INSTANA
an IBM Company

マイクロサービス時代のソフトウェアの正常性を達成

目次

03 はじめに

04 ソフトウェアの問題のカテゴリー

07 ソフトウェア修復パラダイム

新しく登場した修復方法

従来のソフトウェアの問題の修復パラダイム

ヒントとコツ

ソフトウェアの正常性に関するInstanaの利点

14 Instana、IBM Companyについて

はじめに

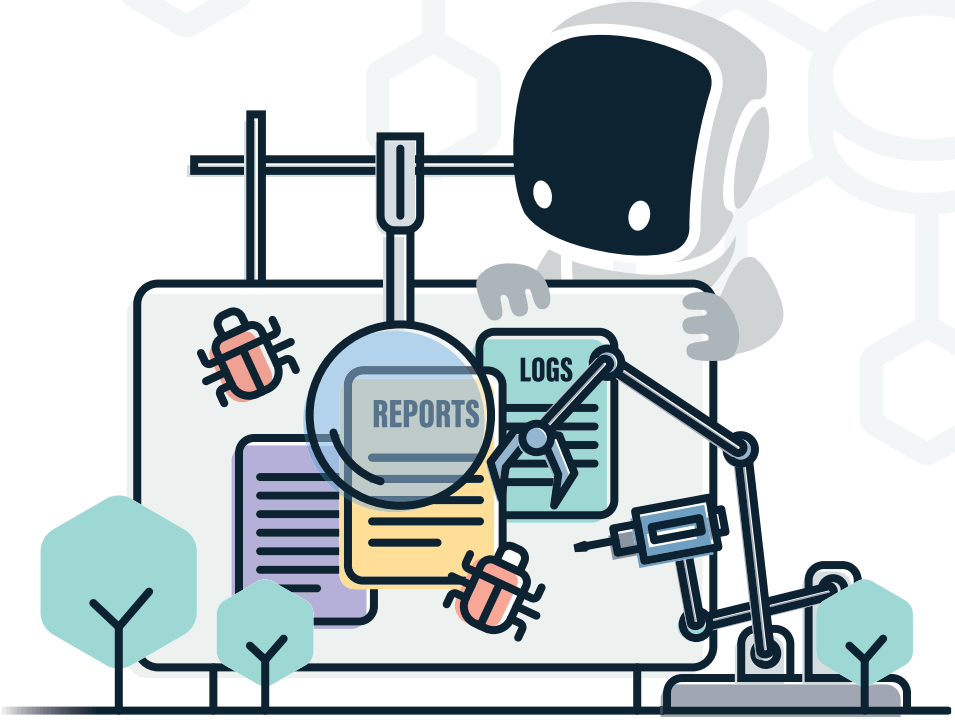
ソフトウェアの正常性保守は、ソフトウェア修復に関連していることが最も一般的です。ソフトウェア・コンテナ、サービス、またはアプリケーションの正常性が低下した場合、企業の直近の関心は、**急速に**ソフトウェアの修復にシフトします。

ソフトウェアの修復は、初めてのソフトウェア・プログラムが作成されて、コードがプログラマーの意図どおりに機能しなかった時点から存在しています。デバッグとして知られる修復プロセスは確立されており、現在でも、正しく動作しないソフトウェア機能を正しく動作させるプロセスです。

デバッグは、予期しない動作を引き起こすソフトウェア・コードの既存のエラーと潜在的なエラーを検知して削除するプロセスです。これは、コード・プロファイラーまたは他のデバッグ・ツールを使用して実行できます。ソフトウェア修復の性質、およびソフトウェアのデバッグに使用できるツールと方法は、時間の経過とともに変化し改善されてきましたが、最終的な目標は同じです。正常に動作しない何かを修復することです。

ソフトウェア（アプリケーション、ツール、ライブラリー、その他のいずれの実装でも）を作成しているすべての企業は、デバッグを使用してソフトウェアの問題を修復します。これらの方法は手動です。コードの問題を解決するのにかかる時間はMTTR（平均修理時間）です。MTTRは、検知（MTTD）、通知（MTTN）、問題を解決するコード修復の開始と検証にかかる合計時間です。これまでの方法であり、これからもずっと使われると思われる方法です。

しかし、良いニュースが入りそうです。ソフトウェアの問題を解決する新しい方法が急速に進化しています。新しい方法では、可能な限り人工知能/機械学習（AI/ML）とAIOpsを使用して、以前は手動のトリアージでしか解決できなかったソフトウェアのパフォーマンスと信頼性の問題を解決します。これらの方法は、自動化された機械的な手順により、手動のトリアージよりもはるかに迅速に多くの問題を解決できるため、より優れた信頼性の高いソフトウェアとアプリケーションを実現していきます。



ソフトウェアの問題のカテゴリー

2つの主要なソフトウェアの問題のカテゴリーがあります。次のとおりです。

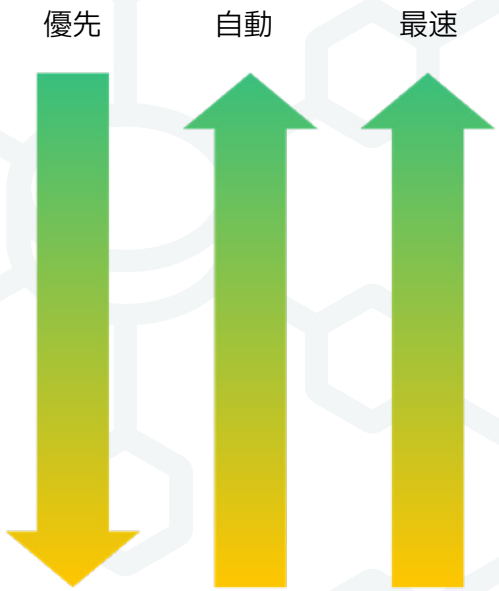
- **運用上の問題**
- **機能上の問題**

運用上の問題は、アプリケーションのコンポーネントが適切かつ期待どおりに動作しているときに、アプリケーションのパフォーマンスに影響するインフラストラクチャーの問題が原因で発生します。これらは、CPU、メモリー、ストレージ、またはネットワーク帯域幅などのリソース不足が原因である可能性があり、アプリケーションのスケーリング中に非常に多く頻発します。システム信頼性エンジニアリング（SRE）によって対処すべきタイプの問題です。

機能上の問題は、一般的に、1つ以上のアプリケーション・コンポーネントに影響を与えるアプリケーション・コードの異常です。これらは、1つのコンポーネントに発生することも、アプリケーションのトランザクションの過程で複数のコンポーネントに連鎖的に発生することもあります。機能上の問題は、コードの問題を解消するために常に手動でトリアージする必要があり、通常はDevOpsチームと開発者が対応します。

マイクロサービス時代のソフトウェア問題の修復領域

問題の検知	問題解決方法
1秒	2秒以内の自動的な防止
1秒	2秒以内から10秒超の自動的な運用手順書の修復
1秒	半自動化されたコード補完手順
1秒	3秒の平均通知時間 - 手動トリアージ



『ソフトウェア問題の修復領域』では、新しいソフトウェア修復オプションや現在利用できるソフトウェア修復オプションの範囲について説明しています。これは、実動および実動前の運用上および機能上のソフトウェアの問題を自動、半自動、または手動で修復するために使用できるさまざまなオプションをまとめたものです。

AIや機械学習（ML）の普及によって、特定の種類のソフトウェアやインフラストラクチャーの問題の防止に至るまでの、ソフトウェアの修復を簡素化および自動化する新しい手段が提供されています。

自動、半自動、または手動の問題修復における最も重要な手段は、正確でリアルタイムのメトリックとトレースです。『修復領域』にリストされている修復のタイプは、これらの測定によって決定されます。測定の時間が少ないほど、修復操作を迅速に開始できます。これらは、ユーザー体験に影響を与える遅延や中断を回避するための自動化されたリソース管理や運用手順書の手順において、特に重要です。

マイクロサービス時代において、メトリックとトレースの集約が遅いと、Cloud DevOpsやSREのイニシアチブの成功は遠くなります。ユーザーや所属する企業に「申し訳ありませんが、可観測性プラットフォームが問題を検知するのに時間がかかりすぎるため、トランザクションに失敗しました」と言いたいDevOpsチームやSREチームは存在しません。

自動化された問題修復の基盤は、侵害の発生しない、精度の高いメトリックとトレースをコンテキストと一緒にリアルタイムで測定および集約する可観測性プラットフォームです。これは、高度に分散されたクラウドとスケーラブル・ベースのマイクロサービス・アーキテクチャーによる問題に対応し、解消するための唯一の選択肢です。

ソフトウェアの正常性を確保するための先進的な可観測性プラットフォームにするには、精度が高く自動化された運用上のソフトウェアの問題の修復**と**機能上のソフトウェアの問題の修復の両方をサポートする必要があります。

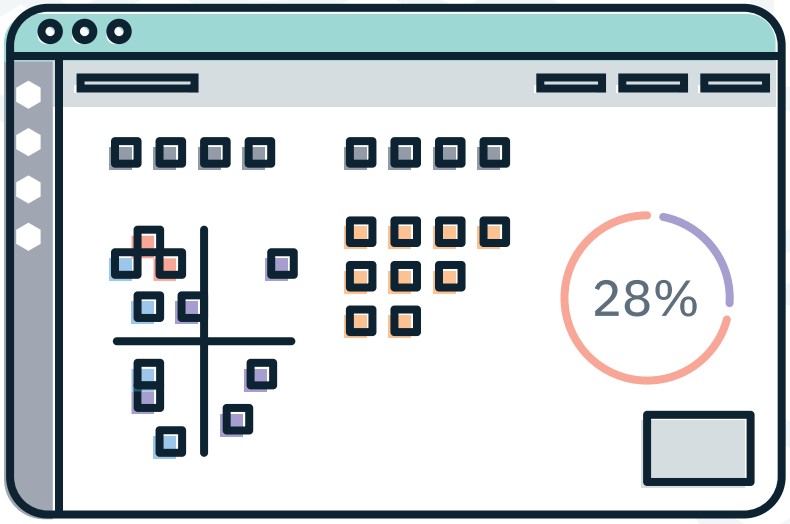
ソフトウェア修復パラダイム

新しく登場した修復方法

可観測性、およびAIやAIOpsは、次世代のソフトウェア正常性テクノロジーの先駆者です。AIやAIOpsによって、ソフトウェア修復の新しい形態は、自動化または半自動化した防止または修復となります。使用する修復手順が完全に自動化されるか、または半自動/アドバイス型になるかの決定は、ソフトウェア・システムが特定の問題に対して常に適切な修復を適用するかについての信頼のレベルに依存します。

自動化したインシデント回避/MTTP

自動化されたインシデント回避は、基礎となるアプリケーション・インフラストラクチャーのアプリケーション・リソースの問題を修復するために使用されます。この問題は、可観測性とアプリケーション・リソース管理（ARM）の組み合わせによって完全かつ確実に処理できます。どれぐらい迅速にシステムがアプリケーション・リソースの問題を防止するアクションを起こせるかを決定する測定値が、平均防止時間（MTTP）です。



MTTPには、自動化されたリソース管理を使用してリアルタイムの修復を確実に行うために、高速かつ極めて精度の高い可観測性メトリックが必要です。ARMは、AIや機械学習を使用して、ユーザーが定義した基準値に基づいてアプリケーションのパフォーマンスを評価し、基準値に達しない場合に発生する問題を解決するためにアクションを起こします。これらの基準値は、パフォーマンス・メトリックやコストになります。

可観測性プラットフォームでパフォーマンス低下が検知された場合、ARMプラットフォームに通知され、しきい値を超えると、問題を修正する手順が自動的に実装されます。例えば、CPU、メモリー、ストレージ、またはその他のリソースの割り当て不足が原因でクラウド・アプリケーションの稼働速度が遅くなった場合、ARMプラットフォームは、割り当てられたリソースを増やしてパフォーマンスの低下を修復します。これは、通常、ユーザーの要求が増加したことに対応するためにアプリケーションがスケールアップするときに発生します。

反対に、ARMプラットフォームは、ユーザーの要求が減少したときにアプリケーションがスケールダウンした後で、アプリケーション・リソースが過剰に割り当てられていることを検知すると、割り当てられたリソースの量を減らして、クラウド基盤のリソースのコストを低減します。

現在、メトリックを収集する最速の速度は1秒であり、この場合にARMが関与し、2秒以下のMTTPを提供できる可能性があります。10秒以上またはサンプリングされたメトリックを提供する低速の可観測性プラットフォームは、効果的なMTTPを提供するための準備として有効ではありません。

なぜこれが重要なのでしょうか。アプリケーションの正常性を維持するための高速な応答が、クラウド基盤のマイクロサービス・アプリケーションにとって今ほど重要になったことはありません。2秒の場合はユーザーは問題に気づきませんが、10から12秒の場合は多くのユーザーが不満を持ちます。マイクロサービスとエンドポイントの広く分散された性質により、アプリケーション・サービス・グリッドの正常性を維持することはかつてないほど複雑になっています。

理想的なサービスのレベルを確保するには、高度に分散されたマイクロサービス環境で増加する問題を迅速に処理するために、インテリジェントな自動化が新たな必須事項となりました。完全に自動化されたMTTPにより、MTTRの時間がゼロになります。多くのマイクロサービス・ベースのアプリケーションを使用する組織では、発生する可能性があるすべての問題に対応するためには、数百人ではないにしても数十人のスキルの高い専門家が必要になるでしょう。

そのため、自動修復への注目が高まっています。ARMのようなAIや機械学習の機能は、リソース管理などの特定のクラスの問題を人間よりも高速かつ効率的に修復できます。また、管理対象の問題による混乱を最小限に抑えるために、最速の可観測性とAIやMLも必要です。

運用手順書の手順

運用手順書には、発生した問題を自動的、半自動的、または手動で解決するために実行する手順と操作がまとめられています。　通常、運用手順書には、システムまたはソフトウェアの開始、停止、監視およびデバッグの手順が含まれています。特別な要求や予測不能な事態に対応する手順も記載されています。実用的な運用手順書によって、オペレーターは、システムを管理したり、トラブルを解決したりできるようになります。さらに、運用手順書を使用することで、新人または若手のDevOpsやSREのチーム・メンバーのオンボーディングで、既存の復旧およびレジリエンシーのポリシーや手順に早く慣れるようにするのにも役立ちます。

運用手順書を自動化すると、これらのプロセスをあらかじめ規定された方法で実行できます。リソース管理や最適化などの特定のプロセスの自動化に加え、運用手順書の結果をユーザーに提示して、次に行うアクションを確認することもできます。また、複数の運用手順書を機械学習とリンクすることによって、対話式のトラブルシューティングとガイド付きまたは自動化された手順を提示することもできます。

運用手順書の自動化は、システムとネットワークの運用プロセスをサポートするワークフローの定義、構築、調整、管理、および報告のプロセスです。運用手順書のワークフローは、コマンド・ライン・インターフェース（CLI）、HTTP RESTおよびSOAP API、SSHセッション、スクリプト、ユーティリティー、コード・ライブラリーなどの多様な通信方式を使用して、アプリケーション、データベース、コンテナ、エンドポイント、ハードウェアなど、あらゆる種類のインフラストラクチャー・エレメントと相互作用できます。

前のセクションで説明したAutomated Resource Management（ARM）の機能は、分散システム・リソースを最適化するための特定の運用手順書のユースケースです。

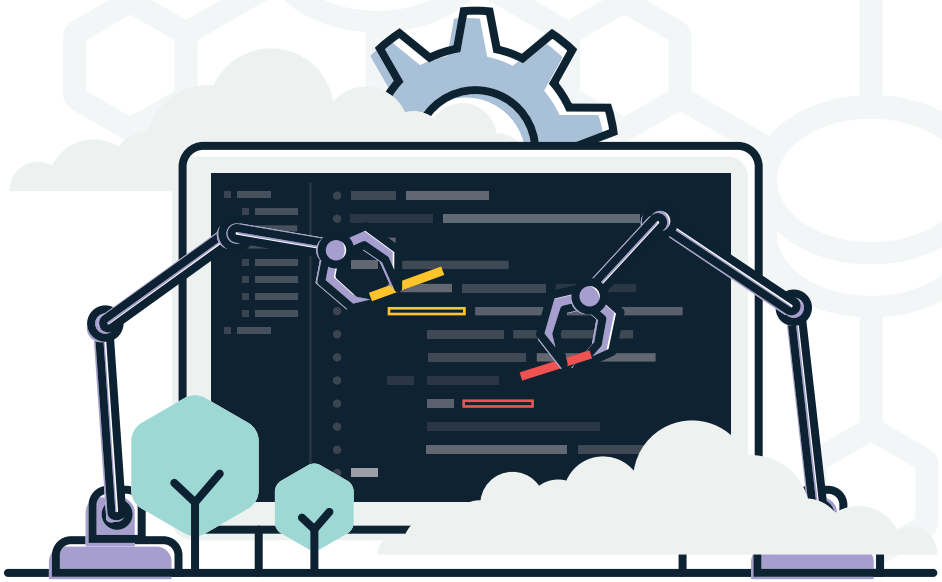
自動の機械学習を搭載したコード補完ツール

新しいソフトウェアの修復と開発の機能として、機械学習を使用したコード補完ツールがあります。これらのツールは、ユーザーがIDE内で入力したコードやコメントに基づいてコードの自動推奨を提供することにより、ソフトウェアの実装を加速します。開発者がサンプル・コード・スニペットを検索してカスタマイズすることなく、全体的な関数とロジック・コード・ブロックを生成できます。

注目すべき2つの新製品があります。GitHub CopilotとAmazon CodeWhisperererです。Asm-Dude、Atom、Captain Stack、GPT-Code-Clippy、Kite、Second Mate、およびYouCompleteMeなどのさまざまなオープンソース製品もあります。

Amazon　CodeWhisperererは、コメントまたは数回のキーストロークのみに基づいて全体的な関数をオートコンプリートできるAIペア・プログラミング・ツールです。CodeWhisperererは、何十億行もの公開されているオープンソース・コードや独自のコードベース、およびパブリック・フォーラムで公開されているドキュメントとコードに基づいています。ソフトウェアのエンジニアは、さまざまなコードの提案から選択し、コメントをオートコンプリートし、それらのコメントに基づいて関数を受け入れることができます。

Microsoft社のGitHub　Copilotは、コードのオートコンプリートによってユーザーを支援するAIを使用します。これは、CodeWhisperererと同様に、数十億行のコードでトレーニングされて、自然言語のプロンプトを数十の言語で表示されるコーディングの候補に変換します。



コード補完ツールは、テストと開発の両方のイニシアチブに適用できるため、コード実装プロセスの加速に役立ちます。コード補完の自動化の簡略化により、コードの候補が提供されますが、適切なコードの機能と安全性を確保するために、候補に関する最終的な決定と統合は担当者に任されています。その点で、コード補完は半自動化されたプロセスです。

総じて、機械学習を搭載したコード補完ツールは、新しいソフトウェア機能の追加または既存のソフトウェア機能の改善を高速化し、多くの場合において、信頼性と安全性を高めます。機械学習を使用したコード補完ツールは、急速に進化し続けており、オープンソースのオプションの数が示すように、ソフトウェアの開発と保守の作業を支援するために、今後も多くの新しい機能が利用できるようになります。

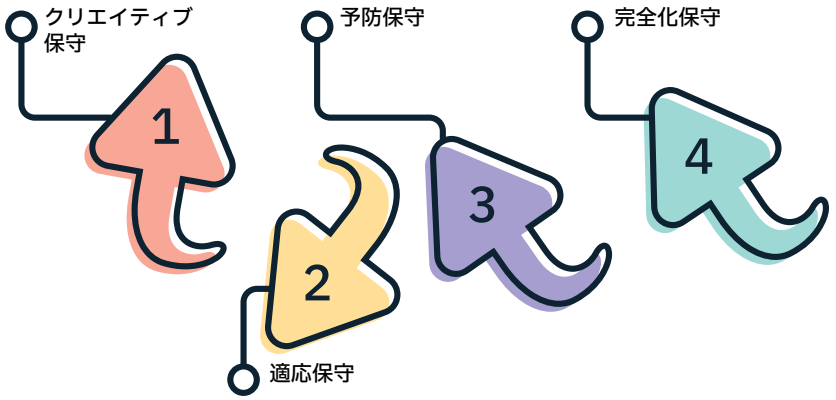
従来のソフトウェアの問題の修復パラダイム

手動修復 - MTTR

手動によるコード修復は、プログラミング言語の誕生以来、ソフトウェアの修復 およびレジリエンシーのための最も一般的な方法です。これは、最も一般的なソフトウェアの修復用語である、MTTR（平均修理時間）とつながっています。

MTTRは、機能不全のソフトウェアをトラブルシューティングしたり、修復したりするために必要な時間の平均を測定する保守メトリックです。

ソフトウェアのMTTRやコード・トリアージ方法は、かなり一般的であるため、ISOとIECによってソフトウェア・ライフサイクル・プロセスとして体系化されています。ISO/IEC14764のカテゴリーは、ソフトウェア保守を次の4つのカテゴリーに分類しています。



- **是正保守：**検知された問題を修正するために、出荷後に実行されるソフトウェア製品の事後対応型の修正です。是正保守は、自動バグ修正を使用して自動で行うことができます。
- **適応保守：**変更された環境または変化している環境で、ソフトウェア製品を使用できるようにし続けるために、出荷後に実行されるソフトウェア製品の修正です。
- **完全化保守：**パフォーマンスまたは保守性を向上させるための、出荷後のソフトウェア製品の修正です。
- **予防保守：**ソフトウェア製品の潜在的な障害が、実際の障害になる前に検知して修正するための、出荷後のソフトウェア製品の修正です。

是正ソフトウェア保守：

是正ソフトウェア保守は、実際にはあらゆる種類のソフトウェア保守です。設計、ロジック、コードなど、ソフトウェアのさまざまな部分に影響を与える可能性があるソフトウェア・アプリケーション内のエラーと障害に対処します。これらの修正は通常、ユーザーまたはお客様からのバグ・レポートによって行われますが 、是正ソフトウェア保守は、お客様よりも先にそれらを発見するのに役立ち、苦情を減らすことができます。是正保守は、最もよく知られている保守タイプであり、一般的にはデバッグと呼ばれるものです。

適応ソフトウェア保守：

適応ソフトウェア保守は、ユーザーのソフトウェア環境が変更した場合に行われます。これは、オペレーティング・システム、ハードウェア、ソフトウェアの依存関係、またはクラウド・ストレージへの変更が原因で行われることがあります。サービスの更新、ベンダーに対する変更、または決済サービス業者の変更を必要とする組織のポリシー変更やルール 変更による場合もあります。

完全化ソフトウェア保守：

完全化ソフトウェア保守は、ソフトウェアの要件や機能が進化して新しい機能や拡張機能が追加された場合に行われます。ユーザーが、アプリケーションを操作するときに、ソフトウェアの一部として必要な新しい機能を提案することがあります。完全化ソフトウェア保守には、ユーザー体験を向上させる新機能の追加と、実用的ではない機能の削除が伴います。完全化ソフトウェア保守の例として、連続するアジャイル・スプリントで実装される漸進的な改善計画 などがあります。

予防ソフトウェア保守：

予防ソフトウェア保守は、平均故障間隔（MTBF）を向上させ、レジリエンシーを高めるために、そしてより長期間にわたって最適に動作できるように、ソフトウェアの変更と適応を行います。このタイプの保守は、連続するアジャイル・スプリントでソフトウェアを継続的に適応や変更する際に、ソフトウェアの機能低下を防ぐために使用されます。これらの手順には、必要に応じてコードの最適化やドキュメントの更新を含めることができるため、ソフトウェアをより安定的で、理解しやすく、保守しやすくできます。これらの変更は、通常、全体的なサービス・レベル目標（SLO）の一部として、ソフトウェアのサービス・レベル指標（SLI）を改善するためのものです。

ISO/IEC 14764定義によって定義されたすべての手動ソフトウェア保守カテゴリーにMTTRの側面があり、それぞれのMTTRの短縮の緊急性は、保守に関する問題の重要性と緊急性に基づきます。いずれの場合も、問題のMTTRを短縮することは、ソフトウェアの機能と運用の両方の正常性において重要です。MTTRの手順により、可観測性のメトリックとトレースから、リアルタイムで修復チームが問題の根本原因に差し向けられるというメリットが得られます。問題の修復に 自動化を使用できる場合、MTTR=0です。

ヒントとコツ

1) ユーザーに影響が及ぶ前に、すべての問題に対処するための堅実なソフトウェアの問題修復計画を構築します。コンテナ化されたマイクロサービスの時代では、ソフトウェアの保守と信頼性が、これまでにないほど複雑になっているため、綿密な計画が必要です。

2) ユーザーのニーズに合ったメソッドを使用します。汎用的な手動の修復戦略に制約されることはなくなり、より高速な修復や問題の防止に役立つさまざまな自動化されたオプションにアクセスできるようになりました。

3) 新しいソフトウェアまたは更新されたソフトウェアを実稼働環境にリリースする前に、実動前のソフトウェアの修復と保守を確認します。これにより予期しない事態を回避できます。

4) 可能な限りソフトウェア修復の自動化を使用して、快適にアプリケーションとインフラストラクチャーの最大限の正常性を確保します。本書で定義されている自動化された手順は、常にオプションであり、担当者が判断します。自動化された問題への対応、コード補完などは、修復担当者によってアクティブ化された場合にのみ呼び出されます。

ソフトウェアの正常性に関するInstanaの利点

Instanaには、ソフトウェアの正常性とレジリエンシーを確保するための重要な可観測性機能があります。Instanaは、1秒のメトリックを提供し、それらを24時間保持する唯一の可観測性プラットフォームです。これらの高い精度とパフォーマンスによって、業界をリードする比類のない運用上の問題および機能上の問題の修復が即座に駆動します。主要なInstanaの属性は次のとおりです。

AI駆動：AIは、Instana可観測性プラットフォームの不可欠な部分です。InstanaのAIは、精度の高いメトリックと完全なトレースに基づいて構築され、スマート・アラート、トレース・コンテキスト、Unbounded Analytics、および継続的な自動検出などの高度な機能を備えています。これらのインテリジェントな機能を使用してMTTRをシンプルにし、AIOpsをエンゲージして、運用上および機能上のソフトウェアの問題を自動または半自動で修復します。

高精度の1秒のメトリック：高精度の1秒のメトリックは、業界をリードする1秒の平均検知時間と3秒の平均通知時間を提供します。Instanaは、これらのメトリックをその細分度で24時間保持します。これにより、自動化されたARMおよび運用手順書の手順よりも決して遅くならない平均防止時間（MTTP）を可能にする最新の測定が確保されます。

完全なエンドツーエンド・トランザクション・トレース：Instanaは、トランザクションごとにサンプリングすることなく、すべてのエンドツーエンドのトレースをマッピングします。つまり、サンプリングによるギャップが発生しません。その情報を使用して、アップストリームとダウンストリームの依存関係をリアルタイムで表示し、問題の根本的な原因を特定します。

自動検出：Instanaは、インストールした瞬間にすべてのアプリケーションとインフレストラクチャーのエレメントを自動的に検出します。その後、Instanaは、アプリケーション・コンポーネント、ノード、コンテナ、およびアーキテクチャー・エンティティーのメトリックとトレースを即座に収集します。

自動コンテキスト：Instanaは、継続的に更新される動的グラフを基盤とするコンテキスト・ガイドを提供します。このグラフは、インフラストラクチャーのすべての物理コンポーネントを追跡し、それらを関連付けて、対応する論理コンポーネントを使用して視覚化します。これは、アプリケーション、サービス、エンドポイント、インフラストラクチャー、またはKubernetesエンティティーの依存関係にナビゲートできるアップストリームまたはダウンストリーム・ボタンを備えています。

アーキテクチャー・モニタリング：Instanaは、アーキテクチャーとアプリケーションを同時に監視することで、アプリケーションがアーキテクチャー・コンポーネントに与える影響と、アーキテクチャーがアプリケーションに与える影響をより詳しく理解できます。正常なソフトウェア運用は、正常なインフラストラクチャーに依存します。Instanaのアーキテクチャー・モニタリングは、自動的なARMと高速なMTTRを可能にする精度の高いメトリックを提供します。

Instanaは、正確で自動化されたソフトウェアの運用上および機能上の問題の修復をリアルタイムでサポートする唯一の可観測性プラットフォームです。
Always Accurate[SM]のインターフェースは、最速の自動修復とMTTRの確保を支援します。

Instana、IBM Companyについて

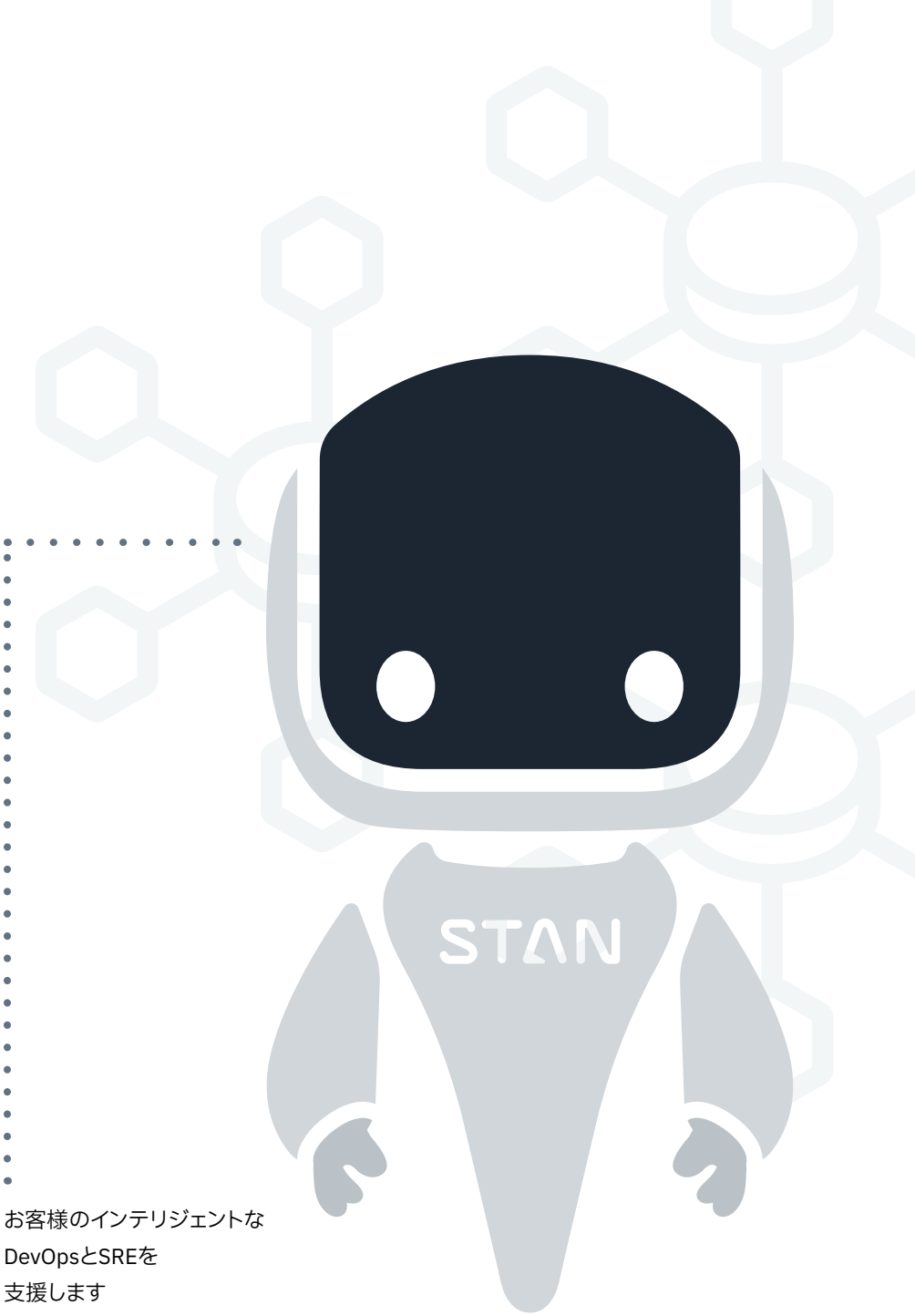
IBMのInstanaでは、モバイル・デバイスやIBM Z®メインフレーム・コンピューターなど、複雑でモダンなクラウドネイティブ・アプリケーションを運用する企業に対し、オンプレミス、パブリックおよびプライベートクラウドなどの常駐場所に関係なく、**自動アプリケーション性能監視**機能を備えた**企業向けの可観測性プラットフォーム（Enterprise Observability Platform）**を提供します。

ハイブリッド・アプリケーション内部の深いコンテキスト依存性をAIで検知するInstanaにより、最新のハイブリッド・アプリケーションを制御します。また、Instanaは開発パイプラインを可視化し、クローズドループのDevOps自動化の実現を支援します。正確で信頼できる可観測性情報でSLI/SLOコンプライアンスを確保するのに役立ちます。

これらの機能は、アプリケーション・パフォーマンスの最適化、イノベーションの実現、リスクの軽減など、お客様が必要とする実用的なフィードバックを提供し、サービス・レベルおよびビジネス・レベルの目標を達成しながら、DevOpsによる効率性の向上とソフトウェア・デリバリー・パイプラインの付加価値向上を支援します。

詳細情報については、**instana.com**を参照してください。

評価版に申し込む



お客様のインテリジェントな
DevOpsとSREを
支援します

INSTANA

an IBM Company