SECTION A – EXAM PROJECT 1

DW WAIT - 28667735

Assignment 1 of 3.

Goals of the project.

The goal of the project was to write a web system that asks the user for input values, and then the systems goal was to change the value that is equal to the first input value the user inserted to the second value they inserted, as well as allowing multiple users to change values at the same time.

Issues of the project

The main problem was to get the system to allow multiple users to edit the file at the same time without causing any inconsistencies in the data. For example, if two users simultaneously open the web system and both changes the same word within the text file, then only the persons' data who was a few split seconds later in the process' data will be changed and the first persons' data will be lost. The problem here was to find a method that would allow multiple users to edit the file at the same time, without wasting the users time by having them wait until the other person has completed, by only locking the file once the user has submitted the update and the system then writes the data to the file.

Issues with locking: If a file is accessed from several different machines on the same time, then a lock that was obtained on one machine cannot be detected by another machine. If this happens, two or more users on different machines can lock the same file for their own use, and all the users think that their read/write processes are being done by only their machines, and then a race condition will occur.

Possible Solutions that might be used.

There are a few possible solutions that might be used to regulate the problem of data loss or data inconsistencies.

One way is to have a single writer, and many readers. Multiple users can access the same file, but everyone except the first person on the server has to be in read only. So only the unique process that can read, write and update, will serve as a parent process, and all other processes that wants to update or write to a file can only do so through interprocess communication with that one process.

Another option is to let all the users access the same file, and all the people will see the updates in near real time.

Another possible solution is to assign permissions or access rights to specific users and groups of users. This controls the user's ability to access, view, change, and execute the contents of a file. You can implement this with an access-control list (ACL). The ACL is just a list of permissions attached to an object. The ACL specifies which users or system processes are granted access to objects, and what operations are allowed to be executed.

File locking is another mechanism to prevent data inconsistencies. File locking restricts the access to a file by allowing only one process or user to access a specific file at a specific time. By implementing file locking you can prevent the classic intercepting update scenario by enforcing serialization of

update processes to any given file. The locks only works on a single machine, therefore the only way to deal with multiple users sharing a file system, is to have all users login to the same machine

Development on the system.

I first started with the HTML form of the system.

```
<form action="index.php" method="POST">
 <input type="text" name="name" />
 <input type="text" name="name2" />
 <input type="submit" value="Submit" />
</form>
```

There is nothing special about this form. It's a straight HTML form with no tags of any kind. The user fills in the two text boxes, and when they hit the submit button, the index.php page is called. The method attribute specifies how to send form-data. The form-data can be sent as HTTP post transaction with the method='POST' and the form-data can be sent as URL variables, with the method='get'.

I then started my php.

```
<?php
echo "Enter the string you wish to enter, then to what.";
$string_to_replace=$_POST['name'];
$replace_with=$_POST['name2'];
```

I used the echo() function to output a string value. Then I placed the two values from the text boxes into two variables with the POST method.

I then added a text file with the name Contents1 to my system.

After I added a text file and inserted a few values, I created a variable to contain the filename. Then you open the file with an fopen method, while also giving permission to write to the text file. $str variable saves the contents of the file into it with the implode method, and then the str_replace method contains the data about which set of strings to replace with another string value. Fwrite() then writes the data into the text field and the file is closed.

```
$filename="Content1.txt";
$str=implode("",file($filename));
$fp=fopen($filename,'w');
$str=str_replace($string_to_replace,$replace_with,$str);
fwrite($fp,$str,strlen($str));
fclose($fp);
?>
```

Then I started with the locks so that multiple users can't edit the file at the same time.

```php
$filename="Content1.txt";

$str=implode("",file($filename));
$fp=fopen($filename,'w');
if(flock($fp,LOCK_EX))
{
    $str=str_replace($string_to_replace,$replace_with,$str);
    fwrite($fp,$str,strlen($str));
    fflush($fp);
    flock($fp,LOCK_UN);
}
fclose($fp);
?>
```

I inserted an if statement to check if the file is locked or not, and if the file is not locked it will lock the file with flock($fp,LOCK_EX). The LOCK_EX is used to acquire an exclusve lock(writer)

The lock jumps in place after the file is opened, and then unlucked with LOCK_UN after the file was written or updated, and before the file is closed.

Conclusion.

With the use of locking, multiple users can access and write/update/change the information in the text file almost simultaneously, without causing any race problems.

SECTION B – EXAM PROJECT 2

Reynhard Taljaard - 24124753

ASSIGNMENT 2

SCHEDULING USING MULTIPLE QUEUES

WINDOWS SERVER VIRTUAL MACHINE

COMPLETED BY

REYNHARD TALJAARD

24124753

Introduction

The purpose of this assignment is to be able to develop an ASP.Net platform on a windows server virtual machine and implement various programming knowledge including C#, HTML and SQL. Combination of these languages and in depth knowledge of operation of OS scheduling algorithms, specifically multiple queue scheduling, will yield a working web platform that shows the workings of OS scheduling in a user friendly interface.

Scheduling Background

Definition

Scheduling in OS is the action of the process manager removing a running process from the CPU and selecting another process based on certain attributes determined by a scheduling algorithm. (greiss, 2018)

History

Originally described as critical path Analysis (CPA) or critical path management (CPM), scheduling was already thought of in the late 1950's but computer based scheduling was mainly reliant in manual scheduling techniques. It wasn't until the early 80's when the increasing spread of cheap and easy to use personal computers sparked the development of centralised scheduling systems including Primavera, TimeLine and CA Super project. (Weaver, 2014)

Workings of Scheduling

Processes are fundamentally organized in three different queues determined by the state of the process:

- Job queue
- Ready queue
- Device queue

The job queue holds all the processes of the system. The ready queue holds a set of processes that is in a ready state waiting to be executed. When a new process is implemented it is always put into the ready queue. The device queue holds all processes that are in a blocked state awaiting an I/O device.

The OS scheduler determines which process is pushed to which queue and implements different scheduling algorithms to handle each queue efficiently.

Schedulers can be divided into three types:

- Long-term
- Medium-term
- Short term

Long-term

This scheduler type is mainly implemented in the job queue because speed isn't as big a necessity and creates a balance between I/O and processor bound processes. Long-term is implemented when a process state ic changed from new to ready.

Medium-term

Medium-term scheduling is in charge of process swapping between memory and secondary storage.

Short-term

The short-term scheduling type is in charge of allocating processes to the CPU for execution and is responsible for better system performance. Thus speed plays a vital role in this scheduling type.

Project Goals

The goal of this project is to create a virtual machine that implements the Windows Server 2016 operating system. When the Virtual machine is created internet information services/server (IIS) needs to be installed and set up.

The purpose of this is to familiarise students with the set up and operation of servers and server operating systems.

Next a web system has to be developed using the desired programming platform (in this case ASP.Net) with a front-end user interface service (HTML) and a Back-end programming platform(C#).

The purpose of this is to improve self-knowledge of web platform development and ease of user interaction

The next step is to connect a SQL database to the web system for inserting and retrieval of valuable information needed for calculations and processes

The purpose of this is to be able to use databases effectively in web development because database connectivity is an essential part of web servers and web functionality.

Finally the web server has to be implemented on the Windows Server virtual machine

Development

Setting up virtual machine

For virtual machine setup Virtual Box and Windows Server 2016 was used

1. Virtual box was downloaded and installed on host computer
2. Windows Server 2016 was downloaded to host computer drive
3. BIOS/UEFI root settings were altered to enable virtualization
4. Virtual machine was created using virtual box
    - CPU cores were allocated
    - Desired memory and storage space was assigned
    - File sharing was enabled for later use

After the virtual machine was set up the installation and setting configuration for windows server was completed on the virtual machine.

For the IIS set up of windows server, server manager was used. New roles and features were downloaded and installed using a wizard provided by the server manager program.

Development of web system

For the web system development visual studio 2017 was used with ASP.Net platform included. A project was created with an ASP.Net core platform using models, views, controllers (MVC).controllers were used to handle the basic operation of processes between web pages and databases. Layout.schtml which implements both HTML and C# programming was used to set up the different pages and page names needed for the scheduling.

Multiple view classes (which also implement HTML and C#) were used to modify the user interface and provide pages with:

- Textboxes
- Drop boxes
- Checkboxes

- Sliders
- Buttons

For user input and:

- Forms
- Labels
- Text areas

For output.

Modules were implemented with various classes to handle data between controllers and the user interface.

Finally a bootstrap template was downloaded and added to the start-up source code for a cleaner user interface.

Conclusion

Scheduling has become an essential part of operating systems in our modern day times. From home computers to large mainframes, scheduling is a key element that cannot be disregarded where ever multiprocessing is implemented.

References

greiss, T. (2018). *Operating System - Process Scheduling*. Retrieved from www.tutorialspoint.com: https://www.tutorialspoint.com/operating_system/os_process_scheduling.htm

Weaver, P. (2014, August). *A Brief History of Scheduling*. Retrieved from mosaicprojects.com: https://mosaicprojects.com.au/Resources_Papers_042.html

SECTION C: EXAM PROJECT 3 REPORT

A P Barnard - 28430093

TABLE OF CONTENTS

INTRODUCTION

Background

Each member of the group was tasked with using their skills in Operating System, specifically, File Systems, Scheduling and Multiple Queues, and Virtual Memory Systems. We had to implement these solutions on Linux and Windows Server Virtual Machines (VM).

What are virtual machines? Discussed in a previous report and analysed below:

Virtual Memory is a concept that was first created in 1961 by J. Fotheringham (Fotheringham, 1961). The idea behind this concept is that every program on a computer has its own address space and that is broken up into smaller chunks called pages. The reason for the chunking of pages is that larger programs are basically too large for the computer to store and process and execute.

Pages are mapped onto physical memory spaces, since pages are made up of contiguous address ranges.

Most virtual memory systems use the concept of Paging. Every program on a computer references a set of memory addresses that executes instructions by copying contents of the memory address that has been indexed using a base and segment register. These program-generated addresses are called virtual addresses and are part of the virtual address space. Virtual addresses are maintained and created by the MMU (Memory Management Unit) (Tanenbaum & Bos, 2015)

Problem Statement

For Section C, I had to implement the concepts of Virtual Memory and Paging on either a Linux or Windows VM.

TASK:

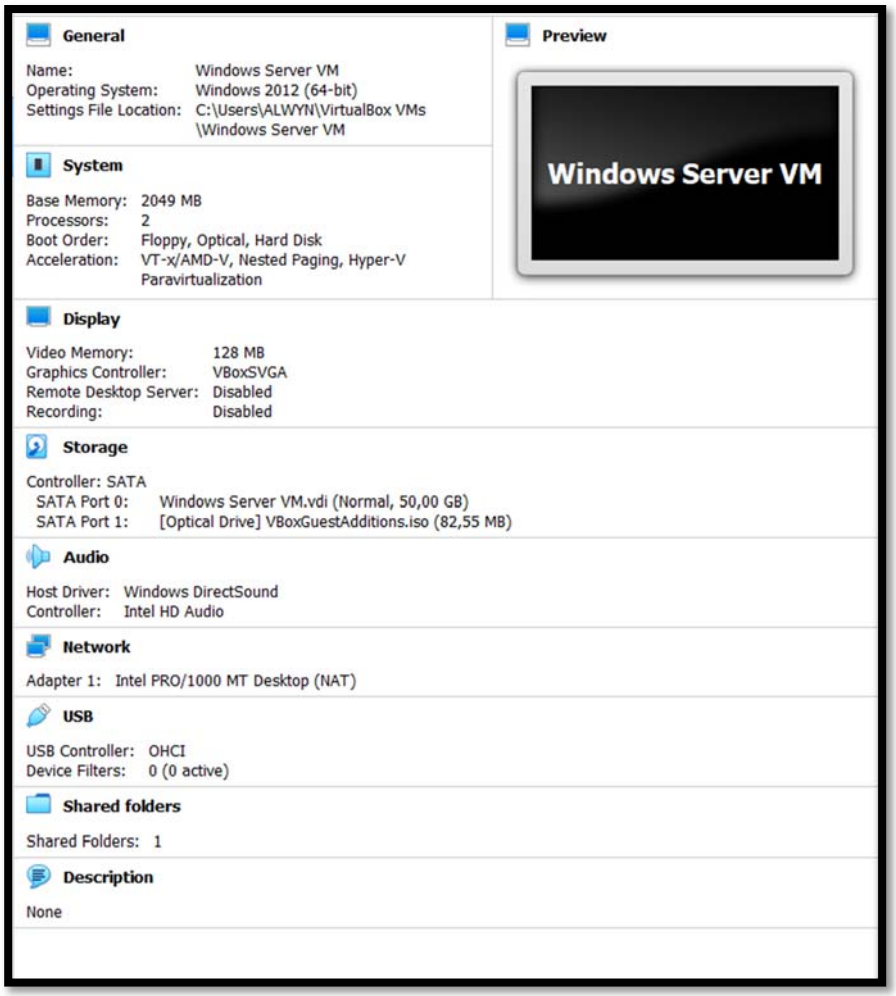
1. Retrieve the amount of memory available on the server.
2. Ask the user for the amount of memory that has to be reserved for the Operating System (OS).
3. Ask the user for the amount of memory that has to be assigned to each Page Frame.
4. Calculate the number of Page Frames that can be created.
5. Simulate Paging by randomly assigning unique page numbers into the page frames.
6. Ask the user for a page to search for.
7. Show if the page is in Memory, or if it causes a page fault.
8. Use a page replacement algorithm to load the page not found into a frame.
9. Deploy the solution on your Virtual Machine.
10. Construct a report on the Project.

CONTENT

The following Virtual Machine was created:

- VirtualBox was used to create the VM

- Windows Server 2012 Virtual Machine with IIS
- Base Memory of 2048 MB
- 2 Processors Used with 100% Execution cap
- Acceleration: VT-x/AMD-V and Nested paging enabled
- 128 MB Video Memory with 2D and 3D Video Acceleration
- Shared folder between Host OS and Guest OS



PROGRAMMING LANGUAGE:

- Visual Studio 2019
- ASP.NET Web Forms Application with additional Web Forms

INPUT COMPONENTS:

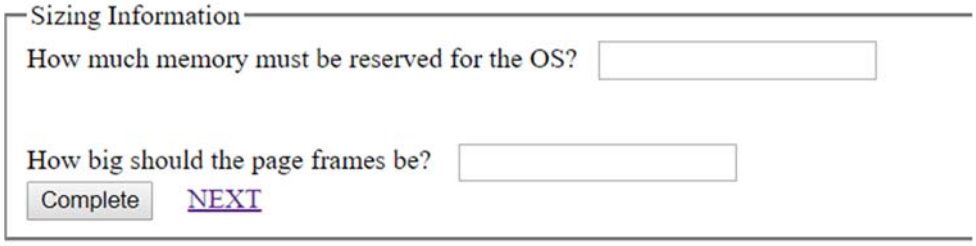
- Textboxes, Buttons and Labels

OUTPUT COMPONENTS:

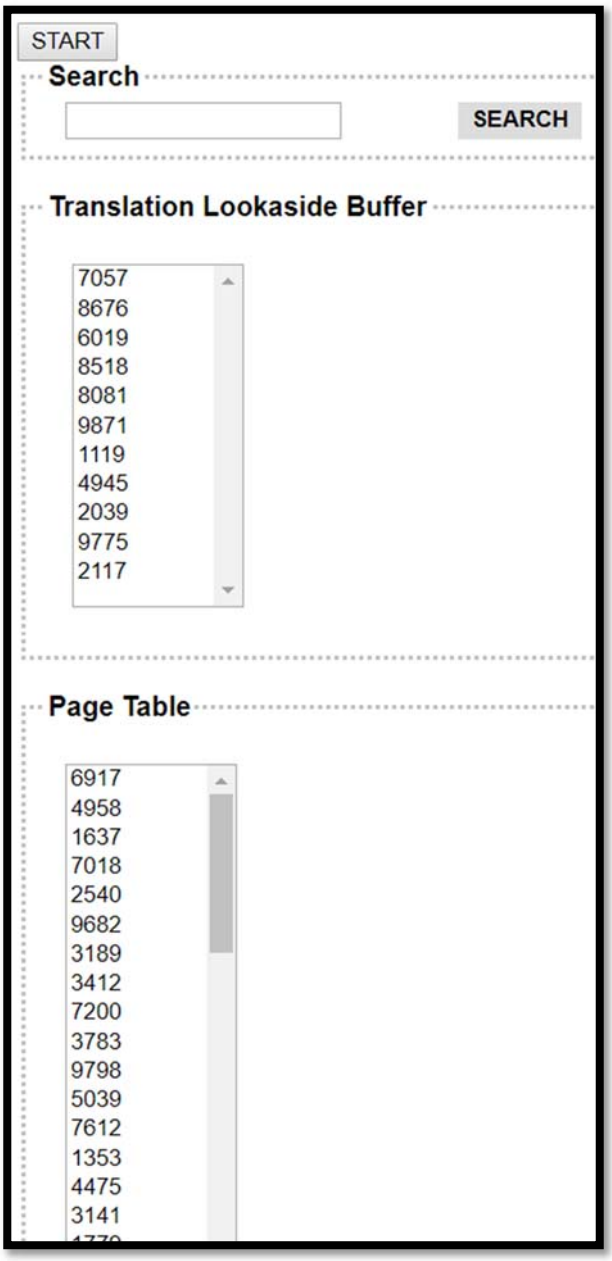
- Textboxes, Labels, Panels and Buttons
- Command Line results

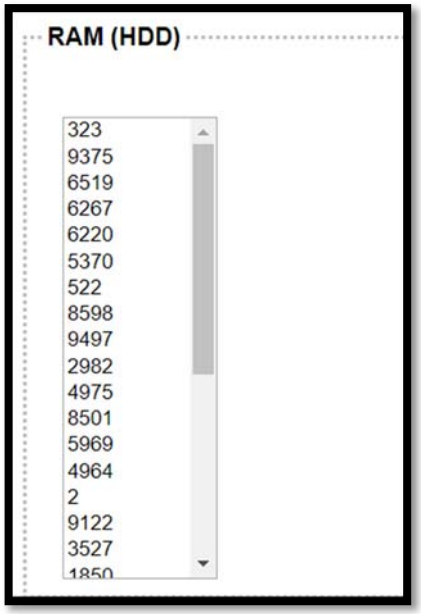
Forms:

MAINFORM



PAGING FORM





RESULTS

The results that can be seen on the above forms are as follows. If incorrect input is received the Validators will show that there are incorrect values. Otherwise the program will look for a value that is either in the TLB, Page Table or in RAM. If no such value is found, then an error will be shown that it does not exist in memory.

Every possible precaution was taken to ensure data validity and where invalid data was entered the user will be prompted to change those values.

CONCLUSION

Virtual memory plays an integral role in Operating Systems. It is the foundation that was used to build virtual address spaces and the accompanying concepts. With it we would not be able to run multiple Operating Systems on one computer. Limitations to virtual memory are “virtually” impossible.

REFERENCES

Fotheringham, J.: ‘‘Dynamic Storage Allocation in the Atlas Including an Automatic Use of a Backing Store,’’ Commun. of the ACM, vol. 4, pp. 435–436, Oct. 1961.

Tanenbaum, A. S. & Bos, H. 2015. Modern operating systems. 4th ed. Upper Saddle River, NJ: Pearson Prentice Hall.

.