Konzepte objektorientierter Programmierung

Prof. Dr. Peter Müller

Werner Dietl

Software Component Technology

Exercises 11: Concurrency and Distribution

Wintersemester 05/06

ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Solution

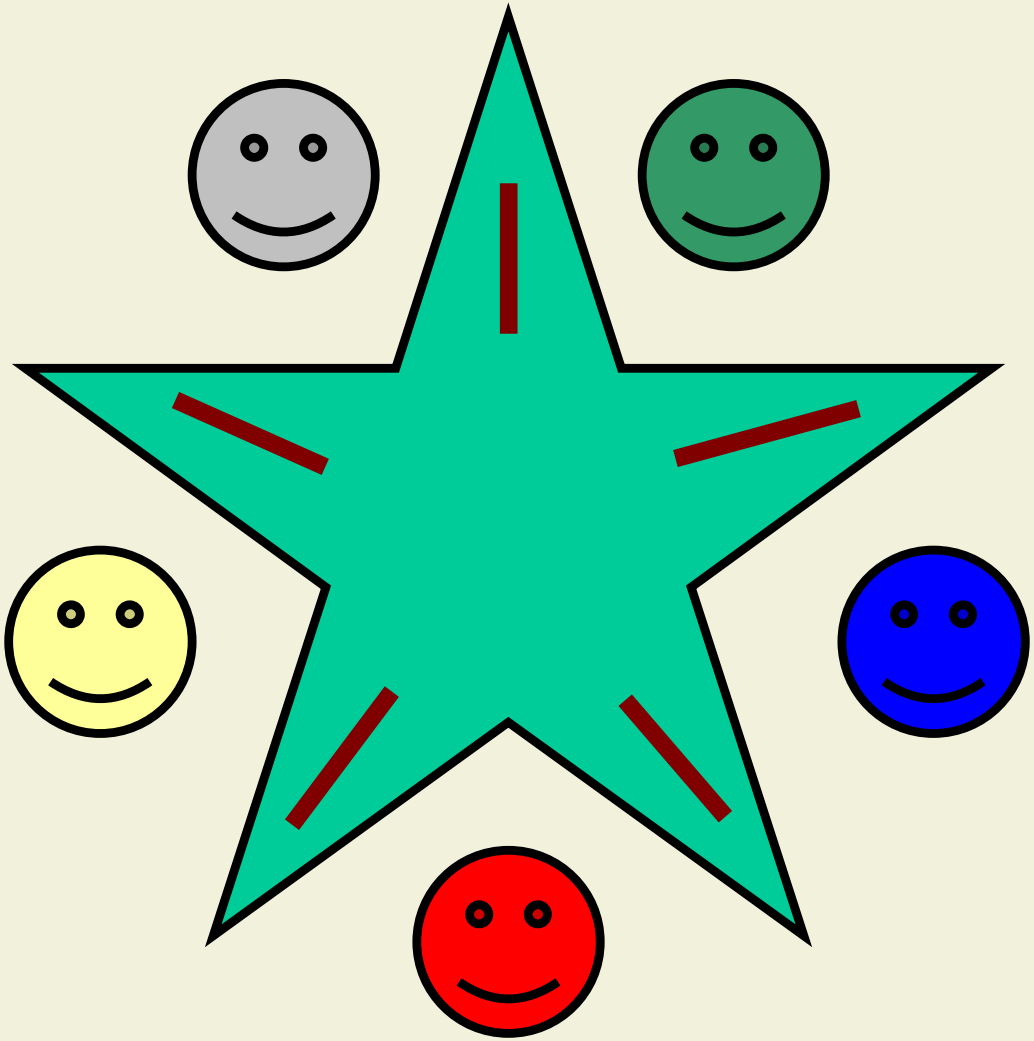
```
public synchronized void put( Prd p ) {
   while( isFull() ) {
      wait();
   }
   …
   notifyAll();
}
```

`notify` vs. `notifyAll`

- `java.lang.Object: final void notify()`
  Wakes up a single thread that is waiting on this object's monitor. If any threads are waiting on this object, one of them is chosen to be awakened. The choice is arbitrary and occurs at the discretion of the implementation. A thread waits on an object's monitor by calling one of the wait methods.
- `java.lang.Object: final void notifyAll()`
  Wakes up all threads that are waiting on this object's monitor. A thread waits on an object's monitor by calling one of the wait methods.

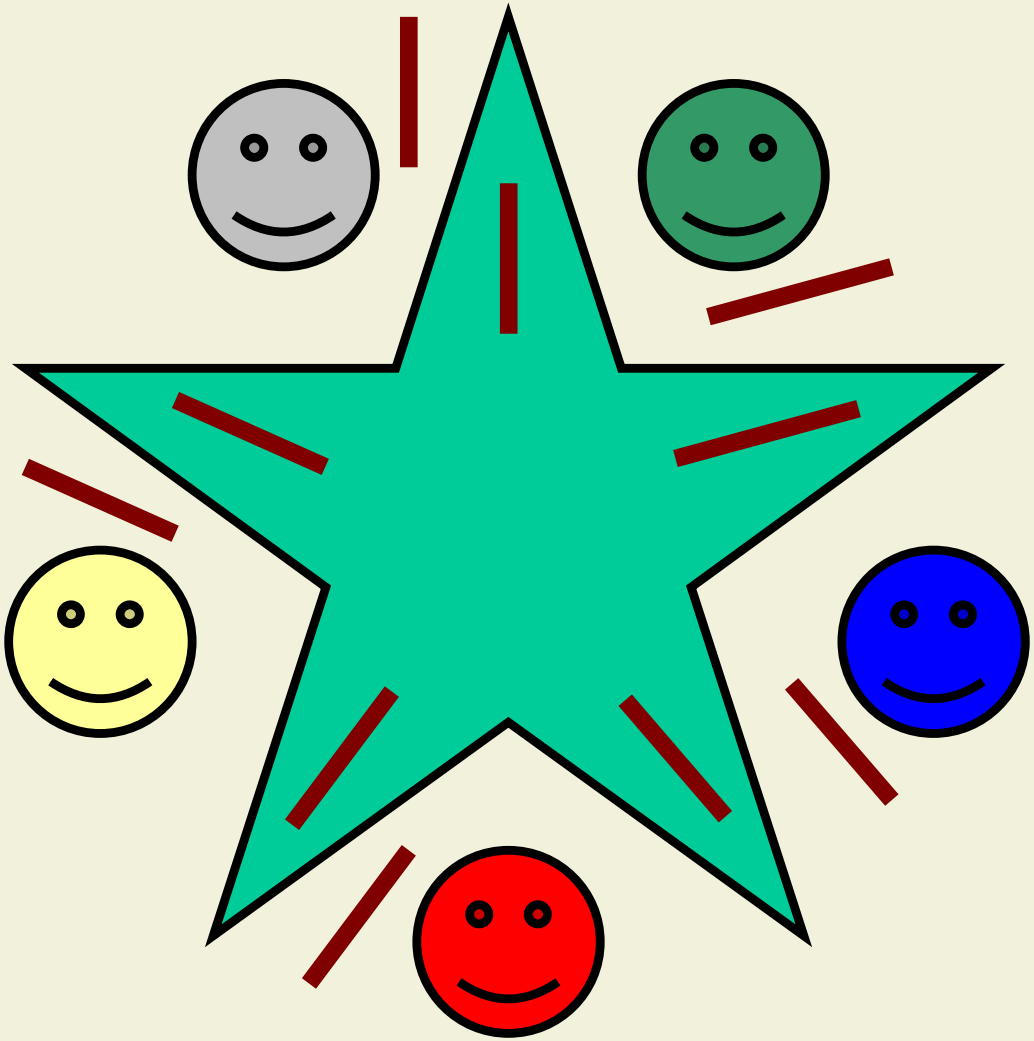
Dining Philosophers Problem

- n Philosophers
- n Chopsticks
- 2 Chopsticks needed for eating



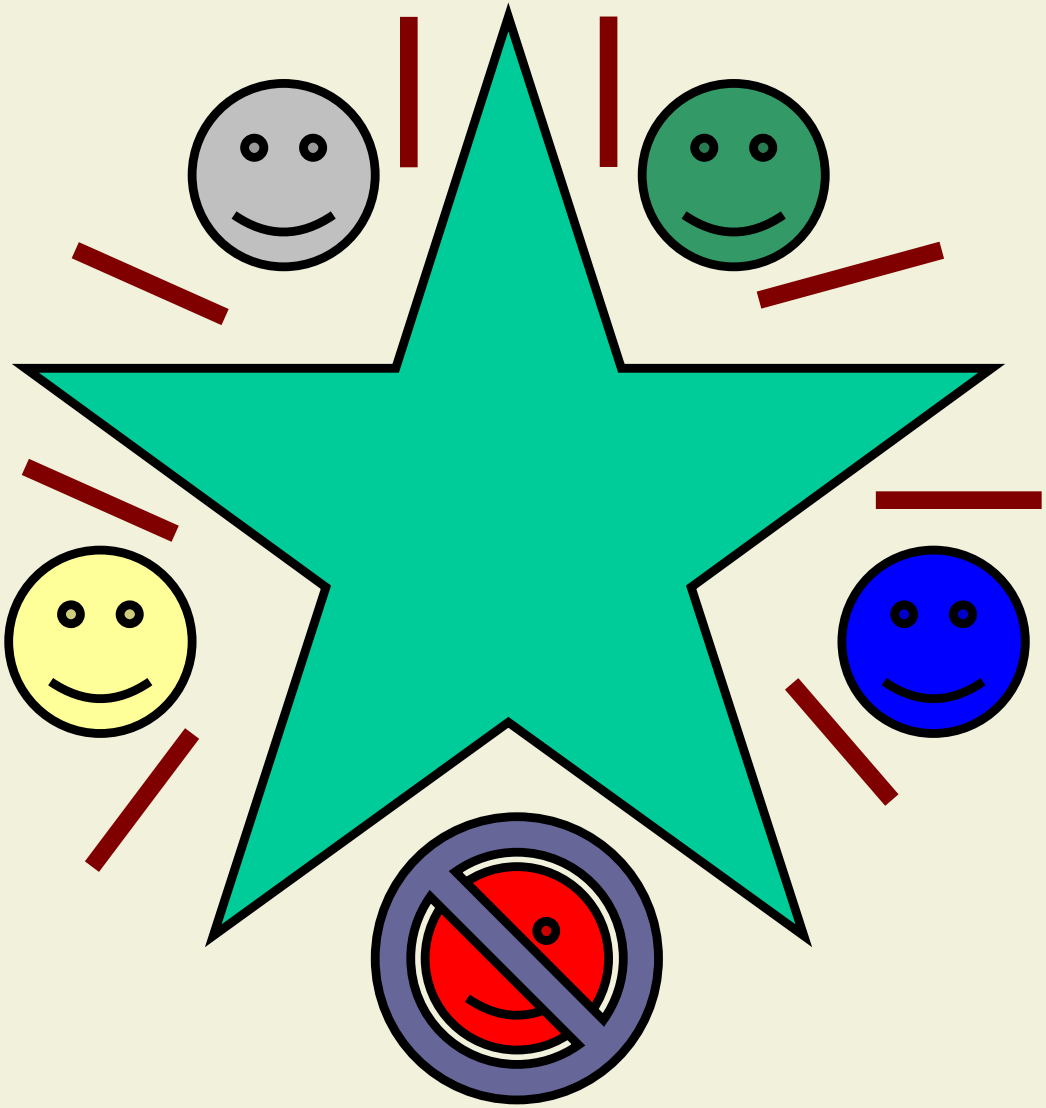
Problem: Deadlock

Everybody picks
the left chopstick
and then waits
forever to get
the right chopstick.

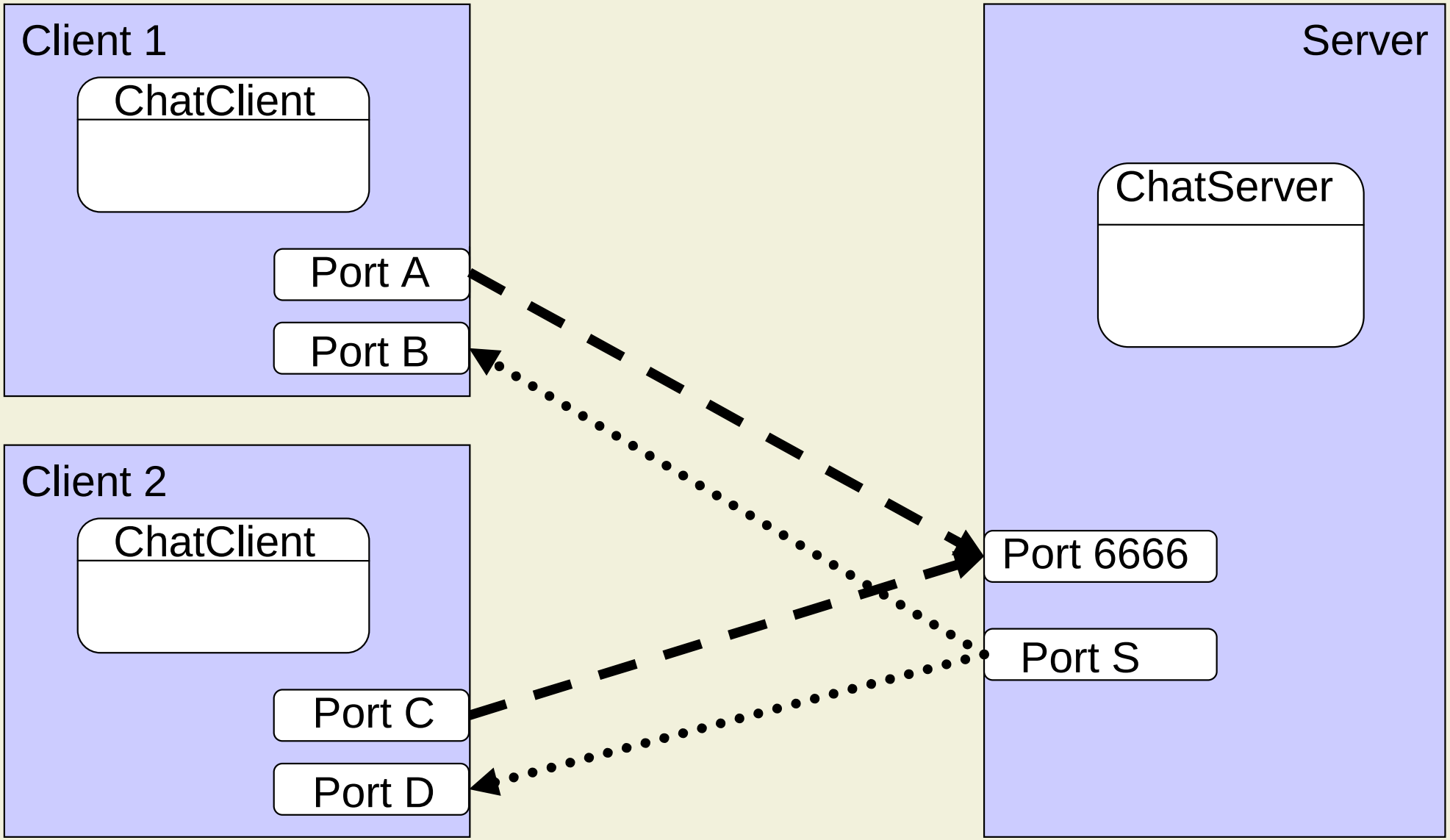


Problem: Starvation

Some philosophers Never get a chance to pick up both chopsticks and “starve”.



Homework 11 – Chat Application



Messages

- Register: Client → Server
  Client host and port
- Bcast: Client → Server
  Message
- Msg: Server → Client
  Message
- Deregister: Client → Server
  Client host and port

Which Threads do we need?

- Client:
  - Keyboard handler
  - Connections from server
- Server:
  - Multiple connections from clients

Questions?