OS

Lecture 2

Today

- Jikes
- pattern: `InputStreamTOText Conversion Pattern`
- Anderson: synchronization
- Synchronization in Java
- HW 1 and throughts about HW 2

Synchronization in Java

- Monitors
- Separate core behavior from synchronization behavior

Basic behavior

```
public class GroundCounter {
  protected long count_;
  protected GroundCounter(long c){
    count_ = c;
  }
  protected long value_(){return count_;}
  protected void inc_() { ++ count_;}
  protected void dec_() { -- count_;}
}
```

Synchronization using Subclassing

```
public class BoundedCounterC extends GroundCounter
  implements BoundedCounter {
  public BoundedCounterC() { super(MIN); }
  public synchronized long value() { return value_();}
  public synchronized void inc() {
    while (value_() >= MAX)
      try {wait();} catch(InterruptedException ex){};
    inc_(); notifyAll();}
  public sychronized void dec() {
    while (value_() <= MIN)
      try {wait();} catch(InterruptedException ex) {};
    dec_(); notifyAll();}
  }
```

BoundedCounter interface

```
public interface BoundedCounter {
  public static final long MIN = 0;
  public static final long MAX = 10;
  public long value();// invariant:
     // MIN <= value() <= MAX
  public void inc(); // only when value()<MAX
  public void dec(); // only when value()>MIN
}
```

Advantage of separation

- Less tangling: separation of concerns.
- Can more easily use different synchronization policy; can use synchronization policy with other basic code.
- Avoids to mix variables used for synchronization with variables used for basic behavior.

Implementation rules

- For each condition that needs to be waited on, write a guarded wait loop.
- Ensure that every method causing state changes that affect the truth value of any waited-for condition invokes notifyAll to wake up any threads waiting for state changes.

Waits

- While loop is essential: when an action is resumed, the waiting task does not know whether the condition is true: it must check again. Avoid busy waits like:

```
protected void spinWaitUntilCond(){
  while (!cond_)
    Thread.currentThread().yield();
}
```

Notifications

- Good to start with blanket notifications using notifyAll
- notifyAll is an expensive operation
- optimize later