ホワイトペーパー：**APM** パフォーマンスライフサイクル

アプリケーションパフォーマンスのライフサイクル全体を最適化する方法

目次

A DIVISION OF COMPUWARE
dynaTrace software

Compuware
THE TECHNOLOGY PERFORMANCE COMPANY

エグゼクティブサマリー

仮想化やクラウドの普及により、アプリケーションの構築、テスト、リリース、管理の方法は変化しつつあります。リリースサイクルは加速しており、部門横断的な課題が発生しています。

- アプリケーションは、開発初期段階で、本番環境に応じた設計を行う必要がある
- テスト環境と実装環境が大きく異なるため、本番システムで問題が発生している
- 本番環境は動的なため、問題解決への道筋を見極めるためには、プロアクティブな管理、継続的なデータキャプチャと分析、迅速な診断が求められる
- 外部から本番環境を監視できない
- 外部ベンダーが提供するコードやサービスの影響範囲が拡大している
- パフォーマンスやスケーラビリティの問題は、早期に、そして、アプリケーションのライフサイクルを通じた対処が必要となる

これに加え、ビジネスサイドからは、効率性向上とコスト削減が要求されています。投資の妥当性は、効果を得るまでの期間、収益に対する直近の影響、長期的なTCOで評価されます。このような観点から、開発サイクルを短縮しながら品質向上を達成する方法として、アジャイル開発が広く採用され始めています。しかしながら、依然として、運用は開発と独立して行われ、開発者は運用フェーズで使用される手順やツールにはアクセスできず、開発で使われるツールとも互換性がありません。その結果、開発を終了して運用に渡されると、アジャイルで達成されるはずのメリットが失われることになります。ここにジレンマがあります。では、これら2つの機能をどうすれば統合できるのでしょうか。

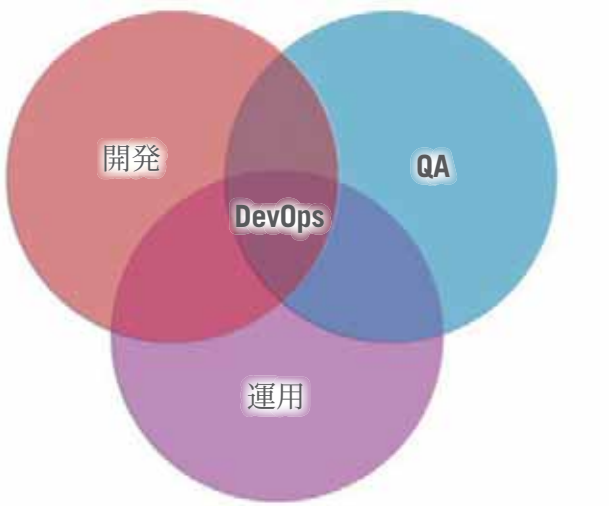

図1：DevOpsの定義

開発と運用の間に存在する壁を打破し、相互のコミュニケーションを円滑にし、プロセスやツールを統一するために、新たな方法論が提唱されています。それは、「DevOps (Development + Operations)」と呼ばれるもので、異なるチームを統合し、アプリケーションの迅速なリリースと展開を目標としています。DevOpsはソフトウェア提供のためのトータルな手法であり、ソリューションプランと一連のタスクを、単一のツールセットに基づいて進められます。

DevOpsの手法では、コンセプト段階から使用終了にいたるまで、アプリケーションのライフサイクルを通して、共通のアプリケーションパフォーマンス管理(APM)プラットフォームを利用し、プロアクティブな問題の解決と防止の基盤を提供することが、従来にも増して重要となります。このようなAPMプラットフォームの構築により、5年から10年に渡る、アプリケーションのライフサイクルを通じて、TCOの大幅な削減を可能にします。

このホワイトペーパーでは、開発／テスト／本番といったアプリケーションのライフサイクル全体にわたって、プロアクティブなパフォーマンス管理を可能にする共通APMプラットフォームの要件とメリットについて解説します。

概要

アプリケーション開発は本番リリースまでの時間を短縮するために、従来の開発手法からアジャイル開発へ移行する動きが広がっています。アジャイル開発により、開発サイクルの短縮、品質の向上、リスク低下、変更等への柔軟な対応が可能となります。アジャイル開発は、次のようなメリットが期待できます。

- 技術上、ビジネス上、いずれの要件変更に対しても迅速に対応可能
- 誤った方向に作業を続けるリスクの低下
- サイクル毎にリリース可能なバージョンを準備
- テストプラクティスの開発ライフサイクルへの組み込み
- ビルド、機能テスト、プロセスレポートなど多くのタスクを自動化

アジャイルによってソフトウェアの設計開発は迅速になりますが、運用部門によって行われるリリースと展開については、ほとんど変化がありません。問題は、ビジネス部門やその役割について、開発部門と運用部門が異なる観点で見ているという点にあります。

図2：開発部門と運用部門は、ビジネスとそれぞれの役割を異なる観点で見ています

開発部門は多くの場合、変更を行うことが自分たちの任務で、要求の変化に応じることがビジネスにとって重要であると考える傾向があり、自分たちに与えられる動機づけや評価の観点から、できるだけ多くの変更を加えようとします。これに対して運用部門は、変更は悪であると見なしています。彼らの役割は、ビジネスプロセスを安定させ、24時間365日システムを稼働させ、収益を生み出すサービスを継続的に提供することです。さらに問題を複雑にする要因として、開発部門と運用部門が通常、異なる組織に属しており、多くの場合には地理的にも離れているため、コミュニケーションが困難であることがあげられます。タイムゾーン、言語、文化などの違いが、部門間に存在する観点の相違をさらに大きくします。

図3：開発チームと運用チームは多くの場合、異なる拠点に存在し、地理的にも離れています。

開発部門が変更を施したアプリケーションをリリースし運用部門に提供した場合、それぞれの部門は異なるツールやテクノロジーを使用しているため、しばしば、コストが増大します。運用部門は、開発部門やQA／テスト環境で完了している作業が本番環境でも正しく動作することを確認するために、同じ作業を繰り返す必要があります。この非効率な作業により、本番環境への展開フェーズで時間がかかり、新しいシステム展開によるメリットの享受が遅延します。運用部門による動作確認のフェーズでは、見逃されていたバグが見つかることもありますが、新たなバグが発生することもあります。

こうした問題が発生すると、トラブルシューティングのために開発者が呼ばれます。ここで、責任の押しつけ合いが始まることになります。運用部門は、開発部門が渡したコードがおかしいと主張します。開発部門は、自分たちの環境ではうまく動作したのだから、運用部門が何か間違えたに違いないと回答します。開発部門は問題の診断を手助けしようとしますが、運用部門とは異なるツールと手順を使用しているため、作業は困難になります。さらに、開発部門は本番サーバーへのアクセス権限を持っていない可能性があります。このため、開発部門の独自環境で問題の再現を試みるしかありません。

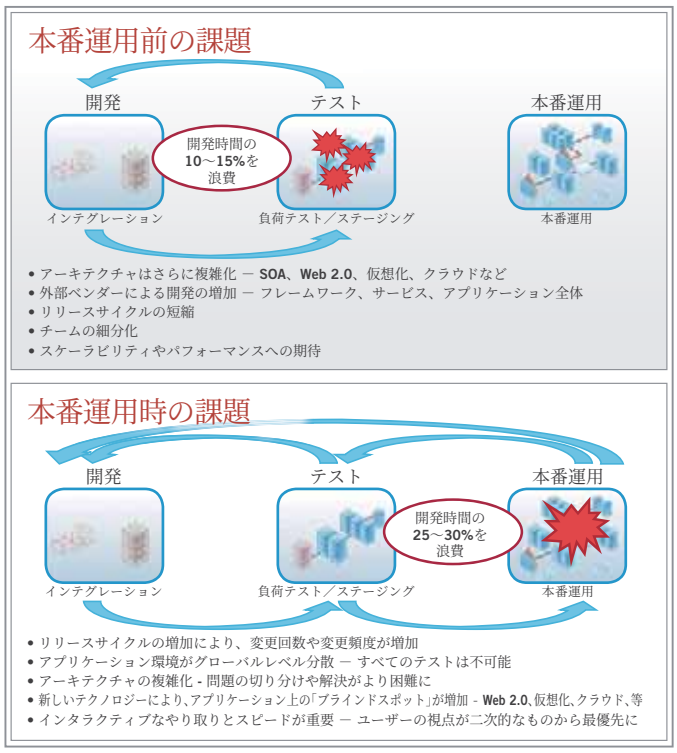

図4：ライフサイクルでの機会

トライ&エラーによって時間をかければ、多くの場合、解決策は見つかります。しかし、バグを引き起こしている問題が明確にならないこともあります。状況によっては、解決策が見つからないこともあります。その場合、特定の機能あるいは機能強化を取り消すか、正しく動作していた以前の環境にロールバックするか、という判断が必要になります。これは、時には本番環境に手を加えて使える状態にすることが必要になり、言わば、技術上の「負債」を負っているようなものです。皆さんも経験があるのではないでしょうか。

開発部門と運用部門のスケジュールや作業ペースが異なっているために、開発から運用にいたるライフサイクルの観点から、別の課題が発生することがあります。開発部門がアプリケーションをリリースするサイクルや頻度に運用部門が対応できず、本番環境への展開が行われる間隔が長くなりがちです。その場合、開発部門はすでに別のプロジェクトや成果物の作業に取りかかっており、運用部門の問題解決に携わることが難しくなります。さらに、このような状況を打破するために、アジャイルライフサイクルから、従来のウォーターフォールライフサイクルに体制を戻すケースもあります。

チームが協力し会社に競合優位性をもたらす方法

双方の間に存在する壁を打破し、相互のコミュニケーションを促進し、プロセスやツールを統一し、アプリケーションの円滑なリリースと展開を推進するという共通の目標に従って、これらのチームを ── 対立ではなく共同作業を行うように ── 統合する、DevOps (Development + Operations) という方法論が新たに登場しました。DevOpsにより、開発と運用の役割やプロセスを、共通のビジネス目標に同期させ、ビジネスの俊敏性や、ITとビジネスの連携を達成できます。どのように組織を設置するかではなく、統一されたビジネスプロセスのサポートと改善をいかに進めるかが、より重要になります。

DevOpsは、組織の環境に応じて、様々なやり方で実装できます。アジャイルの「3つのC」、つまり、継続的なインテグレーション（Continuous Integration）、継続的なテスト（Continuous Testing）、継続的な提供（Continuous Delivery）を十分に取り入れている組織であれば、アプリケーションは定期的に本番環境にリリースされます。これは多くの場合、クラウドベースのソリューションで見ることができます。そうでない場合は、DevOpsの手法を取り入れ、開発チームと運用チームとの間で、より効率的な統合が行われます。特にアプリケーションリリースの引き継ぎ時には、作業をよりスムーズで効率的にするために、この点が重要となります。

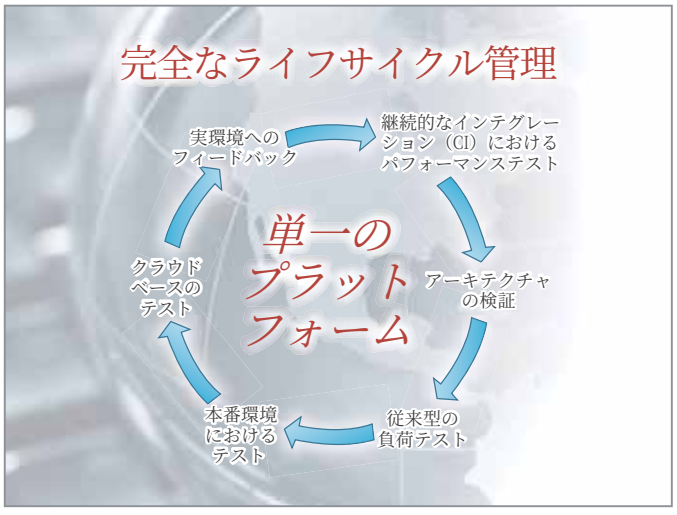

図5：完全なライフサイクル管理

DevOpsはソフトウェア提供に対する包括的な手法であり、ソリューションプランと一連のタスクを、1つのプラットフォームを中心に行うものです。

- チーム配置の集約：チームを地理的に分散させるのではなく、同じロケーションに集約します。開発担当と、アプリケーションの運用や展開の担当者とが同じ部屋で作業できるようにします。
- 単一のプラットフォーム：異なるツールを利用しないため、データ変換の手間や、メンテナンスに必要な時間を削減できます。さらに、本番環境における自動テストの監視設定や、アプリケーションリリース時の自動ビルドやテストの設定を、再利用することができます。
- アプリケーションの俊敏な提供：チームが数日以内に新機能に対応できるようになることで、ビジネス部門やユーザーの要求に俊敏に応えることができます。
- 障害の予期とプロセスへの折り込み：プロセスが最適化されるため、労力のほとんどは、障害の予防ではなく回復に向けられます。
- 責任の明確化：開発、品質保証、リリースエンジニアリング、セキュリティ、運用などの各チームが、プロセスを完了するために各自の役割を果たします。問題が発生した場合、責任の所在が明確に決定されています。

市場の変化に追いつけない旧来の**APM**ツール

> 「アプリケーションの **10** 年間の **TCO** のうち、システム構築にかかる初期投資は **8%** にすぎないということは明白な現実です。実は、**TCO** の **92%** は、アプリケーションの機能拡張、改修、運用のコストです」[1]
>
> Gartner

パフォーマンスを観察し、記録し、対策を行うことができるかどうかが、アプリケーションを投入することができるかどうか、そして、ユーザーが快適にアプリケーションを使用できるかどうかということに、直接影響を与えます。DevOpsにおいて、ライフサイクルの開始から終了まで、すなわち、コンセプト段階から使用終了にいたるまで、共通のアプリケーションパフォーマンス管理（APM）プラットフォームを活用し、プロアクティブな問題の解決と防止に必要となる基盤を提供することが、従来にも増して重要となります。

ここで、「ツール」ではなく「プラットフォーム」という用語が使われていることにお気づきでしょうか。既存のAPMソリューションは、多くの前提に基づいて設計されていますが、現在の状況では不適切なものも多く、パフォーマンスライフサイクルにわたって使うことはできません。それは、データセンターを中心とし、タスクをベースとする、というものです。また、言語を単一のものとして想定しています。しかしながら、ファイアウォールの内部からはクラウドアプリケーションを監視できません。つまり、この前提は柔軟性を欠き、変更に適応することもできないのです。例えば、コンテンツ配信ネットワーク（CDN）がユーザー体感に与える影響を見通すこともできず、

[1] Gartner、「*Enterprise-Class Agile Development Defined*」、2011年3月29日

Web 2.0の機能を十分に活かすことはできません。また、開発とリリースの迅速なサイクルをサポートできません。このように、旧来のAPMツールは、本質的にはポイントソリューションであり、運用コストが大きいため、DevOps環境の変更の速度に追従できません。

ライフサイクル全体を通じた共通APMプラットフォームの必要性

物理、仮想、クラウドのいずれの環境でも利用でき、複数の言語(Java、.NET、C/C++など)で開発されたアプリケーションをサポートし、優れたROIを実現し、迅速に実装できる、使いやすいAPMプラットフォームが必要です。理想的なAPMプラットフォームは、次のようなものです。

- アプリケーションの完全な可視化
- 優れたユーザー体感の提供
- アプリケーションの迅速なリリースとライフサイクル全体を通じた共同作業を推進
- アプリケーションの動作に対する正確かつ詳細なビューを提供(ユーザーが画面をクリックしてからデータベースにいたるすべての階層(ブラウザ、コンテンツ、ネットワーク、サーバー、クラウド)を通して、全てのトランザクションをキャプチャ)
- これらの要件を、開発/テスト/本番で共有可能な単一のプラットフォームで実現

上記の要件を満たすシステムであれば、DevOps環境に基づき、プロアクティブな問題の解決と防止の基盤が提供され、アプリケーションライフサイクルで(5年から10年)にわたり、TCOの大幅な削減が実現できます。旧来のAPMツールが、常に期待を下回り、分断されたパーツを組み合わせるために多くの専門的なサービスを必要とし、毎年メンテナンスを行う必要があることを考えると、これは大幅な改善です。理想的なAPMソリューションは、全く新しい設計により、与えられた要件を予測可能なものにし、対処できるものにします。

APMプラットフォームは、開発/テスト/本番の各チームに「共通言語」を提供するものでなければなりません。そして、運用部門やテスト部門が簡単にデータをキャプチャして他の部門に送り、開発者やパフォーマンス担当者が対処する上で十分な詳細情報が含まれている必要があります。アプリケーションパフォーマンス管理チェーンの全てのステークホルダーである、開発者、アーキテクト、運用担当、ITマネージャー、ビジネスマネージャーへのサポートも必要です。さらに、コンポーネントの再利用が可能で、アプローチや技術が一貫しており、統合された共同作業フレームワークによりデータのキャプチャと共有が自動化されていなければなりません。

図6:運用、システムアーキテクト、開発者のビューを単一のデータソースに基づいて提供

データキャプチャの要件、つまり、トランザクション毎に異常値を適切に記録することも不可欠な要素です。高負荷の状態でアプリケーションの動作を正確に記録することは、ステークホルダーが情報に基づいたコミュニケーションを行う上で必要です。特に、これらのステークホルダーが地理的に分散していたり、または異なる企業のスタッフである場合には、重要になります。記録された情報は、オンラインでリアルタイムに利用でき、24時間365日止まることのないトランザクションを分析できなければなりません。また、開発者はリアルタイムでアクセスできないことが多いため、オフラインでも利用可能でなければなりません。コミュニケーション上のミスを最小化し、責任の押しつけ合いを排除し、適切なチームとメンバーにより問題を迅速に解決できることが理想的です。

APMプラットフォームは、さらに、既存のパフォーマンス環境にシームレスに組み込まれ、ROIを増大する必要があります。

- 開発フェーズでは、開発プロセス(アジャイルやDevOpsなど)を柔軟にサポートし、インテグレーションシステムやIDEと統合
- テストフェーズでは、テスト対象のアプリケーション、使用される負荷テスター、トラブルシューティングのプロセス、インシデントトラッキングシステムなど、あらゆるものへ容易に組み込みが可能
- 本番フェーズでは、アプリケーション自体や使用される管理コンソール、定義されたアラートプロセス、インシデントトラッキングシステムなど、あらゆるものへの容易な組み込みが可能

APMプラットフォームは、スタッフによって行われる冗長で時間を消費するタスクを、高速化または自動化し、ROIを迅速に達成する必要があります。また、アプリケーション開発、テスト、リリースのプロセスが全体として最適化されるにつれ、ROIをさらに拡大するものでなければなりません。

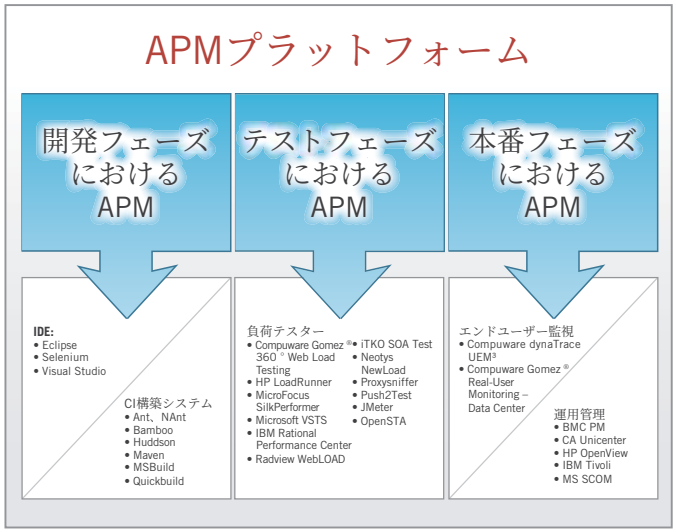

図7:APMプラットフォームは、ライフサイクル全体にわたり、シームレスな統合を提供します。

中長期的なプランを行う場合には、ライフサイクル全体をカバーするAPMプラットフォームは、仮想化されたデータセンターやクラウド用にアプリケーションを構築し最適化する上で、不可欠なものとなります。アプリケーションアーキテクチャと本番環境は、2つの独立したものとして扱われることはなくなります。アプリケーションが仮想化された本番環境で適切に拡張できるように考慮し、アーキテクチャの設計と構築を行うことが、初期設計段階の要件となります。この、

アーキテクチャと本番環境の相互依存性を管理する上では、開発／テスト／本番を効率的に連携・統合できるAPMプラットフォームのアプローチこそが、唯一の信頼できる方法です。

ライフサイクルに基づいたプロアクティブなパフォーマンス管理

今日、アプリケーションアーキテクチャは複数の階層で構成されているため、トランザクションがどのようなパスを経由し、問題がどの箇所にどのような形で現れるかを前もって予測することは、多くの場合不可能です。APMプラットフォームは、開発ライフサイクル全体をサポートし、問題が本番環境で発生する前に、プロアクティブに情報をキャプチャし防止する必要があります。また、時間を要する反復作業を自動化し、アプリケーションの動的な特長を自動でサポートする必要もあります。すべてのトランザクションに関するパフォーマンスデータを、インシデントの発生後だけではなく、24時間継続的に監視することも必要です。また、トランザクションについて詳細に分析する際、データの平均値だけでなく異常値も含まれている必要があります。

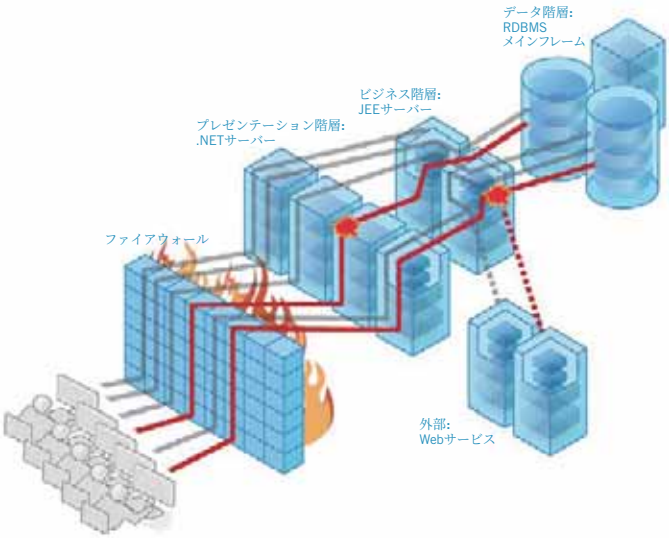

*図8：断続的に発生する問題を把握するには、トランザクショントレースを**24**時間絶え間なく継続する必要があります。*

トランザクションをベースとしたパフォーマンス情報は、関連するすべてのチームメンバーによって参照可能な状態にしておく必要があります。問題が特定されると、チーム間の円滑なコミュニケーションにより、迅速な問題の診断と解決に必要な情報が提供されます。このプロアクティブなライフサイクル手法により、開発とテストをカバーするプロアクティブなパフォーマンス管理が可能となります。そして、新機能を実装したサービスをリリースするまでの期間を短縮できるだけでなく、本番環境での予期できなかった問題（テストフェーズで正しく動作していたアプリケーションが、本番環境の規模に適応できないなど）を減らすことができます。

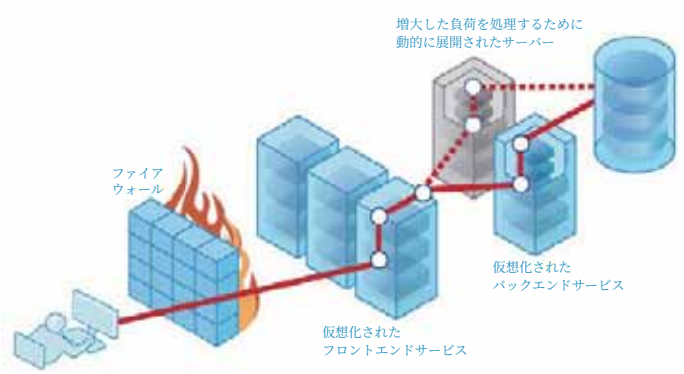

*図9：実行パスの動的な変化を把握するには、トランザクショントレースを**24**時間継続する必要があります。*

Compuware dynaTrace® APMプラットフォーム

dynaTraceは、新しい開発手法や新しいアプリケーションアーキテクチャ、急速に複雑化する本番環境などに起因する、APM上の新しい要件に対応するため、2005年に開発された、業界初のAPMプラットフォームです。dynaTraceは、ビジネストランザクション管理（BTM）、従来型の監視、ビジネストランザクションからコードレベルにいたるトランザクションの詳細診断、アプリケーションライフサイクル全体にわたるプロアクティブな問題回避を、単一APMプラットフォームとして統合しています。このプラットフォームは、革新的で使いやすく、従来のAPMツールが実現できなかった付加価値を提供します。

アプリケーションの動作を最も正確に把握

dynaTraceは、他のAPMシステムと比較して、より多くのアプリケーション階層をカバーし、アプリケーションをより詳細レベルで可視化し、設定やメンテナンスも少なくて済みます。dynaTraceは特許を取得した「PurePath Technology®」により、ユーザーが画面をクリックしてからデータベースのレコードにいたるまで、アプリケーションパフォーマンスに関する詳細かつ正確な全体像を提供します。また、すべてのトランザクションフローを自動検出し、ビジネス関連のグループに自動でマッピングします。さらに、全センサーを自動で設定し、アプリケーションのあらゆる変更に自動で対応するため、手動による設定やメンテナンスは、ほぼ不要です。これは、インフラが動的に変化するクラウド展開においては、特に重要なことです。PurePathは、開発／テスト／本番のすべてのフェーズで、ビジネス上コアとなるアプリケーションを正確かつ詳細に可視化します。そして、その容易な設定と豊富な機能は、次世代APMの基準となるものです。

dynaTraceは、エンドツーエンドのトランザクショントレース、ビジネス上重要なクラウド展開、高度な自動化、容易な使用といったものを実現しており、APMプラットフォームの新しい基準を定義する製品です。主な機能は以下の通りです。

- 次世代のユーザー体感管理**(UEM)**システム：PurePathの対象をブラウザまで拡張することで可能となったこの機能は、UEMとAPMを初めて単一プラットフォームに統合しました。ユーザーのすべての操作を、ブラウザからデータセンターにいたるまで、CDN、広告ネットワーク、マップサービスを含む、すべての階層で監視することができます。UEMは、軽量かつクラウド対応である上に、設定なしで利用でき、すべてのユーザー体感を数分で監視できます。

*図**10**：**UEM**を利用することで、アプリケーションパフォーマンスをユーザーの観点から測定できます。ユーザー体感は、収益、ブランド、顧客満足に影響を与える、非常に重要な要素です。*

- 自動検出と組み込み型のトランザクショントレース：Web階層の動作や負荷分散などに対するより広範な可視化を実現します。簡単な操作により、エンドツーエンドで「事実」を確認できます。情報を統合するための面倒な作業は不要で、時間や労力を浪費することはありません。経験や勘に基づいた作業はなくなり、競合他社よりも優れたユーザー体感を確実に実現できます。

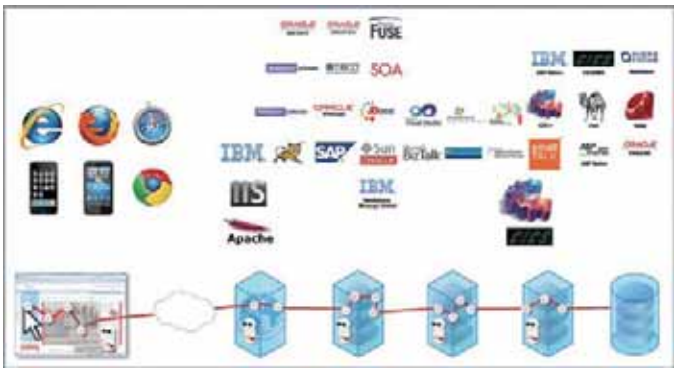

*図**11**：**dynaTrace PurePath Technology**®は、すべてのトランザクションについて、ユーザーのクリックからデータベースのレコードにいたるまで、**24**時間**365**日、コードレベルで情報をキャプチャすることができます。*

- 自動センターにより、アジャイル本番環境と迅速なリリースサイクルを強力にサポート：軽量で設定が不要。卓越した可視化により詳細な情報を提供します。

- 先進的な仮想化およびクラウド環境をより詳細かつ容易に監視：クラウドコレクターや設定不要な自動適応型エージェントを装備しており、JavaScript実行、DOM操作、レンダリング、ネットワーク時間などを詳細に調査することができます。開発部門は、他のどのような方法を使用した場合より10倍も迅速に、問題解決に必要な詳細情報を正確に得ることができます。また、ブラウザエージェントにより、本番環境におけるパフォーマンスや機能の問題に対するトラブルシューティングも可能になります。

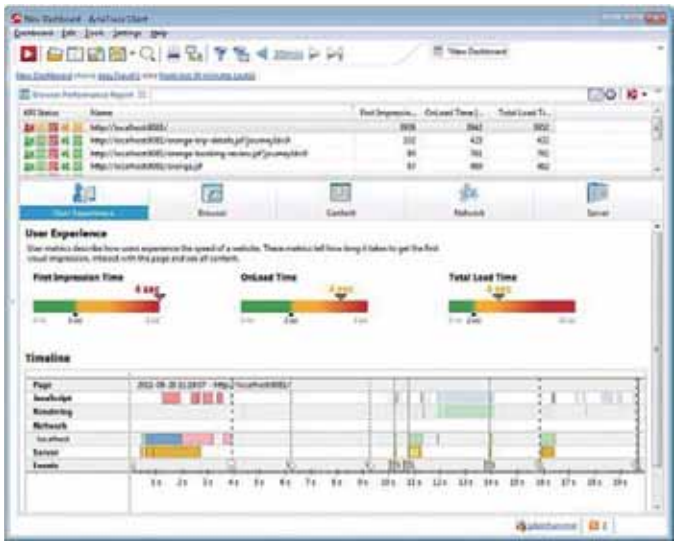

*図**12**：詳細な診断情報が得られるため、優れたエンドユーザー体感を提供できます。*

- スマートトランザクションフローマッピング：データベースセグメンテーションやインフラに関するより多くの情報を得ることにより、大規模な分散環境でも、優先順位づけを迅速かつ正確に行うことができます。

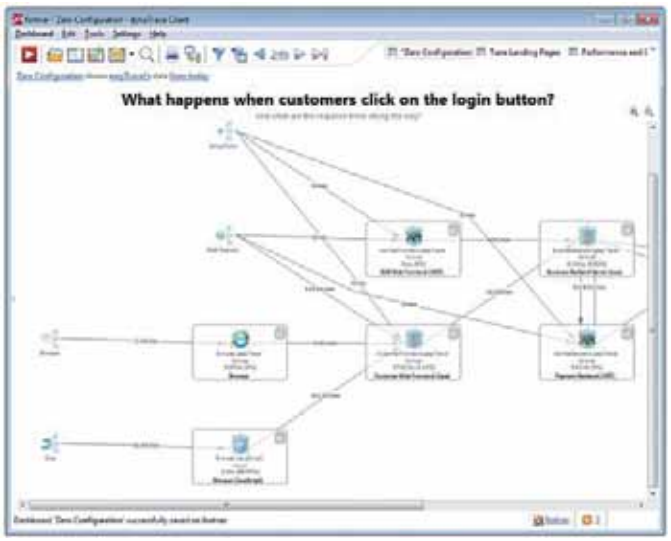

*図**13**：**Transaction Flow Dashboard**は、**24**時間**365**日、異種混在環境において、すべてのユーザーとアプリケーションに関して、エンドツーエンドで（ブラウザからデータベースまで）トランザクションフローを可視化します。*

- インシデント管理とエラーの自動検出：洗練されたイベント監視、オートコレクション、機能問題の特定を行います。dynaTraceはブラウザ側とサーバー側の機能問題を自動検出し、ユーザー体感への影響を明らかにします。ダッシュボードは設定不要で、今週と先週といったような時間軸での比較を自動で行います。クリック1つでダッシュボードを共有し、インタラクティブなレポートとして利用できます。

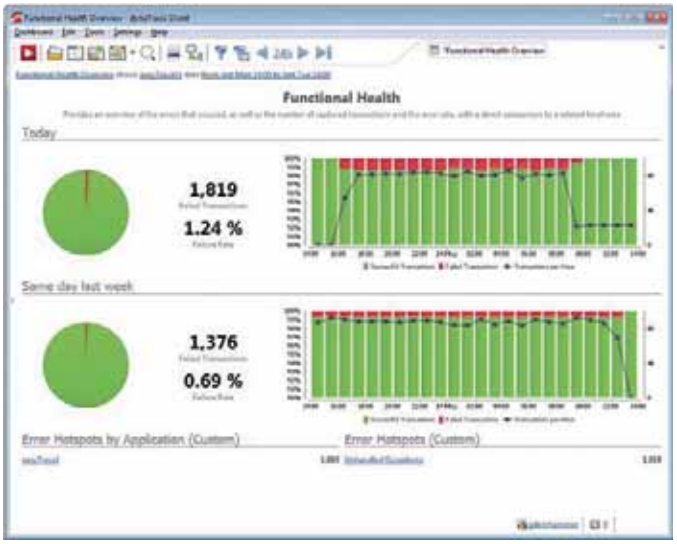

*図14：**Functional Health Dashboard**により、発生したエラーの概要はもちろん、キャプチャしたトランザクションの数、エラーの発生率を確認し、時間軸による比較が可能です。*

- エンタープライズ認証サービス**(LDAP)**とセキュリティ機能：ビジネス上、重要なシステム展開時に利用可能です。

*図15：**Business Dashboard**上に、テナントや目的地毎に予約や収益の概要を表示することも可能です。*

- 分析と**Web**ベースのレポート：パーセントの計算、コンバージョン、比較などにより、アプリケーションの動作をリアルタイムで迅速に把握することができます。

- 詳細診断と先進的な問題分析により、**1**クリックで根本原因を把握：革新的な使いやすさで問題解決へ導きます。次世代APMプラットフォームの基準となるものです。

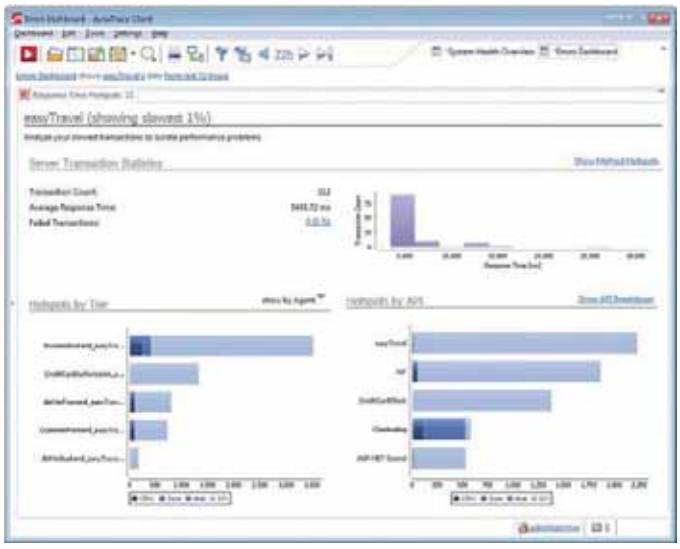

*図16：**Response Time Hotspots**により、最も低速なトランザクションを分析し、パフォーマンス問題の切り分け可能にします。さらに、**1**クリックで根本原因を把握できます。*

部門間の連携を実現する統合ライフサイクル手法

旧来のAPMソリューションベンダーは、複数ツールを集めた「スイート」や、統合を追加するための大規模なプロフェッショナルサービスを提供してきました。それに対し、dynaTraceでは、APMプラットフォームが最初からアプリケーションのライフサイクル全体をサポートするように設計されているため、単一の統合プラットフォームによって、開発／テスト／本番のすべてのフェーズをサポートできます。これにより、時間を浪費する冗長的なタスクを簡単に自動化し、サイロ間（運用部門と開発部門、テスト部門と開発部門、ビジネスユーザーと運用部門など）でビジネスプロセスを改善する環境を提供します。ライフサイクル全体をカバーするには、アプリケーション開発ライフサイクルのコアとなる3つの領域すべてで同一のシステムを実行する必要があります。dynaTraceは、対象となる部門に応じて、「Development Team」、「Test Center」、「Production」の3つの製品を提供していますが、これらは、単一のAPMプラットフォーム基盤を共用しています。それぞれの製品には、設定不要なダッシュボード、統合や設定が不要で自動適応型のツールが同梱されており、開発／テスト／本番のそれぞれについて、迅速にROIを実現できます。

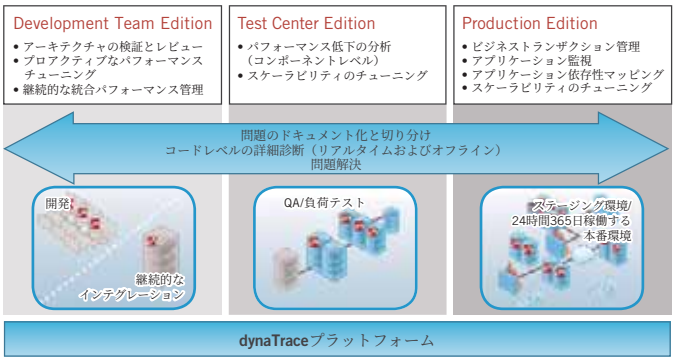

図17：統合ライフサイクルにおける各ブロック

Development Team Edition

Development Team Editionは、リリースするアプリケーションの数や問題発生の予測を向上させることを目標とする部門を対象としています。この製品は、普及しているインテグレーションシステムと統合され、高負荷時のアプリケーション動作を早い段階で可視化します。多くの開発部門では、開発サイクルの初期段階で機能テストが行われますが、パフォーマンスやスケーラビリティのテストを行うことはほとんどありません。dynaTrace Development Team Editionでは、一般的なビルドシステムに対応したプラグインによって、この作業を簡単に行うことができ、また一連の標準ダッシュボードとバージョン比較機能により、早期にROIを達成できます。

図18：パフォーマンス低下の状況をビジュアル化し、パフォーマンステストの結果を理解しやすいものにし、問題への対処がすぐに取れる環境を提供します。

このソリューションを継続的に利用することにより、ビルド単位で、アーキテクチャの欠陥を削減し、開発期間を短縮し、効率的なテストを推進し、リリースまでの期間短縮し、本番システムの品質を向上させることができます。dynaTrace Development Team Editionが実現する、自動化とライフサイクルアプローチにより、統合テストチームと開発者のパフォーマンス作業と間で、継続的なフィードバックが可能になります。これにより、散発的で戦術的なトラブルシューティングから、プロアクティブで戦略的なパフォーマンスエンジニアリングへ移行することができます。この移行は、常に必要とされながらも、これまで達成されなかったものです。アーキテクトは、動的なアプリケーション動作の分析を活用して、アーキテクチャのスケーラビリティを検証し、設計ミスによるコストの増大を避けることができます。

Development Team Editionにより、すべてのコードを社内で作成するかどうかにかかわらず、アーキテクトは自分のアプリケーションアーキテクチャを開発サイクルの初期段階で検証できるようになります。さらに、dynaTrace Development Team Editionでは、アーキテクチャのビューが自動的に生成されるため、変更やパフォーマンスチューニングが行われる毎に、検証を実行できます。本番環境で予期しない問題が発生する可能性は大幅に減少し、より確実なアプリケーション開発を推進できます。

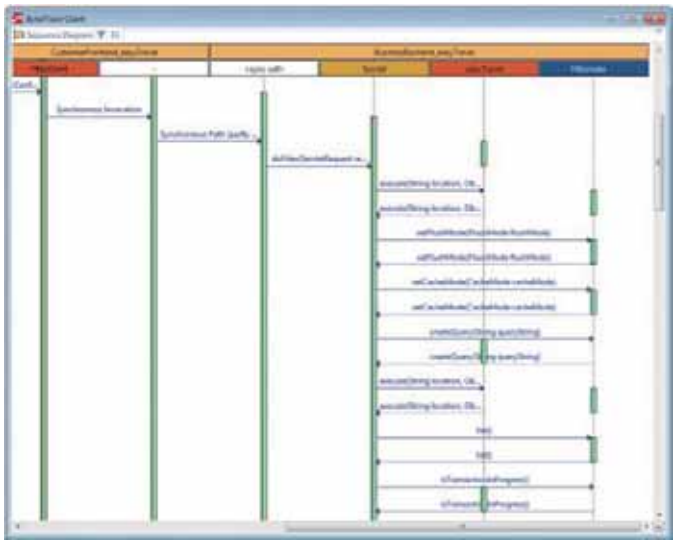

図19：UML図により、物理階層、論理階層、コードの3つのレベルでやり取りを把握することができます。アプリケーションが実際に行うシーケンスをレビューし、相互のやり取り、依存性、パフォーマンスといった側面からビジュアルに表示することが可能です。

Test Center Edition

dynaTrace Test Center Editionにより、テスト時間、テストの繰り返しやサイクルが大幅に減少するだけでなく、どれだけ複雑なアプリケーションでも「ブラックボックス」を開くことができ、テストのカバレージ範囲を広げます。この製品は、時間を浪費する冗長的なタスクや、テストと開発間のコミュニケーションを自動化します。

Test Center Editionは、すべての負荷テストで実行できるように設計されており、負荷テストの結果レポートで特定されたすべての問題についてPurePathデータを提供します。

- テスト担当者による、問題の文書化や関連ログの選別は不要
- 開発者は、正確なトランザクションレベルの記録を取得可能。この記録には、すべての問題箇所に関連するものが含まれているため、経験や勘に基づく作業やテストの再実行は不要で、迅速な診断と改修を実現。外部ベンダーが作成したコードでも、同様のメリットを享受。
- システムは自動的に検出、設定、適用されるため、アーキテクトの支援は不要
- アーキテクトは、すべての問題を解決できるだけでなく、本番環境へのロールアウト前に、高負荷状況下で検証が可能

- PurePathにより、開発者中心のコミュニケーションが可能になり、テストチームと開発チームが地理的に離れていても、迅速なチームワークを実現

Test Center Editionは、Development Team Editionと同様に一般的な負荷テストツール(Compuware Gomez® 360 °Web負荷テスト、HP Loadrunner、Microfocus/Borland SilkPerformer、Apache JMeter、ProxySniffer、PushToTest、BrowserMobなど)や、インシデントトラッキングシステムと統合され、一連の標準ダッシュボードやバージョン比較機能により、早期にROIを実現できます。Test Center Editionを継続的に、すべてのアプリケーションと負荷テストで利用することで、テスト時間の約70%短縮、テストの反復回数の約66%削減、テスト効率とカバレージの大幅な向上を実現することができます。

*図**20**：設定不要ですぐに利用できるダッシュボード*

Production Edition

Production Editionは、本番環境の運用スタッフおよびマネージャー、ビジネスユーザー、パフォーマンス担当者、システムアーキテクトを始めとする一連のステークホルダーを対象としており、問題解決と予防のプロセスについては開発部門も対象としています。本番環境の運用スタッフにとっては、最初は使い慣れた従来の監視ツールと同じように見えます。ダッシュボードの外見は同じで、アラートも同じように発生し、優先順位づけも同じように進行します。しかし、利用するにしたがって、インフラの階層がより多く関連づけされ、より多くのデータ要素が統合され、詳細が正確で、平均値ではないことが分かるようになります。優先順位づけは、より簡単かつ正確にでき、他部門のスタッフに渡される情報は、従来よりも有用なものです。インシデントが発生したときには、データのキャプチャが自動で行われ、選択されているインシデントトラッキングシステムに自動でデータが入力されます。また、トラブルシューティング担当や開発部門へ提供されるデータは、詳細情報を含んだPurePathデータであり、解釈や推測の必要がなくなります。これは、単一のプラットフォームに統合されたシステムの強みであり、分散したツールの組み合わせでは得ることができません。

さらに、Production Editionでは、旧来の監視システムにありがちな多くの手作業が自動化されます。センサーは自動で配置され、学習機能もあるため、アプリケーションの変更に応じて自動で適用を施し、手作業で設定やメンテナンスを行う必要はなくなります。軽量かつ設定不要のエージェントとコレクターが、サーバーコンソールから集中管理され、デフォルトの計測レベルはプリセットされているため、過剰な計測プロセスによる悪影響も減らすことができます。Production Editionには、エンタープライズ管理システムとイベントダッシュボード、インシデントトラッキングシステム、サーバーおよびOSの監視、標準ダッシュボード、といった用途毎に多くのプラグインが同梱されているため、既存の本番環境に簡単に組み入れることができ、各ステークホルダーに対して、最適な方法でROIを実現できます。

*図**21**：**Operations Management Dashboard**により、ホストと監視対象アプリケーションのパフォーマンス(リクエスト、**CPU**、メモリ、ネットワーク負荷、ディスクの利用率、システムの健全性)を単一の画面で把握することができます。*

TCOの削減

今や、より少ないコストで、より多くの成果を生み出すことは、常識となっています。投資の妥当性は、効果を得るまでの期間、収益に対する直近の影響、長期的な**TCO**で評価されます。dynaTrace APMプラットフォームは、どのような部門に対しても、わずか数日間で大きな効果をもたらします。さらに、すべての製品は同じプラットフォームで構築され、単一のシステムに統合されているため、次のようなメリットを享受できます。

- 単一のPurePathデータ、可視化、自動化を共有
- 単一の共同作業システム、ダッシュボード、レポート環境を共有
- 単一のノウハウと技術を共有
- アプリケーションのライフサイクル全体で使用されるツールの種類の減少

dynaTraceをアプリケーションのライフサイクル全体で利用することにより、コストの削減、生産性やチームワークを低下させるサイロの排除、アプリケーションリリースまでの期間短縮、トランザクションの最適化、アプリケーション品質の劇的な向上、といったものを実現することができます。

総括

企業は、アプリケーションパフォーマンス管理の問題を解決するために、様々なツールを購入することがあります。このように、複数のツールを使いこなすのは厄介で、コストも高くなります。利用者は、結局は各部分をつぎはぎする必要があり、ツールやサイロ間の関連づけを手作業で行うことになり、時間を浪費することになります。それぞれの分野にツールを持つという手法だけでは、解決にはなりません。理想的な解決策は、単一プラットフォームで情報を記録し、共通の基準や、共同作業を組み込むアプローチを実現することです。これによって、開発／テスト／本番の全体にわたって、プロアクティブな問題解決と防止の基盤を提供できます。

DevOpsを実現するための新たな要件

要件	説明
ライフサイクルを通じて共有できるプラットフォームの提供	• 「共通言語」により、開発担当が必要とする詳細情報をサポート • アプリケーションを継続的に監視し、詳細な情報をエンドツーエンドで提供。ユーザーが画面をクリックしてからデータベースにいたるまで、すべての階層（ブラウザ、コンテンツ、ネットワーク、サーバー、クラウド）において、すべてのトランザクションのデータを網羅 • アプリケーションのリリースを迅速なものにし、ライフサイクル全体を通じて、適切な共同作業を実現
物理、仮想、クラウド、分散環境で利用可能	• 複数の言語（Java、.NET、C/C++、CICS、IBM WebSphere Message Brokerなど）で開発されたアプリケーションをサポート • 優れたROI • 迅速な実装 • 使いやすさ
プロアクティブな対応	• 信頼性と継続性 • 手戻しや比較作業は不要
迅速な診断	• 24時間365日キャプチャし、問題の再現は不要 • 根本原因を短時間で把握
オンラインでもオフラインでも利用	• 地理的に分散した部門や外部ベンダーをサポート
エンドユーザー視点に基づいたビューの提供	• 優れたユーザー体感の提供 • データセンターからだけでは把握できないシステムの可視化 • データセンターからだけでは把握できないエラーやインシデント（アプリケーションの可用性）の可視化
アプリケーションに内在	• アプリケーションとともに移動 • すべての階層をアプリケーションの視点から把握
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	• データセンター中心から、エンドユーザー中心へ
外部ベンダー作成のコードやサービス（フレームワーク、モジュール、サービス、インフラ）の増加	• すべてを調べ、監視し、シミュレートして根本原因を把握し、問題がある場合は外部ベンダーの協力が必要

お問い合わせ

dynaTraceに関するお問い合わせは（資料請求、デモンストレーション、ご質問等）は、以下までご連絡ください。

TEL：03-5473-4531
Email：marketingjapan@compuware.com

日本コンピュウェア株式会社
東京都港区虎ノ門4-1-20　田中山ビル11F
TEL: 03-5473-4531　FAX: 03-5473-4528