# Linux Kernel Optimization Package

**BY**

**Md.Habibur Rahman**
**ID: 103-15-1106**

This Project Report Presented in Partial Fulfillment of the Requirements for the Degree of Bachelor of Science in Computer Science and Engineering

**Supervised By**

**Md Jabed Morshed Chowdhury**
**Senior Lecturer**
**Department of CSE**
**Daffodil International University**


**DAFFODIL INTERNATIONAL UNIVERSITY**
**DHAKA, BANGLADESH**
**MAY, 2014**

# APPROVAL

This Project titled **"Linux Kernel Optimization Package**" submitted by Md.Habibur Rahman to the Department of Computer Science and Engineering, Daffodil International University, has been accepted as satisfactory for the partial fulfillment of the requirements for the degree of B.Sc. in Computer Science and Engineering and approved as to its style and contents. The presentation has been held on 11th May, 2014.

## BOARD OF EXAMINERS

_____

**Dr. Syed Akhter Hossain** — **Chairman**
**Professor and Head**
Department of Computer Science and Engineering
Faculty of Science & Information Technology
Daffodil International University

_____

**Dr. Sheak Rashed Haider Noori** — **Internal Examiner**
**Assistant Professor**
Department of Computer Science and Engineering
Faculty of Science & Information Technology
Daffodil International University

_____

**Anisur Rahman** — **Internal Examiner**
**Assistant Professor**
Department of Computer Science and Engineering
Faculty of Science & Information Technology
Daffodil International University

_____

**Dr. Muhammad Shorif Uddin** — **External Examiner**
**Professor**
Department of Computer Science and Engineering
Jahangirnagar University

# DECLARATION

We hereby declare that, this project has been done us me under the supervision of **Md Jabed Morshed Chowdhury, Senior Lecturer, Department of CSE** Daffodil International University. We also declare that neither this project nor any part of this project has been submitted elsewhere for award of any degree or diploma.

**Supervised by:**

_____
**Md Jabed Morshed Chowdhury**
**Senior Lecturer**
**Department of CSE**
**Daffodil International University**

**Submitted by:**

_____
**MD.Habibur Rahman**
ID: 103-15-1106
Department of CSE
Daffodil International University

# ACKNOWLEDGEMENT

First we express our heartiest thanks and gratefulness to almighty Allah for His divine blessing makes me possible to complete this project successfully.

Then I feel grateful to our profound indebtedness to **Md Jabed Morshed Chowdhury, Senior Lecturer,** Department of CSE Daffodil International University, Dhaka. Deep knowledge & keen interest of our supervisor in the field of web application influenced us to carry out this project. His endless patience, scholarly guidance, continual encouragement, constant and energetic supervision, constructive criticism, valuable advice, reading many inferior draft and correcting them at all stage have made it possible to complete this project.

I also would like to express my heartiest gratitude to **Dr. Syed Akhter Hossain, Professor and Head**, Department of CSE, Daffodil International University for his kind help to finish my project and also to other faculty member and staff of CSE department of Daffodil International University.

I would like to thank all the class mates in Daffodil International University, who took part in this, discuss while completing the course work.

Finally, I must acknowledge with due respect the constant support and patients of our parents in completing this report.

# ABSTRACT

In operating system, the kernel is a computer program that manages input/output requests from software and translates them into data processing instructions for the central processing unit and other electronic components of a computer. The kernel is a fundamental part of a modern computer's operating system.

Now a day, computing devices are diverse ranging from heavy duty servers to hand held devices like Smart-phone and PDA. Different types of devices have their own types of hardware and specification. In this scenario, general purpose operating system is not suitable for all types of devices. Furthermore, different companies also used their own proprietary hardware. For better performance of the devices, customization of the kernel is a good approach. In this project, a software is developed to compile and build customized kernel based on the processing power and memory of the device.

# TABLE OF CONTENTS

# LIST OF FIGURES

# CHAPTER 1
# INTRODUCTION

## 1.1 Background of the Project

Linux Kernel Optimization Package is a application software that auto configure and build linux kernel from source code according current hardware is present on system. The main point of developing this application is to help users to make their linux base machines more faster. The project is developing because; many linux distributions provide big stock kernel almost have all devices drivers with their distributions. But every user not use all devices in a single PC. This big kernel slow the machine and use more ram. For that we need to compile the linux kernel to make our machines faster. This job is very difficult to do for a end users.

So, Linux kernel optimization package will develop to help the users to compile linux kernel on their PC according the PC needs. This application makes easy to compile the linux kernel for increase the performance of PC's,make the kernel as small as possible to minimize its memory usage, support for the latest networking features.

## 1.2 Reasons for Selecting this Project

We have observed that GNU/Linux operating system is being most popular operating system in operating system revolution,because of its freedom. Linux kernel is the life force of all Linux family of operating systems including Ubuntu, CentOS, and Fedora.If we are running Linux on a system with hardware or wish to use features not supported in the stock kernels, or perhaps we wish to reduce the kernel memory footprint to make better use of our system memory, we may find it necessary to build your own custom kernel.

When a user use any linux base distribution they have big kernel with it. That contains lots of device drivers,modules and a user only use few drivers. Rest of these drivers are make the machine slow to boot up and use much more ram. If users want to make their pc speed up with their limited resources they needs to build custom kernel for their pc and its a difficult job to do.

So I try to solve this problem by building a application that handle all job that need to build custom kernel for a user PC's.

### 1.3 Benefits of the Project

- Users can configure, compile and build linux kernel very easily.
- Reduce the kernel size
- Use less RAM than provided kernel with distributions
- Support for new devices and old devices
- Boot up the machines fast
- Increase the performance of the system.

### 1.4 Methodology to be Used

Methodology can be -

- A systematic study of the existing procedures that are used to build custom linux kernel.
- Analyze existing systems and document the necessary extensions required to develop.
- A documented process for management of projects that contains procedures, definitions and explanations of techniques used to collect, store, analyze and present information as part of a research process in a given discipline[1].

Since i'm re using a lot of re-usable components and the possibilities of upgrades are countless compared to short time i have followed the RAD methodology. Rapid application development is a software development methodology that involves methods like iterative development and software prototyping. According to Whitten, it is a merger of various structured techniques, especially data-driven Information Engineering, with prototyping techniques to accelerate software systems development. This application designed using the Waterfall Development Methodology then implemented and tested on real project

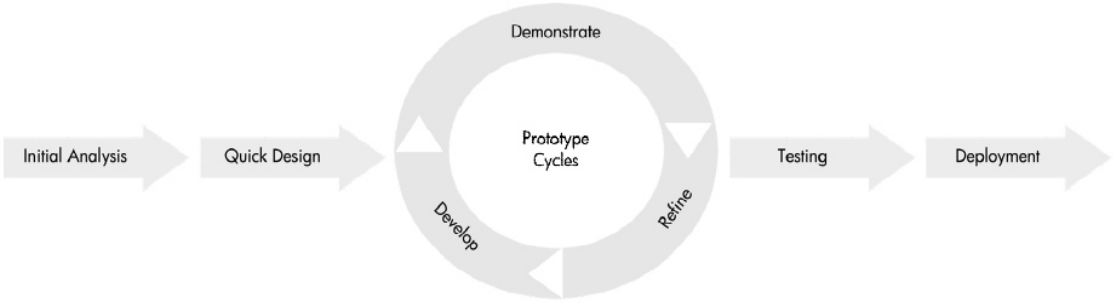


Figure 1.1 : RAD Methodology, Adapted from[2].

## 1.5 Summary

The Linux kernel optimization Package can boost the quality of building custom kernel and make comfortable to user to use their PC's. Also increase the performance of PC's,make the kernel as small as possible to minimize its memory usage, support for the latest networking features.

# CHAPTER 2
# BACKGROUND STUDY

## 2.1 Introduction

A background study is a preliminary analysis which is prepared to determine the relative environmental impacts associated with a proposed project.

## 2.2 Project planning

The project initiation plan is the start planning for a project. I am only a group members for the project. The project initiation planning includes analysis of scope of project, identifying lake of existing system. I have set the timeline and responsibilities to complete the project at due time. My supervisor had given proper instruction and information that was very helpful for the work. Then i started my project. When i face any problem afterheat i communicated to supervisor and solved the problem.

Now we are mostly dependent on linux base system. Because of free and freedom of it. Whenever we have limited resources and have lots of job to do at time. So we need faster machines. Here is chance to make our pc more faster by build custom linux kernel.

## 2.3 About Linux Kernel

The Linux kernel is a Unix-like operating system kernel used by a variety of operating systems based on it, which are usually in the form of Linux distributions.The Linux kernel is a prominent example of free and open source software. The Linux kernel is released under the GNU General Public License version 2 (GPLv2) (plus some firmware images with various non-free licenses), and is developed by contributors worldwide. Day-to-day development discussions take place on the Linux kernel mailing list.

The Linux kernel was initially conceived and created in 1991 by Finnish computer science student Linus Torvalds. Linux rapidly accumulated developers and users who adapted code from other free software projects for use with the new operating system. The Linux kernel has received contributions from thousands of programmers[3].

### 2.4 Linux Kernel Optimization

Linux kernel optimization or customization is the way to make the kernel perfect for specific computers.Add or remove features.Why would we even want to do this? For a number of reasons: to get extra functionality, to weed out unneeded features, to try for better performance, to help with testing new patches, or just because we want to know how. The last two reasons are the best ones.

Most general-purpose Linux distributions ship with kernels and sets of modules that try to please everyone and support all the hardware in the world, so we end up with a couple hundred megabytes of kernel + modules fattening our system. This raises security concerns, and some things are just plain silly such as support for infrared, ham radio, and bales of laptop-specific crud on a desktop system, so many useless features[4].

### 2.5 Linux kernel optimization process

Building custom kernel from source packages is usually a three step process: configure, build, and install. The Linux kernel is configured with the command "make menuconfig", built with the command "make", and installed either manually or with the command "make install"[5]

#### 2.5.1 Get and extract the source

The Linux kernel source code is distributed by kernel.org as tar archives. Grab the most recent "stable" releases from the directory. Then extract this archive with the command "tar xvjf linux-3.x.*.tar.bz2". (Type the command "man tar" for more information on the tar command.) Then cd into the directory created by extracting the archive.

To obtain other Linux kernel versions (such as development releases and kernels supplied by distributions) are also their.

#### 2.5.2 Configuring the kernel

Before we can build the kernel, we have to configure it. Configuring selects which features this kernel build should include, and specifies other technical information such as buffer sizes and optimization strategies. This information is stored in a file named

".config" in the top level directory of the kernel source code. To see the various user interfaces to the configuration system, type "make help".

Note that "make clean" does not delete configuration information, but the more thorough "make distclean" does.

Often when building a kernel, an existing .config file is supplied from elsewhere. Copy it into place, and optionally run "make oldconfig" to run the kernel's diagnostics against it to ensure it matches the kernel version you're using, updating anything that's out of sync.

Several preset configurations are shipped with the kernel source code. Run the command find . -name "*_defconfig" in the kernel source directory to seem them all. Any of these can be copied to .config and used as a starting point.

The kernel can also automatically generate various configurations, mostly to act as starting points for customization:

make defconfig - Set all options to default values

make allnoconfig - Set all yes/no options to "n"

make allyesconfig - Set all yes/no options to "y"

make allmodconfig - Set all yes/no options to "y" and all "yes/module/no" options to "m"

make randconfig - Set each option randomly (for debugging purposes).

make oldconfig - Update a .config file from a previous version of the kernel to work with the current version.

make localmodconfig – Set all modules that currently connected with the system

The most common user interface for configuring the kernel is menuconfig, an interactive terminal based menuing interface invoked through the makefiles via "make menuconfig". This interface groups the configuration questions into a series of menus, showing the current values of each symbol and allowing them to be changed in any order. Each symbol has associated help text, explaining what the symbol does and where to find more information about it. This help text is also available as html.[6]

### 2.5.3 Building the kernel

Now we are ready to start the build. we can speed up the compilation process by enabling parallel make with the -j flag. The recommended use is 'processor cores + 1′, e.g. 5 if we have a quad core processor:

```
make -j5
```

This will compile the kernel and create a compressed binary image of the kernel. After the first step, the kernel image can be found at arch/i386/boot/bzImage (for a x86 based processor). Once the initial compilation has completed, install the dynamically loadable kernel modules:

```
sudo make modules_install
```

The modules are installed in a subdirectory of "/lib/modules", named after the kernel version. The resulting modules have the suffix ".ko".

Instead of the compilation process of above, we can alternatively compile the kernel as installable .deb packages(for debian base system). This improves the portability of the kernel, since installation on a different machine is as simple as installing the packages. Rather than using 'make' and 'make install', we use 'make-kpkg':

```
fakeroot make-kpkg – -initrd – -append-to-version=-some-string-here kernel-
image kernel-headers
```

Unlike above, you cannot enable parallel compilation with make-kpkg using the -j flag. Instead, define the CONCURRENCY_LEVEL environment variable.

```
export CONCURRENCY_LEVEL=3
```

### 2.5.4 Install the Kernel

Once the compilation has completed, we can install the kernel with universal or debian sytem way.

Universal way:

```
sudo make install
```

Debian way:

```
sudo dpkg -i  *.deb
```

## 2.6 Summary

Initial study confirmed me about the necessity of the proposed project.That is why i have decided to develop my project name "**Linux kernel optimization package**" that will provide complete guidelines for the user to build custom kernel from linux kernel source easly.I'm sure that users are benefited from this application. User make their machine more faster than before.

# CHAPTER 3
# SYSTEM ANALYSIS & FEASIBILITY STUDY

## 3.1 System Analysis

The system analysis is a detailed study of the various operations performed by the existing system and their relationships within and outside of the system. One aspect of analysis is defining the boundaries of the system and determining whether a candidate system should consider other related systems. Here i completed system analysis by the input analysis and output analysis of existing system. At the preliminary stage of the analysis, we had followed the Waterfall Development Methodology[7].

### 3.1.1 Waterfall Model

The waterfall model is a sequential software development process, in which progress is seen as flowing steadily downwards through the phases of Conception, Initiation, Analysis, Design, Construction, Testing and Maintenance [8].

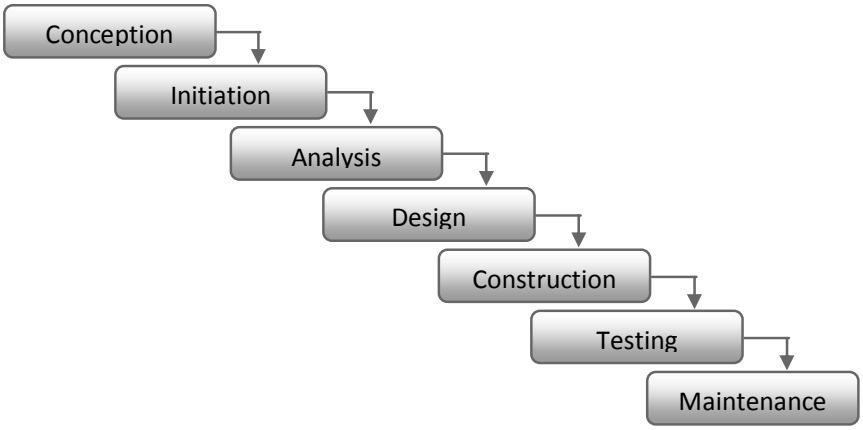


Figure 3.1: Waterfall Model

### 3.1.2 Conception

The kernel optimization idea came from the limited resources of pc's and the freedoms of open software.

### 3.1.3 Analysis

Software analysis patterns or analysis are conceptual models, which capture an abstraction of a situation that can often be encountered in modeling. Using specific analysis, we will do a system-risk-analysis. Based on the results of this analysis, we will define a test concept adapted to the risk requirements [9].

### 3.1.4 Design

A design pattern is a general reusable solution to a commonly occurring problem in software design. A design pattern is not a finished design that can be transformed directly into code. In this section, we will work for how to solve a problem that can be used in many different situations.

### 3.1.5 Construction

Construction means the implementation or the realization of an application, or execution of a plan, idea, model, design, specification, standard, algorithm, or policy.

In computer science, a construction is a realization of a technical specification or algorithm as a program, software component. Many implementations may exist for a given specification or standard. So, in this phase we will build up our application.

### 3.1.6 Testing

Software testing is an investigation about the quality of the product or service under test. Software testing can also provide an objective, independent view of the software to allow the business to appreciate and understand the risks of software implementation. Test techniques include, but are not limited to the process of executing a program or application with the intent of finding software bugs (errors or other defects).

Several testing types are available:

- Black Box Testing
- White Box Testing
- Alpha Testing

- Beta Testing
- Software Application Testing

### 3.1.7 Maintenance

Maintenance is the modification of a software product after delivery to correct faults, to improve performance or other attributes. We will be responsible to solve for all kinds of software faults after delivery.

## 3.2 Feasibility Study

A feasibility study is an evaluation of a proposal designed to determine the difficulty in carrying out a designated task. Generally, a feasibility study precedes technical development and project implementation [10].

Some common factors are referred in feasibility study. These are as follows:

- Technical Feasibility
- Software Availability
- Hardware Availability

### 3.2.1 Technical Feasibility

Technical Feasibility is the process of proving that the concept is technically possible. The objective of Technical Feasibility step is to confirm that the product will perform and to make sure that there are no production barriers.

Technical feasibility is carried out to determine whether the company has the capability, in terms of software, hardware, personnel and expertise, to handle the completion of the project.

### 3.2.2 Software Availability

For implementing the project a couple of software is needed. First of all, we need GNU/Linux base system and that have Qt runtime software which run our application.

At present i use the Qt 5.2.1.

### 3.2.3 Hardware Availability

To maintain Linux optimization package needs to have an Internet facility and strong pc hardware configuration for quick compilation of kernel.

### 3.3 Summary

The trend and feasibility study suggest the need of a new information system to keep pace with the modern world. A system is never quite infeasible. It is often more or less feasible from different perceptions and perspectives. The earlier sections discussed all the feasibility checkpoints and respective problems and conditions. The project is extremely strong technically,

# CHAPTER 4
# SYSTEM DESIGN

## 4.1 Introduction

Systems design is the process of defining the architecture, components, modules, interfaces, and data for a system to satisfy specified requirements. All entities of the system have been modeled as objects in order to separate their logic from presentational code [11].

## 4.2 Flowchart

A flowchart is a type of diagram that represents an algorithm or process, showing the steps as boxes of various kinds, and their order by connecting these with arrows. This diagrammatic representation can give a step-by-step solution to a given problem. Process operations are represented in these boxes, and arrows connecting them represent flow of control. Data flows are not typically represented in a flow chart, in contrast with data flow diagrams, rather, they are implied by the sequencing of operations. Flowcharts are used in analyzing, designing, documenting or managing a process or program in various fields [12].

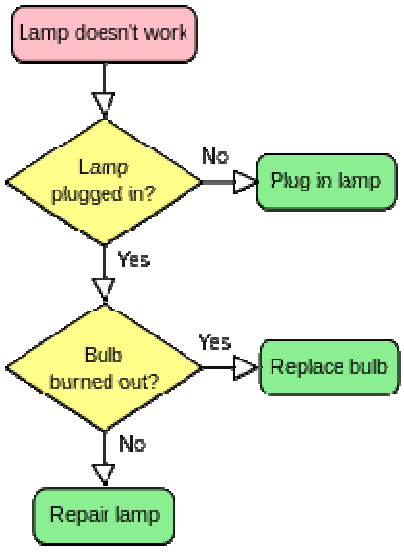


Figure 4.1:Flowchart For Lamp, Adapted from[13]

### 4.2.1 Overview

Flowcharts are used in designing and documenting complex processes or programs. Like other types of diagrams, they help visualize what is going on and thereby help the people to understand a process, and perhaps also find flaws, bottlenecks, and other less-obvious features within it. There are many different types of flowcharts, and each type has its own repertoire of boxes and notational conventions. The two most common types of boxes in a flowchart are:

a processing step, usually called activity, and denoted as a rectangular box
a decision, usually denoted as a diamond.
A flowchart is described as "cross-functional" when the page is divided into different swim lanes describing the control of different organizational units. A symbol appearing in a particular "lane" is within the control of that organizational unit. This technique allows the author to locate the responsibility for performing an action or making a decision correctly, showing the responsibility of each organizational unit for different parts of a single process.

Flowcharts depict certain aspects of processes and they are usually complemented by other types of diagram. For instance, Karakul Ishikawa defined the flowchart as one of the seven basic tools of quality control, next to the histogram, Pareto chart, check sheet, control chart, cause-and-effect diagram, and the scatter diagram. Similarly, in UML, a standard concept-modeling notation used in software development, the activity diagram, which is a type of flowchart, is just one of many different diagram types.[12]

Nazi-Shneiderman diagrams and Dragon-charts are an alternative notation for process flow.

Common alternative names include: flowchart, process flowchart, functional flowchart, process map, process chart, functional process chart, business process model, process model, process flow diagram, work flow diagram, business flow diagram

### 4.2.2 Use of Flowchart

Flowcharts used to be a popular means for describing computer algorithms and are still used for this purpose. Modern techniques such as UML activity diagrams and Dragon-charts can be considered to be extensions of the flowchart. In the 1970s the popularity of flowcharts as an own method decreased when interactive computer terminals and third-generation programming languages became the common tools of the trade, since algorithms can be expressed much more concisely as source code in such a language, and also because designing algorithms using flowcharts was more likely to result in spaghetti code because of the need for go to describe arbitrary jumps in control flow. Often pseudo-code is used, which uses the common idioms of such languages without strictly adhering to the details of a particular one.

### 4.2.3 Types of flow Chart

Stern Eckert (2003) suggested that flowcharts can be modeled from the perspective of different user groups (such as managers, system analysts and clerks) and that there are four general types:

Document flowcharts, showing controls over a document-flow through a system
Data flowcharts, showing controls over a data-flow in a system
System flowcharts showing controls at a physical or resource level
Program flowchart, showing the controls in a program within a system
Notice that every type of flowchart focuses on some kind of control, rather than on the particular flow itself.

### 4.2.4 Project Flow Chart

Flow charts are easy-to-understand diagrams showing how steps in a process fit together. This makes them useful tools for communicating how processes work, and for clearly documenting how a particular job is done. Furthermore, the act of mapping a process out in flow chart format helps Users clarify your understanding of process, and helps users think about where the process can be improved.

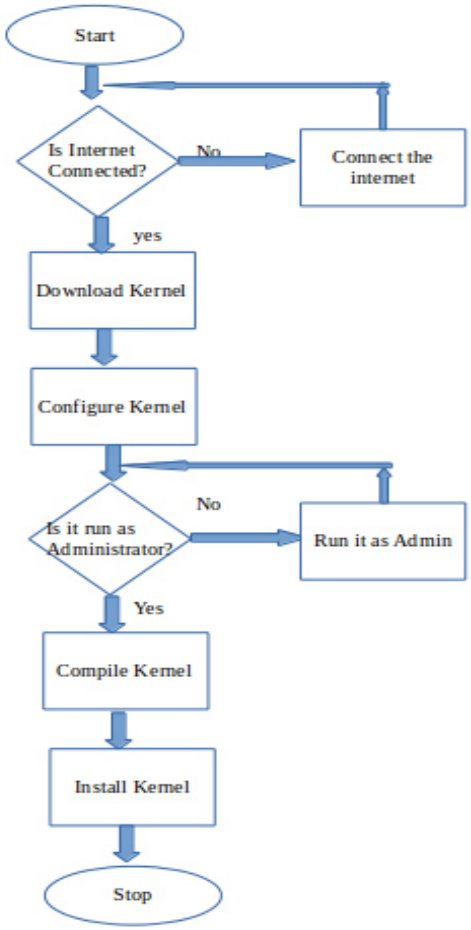


Figure 4.2: Flowchart For Kernel Optimization

### 4.3 Use Case Diagram

A use case diagram at its simplest is a representation of a user's interaction with the system and depicting the specifications of a use case. A use case diagram can portray the different types of users of a system and the various ways that they interact with the system. This type of diagram is typically used in conjunction with the textual use case and will often be accompanied by other types of diagrams as well [13].

In the proposed system there only one types User. User can access three functionality file, actions and Help. Actions functionality is mandatory for user to build the custom kernel and other two are optional. Inside action there are have two option 'start compile' and 'stop compiling'. In File option user can view log of this application, when it run, error message, build message etc. User also get operating system and hardware information of system using File option. By using Help option user can see the manual to use this application,can report bug to the community and also know the information of this application(author,developer,license).

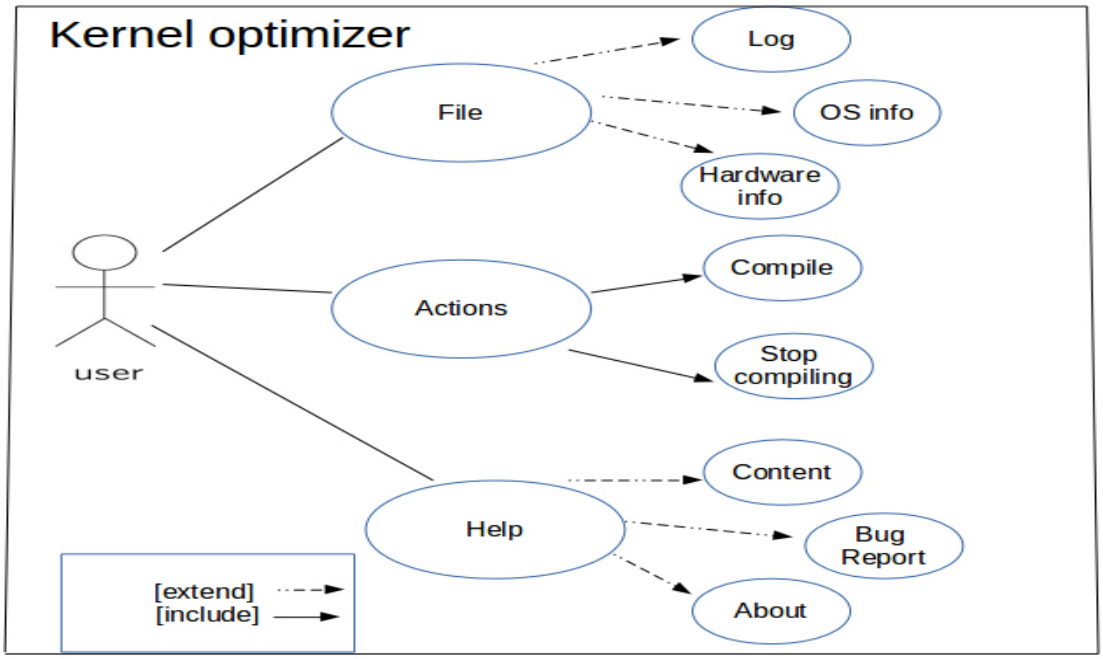


Figure 4.3: Use Case Diagram Of User

# CHAPTER 5
# DEVELOPMENT & TESTING

## 5.1 Introduction

The objective of interface design is to give the detail view of the interface that is how the each modules interface is designed. How the input and output methods are connected to the interface etc. are analyzed here.

## 5.2 Software Development

Software development is the development of a software product. The term "software development" may be used to refer to the activity of computer programming, which is the process of writing and maintaining the source code, but in a broader sense of the term it includes all that is involved between the conception of the desired software through to the final manifestation of the software, ideally in a planned and structured process. Therefore, software development may include research, new development, prototyping, modification, reuse, re-engineering, maintenance, or any other activities that result in software products [14].

First of all i made a sketch diagram of the overall procedure in our mind of proposed project. Then i divide the whole work into different parts to make it simple and easy. After that i select the suitable platform include language. Finally i design and developed the application.

## 5.3 Development Platform & languages

To develop proposed project I had used Qt platform to design and build the application interface and control flow.In this application I used to different languages for both front-end and backend. 'Qt c++' language is used for front-end design and control and used 'Bash shell' script for backend operations.

### 5.3.1 About Qt Platform

Qt is a cross-platform application framework that is widely used for developing application software with a graphical user interface (GUI) (in which cases Qt is classified as a widget toolkit), and also used for developing non-GUI programs such as command-line tools and consoles for servers.

Qt uses standard C++ but makes extensive use of a special code generator (called the Meta Object Compiler, or moc) together with several macros to enrich the language. Qt can also be used in several other programming languages via language bindings. It runs on the major desktop platforms and some of the mobile platforms. It has extensive internationalization support. Non-GUI features include SQL database access, XML parsing, thread management, network support, and a unified cross-platform application programming interface (API) for file handling.

Qt is available under a commercial license, GPL v3 and LGPL v2. All editions support many compilers, including the GCC C++ compiler and the Visual Studio suite.

Qt is developed by Digia, who owns the Qt trademark, and the Qt Project under open governance, involving individual developers and firms working to advance Qt. Before the launch of the Qt Project, it was produced by Nokia's Qt Development Frameworks division, which came into existence after Nokia's acquisition of the Norwegian company Trolltech [15].

### 5.3.2 About Bash Shell

Bash is a Unix shell written by Brian Fox for the GNU Project as a free software replacement for the Bourne shell (sh). Released in 1989,it has been distributed widely as the shell for the GNU operating system and as a default shell on Linux and Mac OS X. It has been ported to Microsoft Windows and distributed with Cygwin and MinGW, to DOS by the DJGPP project, to Novell NetWare and to Android via various terminal emulation applications.

Bash is a command processor, typically run in a text window, allowing the user to type commands which cause actions. Bash can also read commands from a file, called a script. Like all Unix shells, it supports filename wildcarding, piping, here documents, command substitution, variables and control structures for condition-testing and iteration.The keywords, syntax and other basic features of the language were all copied from sh[16].

## 5.4 Black Box Testing of Developed System

Black Box Testing treats an application as a black box and only looks at the outputs that are produced by specific inputs into the application. The black box tester does not need to understand why the code does what it does, and they should not have access to the source code of the application. Requirements are used to determine the correct outputs of black box testing, and these test cases are used to validate that the right software is being built[17].

### 5.5 Application Interface overview

There is the interface overview of proposed project "**Linux kernel Optimization package**".

#### 5.5.1 Main window

It is the front/main view of proposed project application.In this interface user can do all operations like compile the kernel,stop compilation,view log history,os and hardware informations etc.

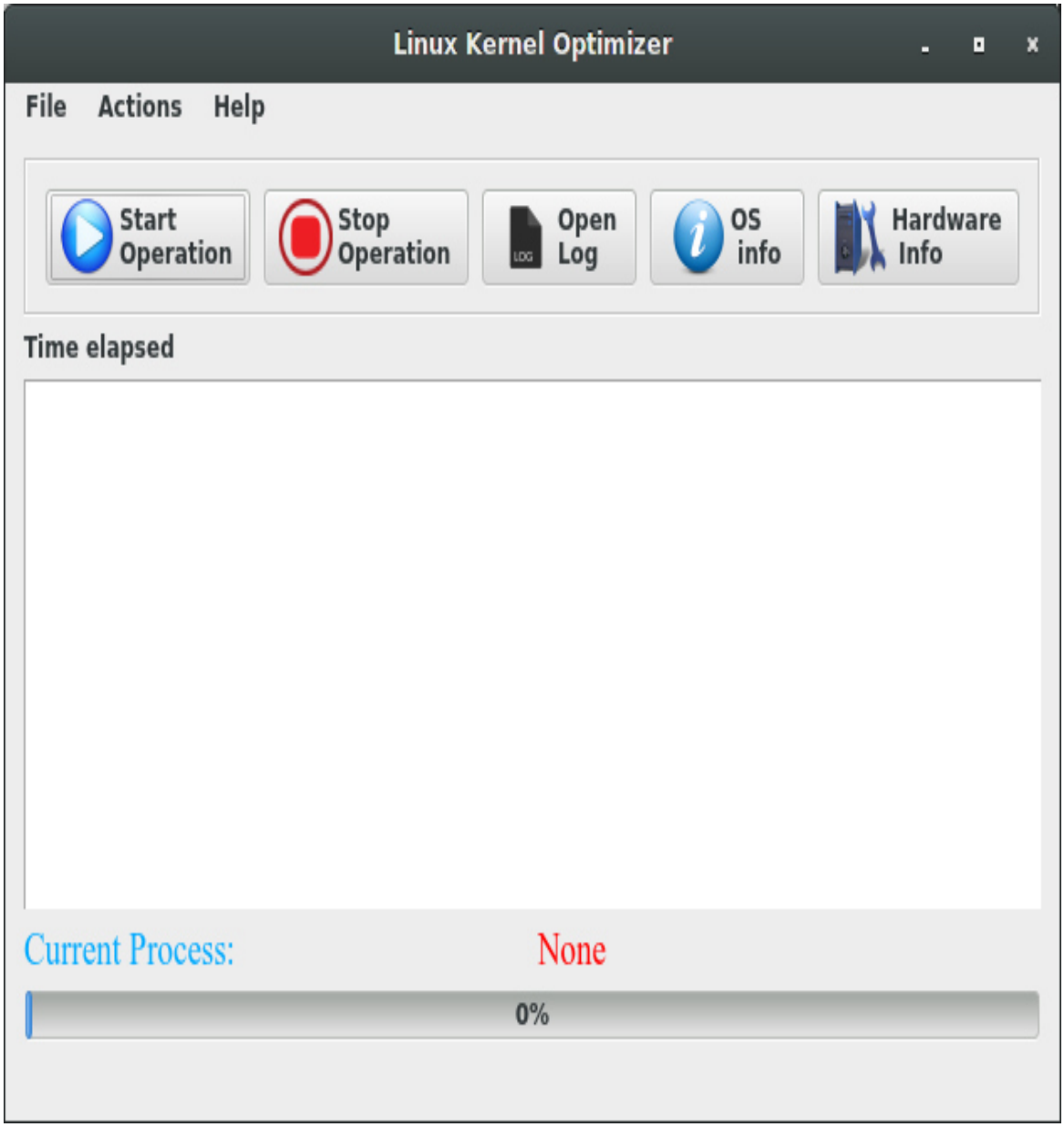


Figure 5.1 : Main Window

### 5.5.2 Main window with running process

After press the start opearation button the kernel compilation process start.First is download the kernel and extract it.In this below figure is represent the kernel extracting process output.

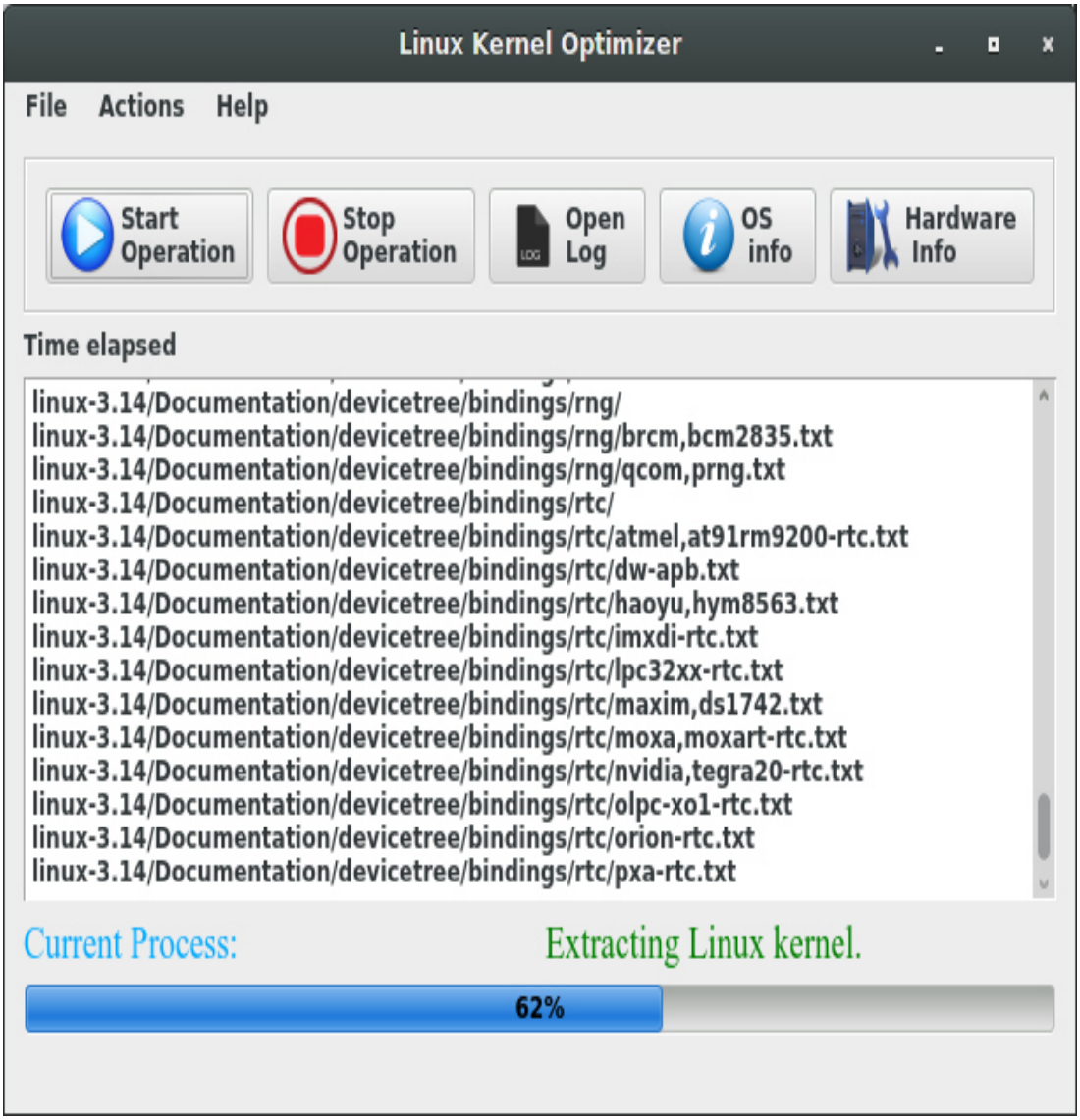


Figure 5.2 : Main Window With Running Process

### 5.5.3 Log view window

After press the log view option below window will come up. This windows view the log as history of the application. Like when it ran, which job are done before, error or success message etc.

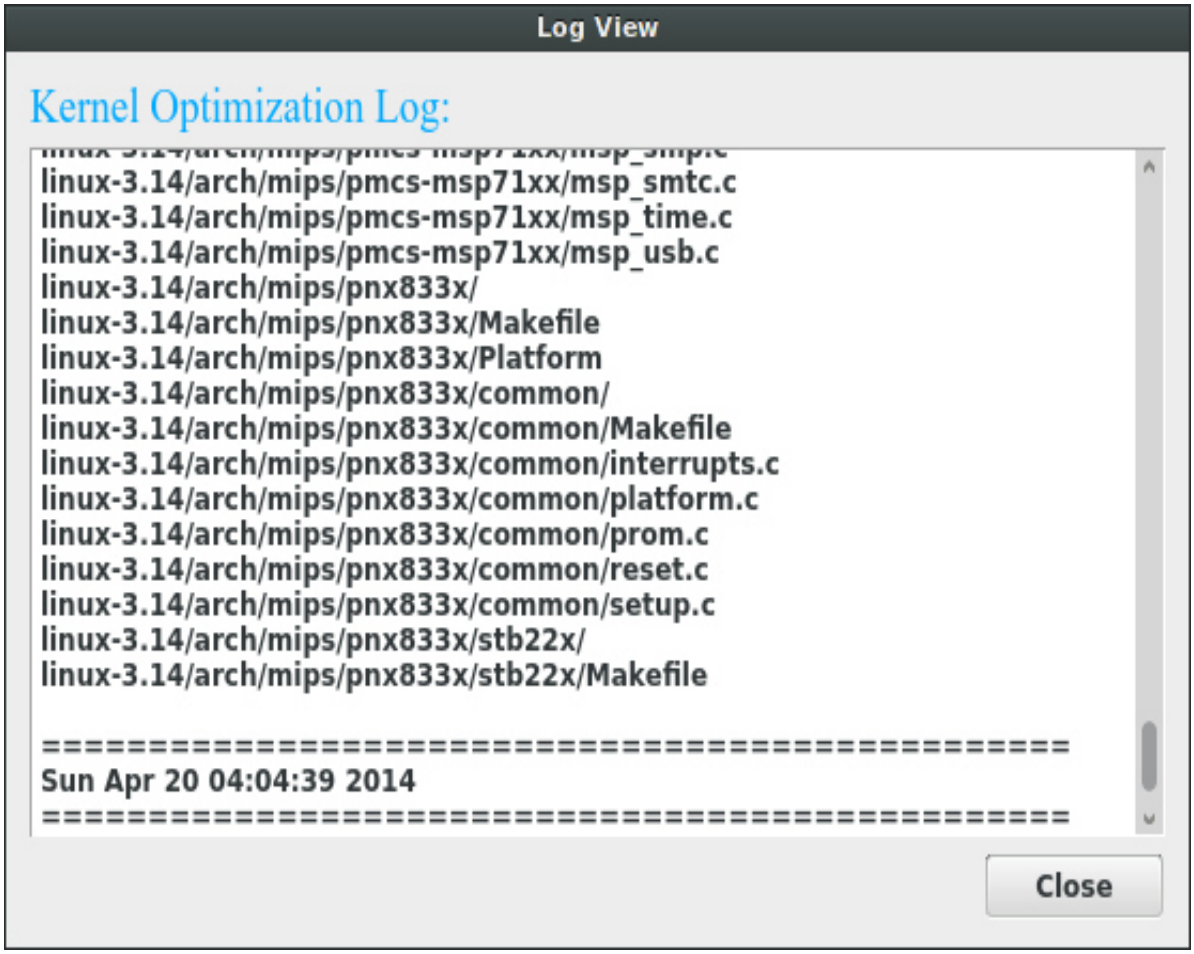


Figure 5.3 : Log View Window

### 5.5.4 Operating system info

Click in the OS info button, below window is come upt. It shows the Operating system information.

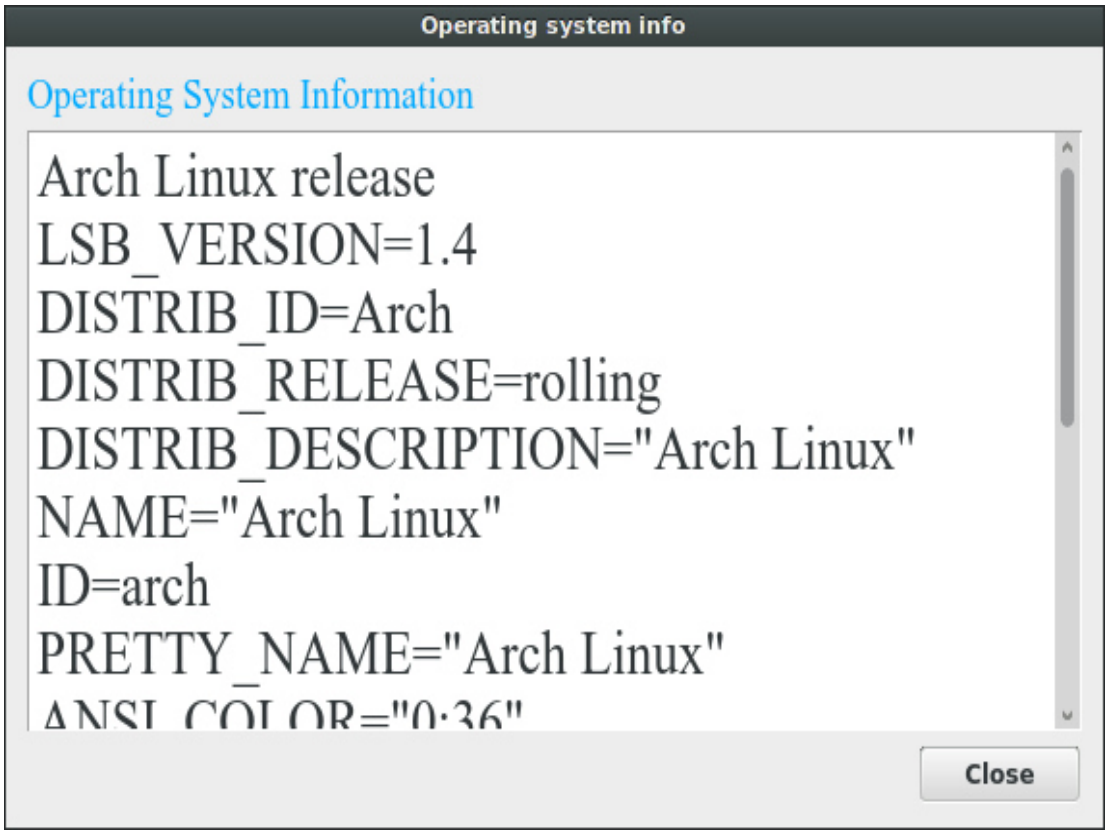


Figure 5.4 : Operating System Info Window

### 5.5.5 Hardware info

By pressing the Hardware info button user can see his/her system hardware information's. When use click that button system will ask administrative password to collect hardware info and show on below window.

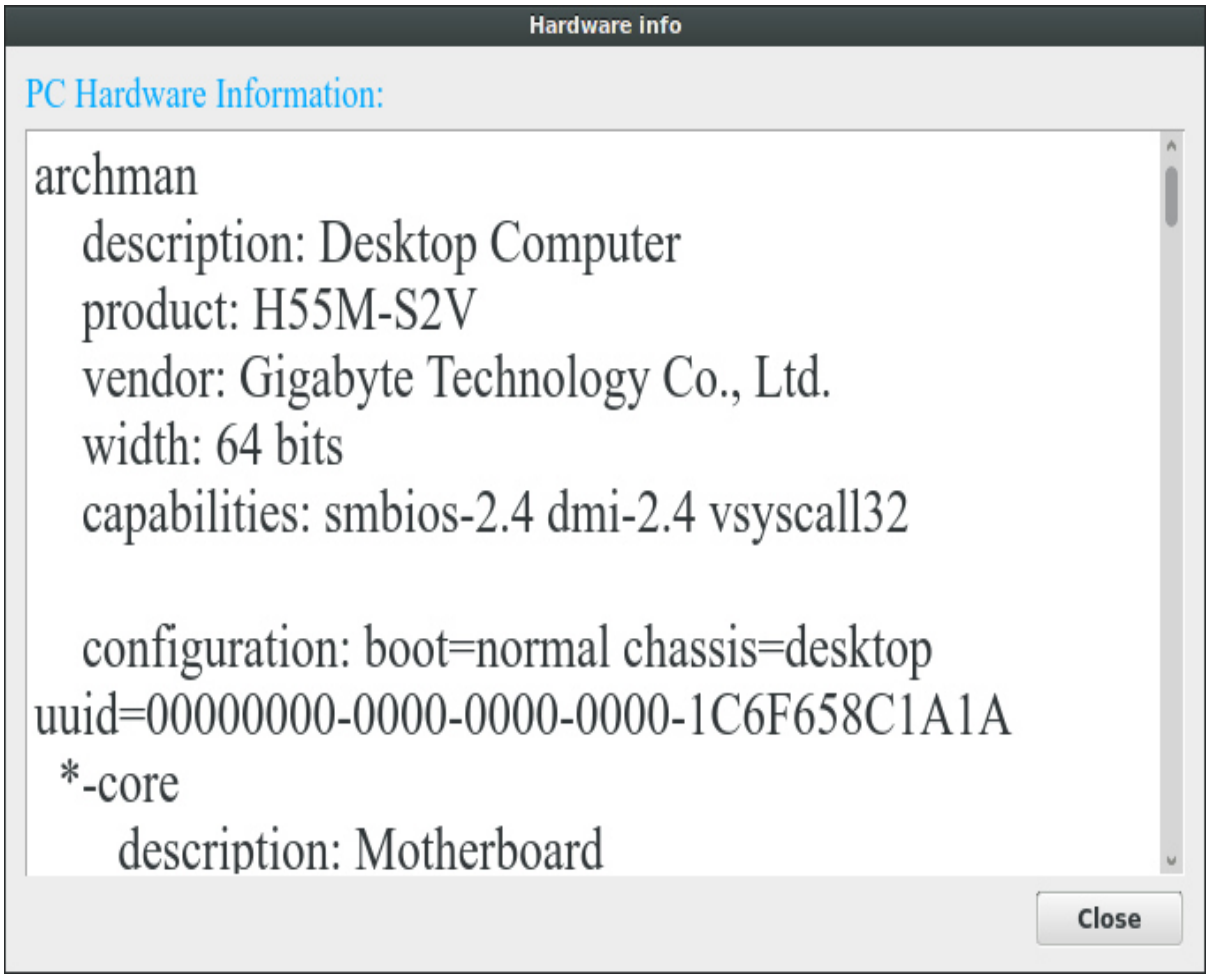


Figure 5.5 : PC Hardware Info Window

### 5.5.6 About window

After clicking the option about under the help menu option the below window will come up. In this window use can see the application information like version name, copy right, developers info, license info etc.

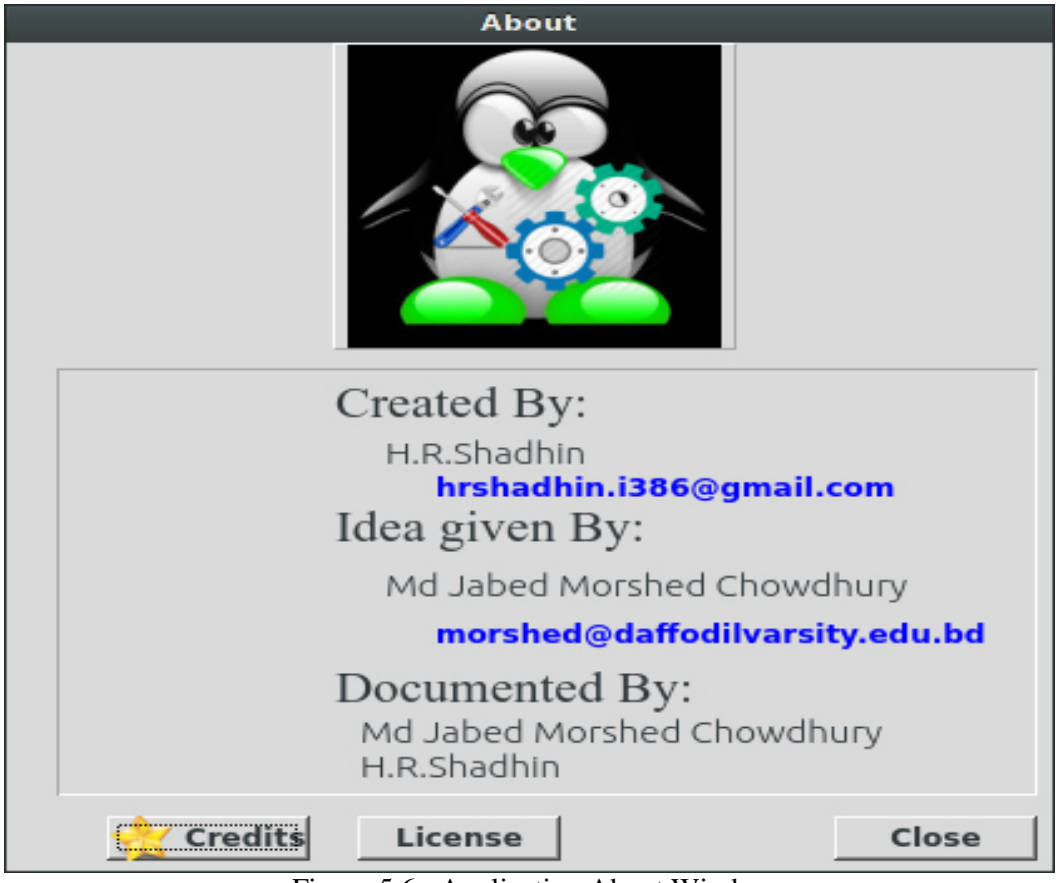


Figure 5.6 : Application About Window

# CHAPTER 6
# CONCLUSION

## 6.1 Introduction

In this time, we are completely depended on linux based system. A kernel is the heart of an operating system. It is a piece of software that securely controls access to hardware. The kernel manages memory, schedules execution of processes, allows for concurrent execution of processes, and provides an abstraction layer between the hardware and user space software. The ability to install a new kernel suitable to a system requirements is a valuable skill for administrators. Making a Custom kernel will help user machine to be a faster box rather than using the GENERIC kernel. This application is a strong of information where the users will get a number of benefits such as knowing linux kernel,freedom of open system,OS and hardware information,build their custom kernel. The users can use both new and old devices with theirs machines. During the last five months that I worked for build custom kernel, I tried my best to give proper support by make this application that user can easily build their own kernel for their machines and make it streamline,faster.

## 6.2 Critical points

It is true that development environment is fully changeable. Many things may go wrong even if we are working on real project. These are the problems for which most of the times are development plan need to be changed. I also faced some these sorts of problems in this project.

- Firstly, I did not get the sufficient information to combine Linux system commands with Qt libraries that handle the process.
- Secondly, I faced some problems when developing the project. It was hard to implement. I had to apply extra hard work to complete the project.
- Finally, implementation of the application in real environment requires different type of distributions and PC's. So, I had only able to test the application in limited environment.

### 6.3 Limitation

- User can not build kernel without internet connectivity.
- Only super user privileged members can build the kernel

### 6.4 Future Work

- Make support for other GNU/Linux distribution(RPM,Slackware).
- Manually kernel source adding option can be added.

# REFERENCES


[1] http://en.wikipedia.org/wiki/Methodology,Last accessed: 20th February,2014

[2]http://www.ramsoft.com.au/images/rad.jpg,Last accessed: 20th February,2014

[3]http://en.wikipedia.org/wiki/Linux_kernel,Last accessed: 10th March,2014

[4]http://www.enterprisenetworkingplanet.com/netos/article.php/3690371/Linux-Custom-Kernels-Trim-Fat-and-Tune-Performance.htm,Last accessed: 12th March,2014

[5]https://www.kernel.org/doc/index-old.html,Last accessed: 12th March,2014

[6]https://www.kernel.org/doc/index-old.html#Building_from_source,Last accessed: 20th March,2014

[7] http://en.wikipedia.org/wiki/Systems_analysis,Last accessed: 20th March,2014

[8] http://en.wikipedia.org/wiki/Waterfall_model,Last accessed: 22h March,2014

[9]http://en.wikipedia.org/wiki/Software_analysis_pattern,Last accessed: 22th March,2014

[10]http://en.wikipedia.org/wiki/Feasibility_study,Last accessed: 22th March,2014

[11] http://en.wikipedia.org/wiki/Systems_design,Last accessed: 26th March,2014

[12]http://en.wikipedia.org/wiki/Flowchart,Last accessed: 26th March,2014

[13] http://en.wikipedia.org/wiki/Use_case_diagram,Last accessed: 26th March,2014

[14] http://en.wikipedia.org/wiki/Software_development,Last accessed: 2th April,2014

[15]http://en.wikipedia.org/wiki/Qt_(software),Last accessed: 25th April,2014

[16]http://en.wikipedia.org/wiki/Bash_shell,Last accessed: 25th April,2014

[17] http://en.wikipedia.org/wiki/Blackbox_testing,Last accessed: 25th April,2014