"Eventually Consistent" - Neubetrachtung von Enterprise-Anwendungsarchitekturen unter BigData

Andreas Tönne

Director R&D
NovaTec GmbH
Dieselstraße 18/1
70771 Leinfelden-Echterdingen
andreas.toenne@novatec-gmbh.de

Ein zentrales Versprechen von BigData ist die Lösung von Problemen bezüglich scheinbar unmöglicher Datenvolumina und Änderungsfrequenzen. Die typischen Beispiele für erfolgreiche BigData Anwendungen decken dabei Probleme ab, die mit traditionellen Enterprise-Architekturen und -Technologien trotz Optimierung dieser Architekturen und Technologien kläglich scheitern – oder mit gutem Grund – überhaupt nicht erst versucht wurden.

Wir betrachten anhand einer Umstellung einer Enterprise-Middleware auf BigData Technologien die technischen und fachlichen Probleme und Konsequenzen einer Migration einer nach traditionellen Prinzipien entwickelten Enterprise-Anwendung nach BigData. Wir konzentrieren uns dabei auf das Spannungsfeld zwischen traditionell quasi selbstverständlichen nicht-funktionalen Eigenschaften der Anwendung, die sich aus dem ACID Prinzip ergeben, und dem sich aus dem CAP-Theorem und Lockfreiheit ergebenden Widerspruch von Konsistenz und Verfügbarkeit bei einer skalierbaren verteilten Anwendung. Dies ist kein theoretisches Problem sondern sehr real: es geht um den mit den Stakeholdern auszufechtenden Konflikt von (zeitnaher) Konsistenz der Daten und Skalierbarkeit der Plattform.

Unser Praxisbeispiel ist eine Middleware zur Datenintegration. Daten aus beliebigen heterogenen, strukturierten und unstrukturierten Datenquellen werden importiert und in ihrem Schema automatisiert vereinheitlicht integriert. Die Integration erfolgt dabei über die Ermittlung von inhaltlichen Verknüpfungen zwischen den Datensätzen anhand von semantischen Analyseverfahren. Anschaulich erhält man als Ergebnis einen Graphen von Objekten und zu einem Objekt über diesen Graphen eine Auswahl von inhaltlich "interessanten" verwandten Objekten.

Diese Middleware wurde vor ihrer Migration mit einer typischen Java EE Schichtenarchitektur unter Nutzung einer relationalen Datenbank implementiert. Die Lösung hat bei Proof-of-Concept-Projekten zu großem Kundenzuspruch in Anwendungsgebieten wie der Integration von Unternehmensdaten geführt. Der reale

Datenumfang hat aber gezeigt, dass die nicht-funktionalen Anforderungen die Leistungsfähigkeit der gewählten Architektur und Technologien um mehrere Größenordnungen überschreiten würde: Es geht um viele Millionen von Dokumenten mit vielen Milliarden von inhaltlichen Verknüpfungen.

Ein sehr charakteristisches Problem ist die Einhaltung von Konsistenzkriterien, die die horizontale Skalierbarkeit hart begrenzen. Man stelle sich vor, dass zwei inhaltlich zueinander bezogene Datensätze zeitgleich importiert werden, wodurch Konsistenzanforderungen des Graphen verletzt werden. In einer traditionellen Enterprise-Architektur wird dies zum Beispiel über Locking oder über Datenbankconstraints abgesichert. Derartige Konsistenzanforderungen sind regelmäßig in Enterprise-Datenmodellen anzutreffen. In einer hochgradig horizontal skalierenden, verteilten Architektur muss man dagegen sicherstellen, dass der Grad der Synchronisation zwischen den Anwendungsknoten minimal ist, damit die Knoten unabhängig voneinander rechnen können und nicht aufeinander warten müssen.

Wir haben uns in dem Projekt aufgrund des existierenden fachlichen Graphmodells für eine Graphdatenbank (Titan) mit NoSQL Storage (Cassandra) und verteiltem Index (ElasticSearch) entschieden. Den oben genannten Konflikt lösen wir durch den neuen Technologiestack aber nicht. Durch die bessere Skalierbarkeit der neuen Architektur wurde der Konflikt sogar noch verschärft und erfordert deshalb einen Paradigmenwechsel.

Zu erheblichem Diskussionsbedarf mit den Stakeholdern und erheblichem Forschungsaufwand hat geführt, dass der Übergang zu einer skalierbaren, verteilt rechnenden und speichernden Anwendungsarchitektur zwangsläufig zu Änderungen der funktionalen und nicht-funktionalen Eigenschaften einer Enterprise-Anwendung führen muss.

Für diese wird in der Regel stillschweigend das ACID Prinzip vorausgesetzt. Für uns ergibt sich daraus die Erwartung der Stakeholder, dass Daten sofort nach ihrer Erfassung konsistent verfügbar sind. In einer BigData Architektur mit hochgradig horizontal skalierenden, parallel ausgeführten Komponenten ist ACID aber nicht immer einzuhalten. Es kann für eine begrenzte Zeit nach einer Datenänderung nicht garantiert werden, dass die Konsistenz der Daten auf allen Knoten gilt, was mit "eventually consistent" bezeichnet wird.

Ganz praktisch ergab sich für uns die Notwendigkeit der Entscheidung, ob wir für das obige Beispiel der Erweiterung des Graphen mehr Wert auf Konsistenz (synchronisierte Änderungen) oder Antwortzeit (nicht synchronisierte Änderungen) legen wollten. Bei einer verteilten, nicht synchronisierten Datenänderung ergibt sich zwangsläufig ein Zeitfenster, in dem die Daten entweder potenziell inkonsistent aber dafür auf allen Knoten verfügbar sind oder in dem die geänderten Daten nicht auf allen Knoten gesehen werden. Wir haben uns für ein Zeitfenster der Inkonsistenz entschieden und damit eine sehr gute horizontale Skalierbarkeit und hohen Durchsatz beim Import von Daten erzielt. Dieses Zeitfenster wird durch periodische Korrekturprozesse auf Hadoop-Basis geschlossen. Wichtig für diese Entscheidung war eine Abschätzung der Auswirkungen

der Konsistenzfehler. Es gibt ebenso Einsatzszenarien der Middleware, in denen bestimmte Daten durch eine alternative Analytik konsistent aber zwangläufig mit deutlich schlechterem Durchsatz zu importieren sind.

Die Umstellung auf BigData hat nicht nur funktionale Änderungen und Einschränkungen bedeutet. Die neue Middleware hat in großem Maße von der Zuverlässigkeit der BigData-Komponenten Titan, Cassandra und ElasticSearch profitiert. Sowohl Cassandra als auch ElasticSearch lassen sich zur Lastverteilung und Hochverfügbarkeit sehr leistungsfähig clustern. Sie bieten eine zuverlässige Replikation an, die gerade für BigData Datenvolumen für eine schnelle Erholung von Ausfällen von Knoten sehr wertvoll ist.

Unser Fazit: Die Migration einer bestehenden Enterprise-Anwendung auf einen BigData-Technologiestack setzt voraus, dass eine Abstimmung mit allen relevanten Stakeholdern erfolgt. Es muss eine Abwägung und Entscheidung der möglichen Änderungen der bisherigen fachlichen und nicht-fachlichen Eigenschaften erfolgen. Ebenso muss beachtet werden, dass die hoch-parallele Verarbeitung, mit zum Beispiel Hadoop, zusätzliche völlig andersartige Algorithmen und Geschäftsregeln benötigt. Ohne eine Entscheidung zu Änderungen an den Anforderungen, und hier besonders an der Abwägung von Skalierbarkeit und Konsistenz, läuft die Migration Gefahr, die Erwartungen an die Skalierbarkeit und damit an den Durchsatz nicht erfüllen zu können.