Technology
Arts Sciences
TH Köln

Betriebssysteme und verteilte Systeme

Praktikumsdokumentation

ausgearbeitet von

Lukas Momberg
Dennis Goßler
Jona Siebel
Daniel Mentjukov

vorgelegt an der

Technischen Hochschule Köln
Campus Gummersbach
Fakultät für Informatik und
Ingenieurwissenschaften

im Studiengang
Allgemeine Informatik

Eingereicht bei: Frau Ekellem
Technische Hochschule Köln

Gummersbach, im Juni 2021

Inhaltsverzeichnis

1 Einleitung

In dem folgenden Kapitel wird die Aufgabenstellung aufgegliedert und erläutert. Die Aufgabenstellung umfasste die Entwicklung eines Serverprogramms in C oder C++. In dem Serverprogramm sollten mehrere Clients über eine Netzwerkverbindung Daten ablegen und abfragen können. Des Weiteren sollten die Clients gleichzeitig per TCP verbunden sein und mit einem gegebenen Befehlsset Daten auf den Server schreiben, abfragen und löschen können. Dabei war es wichtig, dass die Daten dabei auf den Server konsistent gehalten werden.

1.1 Phase 1

Der erstellte Socket Server sollte mit beliebig vielen Clients über den TCP Port 5678 kommunizieren können. Die Datenhaltung sollte auf Grundlage des Key-Value-Stores erfolgen. Das Datenhaltungssystem sollte folgende Funktionen **PUT**, **GET** und **DELETE** bereitstellen. Die Funktionsaufrufe sollten zum Beispiel mit Call-by-Reference erfolgen. *key* und *value* akzeptieren nur alphanumerische Zeichen und keine Sonderzeichen. Dazu darf der *key* keine Leerzeichen enthalten. Im Folgenden kommen nun die Funktionsbeschreibungen.

Die **get()** Funktion sollte einen Schlüsselwert in der Datenhaltung suchen und den hinterlegten Wert zurückgeben. Wenn der Wert nicht vorhanden ist, wird durch einen Rückgabewert, der kleiner 0 ist, darauf aufmerksam gemacht. Wenn der Wert gefunden wurde, sollte eine 1 zurückgegeben werden.

Die **put()** Funktion sollte einen Wert (value) mit dem Schlüsselwert (key) hinterlegen. Wenn der Schlüssel bereits vorhanden ist, soll der Wert überschrieben werden.

Die **del()** Funktion sollte einen Schlüsselwert suchen und zusammen mit dem Wert aus der Datenhaltung entfernen.

Die Zeichenketten, die ein Client über TCP schickt, stellen im wesentlichen Befehle für den Key Value Server dar. Die Zeichenkette sollte folgende Struktur aufweisen:

[Befehl] **[key]** **[value]**

Des Weiteren sollte man den Befehl **QUIT** implementieren, mit dem der Client dafür sorgen kann, dass der Server die Verbindung zum Client beendet, sobald der Client diese Zeichenkette sendet.

1.2 Phase 2

Die Phase 2 beinhaltete das Programm gegen mögliche Race Conditions abzusichern. Dazu sollte man sich entsprechende kritische Situationen im Programm anschauen und gemeinsam in der Gruppe Möglichkeiten zur Lösung des Problems suchen. Dazu sollte man passende Mittel der Prozesssynchronisation nutzen. Eine weitere Aufgabe bestand darin die konsistente Datenhaltung für einen exklusiven Zugriff auf den Key-Value Store zu für Clients zu ermöglichen. Dazu sollte ein Client einen alleinigen Zugriff auf den Key-Value-Store anfordern können, sodass nur dieser Zugriff hat und alle anderen Clients blockiert werden. Dafür sollte man das durch den Befehl **BEG** und **END** realisieren. Der erste Befehl dient dazu, um den exklusiven Zugriff zu beginnen und der zweite um den exklusiven Zugriff zu beenden. Während ein Client exklusiven Zugriff auf den Key-Value-Store besitzt, dürfen alle anderen Clients keinen Zugriff erhalten und müssen warten. Aber weiterhin sollten auch nicht exklusive Interaktionen möglich sein, wenn die beiden Befehle nicht verwendet werden. Des Weiteren sollten wir eine Möglichkeit realisieren, dass ein Client sich auf Veränderung der Werte eines Schlüssels registrieren kann. Ein Client sollte sich mit dem Befehl **SUB** für einen Schlüsselwert ßubscriben"können. Sobald eine Änderung für diesen Schlüssel durch einen anderen Client stattfindet, also "published", sollten alle verbundenen Clients für den entsprechenden Schlüssel durch den Server informiert werden. Die ßubscriberërhalten die gleichen Ausgaben wie der Client, der die Befehle selbst ausgeführt hat. Ein Client sollte sich auf mehrere Keys registrieren können. In der Art eines Chats mit verschiedenen Chaträumen.

1.3 Phase 3

Die Phase 3 besteht darin das Programm so zu erweitern, um Clients die Möglichkeit zu geben, bestimmte Systemprogramme auf dem Server aufrufen zu können, um dann deren Ausgabe in den Key-Value Store zu laden. Dafür sollte man den Befehl **OP** implementieren. Der Befehl soll als ersten ein Argument, einen Key und als Zweites einen der folgenden Systembefehle entgegennehmen können:

date

Who

uptime

Die Zeichenkette des Befehls hat so auszusehen:

OP [key] [systembefehl]

Des Weiteren sollte man das Programm aus der Perspektive der Nachhaltigkeit betrachten. Das bedeutet das man sich die Systemressourcen besser einteilt mit der Hilfe zum Beispiel von Shared Memory, Semaphore, Message Queue und Filedescriptoren. Dazu sollte man die Freigabe aller Ressourcen, die nur für einen Client nötig waren, sobald dieser Client mit der Zeichenkette **QUIT** signalisiert hat, dass er die Verbindung beenden möchte. Dazu sollte man noch eine Funktion **freeResourceAndExit()** implementieren, um alle weiteren nicht mehr benötigten Ressourcen freizugeben und deren Server-Prozess zu beenden. In der Dokumentation sollte man dann erörtern, wie und wann diese Funktion aufgerufen werden sollte.

2 Unser Projekt

Im Folgenden stellen wir unsere Umsetzung der einzelnen Phasen vor und dazu erklären und erörtern wir unsere Entscheidungen.

2.1 Phase 1

2.2 Zitierweise

Dies ist ein Beispiel für ein wörtliches Zitat:

> *"A persona is a rich picture of an imaginary person who represents your core user group."* (Dix u. a., 2004)

Alternativ könnte man auch schreiben:

> *„A persona is a rich picture of an imaginary person who represents your core user group.“* (Dix u. a., 2004)

Das Ergebnis im Dokument ist hier gleich.

Manchmal möchte man die Autorennamen im Text verwenden. Bislang haben wir den `citep{}` Befehl verwendet. Hierdurch werden die Klammern um Autorname und Jahr gesetzt. Man kann den `cite` Befehl allerdings auch variieren:

Dix u. a. (2004) definieren das Konzept "Persona"wie folgt:

> *"A persona is a rich picture of an imaginary person who represents your core user group."* (Dix u. a., 2004)

Das richtige Setzen der Klammern erhöht die Lesbarkeit.

Im APA Format[1] gibt es einige Regeln, wann und wie man Seitenzahlen bei den Literaturverweisen verwendet:

> *"Include page numbers for any citations in the text of your paper that include direct quotations or refer to a specific part of the work you are referencing. Direct quotations must include a page number as part of the citation. The quoted material should be followed by a citation in parentheses that gives the author's name, the year in which the work was published, and the page number from which the quoted material appears."* (Hall, 2013)

[1] American Psychological Association (APA)

Weitere Beispiele und Empfehlungen von Hall (2013) finden Sie hier: `http://www.ehow.com/how_5689799_cite-numbers-apa-format.html`. In LaTeX kann man die Seitenzahlen sehr einfach hinzufügen, zum Beispiel:

Baddeley u. Hitch (1974, S. 86) führen aus:

> *"We hope that our preliminary attempts to begin answering the question will convince the reader, not necessarily that our views are correct, but that the question was and is well worth asking"* (Baddeley u. Hitch, 1974, p. 86)

Im ersten Verweis auf Baddeley u. Hitch haben wir `citet[]{}` verwendet, um die Klammern um das Jahr zu setzen. Im zweiten Fall im Anschluss an das Zitat wurde `citep[]{}` verwendet.

2.3 Verweise innerhalb des Dokumentes

Wenn Sie auf Ihre eigenen Kapitel, Abbildungen, Tabellen o.ä. verweisen wollen, können Sie den `ref{}` Befehl verwenden, wie auch bereits vorab gezeigt im Abschnitt **??** auf Seite **??**.

An dieser Stelle möchten wir nun auf die MI Box verweisen. Sie hat die Abbildungsnummer **??** und ist auf Seite **??** zu finden. Wie Sie an diesem Beispiel auf dieser Seite sehen, kann man mit LaTeX nicht nur auf die Abschnitts-/ oder Tabellennummer verweisen, sondern auch auf die Seitenzahl auf der das Label verweist. Wenn Sie diese dynamischen Codes verwenden (statt zum Beispiel die Nummerierung manuell einzutragen), haben Sie immer die korrekte Nummerierung, auch wenn Sie den Text später umstrukturieren. Wir nutzen den `pageref{}` Befehl hierfür.

3 Weitere Kapitel

3.1 Fußnoten und Listen

Fußnoten können zum Teil sehr nützlich sein. Bitte beachten Sie, dass bei übermäßiger Verwendung von Fußnoten, die Lesbarkeit einschränkt sein kann[1].

3.1.1 Beispiel Unterabschnitte

Sie können in LaTeX Unterabschnitte verwenden. Wenn Sie allerdings einen Unterabschnitt einfügen, sollten es mindestesns zwei sein. Es ist unüblich, dass man beispielsweise nur einen Abschnitt in einem Kapitel hat, oder nur einen Unterabschnitt in einem Abschnitt. Siehe zum Beispiel folgenden Tipp von Dave Patterson:

> *"Its strange to have a single subsection (e.g., 5.2.1 in section 5.2). Why do you need to number it if there is only one? Either eliminate the single subsection, or change the part that precedes the subsection into a second subsection"* (Patterson, 2013)

3.1.2 Listen

Hier folgen einige Beispiele für Listen. Zunächst eine nicht nummerierte Liste:

- Item 1
- Item 2
- Item 3

Nun eine nummerierte Liste:

1. Item 1
2. Item 2
3. Item 3

Man kann auch Symbole verwenden:

→ Item 1

→ Item 2

→ Item 3

[1] da der Lesefluss unterbrochen wird!

Beispiel für Unter-Unterabschnitt

Die gängige Gliederungstiefe von 3 Ebenen (Kapitel, Abschnitt, Unterabschnitt) sollte in der Regel nicht unterschritten werden. Sie können zwar eine weitere Ebene tiefer gehen, da dies ggf. die Lesbarkeit verringert wird diese in LaTeX nicht automatisch im Inhaltsverzeichnis aufgeführt. Hier werden beispielhaft Unter-Unterabschnitte verwendet.

Der folgende Text ist lediglich ein Platzhalter: *Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet.*

Noch ein Unter-Unterabschnitt

Der folgende Text ist lediglich ein Platzhalter, welchen man auch automatisch mit dem Paket "Lipsum" generieren kann: *Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.*

Paragraph als Alternative Für diesen Abschnitt wurde nicht der `subsubsection{}` Befehl verwendet, sondern "paragraph", der auch verwendet werden kann, um einen Unterabschnitt zu generieren. Verglichen zum `subsubsection{}` Befehl beginnt der Text hier nicht in einer neuen Zeile, sondern direkt nach der Überschrift. Wenn Ihr Dokument eher kurz gehalten ist, kann dies kann eher angemessen sein.

3.2 description-Umgebung

Wenn Sie bestimmte Konzepte beschreiben wollen, ist eine Liste oder ein Unterabschnitt ggf. nicht der beste Weg. Als Alternative gibt es außerdem die *description* Umgebung, die hier nützlich sein kann.

Konzept A Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet.

Konzept B Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua.

Abbildungsverzeichnis

Tabellenverzeichnis

Literaturverzeichnis

[Baddeley u. Hitch 1974] Baddeley, Alan D. ; Hitch, Graham: Working memory. In: *Psychology of learning and motivation* (1974), S. 47–89

[Dix u. a. 2004] Dix, Alan ; Finlay, Janet E. ; Abowd, Gregory D. ; Beale, Russell: *Human-Computer Interaction*. 3. Essex, England : Pearson Education Limited, 2004. – ISBN 0-13-046109-1

[Hall 2013] Hall, Shane: *How to Cite Page Numbers in APA Format*. `http://www.ehow.com/how_5689799_cite-numbers-apa-format.html`. Version: 2013. – Last accessed 16 July 2013

[Patterson 2013] Patterson, Dave: *Dave Patterson's Writing Advice*. `http://www.cs.berkeley.edu/~pattrsn/talks/writingtips.html`. Version: 2013. – Last accessed 16 July 2013

Eidesstattliche Erklärung

Ich versichere, die von mir vorgelegte Arbeit selbständig verfasst zu haben.

Alle Stellen, die wörtlich oder sinngemäß aus veröffentlichten oder nicht veröffentlichten Arbeiten anderer entnommen sind, habe ich als entnommen kenntlich gemacht. Sämtliche Quellen und Hilfsmittel, die ich für die Arbeit benutzt habe, sind angegeben.

Die Arbeit hat mit gleichem Inhalt bzw. in wesentlichen Teilen noch keiner anderen Prüfungsbehörde vorgelegen.

Gummersbach, 8. Juni 2021

Max Mustermann