Parallel Programming WS 2014/2015 - Solution 3

Kastens, Pfahler
Institut für Informatik, Fakultät für Elektrotechnik, Informatik und Mathematik, Universität Paderborn
Nov 24, 2014

Solution for Exercise 1

Monitor development in three main steps:

1. **Monitor state and entry procedures**

```
monitor onelanebridge
{
   # state
   int nb = 0;
   int sb = 0;

   #entry procedures
   entry enter_nb
        nb++;
   entry enter_sb
        sb++
   entry leave_nb
        nb--
   entry leave_sb
        sb--
}
```

2. **Monitor invariant and waiting**

```
monitor onelanebridge
{
MI: (nb == 0 AND sb >= 0) OR (sb == 0 AND nb >= 0)

   # state
   int nb = 0;
   int sb = 0;

   # condition variables
   condvar nbfreetogo, sbfreetogo, nbfreetoleave, sbfreetoleave;

   #entry procedures
   entry enter_nb
        while (sb != 0) wait(nbfreetogo);
        nb++;
   entry enter_sb
        while (nb != 0) wait(sbfreetogo);
        sb++
   entry leave_nb
        while (nb == 0) wait(nbfreetoleave)
        nb--
   entry leave_sb
        while (sb == 0) wait(sbfreetoleave)
        sb--
}
```

3. **Signalling, avoiding unnecessary signals**

```
monitor onelanebridge
{
MI: (nb == 0 AND sb >= 0) OR (sb == 0 AND nb >= 0)

   # state
   int nb = 0;
   int sb = 0;

   # condition variables
   condvar nbfreetogo, sbfreetogo, nbfreetoleave, sbfreetoleave;

   #entry procedures
   entry enter_nb
        while (sb != 0) wait(nbfreetogo);
        nb++;
        signal(nbfreetoleave)
   entry enter_sb
        while (nb != 0) wait(sbfreetogo);
        sb++
        signal(sbfreetoleave)
   entry leave_nb
        while (nb == 0) wait(nbfreetoleave)
        nb--
        if (nb == 0) signal sbfreetogo
   entry leave_sb
        while (sb == 0) wait(sbfreetoleave)
        sb--
        if (sb == 0) signal nbfreetogo
}
```

Solution for Exercise 3

The monitor implementation looks as follows;:

```
public class OneLane {
    int nb; // northbound cars on bridge
     int sb; // southbound cars on bridge

    synchronized public void enter_sb() {
        while (nb != 0) {
            try {
                wait();
            } catch (InterruptedException ie) {/* ok to ignore */}
        }
        sb++;
        notifyAll();
        System.out.println("sb enter: " + sb + " southbound cars on lane");
    }

    synchronized public void leave_sb() {
        while (sb == 0) {
            try {
                wait();
            } catch (InterruptedException ie) {/* ok to ignore */}
        }
        sb--;
        System.out.println("sb leave: " + sb + " southbound cars on lane");
        if (sb == 0) {
            notifyAll();
        }
    }

    synchronized public void enter_nb() {
        while (sb != 0) {
            try {
                wait();
            } catch (InterruptedException ie) {/* ok to ignore */}
        }
        nb++;
        notifyAll();
        System.out.println("nb enter: " + nb + " northbound cars on lane");
    }

    synchronized public void leave_nb() {
        while (nb == 0) {
            try {
                wait();
            } catch (InterruptedException ie) {/* ok to ignore */}
        }
        nb--;
        System.out.println("nb leave: " + nb + " northbound cars on lane");
        if (nb == 0) {
            notifyAll();
        }
    }
}
```

The complete One-lane bridge simulation can be found in `blatt3/oneLane`.