Technische Universität München
Informatik VIII
Netzarchitekturen und Netzdienste
Prof. Dr.-Ing. Georg Carle

Heiko Niedermayer, Marc Fouquet

Übungen zur Vorlesung Rechnernetze und Verteilte Systeme

Übungsblatt 0, SS 2010

Tutorenübung

Vielen Dank an die Tutoren und ihre Hilfe beim Erstellen und Vorrechnen dieser einführenden Übung.

Viel Spaß und Erfolg bei der Übung!

Aufgabe 1 - Daten per LKW

Um den Animationsfilm in Münchnen zu Fördern wird eine Kooperation zwischen dem Hochleistungsrechenzentrum in Garching und den Bavariafilmstudios geschlossen. Statt einer Leitung sollen LKWs einer Spedition die Daten vom Rechenzentrum in Garching zu den Bavariafilmstudios in Grünwald bringen. Um die Stadt nicht zu sehr zu belasten, fahren die LKWs den Weg zwischen Garching und Grünwald über A9 und A99, was einer Länge von 52 km entspricht. Im Mittel kann ein LKW die Strecke mit 55 km/h befahren.

Der LKW werde mit einer Rate von 12 Festplatten pro Minute beladen und mit einer Rate von 15 Festplatten pro Minute entladen. Die Kapazität des LKWs sei 512 Festplatten. Zur Anwendung kommen Festplatten mit einer Kapazität von 1 TByte.

a) Wie lange dauert das Beladen des LKWs?

b) Wie lange dauert es, bis die Daten beim Filmstudio angekommen und entladen?

c) Welcher Datenrate entspricht dies?

d) Angenommen, es stehen genug LKWs zur Verfügung, so dass nach 2 min Pause bereits der nächste LKW beladen werden kann. Welche Datenrate ist jetzt zu erreichen?

Aufgabe 2 - In der Fabrik

In einer Fabrik wird ein Produkt produziert, welches mehrere Arbeitsschritte erfordert. Es besucht die Maschinen A und B sowie die Testeinheit C. Die Bearbeitungszeiten:

Maschine	Bearbeitungszeit
A	51 s
B	27 s
C	11 s

Ein Roboter wartet jeweils an der bearbeitenden Maschine und bringt dann das Produkt nach Fertigstellung des Arbeitsschritt zur nächsten Maschine. Die Laufwege sind nicht symmetrisch. Die Zeiten sind:

Von	zu Lager	zu A	zu B	zu C
Lager	-	11 s	15 s	9 s
A	21 s	-	6 s	13 s
B	15 s	11 s	-	4 s
C	11 s	11 s	9 s	-

a) Zeichnen Sie ein Weg-Zeit-Diagramm und bestimmen Sie, wie lange das Produkt vom Lager bis zur Fertigstellung an Maschine C braucht.

Die Testeinheit C überprüft am Ende das Produkt. In 40 % der Fälle stellt es einen Fehlerfall fest und veranlasst Nacharbeit bei Maschine B. Die Nacharbeit von Maschine B dauert 15 s.

a) Modifizieren Sie das Weg-Zeit-Diagramm, für den Fall, dass eine Nacharbeit notwendig ist. Wie lang dauert es jetzt?

b) Wie oft muss 3x nachgearbeitet werden? Welche Verteilung liegt dem Zugrunde? Wie oft muss im Mittel nachgearbeitet werden?

c) Nun soll maximal 5x nachgearbeitet werden. Betriebsliche wie technische Gründe könnten dafür ausschlaggebend sein. Wie oft entstehen dann fehlerhafte Produkte, die entsorgt werden müssen?

d) Was müssten Sie tun, um die mittlere Bearbeitungszeit von fehlerhaften wie korrekten Produkten zu berechnen?

Aufgabe 3 - Datenübertragung und Signaldarstellung

In dieser Aufgabe wird die Codierung und Übertragung einer Zeichensequenz untersucht. Der Text „Rechnernetze" soll ASCII codiert werden, wobei wir ein ASCII Zeichen durch 8 Bit kodieren[1] (siehe Tabelle 1). Anschließend soll die kodierte Nachricht auf einer mehradrigen Leitung übertragen werden.

a) Wie viele unterschiedliche Textzeichen können mit Datenwrtern bestehend aus n Bit dargestellt werden?

b) Geben Sie die zu kodierende Zeichenkette in binärer Schreibweise an.

Die in Teilaufgabe (b) erzeugte binäre Sequenz soll nun über eine vieradrige Leitung übertragen werden. Jede Ader übertragt 1000 Symbole pro Sekunde (Schrittgeschwindigkeit). Die Signalform sei ein idealer Rechteckimpuls wobei eine logische „Null" durch eine Spannung von $+5$ V und eine logische „Eins" durch -5 V dargestellt werde.

c) Welche Übertragungsrate erwarten Sie unter optimalen Bedingungen?

d) Skizzieren Sie das resultierende Zeitsignal $s(t)$ für alle vier Leitungsadern.

e) Wie wirkt sich eine Erhöhung der Schrittgeschwindigkeit oder eine Vergrößerung der Aderzahl aus? Welche Schwierigkeiten sind zu erwarten?

Bislang wurde die idealisierte Annahme von Rechteckimpulsen getroffen. Es soll nun untersucht werden, weswegen ideale Rechteckimpulse keine gute Wahl darstellen. Dazu betrachten wir den idealisierten Rechteckimpuls $r(t)$, der im Zeitbereich gegeben ist als

$$r(t) = \begin{cases} \frac{1}{T} & -\frac{T}{2} \leq t \leq \frac{T}{2} \\ 0 & \text{sonst} \end{cases} .$$

Durch bergang in den Frequenzbereich mittels zeitkontinuierlicher Fouriertransformation können die in $r(t)$ enthaltenen Freuqenzanteile bestimmt werden. Dementsprechend ist der Rechteckimpuls im

[1] Die ASCII Kodierung sieht lediglich 7 Bit pro Zeichen vor.

ASCII (hex)	Zeichen	ASCII (hex)	Zeichen	ASCII (hex)	Zeichen	ASCII (hex)	Zeichen
00	NUL	20	SP	40	@	60	`
01	SOH	21	!	41	A	61	a
02	STX	22	"	42	B	62	b
03	ETX	23	#	43	C	63	c
04	EOT	24	$	44	D	64	d
05	ENQ	25	%	45	E	65	e
06	ACK	26	&	46	F	66	f
07	BEL	27	'	47	G	67	g
08	BS	28	(	48	H	68	h
09	TAB	29	)	49	I	69	i
0A	LF	2A	*	4A	J	6A	j
0B	VT	2B	+	4B	K	6B	k
0C	FF	2C	,	4C	L	6C	l
0D	CR	2D	-	4D	M	6D	m
0E	SO	2E	.	4E	N	6E	n
0F	SI	2F	/	4F	O	6F	o
10	DLE	10	0	50	P	70	p
11	DC1	11	1	51	Q	71	q
12	DC2	12	2	52	R	72	r
13	DC3	13	3	53	S	73	s
14	DC4	14	4	54	T	74	t
15	NAK	15	5	55	U	75	u
16	SYN	16	6	56	V	76	v
17	ETB	17	7	57	W	77	w
18	CAN	18	8	58	X	78	x
19	EM	19	9	59	Y	79	y
1A	SUB	1A	:	5A	Z	7A	z
1B	Esc	1B	;	5B	[	7B	{
1C	FS	1C	<	5C	\	7C	|
1D	GS	1D	=	5D	]	7D	}
1E	RS	1E	>	5E	^	7E	~
1F	US	1F	?	5F	_	7F	DEL

Tabelle 1: ASCII character set.

Frequenzbereich gegeben als

$$R(f) = \mathcal{F}(r(t)) = \int_{-\infty}^{\infty} e^{-j2\pi ft} \cdot r(t)dt.$$

f) Berechnen Sie $R(f)$.

g) Skizzieren Sie $r(t)$ und $R(f)$ im Intervall $[-4T; 4T]$ fr $T = 1$.

h) Ist das Signal $r(t)$ bandbegrenzt? Worin liegt demnach das Problem?

Im Folgenden geben wir Ihnen ein paar einfache mathematische Hilfsmittel an die Hand, die z.T. hier in den Aufgaben Verwendung finden können und auch später auf anderen Aufgabenblättern von Nutzen sein können.

Geometrische Folge und geometrische Reihe

In einer geometrischen Folge $a_0 = a$, $a_1 = \frac{a}{q}$, $a_2 = \frac{a}{q^2}$, $a_3 = \frac{a}{q^3}$, ... unterscheiden sich nachfolgende Folgenglieder immer um einen festen Quotienten q. Ein Beispiel ist die Folge 1, $\frac{1}{2}$, $\frac{1}{4}$, $\frac{1}{8}$, usw.

Eine geometrische Reihe ist eine Folge, deren Folgeglieder s_i die Summe der Folgeglieder der geometrischen Folge a_0 bis a_i sind.

$$s_n = a_0 \sum_{k=0}^{n} q^k = a_0 \frac{q^{n+1}-1}{q-1} = a_0 \frac{1-q^{n+1}}{1-q}$$

Interessant ist dann immer der Grenzwert s der geometrischen Reihe. Sofern Konvergenz vorliegt, gilt:

$$s = \frac{a_0}{1-q}$$

Wahrscheinlichkeiten

Wahrscheinlichkeiten und Zufälle spielen in der Kommunikationstechnik eine nichtzunachlässigende Rolle. Fehler oder Überlastsituationen geschehen nicht geplant, sondern zufällig. Menschen kommunizieren in zufällig erscheinender Weise. In der Vorlesung betrachten wir dazu i.a. nur einfache Situationen.
Nehmen wir an, S sende ein Paket über M an E. Mit $p_1 = 0,3$ Wahrscheinlichkeit (30 %) gehe dies auf dem Weg nach M verloren, d.h. mit Wahrscheinlichkeit $q_1 = 1 - p_1 = 0,7$ kommt dies bei M an (→ **Wahrscheinlichkeit, dass p nicht eintritt, ist $1 - p$**).

Jetzt soll das Paket aber noch bei E ankommen. Mit Wahrscheinlichkeit $p_2 = 0,2$ (20 %) gehe dies auf dem Weg von M nach E verloren. Mit welcher Wahrscheinlichkeit kommt es an? Dazu müssen zwei Bedingungen erfüllt sein. 1. Es muss bei M ankommen und 2. darf auf dem Weg M nach E nicht verloren gehen ($q_2 = 1 - p_2 = 0,8$). **Wenn mehrere Bedingungen gelten müssen und diese Bedingungen unabhängig voneinander sind (davon gehen wir hier i.a. aus), so werden die Wahrscheinlichkeiten der einzelnen Bedingungen multipliziert.** Ergebnis: Sei p_E die Wahrscheinlichkeit, dass E erreicht wird. $p_E = q_1 * q_2 = 0,7 * 0,8 = 0,56$. Mit 56 % Wahrscheinlichkeit wird E erreicht.

Geometrische Verteilung

Die geometrische Verteilung gibt an, wie oft man etwas versuchen muss, bis ein Erfolg eintritt. Erfolgswahrscheinlichkeit sei p und $q = 1 - q$ die Misserfolgswahrscheinlichkeit. Es gilt dann:
$x(iFehlversuchebisErfolg) = q^i p$

Erwartungswert $E[X] = \frac{q}{1-q}$ und Variation $c_x = \frac{1}{\sqrt{q}}$

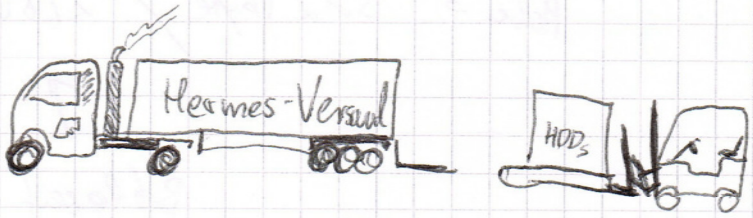
Binomialverteilung

Die Binomialverteilung gibt an, wie oft ein Erfolg bei N Versuchen auftritt. Erfolgswahrscheinlichkeit sei p und $q = 1 - q$ die Misserfolgswahrscheinlichkeit. Es gilt dann:
$x(i, N) = \binom{N}{i} p^i q^{N-i}$

Erwartungswert $E[X, N] = Np$ und Variation $c_x = \sqrt{\frac{q}{Np}}$

Rechnübung 3 + ≠ 3

1.



$\leftarrow$ 52 km $\rightarrow$

$\leftarrow\!\rightarrow$ 55 km/h $\rightarrow$

$\Rightarrow \frac{52}{55}$ h

beladen 12 $\frac{HDD}{min} = \frac{1}{5}\frac{HDD}{sek}$

entladen 15 $\frac{HDD}{min} = \frac{1}{4}\frac{HDD}{sek}$

Kapazität = 512 · 1 Tbyte = 512 Tbyte

a) Beladen: ~~1 HDD~~ 512 HDD / $\frac{1}{5}\frac{HDD}{sek}$ = ~~2560 sek~~

2560 sek ≈ 42 min

Entladen: 512 HDD / $\frac{1}{4}\frac{HDD}{sek}$

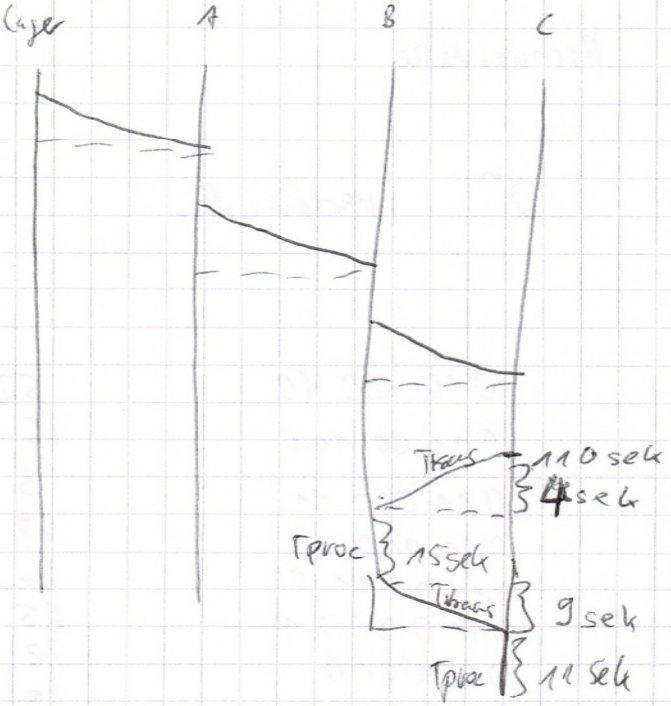
2048 sek = 34 min

b) Dauer = 2560 sek + $\frac{52}{55}$ · 3600 sek + 2048 sek =

= 8011 sek ≈ 2h 14 min

c) $\frac{512\ Tbyte}{8011\ sek} = 65{,}49\ Gbyte/sek$

a)



$a = 110 + 4 + 15 + 9 + 11 = 149$

b)

FAIL	WIN (epic)
$(0,4)^3$	$\cdot\ 0,6$ → $= 3,84\%$

(Genau 3x)

Wie oft?

Verteilung? Binomial-V.

oder

Geometrische V. (unabhängige Ereignisse)

(wie oft versuchen, bis es geht)

Weil: ...

Erwartungswert (mittlerer Erfolg) $= \frac{1-p}{p} = \frac{0,4}{0,6} = 66,7\%$

c) $P = (0,4)^5 \cdot 0,4 = (0,4)^6 = 0,41\%$

d) $T_{AVG} = \sum_{Fall} \cdot p_{Fall} \cdot t_{Fall}$

d)

Rate = 512 Tbyte / 2680 sek = 195,63 GByte/sek

↑

Bedarfszeit und 2 Minuten warten

WTF?

Die erste Ladung verzögert sich um Entladezeit, danach alle 2 Minuten kommen Daten...

2.

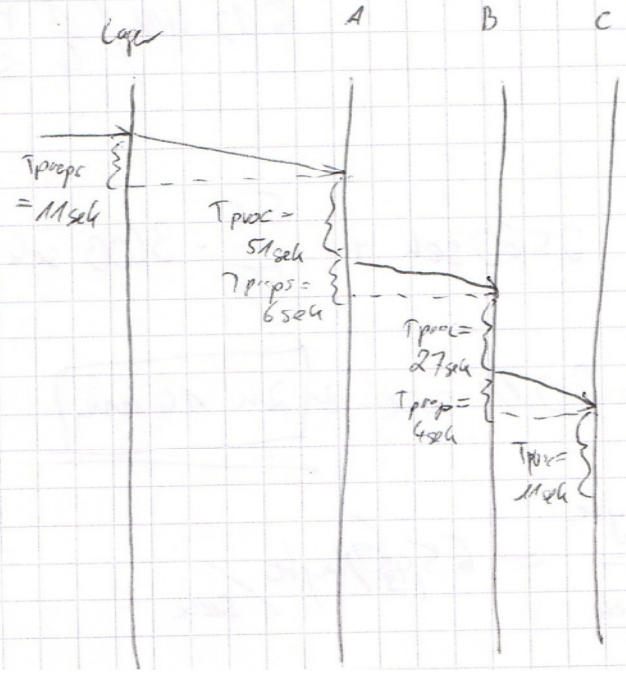
Maschine

A 51 sek

B 27 sek

C 11 sek

a)



Tisk Zeit

$T_{ges} = T_{trans} + T_{prop}$
$(+ T_{proc} + T_{queue})$

a = 11 + 51 + 6 + 27 + 4 + 11 = 110 sek

3.

Rechnernetze

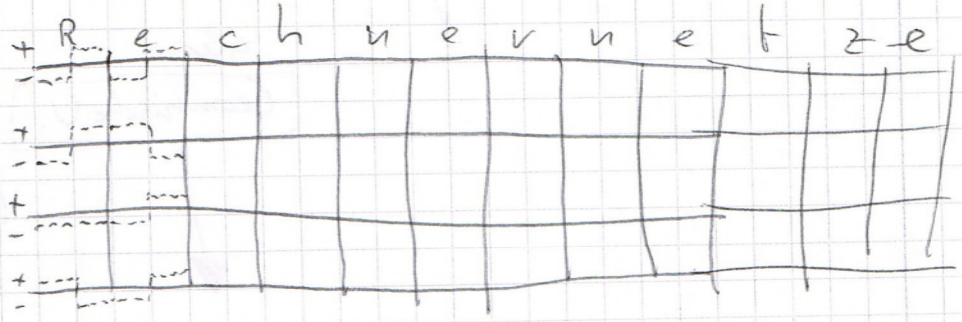
a) 2^n verschieden

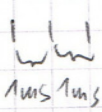
b)

010 100 10	52
011 00 101	65
011 000 11	63
011 010 00	68
011 0 111 0	6e
011 0 010 1	65
011 0 010	72
011 0 111 0	6e
011 00 101	65
01110 100	74
011 11 010	7a
011 00 101	65

c) Rate = 4000 $\frac{\text{bit}}{\text{sek}}$ / 8 = 500 $\frac{\text{byte}}{\text{sek}}$

d)





Σ = 12 ms

e) Synchronität, physikalische Probleme

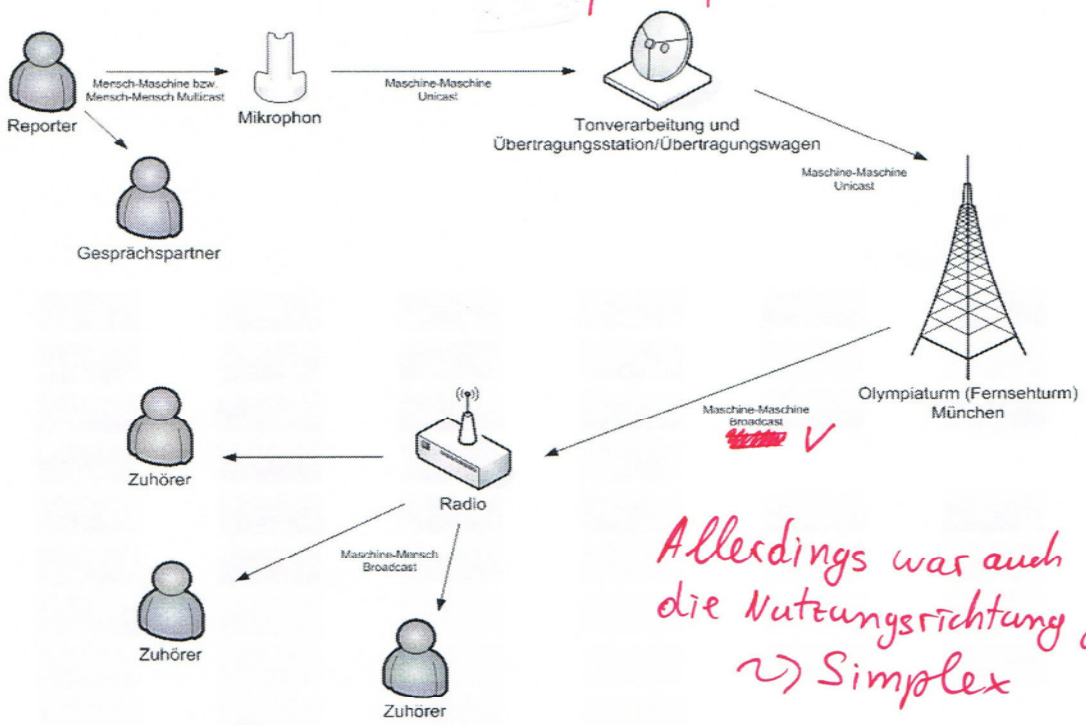
f g h) Nicht so wichtig!

[redacted]

1.

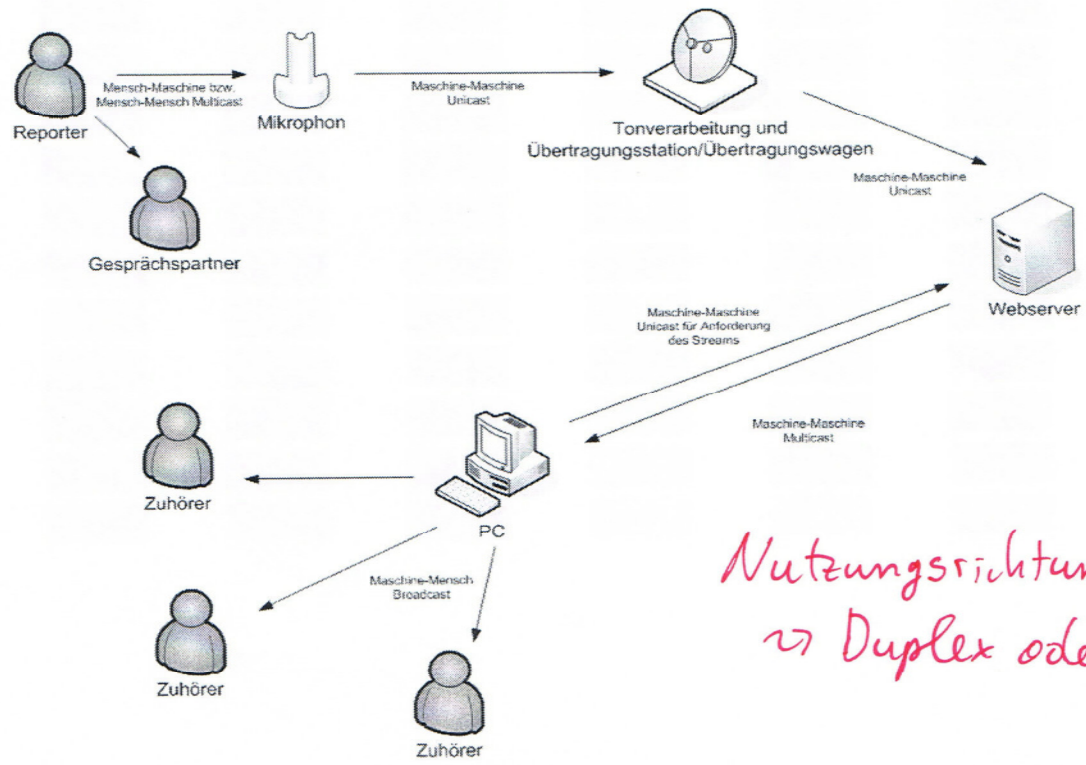
a) 0,5/1

4,5/6 | 5/5 | 12,5/13



Allerdings war auch die Nutzungsrichtung gefragt
↝ Simplex

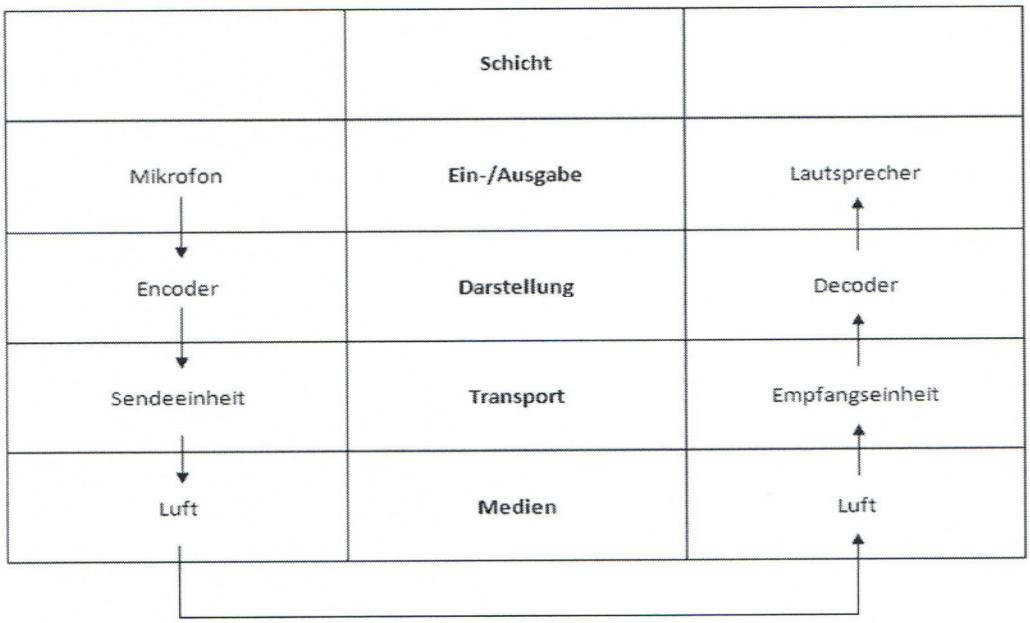
b) 0,5/1



Nutzungsrichtung
↝ Duplex oder Halbduplex

[redacted]

c) 1,5/2



→ Ein Protokoll sprechen Mikrofon mit Lautsprecher, Encoder mit Decoder, Sender und Empfänger

d) 2/2

Die Übertragung mittels Rundfunk ist ungesichert, da keinerlei Mechanismen einen Empfang am Radio sicherstellen. Das Radio könnte auch ausgeschaltet sein, der Fernsehturm sendet trotzdem. ✓

Die Übertragung eines Webstreams ist entweder ungesichert, wenn es sich um eine strombasierte Übertragung (z.B. UDP) handelt, oder gesichert, wenn ein Paket-basiertes Übertragungsprotokoll (z.B. TCP/IP) vom Server zum empfangenden Computer hin verwendet wird. top!

[redacted]

2.

a)

Laufzeit ≈ tprop

1/1 T entspricht der reinen Übertragungszeit eines Signals von S zu M bzw. M zu E.

tN entspricht dem Zeitabstand zwischen Beginn und Ende des Sendevorgangs einer Nachricht von S zu M und gleichzeitig der Dauer, die M benötigt um den Nachrichtentyp zu erkennen und die Weiterleitung der Nachricht zu beginnen. kann man sagen

tP entspricht (analog zu tN) dem Zeitabstand zwischen Beginn und Ende einer Paketübertragung von S zu M und gleichzeitig der Dauer, die M benötigt um Pakettyp zu erkennen und die Weiterleitung des Pakets zu beginnen. kann man sagen

b)

1/1 Übertragungszeit = Zeitfenster zwischen Sendebeginn und Empfangsende

Circuit Switching:	tGes = 2 x T + tN ✓
Message Switching:	tGes = 2 x T + 2 x tN ✓
Packet Switching:	tGes = 2 x T + (n + 1) x tP ✓ (für n Pakete einer Nachricht)

c)

1/1 Die Paketgröße und damit tP muss sich bestmöglich an Null annähern!

Es gilt: n x tP = tN

Für n → Unendlich gilt dann tP → 0

Die Addition (n + 1) fällt dann nicht mehr ins Gewicht!

→ (n + 1) x tP entspricht dann für große n und damit kleine tP nahezu der Zeit tN

d) 2/2

Es können hier generelle Vorteile der Paket-basierten Übertragung angeführt werden, z.B.:

- der Beginn einer Nachricht kann gelesen oder verarbeitet werden, obwohl das Ende noch nicht übertragen wurde
- „verlorengegangene" Pakete könnten anstelle der kompletten Gesamtnachricht erneut gesendet werden
- Die Gesamtübertragungsdauer nähert sich für sinnvoll klein gewählte Paketgrößen der Übertragungsdauer des „Circuit Switching" an

[redacted]

3.

a)

2/2,5 8 Faserpaare, 10.000km, 7,68 TBit/s

vGF = 2/3 x c = 2/3 x 3 x 10^8 m/s = 200.000.000 m/s

1 Faserpaar → 1/8 x 7,68TBit/s = 0,96TBit/s = 960 GBit/s

Zeit für Signal → 10.000.000m / 200.000.000m/s = 0,05s

Gesamtkapazität des Kabels → 7,68TBit/s x 0,05s = 384GBit ≈ 44,703 GiB . 2 Hin-

Gesamtkapazität eines Faserpaars → 384GBit / 8 = 48GBit ≈ 5,588 GiB . 2 und zurück

b)

2,5/2,5 Gesamtlänge eines Bits → 48GBit / (64 x 10.000.000m) = 75Bit/m ✓
→ 1 / (75m/Bit) = 1⅓cm/Bit

c)

1/1 Fast-Ethernet 100MBit/s, 2 Faserpaare

vK = 2/3 x c = 2/3 x 3 x 10^8 m/s = 200.000.000 m/s

Gesamtlänge eines Bits → 200.000.000m/s / 100.000.000 Bit/s = 2m/Bit ✓

d)

Wie in Teilaufgabe c) berechnet → Längenunterschied U = 2m - 1⅓cm = 198⅔cm

1/1 → Ein Bit auf Fast-Ethernet ist also 200cm / 1⅓cm = 150 mal so lang als auf dem gegebenen Glasfaserkabel! ✓

e)

2/2 Fast-Ethernet → 2m/Bit x 8Bit/Byte x 1518Byte = 24288m ✓
Glasfaser → 1⅓cm/Bit x 8Bit/Byte x 1518Byte = 161,92m ✓

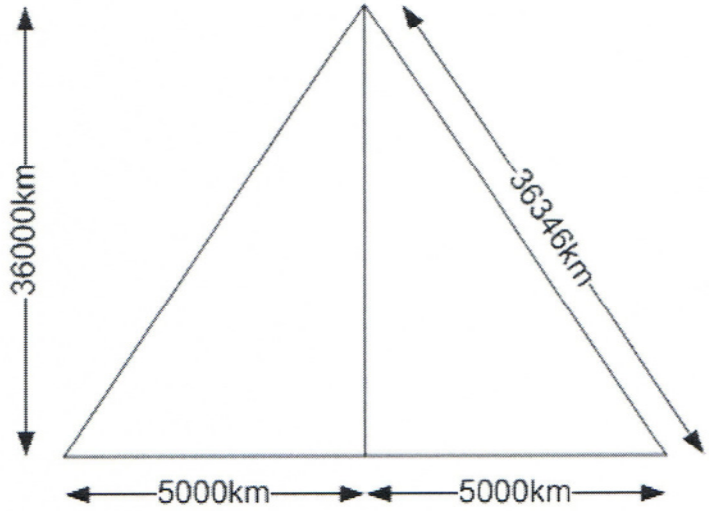
f)

1/1 Glasfaser → RTT = 2 x 0,05s = 0,1s = 100ms ✓

[redacted]

g)

2/2



Abstand Sender/Empfänger zu Satellit → $\sqrt{[(5000km)^2 + (36000km)^2]} \approx 36346km$

Satellit → RTT = 4 x (36346564m) / 300.000.000m/s) ≈ 0,48462 = 484,62ms ✓

Die RTT dauert über Satellitenverbindung etwa 4,85 mal so lange wie über Glasfaser. ✓

h)

1/1

Wie in Teilaufgabe a) berechnet → Gesamtkapazität eines Faserpaars → 48GBit

Dateigröße → 720MiB ≈ 6,040GBit < 48GBit

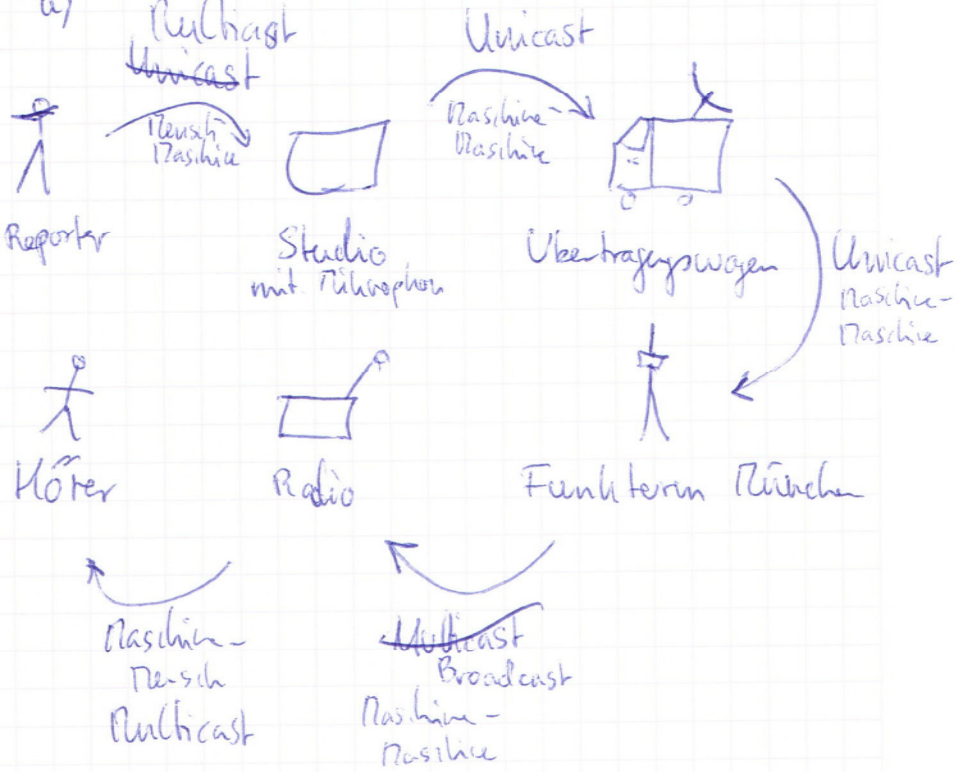
→ Die komplette Datei befindet sich auf dem Faserpaar, bevor das erste Bit beim Empfänger eintrifft. ✓

Management Service

www.tuev-sued.de/tms

Rechenaufgabe 001

1 a)



b) ...

(3)

8 Aderpaare = 16 Adern, 10000 km, 7,68 TBit/sek

Glasfaser: $\frac{2}{3}c = \frac{2}{3} \cdot 3 \cdot 10^8 \frac{m}{s}$

$= 200000000 \frac{m}{s}$

a)

1 Aderpaar $\Rightarrow \frac{1}{8} \cdot 7,68 \text{ TBit/sek}$

$= 0,96 \text{ TBit/sek}$

$= 960 \text{ GBit/sek}$

Zeit für Signal: $\frac{10000 \cdot 1000 \text{ m}}{200000000 \frac{\text{m}}{\text{sek}}} =$

$= \frac{1}{20} \text{ sek} = 0,05 \text{ sek}$

Signale auf Länge: $960 \frac{\text{GBit}}{\text{sek}} \cdot 0,05 \text{ sek}$

$= 48 \text{ GBit}$

Pro Aderpaar $\Rightarrow$ (Hinrichtung: 24 GBit $\approx$ 2,794 GiB; Rückrichtung: dito)

Kabel gesamt $\Rightarrow 7,68 \frac{\text{TBit}}{\text{sek}} \cdot 0,05 \text{ sek} = 384 \text{ GBit} \approx 44,703 \text{ GiB}$

b) pro Ader und Wellenlänge: $\frac{24\ \text{Gbit}}{64 \cdot 10000000\ \text{m}}$

$= 37{,}5\ \frac{\text{bit}}{\text{m}}$

$\Rightarrow$ 1bit hat 2,67 cm

c) Fast-Ethernet $\Rightarrow$ 100 $\frac{\text{MBit}}{\text{sek}}$
(2 Adernpaare)

$v = \frac{2}{3}c = 200000000\ \frac{\text{m}}{\text{sek}}$

Länge $\Rightarrow$ $\frac{200000000\ \frac{\text{m}}{\text{sek}}}{100000000\ \frac{\text{bit}}{\text{sek}}} = 2\ \frac{\text{m}}{\text{bit}}$

$\Rightarrow$ 1bit hat 2 m

$\approx 1{,}97$ m länger

d) 2 Meter

e) $2\ \frac{\text{m}}{\text{Bit}} \cdot 8\ \frac{\text{Bit}}{\text{Byte}} \cdot 1518\ \text{Byte} = 24288$ m Fast Ethernet

$2{,}67\ \frac{\text{cm}}{\text{Bit}} \cdot 8\ \frac{\text{Bit}}{\text{Byte}} \cdot 1518\ \text{Byte} \approx 324{,}24$ m Glasfaser

TÜV SÜD Management Service

www.tuev-sued.de/tms

f)

$$x = \sqrt{36000^2 + 5000^2} \approx 36346\,km$$

Glasfaser: $2 \cdot \frac{10000000\,m}{\frac{2}{3} \cdot 300000000\,\frac{m}{sek}} =$

$= \frac{1}{10}\,sek = \underline{100\,ms}$

g) Sat: $\frac{4 \cdot 36346000\,m}{300\,000\,000\,\frac{m}{sek}} =$

$= 0{,}48\,sek = \underline{480\,ms}$

$\frac{480\,ms}{100\,ms} = \boxed{4{,}8}$

h) Pro Adernpaar 4,8 GBit Kapazität

720178 ≈ ~~6,0398~~ 6,0398 GBit ⇒ komplette Datei unterwegs, bevor sie ankommt

(2) a)

T entspricht der reinen Übertragungszeit eines Signals von S zu M bzw. M zu E, ~~egal wie groß ein Paket ist~~

t_N entspricht dem Zeitabstand zwischen Beginn & Ende von ~~Senden~~ Sendevorgängen von S zu M ~~(Circuit)~~ und gleichzeitig der Dauer, die M benötigt um den Nachrichtentyp zu erkennen und die ~~Nachricht weiterzuleiten~~ Weiterleitung dieser Nachricht zu beginnen

t_p entspricht dem Zeitabstand zwischen dem ein Paket bei M eintrifft und von M weitergeleitet wird

b)

Circuit: $t = 2 \cdot T + t_N$

Message: $t = 2 \cdot T + 2 \cdot t_N$

Packet: $t = 2 \cdot T + (n+1) \cdot t_p$ für n Pakete

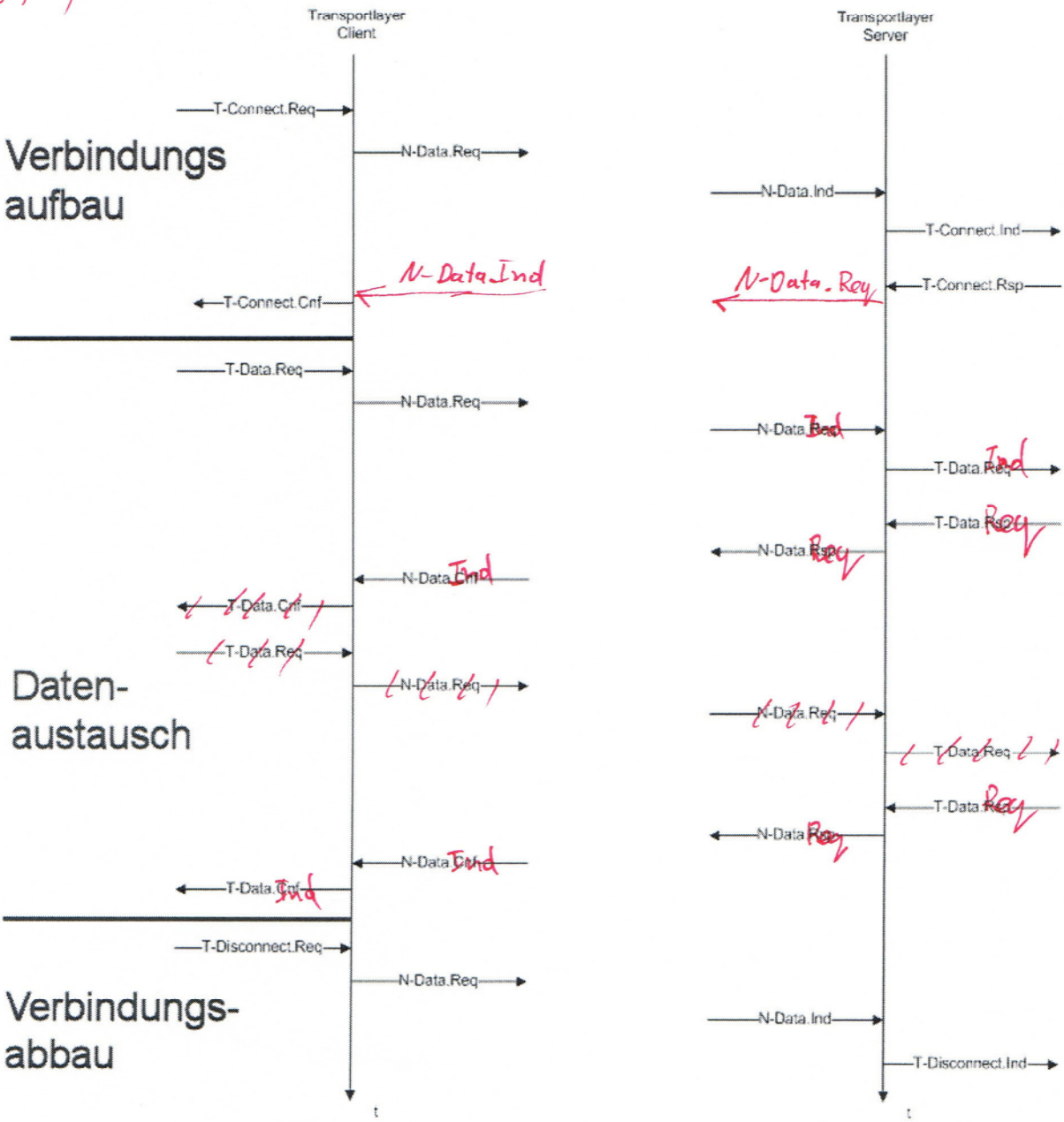
c) Paketgröße und damit t_p muss sich bestmöglich an Null annähern!

$\Rightarrow (n+1) \cdot t_p \approx t_N$

d) Es können hier generelle Vorteile der paketbasierten Übertragung aufgezeigt werden...

4.

(2,5)



A4	A5	A6
3,5	8	11

$\Sigma = 21,5$

[redacted]

5.

a)

(2)

T = Ts + Ta = (l / vs) + (d / c) ✓

b)

(1)

l = 1526 Byte = 12208 Bit
d = 50m
vs = 100.000.000Bit/s = 100MBit
c = 200.000.000m/s

Tges = (12208Bit / 100.000.000Bit/s) + (50m / 200.000.00m/s) = 0,12233ms ✓

Die Serialisierungszeit ist deutlich größer als die Ausbreitungsverzögerung. ✓

c)

(1)

Ta1 → 10MBit-Leitung
Ta2 → 2048KBit-Leitung

Kabellänge → d1 = c x Ta1 = 200.000.00m/s x 0,004s = 800km ✓
d2 = c x Ta2 = 200.000.00m/s x 0,0015s = 300km ✓

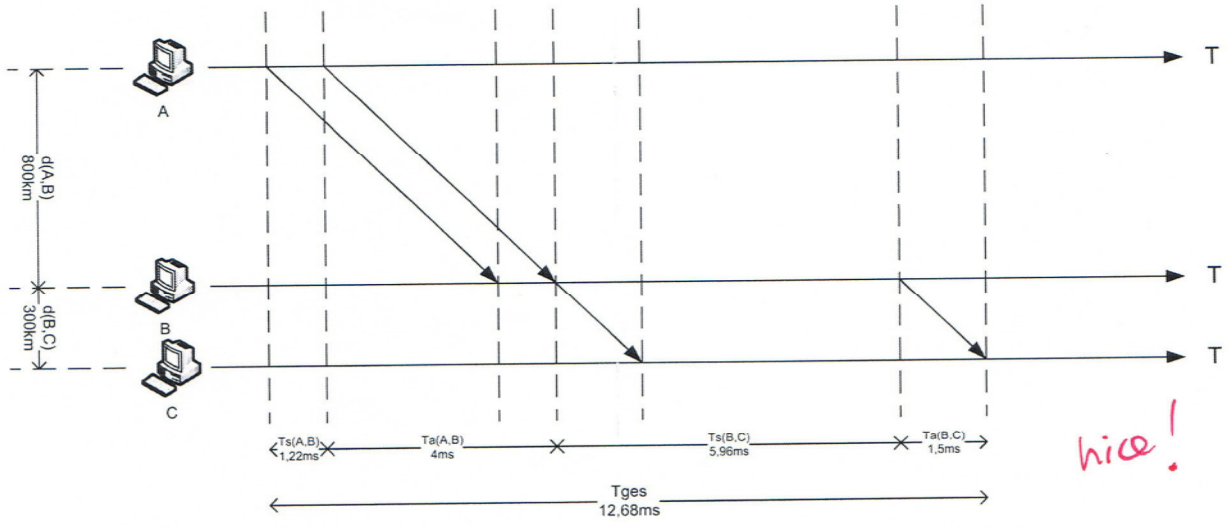
d)

(2)

Tges = Tges1 + Tges2 = (12208Bit / 10.000.000Bit/s) + 4ms +
+ (12208Bit / 2.048.000Bit/s) + 1,5ms =
= 12,68ms ✓

e)

(2)



hice!

[redacted]

6.

a)

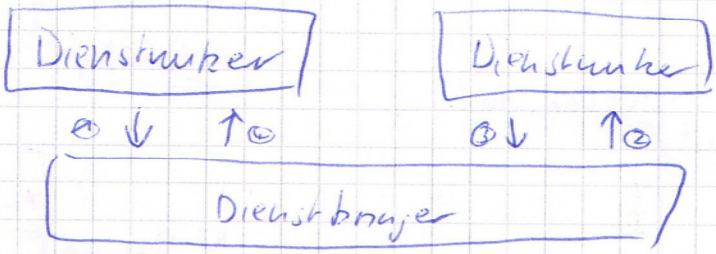
(4)

Der Initiator erzeugt eine Zufallsvariable und schickt diese an den Responder. Dann setzt er seinen Timer auf eine bestimmte Zeit t.

Der Responder wartet währenddessen auf eine Übermittlung. Sobald er die Variable vom Initiator bekommt, ermittelt er selbst eine eigene Zufallsvariable und vergleicht sie mit der zuvor gesendeten: Ist die Eigene kleiner, verwirft er sie und wartet von Neuem auf eine Übermittlung, ist sie größer oder gleich, so sendet er sie an den Initiator und wartet dann.

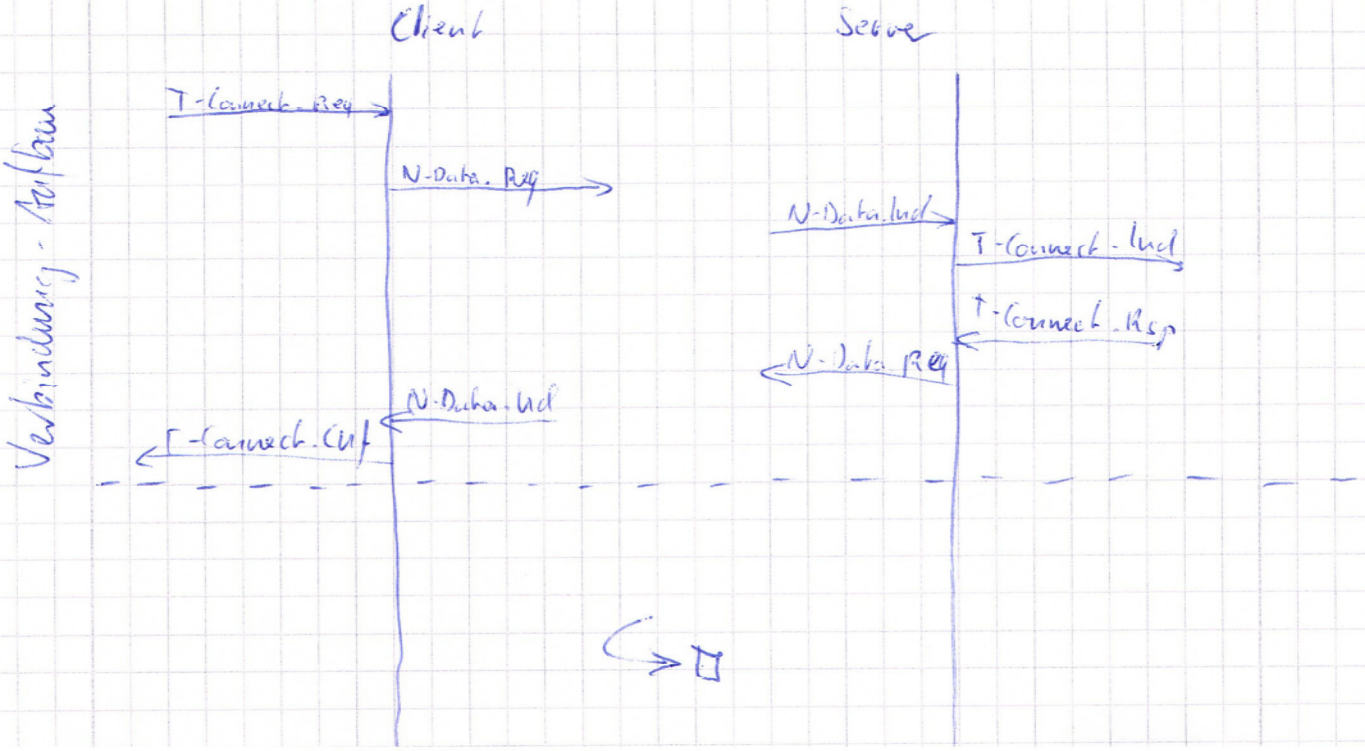
Der Initiator wartet auf die Antwort vom Responder mit einer Ergebnisvariable. Bekommt er diese, bevor sein Timer abläuft, so startet der Vorgang von Neuem und es wird eine neue Zufallsvariable generiert und versendet. Bekommt er diese nicht rechtzeitig vor Timerablauf, wird der Prozess beendet. Er sendet also solange Nachrichten an den Responder, solange er Antworten im Timer-Zeitfenster bekommt - diese Antwort wird allerdings bereits dann nicht bekommen, wenn die Zufallsvariable vom Responder größergleich der Übermittelten ist. Dann würde der Responder auf eine neue Nachricht warten, der Initiator jedoch irgendwann (bzw. nach Ablauf des Timers) terminieren. ✓

(4)

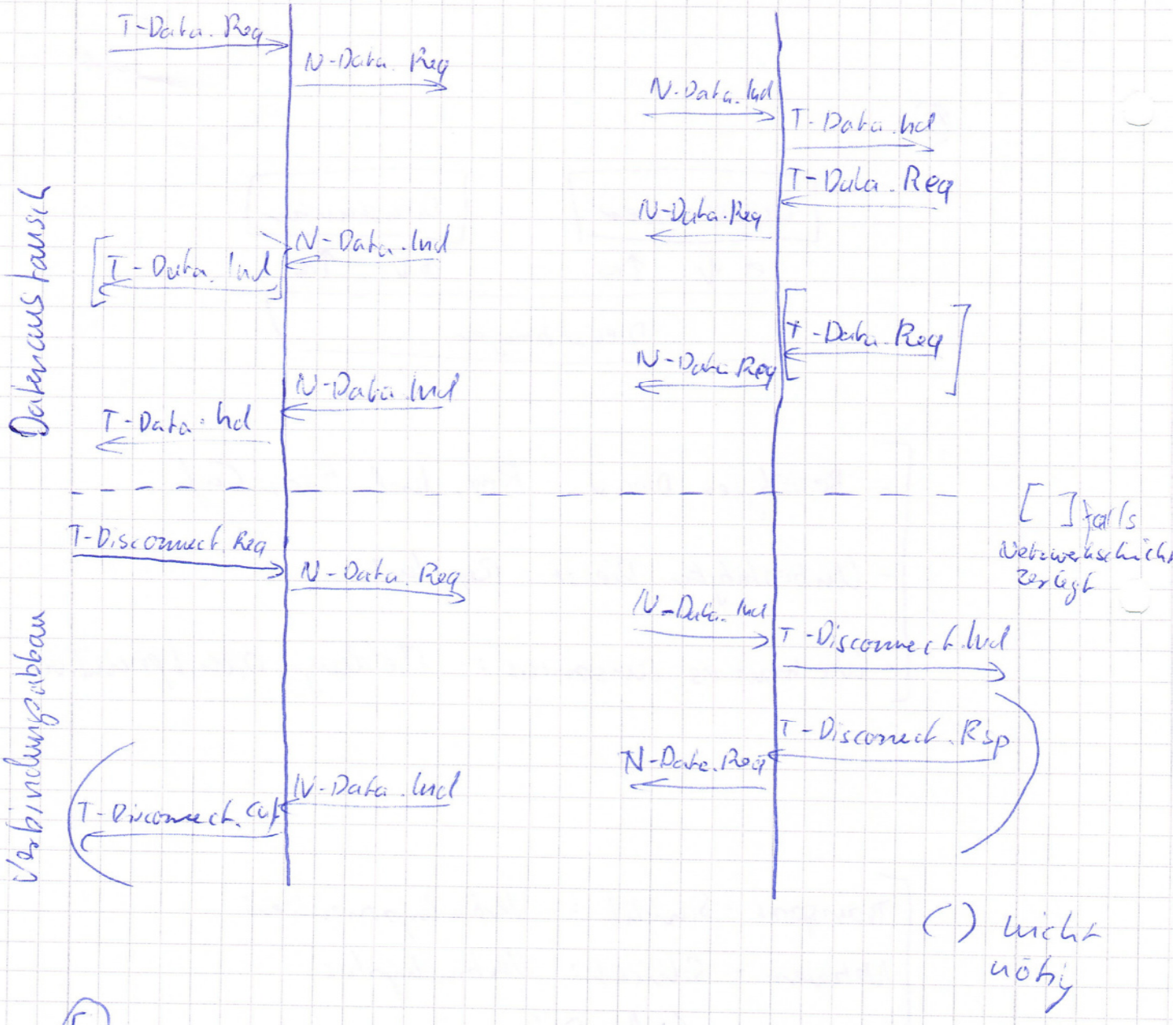


- Bestätigter Dienst: Req, Ind, Rsp, Cnf
- Unbestätigter Dienst: Req, Ind
- Wie man es anspricht: Meldung - Dienstprimitive

- Transport-Schicht: Verbindungsorientiert
- Netzwerk-Schicht: Verbindungslos
  ↳ N-Data



↳ □



(5)

Easy - Credit!

[redacted]

7.

a)

2/2

Codetransparenz kann definiert werden als eine „Vereinbarung von Regeln zur Übertragung von Nutzdaten (d.h. Übertragung beliebiger Bit- bzw. Zeichenkombinationen im Nutzdatenfeld). Somit wird die Übertragung von Nutzdaten ermöglicht, die beliebige Bit- und Zeichenkombinationen enthalten. Möglichkeiten hierfür sind beispielsweise Bitstuffing oder Character Stuffing. Der Nutzdatenblock in dieser Aufgabe ist deshalb für Codestransparenz interessant, weil er Teile enthält, die als „Begrenzungsfeld" bzw. im späteren Teil als „Flag" missinterpretiert werden könnten und somit zur Übertragung und Einhaltung der Codetransparenz gesondert behandelt werden müssen. ✓

0,5/2 *b)*

Der Sender fügt den Nutzdaten

01111101 11111101 01111110 10111110 00010000 11110101

das Begrenzungsfeld = STX = ETX

01111110

voran und nachgestellt hinzu, sowie das Steuerzeichen DLE

00010000

ganz am Anfang und am Ende der Bitfolge. Der resultierende Bitstrom lautet:

DLE STX DLE noch ein Mal!

00010000 01111110 01111101 11111101 01111110 10111110 00010000 11110101 01111110 00010000

* Ende = DLE ETX

Der Empfänger muss die hinzugefügten Bits nun wieder entfernen um den korrekten Nutzdatenblock zu erhalten.

Länge?

0,5/2 *c)*

Es muss nun vom Sender ein Escape Zeichen 00010000 eingefügt werden sobald eine Folge in den Nutzdaten auftritt, die mit einem Steuerzeichen verwechselt werden könnten (vor und hinter der Folge). Der Empfänger muss diese Escape Zeichen dann ebenso wieder entfernen. Die neue Bitfolge lautet:

DLE STX (1) (2) DLE (3) DLE (4)

~~00010000~~ 01111110 01111101 11111101 00010000 01111110 00010000 10111110

~~00010000~~ 00010000 00010000 11110101 01111110 ~~00010000~~

DLE DLE (5) (6) ETX DLE

Länge?

[redacted]

d)

1/2 Beim Bitstuffing wird die Bitfolge

01111110

als „Flag“ verwendet um Anfang und Ende der Nutzdaten zu markieren. Falls diese Folge exakt so in den Nutzdaten vorkommt, muss sie durch Einfügen von Bits verändert werden um nicht für eine „Flag“ gehalten zu werden. In diesem Fall wären genau 6 aufeinanderfolgende „1“er ein Problem. Dies passiert einmal im Nutzdatenblock. Allerdings gibt es auch einmal 7 folgende „1“er und einmal 5, deshalb fügen wir zwei „1“er hinzu. Der Empfänger kann so exakt die Stelle mit den hinzugefügten Bits bestimmten (8 folgende „1“er) und so wieder zwei entfernen. Der resultierende Bitstrom mit voran und nachgestellter Flag sowie hinzugefügten Bits lautet also:

nicht Stuffing!

01111110 01111101 11111101 0111111110 10111110 00010000 11110101 01111110 Länge?

e)

1,5/2 Die Idee ist, dass vor den Nutzdaten die Länge dieses Blocks in Bitdarstellung gesendet wird, so dass der Empfänger prüfen kann ob der empfangene Nutzdatenblock auch die richtige Länge besitzt. Wir legen fest, dass die Länge in einem 8-Bit Block direkt nach dem Begrenzungsfeld übermittelt wird, 8-Bit können eine maximale Länge von 255 Zeichen beschreiben, längere Nutzdatenblöcke würden dann aufgeteilt werden. Unser Nutzdatenblock besteht aus 48 Zeichen, d.h. in Bitdarstellung in einem 8-Bit Block:

00110000 V

Mit dem voran und nachgestellten Begrenzungsfeld lautet die resultierende Bitfolge:

01111110 00110000 01111101 11111101 01111110 10111110 00010000 11110101 01111110 Länge?

f) Die Nutzdaten werden zunächst als 8-Bit Blöcke unterteilt. Jedem

1,5/2 Nutzdatenblock wird nun ein führende „0“ vorangestellt, um 9-Bit Blöcke zu erhalten. Gültige Codewörter würden ebenfalls eine „0“ erhalten. Die „Begrenzer“ werden durch eine führende „1“ gekennzeichnet, und würden damit ungültige Codewörter.

ok. aber Ergebnis?

Länge?

[Lsg: STX

01111101 01111101 ... 1 bei Begrenz

0 bei Daten

8.

a)

Bitstrom: 10101100 → x^7 + x^5 + x^3 +x^2
Generator: x^3 + 1 → 1001 (CRC mit Grad 3)

Es wird an den Rahmen 3x 0 angehängt, weil der Generator diesen Grad besitzt.

→ 10101100000

Es folgt die Division durch den Generator:

```
10101100000 / 1001 = 10011000 R 11
1001
  1111
  1001
   1100
   1001
    1010
    1001
      1100
      1001
       1010
       1001
         11
```

Es bleibt der Rest 11, der durch den Generator-Grad von 3 zu 011 wird.
Der Rest wird nun an den Rahmen angehängt und versendet: 10101100011

Der Empfänger sollte nun bei einer Prüfung der Übertragung mit dem Generator auf einen Rest von 0 kommen.

```
10101100011 / 1001 = 10011001
1001
  1111
  1001
   1100
   1001
    1010
    1001
      1101
      1001
       1001
       1001
          0
```

Der Rest ist 0, die Übertragung sollte korrekt gewesen sein.

[redacted]

2/2 b)

Addition des Fehlermusters:

```
  001 0000 0000
+101 0110 0011
  110 0110 0011
```

Prüfung der neuen Übertragung mit dem Generator:

```
11001100011 / 1001 = 10110101 R 101
1001
 1011
 1001
   1010
   1001
     1100
     1001
      1011
      1001
        101
```

Es bleibt ein Rest 101, somit müsste die Übertragung fehlerhaft sein. V

2/2 c)

Wenn das Fehlermuster ein Vielfaches des Prüfmusters beträgt, so kann die Übertragung fehlerhaft sein, obwohl der Restwert 0 ergibt und die Übertragung damit als korrekt eingestuft wird.

Beispiel:

```
  000 1001 0000
+101 0110 0011
  101 1111  0011
```

Die Prüfung mit dem Generator führt zu:

```
10111110011 / 1001 = 10101001
1001
   1111
   1001
    1100
    1001
     1010
     1001
       1101
       1001
        1001
        1001
           0
```

Die Übertragung mit Fehlermuster 00010010000 wurde als korrekt eingestuft. V

d)

1/1

Die Errechnung der korrekten Daten ist nicht möglich, da (wie in Teilaufgabe *c)* gezeigt) alle Fehlermuster, die Vielfache des Generators darstellen, als korrekt eingestuft werden.

(Damit nur ein möglicher korrekter Wert errechnet werden könnte, müsste der Generator mit hinreichend großem Grad gewählt werden, sodass die Intention einer „schlanken" Fehlererkennung verworfen werden müsste.) ✓

e)

1/2

i]

Polynom x^15 + x^14 + x^10 + x^8 + x^7 + x^4 + x^3 + 1 besitzt Grad 15, es werden also 15 Bit an die Nutzdaten angehängt. ✓

ii]

Unser CRC-Anhang hat die Länge r = 15, der Burst-Fehler die Länge x = 16.

→ für x > r gilt nach Tanenbaum:

Fehlerrate F = (0,5)^(r+1) = (0,5)^14 = 0,00006103515625

Der Fehler wird also mit einer Wahrscheinlichkeit von etwa 0,006% nicht erkannt.

0,003, speich $\frac{1}{2^{15}}$

f)

für jedes Bit wahrscheinlich ⇒ 15 ✓

2/2

i]

Die Nutzdatengröße beträgt 1kByte = 1024Byte → Es werden 1024 Paritätsbits = 128Byte hinzugefügt. ✓

ii]

Die Wahrscheinlichkeit dafür, dass ein Fehler im 8Bit-Block nicht gefunden wird, teilt sich auf in die Wahrscheinlichkeit, dass 2 von 8 Bits nicht korrekt sind, die Wahrscheinlichkeit, dass 4 Bits nicht korrekt sind, analog für 6 und 8 Bits. In diesen vier Fällen ist die Parität für einen Block gleich, obwohl 2, 4, 6 oder 8 falsche Bits im Block liegen, d.h. ein Fehlermuster mit genau 2, 4, 6 oder 8 „1ern" auf einen Block addiert wurde.
Die Fehlerbits werden nun auf die 8 möglichen Stellen im Block aufgeteilt, das entspricht dem Kombinatorik-Modell der ununterscheidbaren Bälle und unterscheidbaren Urnen.
Der 8Bit-Block kann 256 verschiedene Werte annehmen, weswegen nun geprüft wird, wie hoch der Anteil an Fehlermöglichkeiten ist:

(8 über 2) / 256 + (8 über 4) / 256 + (8 über 6) / 256 + (8 über 8) / 256 = 0,496 ✓

[redacted]

Da wir zwei 8Bit-Blöcke betrachten, wird diese Wahrscheinlichkeit quadriert:

F = 0,496^2 = 0,246 ok.

Die Tatsache, dass die Fehlerblöcke direkt hintereinander liegen, sollte wegen der Unabhängigkeit der Blöcke von einander vernachlässigbar sein. Anderenfalls müsste hier nochmals das Kombinatorik-Modell der ununterscheidbaren Bälle und der unterscheidbaren Urnen angewendet werden.

9. 4/4

Bis der Sender vom Empfänger eine Bestätigung über erfolgreichen Datenversand erhält, vergeht für Hin- und Rückrichtung die Zeit 2 x tD + s [sek]. Die Variable s steht für die Zeit, die für die Serialisierung auf Sender-/Empfängerseite benötigt wird.

Damit der Kanal ausgelastet ist, werden kontinuierlich Daten gesendet mit einem Sendefenster der Größe (2 x tD + s) x r [bit]. ok.

Für Paket-basierte Übertragung könnte s jedoch einen zu vernachlässigenden Wert annehmen.

[redacted]

10.

a)

1/1

Die durch den Datenverlust ausbleibende Empfangsbestätigung interpretiert der Sender als implizite Wiederholungsanforderung. Die nochmals gesendeten Daten haben aber wiederum die Sequenznummern 0 und 1, wodurch der Empfänger nicht wissen kann, ob es sich beim empfangenen Paket um eine Wiederholung wegen fehlender ACKs oder um das reguläre Folgepaket handelt. Daher erhält die dienstnehmende Schicht das Paket doppelt. Dieses Problem tritt auf, wenn der Sequenznummernraum zu klein für das gewählte Sendefenster ist. Für den umgekehrten Fall gilt selbiges.

b)

0,5/1

Um beim dargestellten Szenario das Missverständnis zu vermeiden, braucht man mindestens drei Sequenznummern:

Im Falle einer Nicht-Bestätigung von [0,1] würde erneut das Paket mit der Sequenznummer [0,1] gesendet werden. Das nachfolgende Paket würde mit [1,2] gekennzeichnet und wäre somit vom "Wiederholer" unterscheidbar.

2,0

c)

i]

0/1,5

Go-Back-N kann in diesem Fall verwendet werden, da im Falle von verlorengegangenen Paketen die folgenden Pakete vom Empfänger ignoriert werden. (Die Sequenznummern der Folgepakete liegen nicht im Empfangsfenster!) Das verlorengegangene Paket und dessen Folgepaket müssen dann erneut gesendet werden.

ii]

0/2,5

Selective Repeat ist hier nicht anwendbar, da es keinen ausreichend großen Puffer (Annahme Puffergröße = 1) für Pakete gibt, die während der Zeit, bis ein verlorenes Paket erneut gesendet wurde, zwischengespeichert werden können.

d)

Für Sequenznummerngröße = n Bit gilt:

2/2

Go-Back-N: Es dürfen maximal (2^n) – 1 Pakete gleichzeitig übertragen werden.
Beispiel: n = 4 Bit → Maximal (2^4) – 1 = 15 Pakete gleichzeitig. Ginge beispielsweise das ACK von Paket [0] verloren und es wären 16 Pakete erlaubt und auch bereits gesendet, so würde der Empfänger die Wiederholung vom ersten Paket als das Erste der nächsten Sendung identifizieren. Bei maximal (2^n) – 1 = 15 Paketen erwartet der Empfänger als nächstes das sechzehnte und letzte Paket und kann daher das Paket [0] als Wiederholung identifizieren.

Selective Repeat: Es dürfen maximal (2^n) / 2 Pakete gleichzeitig übertragen werden.
Beispiel: n = 4 Bit → Maximal (2^4) / 2 = 8 Pakete gleichzeitig. Wäre das Fenster größer (z.B. 9), so bestünde die Möglichkeit, dass der Empfänger bereits alle sechzehn Pakete erhalten und bestätigt hat und schon auf die Pakete 1 bis 9 der nächsten Sendung wartet. Wenn jedoch das ACK für Paket Nummer 9 der vorherigen Sendung nicht beim Sender angekommen ist, wird er nach einiger Zeit einfach nochmal Paket 9 senden, wobei der Empfänger schon auf Paket 9 der nächsten Sendung wartet.

11.

a) Ein beliebiger Nutzer sendet genau dann kollisionsfrei, wenn (n-1) Nutzer nicht senden und 1 Nutzer sendet:

2/2

$$P = n \cdot (1-p)^{n-1} \cdot p$$ ✓

b) Sei

2/2

$$\frac{d}{dp}(P) = 0$$

Dann folgt:

$$n \cdot \frac{d}{dp}((1-p)^{n-1} \cdot p) = -(n-1) \cdot (1-p)^{n-2} \cdot p \cdot n + n \cdot (1-p)^{n-1} = 0$$ ✓

$$p \cdot n \cdot (n-1) \cdot (1-p)^{n-2} = n \cdot (1-p)^{n-1}$$ ✓

$$p \cdot (n-1) = \frac{(1-p)^{n-1}}{(1-p)^{n-2}} = 1-p$$ ✓

$$p \cdot n - p = 1 - p$$ ✓

$$p \cdot n = 1$$ ✓

$$p = \frac{1}{n}$$ ✓

Die Wahrscheinlichkeit einer kollisionsfreien Übertragung wird bei $p = \frac{1}{n}$ maximiert. ✓

c) $P\left(\frac{1}{n}\right) = n \cdot (1-\frac{1}{n})^{n-1} \cdot \frac{1}{n}$ ✓

1,5/2

$$\lim_{n\to\infty}\left(P\left(\frac{1}{n}\right)\right) = \lim_{n\to\infty}\left((1-\frac{1}{n})^{n-1}\right) = \frac{\lim_{n\to\infty}\left((1-\frac{1}{n})^{n-1}\right)}{\lim_{n\to\infty}\left(1-\frac{1}{n}\right)} = \frac{\frac{1}{e}}{1-0} = \frac{1}{e} \approx 0{,}3679$$ ✓ ✓

was ist das? Antwort? Ja!

d) Alle Sender arbeiten mit p-persistent CSMA, was bedeutet, dass sie bei belegtem Medium warten bis es frei wird um dann mit der Wahrscheinlichkeit p zu senden. Wird ein Slot frei und versucht nur genau ein Sender den Sendevorgang zu starten, gibt es keine Probleme, versuchen es mehrere Sender, kommt es zu einer Kollision und eine Übertragung kommt nicht zu Stande. Eine Übertragung kann also dann nicht mehr von anderen Sendern gestört werden, sobald sie kollisionsfrei gestartet wurde (da alle übrigen Sender aufgrund des belegten Mediums mit ihrem Sendevorgang warten).

1/2

Wann wird diese Kollision gestört?

Besser: Übertragung kann nicht mehr gestört werden, andere Sender das Signal der Sendestation wahrnehmen!
⟹ Signallaufzeitunterschiedlichkeit t_{prop}

[redacted]

1/1

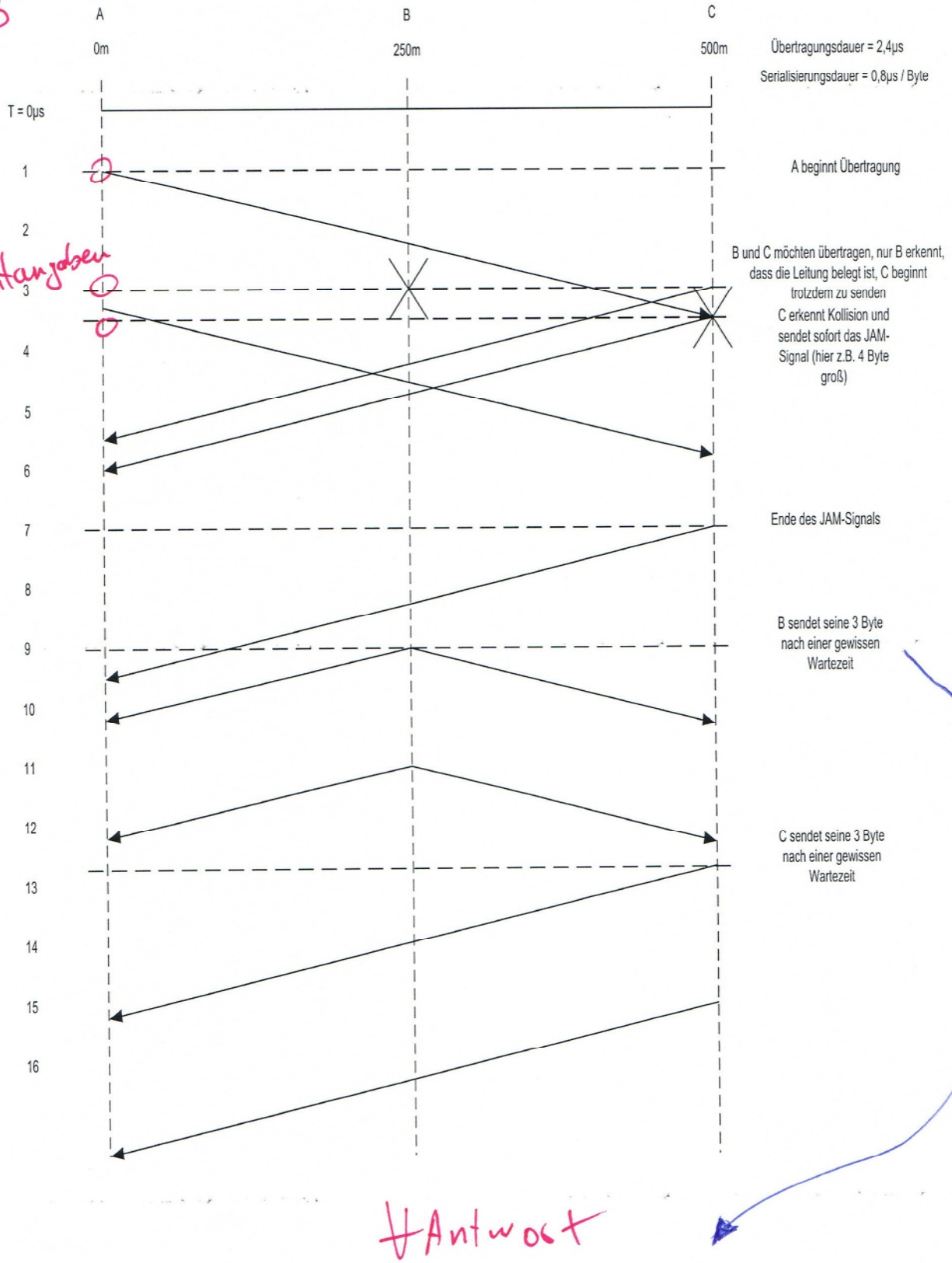
e) Für eine optimale Auslastung des Kanals müsste nach jedem Sendevorgang sofort der nächste Sendevorgang kollisionsfrei (d.h. genau ein Sender startet eine Übertragung) beginnen. Die Wahrscheinlichkeit p ist also in Abhängigkeit von der Anzahl n an Sendern so zu wählen, dass nach jedem Sendevorgang genau ein Sender versuchen wird, eine Übertragung zu starten.

[redacted]

12.

a)

2/3



#Zeitangaben

#Antwort

Zeitslots: $[0, 2^i - 1]$ i = Anzahl Kollisionen

[redacted]

2/2

b)
Fast Ethernet 100MBit, Rahmenlänge 64Byte

$l = (c \times Tr) / 2 = (c \times n) / (2 \times r)$

n = 64Byte x 8Bit/Byte = 512Bit
r = 100.000.000Bit/s

$l = (200.000.000m/s \times 512Bit) / (2 \times 100.000.000Bit/s) = \underline{512m}$ ✓

2/2

c)
Der Hub vermittelt nur auf Ebene 1. Daher kann er nicht als Trenner mehrerer Kollisionen dienen. Alle am Hub angeschlossenen Teilnehmer liegen also in einer Kollisionsdomäne. Die Bridge vermittelt auf Ebene 2, der Switch auf Ebene 2 oder 3, wodurch es möglich ist, das Netz in verschiedene Kollisionsdomänen aufzuteilen. ✓

13.

a)

2/2 Wahrscheinlichkeit dafür, dass einer oder mehr Bits kaputt sind, ist 1 – Wahrscheinlichkeit, dass kein Bit kaputt ist.

→ pR = 1 – (1-pB)^pSize ✓

Funk:

Für pSize = 40Byte = 320Bit → pR = 1 – (1 – 0,001)^320 = 0,274.967

Für pSize = 1500Byte = 12000Bit → pR = 1 – (1 – 0,001)^12000 = 0,999.993 ✓ ← das was gefragt

Luft:

Für pSize = 40Byte = 320Bit → pR = 1- (1 – 0,000000001)^320 = 0,000.000.3

Für pSize = 1500Byte = 12000Bit → pR = 1- (1 – 0,000000001)^12000 = 0,000.012

Für große Rahmen konvergiert die Rahmenfehlerrate gegen 1. OK

2/2 b)

Sei pR = 0,1 = 10%

→ In 9 von 10 Fällen tritt kein Fehler auf. Eine maximale Anzahl an Versuchen gibt es in der Regel nicht. Es könnte ~~e~~ bei einer ~~mit~~ Wahrscheinlichkeit nahe 0 trotzdem eine unendliche Reihe an Fehlversuchen auftreten. genau

Wenn das Protokoll nach 5 Versuchen aufgibt, so geschieht dies, wenn 5x in Folge der Fehler eingetreten ist.

→ pGes = pR x pR x pR x pR x pR = pR^5 = 0,1^5 = 0,00001 = 1/100.000 ✓

Statistisch gesehen tritt eine Meldung an die höhere Schicht in einem von 100.000 Fällen auf.

[redacted]

14.

2/2

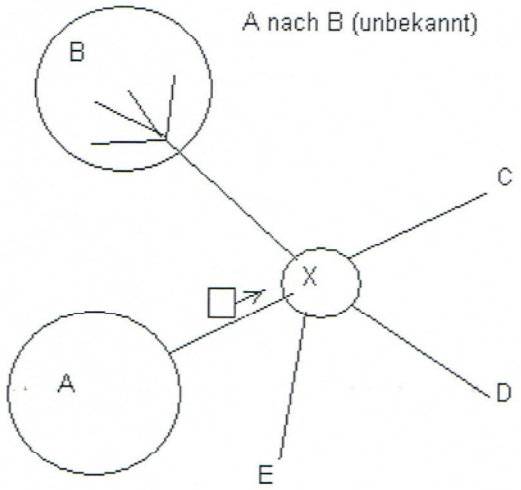
a)

Brücken und Repeater arbeiten in der Netzwerkkopplung. Dabei fungieren beide auf unterschiedlichen Schichten, der Repeater auf Schicht 1, die Brücke auf Schicht 2. ✓
Repeater sind für die Kopplung physikalischer Medien durch Signalregeneration und -verstärkung zuständig, wobei hier keine Zwischenspeicherung und keine Bearbeitung der Pakete erfolgt. ✓
Brücken koppeln Netzwerke gleichen (Non-Translating) und unterschiedlichen Typs (Translating), besitzen eine Filterfunktion, erhöhen die Netzkapazität durch Partitionierung und besitzen im Gegensatz zum Repeater eine gewisse "Intelligenz", da sie nicht nur weiterleiten. ✓

4/4

b)

Eine transparente Brücke "lernt", in welchem Teilnetz sich welche MAC-Adressen befinden, sowie mögliche Empfänger, indem die Absender von Paketen in eine interne Weiterleitungstabelle eingetraten werden. So kann der Weg zum Empfänger bestimmt werden. Anfangs ist die Tabelle leer, die Einträge werden geschrieben, wenn die Brücke Frames empfängt. Ist ein Zieleintrag nicht in der Tabelle enthalten, so wird der Frame über Broadcast an alle gesendet und nachdem das Ziel geantwortet hat, die Route hergestellt.



→ 5 x 1 Broadcast
4 x 2 ABCDE bekannt
3 x 5 innerhalb des Segments
= 28 (ca. 28/5=5,6 Rahmen pro Segment)

✓ nice!

[redacted]

c)

2/2

Puffer zur Zwischenspeicherung sind nötig, da laufend neue Adressen hinzu kommen und verarbeitet werden müssen. Ist eine Adresse bereits als Eintrag in der Tabelle vorhanden, wird "nur" weitergeleitet. Ist sie nicht vorhanden, kommt "flooding" zum Einsatz, die Information wird an alle gesendet, außer an das Interface, von dem es kam. Hierfür ist ein Zwischenspeicher von Nöten. ✓
Ebenfalls für die nächste Stufe, das "filtering", sowie für das "aging" und unterschiedliche Übertragungsgeschwindigkeiten ist ein Puffer brauchbar. ✓
Nach Peterson kann ein einzelnes Segment z.B. 10Mbps Traffic verarbeiten, eine Brücke mit n Ports 10n Mbps. Bei 1000Base-X wird der Datenstrom in 8-Bit-Rahmen zerlegt, wodurch sich die theoretische Durchsatzrate unter Voll-Duplex von n GBit ergibt. ✓

15.

1/1

a) Spanning-Tree Algorithmus ist eine Methode, um in einem mehrfach verschleiften, redundaten Netz eine (virtuelle) Baumstruktur abzubilden. Dazu kommunizieren die Bridges über Konfigurationsnachrichten (*Configuration Bridge Protocol Data Units*) miteinander. ✓

2/2

b) Es wird eine Bridge als *Root Bridge* ausgewählt, und alle anderen Bridges bestimmen den kürzesten Weg von sich selbst zu dieser Root Bridge. Die Bridge merkt sich außerdem den Port, über den sie die Root Bridge erreicht (*Root Port*).
Es werden vom Algorithmus nicht alle Ports berücksichtigt, sondern nur der Root Port sowie die Ports, die zu LANs führen, auf denen diese Bridge als *Designated Bridge* ausgewählt wurde.
Die *designated Bridge (bzw. designated Port)* eines LANs ist die Bridge, über welche der **kürzeste Weg** zur Root Bridge verläuft. Jeglicher Verkehr wird nur auf dem ausgewiesenen Ports angenommen und weitergeleitet. Pakete auf anderen Ports werden nicht beachtet. Eine Bridge hat eine eigene ID und auf jedem Port eine MAC Adresse. Die Konfigurations-Nachrichten werden an alle Bridges (Multicast) verschickt. Sie werden nicht in andere LANs weitergeleitet. ✓

1/1

c) Bridge 2 wird zur Root Bridge. ✓

d)

2/2

Brdige	Costs	Root Port	Designated Ports	Blocked Ports
2	0	-	1, 2	-
3	19	1	2	-
4	19	1	3	2
5	38	1	2	3

✓ ✓ ✓ ✓

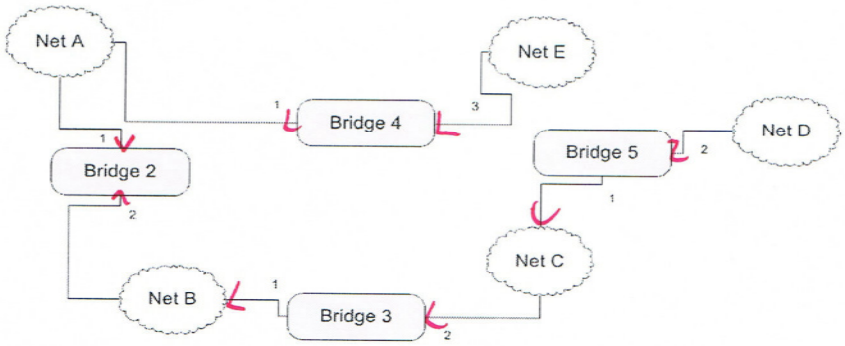
e) Siehe d). ✓

2/2

[redacted]

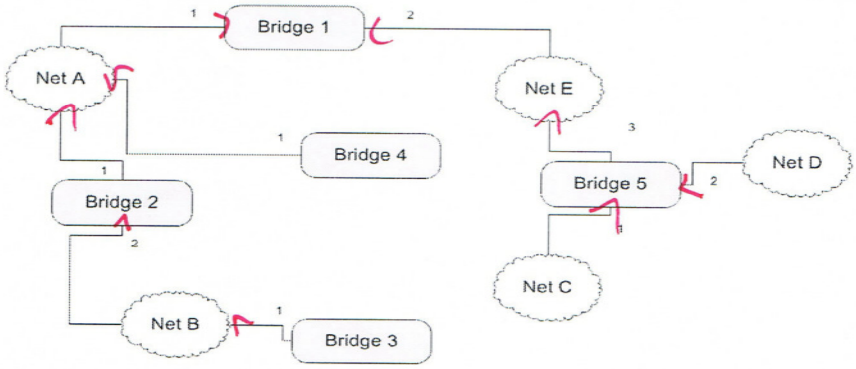
f) Topologie:

0,5/1



g) Bridge 1 wird zur Root Bridge:

2,5/3



Bridge	Costs	Root Port	Designated Ports	Blocked Ports
1	0	-	1,2	-
2	19	1	2	-
3	38	1	-	2
4	19	1	-	2,3
5	19	3	1,2	-

[redacted]

Aufgabe 16

14/5/2

a) Schritt 1 Anfang:

3/3

	A	B	C	D	E
A	0	∞	∞	∞	∞
B	∞	0	∞	∞	∞
C	∞	∞	0	∞	∞
D	∞	∞	∞	0	∞
E	∞	∞	∞	∞	0

✓

Schritt 2:

	A	B	C	D	E
A	0	1	4	∞	∞
B	1	0	1	1	∞
C	4	1	0	2	∞
D	∞	1	2	0	3
E	∞	∞	∞	3	0

✓

Schritt 3:

	A	B	C	D	E
A	0	1	2	2	∞
B	1	0	1	1	4
C	2	1	0	2	5
D	2	1	2	0	3
E	∞	4	5	3	0

✓

Schritt 4:

	A	B	C	D	E
A	0	1	2	2	5
B	1	0	1	1	4
C	2	1	0	2	5
D	2	1	2	0	3
E	5	4	5	3	0

✓

[redacted]

b) Routingtabelle:

0,5/1

Ziel	Nächster Hop	Kosten
~~A~~	~~-~~	~~-~~
B	- (B)	1
C	B	2
D	B	2
E	B,~~D~~	5

2/2

c)

$$T_{ABC} = T_V + 2 \cdot T_A + 2 \cdot \frac{8 \cdot L\,Bit}{\frac{10}{1} \cdot 10^6 \frac{Bit}{s}}$$ ✓

Anmerkung: Falls die Verarbeitungszeit T_V auch direkt beim Sender A und Empfänger C (und nicht nur bei den Zwischenstationen, in diesem Fall nur B) auftritt, so ist in obiger Gleichung ($3 \cdot T_V$) statt T_V anzunehmen!

würde hier 2 sagen

0,5/1

d) $T_{AC} = 2 \cdot T_V + T_A + \frac{8 \cdot L\,Bit}{\frac{10}{4} \cdot 10^6 \frac{Bit}{s}}$

2/2

e) Sei: $T_V = T_A = 1\,ms$

$$T_{ABC} = 5\,ms + 1{,}6 \cdot 10^{-6}\,s \cdot L$$

$$T_{AC} = 3\,ms + 3{,}2 \cdot 10^{-6}\,s \cdot L$$

Sei: $T_{ABC} = T_{AC}$

$\rightarrow 5\,ms + 1{,}6 \cdot 10^{-6}\,s \cdot L = 3\,ms + 3{,}2 \cdot 10^{-6}\,s \cdot L$

$\rightarrow L = 1250$

Für L < 1250 Byte ist das direkte Routing günstiger. ✓

2/2

f) Weitere Kriterien wären zum Beispiel die Verarbeitungszeit der Router, die Leitungsqualität und damit die Ausbreitungsgeschwindigkeit, die topologische Entfernung, das gewählte Übertragungsverfahren (Duplex/Simplex), Wartezeiten, ein Blockingstatus oder ein möglicher Datenverlust beim Senden/Empfangen. top.

[redacted]

g) Schritt 1 Anfang:

2/3

	A	B	C	D	E
A	0	1	2	2	5
B	1	0	1	1	4
C	2	1	0	2	5
D	2	1	2	0	∞
E	∞	∞	∞	∞	0

Schritt 2:

	A	B	C	D	E
A	0	1	2	2	**5**
B	1	0	1	1	**6**
C	2	1	0	2	**5**
D	2	1	2	0	**5**
E	∞	∞	∞	∞	0

Schritt 3 bis
....∞ .... mehr!

Es entsteht der „Count-to-infinity" Effekt, da alle Knoten bei jedem Update immer einen (eigentlich gar nicht mehr vorhanden) Weg zu E finden und dabei ihre Kosten immer erhöhen.

h) **Split Horizon:** Eine Pfadinformation darf nicht über das gleiche Interface veröffentlicht werden, über das sie empfangen wurde, verhindert somit Schleifen. ✓

2/2

Triggered Updates: Bei Änderung des Distanzvektors sofortige Weiterleitung, dadurch Zeitreduktion. ✓

Path Vector: Es wird immer die komplette Route mit allen Knoten versendet, so kann jeder Knoten für sich erkennen, ob er selbst Teil der Route ist. ✓

[redacted]

Aufgabe 17

2/4

a)

Step	N	A	B	C	D	E	F
0	{A}	d=0, p=null	d=inf, p=undef	d=inf, p=undef	d=inf, p=undef	d=inf, p=undef	d=inf, p=undef
1	{B,D,E}	d=0, p=null	d=1, p=A	d=inf, p=undef	d=3, p=A	d=6, p=A	d=inf, p=undef
2	{C,F}	d=0, p=null	d=1, p=A	d=2, p=B	d=3, p=A	d=4, p=B	d=6, p=E
3	{ }	d=0, p=null	d=1, p=A	d=2, p=B	d=3, p=A	d=3, p=C	d=5, p=E

: es geht weiter

Anmerkung: Pseudo-Code unklar, unsere Annahme ist, dass N.popMinimum() die Elemente aus N auch entfernt und nicht nur wiedergibt. Somit verschwinden die Elemente aus N und die WHILE schleife endet sobald keine mehr vorhanden sind (N = { }).

3/3

b)

Ziel	Nächster Hop
A	null
B	B
C	B
D	A
E	C
F	E

[redacted]

```
N = {A,B,C,D,E,F};          // all target nodes
L = { }                     // available hops
h[V]                        // next hop
c(U,V)                      // distance from U to V

BEGIN

V = N.popMinimum()          // get lowest target node
L.add(V);                   // add V to the available hops

h[A] = null;

WHILE N.notEmpty() DO {
        V = N.popMinimum();         // get lowest target node
        d = inf;                    // variable with highest possible value
        FOR all elements of L DO {
                U = L.next();
                IF c(U,V) < d THEN {
                        d = c(U,V);
                        h[V] = U;
                }
        }
}

END
```

2/4

Aufgabe 18

Distance Vector Routing verbreitet die Information über kürzeste Pfade, indem jeder Knoten seine ihm bekannten kürzesten Wege zu allen seinen Nachbarn weiterschickt. Dabei kann es zu Problemen, wie dem zuvor beschrieben „Count-to-infinity" Effekt, kommen.

Link State Routing geht hingegen intelligenter vorBlatt und erforscht zunächst über ein ECHO-Paket die Verzögerung zwischen verschiedenen Nachbarn, um die gesammelten (und z.B. nach Aktualität gefilterten) Ergebnisse dann allen Routern zu übermitteln. Diese können dann z.B. durch den Algorithmus von Dijkstra die kürzesten Pfade bestimmen, und diese dann verwenden.

Mehr!

[redacted]

Aufgabe 19

a)

Subnet jeweils: 255.255.255.0

Broadcast 1: 160.25.10.255
Broadcast 3: 160.25.52.255

Def. Route 1: 160.25.10.254
Def. Route 3: 160.25.52.254

Net-Adress 1: 160.25.10.0 / 24
Net-Adress 3: 160.25.52.0 / 24

b)

Ein Router im Internet hat i.d.R. nur den Eintrag zum Router 159.1.1.1 (bzw. zum Router von dessen Provider, sofern vorhanden), die Subnetze „verbergen" sich dahinter, weswegen sich auch keine Änderung ergibt, wenn weitere Subnetze hinzugefügt werden.

c)

Nach „Longest Match Rule" gilt: Route 78.7.7.0 / 24 wird bevorzugt.

d)

Schicht 2: Data Link, z.B. MAC-Adresse
Schicht 3: Networking, z.B. IP-Adresse
Schicht 4: Transport, z.B. TCP(-Port)
Schicht 7, Application, z.B. HTTP(-Protocol)

e)

16er-Netz → 65534 IP-Adressen
→ 65534 / 8000 ergibt maximal 8 Subnetze mit maximal 8190 Teilnehmern
(8192 abzüglich Netzadresse und Broadcast-Adresse)

→ Subnetzmaske jeweils 255.255.224.0

1. Subnetz 160.25.0.0 / 19
2. Subnetz 160.25.32.0 / 19
3. Subnetz 160.25.64.0 / 19
4. Subnetz 160.25.96.0 / 19
5. Subnetz 160.25.128.0 / 19
6. Subnetz 160.25.160.0 / 19
7. Subnetz 160.25.192.0 / 19
8. Subnetz 160.25.224.0 / 19

[redacted]

f)

2/2

1. Rechner im 1. Subnetz 160.25.0.1 ✓
(10100000.00011001.00000000.00000001)

4000. Rechner im 1. Subnetz 160.25.15.160 ✓
(10100000.00011001.00001111.10100000)

1. Rechner im 2. Subnetz 160.25.32.1 ✓
(10100000.00011001.00100000.00000001)

4000. Rechner im 2. Subnetz 160.25.47.160 ✓
(10100000.00011001.00101111.10100000)

Netzmaske jeweils 255.255.224.0 (11111111.11111111.11100000.00000000) ✓

top!

[redacted]

Aufgabe 20

a) Payload

1/2/3

~~IP from 192.168.1.1 to 192.168.1.254~~

~~ARP who-has 192.168.1.1~~
~~ARP 192.168.1.1 is af:fe:14:af:fe:20~~

1. ARP who-has 192.168.1.254 + Ethernet Header
2. ARP 192.168.1.254 is af:fe:14:af:fe:21 + Ethernet Header

3. zusammen:
- Ethernet from af:fe:14:af:fe20 to af:fe:14:af:fe:21
- IP from 192.168.1.254 (1) to 192.168.5.33 (217.205.134.65)

ARP who-has 192.168.5.33
ARP 192.168.5.33 is af:fe:14:af:fe:22

Ethernet from af:fe:14:af:fe21 to af:fe:14:af:fe:22

IP from 192.168.5.33 to 192.168.5.34

4. ARP who-has 192.168.5.34 + Ethernet Header
5. ARP 192.168.5.34 is af:fe:14:af:fe:23 + Ethernet Header

6.
- Ethernet from af:fe:14:af:fe22 to af:fe:14:af:fe:23
- IP from 192.168.5.34 (1.1) to 217.205.134.70 (65)

ARP who-has 217.205.134.70
ARP 217.205.134.70 is af:fe:14:af:fe:24

Ethernet from af:fe:14:af:fe23 to af:fe:14:af:fe:24

IP from 217.205.134.70 to 217.205.134.65

7. ARP who-has 217.205.134.65 + Ethernet
8. ARP 217.205.134.65 is af:fe:14:af:fe:25 + Ethernet Header

9.
- Ethernet from af:fe:14:af:fe24 to af:fe:14:af:fe:25
- IP .....

Das ARP besitzt eine Tabelle für die Informationen bzgl. IP Adresse und zugehörige MAC Adresse. In welchen Abständen die Informationen gelöscht und erneuert werden müssen ist implementierungsabhängig. Wir gehen davon aus, dass die Informationen den vorgegebenen Sendeprozess überdauern und somit nur einmalig abgefragt werden müssen.

[redacted]

1/1

b) Potenzielles Sicherheitsproblem: Mit ARP-Spoofing ist es möglich, absichtlich eine falsche Hardwareadresse in einem Netz zu verteilen, teilweise sogar von außen über einen Remote-Broadcast. Dadurch kann der Datenverkehr für einen Rechner auf einen anderen umgelenkt und eventuell von diesem sogar gefiltert werden (Man-In-The-Middle-Angriff). ✓

+20 IPv4 Header

c) Router 1 bekommt ein Package der Größe 1000 Byte. Da die MTU zwischen Router 1 und Router 2 nur 580 Byte beträgt, wird es in zwei Packages zerlegt mit der Gesamtgröße (Nutzdaten + Header) 580 Byte sowie ~~420~~ 460 Byte. Dies wird durch die Flags und die Fragment Offsets ermöglicht.

1/2 2/2

Router 1:

Bitte in Dec. angeben!

First Package:

Total Length:	~~00000011 11110100~~ 1020
Identifier:	10110011 01110100 ✓
Flags:	000 ✓
Fragment Offset:	00000000 00000 ✓
Source Address:	11000000 10101000 00000001 ~~11111110~~ 0...01
Destination Address:	~~11000000 10101000 00000101 00100001~~ 217.205.134.65

Router 2:

1020 = 580 + 440

560 + 20 = 580

First Package:

Total Length:	00000010 01000100 ✓
Identifier:	10110011 01110100 ✓
Flags:	001 ✓
Fragment Offset:	00000000 00000 ✓
Source Address:	11000000 10101000 00000101 ~~00100010~~ 0...01
Destination Address:	11011001 11001101 10000110 01000110 ✓

Second Package:

Total Length:	00000001 ~~10100100~~ 460 = 440 + 20
Identifier:	10110011 01110100 ✓
Flags:	000 ✓
Fragment Offset:	~~00010010 00100~~ 70 oder 0x46 70 · 8 = 560
Source Address:	11000000 10101000 00000101 ~~00100010~~ 0...01
Destination Address:	11011001 11001101 10000110 01000110 ✓

[redacted]

Aufgabe 21

a) Altes System: Firma Klein bekommt 254 verfügbare Hosts (Netzklasse C), damit Verschwendung von 249 Adressen, Firma Mittel bekommt 65.534 (Netzklasse B), damit eine Verschwendung von 64.843 Adressen. 0,5/1 ~~843~~ 334

Neues System (CIDR): Firma Klein bekommt 6 Hosts, somit wird nur 1 Adresse verschwendet, Firma Mittel bekommt 1022 Hosts, somit Verschwendung von nur 322 Adressen. ✓

Classless Inter Domain Routing

b) Mit CIDR entfällt die feste Zuordnung einer IPv4-Adresse zu einer Netzklasse, aus welcher die Präfixlänge hervor ging. Durch die zusätzliche Angabe einer Netzmaske wird jetzt die IP-Adresse in den Netzwerk- und Hostteil aufgeteilt. ✓ 2/2

Bei CIDR führte man als neue Notation so genannte Suffixe ein. Das Suffix gibt die Anzahl der 1-Bits in der Netzmaske an. Diese Schreibform, z. B. 172.17.0.0/17, ist viel kürzer und im Umgang einfacher als die „Dotted decimal notation" wie 172.17.0.0/255.255.128.0 und ebenfalls eindeutig. ✓ top.

Komplizierter ist das neue System deshalb, weil nun (vgl. Beispielaufgabe 21a) weniger Adressen pro Firma zugeteilt werden und sich somit ein Bereich von verschiedenen Firmen geteilt wird. Die Routingtabellen werden viel größer und müssen trotzdem genauso aktuell gehalten werden.
Vorher konnte somit einfach in einer Routingtabelle als Next Hop bzw. Gateway einfach der Bereich eines kompletten Unternehmens angegeben werden, durch CIDR muss hier nun eine viel genauere Aufschlüsselung erfolgen. ✓

c) Die /8 Blöcke entsprechen der Klasse A, die /16 Blöcke Klasse B und die /24 Blöcke der Klasse C. 0/2

d) Eine niedrige Auslastung ist ein deutliches Zeichen für eine verschwenderische Adressvergabe. Dunkelblaue bis hellgrüne Felder stehen für eine niedrige Auslastung, ein Negativbeispiel wäre der AT&T Quadrant. ✓ 1/1

e) RIPE Network Coordination Centre, eine für die Vergabe von IP-Adressbereichen zuständige Organisation ✓ 1/1

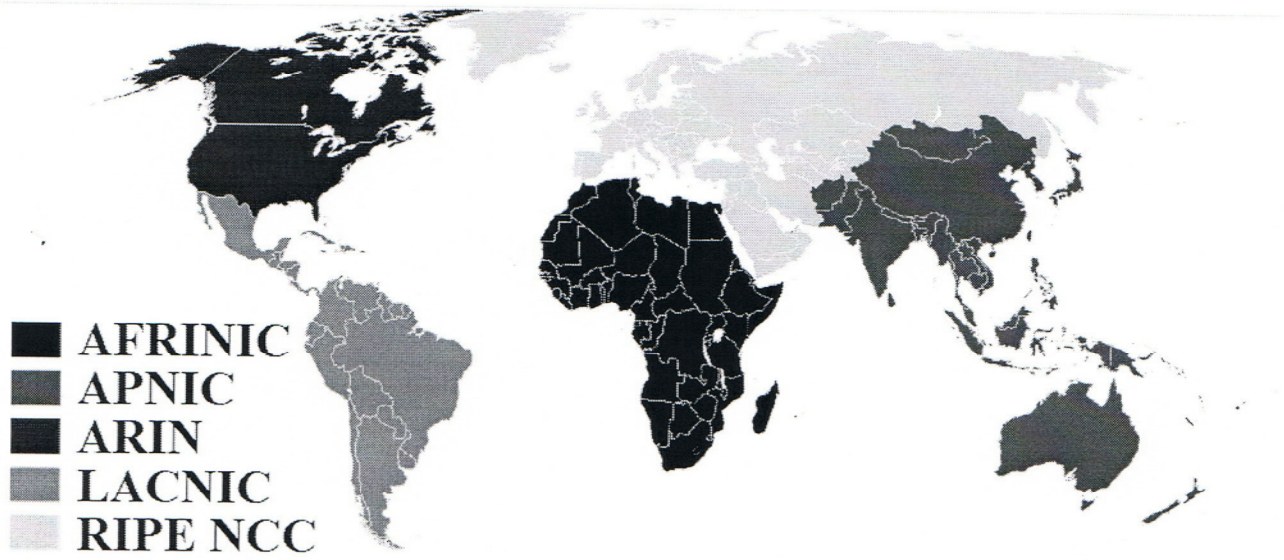
ARIN (American Registry for Internet Numbers) ist die Regional Internet Registry für die USA, Kanada, Bermuda, die Bahamas und Teile der Karibik. ✓

APNIC (Asia Pacific Network Information Centre) ist die Regional Internet Registry für die Region Asien und den Pazifik. ✓

[redacted]

LACNIC (Latin American and Caribbean Internet Addresses Registry) ist die Regional Internet Registry für Lateinamerika und die Karibik. V

AfriNIC ist die zuständige Regional Internet Registry für Afrika. V



1/1 f) Die Bereiche mit einer hohen Dichten wurden wahrscheinlich per CIDR vergeben, ein Beispiel wäre der obere rechte Quadrant aus dem Sektor APNIC 124.0.0.0 /7 V

0,5/1 g) 2006 gab es ~~16~~ 56 UNALLOCATED /8 Blöcke, aktuell sind es ebenfalls 16.

Aufgabe 22

a) Done
b) Done
c)

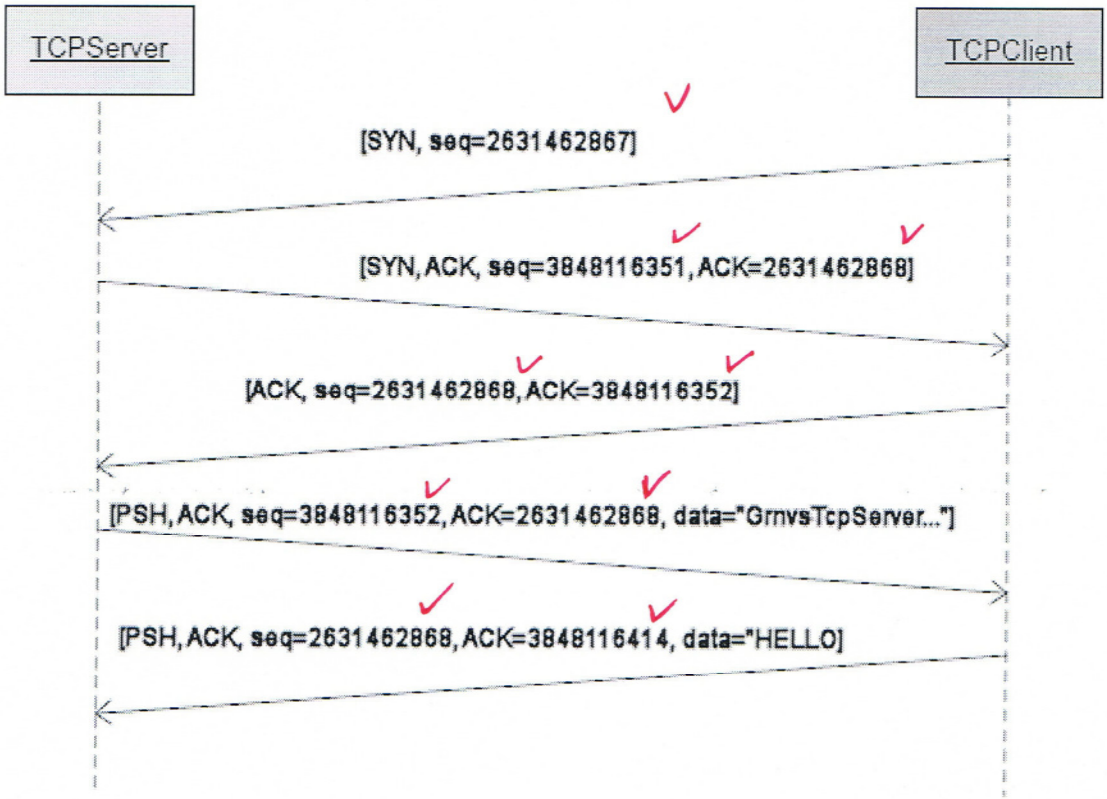
```
/* GrnvsTcpClient.java */

import java.net.*;
import java.io.*;

public class GrnvsTcpClient {
	private static BufferedReader in;
	private static BufferedWriter out;
	private static Socket socket;
	private static final String GROUP_LETTER = "C";
	private static final String TEAM_MEMBER_ONE = "Huber Stefan";
	private static final String TEAM_MEMBER_TWO = "Reimers Jonas";
	private static final String WELCOME_MSG = "Hello";
	private static boolean sendOne = true;	// set true to send
team member 1, false for team member 2

	public static void main(String[] args) {
		if (args.length != 2) {
			System.err.println("Usage: java GrnvsTcpClient
<host> <port>");
			System.exit(1);
		}
		// The communication will get handled by the
communicate-method...
		communicate(args[0], Integer.parseInt(args[1]));
	}

	private static void communicate(String host, int port) {
		try {
			int response = 0;
			String instr = "";
			socket = new Socket(host,port);
			try {
				in = new BufferedReader(new
InputStreamReader(socket.getInputStream()));
				out = new BufferedWriter(new
OutputStreamWriter(socket.getOutputStream()));
			}
			catch (IOException e) {
				System.err.println(e.toString());
			}
			System.out.println(in.readLine()); // Greetings
			say(WELCOME_MSG); // say HELLO
			System.out.println(in.readLine()); // Welcome
			System.out.println(in.readLine()); // Challenge
			for(int i = 0; i < 3; i++)
			{
```

```
                    instr = in.readLine();
                    System.out.println(instr); // to sum
                    response = response + Integer.parseInt(instr);

               }
               say(Integer.toString(response)); // send answer
               System.out.println(in.readLine()); // tutorial
group question
               say(GROUP_LETTER); // send tutorial group
               System.out.println(in.readLine()); // name question
               if(sendOne)
                    say(TEAM_MEMBER_ONE); // send name1
               else
                    say(TEAM_MEMBER_TWO); // send name2

               while((instr = in.readLine()) != null) {
                    System.out.println(instr);
               }

               in.close();
               out.close();
               socket.close();

          } catch (IOException e) {
               System.err.println(e.toString());
               System.exit(1);
          }
     }

     private static void say(String msg) {
          /*
           * We have to do a lot of sending to the Server so we
summed this up
           * here...
           */
          try {
               System.out.println("--> " + msg); // Output locally
for debug...
               out.write(msg + "\r\n"); // Send to the receiving
server
                                        // socket...
               out.flush(); // Send data out now and do not wait
until the send
                            // buffer is full...
          } catch (IOException e) {
               System.err.println(e.toString());
          }
     }
}
```

[redacted]

d)

4/4



e) Source MAC Address: 00:1b:fc:d2:67:2f
Vendor: ASUSTek COMPUTER INC.

0,5/1

und?

Destination Address: 00:26:4d:25:fb:65
Vendor: Arcadyan Technology Corporation

f) Pakete vom Server besaßen eine TTL von 54 bzw. 55. Da laut RFC 791 die TTL auf dem Weg zum Ziel um mind. 1 verringert werden muss, war der TTL Wert zu Sendebegin also mind. bei 55 bzw. 56. ~> so etwa 64 ;)

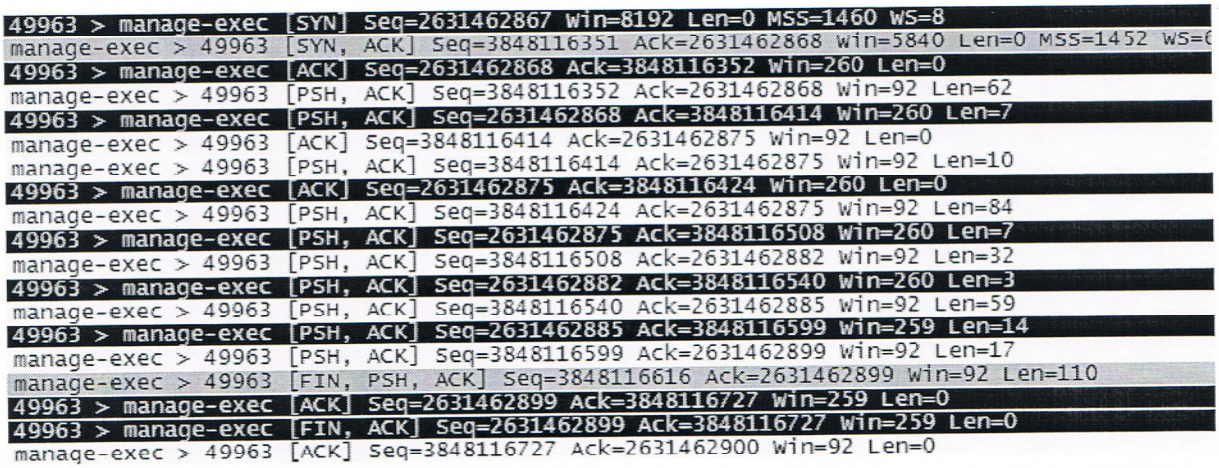
0,5/1

g) Die Anzahl der Sequenznummern stimmt mit der Anzahl der Pakete überein. Serverseitig kamen zunächst die Bestätigung der Verbindung, dann die Welcome Message inklusive Sicherheitsabfrage, die Frage nach dem Gruppenbuchstaben, die Frage nach dem Namen sowie eine Goodbye Message. Von unserem Client wurden insgesamt vier Pakete (HELLO, Additionslösung, Gruppenbuchstabe, Name) versendet.

1/2

Rechnung

erwartete Seq# alte Seq + Length = neue Seq



Wireshark Screenshot zeigt die verschiedenen Sequenznummern für die verschiedenen Pakete (schwarzer Hintergrund Client, weißer/grauer Hintergrund Server).

1,5/2

h) Gesetzt sind die Flags

ACK
PSH
~~SYN~~
~~FIN~~

PSH:
Beim Versenden von Daten über das TCP werden zwei Puffer verwendet. Senderseitig übermittelt die Applikation die zu sendenden Daten an das TCP und dieses puffert die Daten um mehrere kleine Übertragungen effizienter als eine große zu senden. Nachdem die Daten dann an den Empfänger übermittelt wurden, landen sie im empfängerseitigen Puffer. Dieser verfolgt ähnliche Ziele. Wenn vom TCP mehrere einzelne Pakete empfangen wurden, ist es besser diese zusammengefügt an die Applikation weiterzugeben.
RFC 1122 und RFC 793 spezifizieren das PSH-Flag so, dass bei gesetztem Flag sowohl der ausgehende, als auch der eingehende Puffer übergangen wird. Da man bei TCP keine Datagramme versendet, sondern einen Datenstrom hat, hilft das PSH-Flag den Strom effizienter zu verarbeiten, da die empfangende Applikation so gezielter aufgeweckt werden kann und nicht bei jedem eintreffenden Datenfragment feststellen muss, dass Teile der Daten noch nicht empfangen wurden, die aber nötig wären um überhaupt weitermachen zu können.
Hilfreich ist dies, wenn man zum Beispiel bei einer Telnet-Sitzung einen Befehl an den Empfänger senden will. Würde dieser Befehl erst im Puffer zwischengespeichert werden, so würde dieser (stark) verzögert abgearbeitet werden. V

Ohne das PSH-Flag würde die richtige übermittelte Antwort auf die ✓

[redacted]

„Validierungsrechnung“ des Servers sicherlich mehr Zeit in Anspruch nehmen. Allerdings ist es bei einem so großen Zeitfenster (5 Sekunden) unwahrscheinlich, dass das Zeitlimit überschritten wird.

Aufgabe 23

a) Adressraum IPv6: $2^{128} = 3{,}4 \cdot 10^{38}$ (vgl. IPv4 nur 2^{32})

b) IPv6 Header

Version	Traffic Class	Flow Label
Payload Length	Next Header	Hop Limit
Source Address		
Destination Address		

0110		
0000001111101000	00000110	01000000
0000000000000000000000000000000000000000000000000000000000000000 0000000000000010000001110000000000001100011000101011010		
0000000000000000000000000000000000000000000000000000000000000000 0000000000000010000101110001000110100110000000001011011		

Bedeutung Hop Limit:

8 Bit Maximale Anzahl an Zwischenschritten über Router, die ein Paket zurücklegen darf; wird beim Durchlaufen eines Routers („Hops“) um eins verringert. Pakete mit null als Hop Limit werden verworfen. Es entspricht dem Feld Time to Live (TTL) bei IPv4.

c) IPv6 Header

0110		
0000000111110100	00101100	01000000
0000000000000000000000000000000000000000000000000000000000000000 0000000000000010000001110000000000001100011000101011010		
0000000000000000000000000000000000000000000000000000000000000000 0000000000000010000101110001000110100110000000001011011		

[redacted]

Erweiterungsheader

BINARY	CONTENT	TYPE
00000110	TCP	NextHeader
00000000	X	Reserved
0000000000000	NULL	FragmentOffset
00	X	Reserved
1	TRUE	MoreFragmentsFlag
...	RANDOM	Identification

d) Die ersten Bits einer Adressen können Sonderfälle signalisieren:

Adressen mit `ff00::/8` (`ff...`) stehen für Multicast-Adressen.

Nach dem Multicast-Präfix folgen 4 Bits für Flags und 4 Bits für den Gültigkeitsbereich (Scope). Für die Flags sind zurzeit folgende Kombinationen gültig:

- 0: Permanent definierte *wohlbekannte* Multicast-Adressen (von der IANA zugewiesen)
- 1: (T-Bit gesetzt) Transient (vorübergehend) oder dynamisch zugewiesene Multicast-Adressen
- 3: (P-Bit gesetzt, erzwingt das T-Bit) *Unicast-Prefix-based* Multicast-Adressen (RFC 3306)
- 7: (R-Bit gesetzt, erzwingt P- und T-Bit) Multicast-Adressen, welche die Adresse des *Rendezvous Point* enthalten (RFC 3956)

Die folgenden Gültigkeitsbereiche sind definiert:

- 1: interfacelokal, diese Pakete verlassen die Schnittstelle nie. (Loopback)
- 2: link-lokal, werden von Routern grundsätzlich nie weitergeleitet und können deshalb das Teilnetz nicht verlassen.
- 4: adminlokal, der kleinste Bereich, dessen Abgrenzung in den Routern speziell administriert werden muss
- 5: sitelokal, dürfen zwar geroutet werden, jedoch nicht von Border-Routern.
- 8: organisationslokal, die Pakete dürfen auch von Border-Routern weitergeleitet werden, bleiben jedoch „im Unternehmen“ (hierzu müssen seitens des Routing-Protokolls entsprechende Vorkehrungen getroffen werden).
- e: globaler Multicast, der überall hin geroutet werden darf.
- 0, 3, f: reservierte Bereiche
- die restlichen Bereiche sind nicht zugewiesen und dürfen von Administratoren benutzt werden, um weitere Multicast-Regionen zu definieren.

24.

1/2

a)

Flusssteuerung: schützt den Empfänger vor zu großem Zufluss an Paketen vom Sender.

Staukontrolle: passt die Senderate an die Empfangskapazitäten an, um „Datenstau“ zu vermeiden.

→ genau das passiert bei Flusssteuerung...

1,5/2

b)

Slow Start: schützt vor zu hohem Datenverkehr nach einer Stausituation und erhöht die Fenstergröße schrittweise nach jeder empfangenen Quittung. → Staukontrolle → wie verhält sich die Fenstergröße

Receiver Window: Anzahl Bytes (Fenstergröße), die unbestätigt gesendet werden darf und vom Empfänger noch akzeptiert wird. → Flusskontrolle ✓

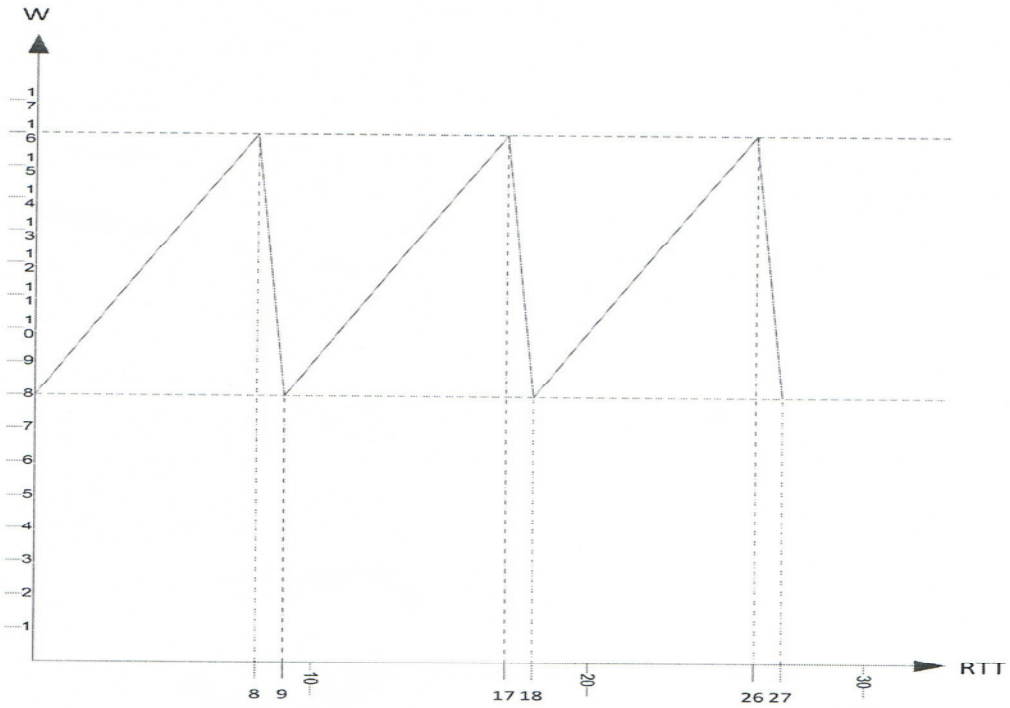
→ jeder Phase!

Congestion-Avoidance: Algorithmus für die Staukontrolle im Internet (TCP), die dafür sorgt, dass die Fenstergröße um 1 Segment erhöht wird, wenn alle Pakete aus dem Fenster bestätigt wurden. → Staukontrolle ✓

Multiplikative Descrease: Halbiert bei Fehlern die Fenstergröße, worauf unter Umständen ein Slow Start folgt. → Staukontrolle ✓

c)

2/2



✓

[redacted]

d)

2/2

Verlustrate V = 1 / { [w * (w + 1) / 2] + [(w/2 - 1) * (w + 1) / 2] }
= 1 / { w^2 / 2 + w / 2 + w^2 / 8 – w / 4}
= 1 / { 3 / 4 * w + 3 / 8 * w^2 }

e)

3/3

i) Maximale Rate rMax bei w-1 → doppelte RTT
→ rMax = gesendete Bits / Zeit
= { 3 / 4 * w + 3 / 8 * w^2 } / (w / 2 + 1) * (MSS / RTT)
= { 3 / 4 * w} * (MSS / RTT)

ii) → rMax = { 3 / 4 * 16} * (1460Byte / 200ms)
= 700,8kBit/s

top.

2/2 *f)*

Bis zu 15 Segmente verlustfrei → 15 * MSS Nutzdaten bzw. 15 * (MSS + Header) Gesamtdaten → UDP-Header ist 8Byte groß → IP-Header ist 20Byte groß → 1472Byte reine Nutzdaten übertragbar!
Bei UDP existieren keine ACKs → pro RTT können 15 Pakete nacheinander übertragen werden.

rMaxUDP = 15 * 1472Byte / 200ms = 883,2kBit/s
rMaxTCP = 700,8kBit/s
Differenz = 883,2kBit/s – 700,8kBit/s = 182,4kBit/s = 22,27kByte/s

top.

[redacted]

25.

1/1 a) Der Sender wählt als letztendliche Größe für das Senderfester das Minimum aus Flusskontroll- und Staukontrollfenster. ✓

2/2 b)

24.000.000Bit/s * 0,8s / 8Bit/Byte = 2400000Byte = 2343,75KByte

Das Sendefenster muss 2400000Byte groß sein. ✓

2/2 c) Das Sendefenster ist auf eine Größe von 2 bis max. 65.535 Byte limitiert. „Window scale option" wird benötigt um die Daten effizienter zu Senden (siehe d). ✓

1/2 d) „Window scale option" nach RFC1323 bietet die Möglichkeit für effizienteres Senden. Da das Sendefenster nicht tatsäclich vergrößert werden kann, wird ein Skalierungsfaktor (von 0 bis 14) verwendet um eine Größte von 65.535 Byte bis max. 1 GByte zu erreichen. → mehr!

1/2 e)

Nach einem Fehler beginnt die Übertragung mit einem Slow-Start, der die Fenstergröße exponentiell bis zur propagierten Staufenstergröße vergrößert.

--> 1.200.000Byte > 2^i * 1500Byte ✓

(i steht für die Anzahl der RTTs)

--> 800Byte > 2^i * 1Byte

Umformung ergibt (ohne Einheiten)

--> log2(2^i) = log2(800)

--> i = ln(800) / ln(2) = 9,...

--> i ist also maximal 9

--> 1500Byte * 2^9 = 768.000Byte

Die Slow-Start-Phase dauert 9 RTTs und geht bis zu einer Fenstergröße von 768.000Byte.
Es bleiben also noch 432.000Byte bis die Fenstergröße optimal ist.
Die Annäherung erfolgt mittels Congestion Avoidance, d.h. pro RTT wird

[redacted]

das Fenster um 1500Byte vergrößert.

432.000Byte / 1500Byte = 288

Es dauert also 288RTTs bis die Fenstergröße optimal ist.

Das ist in gewisser Weise deshalb problematisch, weil der Slow-Start nur maximal bis zum Schwellenwert exponentiell ist und daher sehr schnell funktioniert, die Congestion Avoidance jedoch nur lineares Wachstum erzeugt, und damit ein vielfaches länger zur Annäherung braucht.

Mit einem Beispielwert für die RTT aus Aufgabe b) mit 800ms dauert das in diesem Fall 288 * 800ms = 230,4s.
Das sind annähernd vier Minuten, bis die Geschwindigkeit von etwa 2/3 bis zur vollen Geschwindigkeit anwächst.

→ Idee gut. Man müsste allerdings das Verhältniss $\frac{1,2\ MBytes}{1500\ Bytes} = 800$ nehmen.

f) Standard-TCP-Implementierungen erledigen die Stau- und Flusskontrolle für jede TCP-Verbindung separat, obwohl verschiedene Verbindungen zum gleichen Host führen können. Da die Paketziele die gleichen sind, ist es hochwahrscheinlich, dass die Pakete auch die gleiche Route nehmen, sowie die gleichen Stausituationen erfahren werden. Es könnte daher vorteilhaft sein, Netzwerkinformationen unter den TCP-Verbindungen zu teilen (Network Information Sharing). Diese Idee lässt sich auf verschiedene Arten umsetzen, eine davon ist EFCM (Ensemble Flow Congestion Management), für das bereits ein Linux-Kernelmodul verfügbar ist. ✓

2/2

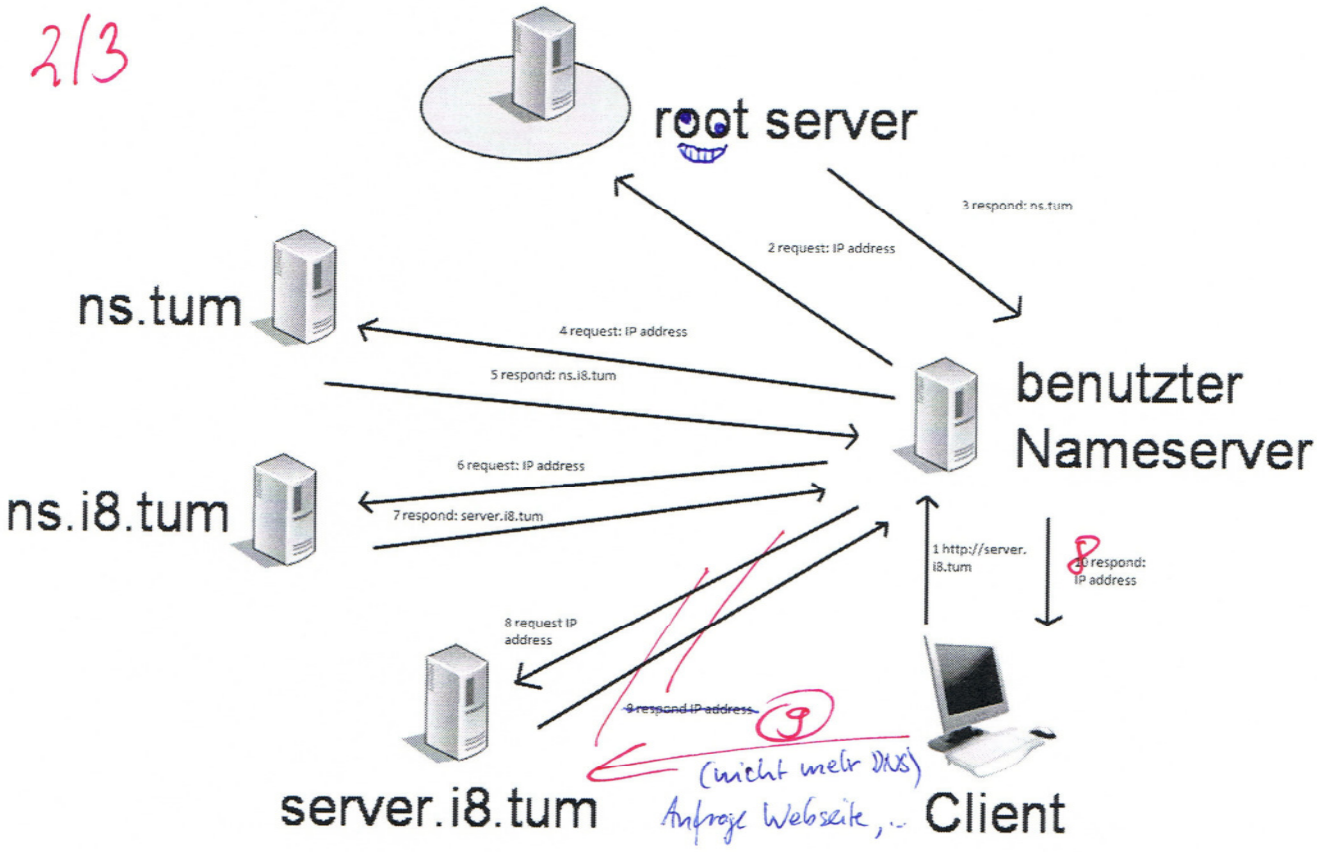
Windows: CTCP

[redacted]

26.

a) DNS

2/3



1/1 *b)* Das Verfahren nennt sich "reverse lookup". Damit wird versucht für eine IP-Adresse den zugehörigen Namen zu finden, in diesem Fall wird mit „20.159.131.in-addr.arpa" das Netzwerk „131.159.20.x" gesucht. ✓ Domain-Name vom Subnetz z.B. siehe oben i8.tum.de

0,5/1 *c)* Im Normalfall sollte nsX nie zu Domäne Y befragt werden. Dies kann nur passieren wenn bereits der „root server", der Anfragen nach nsX und nsY aufteilen sollte bösartig ist bzw. dafür manipuliert wurde. mehr!

[redacted]

27.

2/2 *a)*

Fett: Benutzereingaben
Kursiv: Serverantworten

```
linux:~ # telnet www.net.in.tum.de 80
Trying 131.159.15.231...
Connected to www.net.in.tum.de.
Escape character is '^]'.
GET / HTTP/1.0 → musste man hier machen
host:  www.net.in.tum.de

HTTP/1.1 200 OK
Date: Thu, 08 Jul 2010 15:38:18 GMT
Server: Apache/2.2.9 (Debian) mod_auth_kerb/5.3 DAV/2 PHP/5.2.6-1+lenny8 with Suhosin-Patch mod_ssl/2.2.9 OpenSSL/0.9.8g
X-Powered-By: PHP/5.2.6-1+lenny8
Set-Cookie: fe_typo_user=9caf68856c007c68a9998feb37793f15; path=/
Cache-Control: no-cache
Transfer-Encoding: chunked
Content-Type: text/html; charset=iso-8859-1

3c53
<!DOCTYPE html
      PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<html>
<head>
      <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />

<!--
      This website is powered by TYPO3 - inspiring people to share!
      TYPO3 is a free open source Content Management Framework initially created by Kasper Skaarhoj and licensed under GNU/GPL.
      TYPO3 is copyright 1998-2009 of Kasper Skaarhoj. Extensions are copyright of their respective owners.
      Information and contribution at http://typo3.com/ and http://typo3.org/
-->

      <base href="http://www.net.in.tum.de/" />
      <link rel="stylesheet" type="text/css" href="typo3temp/stylesheet_d3922e1b66.css" />
      <title>TUM Info VIII: Startseite</title>

...

</body>
</html>

Connection closed by foreign host.
```

V

0,5/1 *b)*

Jeder Vorlesungsteilnehmer hat eine eigene IP-Adresse, von der letztendlich die GET-Anfrage stammt. So kann der Webserver den Anfragenursprung unterscheiden. Sonderfall Proxy-Server: Hier kommen mehrere u.U. gleichartige Anfragen von einer IP-Adresse, nämlich der des Proxy-Servers. Bei transparenten Proxy-Servern kann der Webserver nicht unterscheiden, von wem die Anfrage gekommen ist. Manchmal ist jedoch das X-Forwarded-For Flag gesetzt, wodurch der Webserver dann die ursprüngliche IP-Adresse hinter dem Proxy erfahren kann.

5er Tupel

(Source IP, Source Port, Dest IP, Dest Port, Protocol)

[redacted]

c)

1/2

HTTP ↔ TCP ↔ IP ↔ Ethernet ↔ IP ↔ TCP ↔ HTTP
(nach ISO/OSI-Modell, zusätzlich DNS zur Namensauflösung)

mehr!

d)

Manchmal ist auch weniger mehr!

1/1

Früher bestanden Webseiten zumeist aus reinem Quelltext, es genügte eine einzelne Seite zu übertragen, dann wurde die Verbindung (vorerst) nicht mehr benötigt. Heute werden zusätzlich zahlreiche Daten angefordert: Bilder, Flash, Videos, Ton. Für die vielen einzelnen Dateien wäre es nach altem Standard nötig, jedesmal die Verbindung neu aufzubauen. Nach neuem Standard vermeidet man daher die zusätzliche Netzlast und überträgt alle benötigten Daten nach einem einzelnen Verbindungsaufbau. ✓

e)

1,5/1,5

Zuerst ruft man das Wurzelverzeichnis des Webservers auf.

```
linux:~ # telnet www.diadem-firewall.org 80
Trying 131.159.15.231...
Connected to www.diadem-firewall.org.
Escape character is '^]'.
GET / HTTP/1.1
host: www.diadem-firewall.org

HTTP/1.1 200 OK
Date: Thu, 08 Jul 2010 16:05:30 GMT
Server: Apache/2.2.9 (Debian) mod_auth_kerb/5.3 DAV/2 PHP/5.2.6-1+lenny8 with Suhosin-Patch mod_ssl/2.2.9
OpenSSL/0.9.8g
Last-Modified: Thu, 19 Feb 2004 15:04:20 GMT
ETag: "4222c4f-d0-3d3b5e1be0500"
Accept-Ranges: bytes
Content-Length: 208
Cache-Control: no-cache
Content-Type: text/html

<HTML>
<HEAD>
<meta HTTP-EQUIV="Refresh" CONTENT="0; URL=index.php">
</HEAD>
<BODY>
Automatic redirection, if not press
<h2 align="center">
   <a href="index.php">HERE!</a>
</h2>
</BODY>
</HTML>

Connection closed by foreign host.
```

✓

Der Quelltext der Webseite lässt den Browser eine automatische Weiterleitung auf die Datei index.php ausführen. Da die einfache TCP-Verbindung das nicht automatisch kann, muss man die Datei also manuell abrufen.

✓

```
linux:~ # telnet www.diadem-firewall.org 80
Trying 131.159.15.231...
```

[redacted]

```
Connected to www.diadem-firewall.org.
Escape character is '^]'.
GET /index.php HTTP/1.1
host: www.diadem-firewall.org

HTTP/1.1 200 OK
Date: Thu, 08 Jul 2010 16:06:21 GMT
Server: Apache/2.2.9 (Debian) mod_auth_kerb/5.3 DAV/2 PHP/5.2.6-1+lenny8 with Suhosin-Patch mod_ssl/2.2.9
OpenSSL/0.9.8g
X-Powered-By: PHP/5.2.6-1+lenny8
Cache-Control: no-cache
Content-Length: 5770
Content-Type: text/html


<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
   <head>
      <title>
         DIADEM FIREWALL
      </title>

...

   </body>
</html>

Connection closed by foreign host.
```

top!

1,5|1,5

f)

Bei ilab.net.in.tum.de wird die Weiterleitung durch den HTTP-Header (Location Flag) ausgelöst, der auf die Seite pages/view.php?address=p3&config=2010ss führt.

```
linux:~ # telnet ilab.net.in.tum.de 80
Trying 131.159.15.226...
Connected to ilab.net.in.tum.de.
Escape character is '^]'.
GET / HTTP/1.1
host: ilab.net.in.tum.de

HTTP/1.1 302 Found
Date: Thu, 08 Jul 2010 16:09:38 GMT
Server: Apache/2.2.8 (Linux/SUSE)
X-Powered-By: PHP/5.2.6
Location: pages/view.php?address=p3&config=2010ss
Content-Length: 0
Content-Type: text/html

Connection closed by foreign host.
```

Diese Seite muss man dann mit TELNET ebenfalls manuell abrufen, ansonsten macht das der Browser.

```
linux:~ # telnet ilab.net.in.tum.de 80
Trying 131.159.15.226...
Connected to ilab.net.in.tum.de.
Escape character is '^]'.
GET /pages/view.php?address=p3&config=2010ss HTTP/1.1
```

[redacted]

host: ilab.net.in.tum.de

```
HTTP/1.1 200 OK
Date: Thu, 08 Jul 2010 16:20:20 GMT
Server: Apache/2.2.8 (Linux/SUSE)
X-Powered-By: PHP/5.2.6
Transfer-Encoding: chunked
Content-Type: text/html

224b
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<HTML>
 <HEAD>
  <meta http-equiv="Content-Language" content="en">
  <meta name="generator" content="labsystem.m-o-p.de">
  <link rel="stylesheet" type="text/css" href="../css/sys/labsys_mop_basic.css">
  <link rel="stylesheet" type="text/css" href="">

  <link rel="stylesheet" type="text/css" href="../css/labsys_user_style_ss06.css">

  <link rel="stylesheet" type="text/css" href="../css/sys/labsys_mop_print_theme.css" media="print">

  <link rel="shortcut icon" href="../syspix/favicon.ico">

  <script src="../pages/scripts.js" type="text/javascript" language="javascript"></script>


  <TITLE>Welcome to the ilab's portal for SUMMER term 2010! [guest@ilab summer term 2010]</TITLE>

...

 </BODY>
 </HTML>
```

Connection closed by foreign host.

[redacted]

28.

a)

3/3

```
<?xml version="1.0" encoding="UTF-8" ?>

<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
 <xs:element name="assistant">
  <xs:complexType mixed="true">
   <xs:attribute name="type" type="xs:NMTOKEN" use="required" />
  </xs:complexType>
 </xs:element>

 <xs:element name="lecture">
  <xs:complexType>
   <xs:sequence>
    <xs:element ref="name" />
    <xs:element ref="lecturer" />
    <xs:element ref="tutors" />
   </xs:sequence>
   <xs:attribute name="xsi:noNamespaceSchemaLocation" type="xs:string" use="required" />
  </xs:complexType>
 </xs:element>

 <xs:element name="lecturer">
  <xs:complexType mixed="true" />
 </xs:element>

 <xs:element name="name">
  <xs:complexType mixed="true" />
 </xs:element>

 <xs:element name="tutors">
  <xs:complexType>
   <xs:sequence>
    <xs:element ref="assistant" maxOccurs="unbounded" />
   </xs:sequence>
  </xs:complexType>
 </xs:element>

</xs:schema>
```

V

b)

1/1

XML Schema, abgekürzt XSD, ist eine Empfehlung des W3C zum Definieren von Strukturen für XML-Dokumente. Ein XML-Schema beschreibt in einer komplexen Schemasprache Datentypen, einzelne XML-Schema-Instanzen (Dokumente) und Gruppen solcher Instanzen. Sinn der Sache ist eine festgelegte Nutzung entsprechend definierter XML-Dokumente.

V

3/3

c)

Das Basis-CSS sieht folgendermaßen aus:

```
table[class="lecture"] {
 position:absolute;
 top:10px;
 left:10px;
```

```
  background-color:#CCFFCC;
  padding:5px;
}

table[class="name"] {
  position:relative;
  display:block;
  background-color:#ffffff;
  color:#000000;
  font-family:Arial,Helvetica,sans-serif;
  font-size:48px;
  font-weight: bold;
  padding:5px;
}

p[class="lecturer"] {
  position:relative;
  display:block;
  color:#000000;
  font-family:Arial,Helvetica,sans-serif;
  font-size:36px;
  font-style: italic;
  padding:5px;
}

p[class="assistant"] {
  position:relative;
  display:block;
  color:#000000;
  font-family:Arial,Helvetica,sans-serif;
  font-size:32px;
  padding:5px;
}
```

Damit wird dann ein XSLT mit HTML-Content erstellt:

```
<?xml version="1.0" encoding="iso-8859-1"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

<xsl:template match="/">
 <html><head>
 <link rel="stylesheet" type="text/css" href="example.css" />
 </head><body>
<table class="lecture"><tr><td>
<xsl:apply-templates />
</td></tr></table>
 </body></html>
</xsl:template>

<xsl:template match="lecturer">
   <p class="lecturer"><xsl:value-of select="." /></p>
</xsl:template>

<xsl:template match="name">
   <table class="name"><tr><td><xsl:value-of select="." /></td></tr></table>
</xsl:template>

<xsl:template match="assistant[@type='staff']">
   <p class="assistant"><xsl:value-of select="." /> (staff)</p>
</xsl:template>

<xsl:template match="assistant[@type='student']">
```

[redacted]

```
  <p class="assistant"><xsl:value-of select="." /> (student)</p>
</xsl:template>

</xsl:stylesheet>
```

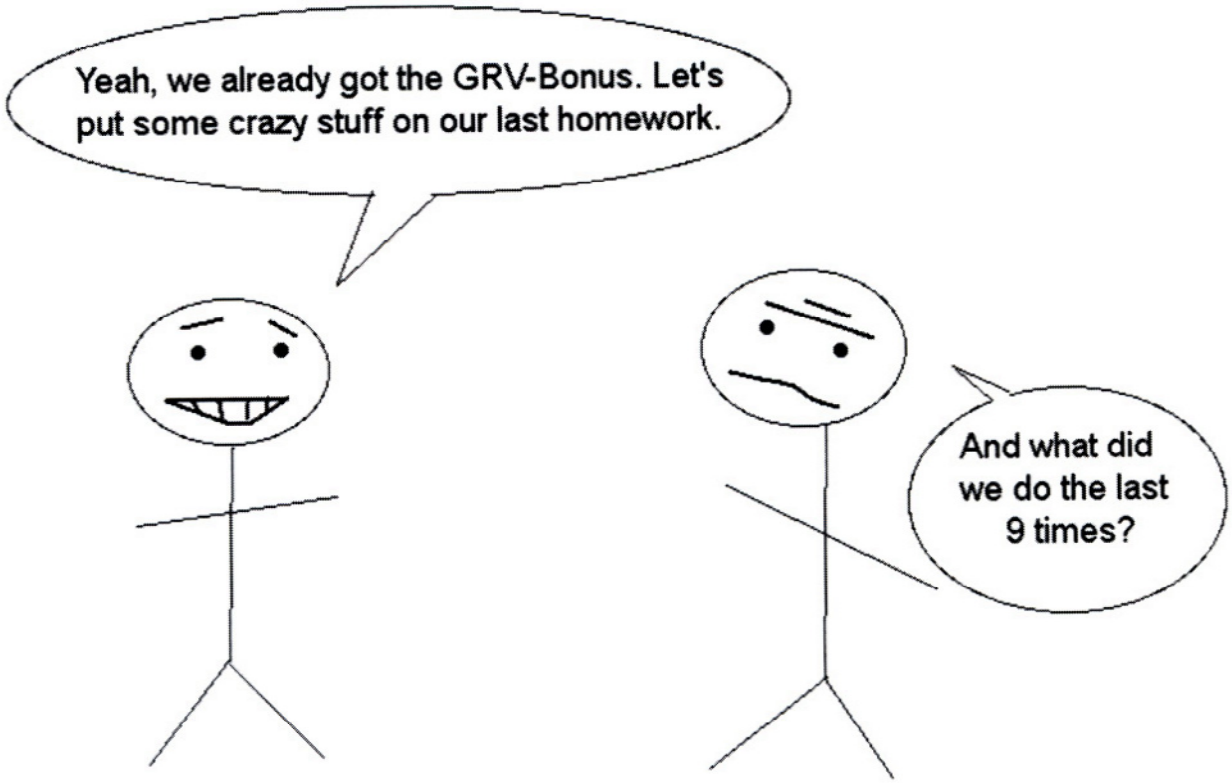
d)

1/1

→ bessere Übersicht
→ Unterteilung in Tag-Klassen
→ sowohl menschen- als auch maschinenlesbar
→ bessere Kooperation und Anpassbarkeit bei dynamischer Generierung von XML-Dateien im Bezug auf Design und Anordnung

2

29.



Es dankt das Team AWESOME in [redacted]
erfolgreiche und angenehme Zusammenarbeit.

Nice!

Nur deswegen muss man den 0,6 Bonus geben!