CSE 451: Operating Systems
Hard Lessons Learned

Windows
Reader/Writer Locks

Gary Kimura

A very simple model of Readers/Writers using semaphores

```
var mutex: semaphore = 1        ; controls access to readcount
    wrt: semaphore = 1          ; control entry for a writer or first reader
    readcount: integer = 0      ; number of active readers
```

```
writer:
        P(wrt)                  ; any writers or readers?
                <perform write operation>
        V(wrt)                  ; allow others
```

```
reader:
        P(mutex)                                ; ensure exclusion
          readcount++                           ; one more reader
          if readcount == 1 then P(wrt)         ; if we're the first, synch with writers
        V(mutex)
                <perform read operation>
        P(mutex)                                ; ensure exclusion
          readcount--                           ; one fewer reader
          if readcount == 0 then V(wrt)         ; no more readers, allow a writer
        V(mutex)
```

Windows Readers/Writers nuances

- Call EResource in Windows.
- Used the terms exclusive and shared access.
- Avoided starving exclusive by making shared requests wait
- Allowed recursive acquisition of a lock. Meant keeping ownership information
- Addressed an issue called priority inversion
- Then one hack added after another.
  - Added call to “Try” to acquire access without blocking
  - Added call to starve an exclusive waiter
  - Added call to release lock for a different thread
  - Augh…

Picture of the resource

Where we started

- ExInitializeResource
- ExAcquireResourceShared
- ExAcquireResourceExclusive
- ExReleaseResource

Added “features?”

- ExAcquireResourceShared(Wait);
- ExAcquireResourceExclusive(Wait);

- ExAcquireSharedStarveExclusive
- ExReleaseResourceForThread

- ExConvertExclusiveToShared
- ExDisableResourceBoost

- ExReinitializeResource
- ExSetResourceOwnerPointer
- ExDeleteResource

More added “features?”

- ExGetExclusiveWaiterCount
- ExGetSharedWaiterCount

- ExIsResourceAcquiredExclusive
- ExIsResourceAcquiredShared

- Bottom line: Learning to say “NO” to requests for adding new features.