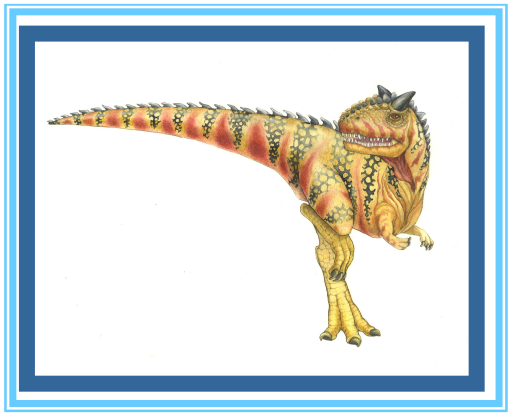
PRINCIPLES OF OPERATING SYSTEMS

Lecture 27

Reader- Writer Problem & Dining Philosopher Problem



Modified from Silberschatz, Galvin and Gagne & Stallings

Readers-Writers Problem

- A data set is shared among a number of concurrent processes
  - *Readers* – only read the data set; they do **not** perform any updates
  - *Writers* – can both read and write
- **Problem** – allow multiple readers to read at the same time.
  - Only one single writer can access the shared data at the same time
- Shared Data
  - Data set
  - Semaphore mutex initialized to 1
  - Semaphore wrt initialized to 1
  - Integer readcount initialized to 0

Readers-Writers Problem (Cont.)

- The structure of a writer process

```
do {
        wait (wrt) ;

            //    writing is performed

        signal (wrt) ;
} while (TRUE);
```

Readers-Writers Problem (Cont.)

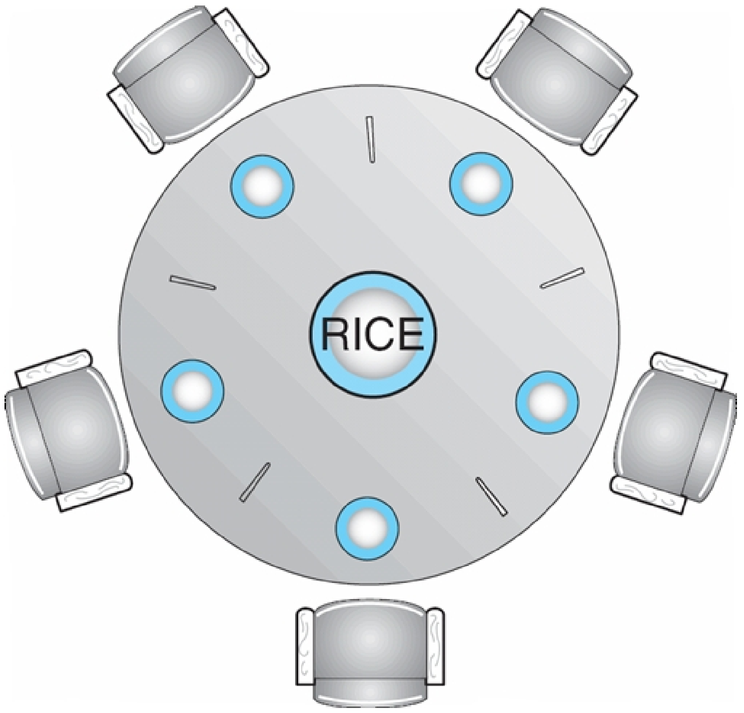
- The structure of a reader process

```
do {
        wait (mutex) ;
        readcount ++ ;
        if (readcount == 1)
                wait (wrt) ;
        signal (mutex)

            // reading is performed

        wait (mutex) ;
        readcount  - - ;
        if (readcount  == 0)
                signal (wrt) ;
        signal (mutex) ;
    } while (TRUE);
```

Dining-Philosophers Problem



- Shared data
  - Bowl of rice (data set)
  - Semaphore chopstick [5] initialized to 1

Dining-Philosophers Problem (Cont.)

- The structure of Philosopher *i*:

```
do  {
        wait ( chopstick[i] );
        wait ( chopStick[ (i + 1) % 5] );

              //  eat

        signal ( chopstick[i] );
        signal (chopstick[ (i + 1) % 5] );

             //  think

} while (TRUE);
```