Portfolio-Aufgabe zur Vorlesung Verteilte Systeme

Dozent: *Prof. Dr. Michael Eichberg*
Version: 2024-02-24 - SE Studiengänge

Aufgabenstellung

> Das Ziel dieser Aufgabe ist es, dass Sie ein besseres Verständnis für die Implementierung von verteilten Systemen sowie die Implementierung von Geschäftsprozessen in selbigen erlangen. Dazu entwickeln wir ein System, dass mittels Nachrichten und mit Hilfe von SAGAs langlaufende Geschäftsprozesse simuliert. Darüber hinaus soll das Verständnis für nebenläufige Programmierung verbessert werden.

Im Rahmen dieser Aufgabe sollen Sie ein verteiltes System zum Buchen von Reisen in Hinblick auf technisch relevante Aspekte simulieren.

Es soll ein Reise-Broker implementiert werden, der es den Kunden ermöglicht eine Reise bestehend aus mehreren einzelnen Hotelbuchungen und ggf. Flugbuchungen zu buchen. Die Gesamtbuchung des Endkunden soll nur dann durchgeführt werden, wenn alle Teilbuchungen erfolgreich durchgeführt werden konnten. Sollte zum Beispiel ein Hotel keine Zimmer mehr frei haben, so soll die Buchung für den Kunden als Ganzes nicht durchgeführt werden und selbstverständlich sollen entsprechende vorherige Teilbuchungen zurückgerollt werden. Das System muss Buchungsverläufe mit bis zu vier verschiedenen Hotels und fünf Flügen verarbeiten können.

Es sind also folgende Kernsysteme zu implementieren:

- Ein System zum Flüge buchen. (Ggf. gibt es zur Laufzeit mehr als eine Instanzen; z. B. von den Airlines „FliegtBeiSonne“ und „StürztNieAb“.)
- Ein System zum Hotels buchen. (Ggf. gibt es zur Laufzeit mehrere Instanzen; z. B. von den Touristikunternehmen „UrlaubInDenBergen“ und „FluchtAnDieSee“.)
- Der Reise-Broker, der die Buchungen von Reiseverläufen koordiniert.
- Ein Message-Broker, der die Kommunikation zwischen den Systemen ermöglicht.
- Ein System, dass Reiseverläufe, die gebucht werden sollen, an den Reise-Broker weiterleitet und die Ergebnisse protokolliert im Sinne von: „Alles gebucht“, „Hotel X konnte nicht gebucht werden“ oder „Flug Y ist nicht mehr verfügbar“. (Diese Funktionalität kann auch lose in den Reise-Broker integriert sein in der Form, dass ein Thread die zentrale Methode des Reise-Brokers für die Endgegennahme von Reiseverläufen mit einem entsprechenden Takt aufruft.)

Sie können alle Systeme innerhalb einer VM simulieren oder Sie nehmen zur Interprozesskommunikation zum Beispiel Java RMI oder den JDK HTTP Server. Die Systeme sollen auf jeden Fall nebenläufig laufen und über einen einfachen Message-Broker (mind. konzeptuell) asynchron miteinander kommunizieren. Zum Beispiel könnte der Reise-Broker eine Anfrage an das Hotel-System via des Message-Brokers senden und das Hotel-System antwortet dann (asynchron) mit einer Bestätigung oder einer Ablehnung.

Hinweise:

- Zum Nachrichtenaustausch können Sie einfache (*immutable*) PoJos verwenden.
- Nutzen Sie ggf. *virtual Threads*, um die Kommunikation zwischen den Systemen zu simulieren bzw. um die simulierten Berechnungszeiten darzustellen.

- Die Hotel- und Flugdaten sowie die Buchungsvorgänge sollten in einem geeigneten Format vorliegen und zur Laufzeit eingelesen werden (Properties-Dateien, XML, JSon, Yaml, Text, ...). Eine dauerhafte Persistenz ist nicht notwendig. Halten Sie das Datenmodell einfach.
- Das System muss mit (simulierten) Ausfällen, langen Latenzen und ganz insbesondere mit fachlichen Situationen, die zu einem Rollback führen, umgehen können.
- Der Reise-Broker und der Message-Broker wird in diesem Szenario als zuverlässig und stets verfügbar angenommen.
- Das System, dass Buchungsanfragen an den Reise-Broker weiterleitet und die Ergebnisse protokolliert, unterliegt keinen simulierten Ausfällen; die Kommunikation zwischen diesem System und dem Reise-Broker ist ebenfalls zuverlässig.

Anforderungen

Die folgenden Parameters sollen in Ihrem Programm variabel sein bzw. über eine Konfigurationsdatei einstellbar sein:

- Ankunftsrate von Buchungsanfragen beim Reise-Broker.
- Bearbeitungszeit der Anfragen durch die Buchungsservices.
- Verzögerung der Nachrichtenzustellung durch den Message-Broker.
- Wahrscheinlichkeit mit der ein Buchungsdienst (Hotel, Flug) eine Nachricht
  - annimmt aber nicht verarbeitet („simulierter Absturz“)
  - verarbeitet (d. h. eine Änderung seines Zustandes vornimmt, aber die Nachricht nicht bestätigt)
  - erfolgreich verarbeitet (d. h. ggf. eine Änderung seines Zustandes vornimmt aber auf jeden Fall die Nachricht bestätigt); wir unterscheiden:
    - Buchung war erfolgreich.
    - Buchung konnte nicht durchgeführt werden, da das Angebot nicht mehr verfügbar ist.

      (Demzufolge muss ein Rollback durchgeführt werden.)

(Nutzen Sie Normalverteilungen mit einem Mittelwert und einer Standardabweichung als Basis für die benötigten Zufallszahlen.)

Randbedingungen

- Die Aufgabe soll in Vierergruppen (nur bei Überhang in Fünfergruppen) gelöst werden. Es werden Einzelnoten vergeben.
- Die Implementierung soll in Java erfolgen.
- Außer für Logging, Tracing und Visualisierung sind keine externen Bibliotheken zu verwenden.
- Achten Sie auf ein sauberes, verständliches Design!

  Die Geschäftslogik soll so implementiert sein, dass ein Austausch der Kommunikationsmechanismen möglich wäre (vgl. *Ports & Adapters*, *Onion Architecture* oder *Clean Architecture*). Die Logik für die Datenhaltung kann im Rahmen dieser Aufgabe mit der Geschäftslogik zusammen implementiert werden.
- Fokussieren Sie sich auf die Nebenläufigkeit, die möglichen Fehlerquellen und die Korrektheit der Transaktionen; in Hinblick auf die Modellierung der Flug- und Hoteldaten nehmen Sie sich die Freiheit, das Modell so einfach wie möglich zu halten. Natürlich müssen die Ressourcen beschränkt sein und es muss möglich sein, dass Buchungen aufgrund nicht mehr vorhandener Ressourcen fehlschlagen.

Abgabe

- pro Gruppe ist eine schriftliche Ausarbeitung zur Architektur und Testkonzept zu erstellen (ca. 3 - 5 Seiten)
- der sinnvoll kommentierte Java-Quellcode und Testdatensätze
- lauffähige `class`-Dateien mit entsprechenden Initialisierungsdateien sowie einer kurzen Bedienungsanleitung sind einzureichen (als JAR Datei(en), die direkt gestartet werden kann bzw. können „java -jar <jar-Datei> <Parameter>“)
- ein mit Sprache kommentiertes Video, das eine Demo-Vorführung der Lösung zeigt und eine Besprechung der Implementation; im Video sollen:

  a. die Funktionalität und das Verhalten bei Fehlerfällen demonstriert werden. Hierfür kann die Rate der Erzeugung von Nachrichten soweit heruntergefahren werden, dass effektiv immer „nur“ sehr wenige Vorgänge (mit entsprechenden Fehlern) beobachtet werden.

  b. wie sich das System verhält (z.B. mit Hilfe von Loggingdaten/Visualisierungen), wenn viele (mindestens Dutzende) Anfragen quasi gleichzeitig bearbeitet werden.

  Große Videodateien sind nach Möglichkeit zu vermeiden dürfen jedoch ggf. in einer Cloud gespeichert werden. Der Link zum Abruf muss dann in der Dokumentation angegeben werden und mind. 8 Wochen nach der Abgabefrist verfügbar sein; achten Sie dabei auf ausreichende Zugriffsrechte.

  Bedenken Sie: «In der Kürze liegt die Würze!»; schneiden/beschleunigen Sie Ihr Video, wenn es Abschnitte gibt, die nicht relevant sind.

- Es ist ein Dokument abzugeben aus dem hervorgeht welches Gruppenmitglied an welchen Abgaben wie mitgearbeitet hat.

  (Z. B. „Lara hat den Message-Broker implementiert“; „Tim hat das Testkonzept und die Testumgebung aufgesetzt“; „Tom hat den Testdatengenerator geschrieben.“; „Lukas und Marlow haben gemeinsam die Hotel- und Flugbuchungsdienste implementiert. Die Zusammenarbeit diente vor allem dem Ausarbeiten der Architektur. Die Implementierungen des Hotelbuchungsservice wurde dann von Marlow verantwortet und die des Hotelbuchungsservices von Lukas.“)

- Eine von allen Gruppenmitgliedern persönlich unterschriebene ehrenwörtliche Erklärung mit folgendem Wortlaut ist ebenfalls mit einzureichen:

  Hiermit erklären wir ehrenwörtlich, dass wir die vorliegende Portfolio-Arbeit zur Vorlesung ”Verteilte Systeme” bestehend aus Ausarbeitung, Programmcode und Video selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt haben.

  `Datum, Unterschriften`

Eine Abgabemöglichkeit im Moodle-Kursraum wird eingerichtet.