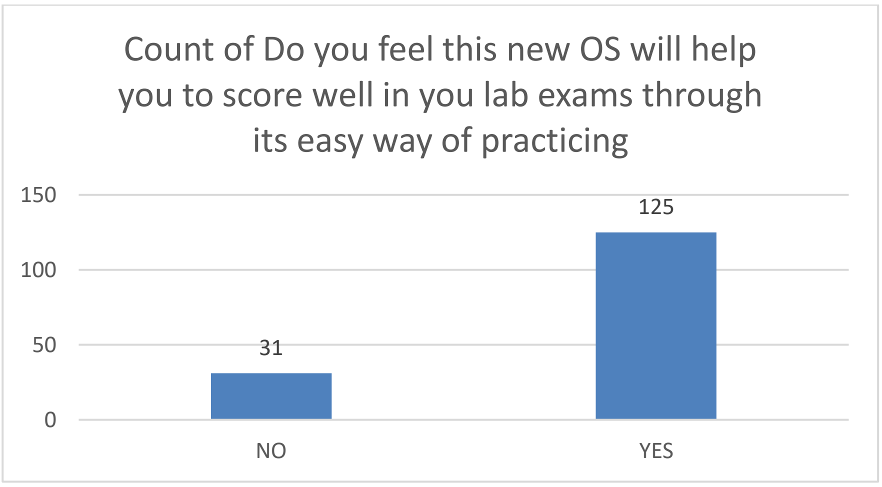
1. INTRODUCTION

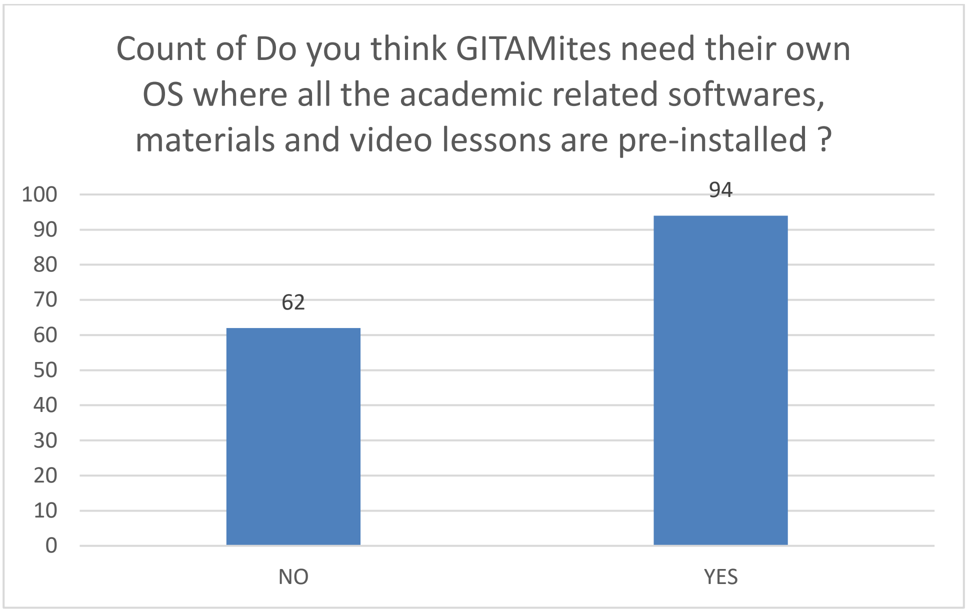
1.1 Motivation

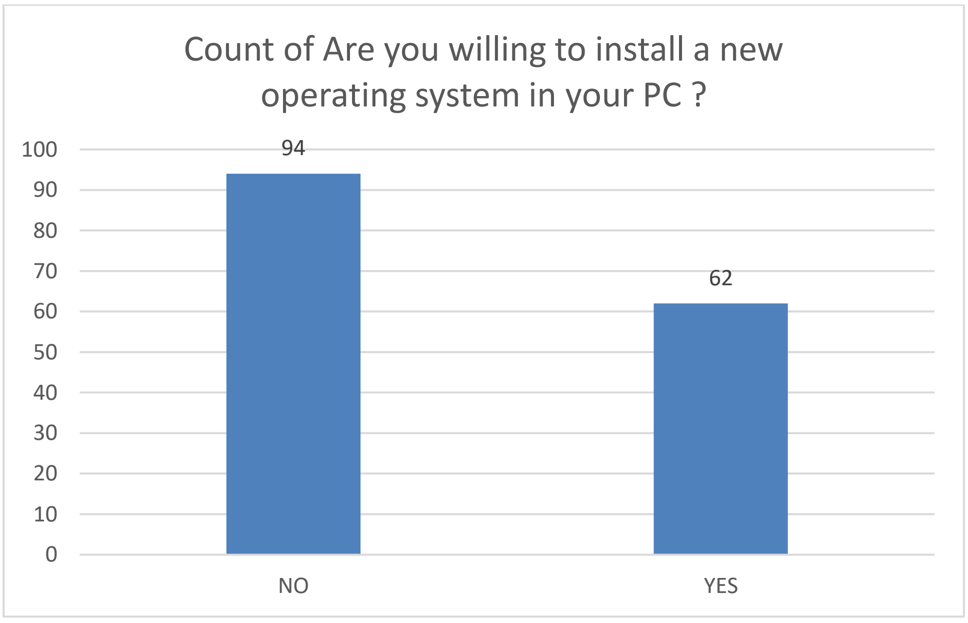
As a student's of computer science engineering it has always been an interest in this field since interest in computing began. As our own experiences of using different operating systems for course-work has led us to ponder many questions as to why there are no such operating systems which include the engineering tools. This project has given us the opportunity to explore the world of operating system to find out the answers to our questions.

The GITOS project aims at the development of an operating system which includes the built-in engineering tools that an engineering student has to deal with during his course time. The idea is that given a system specification, by following the methodology and with the help of the tools integrated to support it, the user will be able to synthesize a system that meets his constraints. The operating system will be having a lucrative graphical user interface which will in turn give a better user experience.

We also took a survey in the campus from our fellow college mates through which we got the parameters and support of creating GITOS with a hope through the survey that it will be very useful for all the fellow students not only for our college students but for other college students. It will be an initiation for make ease of learning for the students which in turn boost lots of scope in further development of the OS.




Count of Do you think GITAMites need their own OS where all the academic related softwares, materials and video lessons are pre-installed ?
100
90
80
70
60
50
40
30
20
10
0
62
94
NO
YES


Count of Are you willing to install a new operating system in your PC ?
100
90
80
70
60
50
40
30
20
10
0
94
62
NO
YES

1.2 Problem Definition

We have many existing operating systems which provide with similar user environment but fail to engage by a student centric application operating system.

Presently the engineering institutions are engaged with proprietary operating system which turns to be very costly and some universities uses an existing general Linux operating system and then investing a lot of time for installing the required applications for student's curriculum which is being a hefty work.

GITOS is an operating system completely for the engineers by the engineers and to the engineers. The Purpose of GITOS to fulfill the needs and requirements of the modern engineering students, it has all the FOSS and Proprietary Licensed Academic software and tools which are more worth for an engineering student irrespective of their branch.

3. Objective of Project

GITOS is an operating system completely for the engineers by the engineers and to the engineers. The Purpose of GITOS to fulfill the needs and requirements of the modern engineering students, it has all the FOSS and Proprietary Licensed Academic software and tools which are more worth for an engineering student irrespective of their branch.

4. Limitations of Project

As GITOS is a Linux based operating system, Linux does not dominate the market like Windows; there are some disadvantages to using the operating system. First, it's more difficult to find applications to support your needs. This is an issue for mostly businesses, but more programmers are developing applications that are supported by Linux. Many more applications are available for the working world compared to what was available a decade ago.

One main issue with Linux is drivers. Before you can install any hardware component in your computer, you must make sure the hardware has drivers available. Hardware manufacturers usually write drivers for Windows, but not all brands write drivers for Linux. This means that some of your hardware might not be compatible with Linux if you decide to switch.

2. LITERATURE SURVEY

2.1 Introduction

GITOS is an operating system completely for the engineers by the engineers and to the engineers. The Purpose of GITOS to fulfill the needs and requirements of the modern engineering students, it has all the FOSS and Proprietary Licensed Academic software and tools which are more worth for an engineering student irrespective of their branch.

GITOS a modern, elegant and comfortable operating system which is both powerful and easy to use.

Some of the reasons for the success of GITOS are:

- It works out of the box, with fully education tools, multimedia support and is extremely easy to use.
- It's both free of cost and open source.
- It's community-driven, Engineering students from different colleges with their ideas can be used to improve GITOS.
- Based on Debian and Ubuntu, it provides about 30,000 packages including all engineering packages and one of the best software managers to install software and tools.
- It's safe and reliable. Thanks to conservative approach to software updates, a unique Update manger and the robustness of its Linux architecture, GITOS requires very little maintenance (no regressions, no antivirus, no anti-spyware, no anti-malware ... etc.). Since your privacy is really important to us.

Our group is working on localizing the GNU/Linux distribution. Inspired by the Free Software philosophy we have come up with the idea of localizing the operating system. We believe that the technology which is not in the reach of a common engineering student is worthless. In order to fill this gap we thought of localizing the operating system and started porting it in engineering colleges.

2.2 Existing System

An operating system (OS) is system software that manages computer hardware and software resources and provides common services for computer programs. Currently we have many existing operating systems which provides variety of user interfaces and environments but fails to satisfy with present engineering student's personal specification needs.

Presently the engineering institutions are engaged with proprietary operating system which turns to be very costly and some universities uses an existing general Linux operating system and then investing a lot of time for installing the required applications for student's curriculum which is being a hefty work.

An engineering student oriented operating system is still not designed with all specifications required or if they have, it might be failed to attract more number of modern engineering student.

Proprietary Operating System like Microsoft Windows and Apple MAC OS are the most used operating system currently but fails to be imbibed in all institutions as it is not a budget friendly operating system and is having lots of bugs which usually hangs and shutdown and encounters n number of problems which can be driver related to. These are more dependent on updates which are always time taking.

2.3 Disadvantages of Existing System

So, what could go wrong with existing Operating Systems? Let's take a look at five possible stumbling blocks:

1) Drivers: A lot of the time, in an operating system from different organization, with just one driver can bring an entire PC down to its knees, and unless you know what you're doing, finding that dodgy driver can be next to impossible.

2) Economy: People are spending less (well, on everything expect Apple products, it would seem). Windows could very well be the best OS ever, but unless people are willing to flip open their wallets or unsnap their purses and buy new PCs with

Windows on them – or purchase an upgrade copy of the OS – Microsoft is once again stuck with an OS it can’t sell.

3) Trust : Operating Systems in the market are enabling the user to compromise with their data and privacy and user here is more vulnerable, whose personal data can be misled and which it turns to be a very costly affair to the user,

4) End user resistance : The fact that over 70 percent of users still run Windows XP is an excellent indication as to just how entrenched users are. But most of them are unaware of the capabilities of the Linux based operating system which are safer, more efficient and less vulnerable to data leak.

5) Value for money: Does Windows 7 offer value for money? Sure, there have been numerous deals available where people can pick up a cheap upgrade in the months running up to the GA date, but this is a tactic to sell something to people before they really know what they are getting.

2.4 Proposed System

GITOS is an operating system completely for the engineers by the engineers and to the engineers. The Purpose of GITOS to fulfill the needs and requirements of the modern engineering students, it has all the FOSS and Proprietary Licensed Academic software and tools which are more worth for an engineering student irrespective of their branch.

GITOS a modern, elegant and comfortable operating system which is both powerful and easy to use.

Some of the reasons for the success of GITOS are:

- It works out of the box, with fully education tools, multimedia support and is extremely easy to use.
- It's both free of cost and open source.
- It's community-driven, Engineering students from different colleges with their ideas can be used to improve GITOS.

- Based on Debian and Ubuntu, it provides about 30,000 packages including all engineering packages and one of the best software managers to install software and tools.
- It's safe and reliable. Thanks to conservative approach to software updates, a unique Update manger and the robustness of its Linux architecture, GITOS requires very little maintenance (no regressions, no antivirus, no anti-spyware, no anti-malware ... etc.). Since your privacy is really important to us.

Our group is working on localizing the GNU/Linux distribution. Inspired by the Free Software philosophy we have come up with the idea of localizing the operating system. We believe that the technology which is not in the reach of a common engineering student is worthless. In order to fill this gap we thought of localizing the operating system and started porting it in engineering colleges.

2.5 Conclusion

As expected the designed Operating System will be more efficient and have tremendous graphical user interface which can attract large number of engineering students which will lead to the respective institutions to adapt to it easily. It can be implemented in a large scale and its availability can reduce the hefty work behind the installation of each application by the lab technicians.

3. ANALYSIS

3.1 Introduction

GITOS is an operating system completely for the engineers by the engineers and to the engineers. The Purpose of GITOS to fulfill the needs and requirements of the modern engineering students, it has all the FOSS and Proprietary Licensed Academic software and tools which are more worth for an engineering student irrespective of their branch.

GITOS a modern, elegant and comfortable operating system which is both powerful and easy to use.

3.2 Software Requirement Specification

As other Operating System it does require some specification as follows:

3.2.1 User Requirement

The GITOS project aims at the development of an operating system which includes the built-in engineering tools that an engineering student has to deal with during his course time. The idea is that given a system specification, by following the methodology and with the help of the tools integrated to support it, the user will be able to synthesize a system that meets his constraints. The operating system will be having a lucrative graphical user interface which will in turn give a better user experience. It works out of the box, with fully education tools, multimedia support and is extremely easy to use. It’s both free of cost and open source. It’s community-driven, Engineering students from different colleges with their ideas can be used to improve GITOS.

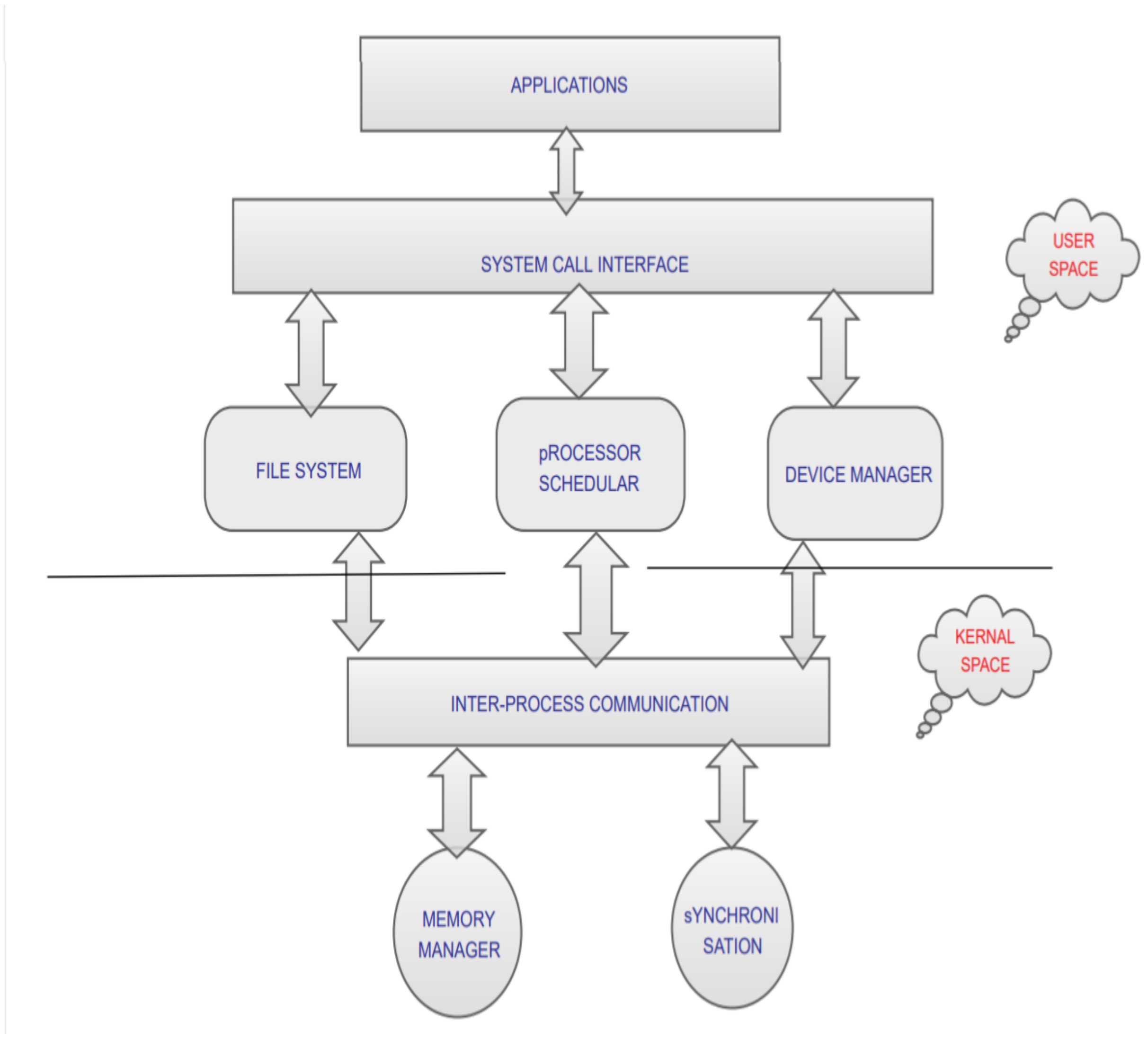
3.2.2 Software Requirement

Base Operating System	: Windows (or) Linux
Virtual Machine	

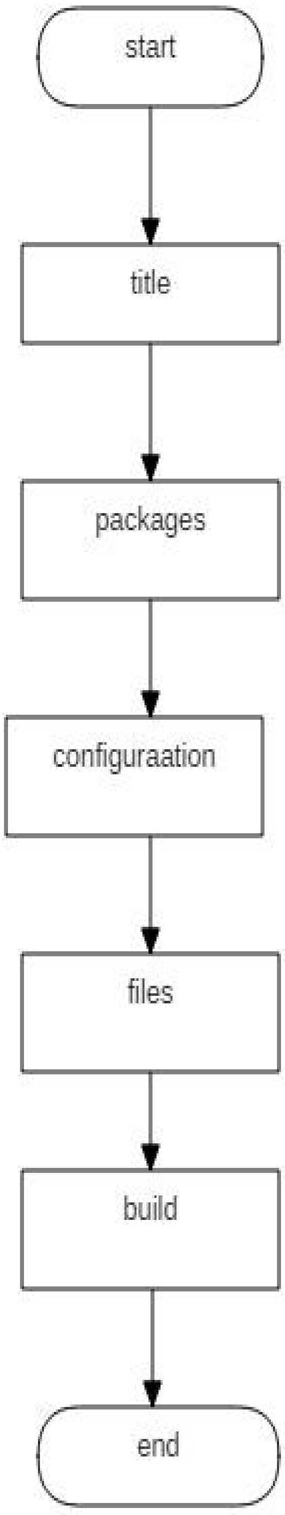
3.2.3 Hardware Requirement (Min. Requirements)

Hardware	:Dual Core
Speed	:2 GHz
RAM	:2 GB
Hard Disk	:25 GB
VGA	:1024x768
Drive/Port	:CD or USB

3.3 Content diagram of Project



3.4 Algorithms & Flowcharts



Cpu Scheduling involves both I/O time and CPU time. In a uniprogramming system like MS-DOS, time spent waiting for I/O is wasted and CPU is free during this time. In multiprogramming systems, one process can use CPU while another is waiting for I/O. This is possible only with process scheduling.

Objectives of Process Scheduling Algorithm

Max CPU utilization [Keep CPU as busy as possible]

Fair allocation of CPU.

Max throughput [Number of processes that complete their execution per time unit]

Min turnaround time [Time taken by a process to finish execution]

Min waiting time [Time a process waits in ready queue]

Min response time [Time when a process produces first response]

Round-robin

Round-robin (RR) is one of the algorithms employed by process and network schedulers in computing. As the term is generally used, time slices (also known as time quanta) are assigned to each process in equal portions and in circular order, handling all processes without priority (also known as cyclic executive).

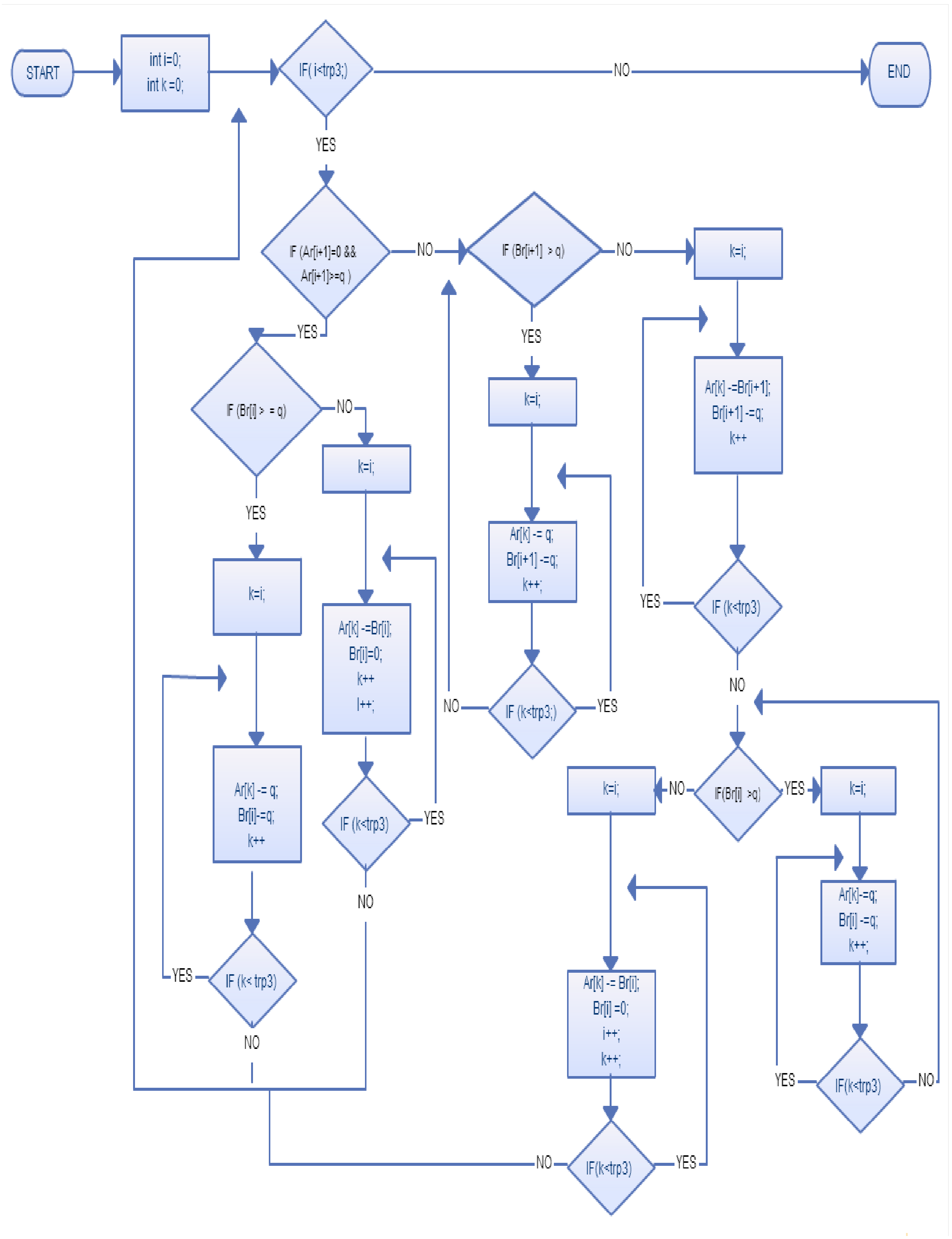
1- Create an array rem_bt[] to keep track of remaining burst time of processes. This array is initially a copy of bt[] (burst times array)

2- Create another array wt[] to store waiting times of processes. Initialize this array as 0.

3- Initialize time : t = 0

4- Keep traversing the all processes while all processes are not done. Do following for i'th process if it is not done yet.

```
a- If rem_bt[i] > quantum
  (i)  t = t + quantum
  (ii) bt_rem[i] -= quantum;
c- Else // Last cycle for this process
  (i)  t = t + bt_rem[i];
  (ii) wt[i] = t - bt[i]
  (ii) bt_rem[i] = 0; // This process is over
```



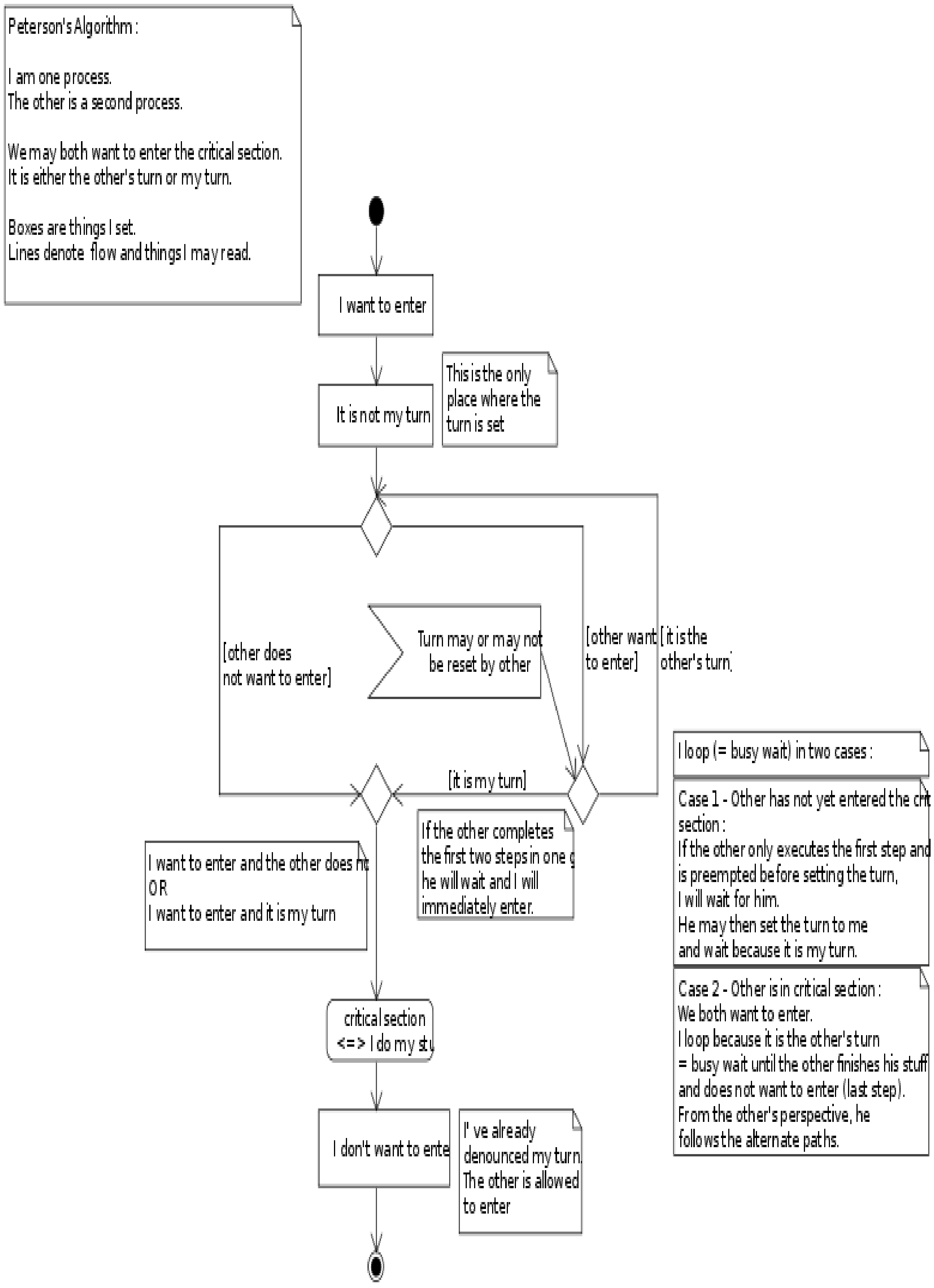
Round Robin Algorithm

Peterson's algorithm

Peterson's algorithm is a programming algorithm for mutual exclusion that allows two or more processes to share a single-use resource without conflict, using only shared memory for communication.

The algorithm uses two variables, flag and turn. A flag[n] value of true indicates that the process n wants to enter the critical section. Entrance to the critical section is granted for process P0 if P1 does not want to enter its critical section or if P1 has given priority to P0 by setting turn to 0.

```
P0 flag[0]=true;

P0_gate:turn=1;

while(flag[1] == true && turn == 1)

        {

         //busywait

        }

        //criticalsection

        ...

        // end of critical section

        flag[0] = false;

P1:flag[1]=true;

P1_gate:turn=0;

        while (flag[0] == true && turn == 0)

        {

           //busywait
        }

        //criticalsection

        ...

        //endofcriticalsection

        flag[1] = false;
```


Peterson's Algorithm :
I am one process.
The other is a second process.
We may both want to enter the critical section.
It is either the other's turn or my turn.
Boxes are things I set.
Lines denote flow and things I may read.
I want to enter
It is not my turn
This is the only place where the turn is set
Turn may or may not be reset by other
[other does not want to enter]
[other want to enter]
[it is the other's turn]
[it is my turn]
I loop (= busy wait) in two cases :
Case 1 - Other has not yet entered the crit section :
If the other only executes the first step and is preempted before setting the turn,
I will wait for him.
He may then set the turn to me and wait because it is my turn.
Case 2 - Other is in critical section :
We both want to enter.
I loop because it is the other's turn
= busy wait until the other finishes his stuff and does not want to enter (last step).
From the other's perspective, he follows the alternate paths.
If the other completes the first two steps in one g he will wait and I will immediately enter.
I want to enter and the other does nc
OR
I want to enter and it is my turn
critical section
<=> I do my stu
I don't want to ente
I' ve already denounced my turn. The other is allowed to enter

Peterson's algorithm

3.5 Conclusion

As expected the designed Operating System will be more efficient and have tremendous graphical user interface which can attract large number of engineering students which will lead to the respective institutions to adapt to it easily. It can be implemented in a large scale and its availability can reduce the hefty work behind the installation of each application by the lab technicians.

4. DESIGN

4.1 Introduction

GITOS is an operating system completely for the engineers by the engineers and to the engineers. The Purpose of GITOS to fulfill the needs and requirements of the modern engineering students, it has all the FOSS and Proprietary Licensed Academic software and tools which are more worth for an engineering student irrespective of their branch.

GITOS a modern, elegant and comfortable operating system which is both powerful and easy to use.

4.2 UML Diagrams

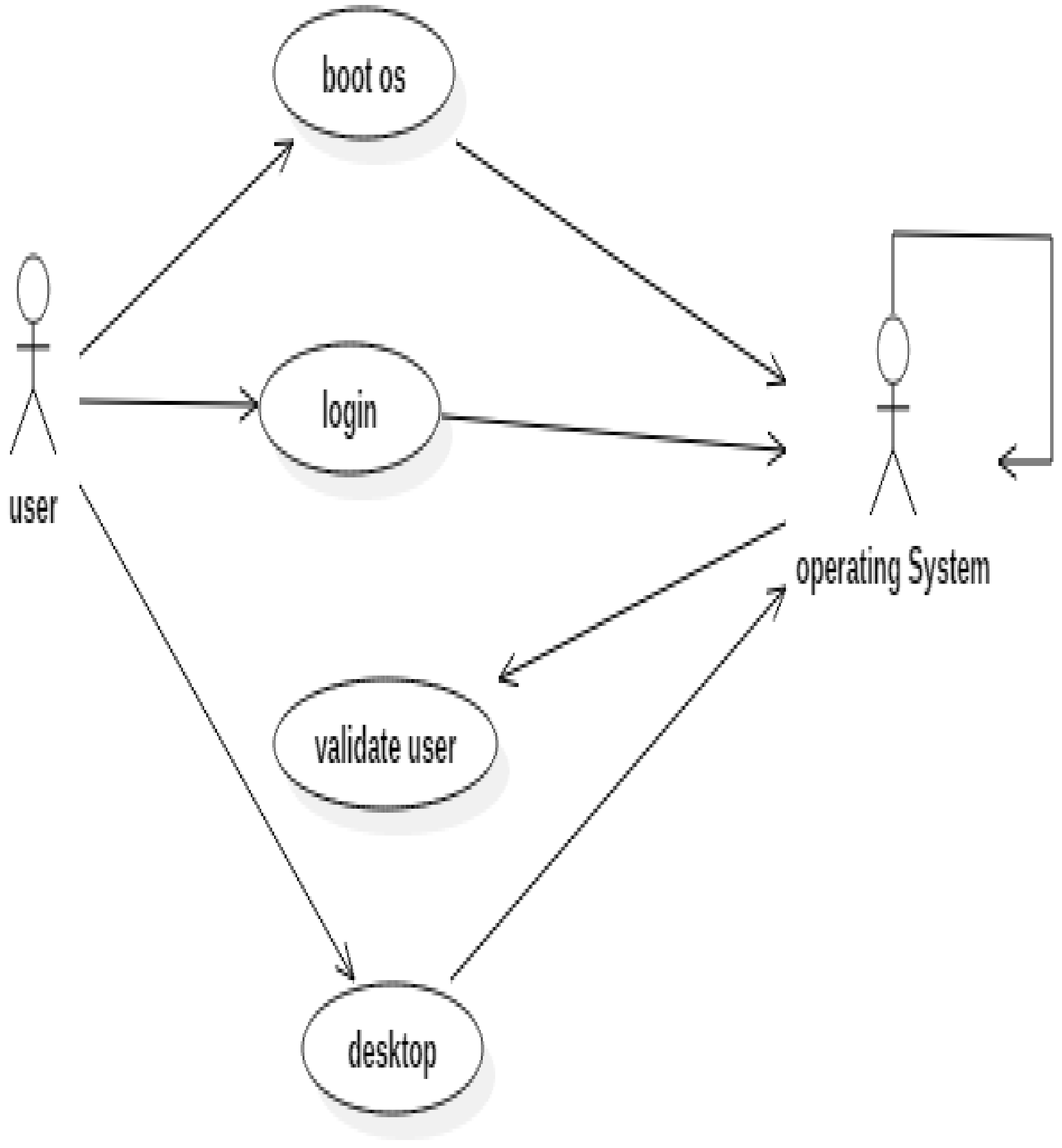
UML is a way of visualizing a software program using a collection of diagrams. The notation has evolved from the work of Grady Booch, James Rumbaugh, Ivar Jacobson, and the Rational Software Corporation to be used for object-oriented design, but it has since been extended to cover a wider variety of software engineering projects. Today, UML is accepted by the Object Management Group (OMG) as the standard for modeling software development.

The current UML standards call for 13 different types of diagrams: class, activity, object, use case, sequence, package, state, component, communication, composite structure, interaction overview, timing, and deployment.

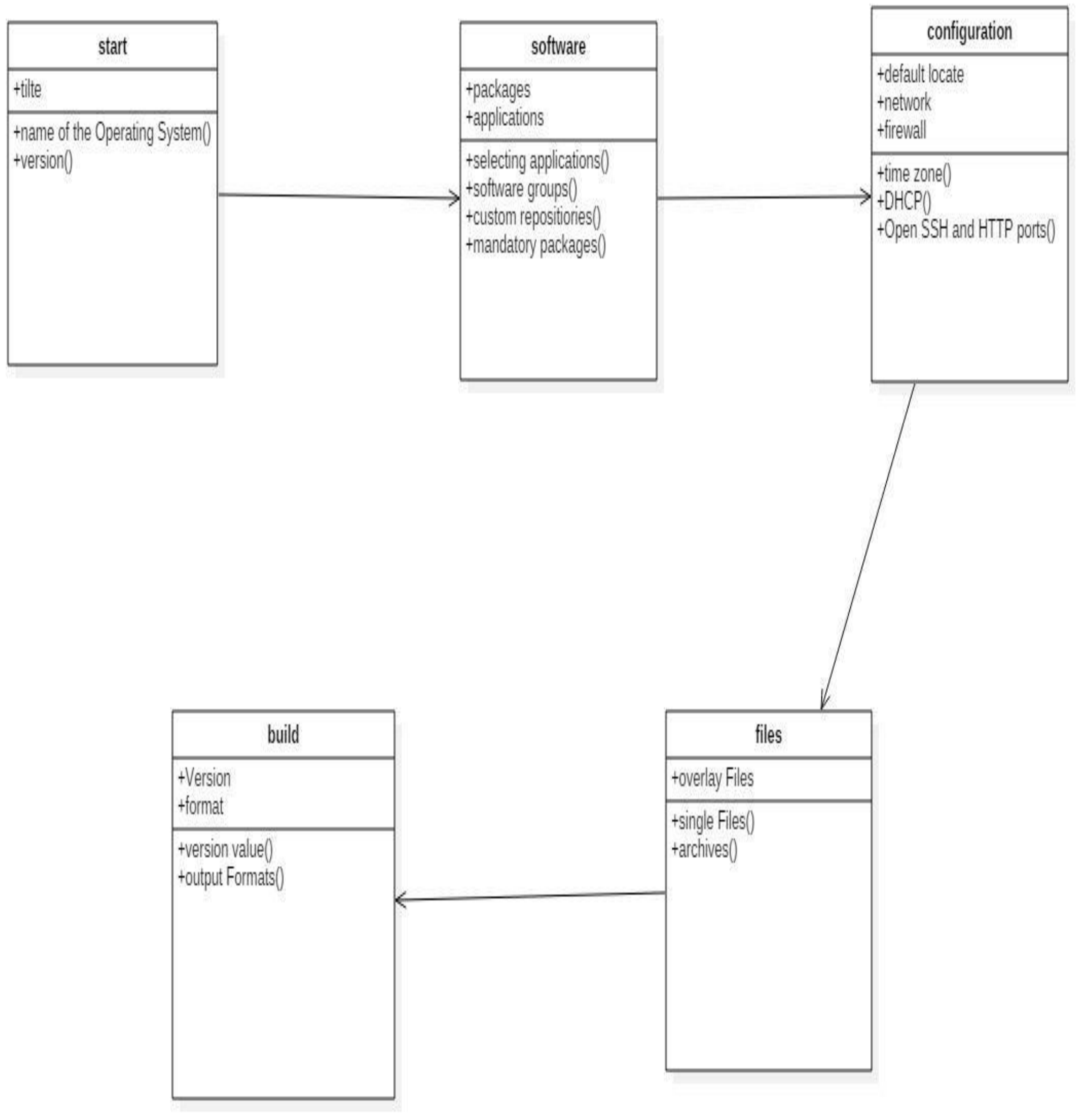
These diagrams are organized into two distinct groups: structural diagrams and behavioural or interaction diagrams.

- Class diagram
- Package diagram
- Object diagram
- Component diagram
- Composite structure diagram
- Deployment diagram

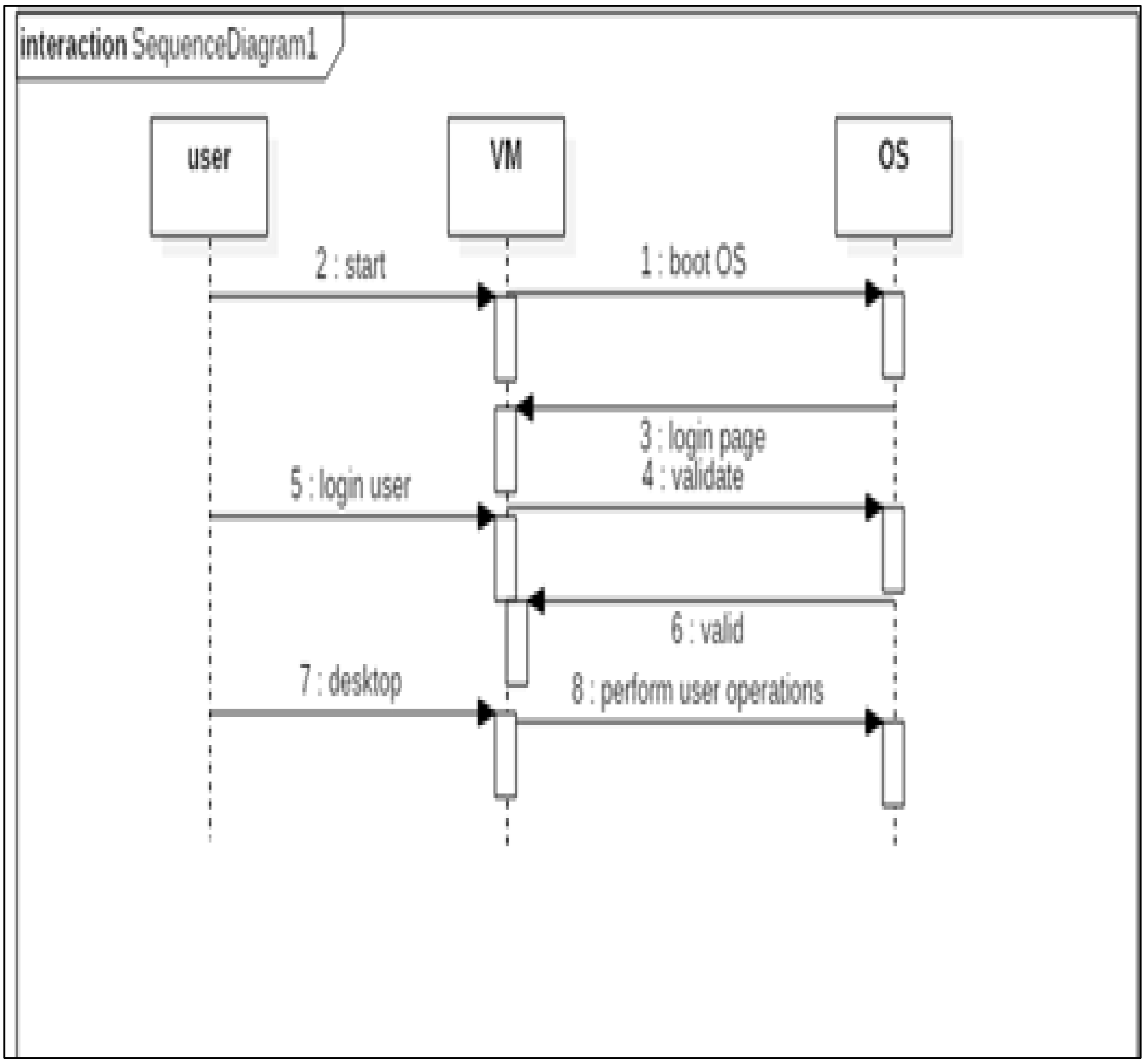
- **Use Case Diagram:**



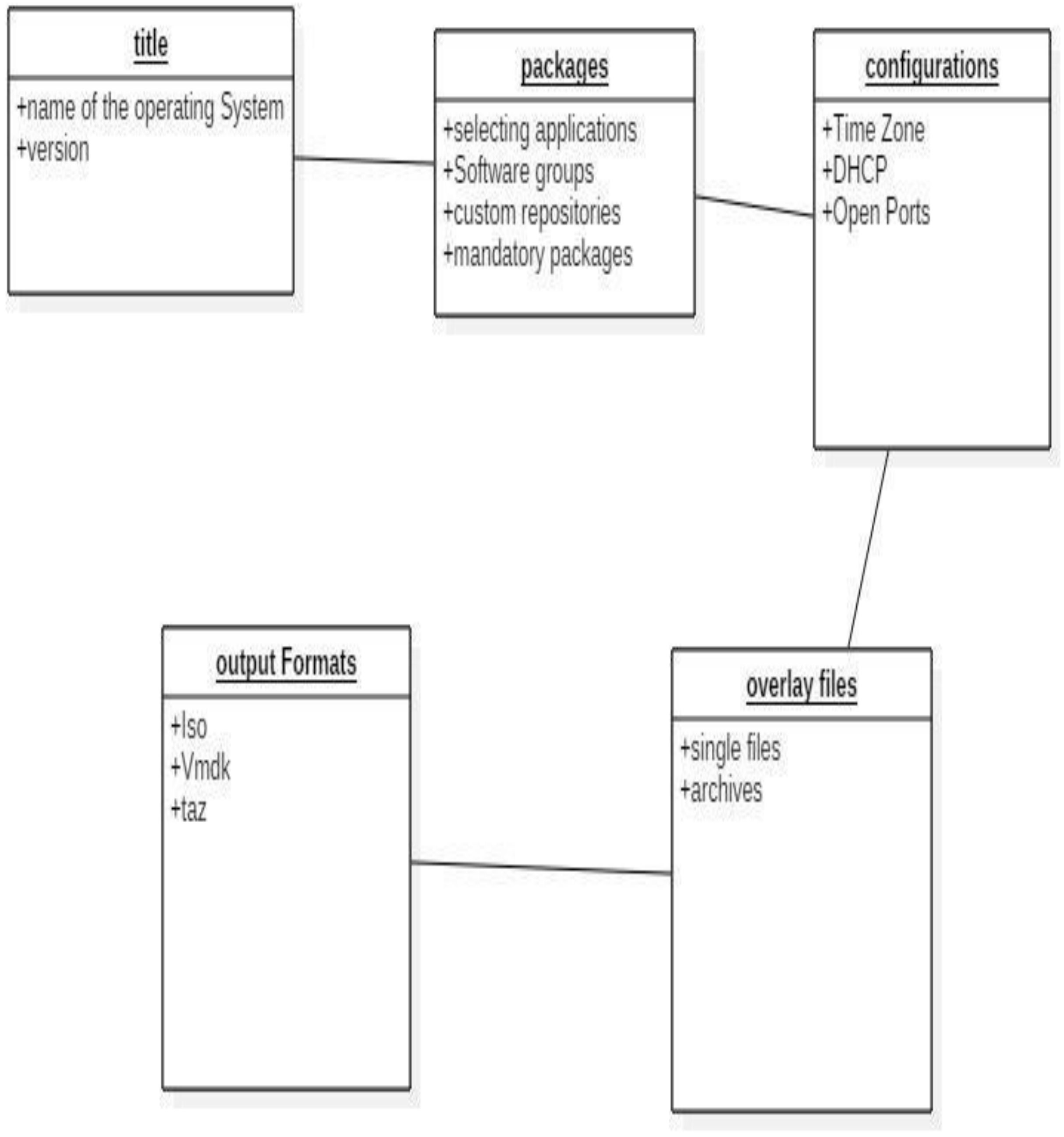
Class Diagram:



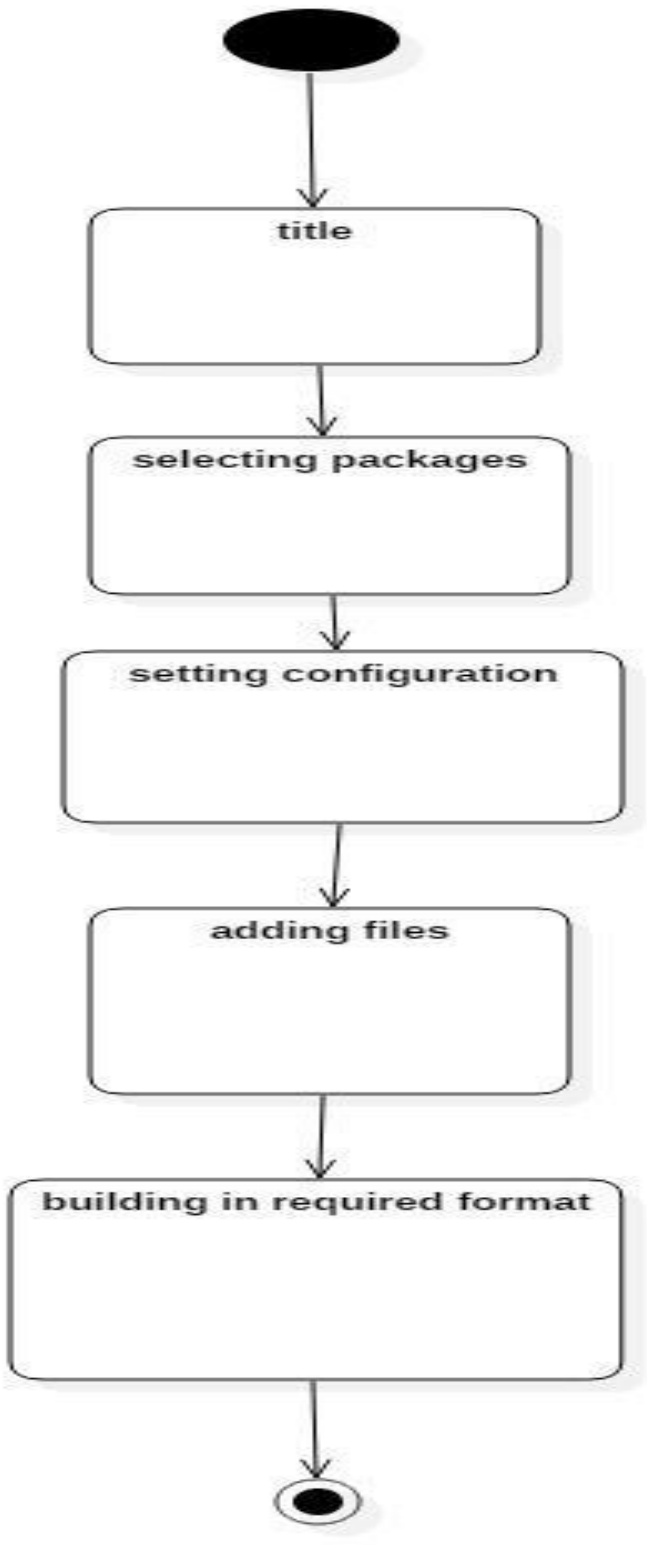
Sequence Diagram



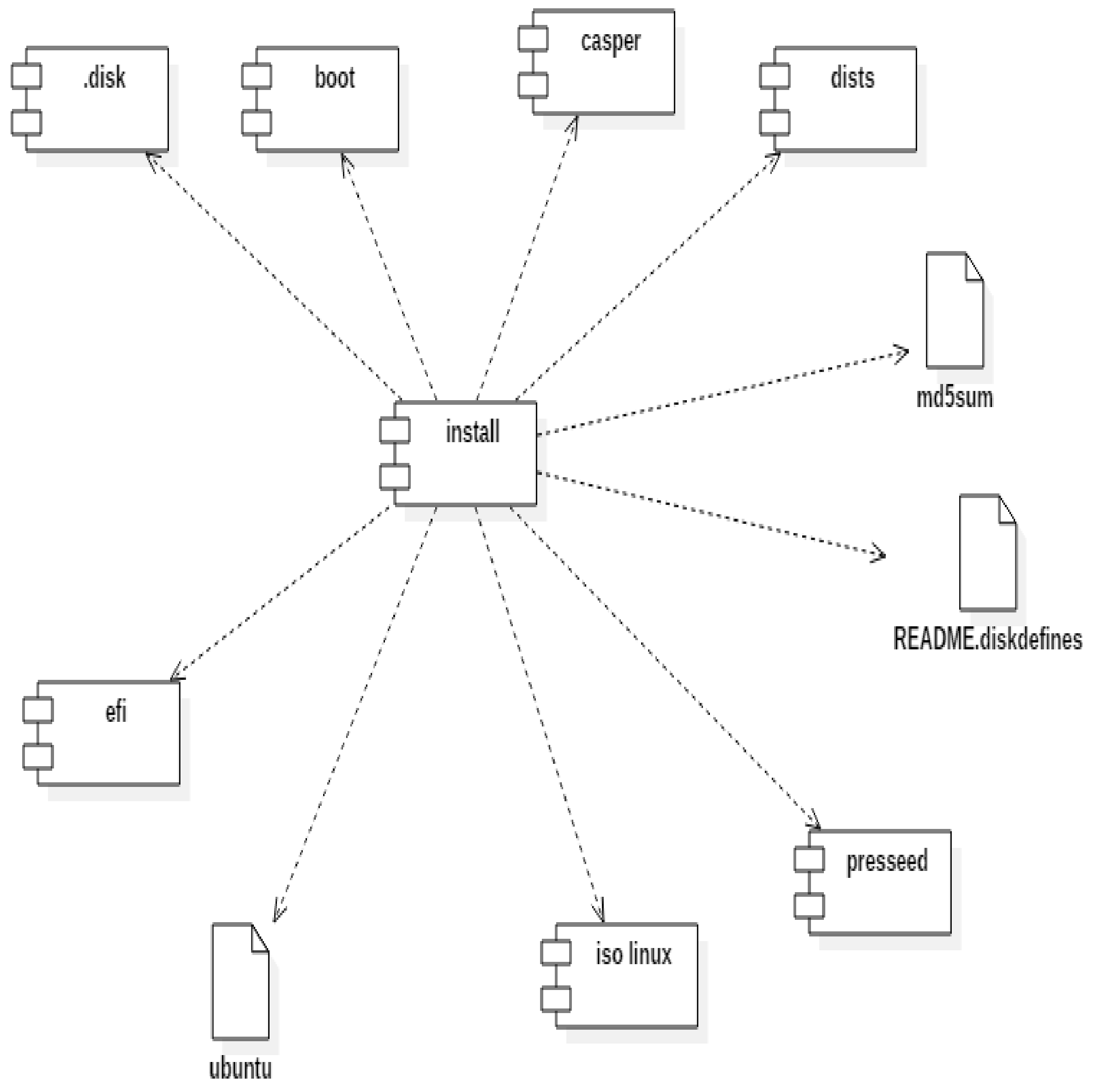
Object Diagram



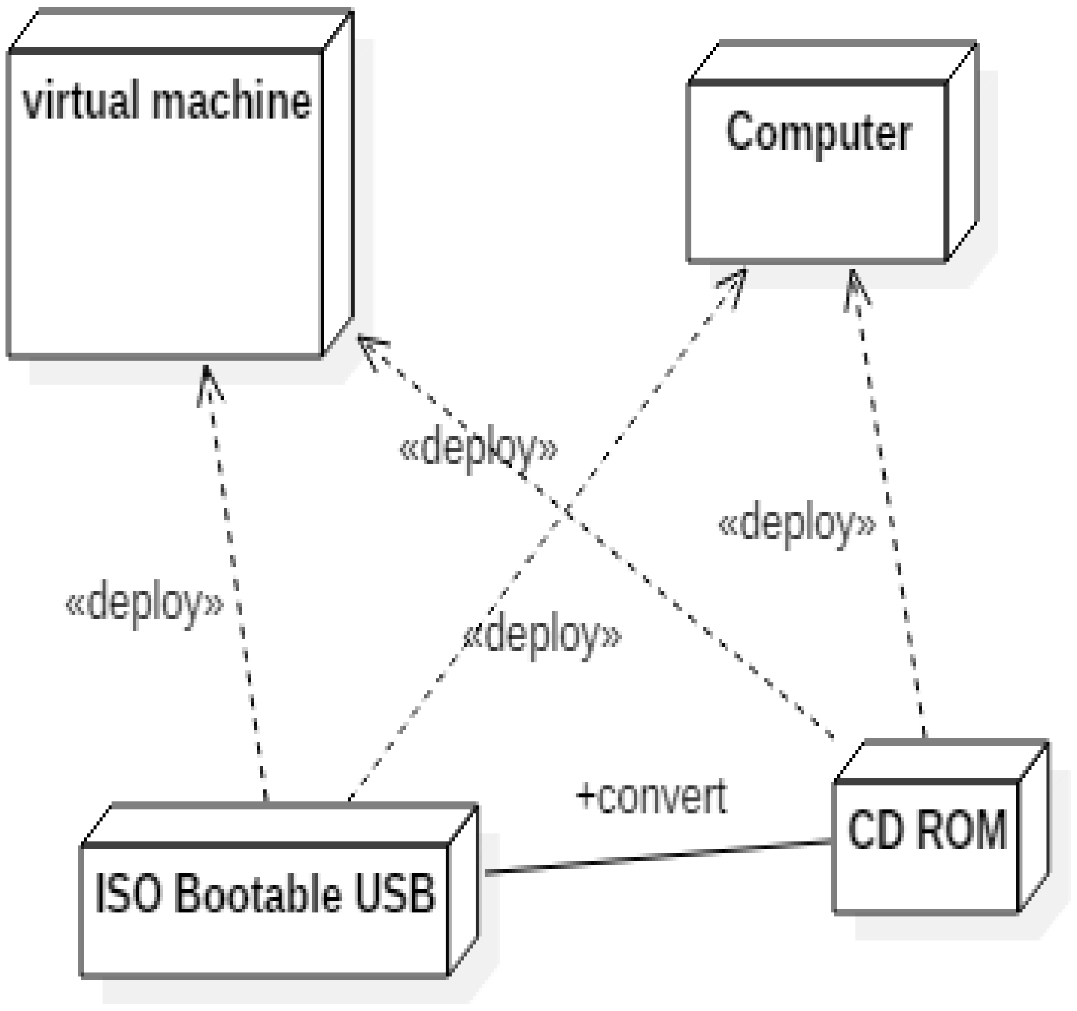
Activity Diagram:



Component Diagram:



Deployment Diagram:



4.3 Module design and organization

Elementary Package

Elementary OS is a Linux distribution based on Ubuntu. It is the flagship distribution to showcase the Pantheon desktop environment. Human Interface Guidelines of the elementary OS project focus on immediate usability with a gentle learning curve, rather than full-fledged customization. The three core rules the developers set for themselves were "concision", "avoid configuration" and "minimal documentation".

Since its inception, elementary OS has received both praise and criticism for its design, which closely resembles that of macOS both visually and in terms of user experience.

Pantheon's main shell is deeply integrated [citation needed] with other elementary OS applications like Plank (a dock), Epiphany (the default web browser) and Scratch (a simple text editor). This distribution uses Gala as its window manager, which is based on Mutter.

It Has a Clear Identity and Vision: The difference between most Linux operating systems (“distros”) is hard to describe to people. Fedora, openSUSE, and Ubuntu all provide largely the same software. Yes, they don’t use the same package formats and choose different defaults, but you could spend the better part of one or two podcast episodes discussing the differences and still not walk away with a clear answer.

That isn’t the case with Elementary OS. This Linux operating system has its own desktop environment (called Pantheon, but you don’t need to know that). It has its own user interface, and it has its own apps. Technically, you can run Elementary OS software inside another distro, but there isn’t much reason to do so.

It’s Easy to Learn: Elementary OS is simple. When you fire up the desktop for the first time, it takes mere seconds to figure everything out. You launch applications from the menu in the top-left corner labelled Applications. When you do, they appear in the dock at the bottom, where you can also save your favorites.

Its Interface Is Consistent: Consistency. Consistency. Consistency. When you open up an app in Elementary OS, it looks and works similarly to the one you opened before. That’s because the team has not only established clear design guidelines, but it sticks to them. Elementary also makes it easy for other developers to create apps that conform to the rules. They’re not left wondering how many pixels should go between buttons in the toolbar. This means once you learn how to use one Elementary OS app, you’ve largely figured out how to use the next one. I find it jarring to switch from a GTK-based app to a KDE one. Even going from a GNOME app to a GTK one like GIMP or LibreOffice can be jarring. Elementary OS isn’t immune to this issue, since you will likely need to install non-Elementary software at some point, but at least all of the software designed for Elementary is similar.

It Has Few Distractions: Thanks to a lack of distractions, Elementary OS helps me stay focused. When I use KDE, I spend a little time each day tweaking various aspects of the interface. I lose hours of productivity moving panels around, searching for themes, tweaking widgets, and altering applications. Elementary doesn’t let you move the panel around and doesn’t provide themes, which is rare for a Linux desktop. Out of the box customizations are limited to the dock and hot corners (the ability to view all windows, see the desktop, or perform other actions when moving the mouse to the corner of the screen). I’ve spent minutes playing around with the options and decided I prefer the defaults. The interface is minimalist, keeping the focus on apps. There is no dashboard. Right-clicking the panel or the desktop doesn’t bring up a context menu. Nearly every option is contained within System Settings, and there aren’t all that many there. The Elementary OS interface doesn’t provide much to see or do, so you might as well stay focused on what you came to your computer to do in the first place.

It Has Great Default Apps: There are differing opinions on whether default apps are all that important. As long as you have a reliable internet connection, you can download alternatives. But I find default apps matter a great deal on desktop environments that don’t fit the usual paradigm, such as GNOME and Elementary OS’s Pantheon. Most alternatives simply don’t integrate well with the rest of the environment or other apps. Even if I ignore looks and integration, I simply love the default apps on Elementary OS. The Mail app (a fork of Geary) is my favorite email client I have ever used, even despite the semi-regular crashes. The file manager does what I need without looking cluttered. The Photos app (based on Shotwell) is capable of importing photos, organizing them, and performing minor tweaks. The Music app doesn’t automatically fetch album art, but the layout is intuitive and the features I want are all there.

It Has a Steady Flow of New App: These days, Elementary OS is enjoying a regular supply of new apps. Sure, the number is nothing compared to what you see in a mobile app store. Sure, new apps launch for Windows and macOS at a faster rate. But we’re comparing with the rest of the Linux landscape, and by that metric, the amount of new releases is impressive. Most of these apps are simple, but that isn’t necessarily a bad thing. Many of these apps do things that other Linux programs can already do.

Ubuntu

Ubuntu is an open source operating system for computers. It is a Linux distribution based on the Debian architecture. It is usually run on personal computers, and is also popular on network servers, usually running the Ubuntu Server variant, with enterprise-class features. Ubuntu runs on the most popular architectures, including Intel, AMD, and ARM-based machines. Ubuntu is also available for tablets and smartphones, with the Ubuntu Touch edition.

Ubuntu is published by Canonical Ltd, who offer commercial support. It is based on free software and named after the Southern African philosophy of Ubuntu (literally, 'human-ness'), which Canonical Ltd. suggests can be loosely translated as "humanity to others" or "I am what I am because of who we all are".

Ubuntu is the most popular operating system running in hosted environments, so-called "clouds", as it is the most popular server Linux distribution.

Development of Ubuntu is led by UK-based Canonical Ltd., a company founded by South African entrepreneur Mark Shuttleworth. Canonical generates revenue through the sale of technical support and other services related to Ubuntu. The Ubuntu project is publicly committed to the principles of open-source software development; people are encouraged to use free software, study how it works, improve upon and distribute it.

Swecha

Swecha is a non-profit organization formerly called as Free Software Foundation Andhra Pradesh (or FSF-AP in short) later changed name to Swecha which is also the first Telugu Operating System released in year 2005, Swecha is a part of Free Software Movement of India (FSMI). This organization is a social movement that works towards enlightening the masses with the essence of Free Software and to liberate knowledge to the commoners. Swecha organizes different workshops and seminars in the Indian state of Telangana and Andhra Pradesh among the youth to spread the idea of knowledge liberation. The Swecha has a sizable number of followers in states of Telangana and Andhra Pradesh and a vibrant community of software users, students, academicians and software professionals/developers determined to provide quality software built on the guidelines of free software development model.

Swecha is a free software project aimed at coming out with a localized version of Linux Operating System in Telugu and providing global software solutions to the local people with the Free Software development model by working together with the community of developers and users all over. The prime objective of Swecha OS is to provide a complete computing solution to a population that speaks and understands only Telugu. The target users of the Distro being the entire community that is a prey of the digital divide. This project helps in coming out with a solution for the digital divide and allows the possibility of digital unite becoming a reality. The project aims at bridging the gap between the computer technology that exists predominantly in English and the Telugu-speaking community of India. The project also aims at providing a framework for development and maintenance of Free Software projects taken up by the community.

Bala Swecha is a free software project initiated by the Swecha for tiny tots, It is a school distro with many of the useful interactive applications for the school goers. Its stack is filled with educational suites for all the standards right from elementary to 10th standards. They cover a wide range of applications which make the student learn

Maths, Physics, Geography, Chemistry etc., very easily. Swecha has taken up many activities in training the school teachers, computer instructors of several government schools. The aim of the Distro is to deliver a Free Software-based operating system for the project of "SarvaShikshaAbhiyan" initiated by the government. There isn't such operating system till now which gives full freedom with an educational stack. Swecha has the plans of localizing Bala Swecha for the benefit of Telugu medium students.

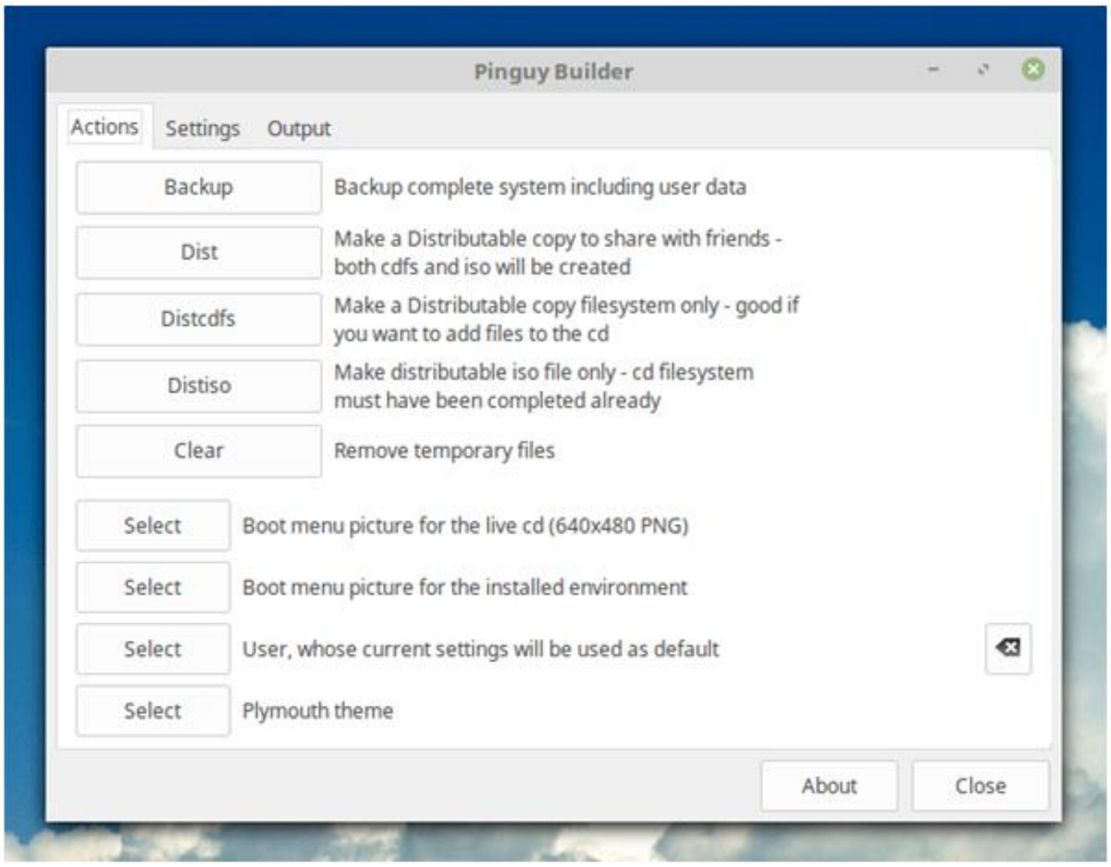
E-Swecha is a free software project initiated by the Swecha and is aimed at developing a free Operating System, which is not built by a software firm. Neither is it built by a few programmers. It is a collaborative work of hundreds of Swecha Volunteers/engineering students in and around Hyderabad to, for and by the engineering students.

Pinguy Builder

There are many Linux distributions out there, and each one of them has its own features and programs. Ubuntu is the most popular Linux distribution, featuring a lot of features that make life easier on end users, therefore creating a distribution based on it will be a good choice if you are beginner and want to learn a new thing, or share your own copy of the operating system with the world.

The Easy Way: For that, you can use PinguyBuilder, it’s a program that was originally forked of Remastersys and developed by the Pinguy OS team, it’s working well with Ubuntu 16.04 LTS.

Download and open the program from the applications menu:



Backup complete system including user data

Make a Distributable copy to share with friends - both cdfs and iso will be created

Make a Distributable copy filesystem only - good if you want to add files to the cd

Make distributable iso file only - cd filesystem must have been completed already

Remove temporary files

Boot menu picture for the live cd (640x480 PNG)

Boot menu picture for the installed environment

User, whose current settings will be used as default

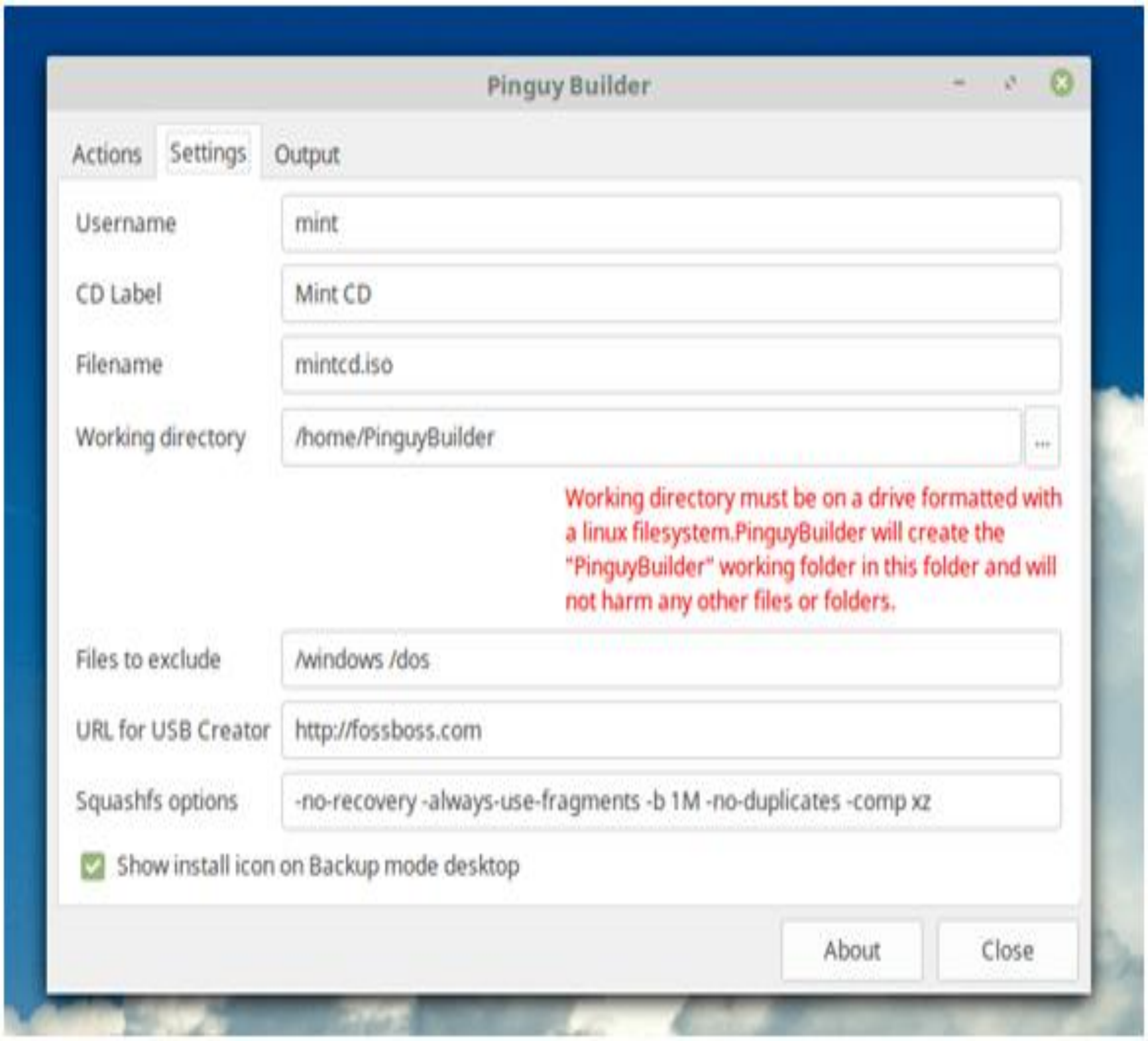
Plymouth theme

Fig-4.3.1 Pinguy Builder Beta for 16.04

- The first option will create a file combining all your files and programs (Will be so big and may not work with huge data).
- ·The second option will create an ISO combining the installed programs and the settings you choose only, not all your data, just the ones you choose from the settings to include.
- ·The third option will create a file-system tree only, no ISO file, you can include files or packages in that file system manually using “chroot” command and then use the 4-th option.
- The fourth option will create an ISO file from the file-system tree. You should have already used option 3 to do this.

As you can see also, you can choose the Plymouth theme (the graphics after the boot loader directly), edit the Live CD boot menu or choose whom user data you want to include. All files that you place under **/etc/skel** will be there by default in the users' home directory who will use your distribution.

Switch for the settings tab for more option:



Username: mint
CD Label: Mint CD
Filename: mintcd.iso
Working directory: /home/PinguyBuilder

Working directory must be on a drive formatted with a linux filesystem.PinguyBuilder will create the "PinguyBuilder" working folder in this folder and will not harm any other files or folders.

Files to exclude: /windows /dos
URL for USB Creator: http://fossboss.com
Squashfs options: -no-recovery -always-use-fragments -b 1M -no-duplicates -comp xz

✓ Show install icon on Backup mode desktop

Fig-4.3.2 Pinguy Builder Settings

The options here are explaining themselves. After you finish, return to the main tab and choose whatever build option you want to start, you will see the ISO file beside the checksums in /home/PinguyBuilder/ folder:

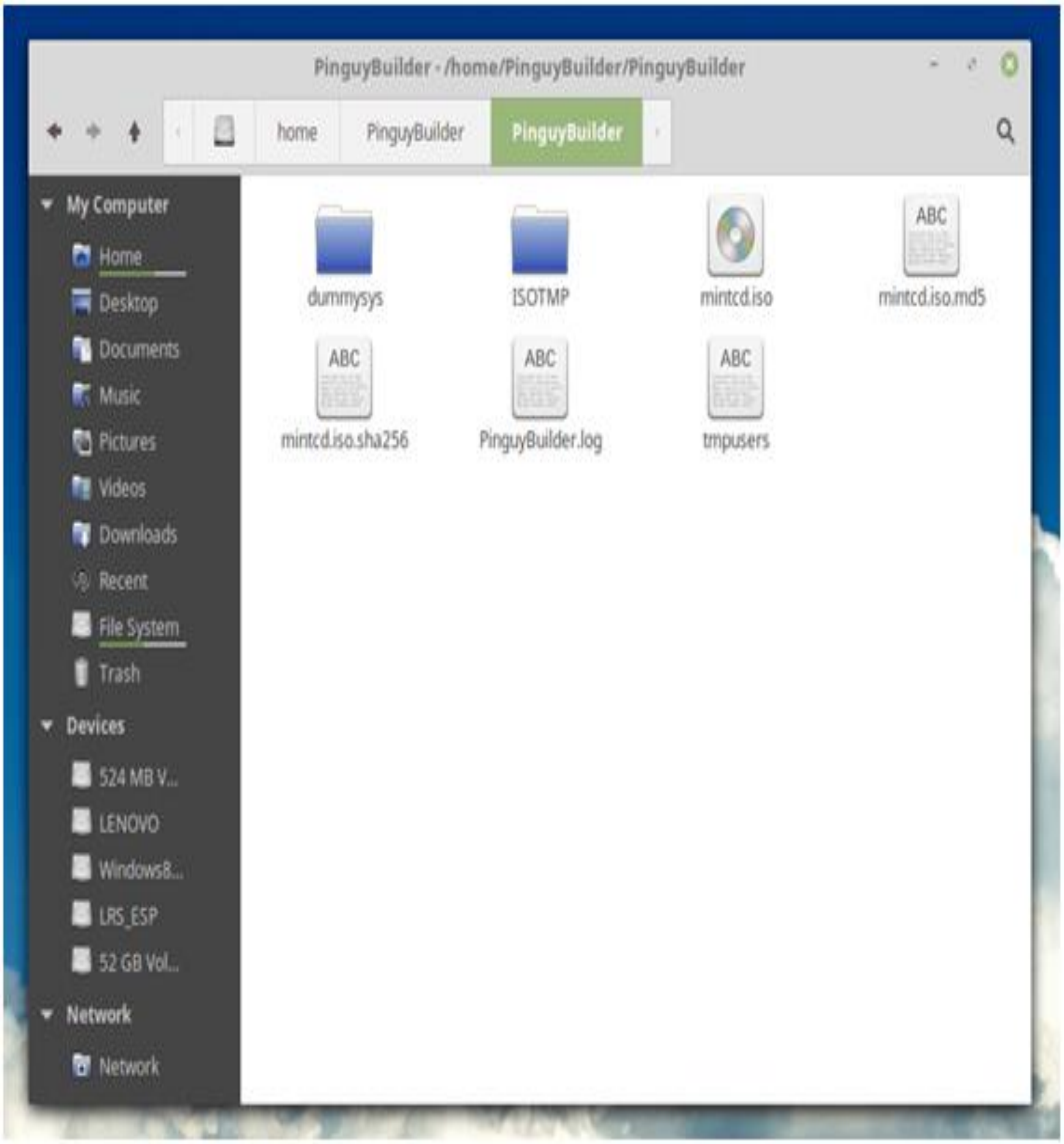
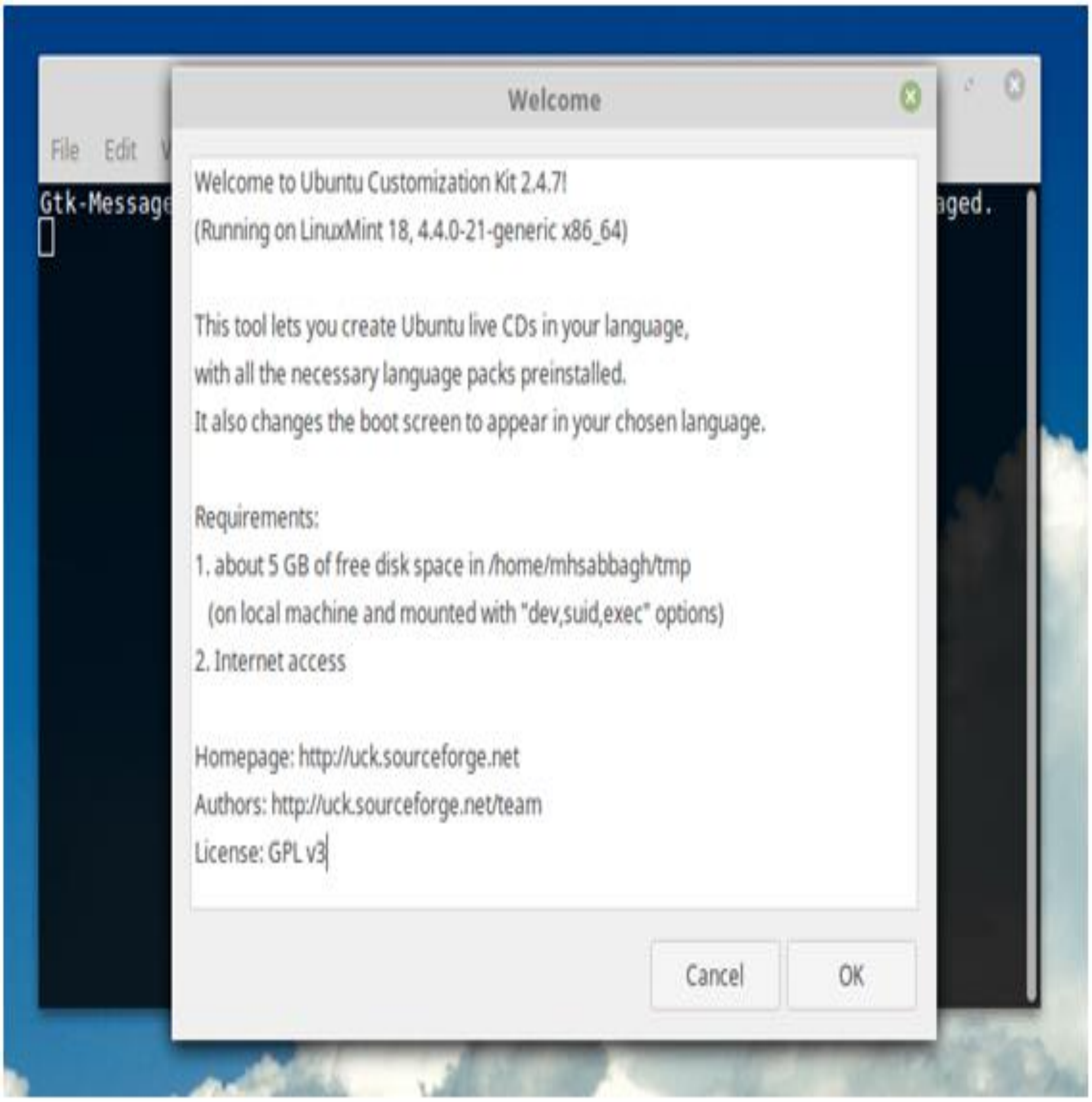


Fig-4.3.3 ISO file and Checksums by built Pinguy Builder

The Hard Way



```
Gtk-Messag
aged.
```

Welcome to Ubuntu Customization Kit 2.4.7!
(Running on LinuxMint 18, 4.4.0-21-generic x86_64)

This tool lets you create Ubuntu live CDs in your language,
with all the necessary language packs preinstalled.
It also changes the boot screen to appear in your chosen language.

Requirements:
1. about 5 GB of free disk space in /home/mhsabbagh/tmp
   (on local machine and mounted with "dev,suid,exec" options)
2. Internet access

Homepage: http://uck.sourceforge.net
Authors: http://uck.sourceforge.net/team
License: GPL v3

Fig 4.3.4 Ubuntu Customization Kit

You can create your distribution based on Ubuntu by extracting the components of the ISO file of Ubuntu to a folder, modify them and then re-build the ISO file. This is how it works theoretically.

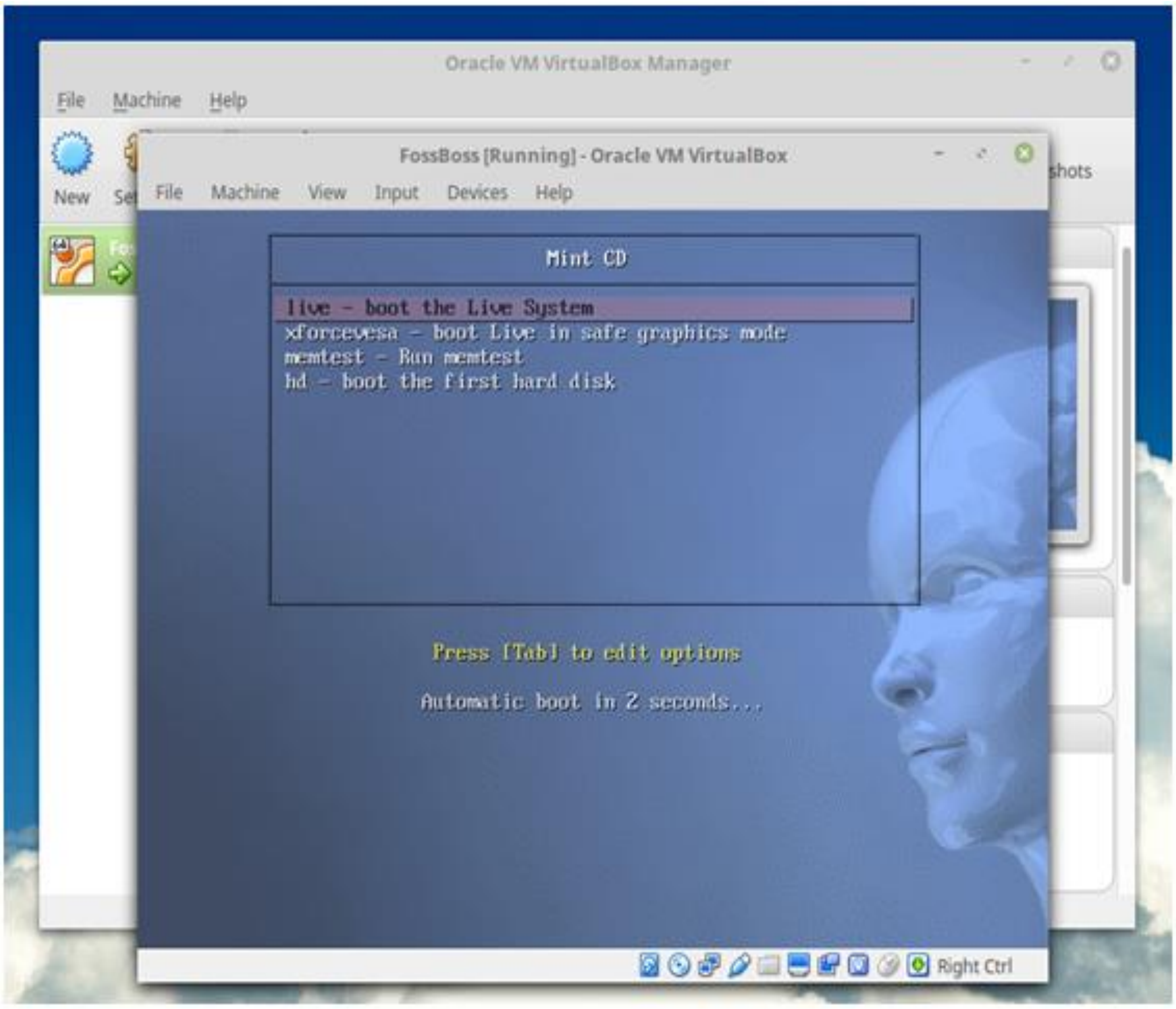
You can use this complete official guide from the Ubuntu community to learn how to do so, as you can see, it takes a lot of effort to do the same results, your time will be wasted between downloading packages and compressing and extracting file-systems, copying and modifying all the files manually, which is why people use programs.

If you want, you can use UCK (Ubuntu Customization Kit), it’s a tool that will do all the mentioned work in that article using graphical interfaces with less-need to tune things, it’s free and available to install from the repositories:

```
sudo apt install uck
```

If you want to be a first class-citizen in the Ubuntu family, you may consider building a distribution based on Ubuntu from the source ISO files for Ubuntu, you will have to download all the ISO files, combine them and then build your distribution manually using them.

Testing the Distribution


Oracle VM VirtualBox Manager
File
Machine
Help
New
FossBoss [Running] - Oracle VM VirtualBox
File
Machine
View
Input
Devices
Help
shots
Right Ctrl

```
Mint CD

live - boot the Live System
xforcevesa - boot Live in safe graphics mode
memtest - Run memtest
hd - boot the first hard disk

Press [Tab] to edit options

Automatic boot in 2 seconds...
```

Fig-4.3.5 Testing the ISO file using VirtualBox

Don’t ever release a thing to the world without testing; it’s an essential thing today if you don’t want to get insults from users who will download your distribution and use it on their own.

You have a lot of methods to do so, like:

- Using KVM, a virtual technology implemented in the kernel itself to run virtual systems. You can use many interfaces available to it like Qemu or GNOME Boxes.
- ·VirtualBox, a famous program developed by Oracle to run and test virtual systems.

All are available from the official repositories for Ubuntu:

```
sudo apt install gnome-boxes qemu
```

Or to install VirtualBox:

```
sudo apt install virtualbox virtualbox-qt
```

Then just search in the applications menu for the program and launch it to start testing.

Plymouth Package

Many people do enjoy a nice boot splash and there is a real desire to customize their spin or even the individual machine to their preferred theme.

Set-up

Obviously you can view Plymouth running by turning on your computer, but that does get rather tiresome if you wish to test frequently. Without restarting your machine, there are three sensible methods of testing Plymouth: using as victim machine, using a virtual victim machine or using the X11 plug-in.

Victim machine

This is the easiest method, although it does require a second physical machine to work with.

If you do not have a KMS compatible system, then you will need to add ***vga=0x318*** to your kernel line. This pushes the console to a VESA graphical mode. The ***0x318*** actually stipulates 1024x768x32bit mode. To see all the modes and find one most suitable for you, add ***vga=askme*** and a list will appear (don’t forget the ***0x***).

SSH into the machine as root (yes you do need to be root). The splash will be tested on tty1, but tty1 has two problems: there is an X session running on it and if you quit that, getty will start and capture keyboard events. To turn off getty on tty1, edit ***/etc/event.d/tty1***and comment out all the start and stop lines with a hash.

```
#start on stopped rc2

#start on stopped rc3

#start on stopped rc4



#stop on runlevel 0

#stop on runlevel 1

#stop on runlevel 6
```

Then drop to init level 3, which kills X and brings you to an empty console, by running init 3. Now install all the themes you wish to try out:

```
yum install plymouth\*
```

This is how to do this in Fedora, you should know the way to do this in your distro already.

Virtual victim machine

The alternate to using a physical machine is to use a virtual one. Both VMware and QEmu work very well. You will have to do an OS install on the virtual machine. These do not yet support KMS (support is coming soon apparently), so you will need to add the vga parameter on the kernel line. The setup is exactly the same as a one for a physical victim machine.

X11 plug-in

This method allows you to execute Plymouth within X. This is the preferred testing method **BUT** it can crash your X if you accidentally pick a text theme. It also does require you to compile your own version of Plymouth to get the x11 renderer. As root get the latest git checkout:

```
git clone git://anongit.freedesktop.org/plymouth
```

Next configure and compile it:

```
>cdplymouth

> ./autogen.sh --prefix=/usr/ --with-system-root-install

> make

> make install
```

Now it is installed, make sure you have the DISPLAY set correctly so even if you run as root, it can connect to your X session and display the windows. When you run it using the directions below, you will notice you get two screens which allows you to test the behavior of a two monitor setup.

Viewing

Now you have the system set up, you can manually execute plymouth to examine a theme. First, as root, start the plymouth daemon (plymouthd lives in /sbin/ so make sure you have that in your path):

```
>plymouthd
```

Now it is executing, you can control it using the plymouth program. To show the splash run:

```
>plymouth --show-splash
```

This should display the splash.

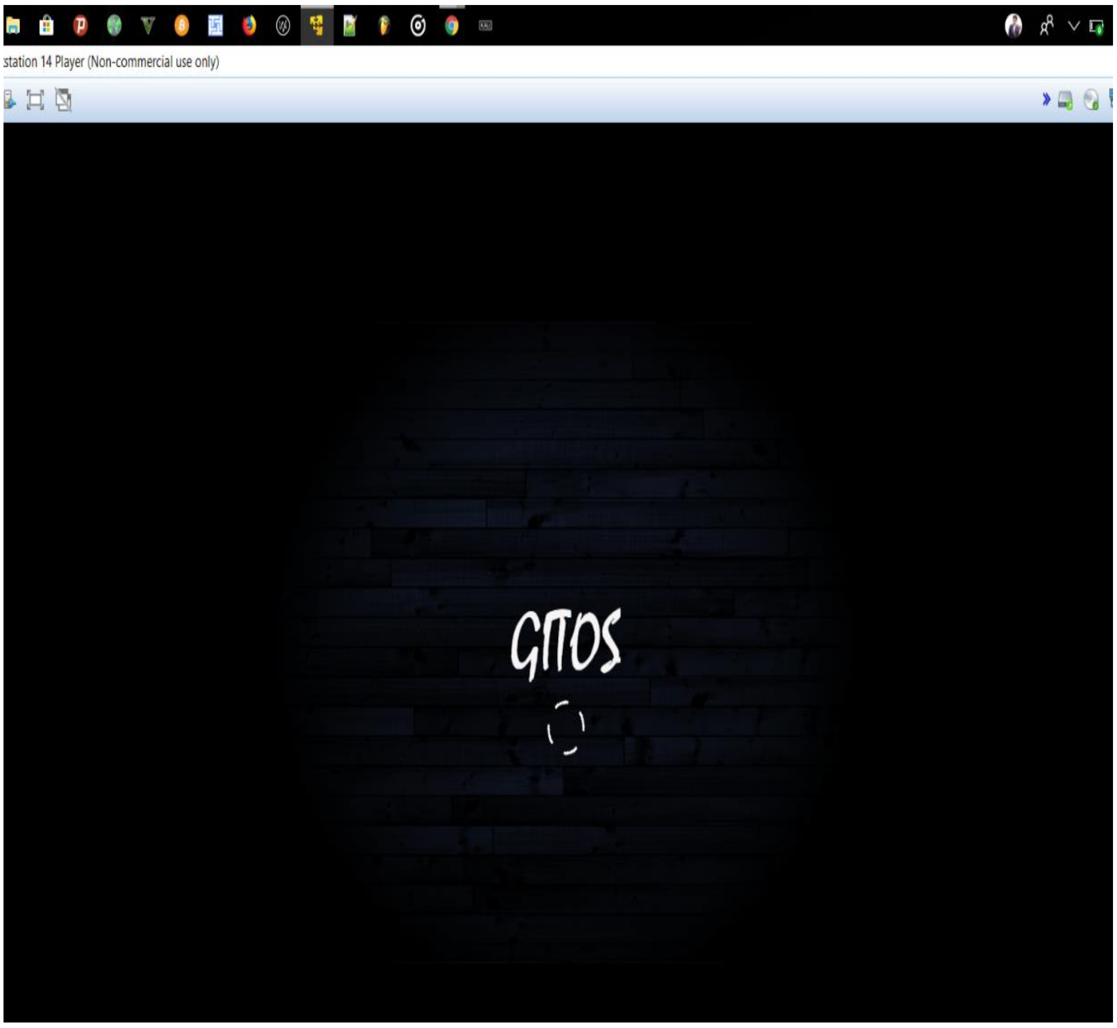


Fig-4.3.6 Plymouth Boot screen display

To quit, execute

```
>plymouth quit
```

The splash should then quit and the terminal return to its text mode. You may have noticed the progress did not advance very quickly. The progress is estimated using events, based on the event arrival time of the previous boot. To show the splash for 10 seconds and watch the progress moving execute the following one-liner:

```
>plymouthd; plymouth --show-splash ; for ((I=0; I<10; I++)); do plymouth --
update=test$I ; sleep 1; done; plymouth quit
```

There are several themes installed on your system. To get the full list run:

```
>plymouth-set-default-theme --list
```

To change to any of these themes replace –list with the theme name e.g.:

```
>plymouth-set-default-themespinfinity
```

Theme editing

It is important to understand the difference between a theme and a plug-in. A theme uses a plug-in and stipulates some parameters the plug-in uses. This information is stored in the directories under ***/usr/share/Plymouth/themes***/. Each directory has a ***.Plymouth*** file which describes the theme, stating the plug-in it uses and any parameters the plug-in allows to be set. To create a new theme, simply copy an entire directory to a new name, rename the ***.Plymouth*** file to the new name, and update its contents. In add ion, you can now also change the images that your new custom theme uses.

4.4 Conclusion

There are a lot of ways to build a customized Linux distribution based on Ubuntu, you may choose any one of them that fits your needs, but don’t forget testing; it’s a very important step in order to insure that your system is error-free and working well on any hardware that a user may have. As expected the designed Operating System will be more efficient and have tremendous graphical user interface which can attract large number of engineering students which will lead to the respective institutions to adapt to it easily. It can be implemented in a large scale and its availability can reduce the hefty work behind the installation of each application by the lab technicians.

5. IMPLEMENTATION and RESULTS

5.1 INTRODUCTION

The GITOS operating system includes the built-in engineering tools that an engineering student has to deal with during his course time. The idea is that given a system specification, by following the methodology and with the help of the tools integrated to support it, the user will be able to synthesize a system that meets his constraints. The operating system will be having a lucrative graphical user interface which will in turn give a better user experience. GITOS is an operating system completely for the engineers by the engineers and to the engineers.

The Purpose of GITOS to fulfill the needs and requirements of the modern engineering students, it has all the FOSS and Proprietary Licensed Academic software and tools which are more worth for an engineering student irrespective of their branch.

As GITOS is a Linux based operating system, Linux does not dominate the market like Windows; there are some disadvantages to using the operating system. First, it’s more difficult to find applications to support your needs. This is an issue for mostly businesses, but more programmers are developing applications that are supported by Linux. Many more applications are available for the working world compared to what was available a decade ago.

One main issue with Linux is drivers. Before you can install any hardware component in your computer, you must make sure the hardware has drivers available. Hardware manufacturers usually write drivers for Windows, but not all brands write drivers for Linux. This means that some of your hardware might not be compatible with Linux if you decide to switch.

5.2 Explanation of Key functions

Before going into the explanation of the key function in GITOS, first we need to justify why use GITOS?
There are many reasons to use GITOS, but here are some of the important ones:
It's easy to use, try or install – you don't have to be an expert.
It's a beautiful, sleek, and stylish operating system
It's stable and fast even on older laptops and computers
It is supported – you can get all the support and advice you need
It's always up-to-date – you get regular updates for free
Its open – meaning shared code, shared efforts, shared principles
Its open source code is transparent – no back doors ensuring security and your privacy
It has no major viruses – elementary OS is immune to computer-crashing Windows viruses.

Key functions

Electric

Electric is a general purpose system for all electrical design. It currently knows about NMOS, CMOS, Bipolar, artwork, schematics, printed-circuit boards, and many other technologies. It has a large set of tools including multiple design-rule checkers (both incremental and hierarchical), an electrical rules checker, over a dozen simulator interfaces, multiple generators (PLA and pad frame), multiple routers (stitching, maze, river), network comparison, compaction, compensation, a VHDL compiler, and a silicon compiler that places-and-routes standard cells.
In addition to the text terminal used to invoke the program, Electric uses a color display with a mouse as a work station. Separate windows are used for text and graphics.
If a *library* disk file is mentioned on the command line, that file is read as the initial design for editing.

Gmsh

Gmsh is a 3D finite element grid generator with a build-in CAD engine and post-processor. Its design goal is to provide a fast, light and user-friendly meshing tool with parametric input and advanced visualization capabilities.

Gmsh is built around four modules: geometry, mesh, solver and post-processing. The specification of any input to these modules is done either interactively using the graphical user interface or in ASCII text files using Gmsh's own scripting language.

Blender

Blender is a 3D modelling and rendering package. Originating as the in-house software of a high quality animation studio, Blender has proven to be an extremely fast and versatile design instrument. The software has a personal touch, offering a unique approach to the world of Three Dimensions.

Use Blender to create TV commercials, to make technical visualizations, business graphics, to create content for games, or design user interfaces. You can easy build and manage complex environments. The renderer is versatile and extremely fast. All basic animation principles (curves & keys) are well implemented.

Ktorrent

Ktorrent is a BitTorrent program for KDE. You can use it to download files from BitTorent network. Its features include speed capping (both down and up), integrated searching, UDP tracker support, UPnP support, IP blocking plugin, protocol encryption, file prioritization and much more.

If you specify URL, KTorrent will load the torrent from the specified location and start downloading it.

DragonPlayer

DragonPlayer plays audio and video in different formats, it does have a lucrative user interface along with multiple audio and video codecs which enables this player to play audio and video format multimedia from different range of media formats.

Audacity

Audacity is a graphical audio editor. This man page does not describe all of the features of Audacity or how to use it; for this, see the html documentation that came with the program, which should be accessible from the Help menu. This man page describes the Unix-specific features, including special files and environment variables.

Audacity currently uses libsndfile to open many uncompressed audio formats such as WAV, AIFF, and AU, and it can also be linked to libmad, libvorbis, and libflac, to provide support for opening MP2/3, OggVorbis, and FLAC files, respectively. LAME, libvorbis, libflac and libtwolame provide facilities to export files to all these formats as well.

Audacity is primarily an interactive, graphical editor, not a batch-processing tool. Whilst there is a basic batch processing tool it is experimental and incomplete. If you need to batch-process audio or do simple edits from the command line, using sox or ecasound driven by a bash script will be much more powerful than audacity.

Libreoffice

LibreOffice is a powerful office suite – its clean interface and feature-rich tools help you unleash your creativity and enhance your productivity. LibreOffice includes several applications that make it the most powerful Free and Open Source office suite on the market: Writer (word processing), Calc (spreadsheets), Impress (presentations), Draw (vector graphics and flowcharts), Base (databases), and Math (formula editing).

Your documents will look professional and clean, regardless of their purpose: a letter, a master thesis, a brochure, financial reports, marketing presentations, technical drawings and diagrams. LibreOffice makes your work look great while you focus on the content.

LibreOffice is compatible with a wide range of document formats such as Microsoft® Word, Excel, PowerPoint and Publisher. But LibreOffice goes much further with its native support for a modern and open standard, the OpenDocument Format (ODF). With LibreOffice, you have maximum control over your data and content – and you can export your work in many different formats including PDF.

Beyond the many features shipped by default, LibreOffice is easily extensible through its powerful extensions mechanisms. Get even more features and document templates on their dedicated websites.

LibreOffice is Free and Open Source Software, available for everyone to use, share and modify, and produced by a worldwide community of hundreds of developers. Our software is tested and used daily by a large and devoted user community – we're open to new talent and new ideas, so get involved and influence its future.

LibreOffice is a successor to OpenOffice.org (commonly known as OpenOffice), which in turn was based on StarOffice. Many years of development have gone into the software, and it has been used in its various incarnations by millions. Today, LibreOffice is by far the most active continuation of the OpenOffice.org codebase, with releases every six months and hundreds of contributors. Also, LibreOffice uses libraries from the Document Liberation Project, handing control back to content creators.

Ipython

An interactive Python shell with automatic history (input and output), dynamic object introspection, easier configuration, command completion, access to the system shell, integration with numerical and scientific computing tools, web notebook, Qt console, and more.

For more information on how to use IPython, see 'ipython --help', or 'ipython --help-all' for all available command-line options.

5.3 Method of Implementation

The following describes the Project Implementation Methodology. This methodology is based on recognized Project Management principles. The project is divided into a number of processes or phases, each phase having its own identity and characteristics, the respective phases are:

- Initiation Phase
- Planning phase
- Control phase
- Execution phase
- Project closing phase

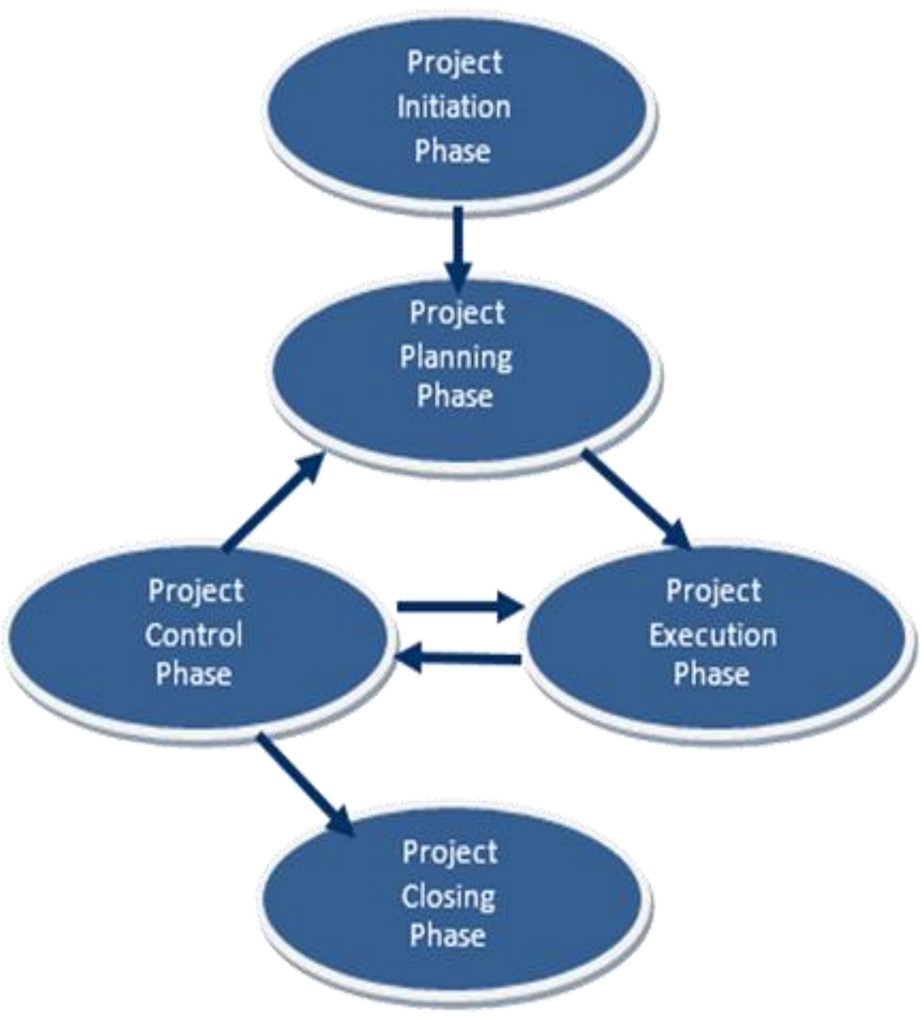


Fig (5.3.1) Project Implementation

Initiation Phase and Scoping: This project initiation phase comprises of the ideas of the project members and the need we got to know from the initial survey done. Also we had meetings with fellow students and our project coordinator, and during these meetings a number of project related issues were discussed and documented, normally in a document called a Project Charter.

In a complex environment it is particularly important to identify the detailed scope of the project, and the needs of the fellow students.

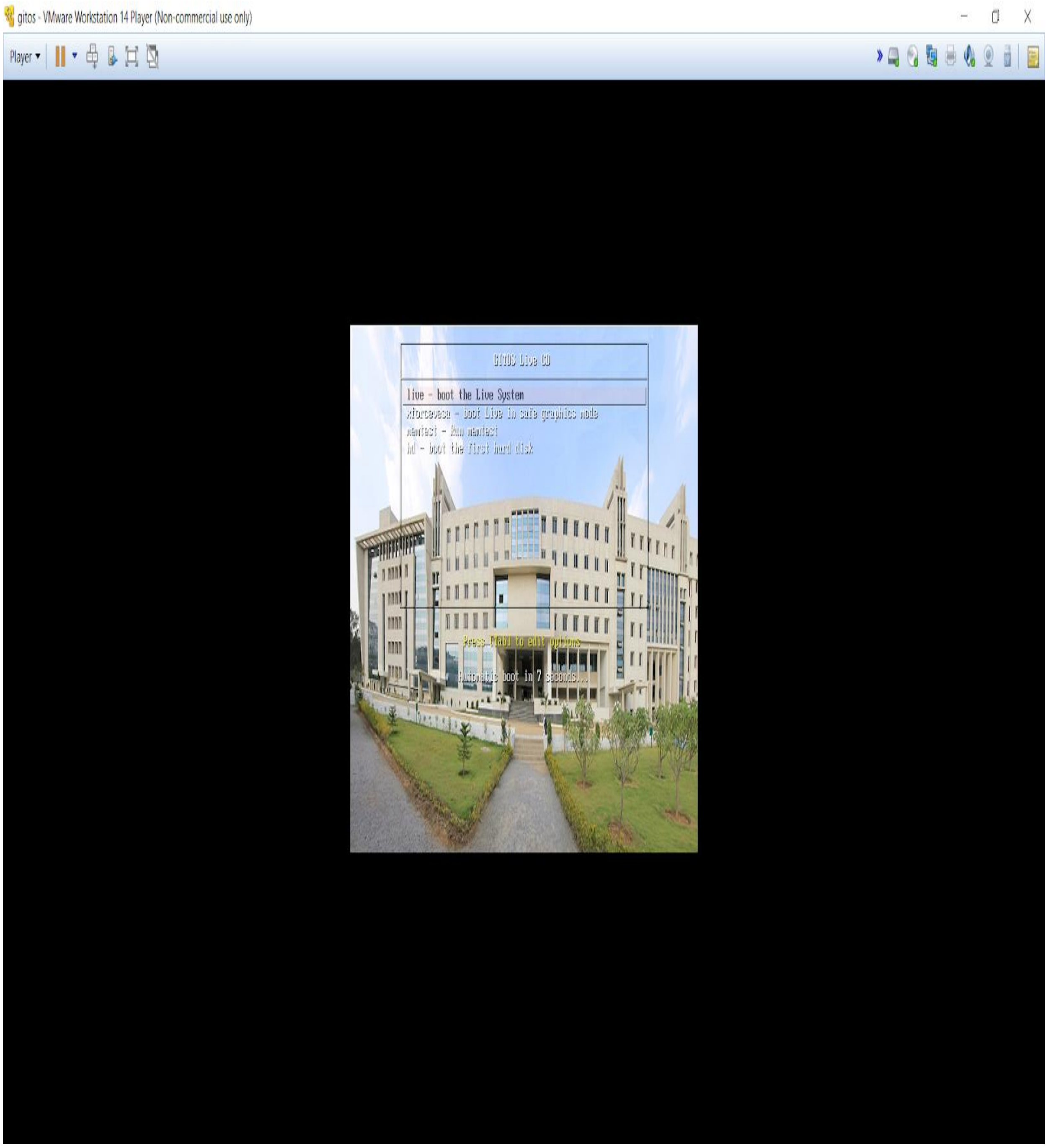
Planning Phase: The project planning phase results in the completion of the Project Schedule and resource planning and allocation. During the planning phase, decisions were taken concerning the various options and based on the functionality required together with incremental cost. Decisions were also taken concerning sequencing of the normal application flow, of existing software that requires interfaces into the proposed solution, and of the efficiencies that are obtained by so doing.

Control Phase: The project control phase comprises of the measurement of activities against the project schedule, and the recording of any variances to these. Variances that could affect the project delivery negatively are given urgent attention (they are referred back to the planning phase) and the necessary planning done or actions taken to restore project delivery as originally planned, if possible.

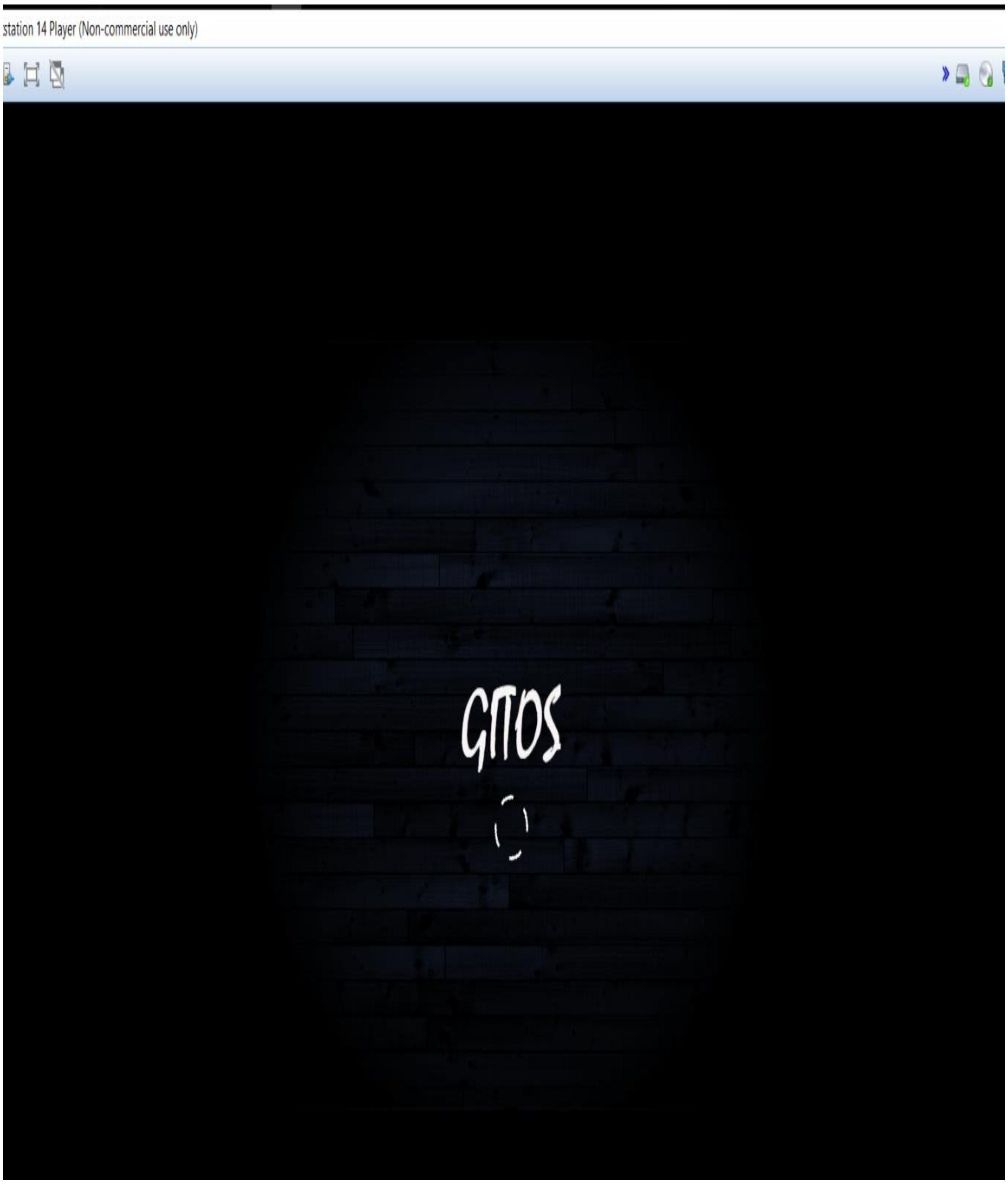
Execution Phase: This project execution phase comprised the implementation of the items detailed under planning above. This is when design, engineering, testing and commissioning activities were executed. The GITOS project was executed well with no crash reports and also by supporting the installation of various platform software applications required for an engineering student.

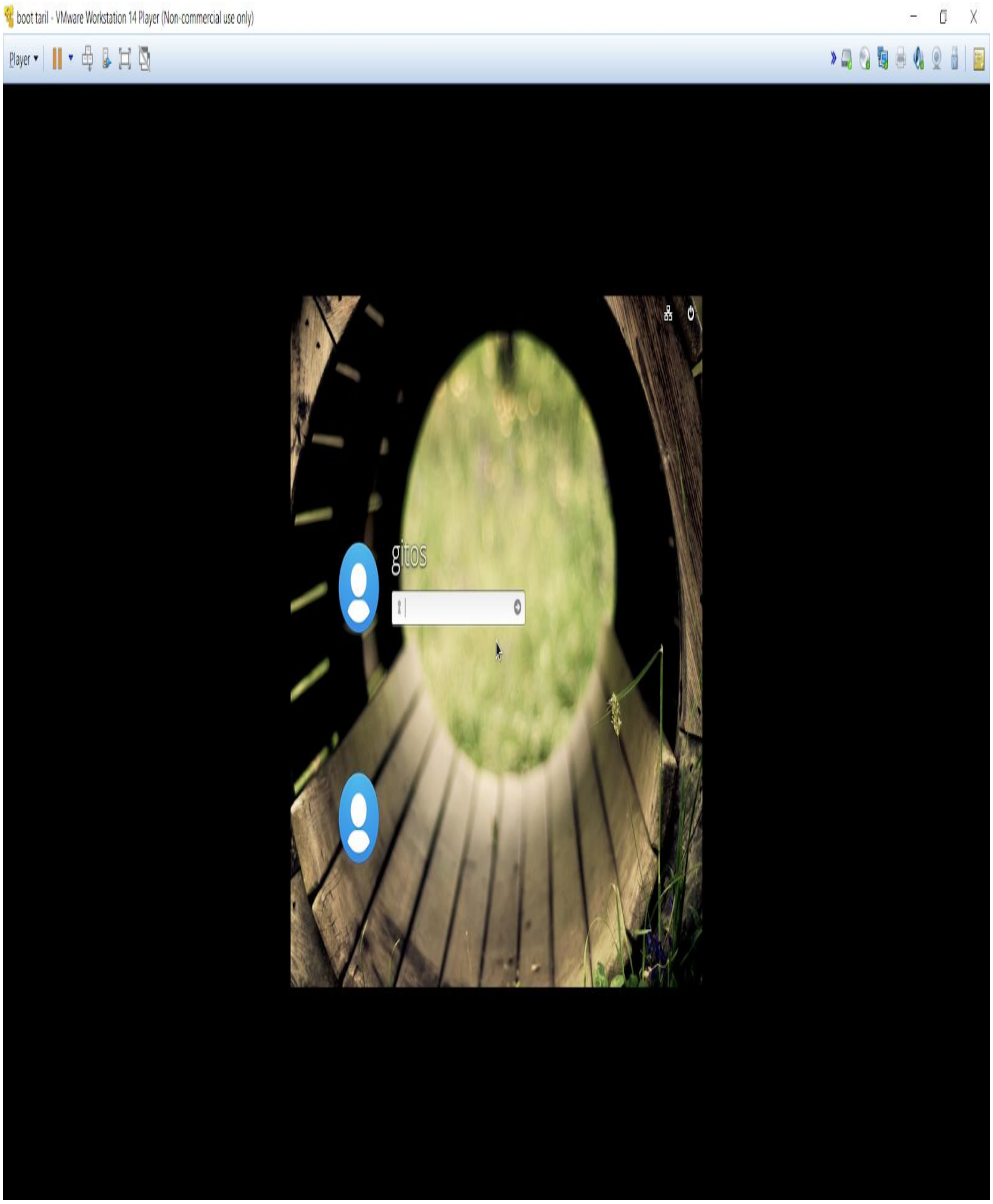
Project Closing Phase: This project-closing phase comprised activities to bring a project to completion. It is also applicable to the acceptance and sign-off of a key deliverable at the end of a phase within the execution of the project.

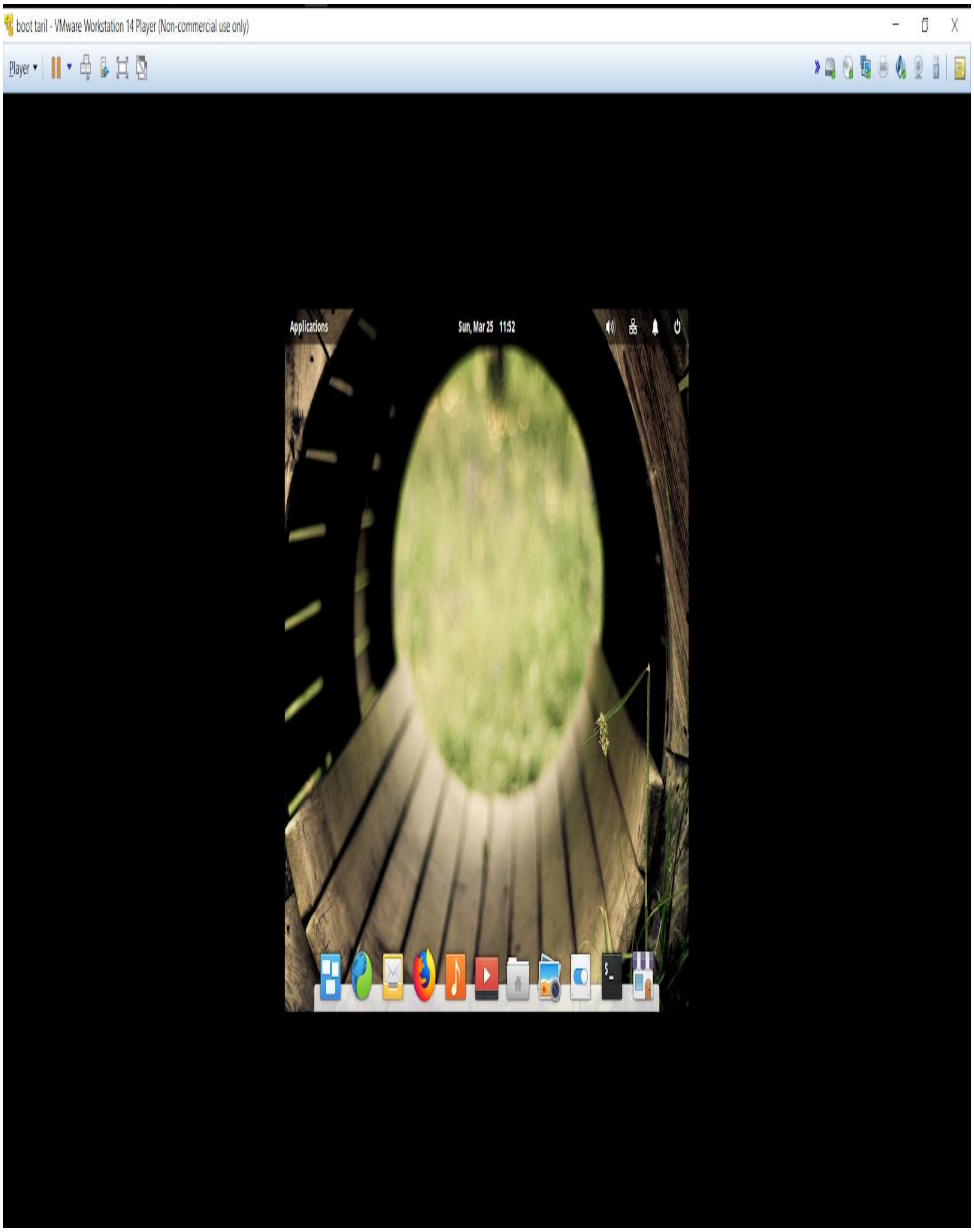
5.3.1 Output Screens

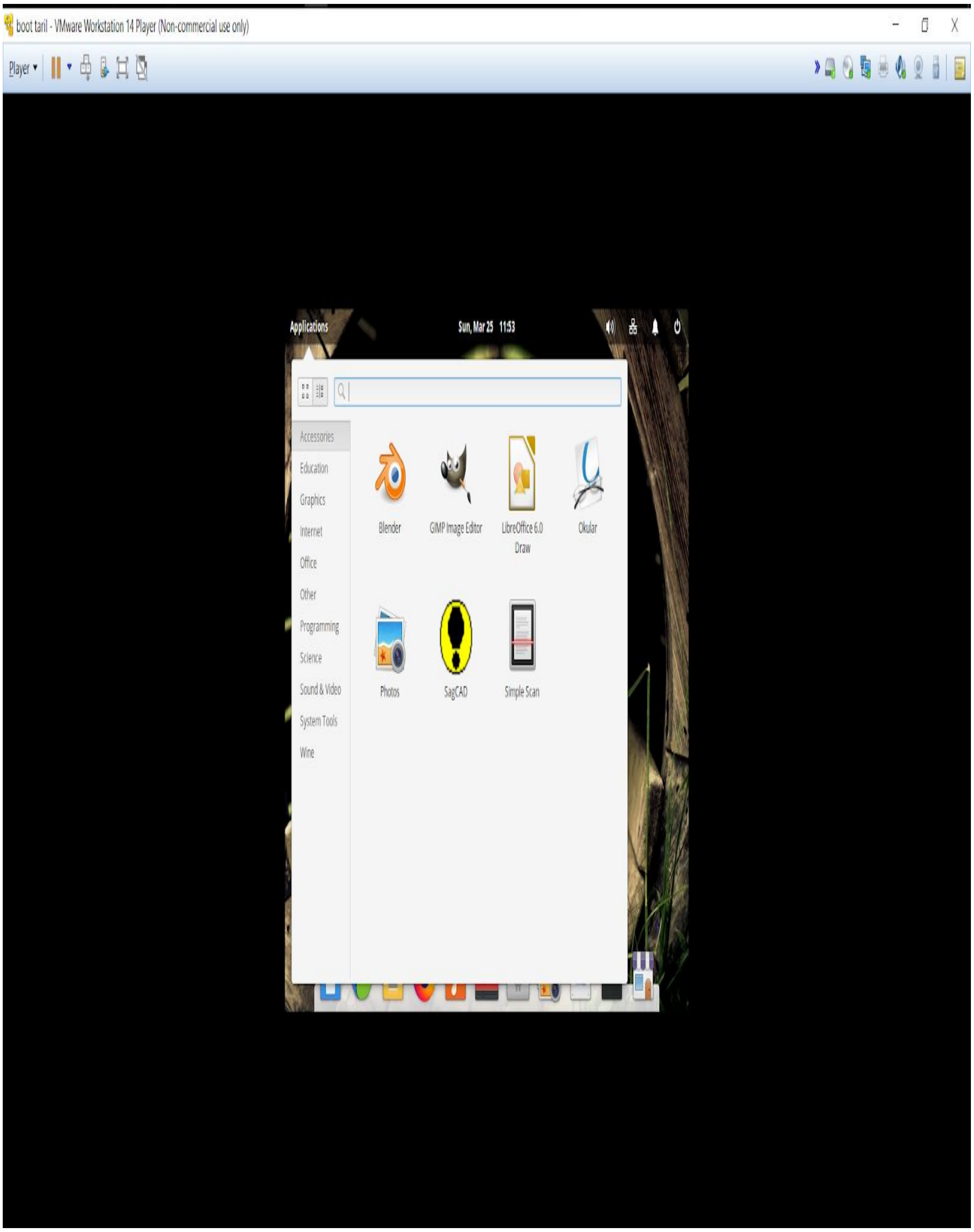

gitos - VMware Workstation 14 Player (Non-commercial use only)
Player

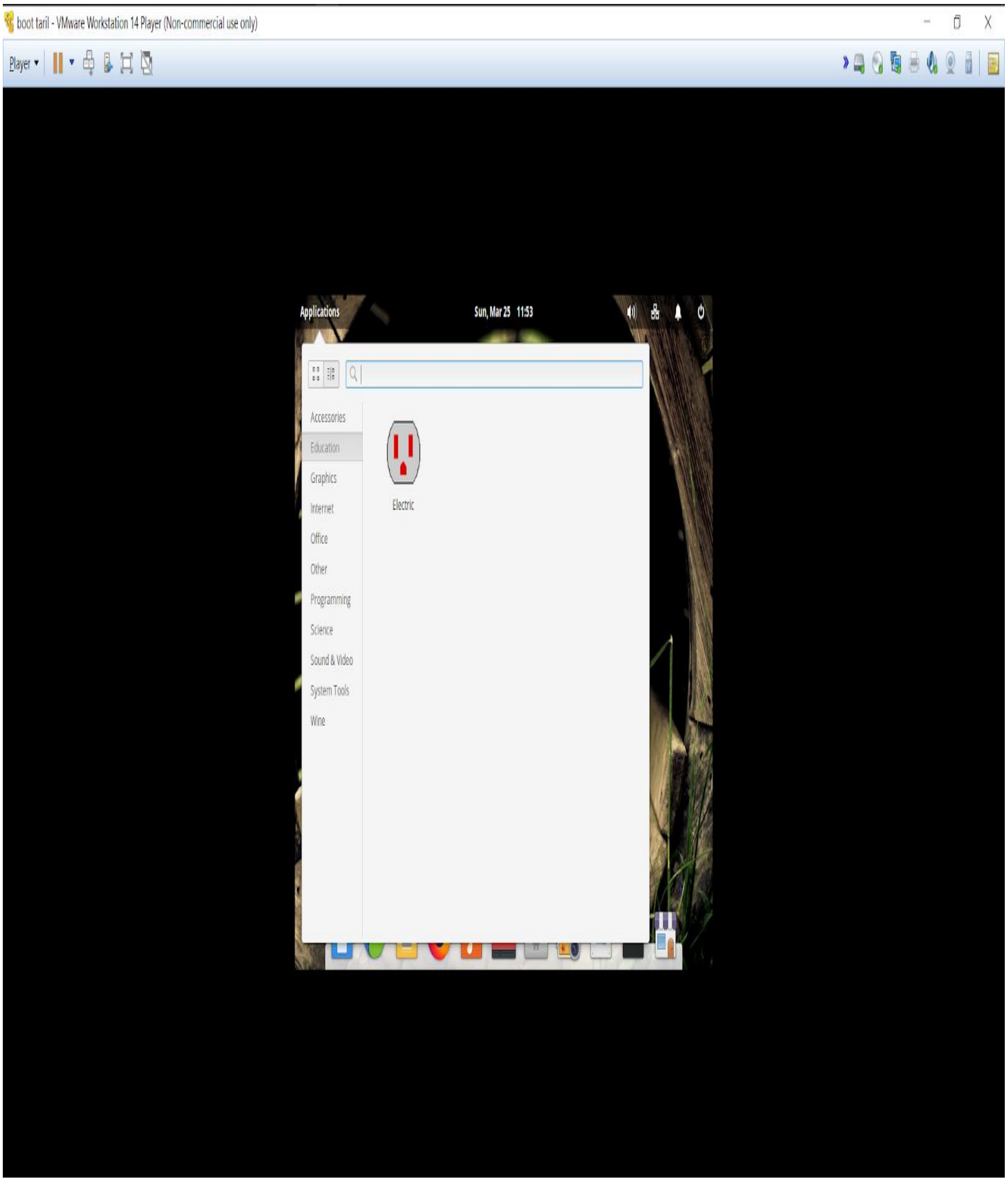
```
GITOS Live CD

live - boot the Live System
xforcevesa - boot Live in safe graphics mode
memtest - Run memtest
hd - boot the first hard disk

Press [Tab] to edit options

Automatic boot in 7 seconds...
```

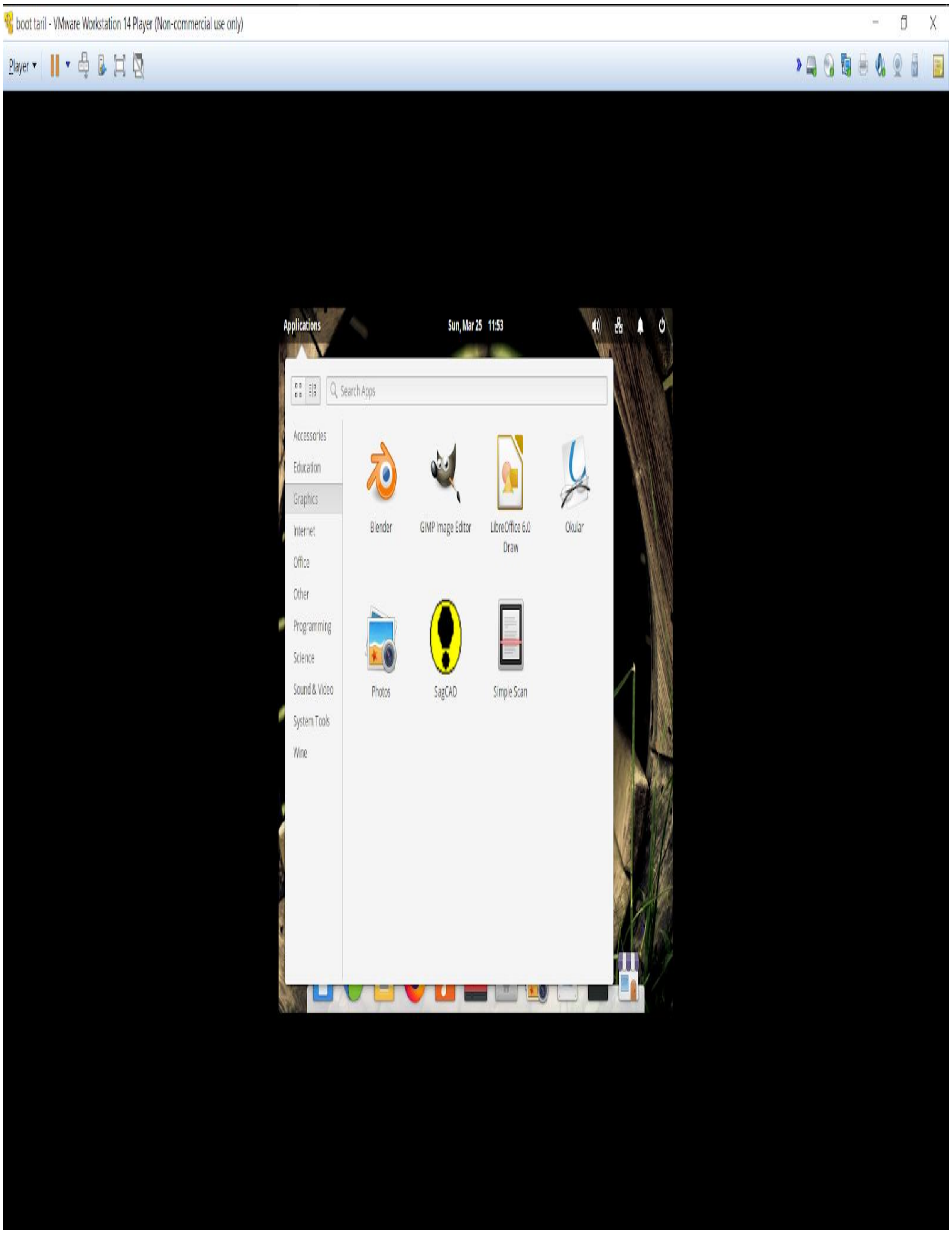

station 14 Player (Non-commercial use only)
GITOS

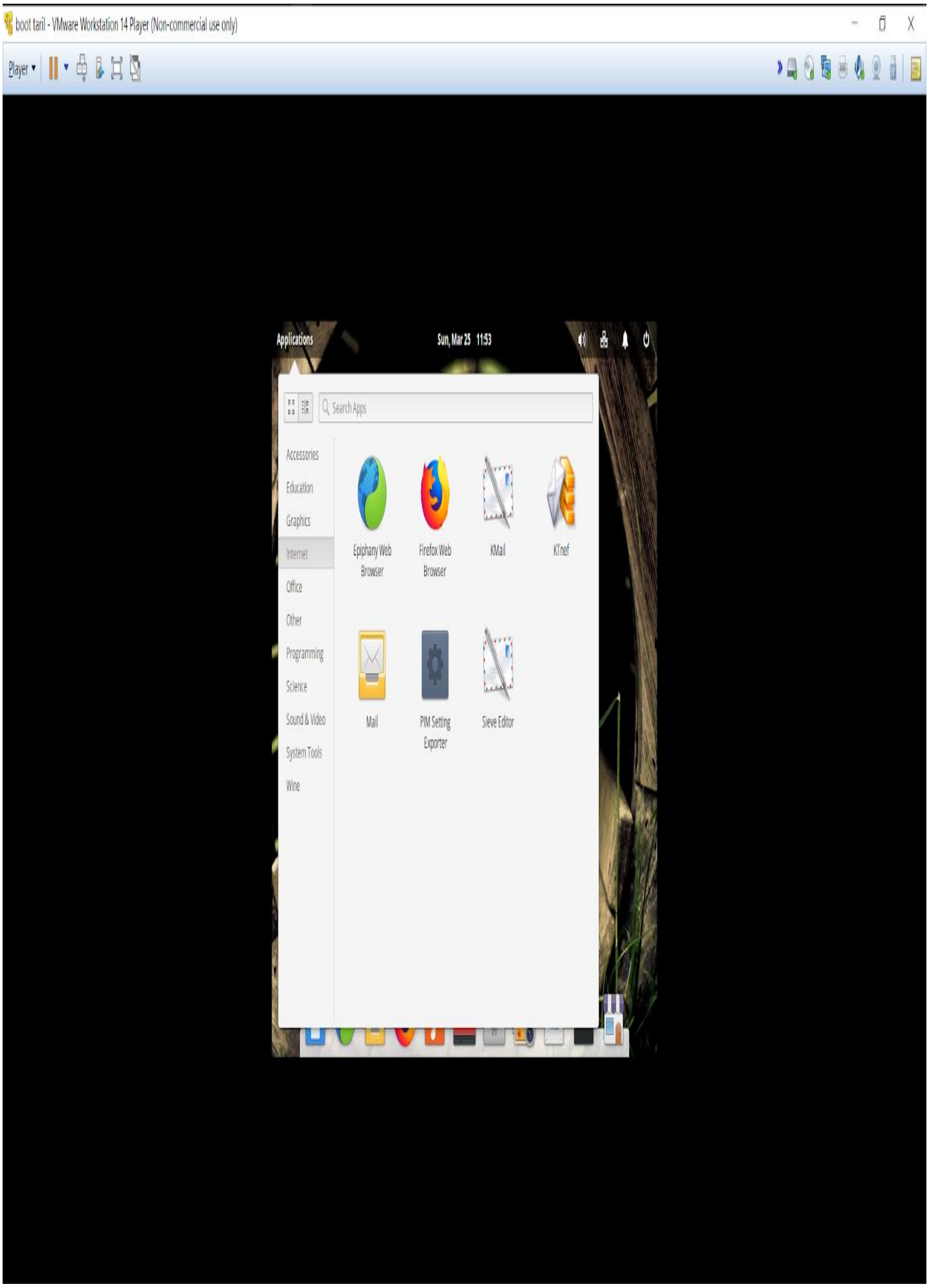

boot taril - VMware Workstation 14 Player (Non-commercial use only)
Player
gitos

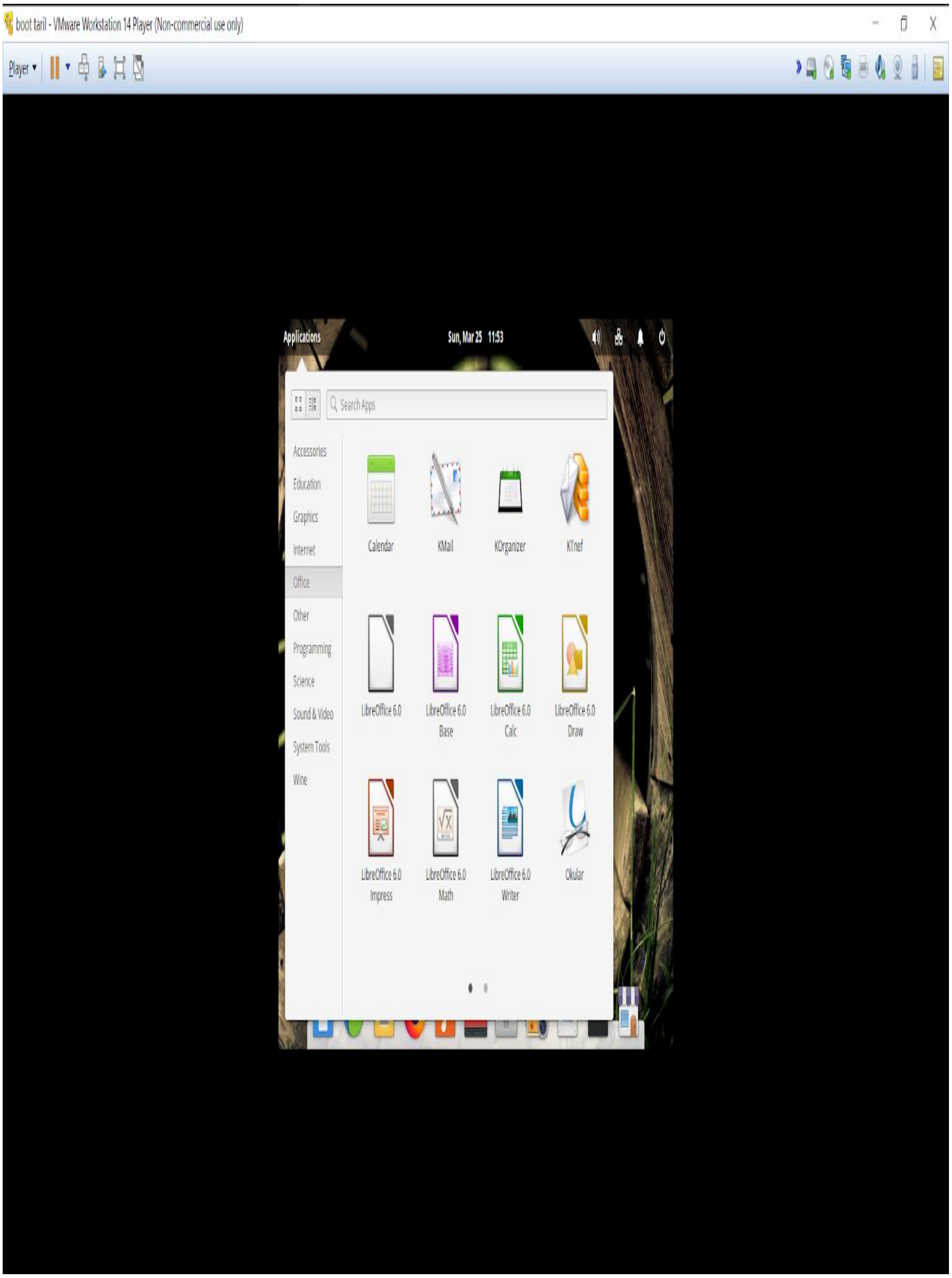

boot taril - VMware Workstation 14 Player (Non-commercial use only)
Player
Applications
Sun, Mar 25 11:52

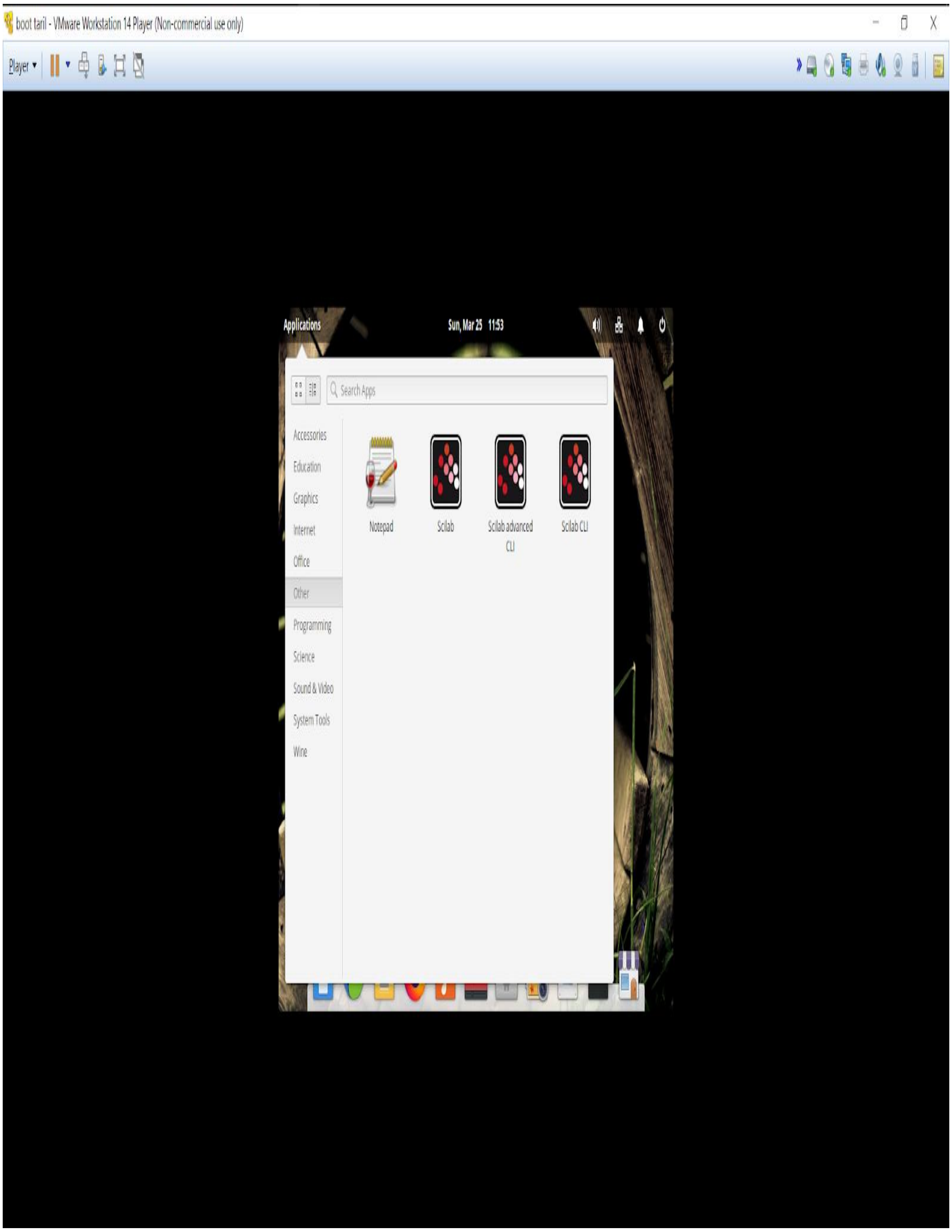

boot taril - VMware Workstation 14 Player (Non-commercial use only)
Player
Applications
Sun, Mar 25 11:53
Accessories
Education
Graphics
Internet
Office
Other
Programming
Science
Sound & Video
System Tools
Wine
Blender
GIMP Image Editor
LibreOffice 6.0 Draw
Okular
Photos
SagCAD
Simple Scan

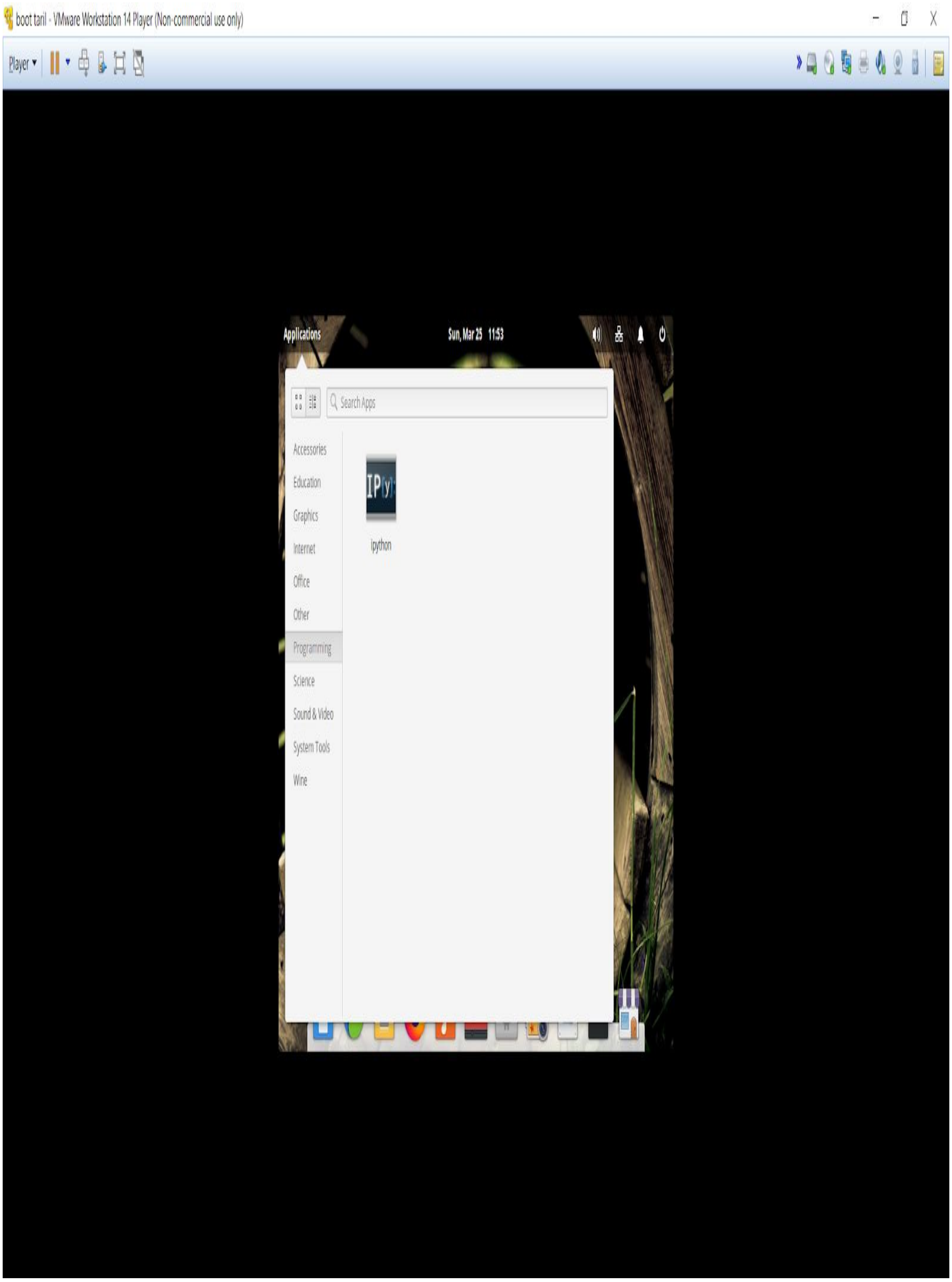

boot taril - VMware Workstation 14 Player (Non-commercial use only)
Player
Applications
Sun, Mar 25 11:53
Accessories
Education
Graphics
Internet
Office
Other
Programming
Science
Sound & Video
System Tools
Wine
Electric

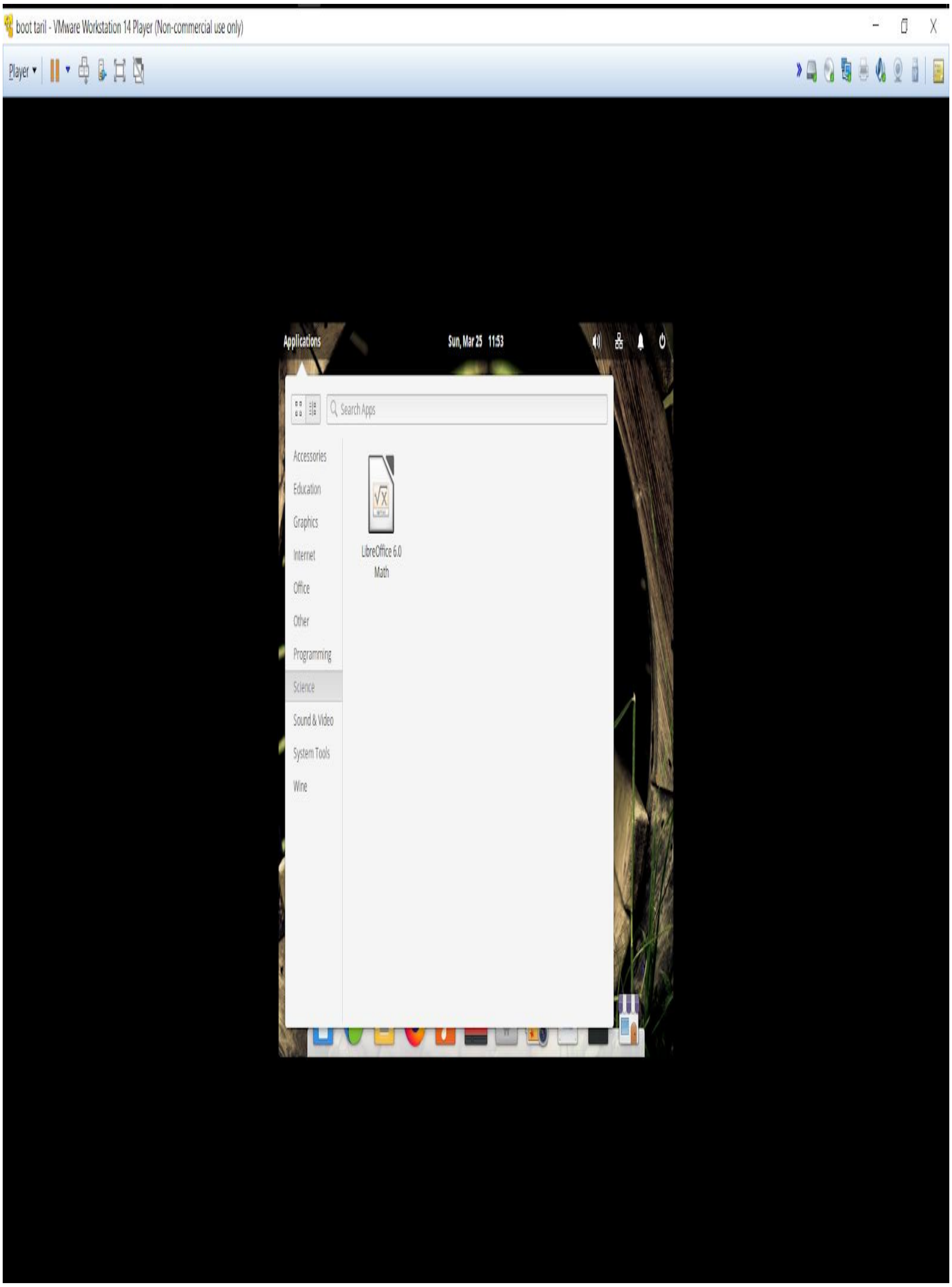

boot taril - VMware Workstation 14 Player (Non-commercial use only)
Player
Applications
Sun, Mar 25 11:53
Search Apps
Accessories
Education
Graphics
Internet
Office
Other
Programming
Science
Sound & Video
System Tools
Wine
Blender
GIMP Image Editor
LibreOffice 6.0 Draw
Okular
Photos
SagCAD
Simple Scan

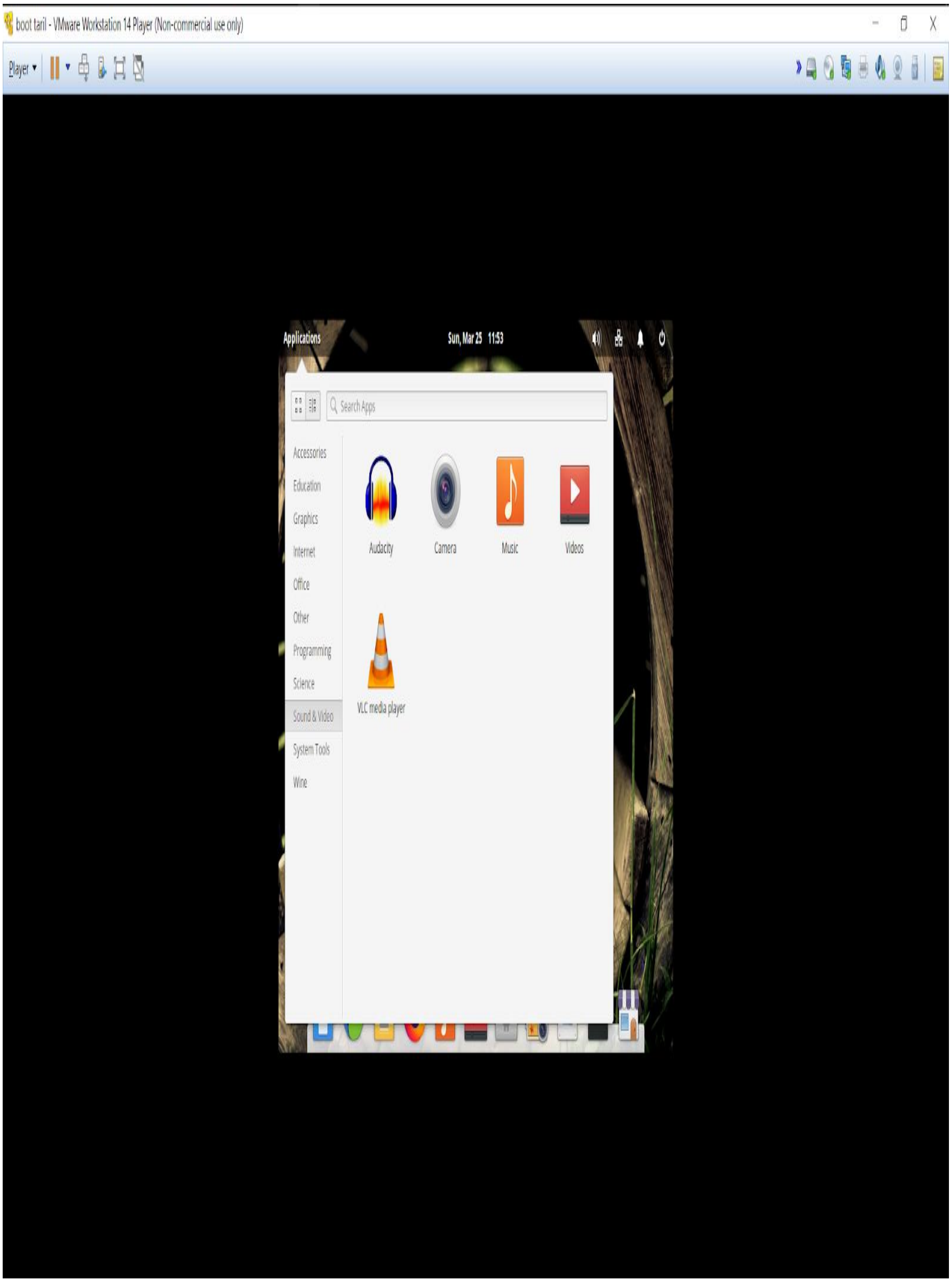

boot taril - VMware Workstation 14 Player (Non-commercial use only)
Player
Applications
Sun, Mar 25 11:53
Search Apps
Accessories
Education
Graphics
Internet
Office
Other
Programming
Science
Sound & Video
System Tools
Wine
Epiphany Web Browser
Firefox Web Browser
KMail
KTnef
Mail
PIM Setting Exporter
Sieve Editor

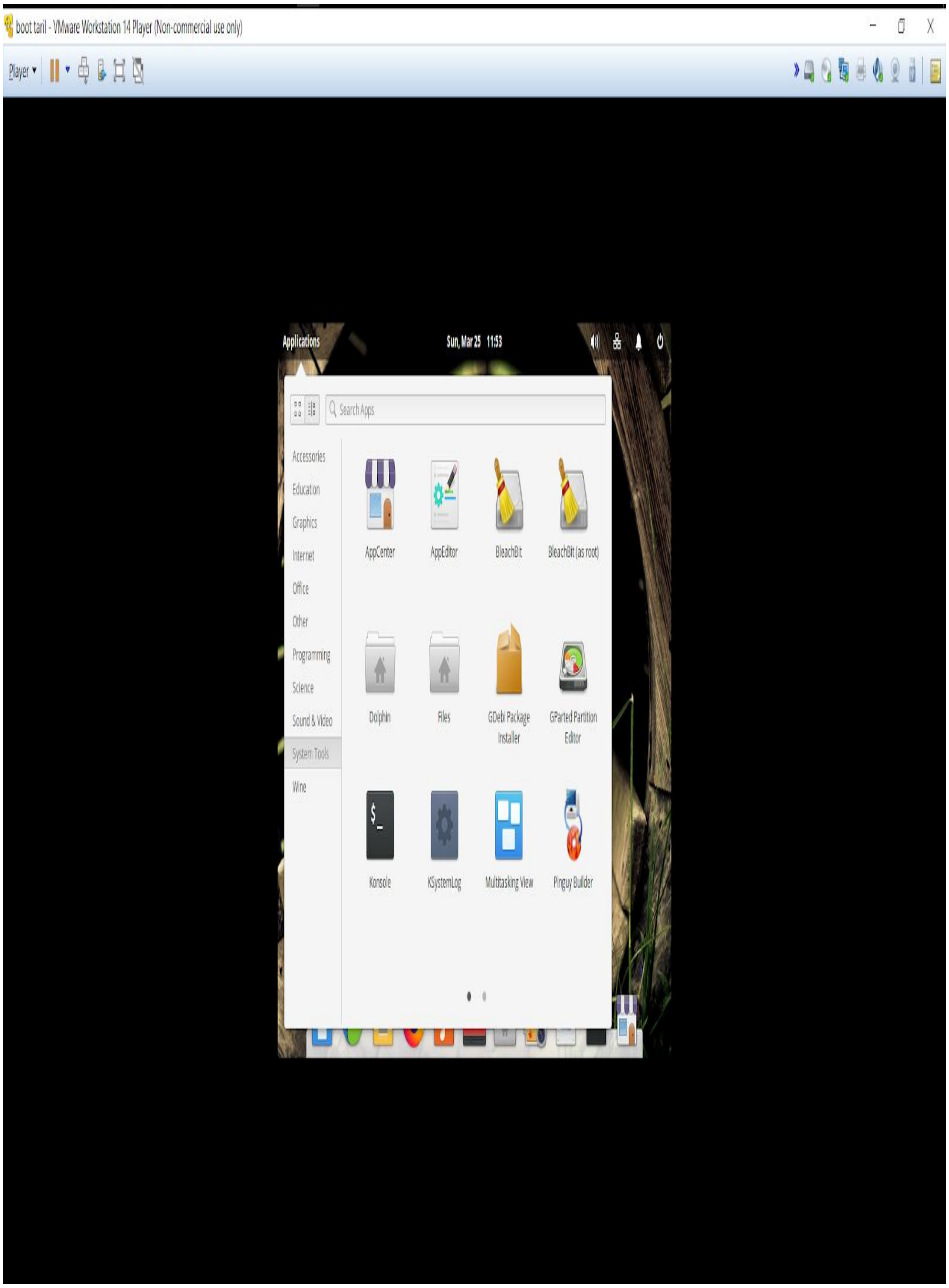

boot taril - VMware Workstation 14 Player (Non-commercial use only)
Player
Applications
Sun, Mar 25 11:53
Search Apps
Accessories
Education
Graphics
Internet
Office
Other
Programming
Science
Sound & Video
System Tools
Wine
Calendar
KMail
KOrganizer
KTnef
LibreOffice 6.0
LibreOffice 6.0 Base
LibreOffice 6.0 Calc
LibreOffice 6.0 Draw
LibreOffice 6.0 Impress
LibreOffice 6.0 Math
LibreOffice 6.0 Writer
Okular

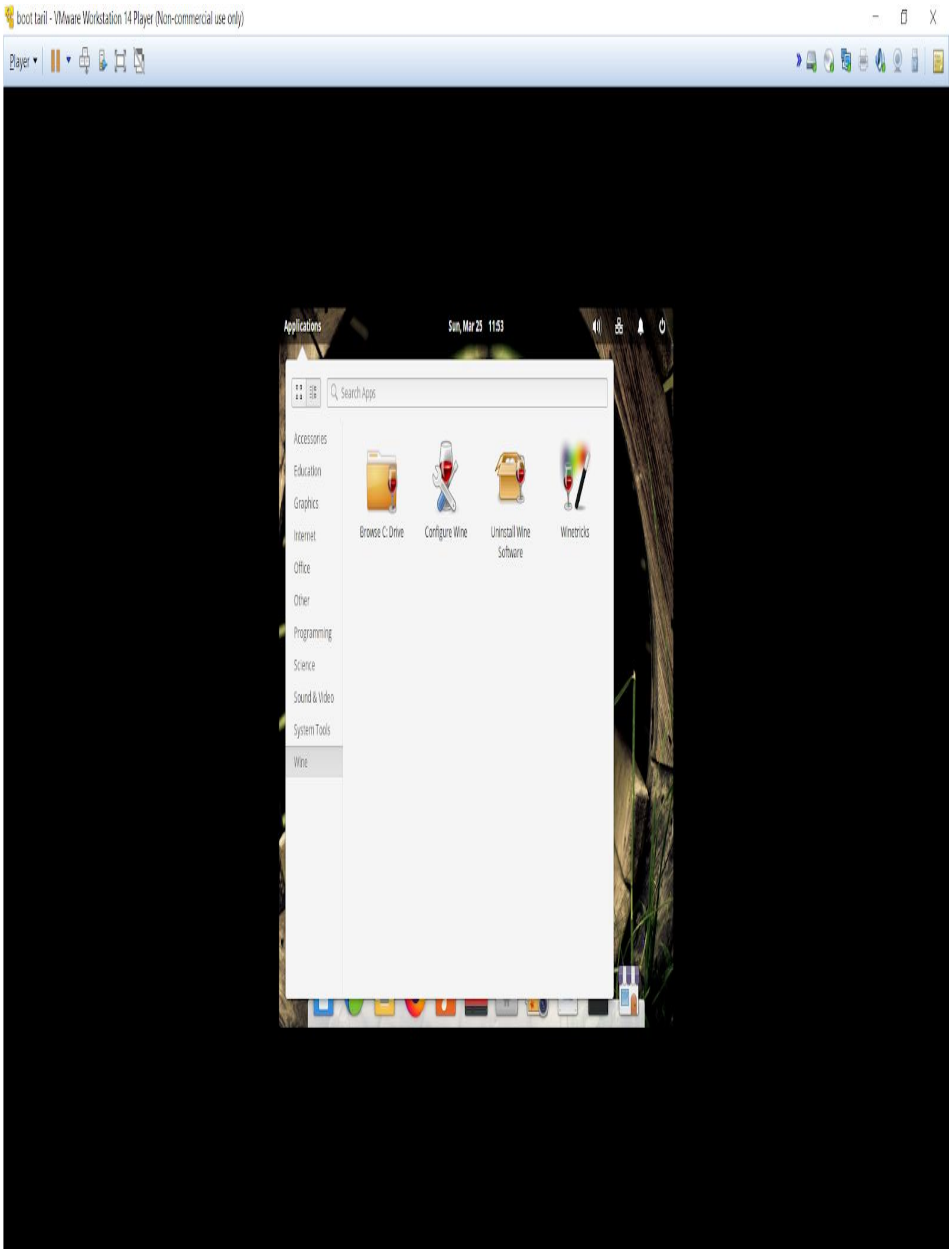

boot taril - VMware Workstation 14 Player (Non-commercial use only)
Player
Applications
Sun, Mar 25 11:53
Search Apps
Accessories
Education
Graphics
Internet
Office
Other
Programming
Science
Sound & Video
System Tools
Wine
Notepad
Scilab
Scilab advanced CLI
Scilab CLI

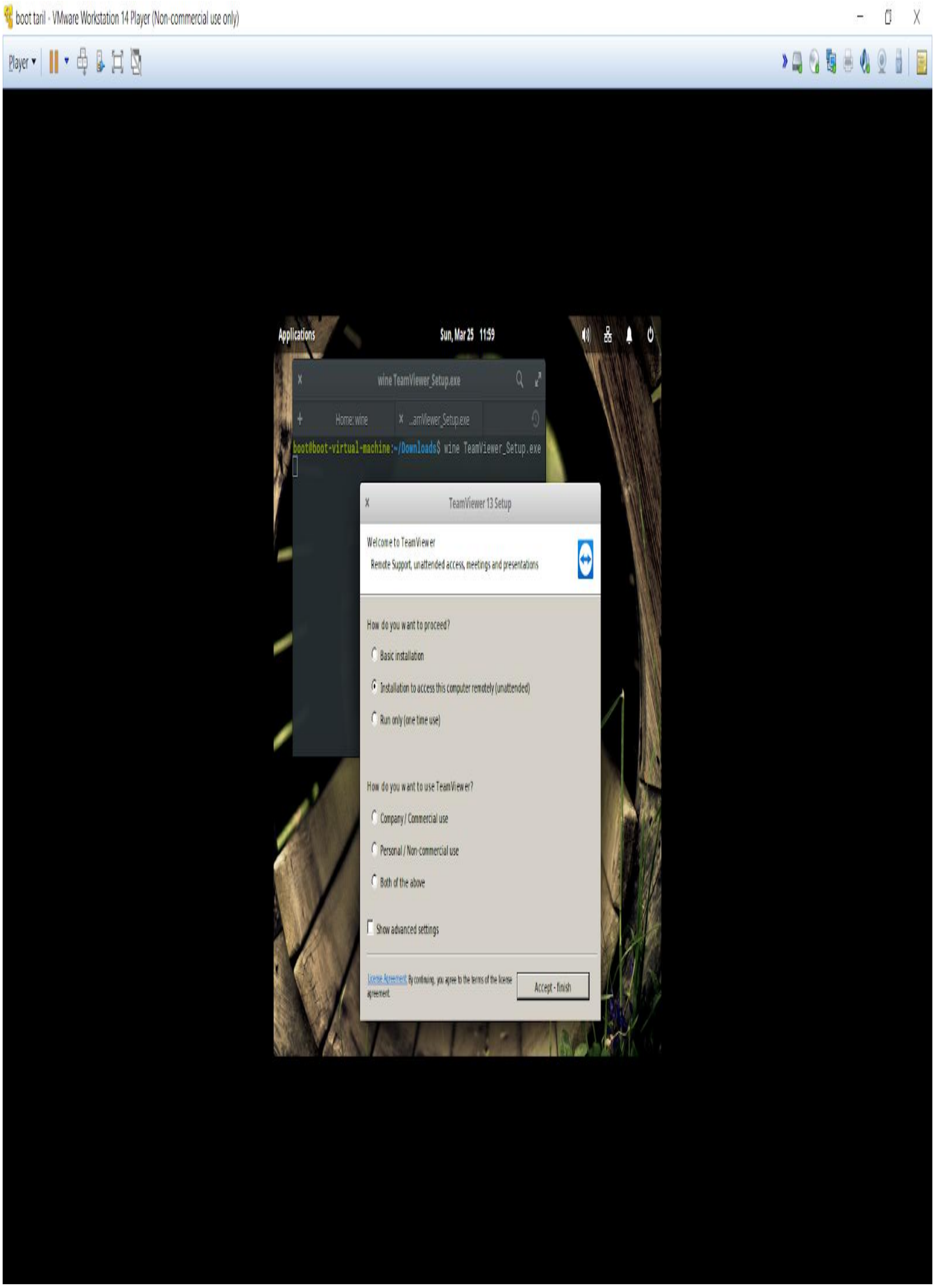

boot taril - VMware Workstation 14 Player (Non-commercial use only)
Player
Applications
Sun, Mar 25 11:53
Search Apps
Accessories
Education
Graphics
Internet
Office
Other
Programming
Science
Sound & Video
System Tools
Wine
ipython


boot taril - VMware Workstation 14 Player (Non-commercial use only)
Player
Applications
Sun, Mar 25 11:53
Search Apps
Accessories
Education
Graphics
Internet
Office
Other
Programming
Science
Sound & Video
System Tools
Wine
LibreOffice 6.0 Math


boot taril - VMware Workstation 14 Player (Non-commercial use only)
Player
Applications
Sun, Mar 25 11:53
Search Apps
Accessories
Education
Graphics
Internet
Office
Other
Programming
Science
Sound & Video
System Tools
Wine
Audacity
Camera
Music
Videos
VLC media player


boot taril - VMware Workstation 14 Player (Non-commercial use only)
Player
Applications
Sun, Mar 25 11:53
Search Apps
Accessories
Education
Graphics
Internet
Office
Other
Programming
Science
Sound & Video
System Tools
Wine
AppCenter
AppEditor
BleachBit
BleachBit (as root)
Dolphin
Files
GDebi Package Installer
GParted Partition Editor
Konsole
KSystemLog
Multitasking View
Pinguy Builder


boot taril - VMware Workstation 14 Player (Non-commercial use only)
Player
Applications
Sun, Mar 25 11:53
Search Apps
Accessories
Education
Graphics
Internet
Office
Other
Programming
Science
Sound & Video
System Tools
Wine
Browse C: Drive
Configure Wine
Uninstall Wine Software
Winetricks


boot taril - VMware Workstation 14 Player (Non-commercial use only)
Player
Applications
Sun, Mar 25 11:59
wine TeamViewer_Setup.exe
Home: wine
...amViewer_Setup.exe

```
boot@boot-virtual-machine:~/Downloads$ wine TeamViewer_Setup.exe
```

TeamViewer 13 Setup

Welcome to TeamViewer

Remote Support, unattended access, meetings and presentations

How do you want to proceed?

- ( ) Basic installation
- (●) Installation to access this computer remotely (unattended)
- ( ) Run only (one time use)

How do you want to use TeamViewer?

- ( ) Company / Commercial use
- ( ) Personal / Non-commercial use
- ( ) Both of the above

☐ Show advanced settings

License Agreement By continuing, you agree to the terms of the license agreement.

Accept - finish

5.3.2 Result Analysis

As expected GITOS has seamlessly integrated the 6 phases of its implementation, it succeeds in being simple to use yet efficient, the system works by being easy in navigation, having a simple layout and keeping the tools accessible, these features, in isolation may only be slightly useful, but when combined each one complements the other and the compounded result is a seamless experience for the user which creates an ease of use which is accompanied with an effective work environment.

5.4 Conclusion

There are a lot of ways to build a customized Linux distribution based on Ubuntu, you may choose any one of them that fits your needs, but don’t forget testing; it’s a very important step in order to insure that your system is error-free and working well on any hardware that a user may have. As expected the designed Operating System will be more efficient and have tremendous graphical user interface which can attract large number of engineering students which will lead to the respective institutions to adapt to it easily. It can be implemented in a large scale and its availability can reduce the hefty work behind the installation of each application by the lab technicians.

6. TESTING AND VALIDATION

6.1 INTRODUCTION

Testing is the stage of implementation, which is aimed at ensuring that the system works accurately and efficiently before live operation commences. The logical design and physical design is thoroughly and continually examined on paper to ensure that they will work when implemented. Thus the system test in implementation was a confirmation that all is correct and an opportunity to show the users that the system works.

There are different types of testing from development phase to closing phase:

Unit Testing

It is done by the developer itself in every stage of the project and fine-tuning the bug and module predicated additionally done by the developer only here we are going to solve all the runtime errors.

Manual Testing

As our Project is academic Leave we can do any automatic testing so we follow manual testing by endeavor and error methods

Deployment of System

Once the project is total yare we will come to deployment of client system in genuinely world as its academic leave we did deployment i our college lab only with all need Software's with having Windows OS.

6.2 DESIGN OF TEST CASES AND SCENARIOS

TC ID	Module Name	Sub module	Test Cases	Steps	Test Data	Expected Result	Pass / Fail
1	GITOS	Booting	To verify when user clicks on the power button.	1.Click on power button	-	User when clicks on power button pc should enter into boot	Pass
2	GITOS	Booting	UI of the boot screen should be as per design	-	-	The UI should be accurate as per design	Pass
3	GITOS	Password screen(new user)	User should be able to enter the new password	Boot>password	valid password	User should be able to confirm new password	Pass
4	GITOS	Desktop (Home Screen)	User should be able to enter to home screen	Boot>password >desktop	-	User after confirming the password, enter to desktop	Pass
5	GITOS	Password screen	User enters the invalid password	Boot>password >desktop	invalid password	User should be able to enter to home screen after invalid password	Fail
6	GITOS	Desktop	To verify UI of the Desktop	Boot>password >desktop	valid password	UI of should be as per design	Pass

7	GITOS	Desktop	To verify there are no crash reports.	Boot>password >desktop		User should not be able to report the crashes	Pass
8	GITOS	Desktop	To verify user is able to open the applications	Boot>password >desktop		When should be able to open the applications	Pass
9	GITOS	Desktop	To verify user is able to make changes in settings	Boot>password >desktop		User should be able to make changes in settings	Pass
10	GITOS	Desktop	To verify free flow of multiple applications	Boot>password >desktop		PC should be able to handle the multiple tasks	Pass
11	GITOS	Desktop	UI of screen			UI design	Pass

Login Validation Table

TC_ID	Module Name	Sub module	Test Cases	Steps	Test Data	Expected Result	Pass/Fail
1	GITOS	login	To verify user enters valid password	Boot>Password	valid password	User should be able to login	Pass
2	GITOS	login	To verify when user enters wrong password	Boot>Password	Invalid password	User should not be able to login	Pass
3	GITOS	Desktop	User is able to open the application	Boot>Password		User should be able to open the applications	Pass
4	GITOS	shut down	To verify functionality of shut down			User when clicks on shut down the computer should log off	Pass

6.3 Validation

- .As a new user, should register with the password after the boot.
- The password should contain minimum of six characters.
- User should be able to register with valid password pattern.
- User should not have permission to access when invalid password is entered.
- User have access to all the applications.
- Admin has the access to make changes in the system.
- The read and write of the files should be accessed by user.

6.4 Conclusion

As expected the designed Operating System will be more efficient and have tremendous graphical user interface which can attract large number of engineering students which will lead to the respective institutions to adapt to it easily. It can be implemented in a large scale and its availability can reduce the hefty work behind the installation of each application by the lab technicians.

7. Conclusion:

As expected the designed Operating System will be more efficient and have an efficient graphical user interface which can assist a large number of engineering students which will help the respective institutions to adapt to it easily. It can be implemented in a large scale and its availability can reduce the hefty work behind the installation of each application.

Since GITOS is easy to adapt and modify, it shall see quick changes as per the suggestions of its users, since the project is open source and free, the engineering user base may create any necessary modifications or change, which they feel might improve the OS, truly it is an OS “of the engineers, by the engineers and for the engineers”.

REFERENCES:

1. **https://susestudio.com/**
2. **https://en.wikipedia.org/wiki/Operating_system**
3. **https://en.wikipedia.org/wiki/GNOME**
4. **https://www.gnome.org/gnome-3/**
5. **http://swecha.org/**
6. **https://en.wikipedia.org/wiki/Elementary_OS**