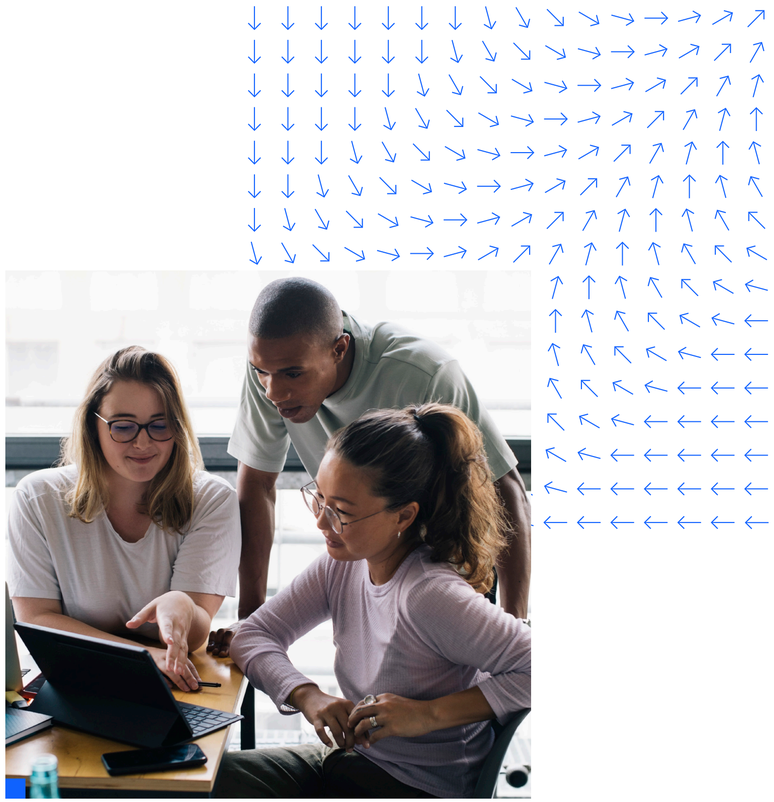
IBM Instana

Beobachtbarkeit für Entwickler

Was versteht man unter Beobachtbarkeit?

Inhalt



01 →
Einführung

02 →
Was ist Beobachtbarkeit?

03 →
Wie lässt sich eine vollkommen lückenlose Beobachtbarkeit erzielen?

04 →
Beobachtbarkeitsstandards und Open-Source

05 →
SLO-Methodik

06 →
Über Open-Source hinaus

07 →
Ist IBM® Instana das Richtige für Sie?

01

Einführung

Entwickler stehen vor einer wachsenden Herausforderung: Wie lassen sich Fehler in Software beheben, die aus vielen unterschiedlichen, voneinander unabhängigen Services besteht, die in einer Vielzahl von Sprachen und Plattformen ausgeführt werden? Wie können wir kritische Änderungen erkennen, Einblick in unsere Black-Box-Dienste erhalten und die wahren Ursachen von Fehlern erkennen?

Vor nicht allzu langer Zeit bedeutete das Debugging eines Programms vor allem eines: das Durchsuchen von Fehlerprotokollen. Dieser Ansatz war kein Problem für kleine Teams, die einfache, lokale Programme in einer kleinen Anzahl von Instanzen nutzten.

Im IT-Bereich finden jedoch laufend Veränderungen statt. Die Softwarearchitekturkonzepte haben sich von Monolithen zu Microservices weiterentwickelt. Die Zuständigkeiten von Entwicklern haben sich mit dem Aufkommen von DevOps verändert, was eine Verlagerung der Verantwortung von Entwicklern für Programme nach deren Fertigstellung bedeutet.

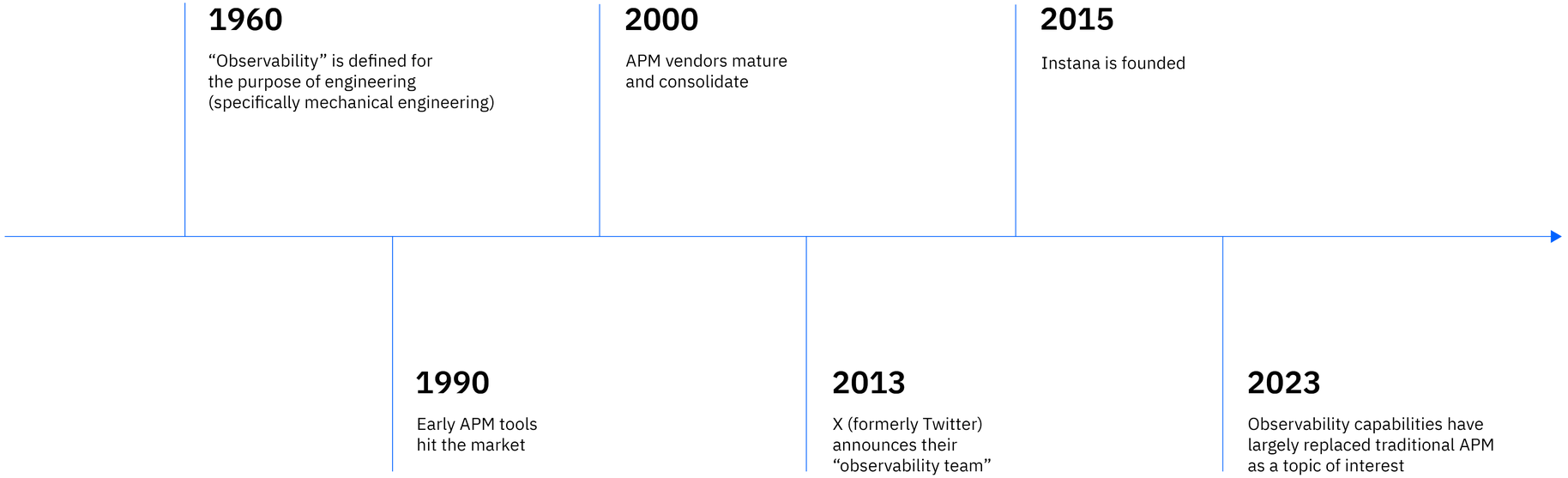
Der Ursprung von Beobachtbarkeit

Der Begriff Beobachtbarkeit wurde erstmals für technische Zwecke in R.E. Kalmans Paper mit dem Titel „On the general theory of control systems“ aus dem Jahr 1960 definiert.[1] Der Begriff Beobachtbarkeit wurde im Maschinenbau als die Fähigkeit, den inneren Zustand eines Systems durch die Messung seines Outputs zu verstehen, definiert.

Fünfzig Jahre später überwachten Entwickler und Softwareexperten ihre Systeme immer noch mit umständlichen Instrumenten – wenn sie überhaupt etwas überwachten. Der Wandel hin zu verteilten Systemen war bereits 2013 im Gange. Damals kündigte X (ehemals Twitter) die Schaffung eines neuen „Beobachtbarkeitsteams“ an, um die Erfassung von Telemetriedaten für die mehreren Hundert Twitter-Services zu zentralisieren und zu standardisieren.[2] Der Begriff „Beobachtbarkeit“ setzt sich durch.

Beobachtbarkeit heute

Mehr als ein Jahrzehnt später werden IT-Teams nach wie vor mit noch detaillierteren Services und funktionsübergreifenderen Aufgaben konfrontiert. Microservices machen den Weg frei für Serverless und DevOps führt zu Site Reliability Engineering (SRE). Beobachtbarkeit hat auch weiterhin eine hohe Bedeutung und der Umfang der Herausforderung wächst exponentiell.



02

Was versteht man unter Beobachtbarkeit?

Von wissenschaftlichen Definitionen abgesehen bedeutet Beobachtbarkeit in der Softwareentwicklung, sich anzusehen, was innerhalb einer Anwendung oder eines Systems vorgeht – und das Beobachtete zu identifizieren. Kurz gesagt: Beobachtbarkeit ist Transparenz plus Verständnis.

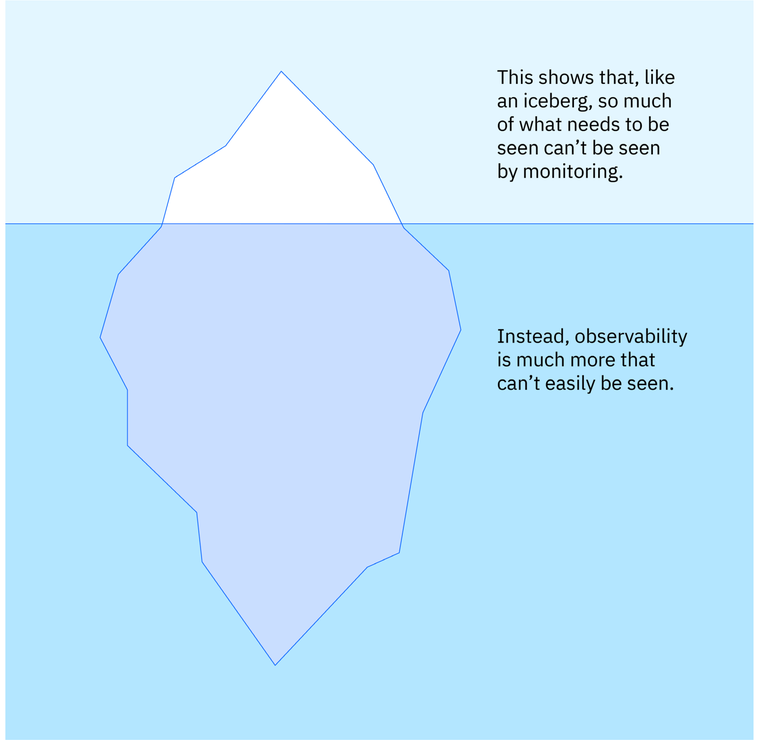
Für moderne verteilte Softwaresysteme sind Beobachtbarkeitstools unverzichtbar. Für Entwickler gibt es einfach keine andere Möglichkeit, die Komplexität, die beim Zerlegen von Anwendungen in viele kleine Teile entsteht, laufend zu überwachen. Glücklicherweise ist es genau diese komplizierte Architektur, die die Existenz von Beobachtbarkeitstools ermöglicht.

Softwaresysteme ermöglichen eine einheitliche Beobachtbarkeit, da die Services in der Regel über eine universelle Steuerungsebene verfügen und über gängige Sprachen wie unter anderem HTTP und RPC miteinander kommunizieren. Darüber hinaus wäre es schwer vorstellbar, sich für einen reibungslosen IT-Betrieb auf veraltete Protokolle zu verlassen.

Wie unterscheidet sich Beobachtbarkeit von Überwachung?
Man kann sich nicht wirklich ein Bild von der Zukunft machen, wenn man keinen Aufschluss über die Vergangenheit hat. Die Überwachung der Anwendungsleistung (APM) umfasst wichtige Funktionen, die neben moderner Beobachtbarkeit existieren.

Traditionell konzentriert sich Überwachung auf die Messung vordefinierter Aspekte bekannter Komponenten. Das war in den 1980er Jahren eine große Idee, aber moderne verteilte Architekturen enthalten Anwendungskomponenten, die sich ständig ändern – manchmal sogar sekündlich. Traditionelle APM-Tools können mit dieser schnelllebigen, dynamischen Umgebung einfach nicht Schritt halten.

Der schlechteste Zeitpunkt, um festzustellen, dass Ihnen eine wichtige Kennzahl fehlt, ist natürlich mitten in der Triage eines Produktionsausfalls. Hier spielt Beobachtbarkeit eine entscheidende Rolle. Statt im Voraus festzulegen, was gemessen werden soll, betrachten Beobachtungswerkzeuge das gesamte Geschehen zwischen und sogar innerhalb von Services. Mit solchen Funktionen können Sie Fragen beantworten, die Sie nicht einmal vorhersehen hätten können – z. B. wenn Sie mit unbekannten Risiken konfrontiert werden.



Überwachung kann nützlich sein – manchmal. Überwachungstools und auf Kennzahlen basierende Alerts eignen sich hervorragend für die Beantwortung grundlegender Fragen wie z. B.:

- Ist mein Service online und für Kunden verfügbar?
- Wie viel Prozent der Anfragen weisen Fehler auf?
- Wie hoch ist die durchschnittliche Latenz einer Anfrage?
- Wie viel Speicher nutzt Redis?
- Muss ich eine lineare Skalierung durchführen?

However, in the course of maintaining healthy applications, we often need to be able to answer deeper questions like these:

- Treten bei Benutzern Fehler entlang geschäftskritischer Pfade auf?
- Warum liegt die Fehlerquote bei dieser bestimmten Teilmenge des Kundenstamms wesentlich höher?
- Wo ist der Engpass im System, der die Latenz für einen bestimmten Endpunkt verursacht?
- Was hat sich in meinem System noch geändert, was diesen Fehler verursachen könnte?
- Welchen Service muss ich aktualisieren, um diesen Fehler zu beheben?

03

Wie erzielt man eine vollständige End-to-End-Beobachtbarkeit?

Die Beantwortung dieser tiefer gehenden Fragen erfordert ein System, das den gesamten Lebenszyklus einer Benutzeranfrage überwacht – vom Client bis zur Persistenzebene. Für Anwendungsentwickler bedeutet dies, Clients und Backends zu instrumentieren, um nützliche Telemetriedaten auszugeben.

Sobald diese Telemetriedaten vorliegen, müssen sie erfasst, verarbeitet, gespeichert und analysiert werden, um die Daten in nützliche, umsetzbare Erkenntnisse zu verwandeln. In der Regel sammelt ein Agent oder Erfasser die Telemetriesignale und leitet sie zur Speicherung an eine Datenbank weiter. Zeitreihen- und Spaltendatenbanken spielen eine sehr wichtige Rolle bei der effizienten Speicherung und Abfrage von Metrikdaten.

Zu guter Letzt ist für Analysen und Erkenntnisse eine Benutzeroberfläche für Datenbankabfragen erforderlich. So können die Daten als Grafiken und Dashboards angezeigt werden. Zudem müssen Alerts konfiguriert werden, um Entwickler schnell zu benachrichtigen, wenn ein Problem auftritt, einschließlich automatisierter Ursachenanalyse und intelligenter Alerts.

Die Telemetriesignale
Dies sind die Datenformate, die zum Erfassen von Informationen aus Anwendungsservices verwendet werden.

- **Traces:** Detaillierte Zeitleisten, die sich jeweils auf eine Anfrage beziehen
- **Ereignisse:** Strukturierte Protokolle verfolgen externe Veränderungen wie Bereitstellungen
- **Metriken:** Aggregierbare Zahlen zu Ereignissen oder Systemen
- **Profile:** Metriken auf Laufzeitebene
- **Protokolle:** Aufzeichnungen von Ereignissen mit Zeitstempel
- **Ausnahmen:** Strukturierte Protokolle für Verfolgungsfehler

Man muss im Hinterkopf behalten, dass Beobachtbarkeit weder ein einzelnes Signal noch eine Reihe von Signalen ist. Das verteilte Tracing stellt jedoch das wohl wichtigste Signal für die Beobachtbarkeit dar.

Was sind verteilte Traces?
Fast jeder Entwickler ist mit Stacktraces vertraut – sie tauchen jeden Tag in Fehlerprotokollen auf. Verteiltes Tracing hingegen findet dann statt, wenn die Funktionsaufrufe eines einzelnen Service durch die Netzwerkaufrufe einer verteilten Anwendung ersetzt werden.

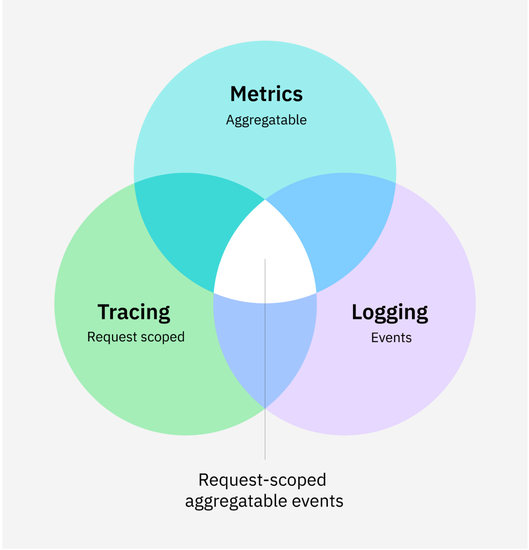
Verteilte Traces bestehen aus „Spans“ – einer Ableitung von Zeitspannen. Jeder Span umfasst eine Startzeit, eine Endzeit, eine beliebige Anzahl benutzerdefinierter Attribute sowie einen Verweis auf den übergeordneten Span.
Wenn ein Service eine Anfrage verarbeitet, erstellt er einen Span, der auf den Span des aufrufenden Services verweist. Dies wird über die Anforderungsheader gemäß der W3C-Trace-Kontextspezifikation empfangen.

Benutzerdefinierte Spans können auch erstellt werden, um der Ablaufverfolgung innerhalb eines bestimmten Services mehr Granularität zu verleihen. Wenn es beispielsweise eine Funktion gibt, die nach dem Empfangen von Ergebnissen aus einer Datenbank ein hohes Maß an Verarbeitung leistet, kann es hilfreich sein, diese Funktion in einen benutzerdefinierten Span einzuschließen.

Traces sind aber mehr als nur Zeitleisten.
Es geht auch um viel mehr als nur Latenz.
Durch die Verfolgung des Pfades einzelner Anfragen können Sie Protokolle, Metriken und andere Signale so kontextualisieren, dass sie zur Beantwortung von Fragen aus einer nutzerzentrierten (anfragebezogenen) Sicht beitragen. Traces enthalten eine Datenstruktur, die Sie bei verschiedenen Punkten unterstützt:

- Verständnis von Anforderungsabläufen im gesamten System.
- Sofortige Visualisierung Ihrer Systemtopologie.
- Ableitung von Metriken aus der Fülle der Trace-Metadaten.
- Anreicherung von Logs um Kontext durch das Anhängen an einen bestimmten Span.

Zusammenspiel
Auch wenn diese Signale manchmal isoliert nützlich sind, können Sie die wertvollsten Antworten erhalten, wenn alle diese Daten auf sinnvolle Weise korreliert und durchsucht werden können. Der bekannte Software-Engineer Peter Bourgon hat elegant den entscheidenden idealen Punkt für Informationen dargestellt, an dem sich Metriken, Traces und Protokolle überschneiden.[3] Das Ergebnis sind anfragebezogene, aggregierbare Ereignisse.



04

Beobachtbarkeitsstandards und Open Source

Im Jahr 2019 schlossen sich die Projekte OpenTracing und OpenCensus unter dem Namen OpenTelemetry zusammen. Dieses Open-Source-Projekt der Cloud Native Computing Foundation (CNCF) etabliert rasch offene Standards für Telemetrie, Beobachtbarkeitsplattformen, die von Anbietern wie IBM® Instana weitgehend übernommen werden.

Das OpenTelemetry-Projekt ist in drei Bereichen tätig, um die Erfassung von Telemetriedaten zu vereinfachen: Spezifikationen und Standards, Instrumentierungstools und Pipelinetools.

Zu den Spezifikationen gehören:

- OTLP – ein Binärformat zur effizienten Darstellung von Metriken, Traces und Protokollen.
- Die W3C-Trace-Kontextspezifikation zur Weitergabe einer übergeordneten traceId an nachgeschaltete Services.
- Sprach-APIs, die Sie von Ihrer Anwendungs- oder Ihrem Bibliothekscode aus aufrufen können.

Zu den Instrumentierungstools gehören:

- – Sprach-SDKs, die die Sprach-APIs mit bestimmten Implementierungen verbinden. Dabei kann es sich entweder so genannte Standard-SDKs
- oder um Plug-ins handeln, die von einer Community oder einem Anbieter bereitgestellt und für die Verarbeitung und den Export von Telemetrie verwendet werden.
- – Bibliotheken für eine automatische Instrumentierung, die automatisch Sprach-APIs aufrufen, wenn relevante Ereignisse in Ihrem Framework oder Ihrer Bibliothek auftreten.

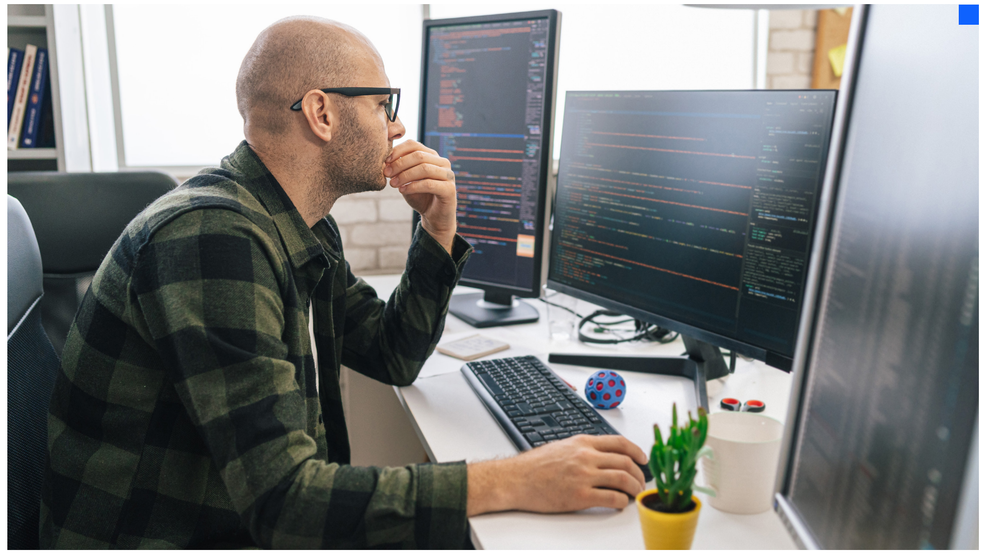
Zu guter Letzt gibt es für die Verarbeitung und Übertragung den OpenTelemetry Collector. Dieses umfangreiche Tool kann als Agent auf Ihrem Knoten fungieren, um alle Telemetriedaten zu erfassen und weiterzuleiten. Darüber hinaus gibt es ein großes Ökosystem an Empfängern, Verarbeitern und Exportern, die als Plug-ins verfügbar sind.

Open-Source-Beobachtbarkeitsbackends
Das OpenTelemetry-Projekt bietet zwar kein Backend zum Speichern und Analysieren Ihrer Überwachungsdaten, aber eine Reihe von Open-Source-Tools sind mit den OTLP-Signalen kompatibel.

Im OpenTelemetry Demo-Projekt finden Sie ein Beispiel für eine Microservices-Anwendung, die in Docker oder Kubernetes ausgeführt werden kann. Darin enthalten sind zwei Open-Source-Telemetriedatenbanken: Prometheus für Metriken und Protokolle und Jaeger für Traces. Ebenso enthalten ist Grafana zur Visualisierung einiger Metriken aus Prometheus.

05

SLO-Methodik



Inzwischen haben Sie wahrscheinlich viel über Service Level Objectives (SLOs) und Service Level Indicators (SLIs) gehört.

Die SLO-Methodik stellt eine neue Denkweise zu Softwareleistung und -integrität dar, die mit Beobachtbarkeit Hand in Hand geht. Beobachtbarkeit als Konzept stammt aus der Kontrolltheorie, die sich auf den optimalen – nicht den perfekten – Betrieb von Maschinen konzentriert. Der SLO-Methodik zufolge erzielen IT-Teams, die nach Perfektion streben, oft schlechtere Ergebnisse, als wenn sie sich realistische Ziele setzen.

Alles beginnt mit dem Service-Level-Indikator. Ein SLI ist eine beliebige Metrik oder Statistik, die in einen Prozentsatz konvertiert werden kann. Im Zusammenhang mit der Ausführung von Software sind SLIs Dinge wie der Prozentsatz erfolgreich bearbeiteter Anfragen oder der Prozentsatz der Anfragen mit akzeptabler Latenz.

Ein SLO ist ein Ziel, das an einen SLI gebunden ist. SLOs werden häufig als bestimmte „Neunerzahl“ ausgedrückt.Beispielsweise würden 4 Neunen bedeuten, dass ein SLI das Ziel in 99,99 % der Fälle erreicht. Ganz wichtig: SLOs sollten niemals „100 %“ sein. Dies ist in nahezu jeder Situation unrealistisch. Planen Sie am besten ein Fehlerbudget ein – etwas Spielraum für geplante und ungeplante Ausfälle.

Tatsächlich fand ein Team bei Google heraus, dass die allgemeine Systemzuverlässigkeit erhöht werden konnte, indem künstlich Ausfallzeiten für den Service herbeigeführt wurden, damit andere Teams keine 100%-ige Zuverlässigkeit erwarteten.[3] Es wird nicht empfohlen, dies in Ihrem Unternehmen nachzuahmen.

Service Level Agreements (SLAs) ähneln auf den ersten Blick SLOs, werden jedoch für sehr unterschiedliche Zwecke verwendet. Ein SLA ist Teil eines Geschäftsvertrags und legt fest, was passiert, wenn das Ziel verletzt wird – in der Regel eine Geldstrafe.

SLAs sind für Entwickler erst dann interessant, wenn sie verletzt werden. SLOs sind für Entwickler äußerst nützlich, da sie ihnen Aufschluss die allgemeine Zuverlässigkeit von Anwendungen geben und ihnen dabei helfen zu ermitteln, ob sich eine Investition in neue Funktionen lohnt.

06

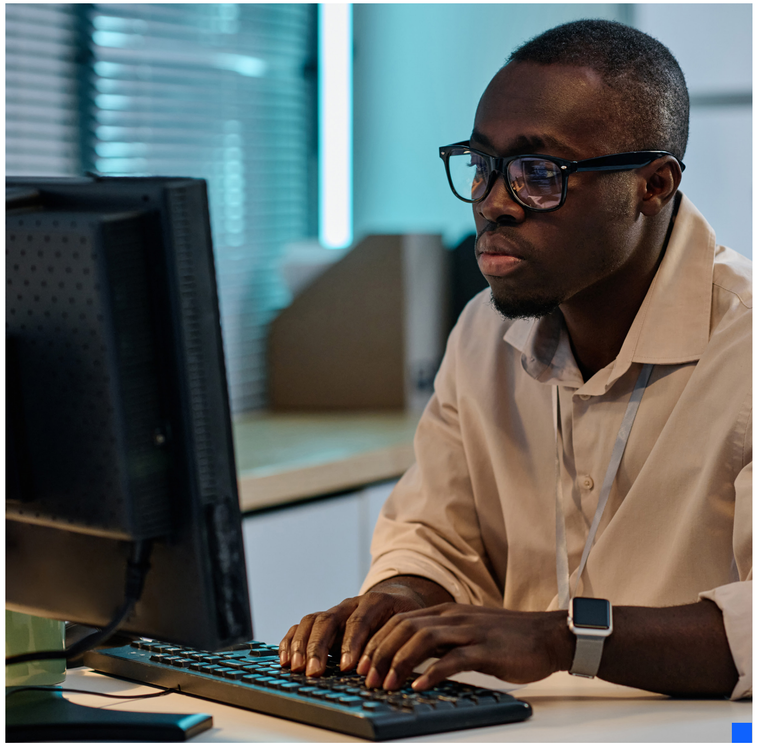
Über Open-Source hinaus

Mit den Open-Source-Tools aus dem vorherigen Kapitel können Sie viele Telemetriedaten von Services sammeln und damit beginnen, sie auf nützliche Art und Weise zu kombinieren. Damit alleine haben Sie jedoch noch keine echte Beobachtbarkeit erzielt.

Für Beobachtbarkeit müssen Sie Fragen zu unbekannten Unbekannten beantworten können. Ihre Telemetrie sollte in der Lage sein, sich anzupassen und sich mit Ihren Services zu ändern. Bislang kann kein Open-Source-Tool diesen Grad an Automatisierung bieten.
Eine Lösung wie IBM® Instana ist jedoch hierzu in der Lage.

Beobachtbarkeitslösungen sind anpassungsfähig und dynamisch
Ihre Anwendungen bestehen wahrscheinlich aus Dutzenden, Hunderten oder sogar Tausenden von Services, die verschiedene Sprachen und Technologien nutzen.Die Aufrechterhaltung einer konsistenten manuellen Instrumentierung auf der gesamten Oberfläche der Anwendung wäre da nahezu unmöglich.

Durch Automatisierungen wie dynamische Serviceerkennung und automatische Instrumentierung können Sie ein wesentlich umfassenderes Verständnis Ihrer Systeme erzielen, bevor sich Vorfälle ereignen. Das liegt daran, dass man schlicht und einfach nicht während der Triage eines Produktionsausfalls feststellen möchte, dass ein wichtiges Puzzleteil fehlt.

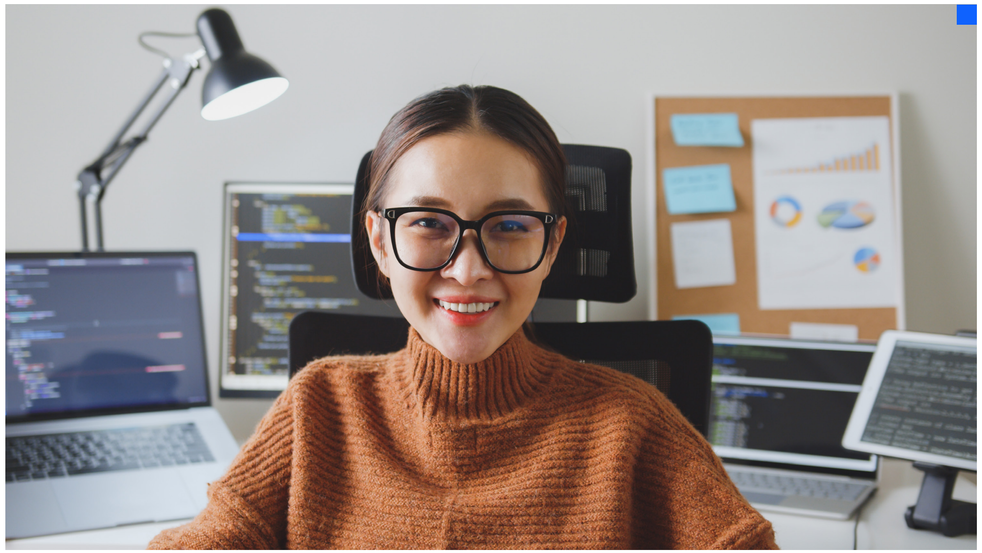


Der automatische Korrelationsvorteil
Ein wesentlicher Vorteil einer Unternehmensbeobachtbarkeitslösung liegt in der Möglichkeit, Metriken und Traces zu Maschine, Infrastruktur und Anwendung oder Services zu korrelieren. Verteilte Traces geben Aufschluss über den Ablauf einer Anfrage und Metriken liefern die erforderlichen Leistungspunkte dazu. All das manuell zu korrelieren ist aber ziemlich umständlich. Der Hauptgrund dafür, dass man mehrere Dashboards für verschiedene Services hat, besteht darin, dass es nahezu unmöglich sein kann, Daten zuzuordnen und den Gesamtkontext des Problems zu ermitteln.

Der größte Vorteil einer automatischen Korrelation ist die unmittelbare Verfügbarkeit von Erkenntnissen. Beim Betrachten eines Problems oder Vorfalls erledigt die Lösung die gesamte Detektivarbeit. Dies bietet wichtige Informationen als kontextbezogene Belege und führt Sie direkt zu dem Bereich, der für Sie von Interesse ist.

07

Ist IBM® Instana das Richtige für Sie?



IBM® Instana ist eine Beobachtbarkeitsplattform für Unternehmen, die Funktionen zur automatisierten Anwendungsleistungsüberwachung umfasst. Sie wurde entwickelt für Unternehmen, die komplexe, moderne, cloudnative Anwendungen unabhängig davon, wo sich diese befinden, betreiben – On-Premises, in Public und Private Clouds, auf Mobilgeräten oder in einer IBM® Z Umgebung.

IBM® Instana gibt Ihnen die Kontrolle über Ihre modernen Hybridanwendungen mit KI-gestützter Erkennung tiefgehender, kontextbezogener Abhängigkeiten in Hybridanwendungen. Außerdem bietet IBM® Instana Einblick in den gesamten Entwicklungsprozess und unterstützt so die Umsetzung geschlossener DevOps-Automatisierungszyklen.

Diese Funktionalität liefert Kunden das erforderliche Feedback, um Anwendungsleistung zu optimieren, Innovationen zu ermöglichen und Risiken zu minimieren. Diese Merkmale helfen DevOps-Teams dabei, die Effizienz zu steigern, einen höheren Mehrwert innerhalb von Software Delivery Pipelines zu schaffen und gleichzeitig ihren Zielen auf Service- und Unternehmensebene gerecht zu werden.

Erleben Sie die Schlagkraft von IBM® Instana selbst. Registrieren Sie sich jetzt für eine kostenlose 14-tägige Testversion der Vollversion des Produkts. Es ist keine Kreditkarte erforderlich.

Kostenlose Testversion von IBM® Instana →

IBM® Instana kennenlernen →

1. R. E. Kalman, „On the general theory of control systems“, August 1960.

2. Twitter-Blog, „Observability at Twitter“, 9. September 2013, abgerufen im Juli 2023.

3. Google Online-SRE-Buch, „Service Level Objectives“, abgerufen im Juli 2023.

IBM Deutschland GmbH
IBM-Allee 1
71139 Ehningen
ibm.com/de
IBM Corporation
New Orchard Road
Armonk, NY 10504

Hergestellt in den Vereinigten Staaten von Amerika, August 2023