モデル**: Workday アプリケーションが受け取るすべての要求は、同じ方法で処理されます。要求は、認証済みの有効なユーザー ID からに限定されます。すべての要求は、処理を待機する同一のキューに送られ、同一方法によって処理され、監査されます。「バッチ」処理もしくは、オンライン処理だからといって、処理のアプローチや言語を変える必要はありません。また、データの読み書きのためにリレーショナル データベースに直接アクセスすることもありません。
- **すべてのデータ アクセスに対応する単一セキュリティ モデル**: Workday アプリケーションでトランザクション形式のデータ アクセスを行う場合は、すべてにおいてロール ベースのセキュリティが使用されます。複雑なシステムの多くでは、データにアクセスする際のセキュリティを、ツールやアプリケーション毎に異なる方法で確保しています。特にカスタム レポートや分析ツールからアクセスする場合に、この違いが顕著に現れます。しかし、Workday アプリケーションでは、カスタムレポート作成ツールを使用してお客様が作成するすべてのレポートのデータにも、ロール ベースのセキュリティが詳細なレベルにおいて適用されます。同じロール ベースのセキュリティは、モバイル アクセス、分析ツール、ダッシュボード、インサイト サービス、当社 Web サービス API 経由のアクセス、ビジネス プロセス フレームワークで処理されるワークフローにも適用されます。
- **アプリケーションの機能設定**: Workday のアプリケーションでは、機能の設定と拡張を簡単に実現できます。お客様は、ビジネス プロセスやカスタム レポートなど独自のメタデータを作成することができ、Workday の既存のオブジェクトに独自のカスタム フィールドを追加することができます。アップグレードに支障をきたすことなくアプリケーションのコード ベースを拡張するのは難しいため、この機能設定はコーディングを使用せずに行われます。機能設定の結果は、テナント毎に異なるアプリケーション メタデータになりますが、リリースが変わっても、その処理方法やアップグレード方法は Workday のランタイムが把握しています。製品のアップデートのたびに作業のやり直しや修正の必要がないため、Workday のお客様は安心して機能設定をすることができます。

テクノロジ

テクノロジ サービスには、多年にわたって、幾多の革新や技術進化が施されてきました。各テナントからのアプリケーションへの要求は、様々なサービスの集合体によって処理されます。各サービスは、安全かつスケーラブルな方法で、最適なパフォーマンスを保証しつつ、要求されたトランザクションを実行します。こうした機能は、単一の中央処理サービスから、高度に連携する分散サービスの集合体へと進化してきました。

テクノロジ サービスは、トランザクション サービス、データ管理サービス、およびコンピューティング サービスという 3 つの主要なサービスに分類できます。

タスクやレポートなどユーザー操作によってトリガされる要求の処理を主に担当するのが、トランザクション サービスです。トランザクション サービスは、テナントにおけるデータのインメモリ表現を利用して、要求に必要なロジックを実行します。各テナントには「更新トランザクション サービス」(オブジェクト トランザクション サービス (OTS)) が 1 つ存在し、情報をアップデートするすべてのトランザクションを一括管理します。

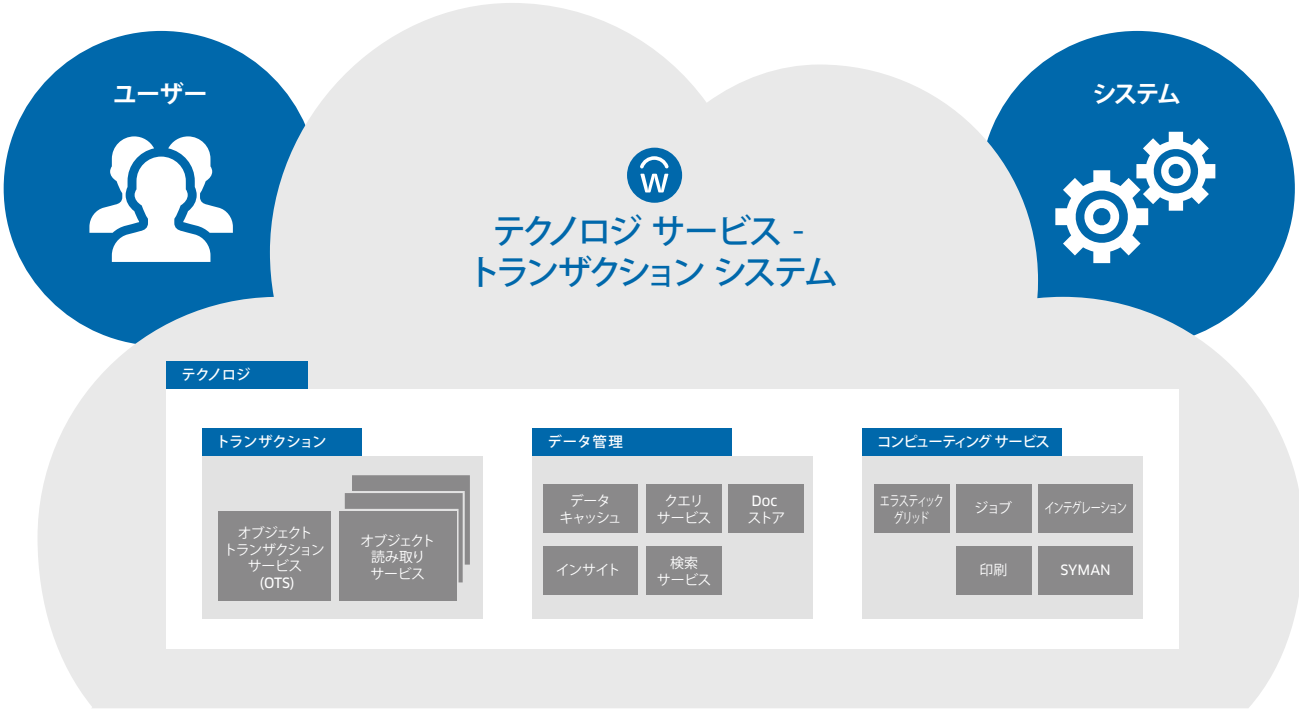

図 6

タスクの要求を受けると、その要求はオブジェクト モデル内でアップデートの処理方法を把握しているクラスに送られます。次に、アップデートしたデータを検証するためのメソッドが呼び出されます。検証をパスすると、そのアップデートがインメモリ データに対して実行され、永続化サービスを使ってデータベースに記録されます。アプリケーションがデータベースアップデートのコミット通知を受け取ると、新しくなったインメモリ データの値は、すべてのアクティブなテナント コンピューティング サービスのトランザクションから参照できるようになります。

データベースは、常にすべてのアプリケーション データにとっての「システム オブ レコード」です。データのインメモリ コピーが利用され、データベースがレポートやクエリに使われることが滅多にないとしても、このことは変わりません。トランザクション サービスとコンピューティング サービスのデータの同期と管理は、データ管理サービスが実行します。

データ管理サービスは、Workday におけるスケーラビリティ向上への取り組みの中核となります。データ管理サービスの主要コンポーネントの 1 つが「データ キャッシュ」です。データ キャッシュは、Workday アプリケーションを構成するすべてのコア アプリケーション データとメタデータを圧縮したコピー（「インスタンス キャッシュ」）を保持します。前述のレポートや計算のようなコンピューティング サービスの 1 つが起動すると、このインスタンス キャッシュを使用してお客様のデータのスナップショットを即座に作成します。このインスタンス キャッシュを使用して、お客様の更新トランザクション サービスや他のコンピューティング サービスのメモリ内の特定の時点のデータを再構築します。これにより、使用頻度の少ないインスタンスにアクセスする場合、ディスク アクセス負荷が負担にならないよう、更新トランザクション サービスのインスタンスのうち「メイン」メモリに保持すべきインスタンスを選択して、インスタンス キャッシュを作成できるようになります。

ただし、すべての処理が、お客様の更新トランザクション サービスで行われるわけではありません。レポート、検索、印刷、インテグレーション ロジック、大量の計算（給与計算など）といったの要求は、別のハードウェア上の独立した Java VM で実行されている専用サービスで処理して、適切なスケーラビリティが確保できるようにしています。Workday のコンピューティング

サービスでは、このようなプロセスを利用して、処理時間がかかる要求や処理容量の増加によって、以前より大きなコンピューティング サイクルが必要になった場合に、処理を効率よく効果的に分散できるようにしています。

一連のビッグ データ サービスにおいては、お客様が Workday のクラウドにアップロードしたサードパーティーのデータも分析が可能です。このサービスは、インサイトサービスを介して Workday レポートを利用できるように、(Hadoop に保存されている) お客様企業のデータ上の MapReduce 操作の実行結果を、データ管理サービスへ返送します。インサイト サービスは離職リスクなどのスコアも提供します。Workday の SYMAN (SYstematic MAss Normalization) サービスを利用してデータにアクセスする機械学習モデルによって、これらのスコアを導き出すことができます。SYMAN のサービスは、データの収集と分析を管理するもので、インサイト コンピューティング サービス向けに予測とレコメンデーションを生成するため、術語を正規化し、機械学習を適用しています。

Workday は常にインメモリ アプリケーションでした。当初から、Workday はお客様のコアデータを Java ベースの OTS のメモリ空間に維持してきました。しかしながら、長い年月を経て、アプリケーションのパフォーマンスとスケーラビリティをさらに高めるため、インメモリ データの利点を活かして、単一の OTS を、多層のサービス (トランザクション、データ管理、コンピューティング) へと進化させました。

Workday は、インメモリ データ管理の強化を続けることが、アプリケーションのスケーラビリティを保持していく鍵であると考えます。これはもはや SQL を介してアプリケーションにデータを提供するディスク上のデータベースだけで解決されるような問題ではありません。当社のインメモリ データ サービスは、アプリケーション データを管理するために、高度に最適化したマルチレベル キャッシュ アプローチの根幹となるものです。

更新トランザクション サービスや、テナントのライブ コンピューティングサービスで処理するデータを取得するために、更新トランザクション管理サービスがデータ管理サービスとやり取りする仕組みを下の図で示しています。

図 7 では、更新トランザクション サービス (コンピューティング サービスに作用するのと同一のメカニズム) は、インスタンス キャッシュが最新状態の「変更セット」を保持するところから始まります。この変更セットが、テナントの更新トランザクション

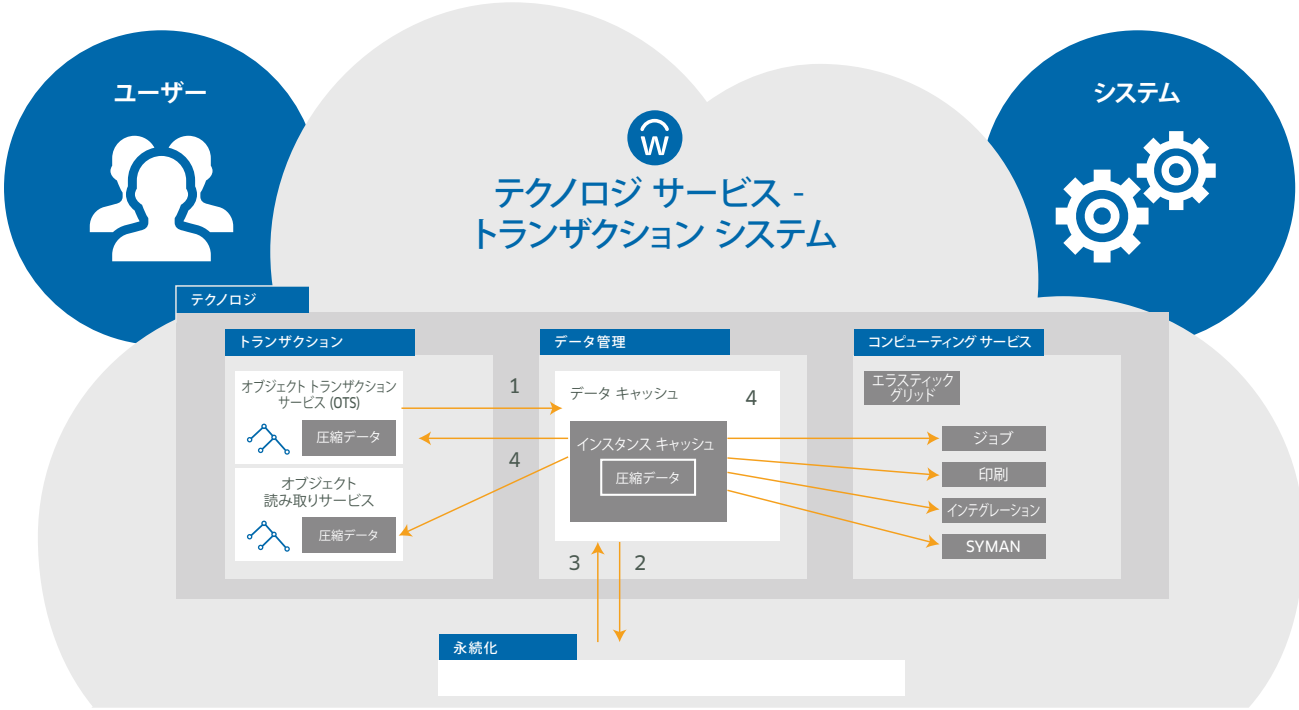

図 7

サービスから発行された実行中のトランザクションに対応しています (図 7、ステップ 1)。トランザクションがデータベースに対してアップデートされる (ステップ 2) と、変更セットがキューに登録されます。このキューを利用するのは、インスタンス キャッシュと、テナントの他のすべてのライブ コンピューティング サービスです (ステップ 3)。要求されたレポートや計算のプロセスを実行する前に、テナントのライブ コンピューティング サービスが、データベースのバージョン フラグをチェックし、最新のアップデートが適用されているかを確認します。最新の状態でなければ、キューに登録されたすべての変更セットを適用してから、レポートや計算を実行します。このプロセスによって、どのコンピューティング サービスで使用する情報も、最新のトランザクションを反映したものであることが保証されます (ステップ 4)。

インスタンス キャッシュは、トランザクション データやメタデータのアクセスを管理します。これに対して、DocStore サービスは、Basho Riak (NoSQL データ ストア) に保存されている非構造化データの保存と取得を管理します。このプロセスの詳細については、本ホワイトペーパーの「永続化」の項をご参照ください。

何百万ものインスタンスが作成されるクラスのために、Workday はインスタンス サービスと連携するクエリ サービスを用意しています。これにより、大きなデータセットを多次元でドリルダウンするレポートの作成が可能となります。このサービスは、財務仕訳明細など、ボリュームの大きなオブジェクトのレポートにおいて頻繁に使用されます。パフォーマンスは、インメモリ検索インデックスによって異なります。

最後にご紹介するのが検索サービスです。検索サービスにより、Workday のデータに含まれる術語や、SYMAN で正規化された単語を検索できるようになります。これにより特定の検索要求によって得られる結果の関連性が、より高められます。たとえば「ジョン・ジョーンズ」という名前の社員を探す場合でも、「ジョン」だけで正しい結果が検索の上位に表示されるようになります。

図 8

プラットフォーム

プラットフォーム サービスにより、テクノロジ層で使用される、再利用可能な汎用コンポーネントが提供されます。テクノロジ層で使用されるサービスには、認証と承認、通信、ログ、監視、顧客状況を示す情報などの機能があります。要求元 (レポート、ブラウザーやモバイル クライアントのタスク、インテグレーション) に関係なく、同じサービスは常に 1 つの情報源を利用するため、アプリケーション動作の整合性が保たれ、管理の手間とコストが抑えられます。

サービス ディスカバリを使って、各テナントが利用できる分散コンピューティング サービスを把握できるようになります。「テクノロジ」の項でご説明したように、各テナントでは常に更新トランザクション サービスが稼働していて、必要なコンピューティング能力の増加に応じて追加のコンピューティング サービスが起動されるため、ニーズに応じた最適なパフォーマンスを得ることができます。新たなテナント要求を受け取ると、Workday はこのサービスとやり取りし、既存のコンピューティング サービスにルーティングするか、あるいは新しいコンピューティング サービスを実行時に起動するかを判断します。Workday のインフラストラクチャがニーズに応じて動的に拡張可能なのは、簡潔に言えば、こうした理由によります。

認証サービスは、セキュリティを一元管理します。このサービスは、すべての Workday アプリケーションでのデータ アクセスにセキュリティを設定し、安全性を確保します。このプロセスは、シングル サインオンが導入されていても、ビジネス スイートのモジュール毎に幾つもの承認フレームワークを必要とするシステムとは対照的です。

ログ サービスと監視サービスが、すべてのトランザクションのアクションを記録します。このため、「だれが、いつ、何を行ったか」を常に監査することができます。

永続化

永続化層のサービスにおいては、Workday のアプリケーションで使用するデータのうち、一定の永続化や時間的な耐久性を必要とするデータへのアクセスを管理します。通常は、テクノロジ層の一連のデータ管理サービスが、この永続化層のサービスと直接やり取りをします。

Workday では、従前とは異なり、永続化をリレーショナル テクノロジだけに頼りません。トランザクション データには MySQL を使用し、非構造化データ (履歴書、ビジネス プロセス関連のドキュメント、写真など) は Basho Riak に保存していま

図 9

す。膨大な量のデータに対するスケーラビリティが高いこと、複製が可能なことなどの理由により、Workday は Basho Riak を採用しました。

Workday のプラットフォームは、お客様が膨大な量の外部データセットを当社のクラウドにアップロードすることを可能にしています。アップロードしたデータは Hadoop ファイル システムに格納されます。Workday が Hadoop を選んだのは、大容量のデータに対応できること、どのような形式のデータも処理できる柔軟性などが理由です。外部データと Workday のアプリケーションデータを連携させることで、お客様が活用する分析やレポートの精度がさらに向上します。

ストレージに対する Workday のこうしたアプローチには、以下のようなメリットがあります。

1. 最適化したスキーマを使用してトランザクション データを格納するため、新機能のリリースや毎週のアップグレードの際に、どのお客様に対してもスキーマの変更を管理する必要がありません。
2. Hadoop のスキーマレス アプローチを採用しているため、お客様はデータを当社のクラウドに簡単にアップロードできます。特定のストレージ標準 (SQL など) に縛られることがないため、その時々のニーズに最適なストレージ テクノロジを選択することができます。

これらのメリットの一例を挙げると、Workday では、すべてのトランザクションの格納方法を変更することができました。これまでは、ビジネス オブジェクトの変更は複数の行の挿入で記録していましたが、すべての変更を BLOB データにまとめて 1 回の挿入で記録することにしました。この最適化により、必要なディスク容量が半分以下となり、最初にかかるディスクからデータのインメモリ インスタンスを作成する際の時間も半減しました。こうした最適化の実現は、複雑なスキーマに SQL のアップデートを実行していては不可能でした。

特定の標準に縛られないという Workday の選択によって、お客様や当社のアプリケーション開発陣は、自由に新しいストレージ テクノロジに乗り換えることができるようになりました。ストレージのテクノロジはかつてない速さで進化しています。この状況を踏まえ、Workday はこのストレージ アーキテクチャを採用することで、ストレージに対して常に最適なアプローチが取れるよう心がけています。オブジェクト指向設計業界のオピニオン リーダー Martin Fowler は、このアプローチを「ポリグロット (多言語) 永続化」と呼んでいます。

インフラストラクチャ

Workday アーキテクチャのインフラストラクチャ層は、典型的な IaaS サービスを扱います。インフラストラクチャ層は、アーキテクチャ全体の機能をまとめるための、Workday のオペレーティング システムと言うことができます。自動化により、サービスをプロビジョニングする複数のサーバーは、お客様固有のコンフィギュレーションに適切なサービスをインストールできるようになります。これらのサービスは、コンピューティング サービスのインスタンスの一部として各テナントが利用できるようになります。このように、完全にエンドツーエンドのインフラストラクチャによる自動的なプロビジョニング システムがなければ、現状のビジネス ニーズに合わせていつでも動的にサービスを拡張することは不可能です。

Workday は、インフラストラクチャを設計、構築する際に、アーキテクチャを「スケールアウト」するという考え方を採用しました。つまり、サーバー、ストレージ、ネットワークを追加することで、保存容量を増やし、パフォーマンスを向上させます。より大きなモデルや機器へと垂直にアップグレードする方法とは対照的ですが、当社は、ソフトウェア アーキテクチャにも同様の考え方を採用し、インフラストラクチャ上で実行されるサービスを追加して、アプリケーションの処理能力を拡張しています。

水平方向の拡張は、サービスの可用性に関して、多くのメリットを生み出します。1 つ目のメリットは、サーバーの冗長化です。1 台のサーバーで障害が発生しても、Workday アプリケーションはオフラインになることがありません。2 つ目のメリットは、パフォーマンスを犠牲にすることなく、ストレージ容量を増加できることです。インフラストラクチャ層がデータの増加に応じて拡張されるため、Workday のお客様のアプリケーションにおいては、パフォーマンスに問題が生じることはありません。

図 10

インテグレーションとユーザー エクスペリエンス

Workday のアーキテクチャに直接関係するその他の重要な側面が、インテグレーションとユーザー エクスペリエンスの 2 つです。これらは後付けではなく、初めからプラットフォームのネイティブ コンポーネントでした。

インテグレーション サービス

Workday のビジネス アプリケーションは多数の機能を備えていますが、お客様がサードパーティー製アプリケーションとのインテグレーションを必要とすることもよくあります。たとえば Workday 導入事例でよくあるのは、Workday ヒューマン キャピタル マネジメントと福利厚生プロバイダとのインテグレーションです。外部システムとのインテグレーションは簡単ではありませんが、Workday は、他のシステムと簡単に接続するための複数の選択肢を用意しています。

アプリケーションのインテグレーションに最適なアプローチは Web サービスであると Workday は考えています。Workday のアプリケーションは、すべてのタスクの Web サービスを自動的に生成するように作られており、他のシステムとの双方向通信が可能です。しかし、Web サービスの仕様には統一された標準がありません。そこで Workday は、プラットフォームにエンタープライズ サービス バス (ESB) テクノロジを組み込みました。このようなインテグレーション サービスが、サードパーティー製アプリケーションの技術仕様に合わせて、送受信のデータ ペイロードを柔軟に変換し、デリバリー プロトコルを変更します。

インテグレーション サービスでは、お客様が独自のインテグレーションを構築して Workday のクラウドで実行することも可能です。Workday はこのインテグレーションを実現するために、エンタープライズ インターフェイス ビルダー (EIB) と Workday Studio という 2 つのツールを用意しています。この 2 つのツールは、Workday Web サービス アプリケーション プログラミング インターフェイス (API) の操作が可能です。

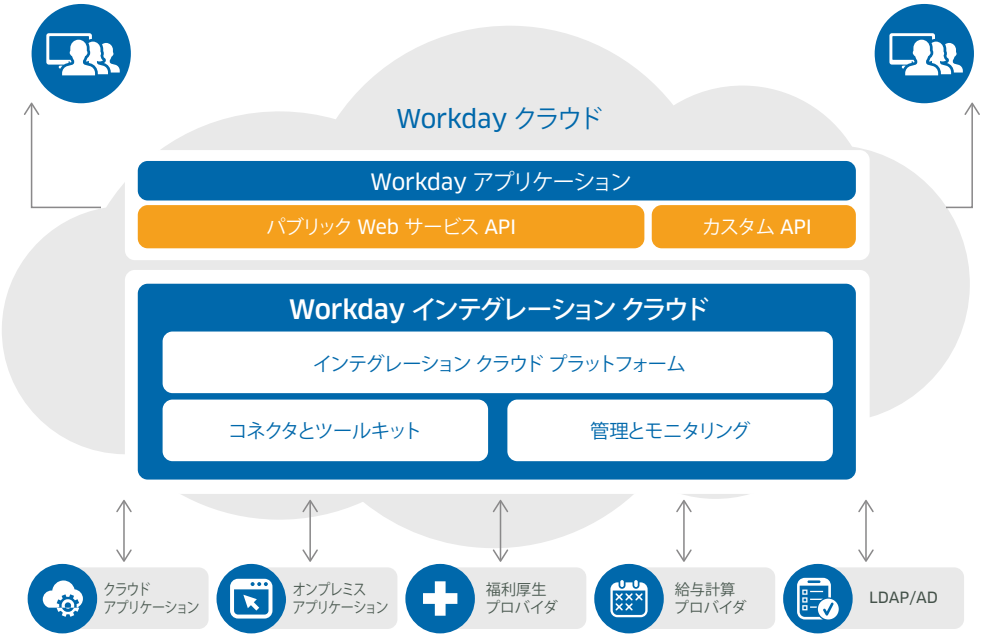

図 11

- **エンタープライズ インターフェイス ビルダー (EIB):** EIB によって、フォームベースの UX を使用して、ユーザーはシンプルな単一方向のインテグレーションの作成が可能となりました。このツールを使って「インテグレーション システム」を構築することにより、お客様はインテグレーションを Workday のクラウド上で実行するスケジュール設定が可能となります。
- **Workday Studio:** 複数のサードパーティー製システムと Workday アプリケーションとの複雑なインテグレーションをお客様が構築できる強力なツールです。EIB と同様、インテグレーション システムを構築して、Workday のクラウド上で実行するスケジュールの設定が可能です。

これらのツールは、Workday のアプリケーションを購入すれば、追加費用なしで利用することができます。お客様は、お好みのプログラミング言語を使用して、Workday のパブリック Web サービス API をいつでも直接操作することができます。Workday は、自社のビジネス サービスにプログラムからアクセスするために、標準ベースのオープンな API を提供しています。当社が提供する API ドキュメントは次の 3 種類です。

- **パブリック Web サービス API:** Workday は、ビジネス サービスにプログラムからアクセスするための、標準ベースのオープンな API を提供しています。
- **RaaS (Report-as-a-Service):** お客様が Workday のレポート ライター ツールを使用してカスタム レポートを作成する際に、独自の Web サービスを構築できるようにします。インテグレーション用のカスタム レポートは SOAP と REST ベースの Web サービスとして公開することができます。
- **REST API:** Workday の REST API が対象とするのは、通常はセルフサービスでユーザーが始動する小さなトランザクションを実行するアプリケーションです。ユーザーが相互のやりとりを開始できるように、REST API は、表示やその後のアクションに備えて、頻繁に使用される少量の情報を、すばやく返すように設計されています。

アプリケーション プラットフォームの中にインテグレーション サービスを組み込むメリットを、いくつかご紹介します。

- データ ペイロードの変換や、適切なデリバリー プロトコルの選択に ESB テクノロジを使用することで、Workday は、接続するすべてのアプリケーションに対して柔軟なエンドポイントのように機能します。
- Workday のアプリケーションにツールを組み込むことにより、当社のクラウド上でどのようなインテグレーションを実行するかを、お客様ご自身で選択できるようになりました。また、お客様が当社のクラウドを有効に活用することで、インフラストラクチャ上で実行されるインテグレーションの構築と保守にかかるコストを削減できます。
- インテグレーションをアプリケーションと一体化し、ビジネス プロセスとシームレスに連携させることができ、コンテキストに応じたコンフィギュラブルなインテグレーションが可能になります。

ユーザー エクスペリエンス (UX)

Workday は、UX の生成と、アプリケーションの定義を完全に分離しています。開発に当たっては、画素の 1 つひとつにこだわったレイアウトや、特定のフィールドがページのどこに表示されるかも指定しません。その代わりに、トランザクションに関係するフィールドを定義し、フィールドをグループ化する方法とその順序を指定します。この情報からトランザクションのプレゼンテーションを生成するのが、UX サービスです。

このアプローチによる UX のメリットはいくつかあります。生成された UX は整合性が確保され、たとえばページをすべて同じ方法でレイアウトするために、何百人もの開発者をトレーニングするといった必要がありません。UX とアプリケーションの定義を分離することで、アプリケーションを変更することなく、UX に使用するテクノロジの変更が可能となりました。

現在、Workday のブラウザーにおけるプレゼンテーション層は HTML 5 です。HTML 5 は、DHTML、Flash に続く 3 代目のブラウザー UI テクノロジですが、Workday は、こうした UI テクノロジがアプリケーションより速いペースで変化し続けると考えています。このため、Workday のアーキテクチャはプレゼンテーション層を頻繁に変更し、常に最新の状態を維持できるようにしています。当社がモバイル エクスペリエンスから学んだのは、少なくとも 1 年毎に設計をアップデートしなければ、UX は時代遅れになってしまうということ。これは、ブラウザー版の UX も同様だと考えています。

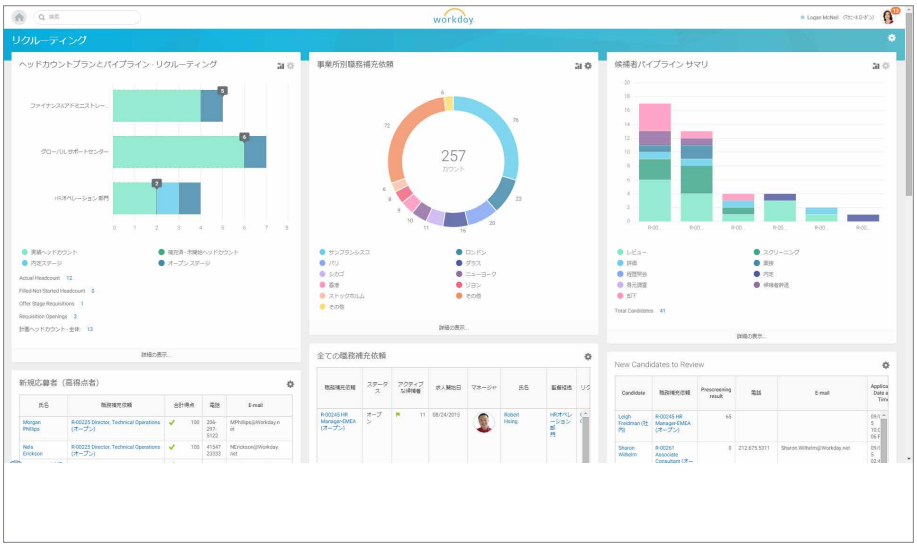

図 12

モバイル

モバイル アクセスは、今日のアプリケーションの基本機能であり、別製品や別プラットフォームとして提供されるべきではありません。Workday では、モバイル アクセスは UX サービスの一面に過ぎず、すべてのアプリケーションにその一部として組み込まれるものだと考えています。

図 13

Workday のアプリケーションは、デスクトップ ブラウザーからアクセスできるだけでなく、iPhone、iPad、Android など、最新のモバイル ブラウザーからもネイティブに利用できます。各クライアントの UX は、タッチ操作とフォーム ファクター (ディスプレイサイズ) に基づいて最適化されるため、ユーザーはデスクトップ ブラウザー向けに設計されたフィールドやボタンを、ピンチしたりズームして操作する必要はまったくありません。ネイティブなモバイル アプリケーションは、最適なユーザビリティを実現します。Workday は、HTML 5 ブラウザー クライアントを、タブレットとスマートフォンの機能セットやフォーム ファクターに完全に適応させるとともに、ネイティブ クライアントのサポートにも注力しています。

Workday のモバイル ソリューションは、単にアプリケーションの UI を拡張しているにすぎません。故にモバイル ユーザーはデスクトップと同じデータ、ビジネス ロジック、機能にアクセスしていることになります。たとえば、デスクトップ上に構築されたダッシュボードは、モバイル端末にデータを保存することなく、どのモバイル機器からでもアクセスが可能で、変更を加えることができます。さらに お客様がブラウザー アクセス用にセットアップするのと同じ、単一のセキュリティ モデルをモバイル クライアントでも利用するように設計しています。このため、お客様は、Workday のモバイル ソリューションを簡単に実装することが可能となります。必要な設定は、(アプリをダウンロードすること以外に) モバイル機器から Workday にアクセスする権限をすべての相応なユーザーに付与するだけです。

開発プロセス

Workday は、アップデートを頻繁に提供すると同時に、お客様のソフトウェアを常に最新バージョンに保つよう取り組んでいます。この取り組みは、プラットフォーム アーキテクチャの設計に大きな影響を与えています。また、製品、プラットフォーム、インフラストラクチャのすべてにかかわる開発プロセスの見直しにもつながっています。

Workday は、新製品とプラットフォーム機能の継続的デプロイメントを単一のコードラインで行っています。このアプローチは、増え続けるお客様のすべてに多くの新機能を滞りなく提供する唯一の方法です。開発に当たっては、小規模のスクラム チームが編成され、機敏に活動しています。設計、開発、品質管理を一貫して担当し、製品に対応する機能を反復手法で開発しています。

ひと度、機能がテスト パイプラインをパスすると、本番運用コード ラインにチェックインされます。トグルと呼ばれるソフトウェア スイッチを使用し、その機能について、開発者、プレビュー モードのお客様、本番運用のお客様のいずれが参照できるかが決定されます。大規模な自動テスト インフラストラクチャが、単一のコード ラインでの開発を可能にしています。

Workday は、新機能のチェックイン時に、お客様のデータをバックグラウンドで変換するプロセスも開発しました。これにより、当社の環境チームやインフラストラクチャ チームがアップデート当日に行う作業が、大きく変わりました。新機能をプレビュー モードから本番運用モードに切り替えるだけでアップデートの実施が可能となり、1 日を要して新しいコード ラインに移行し、変換プロセスを実行するといった必要がなくなりました。

スクラム チームで開発するメリットには、サービス全体に一貫した責任を持てることが挙げられます。当社のスクラム チームは、個々のランタイム サービスの開発、デプロイ、監視を担当し、これにより、Workday の総合的なサービス提供能力が強化され、お客様への応答性も向上します。

データセンター インフラストラクチャも、当社の製品と顧客ベースの成長に合わせて、絶えず改善していかなければならないシステムです。この考え方により、Workday では、インフラストラクチャは開発部門に属しており、Dev-Ops プロジェクトを新しいアップデート毎に組み込んでいます。テクノロジの進歩により、データセンターをソフトウェア リソースの 1 つとして扱うことが可能になりました。成長を続けるお客様に合わせて、より効果的にインフラストラクチャ リソースを割り振り、共有できるようにするために、Workday は、サービスの自動化と、コンピューティング サービスやネットワーク サービスの仮想化といった分野にも、引き続き積極的な投資を行ってまいります。

終わりに: 継続的変化への対応

Workday は、2005 年の設立時に「白紙からのスタート」を切りました。従来のエンタープライズ アプリケーションでは満たすことのできないニーズに応えるために、テクノロジに対して、新機軸を打ち出したアプローチを採用したのです。設立当初、Workday は次のような方針を掲げました。

- (リレーショナルとは正反対の) オブジェクト指向のアプリケーション設計
- (コーディングとは正反対の) 定義型のメタデータ開発
- インメモリのアプリケーション データ
- UX の生成

このガイドラインにより、インメモリ データの管理、永続化、UI テクノロジなどを意識することなく、ビジネス ロジックの開発に専念できることになりました。ビジネス ロジック自体も、特定の基盤テクノロジ (SQL、HTML、JavaScript など) に依存しないよう、アプリケーションを定義するビジネス ロジックと、アプリケーションを実行するテクノロジ プラットフォームは明確に分離されます。これにより、互いに悪影響を及ぼすことなく、個別の開発が可能になりました。

Workday の設立当初のアーキテクチャは、次のようなシンプルな図で表すことができました。

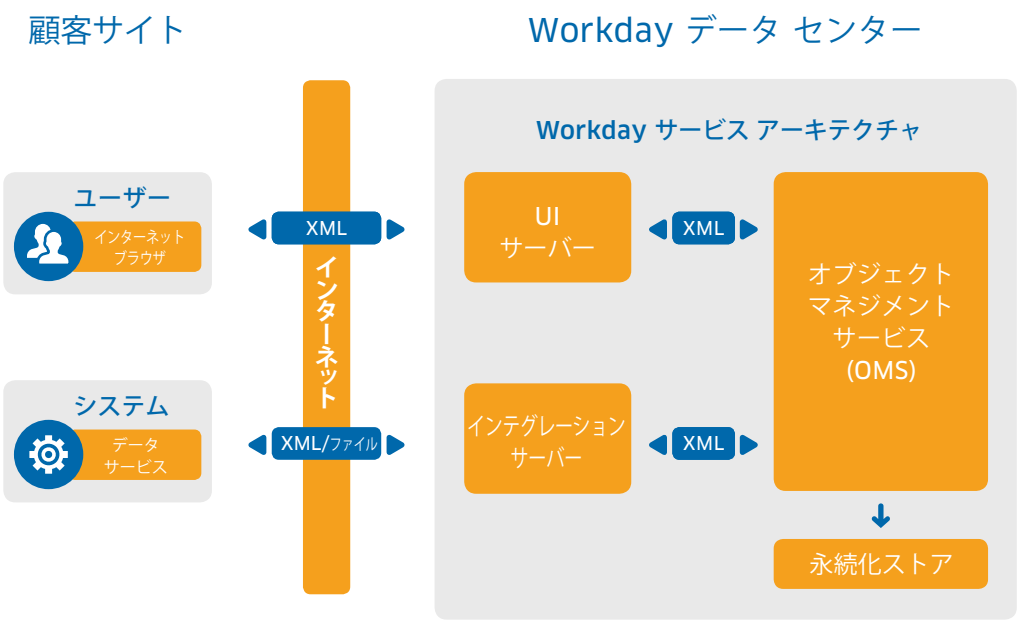

図 14

設立から歳月を経て、アーキテクチャの実装はさらなる進化を続けていますが、当社の初期の方針は何ら変わっていません。現在のアーキテクチャは、少しだけ洗練されたものになってはいますが。

あらゆる変化を受け入れ、お客様のニーズを満たす最新のアプリケーションを常に提供すること。Workday が頑に守り続けている思いです。この信念が、お客様にとっての当社の製品の活用方法を常に進化させることを可能にし、高コストなアップグレードを実施することなく導入可能な新機能を、絶えず提供することを可能にしています。同時に、不断の技術革新への努力と、アーキテクチャの実装を常に最新の状態に維持することも可能にしています。

エンタープライズ アプリケーションは、Facebook、Amazon、Netflix に代表されるエンドユーザー向けの Web サイトのようになるべきだと Workday は考えます。求められているのは、絶えずアップデートを実施し、洗練されたアプリケーションを目指すこと。これらのサイトのお客様は、インフラストラクチャの詳細を意識することなく、優れた完成品だけを求めているのですから。同様に、今使っているバージョンなど気にしなくても、常に最新バージョンが提供されていることを求めているのですから。

エンドユーザー向けの著名なアプリケーションと同じようにお使いいただけるものを。そんな願いのもとにデザインされているのが、Workday のアーキテクチャと開発プロセスです。より優れたアプリケーションとプラットフォーム開発のための改革を常に果敢に断行し、お客様のビジネス目標達成のお手伝いをすること。Workday が目指す何よりの使命です。

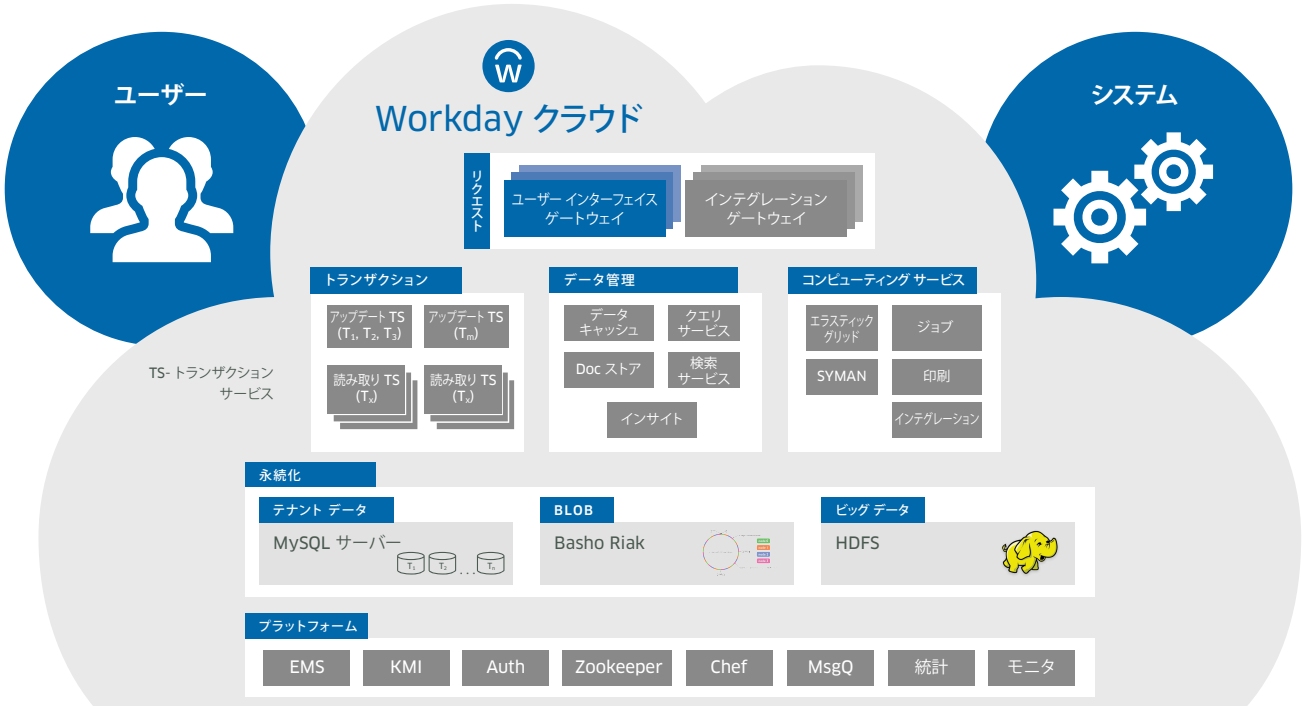

図 15

ワークデイ株式会社 | 代表: 03 4572 1200 | www.workday.co.jp