Name/Matrikel:

Klausur + Musterlösung

Betriebssysteme WS 2015/16

9.03.16

Tragen Sie auf dem Deckblatt Ihre Daten in Druckbuchstaben ein.

Vorname:	
Nachname:	
Matrikelnummer:	
Studienfach:	

Bitte tragen Sie auf jeder Seite Ihre Matrikelnummer ein und überprüfen Sie diese Klausur auf Vollständigkeit (16 Aufgaben!).

Verwenden Sie ausschließlich die beigefügten Blätter. Die Rückseite können Sie als Schmierpapier für Ihre Notizen verwenden. Sollten diese nicht ausreichen, so wenden Sie sich bitte an die Aufsicht. Ein Taschenrechner (kein Handy!) ist erlaubt. Andere Hilfsmittel (z. B. Handys, Bücher, eigenes Papier, etc...) sind verboten. Die Benutzung gilt als Täuschungsversuch und führt zum Ausschluss von der Klausur.

Die Klausur besteht aus zwei Teilen und enthält jeweils 8 Aufgaben mit insgesamt maximal 160 möglichen Punkten. Davon sind für eine volle Leistung (100 Prozent) in jedem Teil 60 Punkte zu erreichen. Die Anzahl der Punkte entspricht ungefähr der Bearbeitungszeit in Minuten. Für Teilnehmer im Bachelorstudiengang werden beide Teile zusammen gewertet; für Masterstudenten wird jeder Teil gesondert bewertet; ebenso die Übungspunkte.

Die maximale Bearbeitungszeit für beide Teile beträgt insgesamt 180 Minuten.

Viel Erfolg !!!

0	1	2	3	4	5	6	7	8	9	10 A	11 B	12 C	13 D	14 E	15 F
0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111

Korrekturnotizen (nicht ausfüllen!) _____

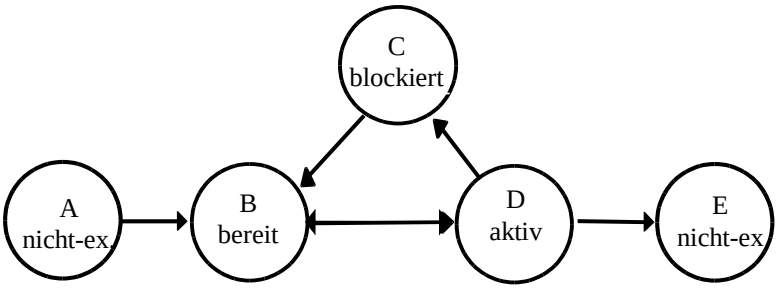
BS1								BS2								Σ	Ü	Note
1 10	2 12	3 10	4 12	5 10	6 8	7 6	8 12	9 12	10 6	11 9	12 12	13 12	14 8	15 6	16 15			

Name/Matrikel:

Multiple Choice-Aufgaben (Mehrfachantworten sind möglich!) 10 Punkte

1. Bei welchen Prozesszuständen arbeiten Warteschlangen?

○ A
○ B
○ C
○ D
○ E



2. Was ist eine Interrupt Service Routine?
   - ○ Die Updatefunktion des Betriebssystems
   - ○ Für einen Interrupt aufzurufender Betriebssystem-Code
   - ○ wird nur von Softwareinterrupts benutzt
   - ○ schneller Onlinesupport

3. Zu einer virtuellen Maschine gehören unbedingt …
   - ○ Schnittstellen zu ihren Funktionen
   - ○ Schnittstellen zu einer reellen Maschine
   - ○ Schnittstellen zum Betriebssystem

4. Der Banker Algorithmus …
   - ○ testet, ob in einem System eine Verklemmung vorliegt.
   - ○ stellt sicher, dass es zu keinen Verklemmungen kommen kann.

5. Im allgemeinem Fall ist „*n* Tasks auf *m* Prozessoren zu verteilen“
   - ○ in O(*n***m*) zu lösen
   - ○ einfach, wenn *n* kleiner ist als *m*
   - ○ ist NP-hart für *m* > *2*

6. Eine Speicherverwaltung, die Segmentierung unterstützt …
   - ○ hat genau einen linearen Adressraum
   - ○ ermöglicht, dass der Adressraum größer ist als der physikalische Hauptspeicher
   - ○ bietet eine automatische Relozierung des Codes
   - ○ teilt den Speicher in Segmente fester Größe
   - ○ bietet explizite Zugriffskontrolle zum Speicher

7. Eine Subnetzmaske dient zur …
   - ○ Identifikation des Routers
   - ○ Test, ob eine IP-Adresse im selben Subnetz liegt
   - ○ Filterung von Fehlerbits im TCP/IP-Protokoll

8. Eigenschaften des Raid 0/1:
   - ○ In Raid 0/1 sind die Platten in Streifen aufgeteilt
   - ○ Raid 0/1 sieht Spiegelplatten vor
   - ○ Raid 0/1 benötigt eine Zwangssynchronisation der Platten
   - ○ Raid 0/1 bietet Fehlerkorrekturmöglichkeit
   - ○ Raid 0/1 ist immer langsamer als Raid 2

9. Ein Geräteadressregister dient …
   - ○ zur Festlegung der Speicheradressen im Gerät für den Datentransfer
   - ○ zur Umrechnung der virtuellen Speicheradresse des Gerätes
   - ○ zur Speicherung der Herstelleradresse
   - ○ als Speicherplatz für die Treiberadresse

10. Threads …
    - ○ sind für Multiuser-Betrieb notwendig
    - ○ ermöglichen parallele Ausführung von Programmabschnitten
    - ○ ermöglichen die Portierung von Programmen zwischen Unix und NT
    - ○ müssen beim Zugriff auf gemeinsame Variablen synchronisiert werden
    - ○ ermöglichen Multiprozessorkommunikation

Lösungen

1.
   - ○ A
   - ✓ B
   - ✓ C
   - ○ D
   - ○ E

2. Was ist eine Interrupt Service Routine?
   - ○ Die Updatefunktion des Betriebssystems
   - ✓ Für einen Interrupt aufzurufender Betriebssystem-Code
   - ○ wird nur von Softwareinterrupts benutzt
   - ○ schneller Onlinesupport

3. Zu einer virtuellen Maschine gehören unbedingt …
   - ✓ Schnittstellen zu ihren Funktionen
   - ○ Schnittstellen zu einer reellen Maschine
   - ○ Schnittstellen zum Betriebssystem

4. Der Banker Algorithmus
   - ✓ testet, ob in einem System eine Verklemmung vorliegt.
   - ○ stellt sicher, dass es zu keinen Verklemmungen kommen kann.

5. Im allgemeinem Fall ist „*n* Tasks auf *m* Prozessoren zu verteilen“
   - ○ in O(*n***m*) zu lösen
   - ✓ einfach, wenn *n* kleiner ist als *m*
   - ✓ ist NP-hart für *m* > *2*

Name/Matrikel:

6. Eine Speicherverwaltung, die Segmentierung unterstützt …
   - ○ hat genau einen linearen Adressraum
   - ✓ ermöglicht, dass der Adressraum größer ist als der physikalische Hauptspeicher
   - ✓ bietet eine automatische Relozierung des Codes
   - ○ teilt den Speicher in Segmente fester Größe
   - ✓ bietet explizite Zugriffskontrolle zum Speicher

7. Eine Subnetzmaske dient zur …
   - ○ Identifikation des Routers
   - ✓ Test, ob eine IP-Adresse im selben Subnetz liegt
   - ○ Filterung von Fehlerbits im TCP/IP-Protokoll

8. Eigenschaften des Raid 0/1:
   - ✓ In Raid 0/1 sind die Platten in Streifen aufgeteilt
   - ✓ Raid 0/1 sieht Spiegelplatten vor
   - ○ Raid 0/1 benötigt eine Zwangssynchronisation der Platten
   - ✓ Raid 0/1 bietet Fehlerkorrekturmöglichkeit
   - ○ Raid 0/1 ist immer langsamer als Raid 2

9. Ein Geräteadressregister dient …
   - ✓ zur Festlegung der Speicheradressen im Gerät für den Datentransfer
   - ○ zur Umrechnung der virtuellen Speicheradresse des Gerätes
   - ○ zur Speicherung der Herstelleradresse
   - ○ als Speicherplatz für die Treiberadresse

10. Threads …
    - ○ sind für Multiuser-Betrieb notwendig
    - ✓ ermöglichen parallele Ausführung von Programmabschnitten
    - ○ ermöglichen die Portierung von Programmen zwischen Unix und NT
    - ✓ müssen beim Zugriff auf gemeinsame Variablen synchronisiert werden
    - ○ ermöglichen Multiprozessorkommunikation

Jede richtige Aufgabe 1 Punkt, jede Falsche null Punkte.

Name/Matrikel:

Aufgabe 2 Multiprozessor-Scheduling (12 Punkte)

Gegeben seien die Prozesse A-K. Die Prozesse benötigen die in Tabelle ausgewiesenen Bedienzeiten.

A	B	C	D	E	F	G	H	I	J	K
2	4	3	4	3	5	3	5	4	2	3

Die Ausführungsreihenfolge unterliegt mit der in der Vorlesung vorgestellten Präzedenzrelation „≫“ den folgenden Beschränkungen:

A ≫ D ≫ G ≫ K, B ≫ D ≫ H≫ K, C ≫ E ≫ H, E ≫ I ≫ K, C ≫ F ≫ J ≫ K

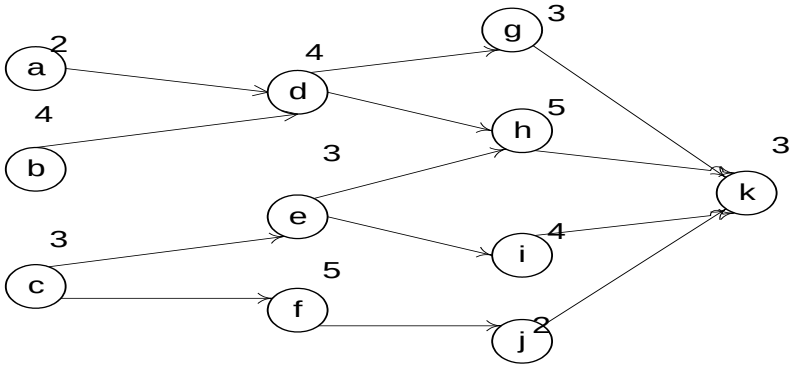
a) Zeichnen Sie den dazu gehörenden Präzedenzgraphen. In den Knoten stehe jeweils die Prozessbezeichnung sowie dabei auch die Bedienzeit. (4 Punkte)

b) Was ist ein *kritischer Pfad* durch den Graphen, wie lautet er und wie lang ist er? (2 Punkte)

c) Zeichen sie jeweils ein Gantt - Diagramm für i) *Earliest Scheduling* und ii) *Latest Scheduling* wenn Sie über ein 3 Prozessor-System verfügen (6 Punkte)

Lösung

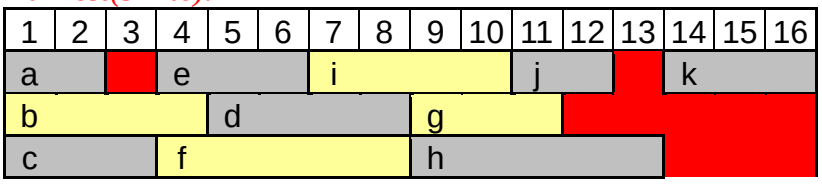
a) *Zeichnen Sie einen Präzedenzgraphen. In den Knoten stehe jeweils die Prozessbezeichnung sowie die Bedienzeit.*



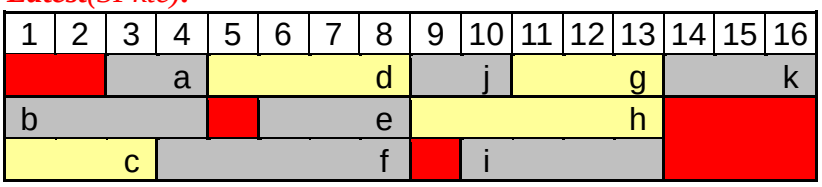
b) *Der kritische Pfad ist der längste Pfad (0,5Pkt) ohne Zykel (0,5Pkt) von einem Startknoten zu einem Endknoten. Hier ist es der Pfad B-D-H-K der Länge 4+4+5+3=16 (1Pkt), was mit den Gantt-Diagrammen übereinstimmt.*

c) *Zeichen sie jeweils ein Gantt - Diagramm für i) Earliest Scheduling und ii) Latest Scheduling*

***Earliest**(3Pkte):*



***Latest**(3Pkte):*



Je 2 Pkte fürs richtige Konstruktionsprinzip, 1 Pkt für die richtige Lösung.

Name/Matrikel:

Aufgabe 3 Echtzeitsysteme (10 Punkte)

a) Was ist der Unterschied zwischen Hart- und Soft-Echtzeitsystemen? (2 Pkte)

b) Angenommen, ein Roboter hat folgende Tasks auszuführen:

- Die Schrittposition zu erfassen dauert 2ms und erfolgt alle 20 ms.
- Alle 4 Sekunden GPS-Position bestimmen. Dauer ist 100 ms pro Position.
- Pro Sekunde 10 mal eine Ultraschall-Nachbarschaftsortung durchführen. Dauer: 20 ms pro Messung.
- Eine Video-Analyse mit einer Frame-Frequenz von 25 Hz durchführen. Sie dauert 24 ms pro Analyse.
- Jede Sekunde muss der Status zur zentralen Steuerung gesendet werden. Die notwenigen Operationen benötigen 30 ms CPU-Zeit.
- Das Abstandsradar liefert alle 100 ms einen Wert, dessen Verarbeitung jeweils 5 ms dauert.

Frage: Welche Lastreserven (Restkapazität) hat das System? (5 Pkte)

c) Erläutern Sie die Funktionsweise des *Guaranteed Percentage-Schedulings* GPS.(3 Pkte)

Lösung

a) Bei Soft-Echtzeitsystemen sind die Zeitbeschränkungen nur gewünscht (1Pkt), bei Hart-Echtzeitsystemen sind es verbindliche, garantierte Zeitschranken (1Pkt).

b) Wir haben eine Auslastung von H = 2/20 + 0,1/4 + 20/100 + 24/40 + 30/1000 + 5/100 = 10%+2,5%+20%+60%+3%+5% = 100,5%. Es gibt keine Reserven, das System ist überlastet und kann nicht funktionieren. *(3Pkte für die richtige Argumentation, 2 Pkte fürs richtige Ergebnis)*

c) Beim GPS wird die Anzahl der Zeitscheiben jedes Prozesses proportional zu der CPU-Belastung gewählt. (3 Pkt)

Name/Matrikel:

Aufgabe 4 Prozess – Verklemmungen (12 Punkte)

a) Geben sie ein Beispiel für eine *Race Condition* (2 Pkte)

b) Die Prozesse P_1 und P_2 benötigen exklusiven Zugriff auf die kritischen Abschnitte A_1, A_2, A_3. Ein Abschnitt wird mit dem Befehl $P(S_x)$ reserviert und mit $V(S_x)$ (x=1,2,3) freigegeben. Wenn ein Prozess versucht einen Abschnitt zu reservieren, den bereits ein anderer Prozess benutzt, blockiert dieser mindestens so lange, bis der Abschnitt mittels V() freigegeben wird. Der Programmcode für die Prozesse P_1 und P_2 ist:

P_1:

```
P(S1);
if(b) {
   P(S2);
   V(S1);
   P(S3);
} else {
   P(S3);
   V(S1);
   P(S2);
}
V(S2);
V(S3);
```

P_2:

```
P(S3);
if(b) {
   P(S2);
   V(S3);
   V(S2);
   P(S1);
} else {
   V(S3);
   P(S1);
   P(S2);
   V(S2);
}
V(S1);
```

Untersuchen Sie unter der Bedingung, dass die globale boolesche Variable `b` beliebig initialisiert und nicht geändert wird, ob es zu Verklemmungen kommen kann. (10 Pkte)

Lösung

a) Geben sie ein Beispiel für eine *Race Condition*

Eine *Race Condition* entsteht z. B. dann, wenn ein Prozess Daten aus einem global bekannten Speicher (1 Pkt) abruft, lokal speichert und unterbrochen wird. Der zweite Prozess wird erweckt bzw. gestartet und ändert die globalen Daten, so dass Prozess 1 lokal nicht mehr aktuelle Daten hat: Prozess 2 überholt Prozess 1. Wenn Prozess 1 wieder zum Zug kommt, so führt er Änderungen durch, basierend auf den mittlerweile ungültigen Informationen. (schreiben + lesen: 1 Punkt)

Ein konkretes Beispiel ist das Einhängen und Aushängen bei der PCB-Warteschlange, siehe Vorlesung.

b) In Abhängigkeit vom Wert von `b` reduziert sich der Ablauf beider Prozesse auf wenige Instruktionen. Für diese parallelen Abläufe müssen wir prüfen, ob alle vier Verklemmungsbedingungen erfüllt werden können oder nicht. Da alle kritischen Abschnitte durch Semaphore ohne Zeitlimits geschützt werden, sind die Bedingungen *mutual exclusion*, *no preemption* und *hold_and_wait* schon gegeben. Es bleibt zu untersuchen, ob auch ein *circular_wait* auftritt.

Fall 1: `b = true`

Die Prozessinstruktionen reduzieren sich zu:

P_1:

```
P(S1);
P(S2);
V(S1);
P(S3);
V(S2);
V(S3);
```

P_2:

```
P(S3);
P(S2);
V(S3);
V(S2);
P(S1);
V(S1);
```

Name/Matrikel:

Dabei ist allerding eine Verklemmung möglich, wenn die Prozesse zeitlich folgendermaßen ablaufen:

P1	P2	
P(S_1)	P(S_3)	P_2 belegt A_3
P(S_2)		P_1 belegt A_2
V(S_1)	P(S_2)	P_2 blockiert, da A_2 belegt ist
P(S_3)		P_1 blockiert, da A_3 belegt ist

Also ergibt sich die typische Situation: P_1 hat A_2 und will A_3, P_2 hat A_3 und will A_2.

Fall 2: `b = false`

P_1:

```
P(S1);
P(S3);
V(S1);
P(S2);
V(S2);
V(S3);
```

P_2:

```
P(S3);
V(S3);
P(S1);
P(S2);
V(S2);
V(S1);
```

Hier ist keine Verklemmung möglich: Für alle Paare (i,j), i, j∈{1,2,3}, i≠j gibt es keine Kombination, in der ein Prozess zuerst A_i und dann A_j während der andere zuerst A_j und dann A_i anfordert, ohne dass die jeweils zuerst angeforderte Ressource (der Abschnitt) wieder frei gegeben worden ist.

Für jeden Fall jeweils 5 Punkte, wenn richtig argumentiert wurde.

Name/Matrikel:

Aufgabe 5 Producer-Consumer-Systeme (10 Punkte)

Gegeben sei der folgende unvollständige Programmcode zur Lösung des Erzeuger-Verbraucher-Problems. Ordnen Sie die folgenden 8 fehlenden Befehle den Zeilen (10, 11, 14, 15, 19, 20, 23, 24) im Code zu.

***Lösung**: 1 Pkt pro Eintrag, 2 Pkte wenn alles richtig*

s mutex.acquire() gehört in Zeile _____ 11 oder 20

s mutex.acquire() gehört in Zeile _____ 20 oder 11
s mutex.release() gehört in Zeile _____ 14 oder 23

s mutex.release() gehört in Zeile _____ 23 oder 14

s empty.acquire() gehört in Zeile _____ 19

s empty.release() gehört in Zeile _____ 15

s full.acquire() gehört in Zeile _____ 10

s full.release() gehört in Zeile _____ 24

Code:

```
1   REGAL = []
2   REGAL_KAPAZITAET = 5
3
4   s_empty    = threading.Semaphore(REGAL_KAPAZITAET)
5   s_full     = threading.Semaphore(0)
6   s_mutex    = threading.Semaphore(1)
7
8   def verbraucher():
9       while True:
10          # Zeile 10
11          # Zeile 11
12          portion = REGAL.pop()
13          print('Lager hat noch "%s", %i Portionen übrig.' % (portion, len(REGAL)))
14          # Zeile 14
15          # Zeile 15
16
17  def erzeuger():
18      while True:
19          # Zeile 19
20          # Zeile 20
21          portion = "McBurger"
22          REGAL.append(portion)
23          # Zeile 23
24          # Zeile 24
```

Name/Matrikel:

Aufgabe 6 Speicherverwaltungs – Strategien (8 Punkte)

In einem Swapping-System gibt es freie Speicherstücke der folgenden Größe in der angeführten Reihenfolge: 10KiB, 4KiB, 20KiB, 20KiB, 18KiB, 7KiB, 9KiB, 12KiB,und 15KiB.

- Welcher freie Bereich wird gewählt für eine Anforderung von 12 KiB, 10 KiB und 9 KiB?
- Wie sieht die Liste der freien Speicherstücke hinterher aus? (bitte zeichnen)

Beantworten Sie beide Fragen jeweils für die Strategien

a) FirstFit, b) NextFit, c) BestFit und d)WorstFit.

Lösung

Speicherstücke:

1	2	3	4	5	6	7	8	9
10KiB	4KiB	20KiB	20KiB	18KiB	7KiB	9KiB	12KiB	15KiB

a) FirstFit: 3, 1, 4 (2 Pkte)

0KiB	4KiB	8KiB	11KiB	18KiB	7KiB	9KiB	12KiB	15KiB

b) NextFit: 3, 4, 4 (2 Pkte)

10KiB	4KiB	8KiB	1KiB	18KiB	7KiB	9KiB	12KiB	15KiB

c) BestFit: 8, 1, 7 (2 Pkte)

0KiB	4KiB	20KiB	20KiB	18KiB	7KiB	0KiB	0KiB	15KiB

d) WorstFit: 3, 4, 5 (2 Pkte)

10KiB	4KiB	8KiB	10KiB	9KiB	7KiB	9KiB	12KiB	15KiB

Je 1 Punkt pro richtige Wahl und pro korrekte Liste.

Aufgabe 7 Virtueller und reeller Speicher (6 Punkte)

a) Die Seitengröße eines Speichers beträgt 1024 Bytes, die Wortbreite n Bits. Beschreiben Sie, wie aus den virtuellen Adressen einstufig die physikalischen Hauptspeicheradressen erzeugt werden. (3 Punkte)

b) Ein Computer mit 32 Bit-Adressen benutzt eine zweistufige Adresskonversion. Virtuelle Adressen bestehen aus 9 Bit für die Top-Level-Seite, 11 Bit für die zweite Seitentabelle und dem Offset. Wie groß ist eine Seite und wie viele Seiten können im Adressraum verwaltet werden? (3 Punkte)

Lösung

a) Bei einer einstufigen Adresskonversion teilt sich eine virtuelle Adresse in den Index der Seitentabelle und den Offset. Der Index wird in den höchstwertigsten Bits gespeichert, der Offset in den geringwertigsten. (1Pkt)

Bei einer Seitengröße von 2^{10} = 1024 Bytes, werden 10 Bit für den Offset benötigt. Bei einer Adresslänge von n Bit nehmen die höchstwertigen n-10 Bit den Index der Seitentabelle ein. (1Pkt)

Die physikalische Adresse erhält man, indem an den Eintrag, der sich in der Seitentabelle an diesem Index befindet, der Offset konkateniert wird. (1Pkt)

b) Offset = 32-(9+11) = 12 Bit (1Pkt)

$\Rightarrow$ Seitengröße = 2^{12} = 4096 Bytes (1Pkt)
$2^9 * 2^{11} = 2^{20} = 1048576 = 1M$ Seiten können verwaltet werden (1Pkt).

Name/Matrikel:

Aufgabe 8 Seitenersetzungsstrategien (12 Punkte)

Wir betrachten die folgende Referenzfolge von Seitenzugriffen: 1, 2, 3, 1, 5, 3, 4, 1, 2, 3, 5, 4, ausgehend von einer Speichergröße von 4 Seiten und einem anfangs vollständig freien Speicher.

a) Skizzieren Sie in einer Tabelle die Auswirkungen der Ersetzungsstrategien FIFO, LRU und optimalen Strategie auf die vier gespeicherten Seiten.
b) Wieviele *page faults* erzeugt die jeweilige Strategie in diesem Fall?

Lösung

1	2	3	1	5	3	4	1	2	3	5	4	*FIFO*
						x	**x**	**x**	**x**	**x**	**x**	*6pf*
1	1	1	1	1	1	**4**	4	4	4	**5**	5	
	2	2	2	2	2	2	**1**	1	1	1	**4**	
		3	3	3	3	3	3	**2**	2	2	2	
				5	5	5	5	5	**3**	3	3	

1	2	3	1	5	3	4	1	2	3	5	4	*LRU*
						x		**x**		**x**	**x**	*4pf*
1	1	1	1	1	1	1	1	1	1	1	**4**	
	2	2	2	2	2	**4**	4	4	4	**5**	5	
		3	3	3	3	3	3	3	3	3	3	
				5	5	5	5	**2**	2	2	2	

1	2	3	1	5	3	4	1	2	3	5	4	*OPT*
						x				**x**		*2pf*
1	1	1	1	1	1	1	1	1	1	**5**	5	
	2	2	2	2	2	2	2	2	2	2	2	
		3	3	3	3	3	3	3	3	3	3	
				5	5	**4**	4	4	4	4	4	

Man könnte zuletzt statt 1 auch 2 oder 3 ersetzen.

Jeweils 2 Punkte für die richtige Seitenersetzung, sowie 2 Punkte für die korrekte Zahl von *page faults.*

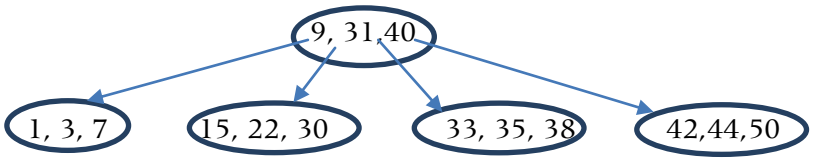
Ende BS1 - Klausur Teil 1

Name/Matrikel:

Beginn BS2 - Klausur Teil 2

Aufgabe 9 Dateiverwaltung: B- und B*-Bäume (12 Punkte)

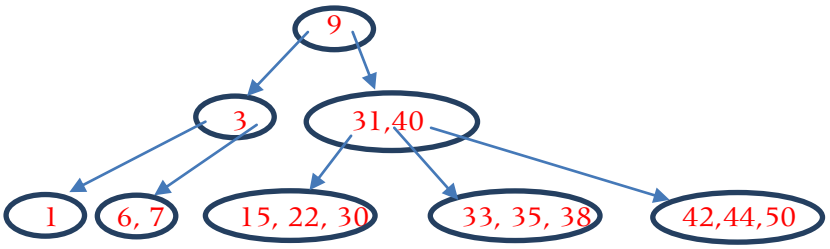
Zeichnen Sie den Baum, der sich für m = 4 nach Einfügen des Schlüssels 6 ergibt, in folgenden Baum,



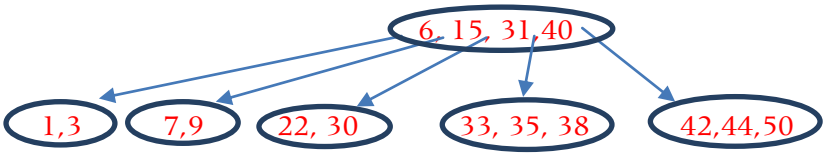
a) wenn dies ein B-Baum ist (6 Punkte)
b) wenn dies ein B*-Baum ist (6 Punkte)

Lösung

a) Der B-Baum zerteilt sich zunächst im (1,3,6,7)-Container und der Schlüssel mit Index $\lceil m/2 \rceil = 2$ wandert nach oben. (3Pkte)
Dies gilt analog für den so entstandenen Container (3,9,31,40), so dass eine weitere Ebene entsteht. (3 Pkte).
1 Punkt Abzug, wenn die Teilungsmethode nicht immer die Selbe ist.



b) Der B*-Baum zerteilt sich ebenfalls, da (1,3,6,7) übervoll und (15,22,30) voll sind. Zusammenfügen aller Elemente der beiden Container und des Schlüssels im Elterncontainer (1,3,6,7,9,15,22,30) und Selektieren der Schlüssel mit den Indizes $A = \lfloor (2m-2)/3 \rfloor + 1 = 3$ und $B = A + \lfloor (2m-1)/3 \rfloor + 1 = 6$, also Schlüssel 6 und 15, die statt der 9 nach oben in den Elternknoten wandern. (4 Pkte)
Die Wurzel kann hier 4 Schlüssel aufnehmen, so dass wir hier fertig sind. (2 Pkte)



Zur Information: Beim Überfließen wandern Schlüssel über den Elternknoten in einen Container gleicher Stufe. Die Schlüssel können nicht einfach nur in den Elternknoten überfließen und dort bleiben.

Name/Matrikel:

Aufgabe 10 Dateisysteme (6 Punkte)

a) Erklären Sie den Unterschied zwischen *hard links* und *symbolic links*. (2 Punkte)

b) Welche Dateioperation kann man als Semaphor verwenden? (2 Punkte)

c) Ist die Operation *open*/*close file* grundsätzlich notwendig? Kennen Sie andere Konzepte, um ohne *open*/*close* auf Daten zuzugreifen? (2 Punkte)

Lösungen

(a) *hard links* sind *device*-intern (nur gültig für ein Gerät) (1Pkt) und die *symbolic links* sind *device*-übergreifend (gültig über Geräte- und Netzgrenzen hinweg).(1Pkt)

(b) Die *create_file()*- Operation. Oder auch eine *lock_file()*-Operation (falls existent) ist brauchbar. (2Pkte) *Eine davon reicht.*

(c) Nein ist sie nicht. (0,5 Pkte) Es gibt noch die Option der verbindungslosen Kommunikation, bei der bei jedem *read/write* alle Dateiangaben (Name, Blocknr., etc) mit übergeben werden. Aus Performanzgründen werden sie aber meist nicht verwendet. (1,5Pkte)

Jedes weitere ex. Konzept: je 2 BonusPkte

Aufgabe 11 RAID (9 Punkte)

Angenommen, Sie haben ein System S_1 aus 5 Laufwerken, die jeweils eine Kapazität von 1TB haben und mit der Wahrscheinlichkeit p = 0,1% ausfallen. Sie sind als RAID 5 zusammengeschaltet.

Außerdem haben Sie noch ein einzelnes Laufwerk S_2 von 4 TB, das mit $p_2 = 0{,}001\%$ ausfällt.

a) Welches System hat mehr Speicherkapazität für Anwenderdaten und warum? (2 Punkte)
b) Welches System ist zuverlässiger als das andere und warum? (5 Punkte)
c) Wenn beide Systeme gleich viel kosten, welches von beiden sollte man kaufen, und warum? (2 Punkte)

Lösung

a) Ein RAID5-System hat jeweils eine Einheit (Platte) als Aufnahmegerät für die Paritätsinformation. Zwar sind die Daten in *striping*-Manier verteilt, aber die Speicherkapazität ändert sich nicht, so dass 4 TB für Nutzerdaten zur Verfügung stehen. Im Vergleich zu dem anderen System gibt es für die Speicherkapazität also weder Vorteil noch Nachteil. (2Pkte)

a) Die Ausfallwahrscheinlich P ist $P(S_1) = 1 - P(ok) = 1 - [P(\text{alle 5 Einheiten ok}) + P(\text{nur eine der Platten ist ausgefallen})] = 1 - [\,(1-p_1)^5 + 5\,(1-p_1)^4 p_1\,] = 1 - 0{,}999^5 - 0{,}005 \times 0{,}999^4 = 0{,}998\ 10^{-5}$. (2Pkte für Ansatz, 2 Pkte für Ergebnis) Im Vergleich zu S_2 mit $p_2 = 0{,}001\% = 10^{-5}$ ist das RAID-System also etwas zuverlässiger, aber nicht viel. (1Pkte)

b) Entscheidend sind hier die Reparaturkosten: Fallen zwei Platten im RAID-System (und damit das ganze System) aus, so kostet es nur zwei Fünftel des zweiten Systems, um es wieder in einen sicheren Zustand zu versetzen. (2Pkte)

Name/Matrikel:

Aufgabe 12 Serverzustände (12 Punkte)

a) Was sind die Vor- und Nachteile von zustandsbehafteten Servern bei Aufträgen? (6 Pkte)
b) Was sind die Vor- und Nachteile von zustandslosen Servern? (2 Pkte)
c) Zu welcher Art von Servern kann man eine verbindungsorientierte Kommunikation aufbauen und warum ? (2 Pkte)
d) Auf welcher Art von Servern ist eine Operationssemantik als Zugriffssemantik möglich und warum ? (2 Pkte)

Lösung

a) zustandsbehaftete Server haben folgende **Vorteile**:
- schneller Datenzugriff nach Initialisierung der Kommunikation durch Bereitstellung von Dateideskriptoren und Puffern. (1Pkt)
- Auftragskopien können eliminiert werden, da die Reihenfolge (Nummerierung) der Nachrichten überprüft werden kann. (1Pkt)
- Sperren von Dateien für die Transaktionssemantik ist möglich. (1Pkt)

Nachteile sind:
- Wenn ein Client „abstürzt“ erfährt der Server dies nicht und reserviert Dateideskriptoren und Puffer unnötigerweise (Gefahr des Daten-overflow). (1Pkt)
- Versagt der Server selbst, so bemerkt der Client dies nicht und reserviert ebenfalls unnötigerweise Platz und Systemtafeln. (1Pkt)
- Es gibt eine maximale Zahl von gleichzeitigen Nutzern. (1Pkt)

b) Prinzipiell sind die **Vorteile** der zustandslosen Server die **Nachteile** der zustandsbehafteten Server und umgekehrt: der zustandslose Server spart sich die Initialisierung, aber benötigt für die einzelnen Transaktionen länger. Auftragskopien können unnötigerweise erledigt werden und der Dateizugriff ist nicht reglementiert. (1Pkt) Dafür bedeutet das Versagen des Servers oder Clients keine Probleme auf der anderen Seite: die Kommunikation vom Client wird solange wiederholt, bis sie erfolgreich war. (1Pkt)

c) Eine verbindungsorientierte Kommunikation benötigt eine Speicherung des Zustandes der Verbindung (1Pkt); dies ist nur mit einem zustandsbehafteten Server möglich. (1Pkt)

d) Die Operationssemantik benötigt keine Sperrung und lässt alle Operationen in der zufälligen Reihenfolge zu, in der sie eintreffen. (1 Pkt)

Damit ist ein zustandsloser Server ausreichend; es müssen keine Zustände der Zugriffe oder Operationen gespeichert werden bis auf die Datei selbst.
Selbstverständlich kann eine solche Zugriffssemantik auch auf zustandsbehafteten Servern implementiert und die Zustände dabei ignoriert werden. (1Pkt)

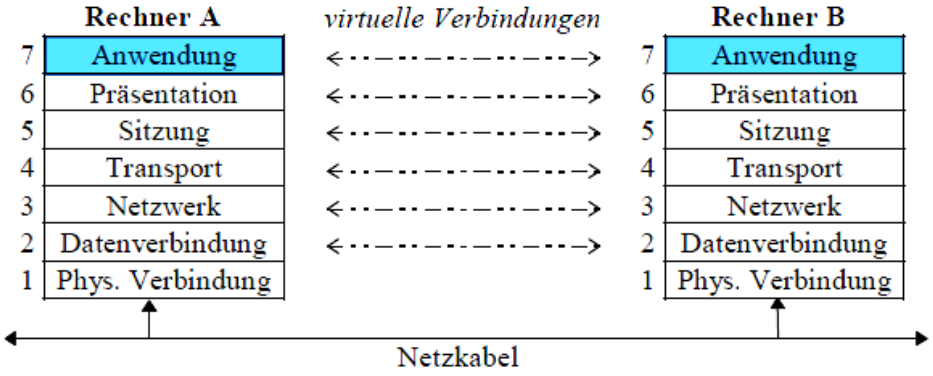
Name/Matrikel:

Aufgabe 13 Transport-Schichtenmodell (12 Punkte)

a) Zeichnen Sie ein Schichtenmodell der ISO-OSI Netzwerkschichten und benennen Sie die einzelnen Schichten (2 Pkte)
b) Welche Aufgaben hat jede einzelne Schicht? (7 Punkte)
c) Welche Modelle zur Punkt-zu-Punkt-Kommunikation haben Sie in der Vorlesung kennen gelernt? (3 Pkte)

Lösung

a) 2 Pkte, *0.5 Pkte pro zwei Schichten*



b)

Layer 7 : Anwendungsschicht
High-level Programme: FTP, Grafik, electronic mail, ...
Layer 6 : Präsentationsebene
Datenformatierung, Kodierung, Gruppierung (Records, Verschlüsselung,)
Layer 5 : Sitzungsebene
open/close-Semantik: Sender, Empfänger, high-level-Fehlerbehandlung, logon-passwords, Daten/Kontrollunterscheidung,...
Layer 4 : Transportschicht
Umwandlung in Datenpakete, Reihenfolge der Pakete, usw. Bei TCP (Transmission Control Protocol): Fehlertoleranzgrad TP0-4 festlegen
Layer 3 : Netzwerkschicht
Fragen der Netztopologie: Übertragungsweg, Umleitung (routing), Netzstatus, Grenzen, Auslastung, usw. Typisch: Internet Protocol IP
Layer 2 : Datenverbindung
Datenpakete, Unterteilung in log. Signalframes, Wiederholung bei NO-ACK. Aber: Frame-Reihenfolge ist unkontrolliert. Z.B.: Ethernet
Layer 1 : physikalische Signale
Bits -> Impulse, Freq. z.B.10BaseT

Je 1 Punkt pro Schicht für Aufgaben (nicht Bezeichnung).

c) Alle Punkt-zu-Punkt-Modelle, also Socket-Modell, Port-Modell, VPN-Modell, TransportLayerInterface-API, Named Pipes, RPC, P2P, VoiceOverIP, …, (je 0,5 Pkte) aber nicht TCP/IP, UDP etc., da Sitzungsprotokolle, für Broadcast geeignet.

Name/Matrikel:

Aufgabe 14 Zugriffssemantiken (8 Punkte)

Erläutern sie die Probleme, die entstehen können, wenn mehrere Personen gleichzeitig ein Dokument bearbeiten:

Wie kann es zu Inkonsistenzen kommen, welche Vorteile und Nachteile bieten die einzelnen Zugriffssemantiken?

Lösung

- *Inkonsistenzen* :
  Werden globale Daten, etwa gemeinsame Dateien, gleichzeitig gelesen und beschrieben, so können *race conditions* auftreten und damit die Daten inkonsistent werden. Allerdings gilt dies nur, wenn auch geschrieben wird. (2 Bonuspunkte)

- *Vor- und Nachteile:*
  1. Bei der **Read-only-Strategie** kann nur gelesen, nicht geschrieben werden. **Vorteil**: Keine Inkonsistenzen. **Nachteil**: Keine Aktualisierung der Datei. **(2Pkte)**
  2. Bei der **Operationssemantik** wird sofort jede Operation ausgeführt. **Vorteil**: Koordinieren sich mehrere Mitglieder geeignet, so kann an verschiedenen Stellen eins Dokuments parallel gearbeitet werden. **Nachteil**: Wenn es Missverständnisse gibt, so kann leicht jede Operation eines Mitglieds durch die eines anderen überschrieben werden und es entstehen Konflikte. **(2Pkte)**
  3. Bei der **Sitzungssemantik** überschreibt die Person, die zuletzt speichert, alle Änderungen, die andere Benutzer vorgenommen haben, seit sie die Datei gelesen hat. **Vorteil**: Jede Version ist in sich konsistent. **Nachteil**: Damit kommt es zu Inkonsistenzen zwischen den lokalen Versionen der Personen. **(2Pkte)**
  4. Bei der **Transaktionssemantik** wird die Datei solange für alle anderen Personen gesperrt, solange eine Person darauf arbeitet. **Vorteil**: Jeder, der arbeitet, weiss, dass seine Version nicht überschrieben wird. **Nachteil**: Wenn einer arbeitet, werden die anderen davon ausgeschlossen. **(2Pkte)**

Allgemeiner Vorteil aller Strategien: Durch die lokale Pufferung werden Performancevorteile erreicht.

Aufgabe 15 Middleware (6 Punkte)

a) Angenommen, Sie haben 100 verschiedene Programme, die über ein Netzwerk auf einer von fünf verschiedenen Datenbanken arbeiten. Dabei wird jeweils eine bestimmte Netzwerkfunktionalität (Protokoll) von zwei möglichen vorausgesetzt. Wieviele Programmversionen können Sie abdecken, wenn Sie dafür eine Middleware einführen und wie viele Versionen müssen damit nicht neu programmiert werden? (2 Punkte)

b) Was sind die Unterschiede und die Gemeinsamkeiten von einem RPC-Aufruf und einer CORBA-Anfrage? (4 Punkte)

Lösungen

a) Ohne Middleware gibt es 100 Programme. Mit Middleware haben Sie 100x5x2 = 1000 Funktions-Versionen, die möglich sind. Sie ersparen sich also 900 Programmversionen.(2Pkte)

b) **Gemeinsamkeiten**: Ein RPC-Aufruf fragt eine Dienstleistung an, ebenso ein CORBA-Aufruf. (1Pkt) Beide nutzen meist dazu die normalen Transportschichten. (1 Pkt)
Unterschiede: Ein RPC-Aufruf kennt genau seinen Server und baut eine Direktverbindung für die Dienstleistung auf. (1 Pkt) Im Unterschied dazu ist beim CORBA-Aufruf nur der CORBA-Vermittlungsrechner bekannt, der prüft, ob für die Dienstleistung ihm ein Server bekannt ist. Danach vermittelt er den Dienst, führt ihn aber nicht aus. (1 Pkt).
CORBA kann für die Anfragen RPCs benutzen; ebenso können RPCs für die Dienstleistung selbst verwendet werden.

Name/Matrikel:

Aufgabe 16 Angriffsarten: Viren und root kits (15 Punkte)

a) Was ist ein *buffer overflow*-Angriff und wie kann man ihn verhindern? (4 Pkte)
b) Was ist der Unterschied zwischen einem Virus und einem root kit? (2 Pkte)
c) Welche Arten von Root kits kennen Sie? (3 Pkte)
d) Wie kann man root kits entdecken? (6 Pkte)

Lösungen

a) Was ist ein *buffer overflow*-Angriff und wie kann man ihn verhindern? (4 Pkte)
Ein *buffer overflow*-Angriff versucht bei der Argumentübergabe den Pufferplatz über seine Grenzen hinaus zu füllen und dabei wichtige Adressen (Startadresse, Rücksprungadresse) zu überschreiben und so die Programmausführung in den eigenen Schadcode zu lenken. (2 Pkte)
Man kann einen solchen Angriff verhindern, indem man durch Prüfen der Argumentgröße vor der Argumentübergabe nicht zulässt, dass der Argumentpuffer überlaufen kann. (2 Pkte)

b) Was ist der Unterschied zwischen einem Virus und einem root kit? (2 Pkte)
Ein Virus ist ein aktives Schadprogramm; ein *root kit* dagegen verbirgt nur ein oder mehrere Schadprogramme vor den Augen des Benutzers, etwa auf den Listen von Dateien, Ordnern oder Prozessen.

c) Welche Arten von *root kits* kennen Sie? (3 Pkte)
Es gibt *user mode root kits* (Fälschung von Systembibliotheken im user mode), *kernel mode root kits* (Fälschung von System Calls im kernel mode) und *Ring-1,-2,-3 root kits* (Fälschung von Hardwarefunktionen).

d) Wie kann man *root kits* entdecken? (6 Pkte)
Die *user mode rootkits* kann man durch Vergleich von Ergebnissen (Prozess/Dateilisten) der Systembibliotheksaufrufe mit den direkten Ergebnissen von *system calls* entdecken. (2 Pkte)
Die *kernel mode rootkits* kann man durch Unterschieben eines Hypervisors und Vergleich der System calls mit den Ergebnissen einer auf dem Hypervisor parallel ablaufenden Ausgabe des gleichen, aber korrekten Betriebssystems entdecken. (2 Pkte)
Für die dritte Art von root kits gibt es zur Zeit keine Möglichkeit innerhalb des Computers, das root kit zu entdecken, sondern nur von außen durch Anschließen eines Meßgeräts (eines weiteren Computers). (2 Pkte)