

# Bilgisayar İşletim Sistemleri, Uygulama 7

## Ölümcül Kilitlenme

İstanbul Teknik Üniversitesi  
34469 Maslak, İstanbul

12 Nisan 2017

# Bugün

## Bilgisayar İşletim Sistemleri, Uygulama 7

### Ölümcül Kilitlenme

### Örnekler

# Ölümcül Kilitlenme

Joseph Heller'in Catch-22 adlı romanı bir çelişki üzerine kuruludur. Romanda, olayın gerçekleşmesi bir şarta bağlıdır ancak o şartın gerçekleşmesi de aynı olaya bağlıdır.

- ▶ Deneyimsiz biri iş bulamaz. İş bulamazsa deneyim kazanamaz.
- ▶ İyi bir takım olmak için iyi futbolcular gerekir. İyi futbolcular iyi takımlara transfer olur.

Benzer çelişkiler prosesleri ilgilendiriyorsa, işletim sistemi ölümcül kilitlenmeyle karşılaşabilir.

## Basit bir ölümcul kilitlenme örneği

```
1 // mutex variable declarations
2 pthread_mutex_t lock_1;
3 pthread_mutex_t lock_2;
```

```
1
2 void* faulty_functionA(void *arg){
3     pthread_mutex_lock(&lock_1); // start of Critical Region 1
4     printf("\nA is in Critical Region 1\n");
5     fflush(stdout); // to print out buffer contents immediately
6     sleep(2); // sleep for 2 seconds
7     pthread_mutex_lock(&lock_2); // start of Critical Region 2
8     printf("\nA is in Critical Region 2\n");
9     fflush(stdout); // to print out buffer contents immediately
10    pthread_mutex_unlock(&lock_2); // end of Critical Region 2
11    pthread_mutex_unlock(&lock_1); // end of Critical Region 1
12 }
```

## Basit bir ölümcul kilitlenme örneği

```
1 void* faulty_functionB(void *arg){
2     sleep(1); // sleep for 1 second
3     pthread_mutex_lock(&lock_2); // start of Critical Region 2
4     printf("\nB is in Critical Region 2\n");
5     fflush(stdout); // to print out buffer contents immediately
6     sleep(2); // sleep for 2 seconds
7     pthread_mutex_lock(&lock_1); // start of Critical Region 1
8     printf("\nB is in Critical Region 1\n");
9     fflush(stdout); // to print out buffer contents immediately
10    pthread_mutex_unlock(&lock_1); // end of Critical Region 1
11    pthread_mutex_unlock(&lock_2); // end of Critical Region 2
12 }
```

## Basit bir ölümcul kilitlenme örneği

```
1 int main(){
2     pthread_t threadA, threadB; // declaring two threads
3     pthread_mutex_init(&lock_1, NULL); // initializing mutex variables
4     pthread_mutex_init(&lock_2, NULL); // initially unlocked
5     if( pthread_create(&threadA, NULL, faulty_functionA, NULL)){ // creating threadA
6         printf("Thread creation error");
7         exit(1);
8     }
9     if( pthread_create(&threadB, NULL, faulty_functionB, NULL)){ // creating threadB
10        printf("Thread creation error");
11        exit(1);
12    }
13    if( pthread_join(threadA, NULL)){ // waiting for threadA to terminate
14        printf("Thread join error");
15        exit(1);
16    }
17    if( pthread_join(threadB, NULL)){ // waiting for threadB to terminate
18        printf("Thread join error");
19        exit(1);
20    }
21    pthread_mutex_destroy(&lock_1); // destroying mutex variables
22    pthread_mutex_destroy(&lock_2);
23    return 0;
24 }
```

## Basit bir ölümcül kilitlenme örneği: Çıktı

A is in Critical Region 1

B is in Critical Region 2

## Basit bir ölümcul kilitlenme örneği - düzeltilmiş

```
1 void* functionA(void *arg){
2     pthread_mutex_lock(&lock_1); // start of Critical Region 1
3     printf("\nA is in Critical Region 1\n");
4     fflush(stdout); // to print out buffer contents immediately
5     sleep(5); // sleep for 5 seconds
6     while(pthread_mutex_trylock(&lock_2)){ // try to acquire lock_2
7         pthread_mutex_unlock(&lock_1); // release lock_1
8         sleep(1); // sleep for 1 second
9         printf("\nA is WAITING\n");
10        fflush(stdout); // to print out buffer contents immediately
11        pthread_mutex_lock(&lock_1); // reacquire lock_1
12    }
13    // start of Critical Region 2
14    printf("\nA is in Critical Region 2\n");
15    fflush(stdout); // to print out buffer contents immediately
16    pthread_mutex_unlock(&lock_2); // end of Critical Region 2
17    pthread_mutex_unlock(&lock_1); // end of Critical Region 1
18 }
```



## Basit bir ölümcül kilitlenme örneği - düzeltilmiş

```
1 void* functionB(void *arg){
2     sleep(1); // sleep for 1 second
3     pthread_mutex_lock(&lock_2); // start of Critical Region 2
4     printf("\nB is in Critical Region 2\n");
5     fflush(stdout); // to print out buffer contents immediately
6     sleep(4); // sleep for 4 seconds
7     while(pthread_mutex_trylock(&lock_1)){ // try to acquire lock_1
8         pthread_mutex_unlock(&lock_2); // release lock_2
9         sleep(1); // sleep for 1 second
10        printf("\nB is WAITING\n");
11        fflush(stdout); // to print out buffer contents immediately
12        pthread_mutex_lock(&lock_2); // reacquire lock_2
13    }
14    // start of Critical Region 1
15    printf("\nB is in Critical Region 1\n");
16    fflush(stdout); // to print out buffer contents immediately
17    pthread_mutex_unlock(&lock_1); // end of Critical Region 1
18    pthread_mutex_unlock(&lock_2); // end of Critical Region 2
19 }
```

## Basit bir ölümcül kilitlenme örneği - düzeltilmiş: Çıktı

### Çıktı 1:

```
A is in Critical Region 1  
B is in Critical Region 2  
B is in Critical Region 1  
A is WAITING  
A is in Critical Region 2
```

### Çıktı 2:

```
A is in Critical Region 1  
B is in Critical Region 2  
A is in Critical Region 2  
B is WAITING  
B is in Critical Region 1
```

## Daha gerçekçi bir örnek (Yarış Durumu)

```
1 class Pair{ // Pair class declaration (C++)
2     int a;
3     int b;
4     pthread_mutex_t plock; // mutex variable
5 public:
6     Pair(int,int); // constructors
7     Pair(void){};
8     ~Pair(); // destructor
9     // overloaded operators for comparison
10    bool operator<(Pair &);
11    bool operator>(Pair &);
12    bool operator==(Pair &);
13    // methods for setting attributes
14    void setA(int);
15    void setB(int);
16    void setAB(int,int);
17    void print(string); // print method
18    // methods for mutex operations
19    void lock();
20    void unlock();
21};
```

## Daha gerçekçi bir örnek (Yarış Durumu)

```
1 // constructor
2 Pair::Pair(int a_in ,int b_in){
3     a=a_in;
4     b=b_in;
5 }
```

```
1 // set methods
2 void Pair::setA(int a_in){ a=a_in;}
3
4 void Pair::setB(int b_in){ b=b_in; }
5
6 void Pair::setAB(int a_in ,int b_in){
7     a=a_in;
8     b=b_in;
9 }
10
11 // print method
12 void Pair::print(string name){
13     cout << endl << name << " : ( " << a << " , "<<b<<") "<<endl;
14 }
```

## Daha gerçekçi bir örnek (Yarış Durumu)

```
1 // overloaded operators
2 bool Pair::operator<(Pair &other){
3     if(a<other.a)
4         return true;
5     if(a==other.a && b<other.b)
6         return true;
7     return false;
8 }
9 bool Pair::operator>(Pair &other){
10    if(a>other.a)
11        return true;
12    if(a==other.a && b>other.b)
13        return true;
14    return false;
15 }
16 bool Pair::operator==(Pair &other){
17    if(a==other.a && b==other.b)
18        return true;
19    return false;
20 }
```

## Daha gerçekçi bir örnek (Yarış Durumu)

```
1 int main(){
2     pthread_t mythreadA; // declaring mythreadA
3     Pair* x=new Pair(1,2);
4     Pair* y=new Pair(2,3);
5     // creating a list of two Pairs (x and y)
6     Pair* pList[]={x,y};
7     // creating mythreadA
8     if( pthread_create(&mythreadA, NULL, thread_function, (void*) pList)){
9         printf("error creating thread");
10        abort();
11    }
12    sleep(1); // to have a race
13    // set attribute a of x to 5 and print x
14    x->setA(5);
15    pList[0]->print("x");
16    // wait for mythreadA to terminate
17    if( pthread_join(mythreadA, NULL)){
18        printf("error joining thread");
19        abort();
20    }
21    delete x;
22    delete y;
23    return 0;
24 }
```

## Daha gerçekçi bir örnek (Yarış Durumu)

```
1 // thread handling function
2 void* thread_function(void *arg){
3     Pair** pList=(Pair**) arg;
4     // print x and y
5     pList[0]->print("x");
6     pList[1]->print("y");
7     sleep(1); // to have a race
8     // compare x and y and print the result
9     if ((*pList[0])>(*pList[1]))
10         cout<<endl<<"x>y"<<endl;
11     if ((*pList[0])<(*pList[1]))
12         cout<<endl<<"x<y"<<endl;
13     if ((*pList[0])==(pList[1]))
14         cout<<endl<<"x=y"<<endl;
15     return NULL;
16 }
```

## Daha gerçekçi bir örnek (Yarış Durumu): Çıktı

x: (1, 2)

y: (2, 3)

$x < y$

x: (5, 2)

x: (1, 2)

y: (2, 3)

x: (5, 2)

$x > y$



## Daha gerçekçi bir örnek (Ölümcül Kilitlenme)

```
1 // constructor
2 Pair::Pair(int a_in ,int b_in){
3     a=a_in;
4     b=b_in;
5     pthread_mutex_init(&plock , NULL);
6 }
7
8 // destructor
9 Pair::~~Pair(){
10    pthread_mutex_destroy(&plock);
11 }
```

```
1 // set methods (using mutex)
2 void Pair::setA(int a_in){
3     lock();
4     a=a_in;
5     unlock();
6 }
```

setB and setAB are modified similarly to include mutex.

## Daha gerçekçi bir örnek (Ölümcül Kilitlenme)

```
1 // mutex lock method
2 void Pair::lock(){
3     pthread_mutex_lock(&plock);
4 }
5
6 // mutex unlock method
7 void Pair::unlock(){
8     pthread_mutex_unlock(&plock);
9 }
```

## Daha gerçekçi bir örnek (Ölümcül Kilitlenme)

```
1 bool Pair::operator<( Pair &other){
2     // acquire own lock
3     lock();
4     sleep(1); // to ensure deadlock
5     // acquire other's lock
6     other.lock();
7     if(a<other.a){
8         // release locks
9         unlock();
10        other.unlock();
11        return true;
12    }
13    if(a==other.a && b<other.b){
14        // release locks
15        unlock();
16        other.unlock();
17        return true;
18    }
19    // release locks
20    unlock();
21    other.unlock();
22    return false;
23 }
```

operator> and operator== are modified similarly to include mutex.

## Daha gerçekçi bir örnek (Ölümcül Kilitlenme)

```
1 int main(){
2     pthread_t mythreadA, mythreadB; // declaring two threads
3     Pair* x=new Pair(1,2);
4     Pair* y=new Pair(2,3);
5     // creating two lists of Pairs (x,y) and (y,x)
6     Pair* pList[]={x,y};
7     Pair* qList[]={y,x};
8     // creating two threads
9     if( pthread_create(&mythreadA, NULL, thread_function, (void*) pList)){
10         printf("error creating thread");
11         abort();
12     }
13     if( pthread_create(&mythreadB, NULL, thread_function, (void*) qList)){
14         printf("error creating thread");
15         abort();
16     }
17     if( pthread_join(mythreadA, NULL)){ // waiting for threadA to terminate
18         printf("error joining thread");
19         abort();
20     }
21     if( pthread_join(mythreadB, NULL)){ // waiting for threadB to terminate
22         printf("error joining thread");
23         abort();
24     }
25     delete x; delete y;
26     return 0;
27 }
```

## Daha gerçekçi bir örnek (Ölümcül Kilitlenme): Çıktı

x: (2,3)

y: (1,2)

x: (1,2)

y: (2,3)