Universität Trier

Fachbereich IV / Informatik
Prof. Dr. Peter Sturm
D-54286 Trier
Telefon 0651/201-3313
Fax 0651/201-3842
Email sturm@uni-trier.de

Betriebssysteme, Übungsblatt 3, Winter 2024

Optimistische Nebenläufigkeit mit ZFS-Snapshots

In diesem Übungsblatt entwickeln und nutzen Sie ein Dateiverwaltungssystem, das mehrere parallele Transaktionen erlaubt, um Dateien optimistisch zu modifizieren und das trotzdem die Konsistenz aller Dateien sicherstellt. Ihr System nutzt ZFS-Snapshots, um zu Beginn jeder Transaktion konsistente Ansichten eines bestimmten Verzeichnisses zu erstellen. Jede Transaktion geht davon aus, dass keine Konflikte auftreten, und Änderungen werden nur dann übernommen, wenn am Ende der Transaktion keine Konflikte erkannt werden. Wird ein Konflikt festgestellt (z. B. wenn eine andere Transaktion dieselbe Datei während des Prozesses modifiziert), muss die Transaktion zurückgesetzt werden, indem das Dateisystem mithilfe der ZFS-Snapshots in den ursprünglichen Zustand versetzt wird.

Java-Bibliothek

Konzipieren und implementieren Sie eine Java-Bibliothek, die diese Transaktionslogik bei Dateimodifikationen verwendet. Sie sollen dabei ZFS-Snapshots programmatisch erstellen und verwalten, Änderungen an Dateien innerhalb von Transaktionen nachverfolgen und Konflikte anhand von Metadaten (z. B. Zeitstempeln oder Hashwerten) erkennen. Ziel ist es, ein System zu entwickeln, das grundlegende Operationen wie Lesen, Schreiben und Löschen von Dateien unterstützt und dabei die Prinzipien der Isolation und Atomizität einhält.

Prototyp einer Brainstorming-Anwendung

Entwickeln Sie ein einfaches kollaboratives Brainstorming-Tool, bei dem jede Idee mit den zugehörigen Kommentaren in jeweils einer Dateien gespeichert wird. Aufbauend auf dem zuvor implementierten System zur gleichzeitigen Dateiverarbeitung mit ZFS-Snapshots sollen Benutzer in diesem Tool neue Ideen hinzufügen, bestehende Ideen lesen und kommentieren können. Jede Änderung an einer Datei muss atomar und konfliktfrei erfolgen, wobei ein optimistischer Ansatz angewendet wird, um Konflikte bei gleichzeitigen Zugriffen zu erkennen und betroffene Transaktionen zurückzusetzen. Einfache textbasierte Programme, die vorzugsweise über die Kommandozeile genutzt werden können, reichen völlig aus. Sie können zum Beispiel auch innerhalb eines entsprechenden Java-Programms jeweils eine Transaktion beginnen, anschließend einen geeigneten externen Texteditor aufrufen und nach dessen Terminierung die Transaktion entsprechend abschließen. Ziel ist es, ein einfaches, aber robustes System zu schaffen, das Transaktionen für Dateioperationen verwaltet und Konflikte korrekt behandelt.

Validierung

Entwickeln Sie ein Validierungstool, das gezielt die Wahrscheinlichkeit von Konflikten bei gleichzeitigen Dateioperationen maximiert und deren Auswirkungen analysiert. Das Tool soll automatisiert mehrere gleichzeitige Transaktionen mit zufälligen Textelementen simulieren, die auf eine gemeinsame Menge von Dateien zugreifen, und dabei unterschiedliche Szenarien erzeugen, z.B. häufige Schreibzugriffe oder konkurrierende Lese-/Schreiboperationen. Ziel ist es, die Effizienz und Robustheit des implementierten Systems zu testen, Konflikte zu erzeugen, deren Behandlung zu beobachten und die Systemleistung unter hoher Last zu evaluieren. Zusätzlich können Metriken wie Konfliktraten, Rücksetzzeiten oder die Systemdurchsatzrate erhoben werden, um Optimierungspotenziale zu identifizieren.

Abgabe

Abzugeben sind ein oder mehrere Links zu den Repositories mit den Implementierungen sowie ein Bericht im PDF-Format, der jeweils auf die gewählten Implementierungen eingeht. Der Schwerpunkt sollte auf Teilaufgabe 1 liegen.

Als Deadline für die Abgabe ist aktuell der 18.3.2025 geplant. So haben Sie maximal viel Zeit für die Bearbeitung und ich schaffe es noch vor dem Semesterende, die Lösungen zu bewerten und die Noten in Porta einzutragen.