Eine Architektur für ausfallsichere Systeme in standortübergreifenden Multiserver-Umgebungen

Andree de Boer

sd&m AG
Lübecker Str. 128
22087 Hamburg
andree.de.boer@sdm.de

Abstract: Die Konstruktion ausfallsicherer Systeme gehört auch heute noch zu den anspruchsvolleren Gebieten des Software-Engineerings. Zur echten Herausforderung wird diese Aufgabe, wenn in einem standortübergreifenden Gesamtsystem aus mehreren weit entfernten Servern nur eingeschränkt zuverlässige WAN-Verbindungen zur Verfügung stehen. Mit welchen Architekturen und Verfahren unter solch erschwerten Gegebenheiten eine für die Abwicklung hochgradig zeitkritischer Geschäftsprozesse ausreichende Verfügbarkeit sichergestellt werden kann, zeigt dieser Bericht am Beispiel eines Buchungs- und Ticketing-Systems für eine Fährreederei.

1 Anforderungen

Hintergrund dieses Beitrags ist ein Projekt zur Entwicklung eines Buchungs- und Ticketing-Systems für eine große Ostsee-Fährreederei. Das Unternehmen verbindet 16 Häfen über unterschiedliche Routen im kombinierten Passagier- und Frachtverkehr. Der Abfahrtstakt beträgt je nach Hafen bis zu zwei Abfahrten pro Stunde, in Spitzenzeiten müssen über alle Häfen bis zu 4000 Eincheck-Vorgänge pro Stunde abgewickelt werden. Entsprechend hoch sind die Last- und Verfügbarkeitsanforderungen an das System, welches zudem rund um die Uhr laufen muss.

Als anspruchsvoll sind auch die fachlichen Anforderungen zu bezeichnen. Eine komplexe Produkt- und Tarifstruktur erfordert einen hohen Rechenaufwand, und ein ausgefeiltes Kapazitätsmanagement zur Vermeidung von Überbuchungen erzwingt eine zentrale Datenhaltung.

Die Herausforderung für die Systemarchitekten bestand in der Erfüllung der divergierenden Anforderungen (Zentrale Datenhaltung vs. Verfügbarkeit; Performance vs. Komplexität). Als Lösung wurde ein Systemaufbau aus einem zentralen Server und über ein WAN angebundenen Hafenservern gewählt. Im Folgenden wird dargestellt, wie verschiedene Aspekte der Ausfallsicherheit innerhalb dieser Architektur adressiert wurden.

2 Verfügbarkeit

Die Ausfallszenarien gliedern sich in zwei Kategorien:

1. Ausfall von Hardwarekomponenten,

2. Ausfall der WAN-Leitung zur Zentrale.

Ein Ausfall von Hardwarekomponenten darf das System nicht negativ beeinflussen. Aus diesem Grund wurde eine skalierbare Architektur gewählt, die es erlaubt, Hardwarekomponenten in kritischen Bereichen mehrfach auszulegen. Das System muss auch bei Ausfall einer Hardwarekomponente für den Benutzer transparent weiterarbeiten können.

Durch die dezentrale Lage der einzelnen Häfen auf der einen Seite und die Anforderung der zentralen Datenhaltung auf der anderen Seite muss sichergestellt werden, dass im Falle eines Ausfalls der WAN-Leitung in den Häfen transparent weitergearbeitet werden kann. Hierfür steht in wichtigen Häfen eine komplette Serverumgebung, die bei Ausfall der WAN-Leitung autark weiter zur Verfügung steht. Dieser Beitrag stellt die zugrunde liegende Architektur vor und geht dabei auf potentielle Probleme und Szenarien ein, die sich durch das autarke Arbeiten ergeben.

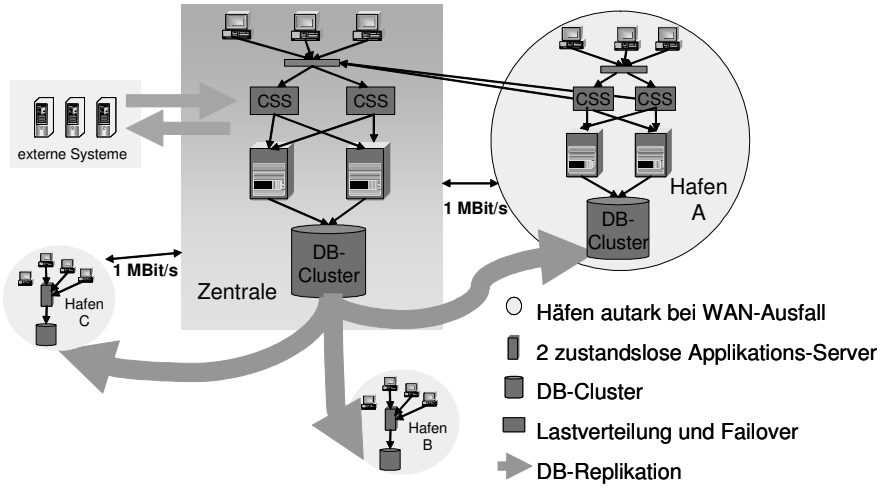


Abbildung 1: Multiserver-Architektur

3 Synchrone Datenhaltung in Zentrale und Häfen

Eine weitere Hauptanforderung ist die zentrale Datenhaltung, um beispielsweise Überbuchungen zu vermeiden und allen Häfen die gleiche Sicht auf die Daten zu gewährleisten. Hierfür wird die Zentrale als Master definiert. Mit Hilfe von Delegationsstrategien können Geschäftsvorfälle, die Datenänderungen vornehmen, direkt an die Zentrale delegiert werden. Der Geschäftsvorfall wird dann in der Zentrale ausgeführt und die Änderungen wiederum in die Häfen repliziert.

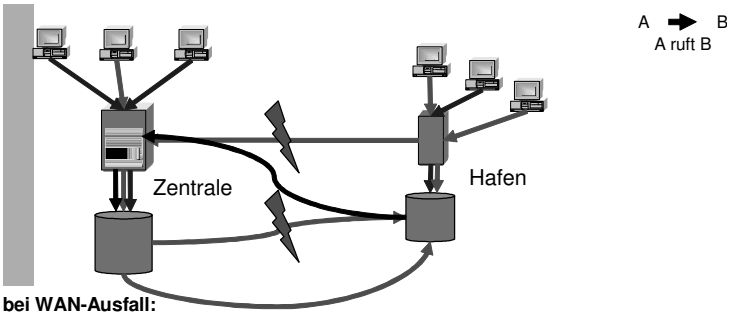
3.1 Replikation von der Zentrale in die Häfen

Für die Replikation wurden sogenannte Refresh-Gruppen eingeführt. Eine Refresh-Gruppe ist eine Gruppierung von fachlich zusammenhängenden Datenbanktabellen. Alle Tabellen, die in einer Refresh-Gruppe definiert sind, werden entweder vollständig oder gar nicht repliziert (Transaktionssicherheit). Die Nutzung von Refresh-Gruppen stellt damit sicher, dass replizierte Daten in sich immer konsistent sind.

Um die durch Replikation verursachte Last möglichst gering zu halten, kann für jede Refresh-Gruppe ein Refresh-Intervall festgelegt werden, in dem die Datenbanktabellen der Refresh-Gruppen repliziert werden. Daten, die nicht in Echtzeit benötigt werden (z. B. neue Stammdaten) werden dabei seltener repliziert als geschäftskritische Daten (z. B. Buchungen), die nahezu ständig repliziert werden.

3.2 Nachziehen von Geschäftsvorfällen nach einem Ausfallbetrieb

Während eines Ausfallbetriebs werden alle Änderungen nur am Hafen durchgeführt. Die anfallenden Änderungen werden in eine Queue geschrieben, die nach der Rückkehr in den Normalbetrieb zunächst von der Zentrale abgearbeitet wird. Erst nach Abarbeitung der Queue wird der Hafen wieder für die Replikation der Daten aus der Zentrale freigeschaltet. Mögliche Konflikte bei der Abarbeitung der Queue können dazu führen, dass Daten, die bereits im Hafen geändert wurden, durch die Replikation wieder überschrieben werden.



1. Betrieb bei WAN-Ausfall:

- Die Häfen arbeiten vollständig gegen ihre lokale Datenbank
- Ändernde Usecases werden in einer Queue in der lokalen Datenbank persistiert

2. nach WAN-Ausfall:

- Queue wird von der Zentrale abgearbeitet: lokale Änderungen gelangen in die Zentrale

3. nach WAN-Ausfall und Queue leer:

- Hafen wechselt wieder in den Normalbetrieb, d.h. delegiert an die Zentrale
- Daten werden wieder in die Häfen repliziert, lokale Änderungen werden überschrieben

Abbildung 2: Szenario eines Ausfallbetriebs

4 Software-Updates im laufenden Betrieb

Der 24x7x365-Betrieb bedingt die Möglichkeit, Releases und Patches im laufenden Betrieb einzuspielen. Bei hochfrequentierten Häfen ist es nicht möglich, ein Wartungsfenster einzurichten, da dort rund um die Uhr Abfahrten abgefertigt werden müssen. Das Installieren einer neuen Softwareversion wird daher zu einer besonderen Herausforderung, da der laufende Betrieb nicht unterbrochen werden darf.

4.1 Ausrollen neuer Releases

Das Ausrollen eines neuen Release im laufenden Betrieb wird durch die Fähigkeit ermöglicht, zwei Releases parallel laufen lassen zu können. Hierbei laufen zeitweise die alte und die neue Softwareversion als zustandsloser Applikationsserver gleichzeitig. Während bereits geöffnete Clients noch mit der alten Version verbunden sind, verbinden sich neu geöffnete Clients bereits mit der neuen Version. Technisch werden die Serverversionen mithilfe der Webstart-Technologie über unterschiedliche IP-Ports angesprochen. Für den Anwender gestaltet sich der Zugriff daher transparent. Sobald die neue Version zur Verfügung steht, wird beim nächsten Öffnen des Client die neue Clientversion heruntergeladen und automatisch mit der neuen Serverversion verbunden. Die nötigen Einstellungen hierfür werden alleine vom Administrator vorgenommen.

Beide Versionen des Applikationsserver greifen auf dieselbe Datenbasis zu. Ein und dieselben Daten können in dieser Übergangszeit also sowohl durch Funktionen der alten Version als auch durch Funktionen der neuen Version modifiziert werden. Als Konsequenz muss bei der Entwicklung eines neuen Release streng auf die Abwärtskompatibilität zum Vor-Release geachtet werden.

4.2 Einspielen von Patches

Die mehrfache Auslegung des Servers ermöglicht es, Patches durch ein rollierendes Durchstarten der Server transparent einzuspielen. Szenarien für das Einspielen eines Patches werden näher beleuchtet.

5 Das System in der Praxis

Das System wurde in mehreren Schritten ausgerollt. Im März 2004 ging der erste Hafen live, bis Ende 2005 waren alle 16 Häfen angebunden. Bemerkenswert ist, dass sich die WAN-Leitungen als noch instabiler herausstellten als ursprünglich angenommen – und das System dennoch stabil läuft. Die dargestellten Maßnahmen zur Ausfallsicherheit haben sich somit in der Praxis bestens bewährt.

Im Sommer 2006 während der Hauptsaison wurden an einem Tag an den beiden Zentralservern insgesamt 1,1 Mio Geschäftsvorfälle ausgeführt (12,8 GVs pro Sekunde). Die Durchschnittliche Ausführungszeit pro Geschäftsvorfall betrug dabei 389 ms. An diesem Tag wurden außerdem insgesamt 235 Geschäftsvorfälle nach einem Ausfallbetrieb in der Zentrale nachgezogen.

Wie bei jedem komplexeren System ist auch dieses System im Betrieb natürlich nicht vollständig frei von Problemen. Ein großes Problem war z. B. die Tatsache, dass aufgrund der Architektur von Java nicht bestimmt werden kann, wie viel Speicher ein Thread (fachlicher Geschäftsvorfall) aktuell verbraucht. Dies führte dazu, dass eine fehlerhaft programmierte Funktion den Server dahingehend beanspruchte, dass dieser nicht mehr in ausreichender Zeit auf Anfragen reagieren konnte. Mögliche Lösungsansätze für dieses Problem werden im Vortrag skizziert.