Journal of computer science and software applications
https://www.mfacademia.org/
ISSN:2377-0430
Vol. 3, No. 2, 2023

Design and Implementation of a Visualized Operating System Simulation Using Java Multithreading

Harper Robinson[1],Matthew Walker[2]

Indiana University of Pennsylvania-Main Campus[1],Indiana University of Pennsylvania-Main Campus[2]

Haeper000@gmail.com[1],matthew@gmail.com[2]

Abstract:

The design and implementation of a simulation of the operating system provide insight into the fundamental working principles of computer operating systems and their various functional modules. By utilizing a visual interface, users can rapidly acquire basic theoretical knowledge of operating systems and understand the scheduling processes. As a program responsible for managing a computer's hardware and software resources, an operating system addresses issues such as memory management, process management, storage management, and device management. Employing a multi-threaded Java programming approach, the simulation system manages operating system transactions through coordinated thread management and visually displays various management functions using Java Swing components. This simulation system not only enhances users' interest in learning about operating systems but also offers a solution for visually demonstrating each function of the operating system.

Keywords:

Operating system; visual display; simulation design; Java language; multithreading.

1. Introduction

Computer operating system is important to learn computer professional knowledge theory part. In the operating system process scheduling, memory allocation, file read and disk management algorithm and the design idea is the essence of computer program design concept. Knowledge of computer operating system theory study cannot rely on interpretation and problem sets, each functional module of visual image display can help learners understand the design principle and operation mode, greatly improve the efficiency of learning [1]. The traditional operating system simulation learning software design implementation, some of these modules separately restricted to a particular module algorithm independently, learners in the process of learning can't perceive the internal relations of various modules, couldn't reflect the operating system process continuity and systemic.

Java language as a cross-platform design programming language, the use of the JVM running design application. Integrated visualization development component, the corresponding Java itself is good support for multithreaded programming, this article through to the various functions of the operating system decomposition and design in detail, using the Java language as the programming language, the operating system process management, storage management, equipment management, file management four modules are simulated and user interface design, design and implement a multi-threaded simulation single-user operating system, this system friendly interface, the learners can learn through interaction to master the theoretical knowledge of the operating system and the corresponding module design principle method [2].

1.1. File Management and User Interface

File management and user interface refers to a single user disk file management, including document logical structure, physical structure, directory, recovery disk allocation, file protection and the realization of the user interface.

Document logical structure uses the flow structure, as a result of the simulation system focused mainly on the logical structure of the file, system files can be used in a text file. The file structure is used in the simulation of disk can also be read through the data stream, according to the structure of the data stream setup disk the size of the plate and each disk blocks. Combining with the characteristics of real disk storage and define the 0, 1 piece for file allocation table, 2 pieces is defined as the root directory, directory entry visual requirements, the rest of the disk storage directories and files, directory structure USES the directory tree structure, the concrete contents are shown in table 1. In order to facilitate design recovery disk allocation algorithm, files on disk blocks is exclusive.

Table 1. Directory entry byte definition

Table of contents	Number of bytes	remarks
Each directory entry	8	
Directory name or file name	3	
Extension	1	The extension of the executable needs to be defined
Directory and file attributes	1	
Starting plate number	1	
File length	2	Directory has no length

Disk allocation can refer to MS-DOS fat (file allocation table) allocation method, using link structure. The system uses file allocation table to record the usage of disk space and the pointer of link structure. Each item in the file allocation table should be defined according to the fixed byte size, and the number of items in the allocation table should be calculated according to the number of disk blocks. The file allocation table is stored on the 0 and 1 blocks in the disk space.

When the user operates the simulation operating system, it needs to complete through the corresponding interface. Based on the interactive and friendly design concept, the system should design a visual interface of commands that can be typed from the keyboard. In this module, the disk directory and disk usage should be visually displayed on the interface of the simulation operating system. Users can see the directory structure of the disk and the occupancy of disk blocks through data or intuitive graphics.

1.2. Storage Management and Device Management

In the storage management part, array can be used to simulate the memory space, which mainly completes the allocation and recovery of memory space and storage protection.

In the simulation system, the memory is divided into two parts: system area and user area. The system area stores the process control block (PCB) and memory allocation table. The user area is used to store the running process. The user area can define the byte size of the user area according to the simulation requirements. The dynamic partition storage management mode is adopted in the storage management. The memory usage in this module should be displayed in a visual form on the interface of the simulation operating system, and the allocation of memory area can be marked in different colors [3].

Equipment management mainly includes equipment distribution and recycling. A variety of exclusive devices can be set up in the simulation system. The number of devices is different according to the type, and different entities of the same device can be considered as the same. The system uses deadlock prevention method to deal with the deadlock caused by the application of exclusive device. The usage of the devices in this module should be displayed in the interface of the simulation operating system in a visual form. You can see whether each device is used, which process is used, and which processes are waiting for the device.

1.3. Process Management

Process management is the brain of the whole simulation operating system. Files, storage and devices are scheduled by process. The function of process management mainly includes process scheduling, process creation and revocation, process blocking and wake-up, and the realization of interrupt. From the simulation module, the CPU (central processing) is the main part Unit) function simulation, main register function simulation, interrupt simulation, clock simulation, process control simulation, process scheduling simulation, etc.

CPU simulation is mainly responsible for interpreting the instructions in the "executable file", as shown in Table 2.

Table 2. Executable file definition

Executable file content	definition
Y=?	Assign a single digit to y
Y++	Y plus 1
Y--	Y minus 1
#??	The first? Is the device, and the second? Is a one digit integer indicating the time of using the device
end	End of file
file length	2

The CPU can only interpret the instructions in the instruction register IR (instruction register). When a process is running, the corresponding instructions should be stored in the simulated "instruction register IR" according to the execution position of the process. For several registers used by the core: program status register PSW (program status word), instruction register IR, program counter PC (program counter), data buffer register DR (data register), global variable or array simulation can be used.

Table 3. Interrupt handling

Interrupt type	Treatment method	remarks
End of program (interrupt formed by executing instruction end, soft interrupt)	Write the file pathname and calculated value to the file, cancel the process, and schedule the process.	
Time slice rotation	The CPU of the running process is saved in the process control block, and then the process is scheduled	CPU field:values of each register
I / O interrupt (device I / O)	Wake up the process of input and output completion, and wait for another process of the device to wake up。	

Interrupt simulation is carried out by detecting the defined values in the PSW array of program status registers. Before executing the instruction, the array of program status registers is checked to determine whether there is an interrupt. If there is an interrupt, the interrupt processing is performed first, and then the interpretation instruction is run. Instruction interpretation and interrupt running should be completed in the thread. The types and processing methods of interrupts are shown in Table 3.

There are two clock variables in the system: absolute clock and relative clock. The clock can be simulated with global variables. The absolute clock simulates the running time of the entire operating system. The relative clock is used as the time slice count in the process of process execution. A fixed value is set at the beginning of process scheduling. With the operation of the absolute clock, the corresponding value will be accumulated and automatically reduced. When the value is 0, the corresponding value will be automatically reduced The interrupt operation of time slice rotation is issued.

Process control block (PCB) simulation mainly includes process identifier, main register group, process status, blocking reason, etc. The process control block consists of three queues: blank queue, ready queue and blocking queue. There is only one running process and the system starts with a blank queue.

The main functions of process scheduling are as follows: when the time count of time slice is 0, the field (register group) of the running process is saved in the process control block of the process, and then a process is selected from the ready queue, and the contents of each register recorded in the process control block of the process are restored to each register of the CPU 。In order to ensure the normal operation of the system, we should simulate the idle wandering
process similar to windows. When the thread queue is empty, the system calls the process to run. Ready to call the process when it is ready.

The visual interface should display the system clock; display the process ID of running process, running instructions, intermediate results, relative clock, register content; process ID in ready queue; process ID in blocking queue, etc.

1.4. User Interface

The user interface mainly receives commands typed from the keyboard through visualization. It can complete the basic functions of the operating system by defining command keywords, such as creating files, deleting files, displaying files, copying files, editing files, establishing directories, deleting empty directories, running executable files (creating processes), etc.

2. System Detailed Design

2.1. Module Division

According to the requirements, the whole system is divided into five modules, as shown in Figure 1, which are bus module, CPU module, memory module, device module and disk module. Each module creates a class to realize its function. The data communication and corresponding algorithm flow control are carried out among the modules through functions.

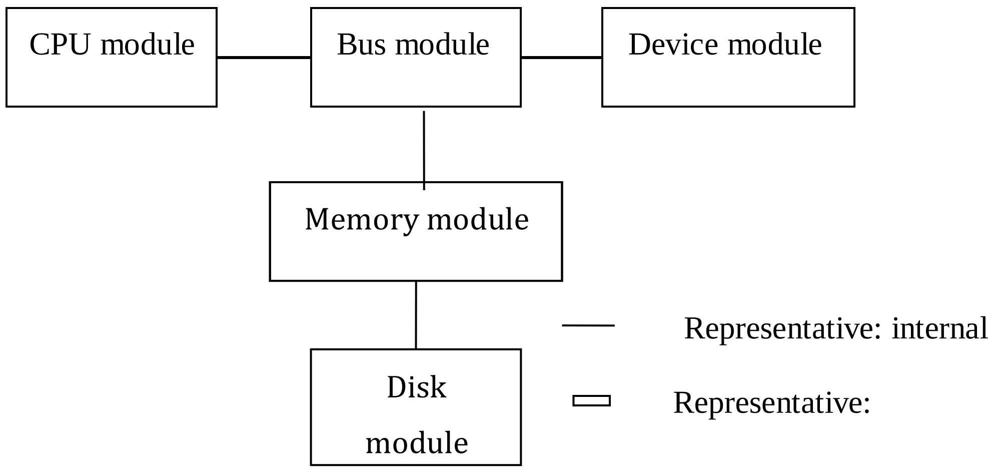


Figure 1. System structure

The Java language is used to decompose each module, as shown in Table 4:

Table 4. Simulation program design

Class name	Function
Cpu Class	Function of analog CPU
Sys Class	Function of analog bus
Allocation Class	Functions of analog devices
Memory Class	Function of analog memory
Index Class	Function of analog disk

The bus simulates sys class to create CPU simulation CPU class, device simulation allocation class, memory simulation memory class and disk simulation index class to initialize them. Each class only creates an object once. The internal interface is provided in the class of CPU simulation, device simulation and memory simulation, and the communication of each module is coordinated through bus simulation. In CPU simulation, device simulation and bus simulation, threads are added to drive them to make the three modules run independently. In disk simulation and CPU simulation, providing external user interface is mainly to operate files and directories and create processes.

After each module has been successfully implemented, it is integrated and controlled by sys class (bus simulation).

2.2. Module Execution Relationship

Thread is added to the bus simulation module as the whole driver of the system. The threads of CPU simulation module and device simulation module are added into the bus simulation module. As the entrance of the whole system, bus simulation module drives the execution of CPU simulation module, device simulation module and memory simulation module.

When the simulation system starts to run, the system thread is opened to drive the system time. Open the thread of CPU simulation module and device simulation module. The CPU simulates the thread of the module to execute the interrupt and explain instruction cycle for 1 second, and the thread of the device executes the loop for 3 seconds to drive the detection of waiting queue once. Each thread runs independently to ensure that the whole simulation system is always running.

The simulation of file system first simulates a 64 * 128 two-dimensional array, and initializes five root directories in the array. When creating files or subdirectories, first find the root directory and create it under the root directory. When writing the contents of the file (operation instructions include: create file, create directory, delete file, delete directory, display file, edit file, copy file), first find the directory entry of the file, find the starting address of the file from the directory entry for writing. When writing a data block, find the data block linked to this file in the file allocation table of Block 0 and block 1, and then At the end, write it down disk.txt Analog disk.

The simulation of the process first creates the process, receives the file path name and file name, which is received by the bus and then sent to the CPU simulation module. The CPU simulation module transfers the file path name and file name to the memory simulation module, so that the memory simulation module creates the process. When creating the memory, it first detects the available memory space, and uses the first adaptation algorithm to allocate the dynamic memory and read the data from the disk According to and write to the memory simulation module, create the initialization PCB, add the PCB ready queue in the system area in the memory, and hand it to the CPU simulation module for execution.

The thread of CPU simulation module detects the ready queue every 1 second. After detecting, the information of PCB is loaded into each register of CPU simulation module to detect interrupt and interpret program. The detection contents are divided into two situations: when no device is used in the process, the instructions are executed in sequence; when the time slice arrives, the values in registers are saved to the PCB, and the PCB is added to the ready queue. If the time slice does not arrive and the end instruction is encountered, the execution result is transferred to the memory, and then written back to the out file of Disk C directory from the internal memory to destroy the PCB and memory; when the device is used in the process, every If the instruction of the device is checked, the values in registers will be saved to the PCB, and the PCB will be added to the waiting queue. The device manager will perform the read operation.

The simulation of interrupt is as follows: the interpretation and execution of an instruction will change the value of PSW (program status register) with the arrival of time slice and the appearance of end instruction, or the appearance of! B3 instruction. If there is no above situation, the value of PSW (program status register) will remain unchanged. During the instruction interpretation and execution, interrupt control can be carried out first, PSW is detected to make it respond, and then the instruction is interpreted to modify PSW. In this way, the CPU simulation module can run in executing instructions and responding to interrupts. Therun() instruction is as follows:

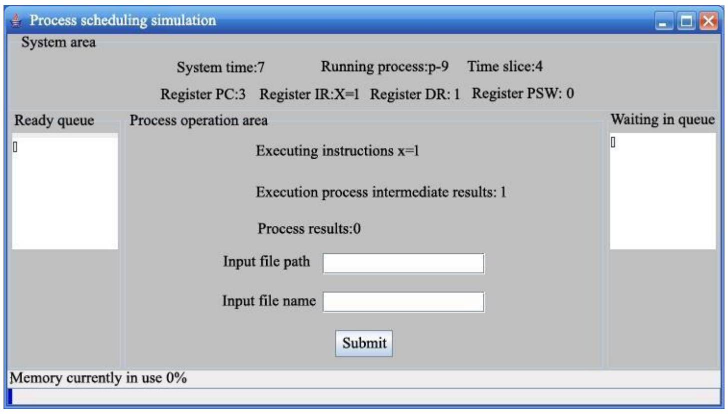
```
public void run(){                              // Driver of CPU execution
   while(a==1){
           try{
                   Thread.currentThread().sleep(1000);        // Thread sleep for one second
           }catch(InterruptedException e){}; interrupt();
                                        // Interrupt control
           dictite();                                              // Instruction interpretation
   }
}
```

The operation of the device simulation module first detects the PCB blocking queue in memory every 3 seconds. If the queue is not empty, the first PCB will be out of the queue, read its information, judge the device required by the process and the running time of the device, put it into the corresponding device waiting queue, and scan 5 device waiting queues at the same time.

If there is a process, execute it. After the execution, save the PCB and add it to the PCB ready queue in memory, otherwise continue scanning.

3. System Testing

After the coding of each class is completed, the corresponding visual interface needs to be designed for visual display. The system test part mainly takes the visualization interface display as an example. After entering the main interface, click the start menu and select process management to enter the following interface:



System time:7 Running process:p-9 Time slice:4
Register PC:3 Register IR:X=1 Register DR: 1 Register PSW: 0

Executing instructions x=1
Execution process intermediate results: 1
Process results:0

Figure 2. Process management interface

Table 5 shows the description of each part of the visualization interface.

Table 5. display description of process management visual interface

Interface display	Display content explanation	remarks
system time	display system time	
Running process	Displays the name of the running process	Now in the process of loitering x=1
Time slice	Displays the time of the time slice	
Register PC	Displays the subscript of the first address of the instruction in memory	
Register IR	Displays the instruction being executed	
Register DR	Intermediate result of instruction running	
Register PSW	Displays the label of the interrupt	
Ready queue	Displays the process name of the ready queue	Sequential display
Executing instructions	Displays the instruction being executed	
Execution process intermediate results	Intermediate result of instruction running	
Process results	Process results	
Input file path	Input file path, such as: C:	
Enter file name	Input file name, such as: ABC	
Click the OK button	Create process	
Waiting in queue	Displays the name of the process waiting for the queue	Sequential display
Memory currently used	*%Display scale, progress bar auxiliary display	

In the file view, you can see the directory tree structure of the folder, and open the handle to display the hierarchical results of the file directory. In the system disk usage box: the upper part of the box shows the size of the entire disk space of 8192b, of which the available space is 7232b, and the other space is used by the system. The middle part is the usage of five root directories. The usage percentage of each root directory is: Usage / total disk size 7232b. The percentage changes with the change of the file capacity in this directory. The following section shows the usage view of each disk block. There are 128 disk blocks, where 'U' indicates that the space of this disk block has been used, and 'n' indicates that the space of this disk block is empty.

In the user operation area of the interface, the following instructions can be executed: create file: create; edit file: edit; create directory: MKDIR; display file: type; copy file: copy; delete file: delete; delete empty directory: rmdir.

The source file path is defined as follows: the format is "C:" where C: is the root directory, ABC is the first level directory, and DSA is the second level directory. The name of each level directory is three bytes.

Name of file / directory (when creating a new directory): the name of the file / directory mainly written (when creating a new directory). The format is as follows: "ass", where ass is the file or name, and the name of each level of file / directory is three bytes.

Extension name (directory has no extension): the extension of executable file is e, the extension of text file is t, and the extension of directory is not. File / directory (when creating a new directory) attribute: there are two kinds of attributes: "read-only" and "read-write", which can be selected according to different situations. Target file path: it can only be used in copy file operation. It mainly writes to the target file. The usage is the same as the source file path. File name: only used in copy file operation. The name of the target file is mainly written. The usage is the same as the source file path.

4. Conclusion

In this paper, through the analysis of the structure of the computer operating system, the functional requirements design, system detailed design, system testing and so on are carried out. The process management, storage management, device management, file management and user interface are coded with Java programming language. A simulation single user operating system using multithreading is realized. The system can not only demonstrate the operation principle of specific functional modules, but also reflect the internal relationship of different functional modules, which is helpful to the system integrity Master the concept and principle of operating system, solve the problem of incoherent knowledge of each module and poor visualization effect in existing operating system learning [4]. The simulation system is limited to the number of existing modules is not enough, other interface functions in the operating system have not been reflected in the simulation system, the platform interface can be further beautified, combined with the current online learning needs, network, mobile terminal become the next step of the system work objectives.

References

[1] Sun Zengguo, Lu fanbi, Chen Junbiao, Yang Le. Development of operating system simulation experiment platform based on C# language [J]. Software Guide (Education Technology), 2016,15 (11): 85-88.

[2] Zhu Qi. Design of network teaching platform based on visual learning supervision [J]. Automation and instrumentation, 2019 (08): 29-32.

[3] Ma Honglin, Yan Lei, Yu Junwei. Exploration and practice of "learning centered" reform of operating system teaching paradigm [J]. Computer education, 2020 (06): 119-123.

[4] Wang Shuyan. Design of Linux experimental teaching system based on virtual simulation platform[J]. Science and technology information, 2018,16 (33): 5 + 7.