OSLab: A Hand-on Educational Software for Operating Systems Concepts Teaching and Learning

Farzaneh Zareie
Hamedan University of Technology
6 PUBLICATIONS **5** CITATIONS

OSLab: A Hand-on Educational Software for Operating Systems Concepts Teaching and Learning

Farzaneh Zareie, Mahsa Najaf-Zadeh
Computer Engineering Faculty
Iran University of Science and Technology (IUST)
Tehran, Iran
{Farzaneh_Zareie, Mahsa_Najafzadeh}@Comp.iust.ac.ir

Abstract— **In this paper an interactive educational software is introduced. The introduced software assists learners and teachers in learning and teaching operating systems concepts in a more efficient manner than traditional way (using paper and pen). It covers two fundamental operating systems' subjects namely memory management and CPU scheduling. OSLab is developed by Java programming language and takes advantage of its concurrent programming features. User can define new process (with its name, size and needed time to complete) any time he/she wants and once it created, system updates memory and CPU maps. It also updates both maps in time intervals according to defined CPU time slice. All of the system settings including memory's page size and CPU's time slice, can be changed by user. Using this system, students can find out what happens in CPU and memory every time. It includes a set of prominent algorithms in both areas. For convenience of users it also provides users the ability of adding his/her algorithms and running them.**

Keywords- educational software; operating systems; cpu scheduling; memory management.

I. INTRODUCTION

Operating systems is one of the most fundamental courses in computer science. In this course students learn what operating systems do and how they work. Two of the most important tasks an operating system does, are CPU scheduling and memory management.

CPU scheduling is the basis of multi-programmed operating systems. An operating system handles several running processes by switching the CPU among them. There are some CPU-scheduling algorithms that an operating system can use.

Whenever a process runs, all its needed data should load in memory. Regards to the limited size of RAM, all needed data can't load once in it. So RAM's content should be replaced by needed data every time. Several memory management algorithms have been introduced, too. For more information on these two areas please refer to [1].

In this paper we have introduced an educational visualization software for learning and teaching these two important concepts. We named the software OSLAB. The distinctive feature of OSLAB is that user can develop his/her own algorithm, run it and see its result. So OSLAB can be used by everyone who wants to contribute to develop and examine new innovative algorithms in these two areas.

In the rest of this paper, we described the reasons that make educational software an essential and important part of learning and teaching in part II. Part III studies related works on educational softwares. The features of OSLAB are addressed in part IV. Part V has explained how OSLAB works with a simple example. Conclusion remarks are explained in VI.

II. WHY EDUCATIONALVISUALIZATION SOFTWARE?

Computer science students use computer to solve problems of some practical courses such as programming but for theoric courses such as operating systems, compiler design concepts and etc, they should solve problems manually. It has some drawbacks among them the most important is that they don't receive any immediate feedback on their solutions. So they don't know whether their solution is correct or not. Another problem with this way is that manual solving problem is so tedious. It's a confusing and time consuming process.

The same story is held about teachers. Teachers of these courses also bother from traditional teaching for some reasons. First of all, solving problems on blackboard is time consuming and error prone. So teacher has to select a few small problems to save time. Secondly, teacher can't let every student participate in solving problem. So he/she can't be sure that all of the students learn the concepts.

Regards to what said, applying educational visualization software which enables students and teachers to solve problems step by step is essential. Taking advantage of such tools teachers can select every problem to solve. Students also can repeat solved problem as many times as they want. Both of teachers and students can introduce their desired problems and see the steps of their solution, too.

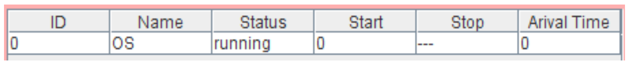
III. RELATED WORKS

The trend to develop and use educational visualization software for learning and teaching is relatively new. In [2] an online educational tool is introduced that covers *Data Structures* concepts. It enables students to guess the solution of problems. Another tool for teaching and learning *Data Structures* is introduced in [3]. There are such educational tools for *Algorithms Analysis and Design*, too [4]. [5, 6] have described educational tools that simulate virtual laboratories for *Formal language and Automata Theory* learning. Such educational softwares for *Compiler Design* concepts are developed in [7, 8, 9], too.

Studied educational softwares work as a virtual laboratory for courses. They have user friendly GUIs that help users to work with them in a convenient manner. In [10] it has shown that such tools have a considerable impact on student's learning.

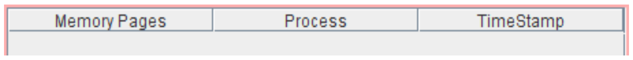
IV. Overview Of OSLAB

OSLAB is a java applet with one main page. This page has two parts. One part is related to memory management and another part is related to CPU-scheduling. Fig 1. shows the CPU-scheduling part and Fig 2. shows the memory management part.



ID	Name	Status	Start	Stop	Arival Time
0	OS	running	0	---	0

Figure 1. OSLAB's CPU-scheduling part



Memory Pages	Process	TimeStamp

Figure 2. OSLAB's memory management part

As Fig 1. shows, CPU-scheduling part is a table with six columns. Each row of this part is relevant to one process. According to the fact that operating system (OS) is running all the time, the first row is initiated by it. It's columns are as bellows:

1) *ID: This column shows the unique identification of each process.*
2) *Name: This column shows the name of process.*
3) *Status: This field determines the status of the process. Status of process includes running, suspended and completed.*
4) *Start: This column shows the last time that a process starts to run.*
5) *Stop: This column shows the last time that a process comes back to processes'queue (a queue in which processes are stored).*
6) *Arrival Time: This field shows the arrival time of process.*

When a process wants to start, if all of its needed resources are available, it is added to processes' queue. CPU has some defined time slices. After each time slice CPU switches from running process to another process in front of the queue. After each time slice these fields change: Status, Start and Stop.

Each RAM is divided into equal parts called pages. According to Fig 2. Memory management part is a table that has three columns. As operating system has its own part of RAM, this part is not initialized. In the other words, this part shows only user part of RAM. This part's columns are as bellows:

1) *Memory Pages: This column shows the number of pages that are loaded into RAM.*
2) *Process: This field shows the name and ID of the process related to loaded page.*
3) *Time Stamp: This fieald shows the order of page replacement.*

Thanks to concurrent programming in Java, the two parts are updated for each CPU switch regards to time slices.

OSLAB has a menu with three keys. The first key is *System* that enables user to create new processes, define new algorithm for memory management and define new algorithm for CPU-scheduling. Fig 3. shows the *System* key.

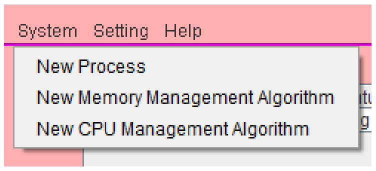


Figure 3. The *System* key of OSLAB's menu

The second key of the menu is *Setting*. This key enables the user to change the configuration of system such as CPU time slice, memory's page size, algorithms he/she wants to execute and so on. Fig 4. shows this key.

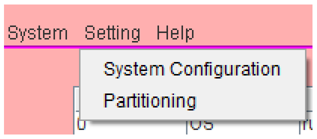
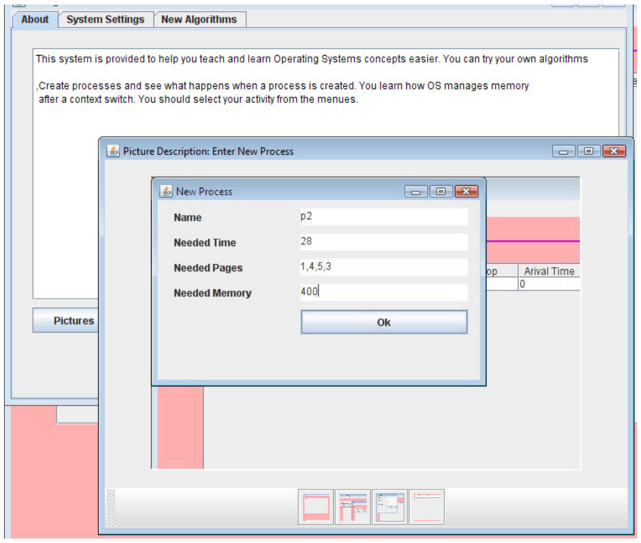


Figure 4. The *Setting* key of OSLAB's menu

The last key of the menu is *help* which provides essential guidance to use OSLAB. Fig 5. shows the help of OSLAB.



This system is provided to help you teach and learn Operating Systems concepts easier. You can try your own algorithms

,Create processes and see what happens when a process is created. You learn how OS manages memory after a context switch. You should select your activity from the menues.

Picture Description: Enter New Process

New Process

Name	p2
Needed Time	28
Needed Pages	1,4,5,3
Needed Memory	400

Figure 5. *Help* of OSLAB

As shown in Fig 5. help of system also contains pictures to illustrate how it works.

The next section shows how OSLAB works with a running example.

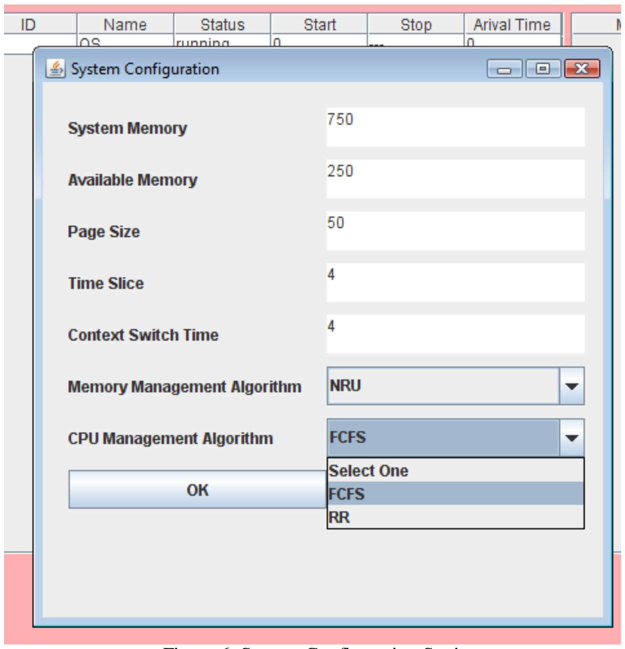
V. A Running Example

This section describes how OSLAB works. For each part (CPU-scheduling and memory management) two well known algorithms are implemented. Implemented algorithms for CPU-scheduling are First Come First Served (FCFS) and Round Robin (RR). Implemented algorithms for memory management are First In First Out (FIFO) and Not Recently Used (NRU). As said before, user can add as many algorithms as he/she wants.

The running example pass through these steps:

1) *Set system configuration*
2) *Create two processes*
3) *Develop a new CPU-Scheduling algorithm*

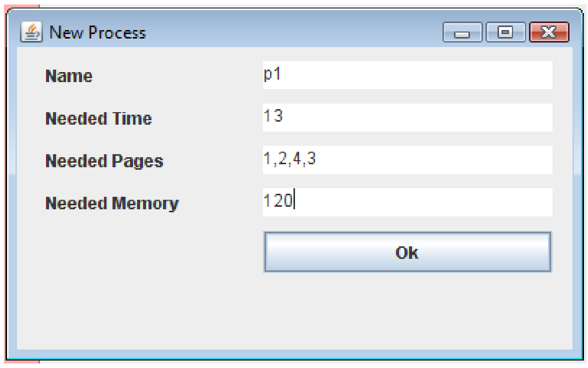
Fig 6. shows new configuration of OSLAB. System configuration includes determination of system memory, available memory, memory page size, time slice, context switch time and desired algorithms for both memory management and CPU-scheduling.



System Memory	750
Available Memory	250
Page Size	50
Time Slice	4
Context Switch Time	4
Memory Management Algorithm	NRU
CPU Management Algorithm	FCFS

Figure 6. System Configuration Setting

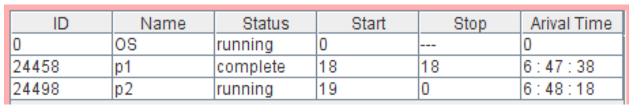
The next step of running sample is creating two processes. We have created two processes named p1 and p2. Fig 7. shows the creation of p1.



Name	p1
Needed Time	13
Needed Pages	1,2,4,3
Needed Memory	120

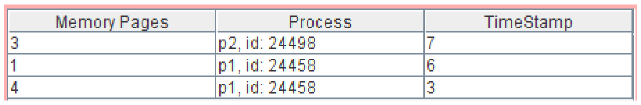
Figure 7. Creation of a New Process

Once each process is created, OSLAB updates CPU and memory maps. Fig 8. shows a snapshot of CPU-scheduling part (a) and memory management part (b) and Fig 9. shows another snapshot of OSLAB parts after completion of the two processes.



ID	Name	Status	Start	Stop	Arival Time
0	OS	running	0	---	0
24458	p1	complete	18	18	6 : 47 : 38
24498	p2	running	19	0	6 : 48 : 18

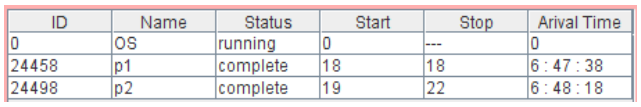
(a)



Memory Pages	Process	TimeStamp
3	p2, id: 24498	7
1	p1, id: 24458	6
4	p1, id: 24458	3

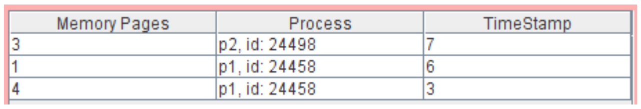
(b)

Figure 8. A Snapshot of System (a) CPU-Scheduling Part and (b) Memory Management Part



ID	Name	Status	Start	Stop	Arival Time
0	OS	running	0	---	0
24458	p1	complete	18	18	6 : 47 : 38
24498	p2	complete	19	22	6 : 48 : 18

(a)



Memory Pages	Process	TimeStamp
3	p2, id: 24498	7
1	p1, id: 24458	6
4	p1, id: 24458	3

(b)

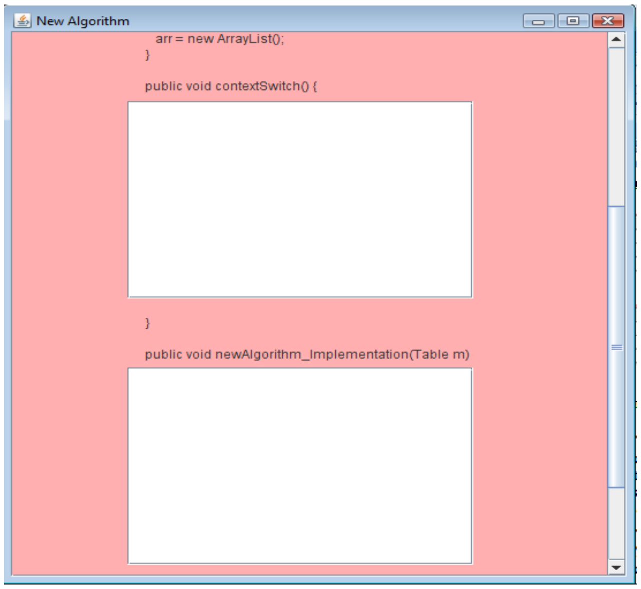
Figure 9. The Last Snapshot of System (a) CPU-Scheduling Part and (b) Memory Management Part

The last step in this example is implementing a new algorithm for CPU-scheduling. As said before the distinctive feature of OSLAB is its ability to add new algorithms. Using this feature, user can develop new algorithms for CPU-scheduling and memory management problems.

To implement a new algorithm, at first OSLAB asks user his/her algorithm name. In this example we called new algorithm as "newAlgorithm". Then OSLAB opens a new window in which user can add his/her desirable code. Fig 10. shows this window.

As demonstrated in Fig 10. all needed data structures for implementing a new algorithm is available. User just fill body of two functions: contextSwitch() and newAlgorithm_Implementation(Table).

After adding a new algorithm, System Configuration window (Fig 6) changes. So user can select new algorithm as well as OSLAB's predefined algorithms. Figure 11. shows this change.



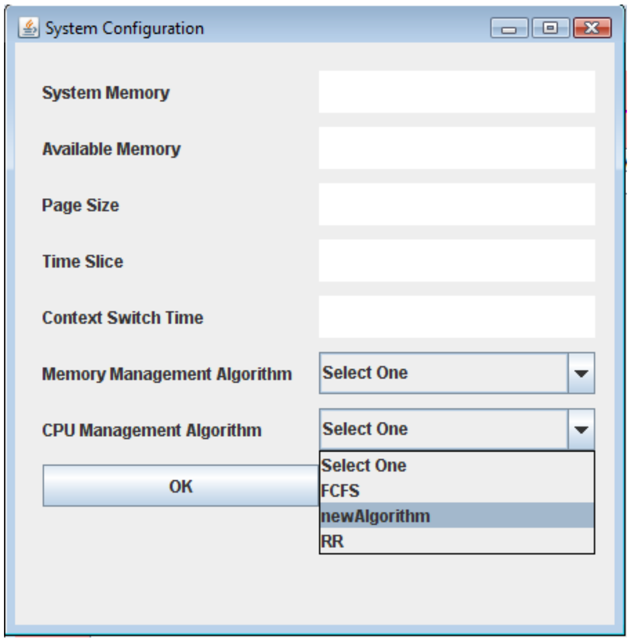
```
    arr = new ArrayList();
  }

  public void contextSwitch() {

  }

  public void newAlgorithm_Implementation(Table m)
```

Figure 10. OSLAB's Window for Adding New User Defined Algorithms



System Memory

Available Memory

Page Size

Time Slice

Context Switch Time

Memory Management Algorithm: Select One

CPU Management Algorithm: Select One

Figure 11. System Configuration Window Changes Related to Developing New Algorithm

VI. CONCLUSIONS AND FUTURE WORLS

In this paper a new educational visualization software is introduced. This software which called OSLAB helps learners and teachers to learn and teach Operating Systems concepts in a convenient manner. OSLAB's distinctive feature is that it allows user to add new algorithms to the system and execute them. This capability doesn't have seen in other educational softwares for other courses.

Regards to the fact that computers make our way of life more easier, applying computer-based systems in education is desirable. We are going to develop our research in this area and make educational softwares more intelligent and user friendly. Another goal we want to reach is to make these tools web-based and therefore widespread.

REFERENCES

[1] A. Silberschotz, P. B. Galvin, G. Gagne, Operating system concepts, 6th ed., John Wiley and Sons, 2002.

[2] D. J. Jarc, M. B. Feldman, "An empirical study of web-based algorithm animation courseware in an Ada data structure course", ACM SIGAda Int. Conf. on Ada, 1998.

[3] D. J. Jarc, "Interactive data structure visualizations", University of Maryland, 2004.

[4] A. Korhonen, "Visual algorithm simulation", Ph.D. Thesis, Helsinki University of Technology, 2003.

[5] White T. M., Way T. P., "Jfast: A java finite automata simulator". The thirty-seventh SIGCSE Technical Symposium on Computer Science Education.SIGCSE, March 2006.

[6] Rodger S. H., Lim J., Reading S., "Increasing interaction and support in the formal languages and automata theory course", The 12th Annual Conference on Innovation and Technology in Computer Science Education, 2007.

[7] Blythe S., Rodger M. J., Rodger S. H., "LLparse and LRparse: Visual and interactive tools for parsing". Twenty _fifth SIGCSE Technical Symposium on Computer Science Education, 1994.

[8] James M., "A software tool to aid in understanding LL parsing", Master's Project, Rensselaer Polytechnic Institute, May 1993.

[9] N. S. Bathaeian, F. Zareie, S. Homaei-pour, "CDLP: An interactive educational software for compiler designing", 4th Conference on Information and Knowledge Technology, 2012.

[10] Badre A., Lewis C., Stasko J., "Empirically evaluating the use of animations to teach algorithms", Proceedings of the 1994 IEEE Symposium on Visual Languages,1994.

Filename:	OSLAB
Directory:	C:\Users\farzaneh\Documents
Template:	C:\Users\farzaneh\AppData\Roaming\Microsoft\Templates\Normal.dotm
Title:	Paper Title (use style: paper title)
Subject:	
Author:	IEEE
Keywords:	
Comments:	
Creation Date:	2/14/2013 10:47:00 PM
Change Number:	100
Last Saved On:	3/29/2013 2:35:00 PM
Last Saved By:	farzaneh
Total Editing Time:	484 Minutes
Last Printed On:	3/29/2013 2:35:00 PM
As of Last Complete Printing	
Number of Pages:	4
Number of Words:	1,917 (approx.)
Number of Characters:	10,933 (approx.)