Barry Linnert

Nichtsequentielle und verteilte Programmierung, SS2021

Übung 8

Tutor: Florian Alex
Tutorium 3

Rui Zhao, William Djalal, Simeon Vasilev

18. Juni 2021

1 Programmierung in Java (8 Punkte)

(3 Pkt. Umgebung, 2 Pkt. Testen, 3 Pkt. Fehlerbehandlung)

Richten Sie sich eine Programmierumgebung (z.B. Eclipse) für die Programmierung in Java ein. Beschreiben Sie Ihre Umgebung kurz (stichpunktartig) und testen Sie die Beispiele aus der Vorlesung. Geben Sie zudem an, welche Möglichkeiten der Fehlersuche und -behandlung Ihnen in Ihrer Umgebung zur Verfügung stehen.

Umgebung Als Programmierumgebung haben wir IntelliJ gewählt. IntelliJ bietet alle in der Programmierumgebung bekannten Standardfunktionen. Es verfügt über einen Dateibaum, einen Editor, bearbeitbare Verknüpfungen zum Erstellen und Ausführen und ein integriertes Terminal.

Testen

Wir laden die Vorlesungsbeispiele in die Umgebung, bauen sie und lassen sie laufen. Sowohl die UPD- als auch die TCP-Beispiele funktionieren gut.

Siehe den Screenshot unten für das Testbeispiel.

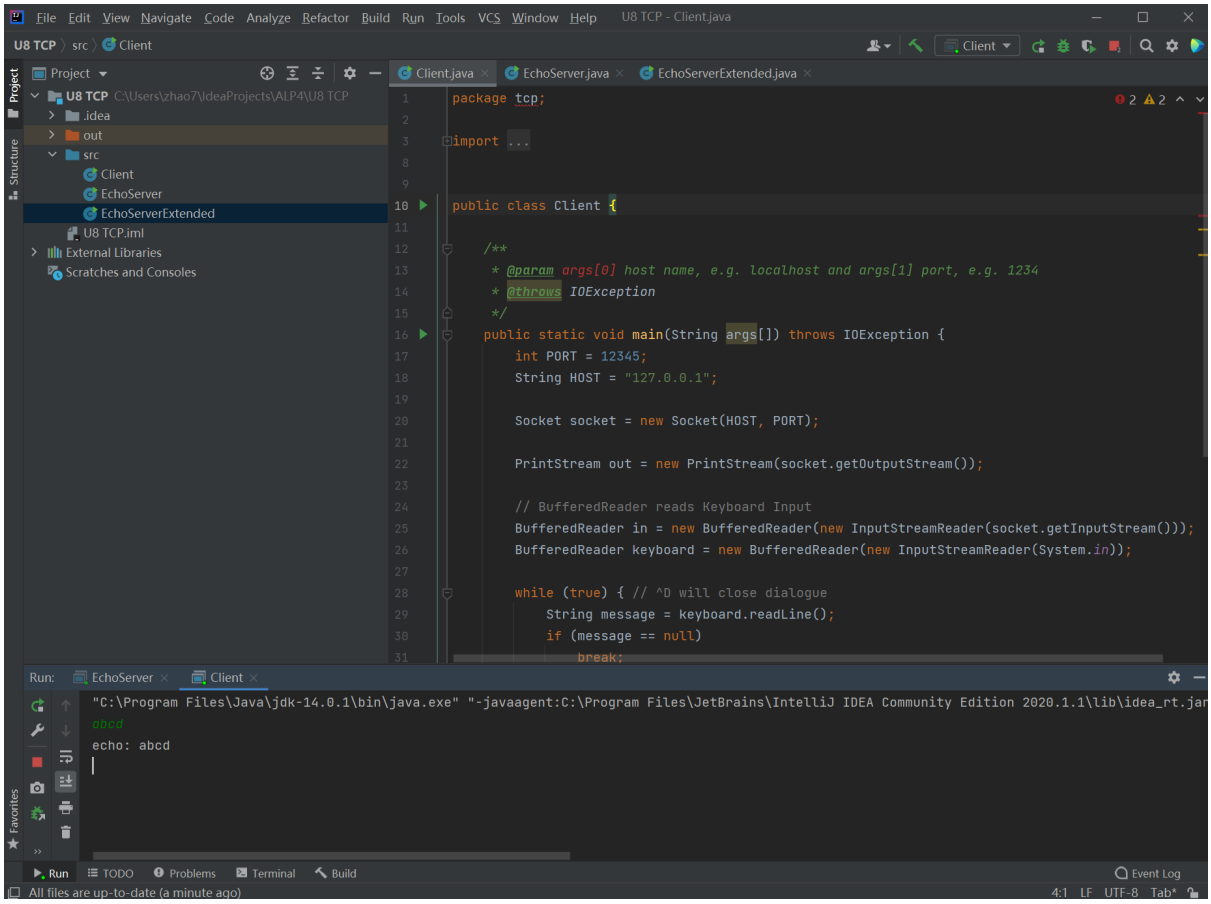

Abbildung 1: TCP ECHO TEST

✓

Fehlerbehandlung

Zum Debuggen verfügt IntelliJ über Haltepunkte und zeigt vom Compiler ausgelöste Fehler an. An den Haltepunkten können die Werte von allen verwendeten Variablen inspiziert werden. Um Fehler zu vermeiden, bietet IntelliJ bei der Eingabe von Funktionen auch ein Dropdown-Menü, das die verfügbaren Funktionen/Variablen auflistet. Darüber hinaus hilft die Syntaxhervorhebung, Syntaxfehler wie fehlende schließende Klammern und Anführungszeichen zu finden.

8/8 ✓

2 (Verteilungs-)Transparenz (14 Punkte)

(2 Pkt. Definition, 1 Pkt. je Form, 4 Pkt. Wichtigkeit)

Erklären Sie, was unter (Verteilungs-)Transparenz zu verstehen ist und geben Sie Beispiele für verschiedene Arten von Transparenz. Bitte beschreiben Sie kurz, welche Transparenz am wichtigsten in einem verteilten System sind?

Tipp: Das ANSA-Referenzhandbuch[1] und das Referenzmodell der Internationalen Organisation für Normung[2] nennen diese Formen der Transparenz.

Definition:

[1] https://www.computerconservationsociety.org/ansa/89/TR0302.pdf

[2] Das International Organization for Standardization's Reference Model for Open Distributed Processing (RM-ODP) ist nicht frei verfügbar.

Transparenz beschreibt allgemein die Zuständigkeiten des Entwicklers für den Softwareteil. Je höher die Transparenz, desto weniger müssen sich Programmierer um den genauen Ausdruck des Systems kümmern, sondern können sich auf das Systemmodell verlassen. Verteilte Transparenz bestimmt den Grad, in dem Programmierer auf die verschiedenen Komponenten des verteilten Systems achten müssen. In einem Verteilungstransparenten Programm ist diese Verteilung teilweise implizit und muss vom Programmierer nicht vollständig berücksichtigt werden. Ok ..

Form: [1] [2]

Location transparency: Location transparency bedeutet, dass der Standort jeder Schnittstelle der Software im System vertraulich ist und der Benutzer den physischen Standort der Ressource im System nicht bestimmen kann.

Concurrency transparency: Jeder Benutzer eines Systems wäre als einziger Benutzer des Systems. Der Benutzer wird zu keinem Zeitpunkt feststellen, dass andere Benutzer das System gleichzeitig verwenden.

Failure transpareny: Benutzer werden nicht bemerken, dass eine bestimmte Ressource nicht richtig funktioniert und der anschließende Prozess der Systemwiederherstellung nach einem Ausfall.

Migration transparency: Es handelt sich um eine dynamische Form der 'Location transparency'. Wenn sich die Daten im System während der Nutzung bewegen, ist dies für den Benutzer nicht ersichtlich. Die Verschiebung von Ressourcen in einem verteilten System hat keinen Einfluss auf die Art und Weise, wie auf die Ressource zugegriffen wird.

Access transparency: ermöglicht den Zugriff auf lokale und entfernte Ressourcen mit identischen Operationen.

Replication Transparency: Das Verbergen der Tatsache, dass es mehrere Kopien derselben Ressource gibt.

Relocation transparency : Sollte sich eine Ressource während der Nutzung bewegen, sollte dies für den Endbenutzer nicht wahrnehmbar sein.

Persistence transparency : Ob eine Ressource im flüchtigen oder permanenten Speicher liegt, sollte für den Benutzer keinen Unterschied machen.

Security transparency : Die Aushandlung eines kryptographisch sicheren Zugriffs auf Ressourcen muss ein Minimum an Benutzereingriffen erfordern, oder Benutzer umgehen die Sicherheit zugunsten der Produktivität.

Wichtigkeit:

Es ist schwer zu sagen, welche Transparenz am wichtigsten in einem verteilten System sind. Weil nicht alle diese 'transparency' für jedes System geeignet sind oder auf der gleichen Schnittstellenebene verfügbar sind. Tatsächlich sind alle Transparenzen auch mit Kosten verbunden.[3] Okay :D 14/14

3 Anwendungsbeispiele für Probleme (8 Punkte)

(2 Pkt. pro Beispiel und jeweils 2 Pkt. für Probleme)

Sie nutzen tagtäglich verteilte Systeme. Beschreiben Sie anhand von zwei Beispielen, welche Probleme Sie bei der Verwendung von verteilten Systemen erlebt haben und worauf diese wahrscheinlich zurückzuführen sind.

Als erstes Beispiel für ein verteiltes System haben wir Cisco Webex gewählt. Das größte Problem bei der Nutzung lokaler Client ist die Echtzeitübertragung von Audio und Video. Es gibt oft verzerrte Töne und verzögerte Videos. Um eine gute Übertragungsqualität zu erreichen, muss die Internetverbindung der beiden Teilnehmer, des Sprechers und des Hörers, eine ausreichende Datenrate und kurze Übertragungszeit bieten. Die Schwierigkeit besteht darin, dass die Übertragung in Echtzeit erfolgen muss und jede übermäßige Verzögerung das Programm unbrauchbar macht. Wir gehen davon aus, dass alle Anrufdaten aller Teilnehmer zuerst an den Webex-Server gesendet werden und der Server sie dann an alle anderen Geräte sendet (es gibt keinen Beweis dafür, dass Webex Peer-to-Peer funktioniert). Der Ausfall einer Komponente im Übertragungsweg oder eine große Verzögerung können auch die oben genannten Probleme einer schlechten Anruf- oder Videoqualität verursachen.

Als zweites Beispiel für ein verteiltes System sei Whatsapp erwähnt. Whatsapp hat jetzt Hunderte von Millionen Benutzern, die alle gleichzeitig Nachrichten senden und empfangen können. Darüber hinaus gibt es weltweit verteilte Server, die Informationen empfangen und an den Empfänger senden. Manchmal gibt es ein Problem: Die Reihenfolge der Nachrichten, die von beiden Benutzer:in im Chat gesehen werden, kann leicht unterschiedlich

sein. Wir vermuten, dass dies daran liegt, dass die Informationen zu einem anderen Zeitpunkt auf dem Gerät eintreffen. Aufgrund der Übertragungszeit oder der erneuten Übertragung nach Übertragungsfehlern kann es zu Abweichungen kommen. Wenn sich beispielsweise zwei Nachrichten fast gleichzeitig zusenden, denkt jedes Handy-Client, dass seine eigene Nachricht die erste Nachricht ist, weil die andere Nachricht noch nicht angekommen ist.

✓

04 8/8

Literatur

[1] https://en.wikipedia.org/wiki/Transparency_(human%E2%80%93computer_interaction)

[2] https://www.computerconservationsociety.org/ansa/89/TR0302.pdf

[3] https://www.cl.cam.ac.uk/~jac22/books/ods/ods/node18.html

30/30