Allgemeines

Fächer:	Betriebssysteme, Verteilte Systeme
Prüfer:	Dr. Jürgen Kleinöder
Beisitzer:	Julio Sincero
Note:	1,0

- Ich durfte entscheiden, mit welchem Fach ich anfangen wollte. Ich habe BS gewählt, was von den 30 Minuten schon mal 18 eingenommen hat.
- Jürgen prüft in erster Linie kein detailliertes Lernwissen ab, sondern stellt sehr detaillierte Verständnisfragen. Wenn man da erst mal stutzt, hilft er einem aber auch auf die Sprünge.
- Gerade für den BS-Teil war das Bearbeiten der Übungsaufgaben enorm hilfreich.
- Anhand der alten Prüfungsprotokolle lässt sich sehr gut herausfinden, welche Themengebiete am relevantesten sind.

Fragen

Gegeben sind zwei Prozesse. Der erste berechnet π und wurde verdrängt, der zweite ist aktiv und will ein Zeichen von der Tastatur einlesen. Welche Vorgänge spielen sich innerhalb des Betriebssystems ab?

- Betriebssystem verwaltet intern einen Zeichenpuffer, gesichert mit einem Semaphor
- Lesendes Programm macht P, Interrupt-Handler (bzw. Epilog) macht pro Tastendruck ein V

Wie ist der Semaphor implementiert?

- Alle Prozesse, die im P blockieren, warten passiv und werden in einer Warteschlange verwaltet
- V verschiebt das erste Element der Warteschlange wieder in die „Bereit“-Liste des Schedulers
- P und V dürfen sich nicht überlappen:
  - Interrupts sperren → Zustand des Semaphors liegt auf Unterbrechungsebene
  - P und V auf Epilogebene ausführen (bessere Wahl)
  - Synchronisation mit Mutex geht **nicht**, weil V nicht blockieren darf (*run-to-completion* muss sicher gestellt sein)

Was passiert bei einem Tastendruck?

- Interrupt
- Zustandssicherung, Behandlungsroutine wird angesprungen und fordert einen Epilog an
- Epilog: Zeichen werden ausgelesen und im Puffer abgelegt, pro Zeichen ein V
- Scheduler wird aufgerufen

Wer sorgt für die Zustandssicherung?

- Programmzähler, Statusregister: automatisch durch die CPU
- Flüchtige Register: Kopplungsroutine in Assembler
- Nicht-flüchtige Register: Compiler

Wie funktioniert ein Koroutinenwechsel?

- Register auf dem Stack sichern
- Stack-Zeiger umbiegen
- Register vom Stack wiederherstellen
- Rücksprung an die Stelle, an der die Koroutine vorher verdrängt wurde (oder freiwillig `resume()` aufgerufen hat)

Welche grundsätzlichen Scheduling-Verfahren gibt es?

Je nach Betriebsart:

- Stapelbetrieb (rechenintensiv, wenig E/A)
- Interaktiver Betrieb (E/A-lastig): probabilistisch, online, präemptiv
- Echtzeitbetrieb (harte Zeitlimits): deterministisch, offline, kooperativ

Was ist ein verteiltes System?

- Mehrere Rechner
- Netzwerk
- Kooperation

Wo liegen die Herausforderungen bei verteilten Systemen?

- Heterogenität
- Ziel: möglichst hohe Transparenz
- Fehler, die im lokalen Fall nicht auftreten können (Verbindungsabbrüche, Rechnerausfälle)
- Nebenläufigkeit

Wie funktioniert ein Fernaufruf?

Das altbekannte Bild mit Stub & Skeleton, das man aus dem Effeff können sollte

Welche Probleme gibt es bei der Übergabe von Zeigern als Parameter?

- Wofür steht der Zeiger: Referenz auf Ein- oder Ausgabeparameter? Array?
- Lösung: genaue Spezifikation in IDL oder durch „Smart Pointer“

Wie versucht man Fehlertransparenz herzustellen?

Anfragewiederholung, Duplikatselimination
→ Fernaufrufsemantiken:

- Exactly-once (lokale Semantik, im verteilten Fall nicht erreichbar)
- Maybe (keinerlei Fehlertransparenz)
- At-least-once (Anfragewiederholung, nur idempotente Operationen)

Nicht idempotente Operation: Beispiel? Abhilfe?

Geldautomat:

- Geldbetrag vom Kontostand abziehen
- Alten Kontostand abfragen, Subtraktion vor Ort erledigen, neuen Kontostand setzen

Beispiel für ein System, das (fast) ausschließlich auf idempotenten Operationen basiert?

NFS (ist zustandslos)

Welche Operationen sind in NFS nicht idempotent realisierbar?

- Datei im „Append“-Modus öffnen
- Locking