ISSN: 2637-6067 DOI: 10.6918/IJOSSER.202112_4(12).0033

Teaching Demonstration Software of Operating System Based on AnimeJs

Ping Wang, Kai Liu

School of Control and Computer Engineering, North China Electric Power University, Baoding, 071003, China

Abstract

Operating System is an important course for computer majors in colleges. The course have many abstract concepts, which are difficult for students to understand. To help the students in class, a teaching demonstration software,which can simulate the main functions of operating system, based on AnimeJs was designed and implemented. The whole system adopted the front-end technology stack. Layui, AnimeJs and other development frameworks were used to realize UI and animation. To simulate the underlying software and hardware resources of the operating system, a set of instruction primitive system was defined to realize a set of custom virtual machine. The practices suggests that on this platform, users can intuitively understand the basic mechanism of the operating system.

Keywords

Teaching Demonstration Software; AnimeJs; Operating System; Function Simulation.

1. Introduction

The importance of Operating System in computer education is beyond doubt. It is a basic compulsory course and plays a connecting role in the setting of computer software and hardware courses [1]. However, there are many abstract concepts in opertating system for beginners to understand. At present, the relatively mature platform of operating system curriculum teaching mainly concentrated in three aspects:the micro operating system,the modified Linux operating system and the simulation experiment system [2]. But, they aim at deepening the understanding of concepts, which has little help for the beginners. To help the beginners, visualization of the operating system is necessary. A teaching demonstration software,which can simulate the main functions of operating system visually, based on AnimeJs was designed and implemented in this paper to help the beginners.

2. Research Status

In some western countries, many different types of operating system teaching experimental systems have been developed for a long time. Some universities of project 985 and 211 in China have also developed the operating system experimental system for their own teaching [3]. As the most popular systems,MacOS and Windows systems are closed source,Linux was often used as a reference. For example, Li Fengjie implemented a Linux kernel programming environment by Java [4]. It can deepen students' understanding of knowledge, but the process is very difficult,because there were a large number of codes, APIs and usage specifications to read and understand.

Meng Jia useed Java for development and simulate the functions of typical operating system modules in detail by Swing of UI component library. Fubeilian abstracted the hardware such as memory, disk and CPU by Java data structure [5]. For each device, it proposed a series of API interfaces for external calls. Although the above two schemes simulated the typical functions of

ISSN: 2637-6067 DOI: 10.6918/IJOSSER.202112_4(12).0033

the operating system in detail, their visualization and interaction was limited. Lu yuhong's team put forward a scheme to visually demonstrate the process state transition in the operating system by using basic graphic queue[6]. Her team implemented this scheme by using C# and Visual Studio's window GUI tools. According to the state of all processes in the window, Multiple state queues were presented in a table. The producer-consumer problem is visualized in the article. However, the keys of this problem, synchronization and mutual exclusion of semaphore, were not mentioned.

The majority of current visualization and animation of operating system have focused on the execution of scheduling algorithm. There are more contents to demonstrate in teaching. So it is necessary to develop an animation demonstration demonstration system, which can simulate the main function of operating system.

3. Frame and Model Design

In order to provide easier access, the front-end technology of Web is selected as the overall technology stack, and AnimeJs is selected as the animation implementation solution within the platform. AnimeJs is a lightweight JavaScript animation library that supports a native front-end development environment and has a simple and powerful API for animating binding CSS properties, SVG, DOM and JavaScript objects.

In the process of displaying the overall operation of the system, it is necessary to show the management of system file, allocation and recycling of memory, and process scheduling. In the process of showing the operation of a certain process, it focus on the display of PCB and the changes of file open table and page table. The memory management of the system adopts the most common management mode of request paging storage by default. When executing commands, it focus on the process of address change. Based on the displayed contents, it determines that the overall framework of system consists of function library, component library, sub-module set, and user-defined virtual machine. Their dependence relationship is shown in Figure 1.

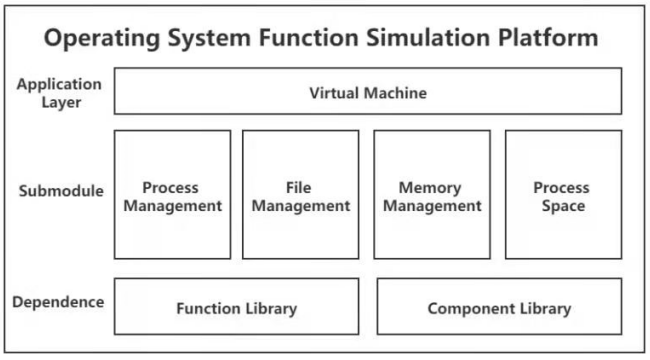

Figure 1. The framework of the simulation platform

Some general components and functions are encapsulated to form a function library and component library. The sub-modules are independent from each other, and each sub-module is an independent unit of state change. In other words, the animation will only happen the module inside, and the linkage between the two modules will not be generated. On the basis of integrating all modules, virtual machines at the application layer customize instructions and design interpretation mechanisms of instruction, translate instructions into operations and animation sequences, and finally deliver animation sequences to automata for execution one by one. Virtual machine consists of instruction queue, action queue and execution parsing

ISSN: 2637-6067 DOI: 10.6918/IJOSSER.202112_4(12).0033

mechanism. The modular design reduces the coupling degree of the system and enhances the scalability of the system.

When the command text is input into the system, for example, an instruction to start a process, the instruction interpretation mechanism of the VIRTUAL machine will process and parse the text of the input instruction at first, and parse the animation sequence that this instruction needs to be executed. If it is a high-level instruction, that is, the instruction can be further parsed into a complex sequence of sub-instructions. After being processed by the instruction interpretation mechanism of the VIRTUAL machine, the current instruction queue will be expanded. Then the virtual machine will parse, translate and execute the instruction queue from beginning to end. When all instructions in the instruction queue are parsed into the lowest level of actions that cannot be further parsed, the animations in the action queue will be executed in sequence. Each animation is designed as a state change within a submodule. When all animations have been executed, the execution process of the input text command ends. The program flow chart of the virtual machine level is shown in Figure 2.

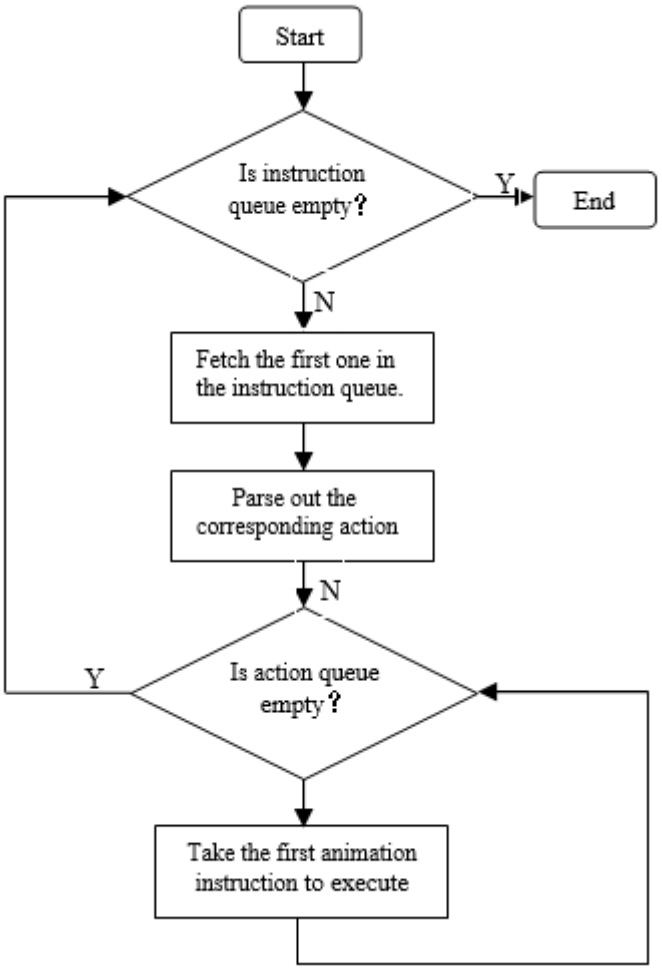

Figure 2. Flow chart of virtual machine

4. Implementation and Demonstration

4.1. Design and Parse of Instruction

The instruction is a string separated by space and is parsed according to the format "module identifier Action Parameter" .After being handed to the system, it is parsed through parseCmd() function at first. For example, in instruction " run mov 0102H 12H ", "run" indicates that the current action is in the execution of process, which is an animation in the module of "process

ISSN: 2637-6067 DOI: 10.6918/IJOSSER.202112_4(12).0033

space", "mov" means assigning a value to certain unit, "0102H" and "12H" are the parameters,which means to assign 02H to 0102H unit. To demonstrate the process of this action, further parse is our focal point. 0102H is a virtual address, which should be mapped to the memory first, the content of page table should be showed to demonstrate this process and then assign value, in which the content of memory should be showed to demonstrate.All the instructions are parsed according to the module distinction code and the action.

The simplest instruction, which is animation instruction to represent a specific location directly, will be parsed directly into an object that AnimeJs can recognizes. When the system recognizes that the current parsed instruction is a complex instruction, it calls the corresponding function, parses it into a new sub-instruction queue, puts it into the head of the whole instruction queue, and then takes the head instruction to perform the above steps.

4.2. Serialized Execution of an Animation

When a queue of AnimeJs animation objects is generated and executed by a body of circulation in the program, the animations will start simultaneously by frame one by one. When animation is interspersed with data modification, the inconsistency of data or the errors of sequencing often occur. The reason is the mechanism of its animation drawing. JS does not participate in drawing directly, instead, it gives the animation to the browser for rendering. When an animation is delivered to the browser by JS, the complete process of the animation is not completed, the body of circulation will directly execute the next loop.The callback function is used to solve the problem. AnimeJs provides several possible event interfaces for using in the lifecycle of animation action, such as begin, complete, update and etc. It set up a unified complete function for all AnimeJs animation objects. In this function, after the animation is completed, it will hand back the initiative to the system, which will execute the next instruction from the animation queue, thus the queue of animation action is executed in sequence.

4.3. Comprehensive Demonstration

Because the operating system has many functions, the operation of the system is a continuous process. Only part of the UI is shown in this part, as shown in Figure 3.

Figure 3. The Interface after Executing the Command

As shown in Figure 3, the two commands "System start go. exe" and "Run mov 0102H 12H" were executed in the test, which respectively represented to start the file "go. exe" in the root directory and set 12H to 0102H. According to the test, it can observe the sliding switch among modules, the addition and deletion of the system page table, the change of free linked list in the system, the change of the status code and the page table in the process space, the setting of data in the final memory page, and other actions. The order is consistent with the expectation. The

ISSN: 2637-6067 DOI: 10.6918/IJOSSER.202112_4(12).0033

system can meet the goal of animation simulation demonstration of typical functions of the operating system.

5. Conclusion and Prospect

Typical algorithms are demonstrated by visualization in the system. It can intuitively show switch of process status, address transform, file management, and other functions of the operating system by animation. Users can observe the implementation details of command. It concretizes the abstract concepts in operating system and makes the understanding easy for beginner. In the test, it can meet the requirements of simulation and simulation in the teaching process and improve the actual effect of teaching. More instructions will be added in the further.

Acknowledgments

This work was financially supported by Project of Research and Practice of Higher Education Teaching Reform in Hebei Province (2019GJJG410) and Project of Association of Fundamental Computing Education in Chinese Universities (2019-AFCEC-134).

References

[1] Pan Lei. Operating system syllabus. July 2010 http://www.njxzc.edu.cn/s/20/t241/0d/a9/info3497.html.

[2] Todd Peterson, Sean Crosby. Teaching OS design through implementation of a simulated operating system. Journal of Computing Science in Colleges, 2007,23(1):119~126.

[3] Yang Rui. Design and implementation of operating system simulation experiment system [J]. Information system engineering, 2013 (11): 36-37.

[4] Li Jiefeng. Design and implementation of operating system virtual experiment platform [D]. Central South University, 2010.

[5] Fu beilian. Research and implementation of operating system virtual support experimental platform [D]. Central South University, 2011.

[6] Lu Yuhong, Li Hao, Xing Lili, Liu Ying, Li Zhong. Visualization of operating system process management simulation [J]. Zhejiang: Science and Technology Bulletin, 2015, (12): 99-101.