Teaching Cases Design of Process Mutex

Chunmei Wang

Institute of Information Engineering, Binzhou University, Binzhou 256600, China.

cmwang1982@163.com.

Abstract

The process mutex is not only the focus, but also the difficulty of operating system. For students, the process mutex is very difficult to understand. This paper designs a set of teaching cases. Through those cases, the important concepts such as process mutex, lock, integer semaphore, record semaphore, their relationship and characteristics are linked, so as to help students understand and master abstract knowledge points effectively.

Keywords

Operating system, Process mutex, Semaphore.

1. Introduction

Operating system course is the core course of computer-related subjects. As a principle course, it has so many concepts, theories and abstract contents that it is hard for students to understand and master. Especially the process mutex, it contains the concepts of process concurrence, mutex sharing mode of critical resource and record semaphore. Obviously, it is the focus and difficulty of operating system. Most of the students can not really understand the principle of operating system, and thus affect the learning effect of students. In view of the above situation, the corresponding teaching cases are designed in the teaching process. Through those teaching cases, students can easily grasp the knowledge of the process mutex, thus mobilize the enthusiasm of students, enhance the confidence of learning .

2. Process Mutex Cases and Solutions

2.1 Cases Design

2.1.1 Sub-section Headings

Process mutex is an indirect restriction caused by sharing critical resources. The solution is how to ensure the process concurrency while ensuring the mutual exclusion of critical resources. In fact, there are many critical resources in real life, such as dining room utensils, cinema seats, classroom of teaching building, shared bicycle and so on. Next, I will take the classroom of the teaching building as an example to explain the process mutex.

We concrete classroom examples. One day, Mr. Zhang wants to occupy Classroom 101 of Teaching Building NO.1 to teach the Data Construction (DC) for class A, meanwhile Ms. Wang wants to occupy the same classroom to give Class B an Operating System (OS) course. But for Classroom 101, it can only be used by the first class to enter the classroom, while the other class must wait. The process of regular classes is as follows:

```
Mr. Zhang( )
{ if(Classroom 101 is occupied)
    waiting;
 else
    {
     Lecture for 45 minutes;
Take a 10-minute break between classes;
```

```
Lecture for 45 minutes;
    }
}
Ms. Wang( )
{ if(Classroom 101 is occupied)
    Waiting;
 else
    {
     Lecture for 45 minutes;
Take a 10-minute break between classes;
Lecture for 45 minutes;
    }
}
```

2.2 Lock.

In this case, the teacher corresponds to the process and the classroom corresponds to the critical resource. Between the two lectures, there will be a 10-minute break. Classroom 101 is considered to be free at this break, but other classes are not allowed to enter. That is to say, the classroom is only allowed to be used by one class at a time, so how to ensure that other classes do not enter Classroom 101 during the break time? In reality, we can install a door lock for each classroom through the door lock. When the classroom is free, the lock is open; if the classroom is used, it can lock the door..

The above process can be revised as follows:

```
Mr. Zhang ( )
{if( lock is closed)
Waiting;
else
{
Lock the door;
Lecture for 45 minutes;
Take a 10-minute break between lectures;
Lecture for 45 minutes;
Turn on the lock;
}
 }
Ms. Wang( )
{ if( lock is closed)
Waiting;
else
{Lock the door;
Lecture for 45 minutes;
Take a 10-minute break between lectures;
Lecture for 45 minutes;
Turn on the lock;
}
```

```
}
```

Boolean variable W stands for the lock, false means that the lock is open, which further means that the classroom is free; true means that the lock is closed, which further means that the classroom is occupied. lock (W) stands for the lock operation; it also represents the application for classroom operation; unlock (W) stands for the unlock operation, which means that classroom resources are released.

The above lock is implemented in C-like language as follows:

```
lock(W)
{ while(W==true) ;
  W=true;
}
unlock(W)
{
   W=false;
}
```

The mutually exclusive use of classroom is solved by lock, which is realized as follows:

```
boolean W=false;
Mr. Zhang( )
{lock(W);
Lecture for 45 minutes;
Take a 10-minute break between lectures;
Lecture for 45 minutes;
unlock(W);
}
Ms. Wang( )
{lock(W);
Lecture for 45 minutes;
Take a 10-minute break between lectures;
Lecture for 45 minutes;
unlock(W);
}
```

2.3 Integer Semaphore.

Modify the above case: There are 10 classrooms in Teaching Building No. 1 from 101 to 110. All teachers can attend classes as long as they apply to any of them. Then 10 locks W1-W10 should be set up for 10 classrooms. If teachers apply for classrooms in the order of 101-110, Then the first teacher applies for 101 successfully, the second teacher applies for 101 unsuccessfully and 102 successfully, and so on. All teachers starting from the eleventh need to apply for 101-110 before they can be sure that they can't apply for any classroom and need to wait.

Through the above cases, it is obvious that the lock guarantees the mutually exclusive sharing of the classroom, but the efficiency is too low. If there is a counting card in front of the teaching building, which has an integer variable S records the free classroom of teaching building No. 1 at the current time, when the teacher finds S > 0, let the S minus one, and then occupy a classroom. When S <=0, the teacher wait outside the building; When a teacher finishes his lecture, he releases a classroom and adds the value of S to one. This not only ensures the mutually sharing of each classroom, but also

improves the efficiency of applying for the classroom. Use C-like language to achieve the above functions:

```
wait(S){
while(S<=0)  ;
S--;
}
signal(S){
S++;
}
```

Then the teacher process of teaching becomes:

```
int S=10;
Teacheri( )
{wait(S);
Lecture for 45 minutes;
Take a 10-minute break between lectures;
Lecture for 45 minutes;
signal(S);
}
```

The above use of an integer variable S to represent the number of resources and wait (S) and signal (S) operations is called the integer semaphore. Obviously, the integer semaphore is more effective than the lock.

2.4 Record Semaphore.

In the integer semaphore, there are two shortcomings: first, when S<=0 in the wait(S), the blank statement is executed and then the while loop is continued to judge whether the S<=0 is valid. That is to say, when the application resource of the process is not satisfied, the CPU is still needed to execute the while loop, which is locked. This shortcoming also exists in the lock. Second, when S==0: there are two situations: one is that 10 teachers occupy 10 classrooms and there is no class waiting; the other is that 10 teachers occupy 10 classrooms while other teachers are waiting. If we can distinguish the above two situations, we can choose whether to notify the waiting teacher to go to the classroom after the teacher releases the classroom resources. In the record semaphore, the initial value represents the number of free classrooms. List is a process list pointer that links all waiting processes which apply for classrooms unsucessfully. Then in the record semaphore, all the teachers that want to apply for classroom should first reduce S.value by one, if the S.value >= 0, it means that the application classroom is successful ;Otherwise, S.value < 0 indicates that all classrooms are occupied, they will link to the list and wait. At this time, S.value is negative and its absolute value is the number of teachers who are waiting in the list. When the teacher releases the classroom, S.value should plus one first, if S.value <=0 means that there are waiting teachers in the list, so go to the list to wake up a teacher, otherwise it means that there are no waiting teachers .Record semaphore is implemented in C-like language as follows:

```
typedef  struct{
int value;
struct process_control_block  *list;
}semaphore;
wait(S){
S.value--;
```

```
if(S.value<0) block(S.list);
}
signal(S){
S.value++;
if(S.value<=0) wakeup(S.list)
}
```

Then record semaphore can solve classrooms as follow:

```
semaphore  S.value=10;
Teacheri( )
{wait(S);
Lecture for 45 minutes;
Take a 10-minute break between lectures;
Lecture for 45 minutes;
signal(S);
}
```

In the record semaphore, it ensures that the classroom is free and the teacher who applies for classroom can enter into use. If the classroom is occupied, other teachers will give up CPU voluntarily and wait for in the list.

3. Conclusion

In this case, the concepts of process mutex, lock, integer semaphore and record semaphore in operating system, as well as the relationship between them, are well displayed. Through specific cases, beginners can better grasp the above outline on the basis of understanding. Read, in order to make the best use of one example and another. This case has been used in teaching, and has achieved good teaching results. Students' mastery of process mutex has been significantly improved.

Acknowledgements

This research is supported by Professional Core Course Project of Binzhou University (Grant NO.BZHXKC201412).

References

[1] X.D. Tang, H.B. Liang, F.P. Zhe, Z.Y. Tang. Computer Operating System (4th edition) (Xi'an University of Electronic Science and Technology Press,China 2014)

[2] D.H. Guo: Case Teaching: History, Essence and Development Trend. Higher Science Education, Vol.77(2008) No.1,p.22-24.

[3] J.Z. Zheng. Case Teaching: A New Approach to Teachers'Professional Development. Educational Theory and Practice, Vol.22(2002) No.7,p.36-41.

[4] A.P. Wu, M.Z. Wang, J.Z. Wang. Research and practice of case teaching method. Journal of Changchun University, Vol.12(2002) No.5,p.22-23.

[5] X.S. Cai, S.H. Yu, J.B. Dai, et al. Application of Case Method in the Teaching of Operating System Principles. Journal of Changchun Normal University, Vol.34(2015) No.8,p.123-125