Oracle Business Transaction Management (OBTM) in der Praxis

Marcus Schröder
Oracle Deutschland B.V. & Co. KG
Nürnberg

Einleitung oder „Warum Business Transaction Monitoring?“
Das technische und fachliche Monitoring von Anwendungs-Infrastrukturen stellt den IT-Betrieb vor immer größere Herausforderungen. Der Grund hierfür liegt einerseits in der steigenden Komplexität und andererseits in den steigenden Anforderungen der Fachabteilungen/Endbenutzer.
Die Erhöhung der Komplexität kommt oft durch einen hohen Integrationsgrad und eine Verteilung von Anwendungskomponenten. Moderne Anwendungs-Architekturen beziehen ihre Daten meist aus unterschiedlichen Sub-Systemen, die innerhalb oder außerhalb des eigenen Rechenzentrums betrieben werden.
Die enge Kopplung von Business und IT erhöht die Anforderungen der erwarteten Service Level Agreements. In einem modernen Unternehmen ist der störungsfreie Ablauf der Geschäftsprozesse sehr eng an die unterliegenden IT-Systeme gebunden. Ausfälle von Prozesskomponenten sind mit sehr hohen Kosten verbunden. Zusätzlich und schwer in Zahlen zu fassen sind interne/externe Imageschäden, die ein Ausfall von Prozesskomponenten nach sich zieht.

Die Erhöhung der Servicequalität durch Hochverfügbarkeit verringert oft die Anzahl von Störungen, diese sind jedoch grundsätzlich nicht ganz auszuschließen. Einerseits sind hoch integrierte und komplexe Systeme durch ihre Architektur fehleranfälliger, auf der anderen Seite gibt es auch fachliche Fehler, die einen Prozess stören können und nicht durch Überwachung von Metriken zu entdecken sind.

Resultierend aus den vorangegangenen Thesen wird ein Monitoring System benötigt, das es ermöglicht, sowohl fachliche als auch technische Fehler innerhalb eines IT-unterstützten Businessprozesses zu finden. Das System muss in der Lage sein, auch bei verteilten und integrierten Systemen alle beteiligten Prozessschritte zu überwachen.

Oracle Business Transaction Monitoring (OBTM)
OBTM ist ein Mitglied der Oracle Enterprise Management-Familie und füllt die Lücke zwischen metrikbasiertem System Monitoring und rein fachlichem Prozess Monitoring wie zum Beispiel durch Oracle Business Activity Monitoring (BAM). Das Produkt OBTM wurde durch die Übernahme der kalifornischen Firma Amberpoint im Frühjahr 2010 in das Oracle Produkt-Portfolio integriert.
Die Monitoring-Funktionalität von OBTM erstreckt sich über drei Bereiche: Service Level Monitoring, Fehler-Diagnose und Transaktionsverfolgung.
Das Service Level Monitoring basiert auf Messen von reellen Prozessdurchläufen (Anzahl) und Laufzeiten. Die gewonnenen Informationen können im System verarbeitet werden oder in ein gängiges XML-Format exportiert werden.
Die Fehlerdiagnose ermöglicht das schnelle Auffinden von Ausnahmen, die durch Laufzeitprobleme und/oder Programmfehler verursacht werden.

Die Transaktionsverfolgung ermöglicht die detaillierte Einsicht und Verfolgung einzelner Prozesse und die Einsicht in die vorhandenen Prozessdaten. Die Transaktionsverfolgung ermöglicht das Suchen von bestimmten Dateninhalten (z. B. Kundenname, Transaktionsnummer etc.), die es ermöglichen eine bestimmte einzelne Transaktion zu finden und ihren „Weg“ durch die verteilte Anwendung nachzuvollziehen. Dieses „Feature“ erleichtert die Suche nach Fehlern innerhalb einer Transaktion. Diese Art von Fehlern müssen ohne OBTM in Untersuchung von Log-Dateien gefunden werden. Dieses Vorgehen ist äußerst zeitaufwendig und blockt wertvolle Ressourcen, da für ein zielgerichtetes Debugging oft Entwickler/Applikations-Kenner hinzugezogen werden müssen.

Technischer Hintergrund
Interessant ist die technische Implementierung dieser Lösung und der Frage: „Wie ist es möglich an die detaillierten Transaktionsdaten zu kommen?“ OBTM ist in mehrere Architekturkomponenten implementiert. Man unterscheidet die Schichten: Observer, Monitor/Agent, Sphere, Performance Monitor, Transaction Monitor und die dazugehörigen Database Repositories.

Der **Observer** „unterbricht“ Nachrichten zwischen Server Provider und Service Consumer. Der Observer generiert eine Kopie des Nachrichtinhalts und erzeugt die dazugehörigen Metadaten (wie z. B. Service Name, Timestamp, Laufzeit, Servicename etc.). Da OBTM historisch aus dem SOA Service Registry Bereich entwickelt wurde, wird aus den Nachrichten und Metadaten eine WSDL generiert, die an den weiter verarbeitenden Agenten/Monitor übertragen wird.
Der Observer verwendet zwei sogenannte „Interception“-Modelle, um an die Nachrichteninhalte zu kommen, das Pipeline- und das Injection-Modell. Das Pipeline-Modell verwendet Java Schnittstellen (JAX-WS etc.), um an die Informationen zu kommen, das Injection Model verwendet AOP also Byte Code Injection.
Java Byte Code Injection ermöglicht das Auslesen des Bytecodes während der Programmausführung. Während des Ladens des Programmcodes in die Java Laufzeitumgebung wird der Byte Code mit zusätzlichen Programmteilen angereichert, die es ermöglichen die gewünschten Informationen auszulesen. Voraussetzung hierfür ist natürlich, dass die Daten nicht verschlüsselt sind.
Der Observer wird als Java Library in den Klassenpfad des JEE Servers (primär WebLogic Server) eingebunden und während des Startens geladen. Dies setzt voraus, dass bei der Implementierung des Observers ein Neustart des JEE Servers durchgeführt wird.
Es gibt verschiedene Observer-Typen für unterschiedliche Anwendungsfälle. Es gibt die Gruppe der Web Service Observer für JAX-RPC, JAX-WS, WCF und ASMX. Die Gruppe der generischen Java Observer wie EJB, JDBC, JMS, RMI und POJO und spezifische Observer für bestimmte Middleware-Produkte wie Oracle Service Bus, Biztalk, Tibco Business Works etc.

Die vom Observer generierte WSDL wird mittels Socket-Connection oder Datei an den **Monitor** gesendet. Der Monitor verarbeitet die WSDL und wendet zum Beispiel Richtlinien für SLM Alerts auf die enthalten Daten an. Der Monitor ist in der Lage zwischen 50 und 100 Services zu verarbeiten, die genaue Anzahl ist abhängig von der Größe des Payloads (also der enthaltenden Daten, die zu verarbeiten sind). Für Systeme, die größere Mengen von Daten zu verarbeiten haben, ist es möglich mehrere Monitore für einen Observer zu verwenden. Ab der OBTM-Version 11g ist es möglich die angeforderten Daten bereits im Observer zu filtern, um so die an den Monitor zu übergebene Datenmenge zu reduzieren. Zum Beispiel besteht die Möglichkeit nur die JDBC, EJB und POJO Services zu überwachen und so die anderen Objekte herauszufiltern. Durch diesen Ansatz kommt es zu einer relevanten Reduktion der bereitgestellten Daten, die der Monitor zu verarbeiten hat.
Die Implementierung des Monitors erfolgt durch ein Java Programm, das auf einem JEE Server bereitgestellt wird. Die Ablage der Daten erfolgt mittels JDBC in eine Oracle-Datenbank und nicht in ein Filesystem.

Die **Sphere** ist die Benutzerschnittstelle von OBTM, in der Sphere erfolgt das Discovery (die Erhebung aller Daten, die an dem Ablauf beteiligt sind) der beteiligten Servicekomponenten und die Zuordnung in einem Prozessablauf. Das Prozessmodell wird grafisch angezeigt und kann an die Anforderung angepasst werden. Das Prozess Discovery erfolgt automatisch für alle erkannten Prozess-Schritte. Es stellt sich in der Praxis so dar, dass Services teilweise erst angezeigt werden, wenn diese aufgerufen wurden. Die Zuordnung zwischen Services unterschiedlicher Systeme (zum Beispiel auf verteilten Applikations-Server) erfolgt manuell, um diese Zuordnung persistent zu halten, wird eine eindeutige Kennung benötigt. Dies kann zum Beispiel eine Kunden-ID oder Bearbeitungsnummer sein. Mit Hilfe dieses Parameters ist OBTM in der Lage auch bei stark verteilten Anwendungen den Ablaufkontext zu erhalten. Eine weitere Aufgabe der Sphere ist die Erstellung von Policies oder Richtlinien. Policies werden vom Benutzer in der grafischen Benutzeroberfläche erstellt und dann automatisch an die Monitore weitergeleitet. Die Policies können sich auf die Laufzeitdaten von Services beziehen oder auf Transaktionsinhalte (z. B. Gesamtbetrag einer Onlinebestellung, Menge eines bestellten Artikels etc.). Die erstellten Policies generieren einen Event der auch dazu verwendet werden kann eine nachfolgende Aktion auszulösen.
Die Bereitstellung der Sphere erfolgt als Java Programm auf einem Applikations-Server. Alle anfallenden Meta-Daten bezüglich Policies und Discovery werden über eine JDBC-Verbindung in einer Oracle-Datenbank persistiert. Die Aufbewahrungsdauer der Daten hängt von der Größe der verwendeten Datenbank ab und kann auf die Bedürfnisse der Administratoren angepasst werden.

Der **Performance Monitor** und **Transaction Monitor** verarbeiten die Laufzeit- und Transaktionsmetriken aus dem/den Monitor/en. Alle Informationen werden von den Performance und Transaction Monitoren ausgewertet und mittels SOAP über HTTP(S) an die Sphere übertragen. Die Sphere speichert die gewonnen Daten in der Repository-Datenbank und ermöglicht den Zugriff auf historische Transaktions- und Performancedaten.
Die Implementierung des Performance und Transaction Monitors als Java Applikation ermöglicht eine architektonische Anpassung des Systems auf die Performanceanforderung. Performance und Transaction Monitor können z. B. auf einem Applikations-Server deployed werden oder getrennt auf zwei Server verteilt werden.

Eine typische Implementierung einer OBTM-Lösung könnte folgendermaßen aussehen:

Anforderung: Der zu überwachende Businessprozess durchläuft mehrere Systeme, die stark verteilt auf mehrere Sub-Systeme implementiert wurden.

Implementierungen: Alle an dem Businessprozess beteiligten Komponenten, die eine zentrale Rolle in diesem spielen, werden mittels Observer überwacht. D. h. abhängig von der eingesetzten Applikations-Software werden die dafür vorgesehenen Observer installiert und konfiguriert.
Je nach Anzahl der überwachten Services und deren Payloads wird eine ausreichende Zahl an Monitor-Komponenten installiert und den Observer-Instanzen zugeordnet. Das ungefähre Sizing für die Monitor-Komponenten entspricht zwischen 50 und 100 überwachten Services. Führt eine Observer-Instanz eine größere Anzahl von Services, die zu überwachen sind, oder haben die Services einen höheren Payload, besteht die Möglichkeit mittels Loadbalancers mehrere Monitor-Komponenten mit einen Oberserver zu verbinden. Eine weitere Möglichkeit die Größe der erfassten Daten im Vorfeld zu reduzieren, ist das Einrichten von Observer-Filtern. Diese Filter erlauben es nur bestimmten Objekt-Typen zu überwachen/monitoren und daraus resultierend reduziert sich das Aufkommen von Metrik-Daten auf das Wesentliche.

Der/die Monitor-Instanz/en werden je Instanz auf einen WebLogic Server installiert.
Bei einer großen Anzahl von zu verarbeitenden Serviceaufrufen können die Komponenten Sphere, Transaction Monitor und Performance Monitor entweder auf eine WebLogic-Instanz installiert werden oder einzeln je Komponente einer WebLogic Server-Instanz zugewiesen werden.

Fazit
OBTM ermöglicht das Überwachen, die Fehlersuche und Transaktionsverfolgung in einer inhomogenen, verteilten Applikations-Infrastrukur. Die OBTM-Architektur ermöglicht größtmögliche Flexibilität in Bezug auf Implementierung und Skalierung.
OBTM ist nicht dafür ausgelegt eine große Anzahl an unterschiedlichen Metriken zu überwachen, sondern ermöglicht es, einzelne Transaktionen über mehrere Stationen detailliert zu verfolgen.

Kontaktadresse:

Marcus Schröder
Oracle Deutschland B.V. & Co. KG
Lina-Ammon-Straße 19
D-90471 Nürnberg

Telefon:	**+49 (0) 911 98182471**
Fax:	**+49 (0) 911 98182471**
E-Mail	**marcus.schroeder@oracle.com**
Internet:	**www.oracle.de**