HOCHSCHULE FURTWANGEN UNIVERSITY | HFU

Blatt 7

Betriebssysteme

Luis Staudt

Aufgabe b)

I) Beide Reisebüros verkaufen Ticket 3

1. Ausgangszustand: `VerfuegbareTickets = 4`
2. Thread Reiseland: ruft texttttTicketsVerfuegbar() auf → `return true` ($4 > 0$)
3. Thread Happy Travel: ruft `TicketsVerfuegbar()` auf → `return true` ($4 > 0$)
4. Thread Reiseland: ruft `TicketVerkauf()` auf
   - `nr = VerfuegbareTickets` → `nr = 4`
   - Context Switch zu Happy Travel
5. Thread Happy Travel: ruft `TicketVerkauf()` auf
   - `nr = VerfuegbareTickets` → `nr = 4` (noch unverändert!)
   - `VerfuegbareTickets = 3`
   - `return 4`
6. Thread Reiseland: setzt Ausführung fort
   - `VerfuegbareTickets = 3` (überschreibt den Wert von Happy Travel)
   - `return 4`
7. Beide Threads geben aus: "verkauft Ticket 4"
8. `VerfuegbareTickets` ist jetzt 3, aber beide haben Ticket 4 verkauft

II) Verkauf von Ticket 0 trotz synchronized

1. Ausgangszustand: `VerfuegbareTickets = 1`
2. Thread Reiseland: ruft `synchronized TicketsVerfuegbar()` auf
   - Erhält Monitor
   - `return true` ($1 > 0$)
   - Gibt Monitor frei
3. Thread Happy Travel: ruft `synchronized TicketsVerfuegbar()` auf
   - Erhält Monitor
   - `return true` ($1 > 0$)
   - Gibt Monitor frei
4. Thread Reiseland: ruft `synchronized TicketVerkauf()` auf

- Erhält Monitor
- `nr = 1`
- `VerfuegbareTickets = 0`
- `return 1`
- Gibt Monitor frei

5. Thread Happy Travel: ruft `synchronized TicketVerkauf()` auf
   - Erhält Monitor
   - `nr = 0` (aktueller Wert!)
   - `VerfuegbareTickets = -1`
   - `return 0`
   - Gibt Monitor frei
6. Ausgabe: “ Happy Travel verkauft Ticket 0“

Problem: Die Prüfung (TicketsVerfuegbar()) und die Aktion (TicketVerkauf()) sind zwar einzeln synchronisiert, aber nicht als atomare Einheit. Zwischen der Prüfung und dem Verkauf kann ein anderer Thread dazwischenkommen.