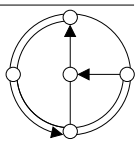
ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich



SS 2007 Prof. R. Wattenhofer / Michael Kuhn / Remo Meier

Vernetzte Systeme
Lösungsvorschlag 5

1 IM-Service

Als Beispiel für einen einfachen Dienst haben wir uns einen *Quote-of-the-Day*-Service (QotD) ausgedacht und in Java implementiert, auch wenn es sicherlich spannendere und komplexere Dienste gibt. Der Quellcode befindet sich wie immer auf unserer Homepage.

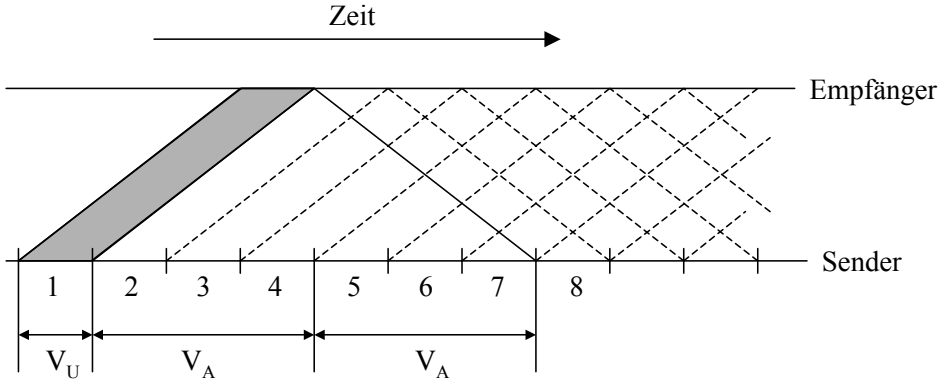
Wir haben den `RegistrationClient` aus Übung 4 genommen, dort eine `main`-Methode eingefügt und das Ganze dann `QuoteOfTheDayService` genannt. Einmal in der Minute registriert sich der Dienst beim Server mit dem Namen `SERVICE:QOTD`.

Die neue Klasse `Messenger` beinhaltet die Funktionalität vom vorherigen `MessageReceiver` und `MessageSender`. Wird eine Nachricht empfangen, so wird diese in der `run`-Methode des Messengers untersucht. Entspricht diese Nachricht dem Text `get qotd`, so wird zufällig einer der mehr oder weniger lustigen, gespeicherten Sprüche bestimmt und an den Absender zurückgeschickt. Handelt es sich um die `help` Nachricht oder eine nicht spezifizierte Anfrage, so wird ein kurze Beschreibung verschickt, wie in Teilaufgabe c) gefordert wurde. Der Absender wird dabei aus dem empfangenen Paket ermittelt und nicht in der Benutzerliste gesucht.

2 Minimale Fenstergrösse

Über eine Glasfaserstrecke von d=5000 km mit einer Bandbreite von R=2 Gbps werden Datenblöcke der Grösse L=1000 Byte mit dem Sliding-Window-Protokoll gesendet. Die Ausbreitungsgeschwindigkeit s in Glasfaser sei s=$\frac{2}{3}$c (c=Lichtgeschwindigkeit).

a) Entscheidend für die Fenstergrösse ist die Verzögerung V, die zwischen dem Absenden eines Blocks und dem Eintreffen der Bestätigung für diesen Block beim Sender vergeht. Für einen kontinuierlichen Datenstrom muss der Sender während der Zeit V senden können, ohne auf Bestätigungen warten zu müssen. Wir nehmen hier an, dass sich V nur aus der Übertragungsverzögerung $V_U = \frac{L}{R}$ und der Ausbreitungsverzögerung $V_A = \frac{d}{s}$ zusammensetzt. Zusätzlich sei die Grösse der Bestätigung vernachlässigbar klein, so dass deren Übertragungsverzögerung nicht berüchsichtig werden muss. Also ist $V = V_U + V_A + V_A$, da ein Block erst bestätigt werden kann, wenn er ganz angekommen ist.



Während der Zeit V können $n = V \cdot R = L + 2 \cdot V_A \cdot R$ viele Bytes gesendet werden, was einer Sendefenstergrösse von $w = \frac{n}{L} = 2\frac{V_A \cdot R}{L} + 1$ entspricht.
Mit den Zahlen aus der Aufgabenstellung erhält man für $V_A = \frac{5 \cdot 10^6 m}{2 \cdot 10^8 m/s} = 0.025s$ und damit für die Sendefenstergrösse $w = 2\frac{0.025s \cdot 2Gbps}{8 \cdot 1000b} + 1 = 2\frac{50 \cdot 10^6 b}{8 \cdot 1000b} + 1 = 12501$ Blöcke.

b) Für eine ideale Verbindung (keine Fehler, keine zusätzlichen Verzögerungen) macht es keinen Sinn, das Sendefenster noch grösser zu machen, da dann nur unnütz Pufferspeicher verschwendet wird, ohne die Ausnutzung der Bandbreite zu erhöhen.
Wenn die Verbindung jedoch eine hohe Fehlerrate aufweist, dann wird der kontinuierliche Datenfluss durch fehlerhaft übertragene Blöcke ins Stocken geraten, da der Sender durch die fehlende Bestätigung sein Sendefenster nicht weiterrücken kann und einfach bis zum Timeout wartet. Bei einem grösseren Sendefenster kann er weitersenden anstatt zu warten. Ein ähnliches Phänomen tritt auf, wenn Blöcke zusätzlich (beispielsweise in Routern) verzögert werden.

3 TCP Fenstergrösse und Effizienz

a) Wie Sie in der vorangehenden Aufgabe gesehen habe, sollte die Empfangsfenstergrösse idealerweise eine bestimmte Grösse haben. Dann kann der Sender ständig senden, und es ergibt sich eine hohe Effizienz. Da in TCP jedem Byte eine Sequenznummer zugeordnet ist, bedeutet eine 16-Bit-Fenstergrösse, dass maximal 65535 Bytes gesendet werden können, bevor auf eine Bestätigung des Empfängers gewartet werden muss. Bei hohen Verzögerungen und hohen Bandbreiten könnten aber viel mehr Daten "unterwegs" sein, als diese 65535 Bytes. Daher ist die Effizienz des ursprünglichen TCP in diesem Fall sehr gering.

b) Geostationäre Satelliten befinden sich in 36000 km Höhe. Daher gilt für die Ausbreitungsverzögerung $V_A = 2 \cdot \frac{36000km}{3 \times 10^8 \frac{m}{s}} = 0.24s$, und wir können die Roundtrip-Zeit approximieren zu $RTT = 2 * V_A = 0.48s$. Wir nehmen an, mit dem empfangenen SYN-Segment wurde die maximale Fenstergrösse von $w = 65535$ Bytes mitgeteilt. Der Sender kann dann w Bytes übertragen, bevor er auf ein Acknowledgement warten muss.

Die Übertragung dieser w Bytes dauert $t_{send} = \frac{65535Bytes}{100Mbps} = \frac{65535 \cdot 8b \cdot s}{100 \cdot 10^6 b} = 5.24ms$. Das heisst, der Sender kann immer 5.24 ms lang senden, dann muss er auf eine Bestätigungen warten. Diese trifft nach der Zeit RTT ein. Damit kann die Effizienz U approximiert werden zu $U = \frac{t_{send}}{t_{send} + RTT} = 1.08\%$. Der tatsächliche Durchsatz über die 100 Mbps-Verbindung liegt also nur bei rund 1 Mbps.

c) Das Problem ist in TCP gelöst worden, indem in RFC 1323 eine *Window Scale Option (WSO)* eingeführt wurde. Diese wird beim Verbindungsaufbau zusammen mit dem SYN-Segment geschickt und kann während der Verbindung nicht mehr geändert werden. Eine TCP-Implementierung, die diese Option nicht kennt, ignoriert sie einfach. Der Sender der Option bekommt in diesem Fall vom Empfänger keine WSO zurück und verwendet das ursprüngliche Verfahren.

In der WSO wird die Anzahl der Bits angegeben, um die der Empfänger den Wert im Window-Feld nach links shiften muss (was einer Multiplikation mit einer 2er-Potenz entspricht), um ihn richtig zu interpretieren.