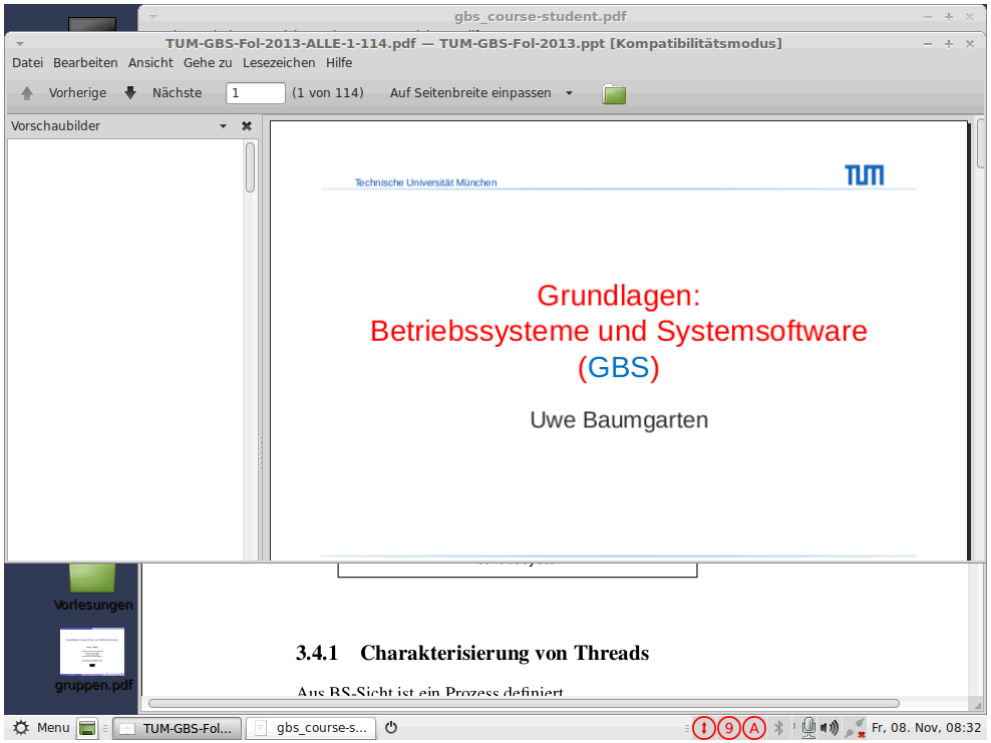
Script **generated by TTT**

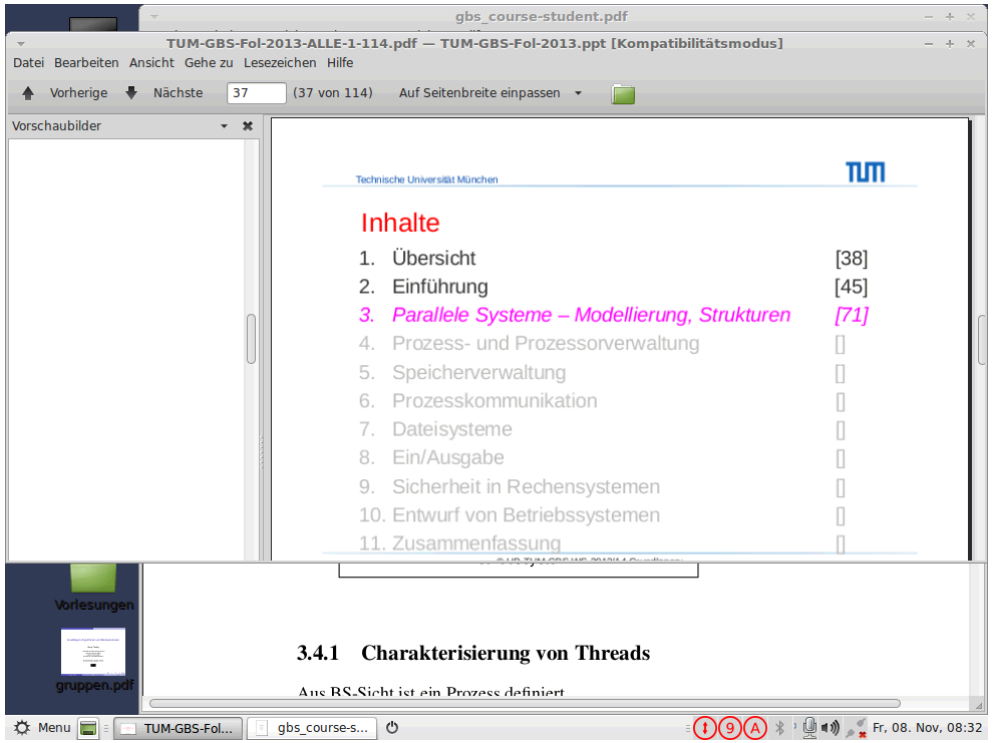
Title: Baumgarten: GBS (08.11.2013)

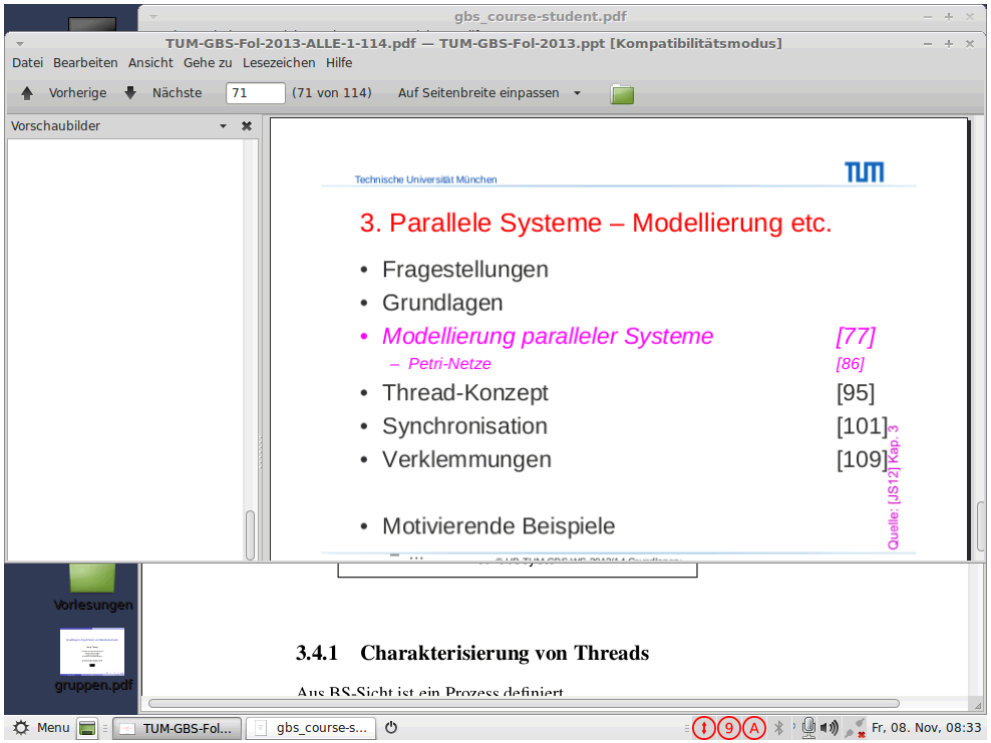
Date: Fri Nov 08 08:31:36 CET 2013

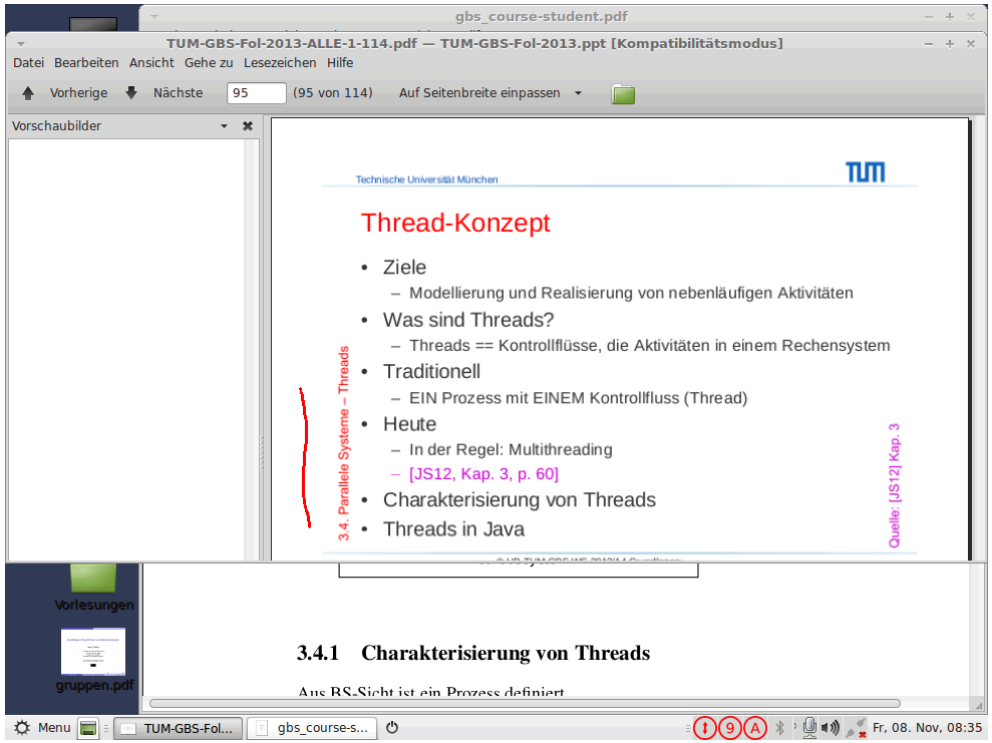
Duration: 86:30 min

Pages: 46









Charakterisierung von Threads

- Prozessbestandteile
  - Ein Adressraum (i.d.R. virtueller Adressraum (VA))
  - Eine Handlungsvorschrift (Programmcode)
  - Ein oder mehrere Aktivitätsträger (Threads)
- Motivation
  - Thread als Abstraktion eines physischen Prozessors
  - Parallelität innerhalb eines Prozesses mit einfacher Interaktion der Threads über gemeinsamen Speicher
  - Geringer Aufwand für Erzeugen und Löschen
    - Thread = leichtgewichtiger Prozess
    - Normale Prozesse = schwergewichtige Prozesse
  - Verbesserung der Performanz
    - Sowohl CPU- als auch E/A-intensive Applikationen
  - Echte Parallelität in Multiprozessor/Multicore-Systemen

Abgrenzung: Threads vs. Prozesse

- Zielsetzung
  - Prozesse gruppieren Ressourcen
  - Threads sind Aktivitätsträger
- Threadspezifische Informationen
  - Befehlszähler (PC)
  - Registerwerte
  - Keller/Stack
  - Threadzustand
- Prozessspezifische Informationen
  - Adressraum
  - Globale Variablen
  - Offene Dateien
  - Kindprozesse
  - Eingetroffenen Alarme bzw. Interrupts
  - Verwaltungsinformationen

Beispiel: Threads in Web-Servern

- Verteiler-Thread
- Server-Thread
- [JS12, Kap. 3, p. 62]

gbs_course-student.pdf
TUM-GBS-Fol-2013-ALLE-1-114.pdf — TUM-GBS-Fol-2013.ppt [Kompatibilitätsmodus]
Datei Bearbeiten Ansicht Gehe zu Lesezeichen Hilfe
Vorherige Nächste 98 (98 von 114) Auf Seitenbreite einpassen
Vorschaubilder
Technische Universität München
Beispiel: Threads in Web-Servern
• Verteiler-Thread
• Server-Thread
• [JS12, Kap. 3, p. 62]
3.4. Parallele Systeme – Threads
Quelle: [JS12] Kap. 3
Vorlesungen
gruppen.pdf
3.4.1 Charakterisierung von Threads
Aus BS-Sicht ist ein Prozess definiert
Menu TUM-GBS-Fol... gbs_course-s... Fr, 08. Nov, 08:47

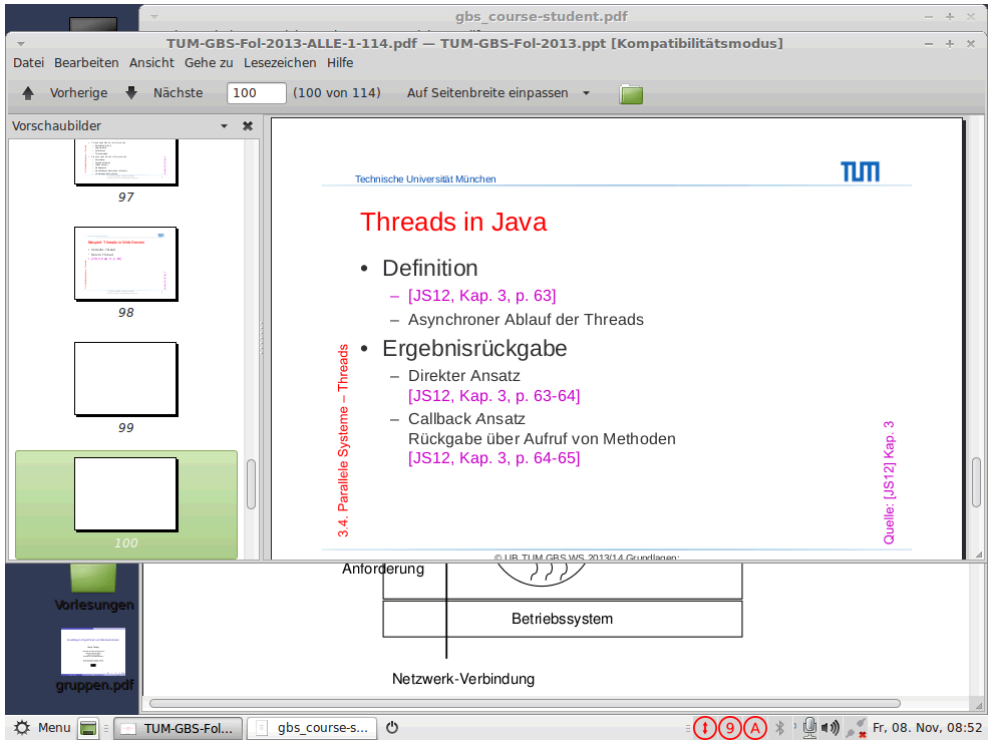
gbs_course-student.pdf
Datei Bearbeiten Ansicht Gehe zu Lesezeichen Hilfe
Vorherige Nächste 60 (67 von 228) 150%
Vorschaubilder
95
96
97
98
Vorlesungen
gruppen.pdf
Schlichter, TU München 3.4. THREAD-KONZEPT
jeder Prozess genau einen Thread, der den Kontrollfluss des Prozess repräsentiert, während in Multithreaded-Umgebungen ein Prozess mehrere Threads besitzen kann.
Prozess
Thread
Thread
Benutzer Adressraum
Betriebssystem
3.4.1 Charakterisierung von Threads
Aus BS-Sicht ist ein Prozess definiert
Menu TUM-GBS-Fol... gbs_course-s... Fr, 08. Nov, 08:47

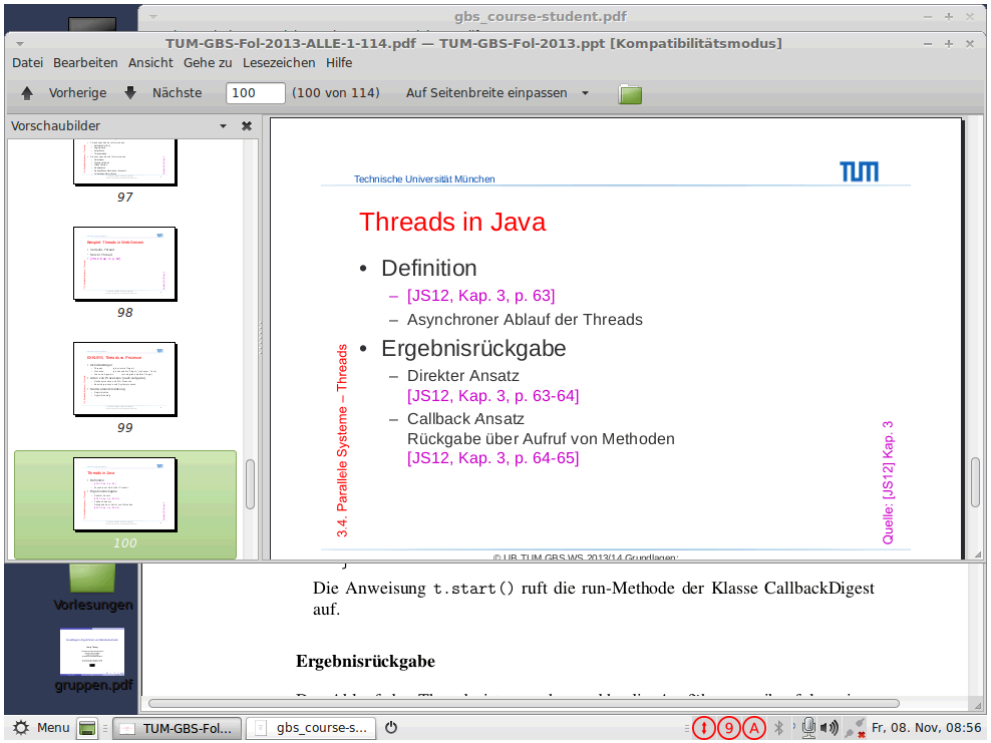
EXKURS: Threads vs. Prozesse

- Aktivitätsträger
  - Threads (als primäre Träger)
  - Prozesse (als sekundäre Träger) (synonym: Task)
  - Jobs und Agenten (als mögliche weitere Träger)
- Arten von Prozessen (nach Aufgabe)
  - Rechenprozesse und E/A-Prozesse
  - Benutzerprozesse und Systemprozesse
- Hardwareunterstützung
  - Registersätze
  - Hyperthreading

Threads in Java

- Definition
  - [JS12, Kap. 3, p. 63]
  - Asynchroner Ablauf der Threads
- Ergebnisrückgabe
  - Direkter Ansatz
    [JS12, Kap. 3, p. 63-64]
  - Callback Ansatz
    Rückgabe über Aufruf von Methoden
    [JS12, Kap. 3, p. 64-65]





Technische Universität München

Threads in Java

- Definition
  - [JS12, Kap. 3, p. 63]
  - Asynchroner Ablauf der Threads
- Ergebnisrückgabe
  - Direkter Ansatz
    [JS12, Kap. 3, p. 63-64]
  - Callback Ansatz
    Rückgabe über Aufruf von Methoden
    [JS12, Kap. 3, p. 64-65]

3.4. Parallele Systeme – Threads

Quelle: [JS12] Kap. 3

© UB TUM GBS WS 2013/14 Grundlagen:
Betriebssysteme und Systemsoftware (IN0009)

100

Synchronisation

- Aufgabe der Synchronisation
  - Abstimmung des zeitlichen Verhaltens nebenläufiger Aktivitäten untereinander
- Beispiele
  - Einfache nebenläufige Prozesse
    - Nichtdeterministisch
    - [JS12, Kap. 3, p. 66]
    - Lösung durch Verfahren zum wechselseitigen Ausschluss
  - Erzeuger-Verbraucher-Problem mit WA
    - [JS12, Kap. 3, p. 53]
- Synchronisation in Java (3.5.6)

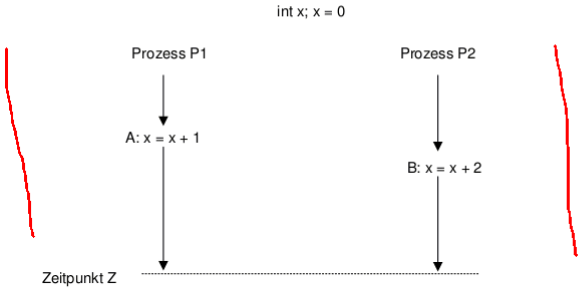
gbs_course-student.pdf
TUM-GBS-Fol-2013-ALLE-1-114.pdf — TUM-GBS-Fol-2013.ppt [Kompatibilitätsmodus]
Datei Bearbeiten Ansicht Gehe zu Lesezeichen Hilfe
Vorherige Nächste 101 (101 von 114) Auf Seitenbreite einpassen
Vorschaubilder
98 99 100 101
Technische Universität München
Synchronisation
- Aufgabe der Synchronisation
  - Abstimmung des zeitlichen Verhaltens nebenläufiger Aktivitäten untereinander
- Beispiele
  - Einfache nebenläufige Prozesse
    - Nichtdeterministisch
    - [JS12, Kap. 3, p. 66]
    - Lösung durch Verfahren zum wechselseitigen Ausschluss
  - Erzeuger-Verbraucher-Problem mit WA
    - [JS12, Kap. 3, p. 53]
- Synchronisation in Java (3.5.6)
3.5. Parallele Systeme – Synchronisation
Quelle: [JS12] Kap. 3
Vorlesungen
gruppen.pdf

Die Anweisung `t.start()` ruft die run-Methode der Klasse CallbackDigest auf.

Ergebnisrückgabe

Menu TUM-GBS-Fol... gbs_course-s... Fr, 08. Nov, 08:58

gbs_course-student.pdf
Datei Bearbeiten Ansicht Gehe zu Lesezeichen Hilfe
Vorherige Nächste 66 (73 von 228) 150%
Vorschaubilder
98 99 100 101
Vorlesungen
gruppen.pdf



- Das Ergebnis ist vom zeitlichen Ablauf abhängig. Es sind folgende Fälle möglich:
  - **Fall 1**
    P1 liest x = 0, erhöht, speichert x = 1;
    P2 liest x = 1, erhöht, speichert x = 3; => Wert von x = 3
  - **Fall 2**
    P2 liest x = 0, erhöht, speichert x = 2;
    P1 liest x = 2, erhöht, speichert x = 3; => Wert von x = 3
  - **Fall 3**

Menu TUM-GBS-Fol... gbs_course-s... Fr, 08. Nov, 08:59

gbs_course-student.pdf
Datei Bearbeiten Ansicht Gehe zu Lesezeichen Hilfe
Vorherige Nächste 66 (73 von 228) 150%
Vorschaubilder
98 99 100 101
Vorlesungen
gruppen.pdf

- Das Ergebnis ist vom zeitlichen Ablauf abhängig. Es sind folgende Fälle möglich:
  - **Fall 1**
    P1 liest x = 0, erhöht, speichert x = 1;
    P2 liest x = 1, erhöht, speichert x = 3; => Wert von x = 3
  - **Fall 2**
    P2 liest x = 0, erhöht, speichert x = 2;
    P1 liest x = 2, erhöht, speichert x = 3; => Wert von x = 3
  - **Fall 3**
    P1 und P2 lesen x = 0;
    P1 erhöht, speichert x = 1;
    P2 erhöht, speichert x = 2; => Wert von x = 2
  - **Fall 4**
    P1 und P2 lesen x = 0;
    P2 erhöht, speichert x = 2;
    P1 erhöht, speichert x = 1; => Wert von x = 1
- Verhinderung des Nichtdeterminismus nur dadurch möglich, dass man

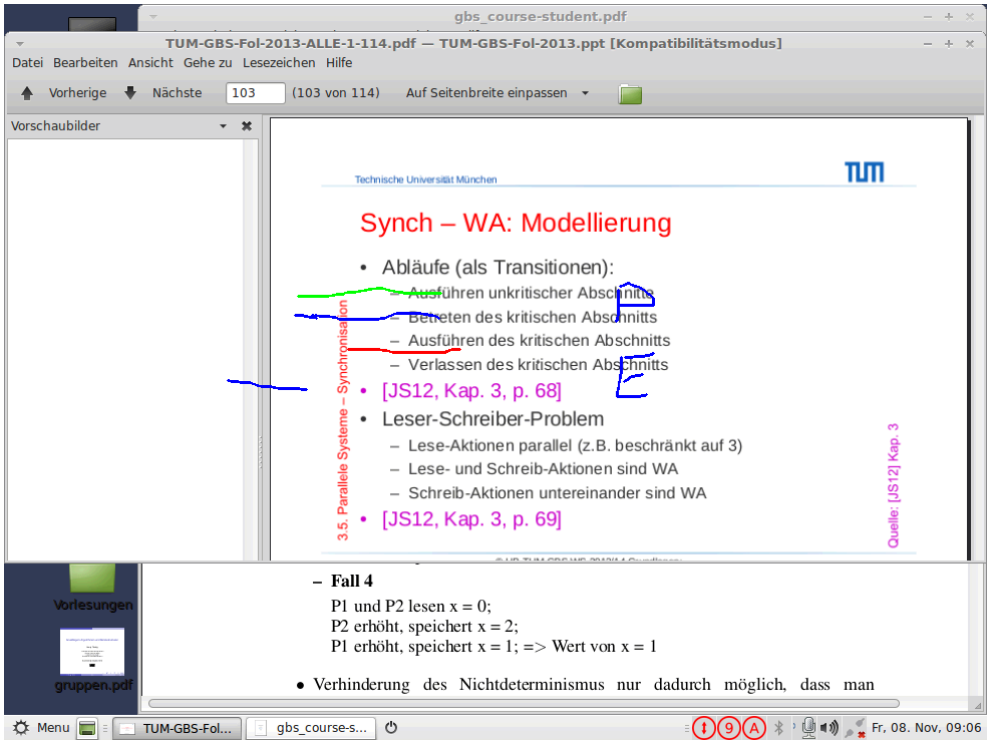
Menu TUM-GBS-Fol... gbs_course-s... Fr, 08. Nov, 08:59

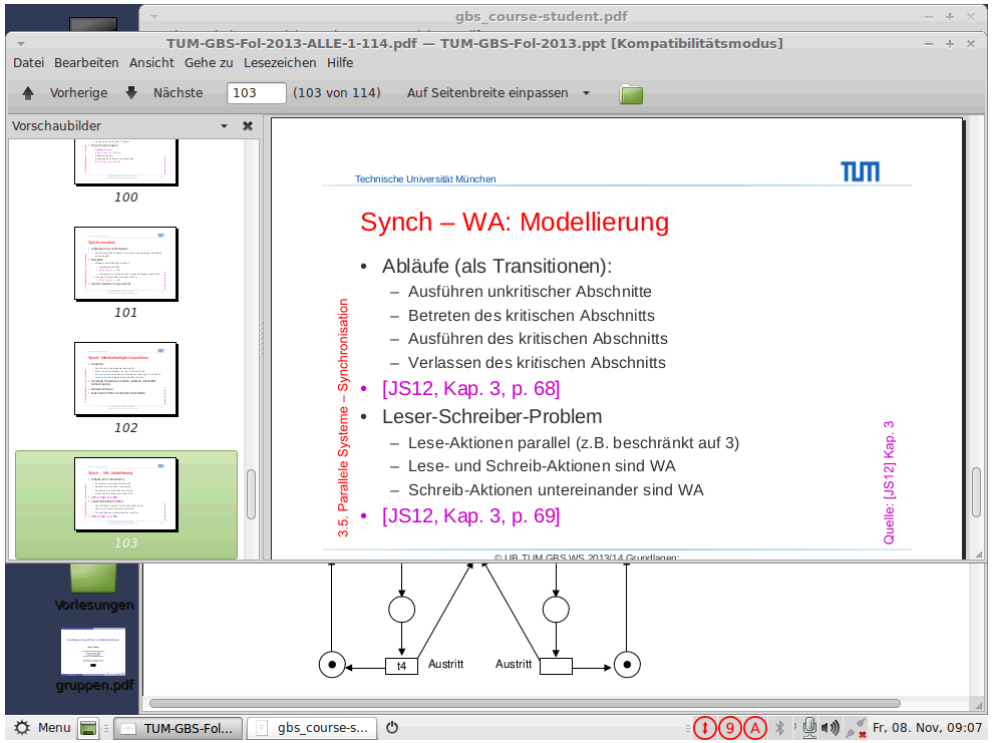
Synch: Wechselseitiger Ausschluss

- Gegeben:
  - Petri-Netz mit Anfangsmarkierung M0
  - Wenn zwei Transitionen t1 und t2 wechselseitig ausgeschlossen sind, dann ist keine Markierung M‘ erreichbar, so dass t1 und t2 gleichzeitig schaltbereit sind.
- Derartige Transitionen heißen „kritische Abschnitte“ (critical region).
- Mutual exclusion
- Nutzung der XBM in kritischen Abschnitten

Synch – WA: Modellierung

- Abläufe (als Transitionen):
  - Ausführen unkritischer Abschnitte
  - Betreten des kritischen Abschnitts
  - Ausführen des kritischen Abschnitts
  - Verlassen des kritischen Abschnitts
- [JS12, Kap. 3, p. 68]
- Leser-Schreiber-Problem
  - Lese-Aktionen parallel (z.B. beschränkt auf 3)
  - Lese- und Schreib-Aktionen sind WA
  - Schreib-Aktionen untereinander sind WA
- [JS12, Kap. 3, p. 69]





gbs_course-student.pdf

Datei Bearbeiten Ansicht Gehe zu Lesezeichen Hilfe

Vorherige Nächste 68 (75 von 228) 150%

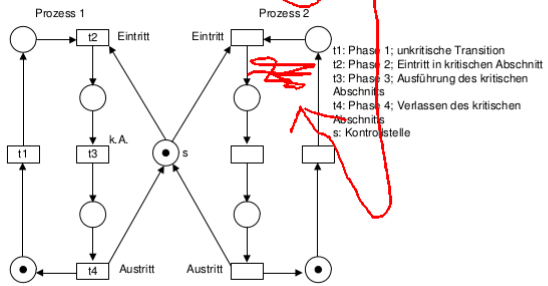
Vorschaubilder

3.5.3 Modellierung

Modelliert man parallele Einheiten, die kritische Abschnitte besitzen, durch Petri-Netze, so sind vier Phasen dieser parallelen Aktivitäten von Interesse:

1. Ausführen unkritischer Abschnitte/Transaktionen
2. Betreten eines kritischen Abschnitts
3. Ausführen der Transaktion(en) des kritischen Abschnitts
4. Verlassen des kritischen Abschnitts.

Modellierung jeder Phase durch eine Transition; Koordinierung des wechselseitigen Ausschluss durch Kontrollstelle s.



Menu TUM-GBS-Fol... gbs_course-s... Fr, 08. Nov, 09:08

Technische Universität München

Synchronisationskonzepte

- Konkretisierung von Prozessen
  - Ein Prozess ist ein eindeutig identifizierbarer Ablauf eines Programms.
- Zustände von Prozessen (ggf. auch Threads)
  - ERZEUGT
  - RECHNEND
  - RECHENWILLIG
  - WARTEND
  - TERMINIERT
- [JS12, Kap. 3, p. 70]
  - Alternative: „start“ überführt in RECHENWILLIG

3.5. Parallele Systeme – Synchronisation

Quelle: [JS12] Kap. 3

gbs_course-student.pdf

TUM-GBS-Fol-2013-ALLE-1-114.pdf — TUM-GBS-Fol-2013.ppt [Kompatibilitätsmodus]

Datei Bearbeiten Ansicht Gehe zu Lesezeichen Hilfe

Vorherige Nächste 104 (104 von 114) Auf Seitenbreite einpassen

Vorschaubilder

Technische Universität München

Synchronisationskonzepte

- Konkretisierung von Prozessen
  - Ein Prozess ist ein eindeutig identifizierbarer Ablauf eines Programms.
- Zustände von Prozessen (ggf. auch Threads)
  - ERZEUGT
  - RECHNEND
  - RECHENWILLIG
  - WARTEND
  - TERMINIERT
- [JS12, Kap. 3, p. 70]
  - Alternative: „start“ überführt in RECHENWILLIG

3.5. Parallele Systeme – Synchronisation

Quelle: [JS12] Kap. 3

Austritt Austritt

Menu TUM-GBS-Fol... gbs_course-s... Fr, 08. Nov, 09:14

Technische Universität München

Synchronisationskonzepte

- Konkretisierung von Prozessen
  - Ein Prozess ist ein eindeutig identifizierbarer Ablauf eines Programms.
- Zustände von Prozessen (ggf. auch Threads)
  - ERZEUGT
  - RECHNEND
  - RECHENWILLIG
  - WARTEND
  - TERMINIERT
- [JS12, Kap. 3, p. 70]
  - Alternative: „start“ überführt in RECHENWILLIG

3.5. Parallele Systeme – Synchronisation

Quelle: [JS12] Kap. 3

© UB TUM GBS WS 2013/14 Grundlagen: Betriebssysteme und Systemsoftware (IN0009) 104

Anforderungen an Lösungen des WA

- Die kritischen Abschnitte sind wechselseitig ausgeschlossen.
- Die Realisierung darf nicht von der Reihenfolge der Ausführung der kritischen Abschnitte abhängen.
- Die Realisierung darf nicht von Annahmen über die Ausführungszeit der Prozesse abhängen.
- Kein Prozess darf unendlich lange daran gehindert werden, seinen kritischen Abschnitt auszuführen.

- Grundlage für Lösungen:
  - Atomare (nicht teilbare) Operationen

Lösungen des WA

- Unterbrechungssperren
  - Disable Interrupt in Ein-Prozessorsystemen
- Test-and-Set Operationen
  - Atomarer Austausch von Werten einer Speicherzelle
- Dienste mit passivem Warten
  - Bei Bedarf Überführen in Zustand „WARTEND“
  - Beim Wecken Überführen in Zustand „RECHENBEREIT“
  - Alternative: aktives Warten
- Semaphor-Konzept
  - „Signal“ für FREI und BELEGT
- Monitor-Konzept
  - Objektorientiert mit WA-Methoden

Technische Universität München

Anforderungen an Lösungen des WA

- Die kritischen Abschnitte sind wechselseitig ausgeschlossen.
- Die Realisierung darf nicht von der Reihenfolge der Ausführung der kritischen Abschnitte abhängen.
- Die Realisierung darf nicht von Annahmen über die Ausführungszeit der Prozesse abhängen.
- Kein Prozess darf unendlich lange daran gehindert werden, seinen kritischen Abschnitt auszuführen.

- Grundlage für Lösungen:
  - Atomare (nicht teilbare) Operationen

3.5. Parallele Systeme – Synchronisation

Quelle: [JS12] Kap. 3

© UB TUM GBS WS 2013/14 Grundlagen: Betriebssysteme und Systemsoftware (IN0009)
105

Technische Universität München

Lösungen des WA

- Unterbrechungssperren
  - Disable Interrupt in Ein-Prozessorsystemen
- Test-and-Set Operationen
  - Atomarer Austausch von Werten einer Speicherzelle
- Dienste mit passivem Warten
  - Bei Bedarf Überführen in Zustand „WARTEND“
  - Beim Wecken Überführen in Zustand „RECHENBEREIT“
  - Alternative: aktives Warten
- Semaphor-Konzept
  - „Signal“ für FREI und BELEGT
- Monitor-Konzept
  - Objektorientiert mit WA-Methoden

3.5. Parallele Systeme – Synchronisation

Quelle: [JS12] Kap. 3

© UB TUM GBS WS 2013/14 Grundlagen: Betriebssysteme und Systemsoftware (IN0009)
106

Synchronisationskonzepte

- Konkretisierung von Prozessen
  - Ein Prozess ist ein eindeutig identifizierbarer Ablauf eines Programms.
- Zustände von Prozessen (ggf. auch Threads)
  - ERZEUGT
  - RECHNEND
  - RECHENWILLIG
  - WARTEND
  - TERMINIERT
- [JS12, Kap. 3, p. 70]
  - Alternative: „start“ überführt in RECHENWILLIG

Lösungen des WA

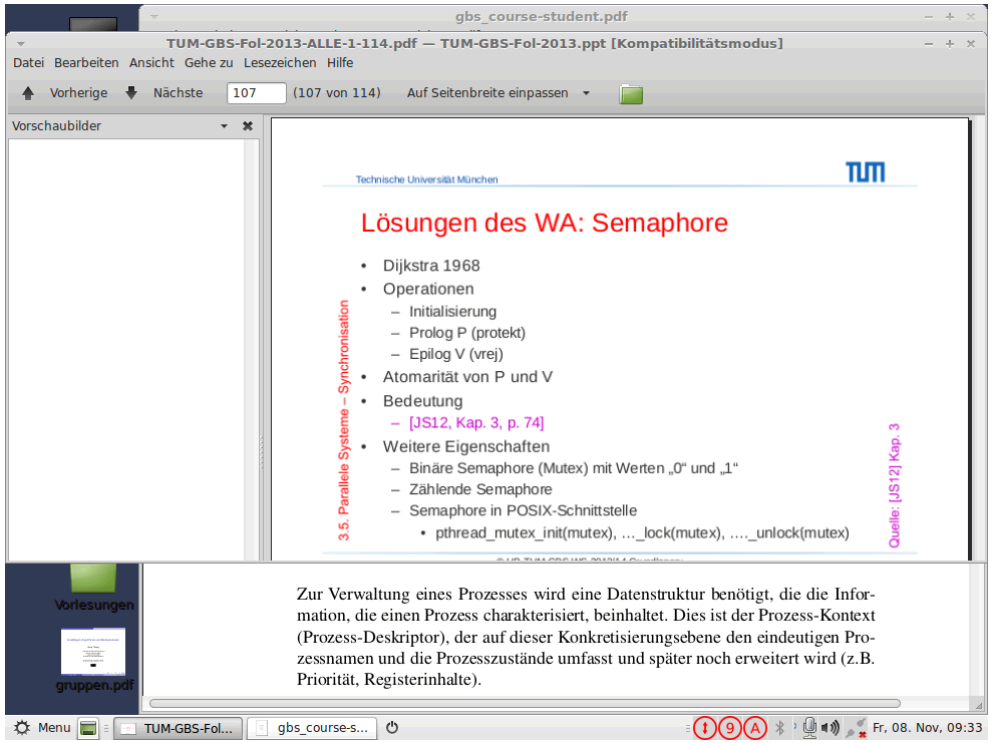
- Unterbrechungssperren
  - Disable Interrupt in Ein-Prozessorsystemen
- Test-and-Set Operationen
  - Atomarer Austausch von Werten einer Speicherzelle
- Dienste mit passivem Warten
  - Bei Bedarf Überführen in Zustand „WARTEND“
  - Beim Wecken Überführen in Zustand „RECHENBEREIT“
  - Alternative: aktives Warten
- Semaphor-Konzept
  - „Signal“ für FREI und BELEGT
- Monitor-Konzept
  - Objektorientiert mit WA-Methoden

Mono

Multi

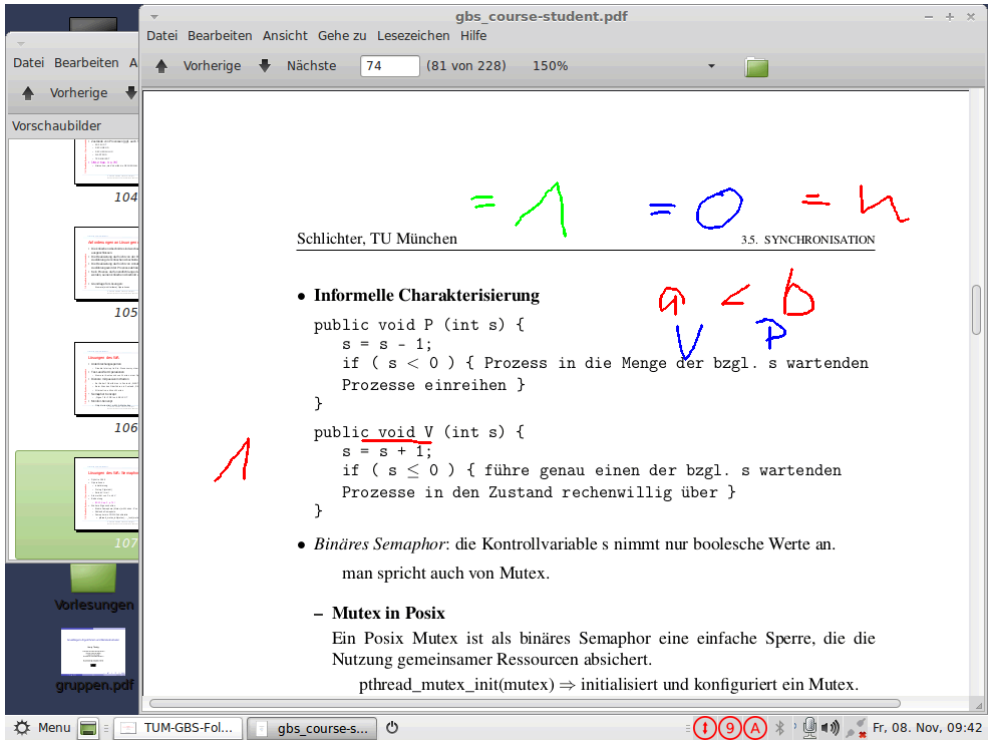
Lösungen des WA: Semaphore

- Dijkstra 1968
- Operationen
  - Initialisierung
  - Prolog P (protekt)
  - Epilog V (vrej)
- Atomarität von P und V
- Bedeutung
  - [JS12, Kap. 3, p. 74]
- Weitere Eigenschaften
  - Binäre Semaphore (Mutex) mit Werten „0“ und „1“
  - Zählende Semaphore
  - Semaphore in POSIX-Schnittstelle
    - pthread_mutex_init(mutex), …_lock(mutex), ….._unlock(mutex)



Lösungen des WA: Semaphore

- Dijkstra 1968
- Operationen
  - Initialisierung
  - Prolog P (protekt)
  - Epilog V (vrej)
- Atomarität von P und V
- Bedeutung
  - [JS12, Kap. 3, p. 74]
- Weitere Eigenschaften
  - Binäre Semaphore (Mutex) mit Werten „0“ und „1“
  - Zählende Semaphore
  - Semaphore in POSIX-Schnittstelle
    - pthread_mutex_init(mutex), …_lock(mutex), ….._unlock(mutex)



Technische Universität München

Lösungen des WA: Semaphore

- Dijkstra 1968
- Operationen
  - Initialisierung
  - Prolog P (protekt)
  - Epilog V (vrej)
- Atomarität von P und V
- Bedeutung
  - [JS12, Kap. 3, p. 74]
- Weitere Eigenschaften
  - Binäre Semaphore (Mutex) mit Werten „0“ und „1“
  - Zählende Semaphore
  - Semaphore in POSIX-Schnittstelle
    - pthread_mutex_init(mutex), …_lock(mutex), ….._unlock(mutex)

3.5. Parallele Systeme – Synchronisation

Quelle: [JS12] Kap. 3

© UB TUM GBS WS 2013/14 Grundlagen:
Betriebssysteme und Systemsoftware (IN0009)
107

Technische Universität München

Lösungen des WA: Semaphor-Beispiele

- Kritische Abschnitte
  - [JS12, Kap. 3, p. 75]
- Erzeuger-Verbraucher-Problem mit beschränktem Puffer
  - Variante 1 nur mit Wechselseitigem Ausschluss
  - [JS12, Kap. 3, p. 75]
  - Variante 2 zuzüglich der Absicherung von LEER durch „voll“
  - [JS12, Kap. 3, p. 76]
  - Variante 3 zuzüglich der Absicherung von VOLL durch „leer“
  - [JS12, Kap. 3, p. 76]

3.5. Parallele Systeme – Synchronisation

Quelle: [JS12] Kap. 3

gbs_course-student.pdf

TUM-GBS-Fol-2013-ALLE-1-114.pdf — TUM-GBS-Fol-2013.ppt [Kompatibilitätsmodus]

Datei Bearbeiten Ansicht Gehe zu Lesezeichen Hilfe

Vorherige Nächste 108 (108 von 114) Auf Seitenbreite einpassen

Vorschaubilder

Technische Universität München

Lösungen des WA: Semaphor-Beispiele

- Kritische Abschnitte
  - [JS12, Kap. 3, p. 75]
- Erzeuger-Verbraucher-Problem mit beschränktem Puffer
  - Variante 1 nur mit Wechselseitigem Ausschluss
  - [JS12, Kap. 3, p. 75]
  - Variante 2 zuzüglich der Absicherung von LEER durch „voll“
  - [JS12, Kap. 3, p. 76]
  - Variante 3 zuzüglich der Absicherung von VOLL durch „leer“
  - [JS12, Kap. 3, p. 76]

3.5. Parallele Systeme – Synchronisation

Quelle: [JS12] Kap. 3

man spricht auch von Mutex.

– **Mutex in Posix**

Ein Posix Mutex ist als binäres Semaphor eine einfache Sperre, die die Nutzung gemeinsamer Ressourcen absichert.

pthread_mutex_init(mutex) ⇒ initialisiert und konfiguriert ein Mutex.

Vorlesungen

gruppen.pdf

Menu TUM-GBS-Fol... gbs_course-s... Fr, 08. Nov, 09:49

gbs_course-student.pdf

Datei Bearbeiten Ansicht Gehe zu Lesezeichen Hilfe

Vorherige Nächste 76 (83 von 228) 150%

Vorschaubilder

105 106 107 108

Schlichter, TU München 3.5. SYNCHRONISATION

Problem: es kann eine Verklemmung auftreten, wenn der Verbraucher wa.P ausführt und warten muss, weil der Puffer kein Element enthält.

- **Variante 2**

Einführen eines zusätzlichen Semaphors voll: semaphor(0), das die Datenelemente im Puffer zählt:

```
Erzeuger E:
while (true) {
    produziere
    wa.P
    schreibe nach W
    wa.V
    voll.V
}
```

```
Verbraucher V:
while (true) {
    voll.P
    wa.P
    entnimm aus W
    wa.V
    verarbeite
}
```

Für den Erzeuger ergibt sich natürlich ein analoges Problem, falls der Puffer W nur eine beschränkte Kapazität besitzt.

Vorlesungen

gruppen.pdf

Menu TUM-GBS-Fol... gbs_course-s... Fr, 08. Nov, 09:51

gbs_course-student.pdf

Datei Bearbeiten Ansicht Gehe zu Lesezeichen Hilfe

Vorherige Nächste 76 (83 von 228) 150%

Vorschaubilder

105 106 107 108

Schlichter, TU München 3.5. SYNCHRONISATION

Problem: es kann eine Verklemmung auftreten, wenn der Verbraucher wa.P ausführt und warten muss, weil der Puffer kein Element enthält.

- **Variante 2**

Einführen eines zusätzlichen Semaphors voll: semaphor(0), das die Datenelemente im Puffer zählt:

```
Erzeuger E:
while (true) {
    produziere
    wa.P
    schreibe nach W
    wa.V
    voll.V
}
```

```
Verbraucher V:
while (true) {
    voll.P
    wa.P
    entnimm aus W
    wa.V
    verarbeite
}
```

LEER

Für den Erzeuger ergibt sich natürlich ein analoges Problem, falls der Puffer W nur eine beschränkte Kapazität besitzt.

- **Variante 3**

Einführen eines zusätzlichen Semaphors leer: semaphor(n), das die Anzahl der

Vorlesungen

gruppen.pdf

Menu TUM-GBS-Fol... gbs_course-s... Fr, 08. Nov, 09:51

gbs_course-student.pdf

Fenstermenü ten Ansicht Gehe zu Lesezeichen Hilfe

Vorherige Nächste 76 (83 von 228) 150%

Vorschaubilder

105 106 107 108

Schlichter, TU München 3.5. SYNCHRONISATION

Problem: es kann eine Verklemmung auftreten, wenn der Verbraucher wa.P ausführt und warten muss, weil der Puffer kein Element enthält.

- **Variante 2**

Einführen eines zusätzlichen Semaphors voll: semaphor(0), das die Datenelemente im Puffer zählt:

```
Erzeuger E:
while (true) {
    produziere
    wa.P
    schreibe nach W
    wa.V
    voll.V
}
```

```
Verbraucher V:
while (true) {
    voll.P
    wa.P
    entnimm aus W
    wa.V
    verarbeite
}
```

Für den Erzeuger ergibt sich natürlich ein analoges Problem, falls der Puffer W nur eine beschränkte Kapazität besitzt.

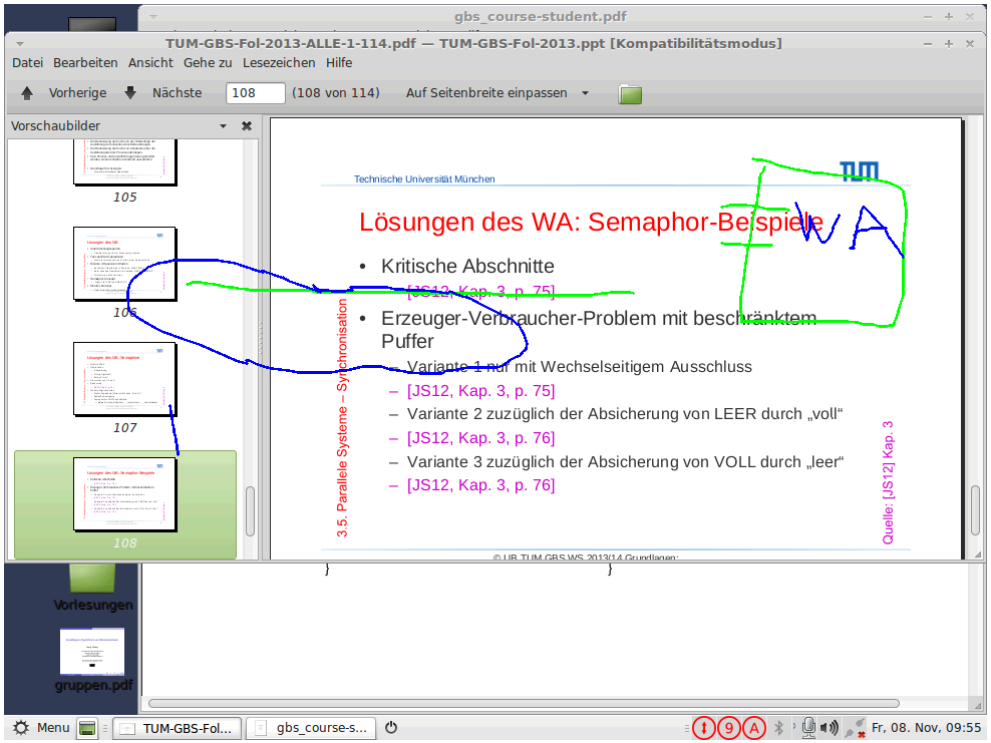
- **Variante 3**

Einführen eines zusätzlichen Semaphors leer: semaphor(n), das die Anzahl der freien Elemente im Puffer zählt:

Vorlesungen

gruppen.pdf

Menu TUM-GBS-Fol... gbs_course-s... Fr, 08. Nov, 09:53



Technische Universität München

Verklemmungen

- Definition
  - Eine Verklemmung ist ein Zustand, in dem die beteiligten Prozesse wechselseitig auf den Eintritt von Bedingungen warten, die nur durch andere Prozesse in dieser Gruppe selbst hergestellt werden können.
- Notwendige Bedingungen
  - BM sind exklusiv (XBM)
  - BM sind nicht entziehbar (DBM)
  - „Haben-und-Anfordern“
  - Zyklische Wartebedingung

3.6. Parallele Systeme – Verklemmungen

Quelle: [JS12] Kap. 3