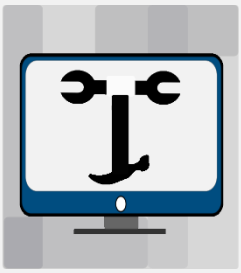


LLTT

Live Linux Technician Toolkit

UNIVERSITY OF PLYMOUTH

Philip Robinson
24/05/2019
ID: 10572891

Dissertation submitted in partial

fulfilment for

the degree of

Bachelor of Science in Applied Computing Technologies

University of Plymouth

2018 - 19

Abstract

Research suggests that the most common soft computer issues can be diagnosed and resolved using widely available and open source software. However, primary research has found that these solutions are often aimed at users with high technical abilities and focus on functionality over usability. The proposed artefact aims to provide these criteria through creating an open source collection of software that is designed to help diagnose, repair, recover and benchmark computer systems. This all-in-one solution will be based on current solutions but will provide usability and accessibility features including custom help systems and locally stored tool guides. To do this, the artefact will be developed using an agile methodology for flexibility and the project will overall be managed with a traditional approach. Through testing, the artefact has demonstrated that it can deliver a balanced solution that meets the requirements for users with a range of skills and variety of use cases. Ultimately, the project has economic, social and environmental implications as the proposed solution gives users more control over their technological issues.

Attestation

I understand the nature of plagiarism, and I am aware of the University's policy on this. I certify that the work submitted in this dissertation report is my own, and that appropriate credit has been given where reference has been made to the work of others. I understand that failure to attribute material which is obtained from another source may be considered as plagiarism.

Statement of Ethics

This project will comply with the Data Protection Act, which will ensure that all data provided from the client will only be used for the stated purpose, and will not be used otherwise. All data provided by the client will be anonymised, and in some cases modified to ensure anonymity of personal/sensitive information. Policies and procedures set by the client will be followed and no children or vulnerable groups will have any involvement in this project. Furthermore, the client is under no obligations to use, or implement, the solution once complete.

Signature: Date:

Acknowledgements

The successful completion of this dissertation would not have been possible without the guidance and discussion provided by Clint Washington and Mike Smith. They offered dynamic and propitious advice surrounding the design and development of the artefact and through polite encouragement ensured the project was finished *dreckly*. Additionally, I would like to thank the rest of my cohort for their useful contributions and willingness to act as copiously competent testers. Lastly, to my family for enabling me to take this course and move to Cornwall, and to my semi-professional proof-reader and *ansom*, Evelyn.

TABLE OF CONTENTS

Table of Figures

Table of Tables

Definition of Terms

Term	**Definition**
Benchmarking	Testing a system's performance potential
CLI	Command Line Interface
CPU	Central Processing Unit
DART	Diagnostics and Recovery Toolset
FOSS	Free Open Source Software
GUI	Graphical User Interface
Live CD | environment | OS	OS runs off host system memory rather than storage
LLTT	Live Linux Technician Toolkit, the proposed artefact
OS	Operating System
Tool	System maintenance software

1 INTRODUCTION

1.1 BACKGROUND OF THE STUDY

Since the adoption of home desktop computers, their availability and affordability have increased dramatically, resulting in 88% of people in the UK owning at least one home computer (Statista, 2019). This means that there are over 58 million people with computers that will likely experience hardware or software faults that will require resolving. This resolution features three main options: hire someone to repair it, costing money; repair it themselves, costing time and possibly money; or to throw the device away and replace it. All solutions result in consumer expenditure, which may leave them further out of pocket that necessary unless they are to fix the device by themselves. This links to the Right to Repair (R2R) trend, backed by organisations like Open Repair Alliance, pushing legislation to support a user's right to repair their electronics, improve user choice and for environmental benefits (OPA, 2019). The latter is a substantial issue; with over 2 million tonnes of electrical waste being discarded every year in the UK the environmental impact is significant (HSE, 2019). Therefore, any solution that could help to reduce this would present benefits to the environment as well as the consumer.

The open source software movement aims to make proprietary and paid software redundant by providing free software for every computer need, allowing users to freely choose the applications they use for home and business (Smith, 2007). In a similar fashion to the R2R trend, the open source movement increases the accessibility of resources with the intention of allowing consumers to attain better control over their electronics. Similarly, the significance of data protection is stronger than ever, as demonstrated by legislation like the EU General Data Protection Regulation (GDPR). By enabling consumers to perform repairs themselves, they have greater control of their devices and data.

Based on these trends and statistics, plus personal experience working as an IT technician, the study investigates the problem, evaluates current solutions, and presents a proposed artefact as a better solution.

1.2 STATEMENT OF THE PROBLEM

Many common computer hardware and software issues can be diagnosed and potentially resolved by an average home user with minimal IT skills by using the huge range of free and accessible software available online. However, some users may not be familiar with the process of sourcing and using appropriate tools and as such may have difficulty diagnosing common computer issues and identifying the best way to resolve them. For example, on the popular software source, www.download.cnet.com, there are 895 applications under the term '*antivirus*' that includes free, paid and variously rated software. Laypersons trying to find an antivirus would be inundated with choices and would not know which one to pick. Furthermore, the surplus of dubious publishers and reviews make it difficult to select a reputable one. This can cause issues before the person even attempts to use the software.

There are current and popular live CD solutions such as Knoppix, Ultimate Boot CD, Hiren's Boot CD, and AIO USB; however, these tend to be outdated, difficult to use, command line focused, lack local support and are designed for functionality rather than usability.

1.3 SIGNIFICANCE AND LIMITATIONS OF THE STUDY

The implications for users include enabling the user to: save money, have more control, be more time efficient, and produce less e-waste. Primary research found that the typical price paid by questionnaire respondents was £50 per fix; however, some companies charge up to £75 per hour (see chapter 6). As some operations such as recovering data from a hard drive can take multiple hours, the financial impact of IT repair can be substantial for individuals and more so for businesses (see Feasibility Study). As the aim of the artefact is to be accessible by most computer users and offer a more comprehensive array of diagnostic, repair and benchmarking tools. It is applicable to a wide scope of audiences and instances, such as from a layperson using it once a year to a fully qualified technician using it as part of their organisation's IT maintenance. The main target is for Windows OS users as it is used by 79.48% of desktops (Statcounter, 2019) so some of the tools are better suited to this OS. Despite this, the artefact is cross-platform and should function on any 64-bit capable machine.

The limitations of the project include access to appropriate and reliable tools depending on supported and compatible programs and their license, as only some *free* programs are freely distributable. Academically, there is little formal research about developing Linux technician tool kits, possibly due to the complexity of diagnosis-specific hardware or software issues. Fortunately, there are many online forums discussing and reviewing current solutions, however, the reputability of the sources is questionable.

2 RESEARCH

2.1 PROBLEM SCOPE

2.1.1 Why does a system fail?

Randell et *al.* (2001) state that a system fails either because it doesn't meet the specification or the specification did not suitably define its function.

e.g.

1. A highly demanding game will not run on an insufficiently powered system as the system doesn't meet the requirements.
2. A system gets infected with ransomware as the user's antivirus fails to prevent other malware infections.

A computer may fail after experiencing a fault, which results in an error in the system that may cause an alert such as motherboard error code beeps. E.g. a drive fails [fault] causing the operating system not to boot [error] results in system failure. The cause of these faults can be many, with some origins overlapping in nature (Hongxia et *al.*, 2008). Notably, it is identified by research that most failures are not independent occurrences but in fact recurrent; specifically, that a PC that experiences a hardware fault is 100 times more likely to fail later, 97% of those will occur within 10 days (Nightingale et *al.*, 2011).

2.1.2 Design Faults

The faults generally occur during the development process for either software or hardware, which tend to be caused by human error, affecting the internal system. Therefore, these faults are normally accidental and non-malicious; however, they can be caused by malicious intent such as *deliberate software obsolescence*. Similarly, *software aging* can result in performance degradation like memory bloating, and disk fragmentation. The former is a key point by the Right to Repair movement as obsolescence forces the consumer to replace the component. Examples of design faults include Windows update failure and file system corruption (Powell, 2001; Randell et *al.*, 2001).

2.1.3 Physical Faults

These faults can occur during hardware development or operation and can be caused by internal and external systems. The cause may be natural and accidental, such as physical deterioration, and may result in permanent or temporary system failure. Examples include hard drive failure or a CPU overheating due to insufficient thermal paste (Powell, 2001; Randell et *al.*, 2001).

2.1.4 Interaction Faults

These faults are caused during operation by human error that affects both software and hardware. The intent of the fault may be accidental or malicious, such as input mistakes or attacks and intrusion. Examples include malware, user forgetting passwords, or accidentally deleting a file. Furthermore, malicious faults can be malicious logics like malware, or intrusions which can be exploiting a physical or internal fault (Powell, 2001; Randell et *al.*, 2001).

2.1.5 Fault Management

Once the fault has been identified it is important to ensure it is managed to resolve the issue and mitigate reoccurrences. There are four main stages to fault management: Diagnosis**,** identifying and logging the cause(s) of an error; Isolation**,** physical or logical isolation of faulty components; Reconfiguration**,** switching out components (hardware or software); and Reinitialisation**,** record of configuration and checking success (Randell et *al.*, 2001; Kshirsagar & Patrikar, 2008).

E.g. Running a CPU stress test – Removing heatsink – Replacing thermal paste and reassemble – Check repair and log change

2.1.6 Real Life Statistics

The literature review contains some of the more academic sources of computer system failure, as there are few large data sources for statistical review which can be used to create an overview of why computers are likely to fail. Miller (2015) explains that there is a lack of data for current large-scale hardware and software failure data because of the cost and time requirements of such a study and company reluctance to share information. As demonstrated in this dissertation and by the purpose of the developed artefact, identifying and troubleshooting faults can be a complex and time-consuming task, so doing it on a large enough scale to produce valuable and reliable quantified data would be unfeasible. Unmanaged statistics from hardware vendors or manufacturers would offer little benefit to the IT market as their reliability would be questioned, especially considering manufacturing competition. Despite this, organisations like the Open Repair Alliance are pushing an open standard for creating a data repository for electronics repair to better share real-world findings.

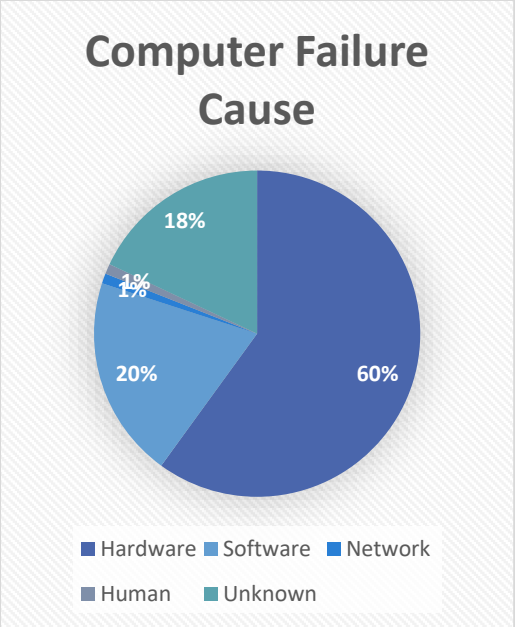


Figure 1. Component Failure Rate (Gibson & Schroeder, 2007b)

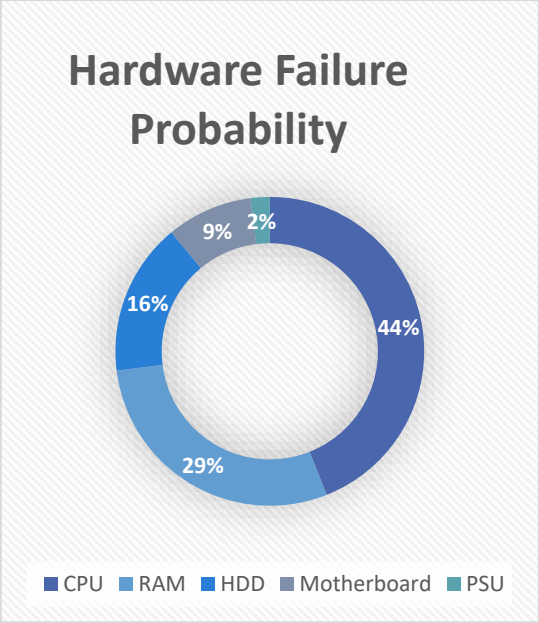


Figure 2. Hardware Failure probability (Gibson & Schroeder, 2007a)

Most notably, Gibson & Schroeder (2007b) found that the main cause for system failures were hardware related issues, followed by software and unknown causes at 60%, 20%, and 18% respectively (see Fig. 1). This is supported by Nightingale et *al.* (2011), who found that the main causes of hardware faults were the processor, memory and hard drive, with varying results including data loss, underperformance and system crashes. Both sources suggest that the processor was the most likely component to fail (see Fig. 2). Furthermore, the most common causes of CPU failure are due to user modifications to the processor, such as overclocking. Overclocked CPUs were between 4-20 times more likely to fail than CPUs running at the standard clock speeds and if underclocked they were between 39-80% less likely to fail. The failure in the chipset causes a machine-check exception meaning that the CPU has experienced an internal error: most likely a microcode bug or errors in the bus or cache. Memory failures were most likely to fail due to an incorrect value in the kernel code

page, caused by an error (often corruption) in the OS kernel or drivers. Finally, the OS often crashes due to read errors from a disk caused by problems with the drive, the disk controller and bus controller, or the data connection cable. Interestingly, the research found that white box (non-commercially built) systems were more likely to fail than branded systems, namely due to using non-main brand hardware (Nightingale et *al.*, 2011; Gibson & Schroeder, 2007b). Additionally, laptops were 25-60% less likely to fail than desktops, most likely due to durability features and specialised hardware counterparts like mobile CPU chipsets (Nightingale et *al.*, 2011).

Table 1. Component Failure Probability (Nightingale et *al.*, 2011)

Component	Failure Probability (30 days) [Occurrence: 1 | 2 | 3]
CPU	0.5% to 34% to 58%
Memory	0.06% to 8.3% to 50%
Hard Disk	0.4% to 28.6% to 58.8%

Applying the statistics from Table 1 demonstrates the significance of even low probability failure rates, as even with a 0.5% probability of CPU failure after 30 days of operation then at least 290,000 CPUs from the estimated 58 million household computers in the UK will experience a fault.

2.2 Preliminary Primary Research

Primary research was conducted prior to the development of the artefact to analyse the study area and assess the scope of the problem, by using questionnaires and practical work evaluating current solutions. The data gathered is both quantitative and qualitative.

2.2.1 Questionnaire

The questionnaire was comprised of 10 questions covering a range of areas with both quantitative and qualitative. Additionally, open and closed questions were used to allow respondents to answer freely while ensuring the data could be used for quantitative analysis. It was distributed to the general public to get an accurate view from people with a range of technical skills, which can be identified from question 1 (see Appendix: 1. Research). The aim of the questionnaire was to assess the common computer issues people face, the cost of getting them fixed, and their opinions on if they would use a tool such as LLTT.

An important stage of research with participants is to conform to the relevant data protection and ethical legislation and academic bodies. This was done using a consent form to inform the participant of the purpose of the research, how the data will be used, how their data will be protected and their rights to their data.

2.2.1.1 Results & Analysis

Key results that aided the development of the artefact include:

- 64.71% of respondents have never hired someone to fix their computer, suggesting that they are confident doing it themselves.
- 95.92% of respondents would be interested in using a toolkit like LLTT, despite 89% never having used software like this.
- 87.76% of respondents stated that PDF guides for the tools would make them more likely to use LLTT, supporting the inclusion of these in LLTT as planned.

Overall, the findings of the questionnaire suggest that most people would use LLTT and supports the inclusion of accessibility and usability additions like PDF guides and a possible support website. Since the majority of data is quantifiable it can be used to create statistics to aid comparative analysis and evaluation of the results.

2.2.2 Current Solution Analysis

An important stage of the initial research involved assessing the current options available to investigate the shortfalls, discover common trends and create a comparative evaluation with the finished artefact. The term for diagnostic and repair toolkits is commonly *repair disk* or *rescue CD* and one of the earliest is FreeBSD from 1993 (Hubbard, 1999). These rescue disks are often available as an ISO file which can be downloaded and then *burned* onto media, principally a CD disc or USB flash drive. Using a live environment is praised by many as improving functionality, usability, and convenience over traditional individual application installation on the host machine (Diesburg et *al.*, 2005).

The top options from technician online sources like *majorgeeks.com*, *slant.co* and review sites include Hiren's Boot CD, The Ultimate Boot CD, Trinity Rescue Kit, and Rescatux. To test the selected CDs including some of the above, the latest available versions were downloaded and run virtually to gain a measure of usability, functionality and applications included (see Appendix: 6. Testing). The usability and functionality were measured using common factors and an overall opinion, which is subjective (see Table 2).

Table 2. Summary of Existing Solution Evaluation

	AIO SRT	HB CD PE	UB CD	KNOPPIX
Functionality	4/5	4/5	4/5	2/5
Usability	5/5	4/5	2/5	3/5
Accessibility	2/5	3/5	1/5	4/5

2.3 LITERATURE REVIEW

2.3.1 Introduction

The concept of fault detection and diagnosis has always existed in every technical field. Whilst there is little academic literature on developing a live Linux environment for repairing computer systems, the study of fault detection and diagnosis methodologies has been well documented due to technical profession dependency on problem solving. Thus, this review shall analyse past and current thinking of traditional methodologies against modern approaches and how the two can be used together, within the IT sector and beyond. Secondly, there shall be an evaluation of computer hardware and software fault detection methodologies using statistics on the types of failure that are most likely to occur and the effect this can have on the industry. Finally, similar research shall be synthesised to find how an artefact can be developed as a proposed solution.

2.3.2 Diagnostic Methodologies

An early IT fault diagnostic tool based on knowledge-based methodologies was established by Bennet & Hollander in 1981. They developed an application using artificial intelligence practices for diagnosing faults in a computer system for IBM. The expert system was named DART and referred to as a consultant for identifying the system hardware and software components that were most likely responsible for a given fault, offering an explanation and solution. It worked through an at-the-time comprehensive system of fault trees that used 300 parameters and 190 production rules which took 8 months to develop. They estimated that these types of diagnostic systems could reduce 30-60% of user interaction with the computer, greatly speeding up the diagnosis process. However, they found that the way the engineers used DART greatly affected its success, with human error still occurring (Bennet & Hollander, 1981). This suggests that a system with guides on how to use tools effectively like in LLTT may improve the likelihood of successful diagnostics. Bennet & Holland were the founders of the diagnosis tool and DART systems that are still used today, which commends their methodology.

Alzghoul et *al.* (2014) introduced the subject of fault detection and diagnosis as completely different steps that are both of significant interest to the IT industry for meeting the fundamental requirement, Quality of Service (QoS). They describe the main methodologies of fault detection as being the traditional knowledge-based (KB), and more modern analytical and data-driven (DD) methodologies. KB methodologies include models like fault trees or flowcharts as applied in the DART project, whereas the DD methods use principal component analysis techniques. They found that to increase system availability a hybrid model using both methods was the most effective.

The findings of Gao et *al.* (2015) supports this, as in their study they provided an overview, review, and evaluation of fault diagnostic techniques, including the traditional KB and hybrid approaches. They explain that knowledge-based techniques depend on reliable and large quantities of historical data, hybrid approaches that use a combined hybrid and active approach for adding regular tests to check the fault diagnostic procedure. This makes hybrid approaches more effective than standalone KB methodologies while maintaining its efficacy.

Equally, Tian & Jia (2018) describe how the traditional NB methods feature a detection delay due to the non-linear characteristics of dynamic processes. This, in turn, can cost more in terms of money and time invested to resolve issues. They state that common methods like the multivariate statistical method and multivariate statistical process monitoring methods, which are based on using historical data as the factor for determining fault detection, lack an ability to discriminate faulty historical samples compared to modern dynamic models. They introduce a DD methodology alternative,

namely the Fisher discriminant analysis (FDA) for calculating the mean and variance of projection error which is critical in statistically predicting faults as part of a dynamic detection system. The findings by Tian & Jia align with the other studies by Gao et *al.* and Alzghoul, in both their definition of traditional and modern fault detection methodologies and the agreement that using a mixture of the two as a hybrid methodology is the most effective.

To relate problem solving to the modern IT industry, Sztandera (2010) designed a learning system for automating IT equipment repair tools based on analysis of historical data on computer equipment failure, a hybrid fault detection methodology. To highlight its importance, they suggest hardware is invariably going to fail at some unknown point, which creates a time sensitive demand for repairing the systems, especially when they can be critical to the operation of a business.

The consequences in the event of equipment failure or malfunction could cause significant losses to the company. – Sztandera (2010)

Therefore, they proposed a more economical and time saving method for storing replacement parts in an inventory which is optimised based on past data.

2.3.3 Computer Faults

Schroeder & Gibson produced many reports and meta-analyses on the rates and probability of both hardware and software failure in real-world settings, such as server farms. In a 2007 study, they analysed over 100,000 hard drives with a mean time to failure ranging 1 million hours to 1.5 million hours. The study highlighted a problem in how to define hard drive failure, which some determined a fail-stop model where the drive either works faultlessly or fails absolutely; however, this isn't appropriate to real-world failure. In a corporate setting, once a drive is identified as the probable cause in a problem, the operator performs tests to analyse the drive to find the root trigger. For both corporate and home users, data integrity is important, so most organisations adopt a 'better safe than sorry' attitude and replace the drive as soon as possible. This can lead to premature drive replacement which shortens the use and economic value of the component (Schroeder & Gibson, 2007a). This can be applied to almost all other computer components, which tend to not fail outright but rather fail over time and their performance degrades. More recently, Schroeder & Gibson (2009) discuss the Computer Failure Data Repository (CFDR), a 9-year project by Los Alamos National Laboratory, which is a development of a large-scale database storing information on computer failures from around the world. The aim of this is to provide an empirical data source for real-life system failures for analytics. It encompassed software failures, hardware failures, network failures, operator error failures, and environmentally caused errors; with the intention of making failure prediction an easier task to manage.

Li et *al.* (2007) state that for reliable service delivery the integrity of the system's hardware is "*vital".* Therefore, to produce an effective protection mechanism for system failure, firstly the failure mechanisms and error rate must be comprehended. In application to memory, it is one of the few components that is especially vulnerable to environmental factors including particle strike from radioactive decay and cosmic ray-induced neutrons. They suggest that the comprehension of hardware faults is imperative to develop an overall strategy of maintaining system dependability, regardless of their status as a service critical server, or a calculator. Additionally, while perpetual errors appear to require hardware replacement to avoid future disastrous failures, it may be that system-level mechanisms can assist with managing less significant isolated errors to diminish unrequired component disposal as well as the associated human administration. Ultimately, their findings concurred with past papers on typical memory and other core component failure and faults like the studies of Schroeder & Gibson.

Similarly, Sanker & Gurumurthi (2013) analysed data from system failures in data centres over a 12-month period to identify the main causes and consequences of 'soft' failures. They class these as failures that can be immediately resolved through replacing a component. They found that 43% of fixes were categorised as soft, or as being ambiguously caused. 27.3% of fixes were due to hard drive failure which was the single most common cause of data centre server downtime. As with Nightingale et *al.* (2011), they found that the more times a system experiences failure, the more likely it was to fail again. This is despite the fault being apparently fixed, which appear especially for soft failures. Specifically, their results showed only 28.18% of fixes actually resolved the issue in these instances and this figure degrades after each repeated failure. Secondly, they propose architectural approaches to counter soft failures which replace the traditional iterative methodology where the failing component is fixed in order to resolve the issue. These revised approaches were based on their findings understanding soft failures occur making them more effective at resolving them compared to traditional fix-what-you-see methods. Their approach to hardware modifications to reduce soft failures like incorrect seating and cabling of components includes removing unnecessary cables to allow for easier access to internal components. For software, they propose better using sub-system level detection capabilities like SMART in drives and criticise redundancy techniques for masking faults at service level visibility.

2.3.4 Live Linux Environments

Knoppix is described as one of the most comprehensive live Linux environments ever made which includes an array of software for office, productivity, graphic design, multiple desktop environments, and computer repair tools (Suzak et *al.,* 2005). In 2008, Hamada in conjunction with Knoppix, developed a live environment for mathematicians called KNOPPIX/Math, which is based on the popular Debian based live CD, KNOPPIX, by Klaus Knopper. KNOPPIX/Math offers many algebraic computation tools, visualisation systems, geometry software and a magnitude of low and high-level programming and development tools, which all come supported with an archive of documentation. Their project has been critically acclaimed and was featured by international universities and mathematical meetings. For future developments they discuss the benefits of using the system as a virtual machine; however, in terms of this dissertation, it would mean the software cannot interact with the hardware sufficiently for LLTT. The utilisation of live systems for scientific purposes demonstrates its efficacy and potential of Linux environments as they can be used for various and dynamic use cases.

Furthermore, Suzak et *al.* (2005) propose that one of the features that make Knoppix unique among live systems is their suggested Self-certifying File System (SFS) version that allows for Knoppix to be booted from the internet. They found that this version ran faster than using the CD-ROM version due to the greater bandwidth available, which reduced the boot time from 180 seconds to 80 seconds. However; they suggested that since it uses a client-server model for delivery, scalability is limited to the server side and this additionally restricts the system to being read-only. This form of accessing a system through internet protocols is interesting and could be applied to modern developments like in LLTT to improve accessibility and potentially performance.

Vetrivel & Pilla (2008) created a collection of bioinformatics tools titled 'Open discovery', which was built on the Linux distribution, Fedora. As with LLTT, they used a combination of software package customisation and creating binaries of the packages for better live environment functionality, including developing their own scripts. However; they used data persistence on the USB which allows data and settings to be saved, that they claim gives it better functionality than alternative solutions. Likewise, Yu et *al.* (2012) developed a live Linux environment for bioinformatics science which was made using FOSS software and was found to be more useful for a layperson. They

explained the benefits of live systems including it not affecting the system it is being used on as it required no installation, thus making it portable and therefore more useful to new users.

In conclusion, while the literature related to the development of LLTT is limited and could be argued historically irrelevant, it provides an insight for the fundamental characteristics of computer diagnostics methodologies, computer repair, and applications of live OSs. The findings from the multiple literature sources regarding fault diagnostic methodologies concur that adopting a hybrid approach is most effective for the entire process, from detection to resolution management. Parallels can be made to the development of LLTT and the tools included that apply to the diagnosis process to validate the integrity of a full live system and all the corresponding individual software components. Synthesising research findings for component failure rates, the different classifications of faults and using failure prediction is used to design a mechanism for corporate and home user level IT management. These studies indicate that a technical system like LLTT could be made a part of this management strategy, and by using the findings the most useful tools can be implemented to maximise the proposed artefact's practicality and suitability.

3 Project Management

3.1 Methodology

The project was managed using a traditional methodology with linear progression steps recorded with a Gantt chart. This method was chosen as it is the most established and researched methodology and offers advantages including simplicity, predictability, and robustness, which makes it suitable for easy planning and efficiency (Špundak, 2014). Project management is defined by the Project Management Institute as "*the application of knowledge, skills, tools, and techniques to project activities to meet the project requirements*" (PMI, 2019), which is demonstrated by the range of resources used throughout the project, including primary and secondary research, meetings with the project supervisor and reputable management methods like Gantt charts (see Appendix: 2 Project Management).

An agile approach, namely an adapted Test-Driven Development (TDD), was used for the software development of the artefact as testing was performed during each stage (see Chapter 6.2 for more information). Agile approaches are well reputed as being the best methodology for software development due to the flexibility and scalability benefits, which is demonstrated by it being used for over 70% of software developments (Kruchten, 2013). One of the key benefits of using this methodology is that it allows for teamwork and work sharing which will aid the future development of the artefact beyond the scope of this project.

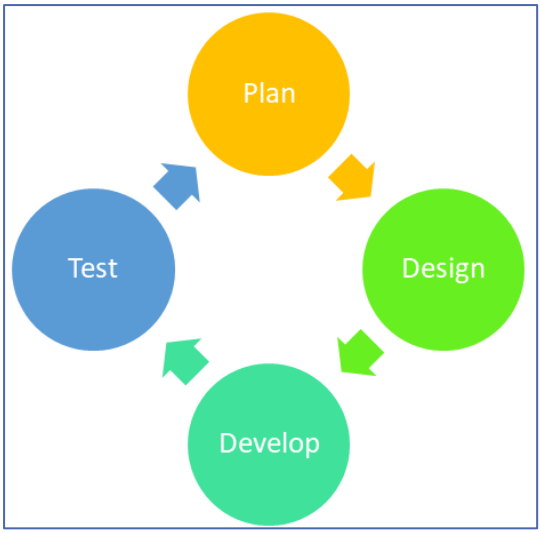


Figure 3. The Four Stages of Development

The methodology used was based around the four main stages: Planning, Designing, Developing and Testing; however, within each of these stages TDD was integrated. The main management methods used were scrum backlog and sprint sheets to manage the steps and tasks required in the sprints of development. Lab reports were used to document specific processes such as imaging the system using Pinguybuilder (see Appendix), and a development log recorded the weekly tasks.

3.2 Project Proposal, Feasibility Study, Risk Register

As part of the project proposal and management, a feasibility study was conducted to assess the probable benefits and costs of the study, plus the technical, economic, legal, operational and scheduling feasibility of the project. The aim of the study was to analyse the domain of the study, namely the various implications and constraints that could affect the success of the project. For instance, as the toolkit relies on third-party software it is important to ensure the public license allows redistribution, for legal and ethical purposes.

Another part of the documentation is the risk register that acknowledges, details, mitigates and allocates responsibility to each of the potential risks to the project. Williams (1994) describes risk assessments as being integral to the management of a project and that risk management should be integrated to fundamentally ensure the security of the project (see Appendix: 2. Project Management).

4 System Design

4.1 Accessibility

Accessibility is the degree of how users with different needs or people with various disabilities can access information or have control of a system and services. This includes physical and perceptive conditions such as visual impairment, as well as cultural and social abilities such as language issues. It is a concept that coincides with the ethos behind the free software movement that focusses around freedom of use and the availability of software. A report by the World Bank Group (WBG) stated that over one billion people around the world have a disability and published a commitment that all WBG digital development projects will be "*disability sensitive*" by using accessibility standards (WBG, 2018). This is a common initiative that is being adopted by major software developers like Google, Microsoft, and Facebook who produce mainly proprietary software. However, application accessibility design is suggested to not be a priority for free software projects due to developers working independently and without the experience of accessibility issues (Alves & Cagnin, 2014). A GNU report estimated that 85% of software including applications and websites fail their accessibility standards which highlight the spread of the issue in the software development industry, especially for GNU and open source movements. They suggest proprietary file formats and developers having a lack of diligence to be able to consider the impact of their development choices on accessibility in their software (GNU, 2018). This is a recurrent explanation, but despite this, there are many proposed solutions.

Goncalves de Branco et *al.* (2014) admit that "*building accessible products is not an easy task"* but recommend industry standards by ISO and W3C. There are many frameworks for accessible design produced by global organisations for standards like World Wide Web Consortium (W3C) and Web Accessibility Initiative (WAI), International Organisation for Standardisation (ISO), and British Educational Communications and Technology Agency (BECTA). Therefore, relevant accessibility frameworks by these bodies were consulted for the design and development of the artefact to ensure its accessibility, a problem identified in the analysis of alternative solutions in DART software and thus became a key aim of the project.

Table 3. Accessibility Framework Overview

Organisation	Framework	Description
BECTA	Guidelines based on ISO and other frameworks.	A recommendation of the main characters based on the standards by ISO and W3C, which is a brief introduction to the most important features in application accessibility (BECTA, 2019).
ISO	ISO 9241-171:2008	A paid for ergonomics standard for user interaction on systems focused on usability as well as accessibility. It is well documented in academia and is applicable to a range of software (ISO, 2019).
W3C	WCAG 2.1 (ISO/IEC 40500:2012)	A free to access framework based on 5 main principles at 3 levels of adherence (A-AAA) to application accessibility including: Perceivable, regarding content like text, audio and colour; Operable, using the interface with details for navigation;

		Understandable, for language use and readability of content; Robust, which is for code compatibility; and Conformance, to review accessibility support (W3C, 2018).

The guideline by BECTA acts as an introduction of design accessibility with it explaining the main features and functions that should be considered during application development. They also reference and give hyperlinks to professional frameworks, including those by W3C and ISO. Therefore, it has limited information but is good to gain an overview of typical accessibility guidelines in development. The ISO 9241-171 framework is the most institutional and almost considered legally required as ISO develop the majority of global standards across industries. However, the main negative is that it costs money to access the documents (CHF 198 / ~ £152), but a consensus can be gained through the many academic reviews and sources that document the use and implementation of the standard through application development. WCAG 2.0 is a subsidiary framework to W3C's main guidance on web accessibility. Nevertheless, it is a fully comprehensive, 150-thousand-word document covering everything accessibility related to software development and document creation. Therefore, it was the main source for specific accessibility criteria in the design stage of the artefact development, using a mixture A to AAA W3C level requirements.

A check sheet was implemented during the design stage for easy consideration on how to best implement the main accessibility features from the reviewed frameworks which included icon sizes, window characteristics, UI controls and much more.

Table 4. Key Accessibility Features Checklist

Feature	Function
Icons & Fonts	☑ Large icons (mouse pointers [48px], desktop icons) ☑ Options to change font/icon sizes that are easy to find and use ☑ Icons should have tooltips/labels
Naming System	☑ Names and titles should be short, clear and meaningful (assists screen-readers & narration)
Windows	☑ Application windows should be easy to identify and manipulate (reposition and resize) ☑ Window design scheme should be clear
Navigation	☑ Keyboard controls for menu and tools
Documents	☑ Documentation should be just as accessible as product itself ☑ Document accessibility options within the document reader
Text	☑ Simple English ☑ Can be copy and pasted ☑ Multiple language support ☑ Diagrams are clear and visible ☑ Warning and error messages are clearly marked
Display	☑ No flashing screens (photosensitivity issue at 16–25Hz)
Colours	☑ Good contrast ☑ Option to change colours ☑ Warning and error messages are clearly marked

4.2 USABILITY

Another measure of a system is its usability, an assessment of how efficient the target user can utilise the resource to achieve the desired outcome. As with software accessibility, there are frameworks for developing applications that are easy and intuitive to learn, such as ISO 9241, that specifies the concept, considerations, and outcomes (ISO, 2018). Research of computer usability began during the emersion of human-operated electronic systems during World War II; for instance, due to research by engineering psychologists, the number of controls was reduced in fighter planes to improve pilot performance (Moreno et *al.*, 2013). Similarly, the study of human behaviour, namely Human Computer Interaction (HCI), is used by academic and practical fields including computer science to advance a system's usability. It is suggested that IT systems are made up of interactors that a user operates, and these cognitive processes can be modelled to better predict and plan user behaviour to optimise the usability of the system (May, 2001). The general consensus is that applying the study of HCI in developing an application improves the usability and accessibility design to benefit the performance of the system and the user's interaction (Joshi, et *al.*, 2010; Lazar et *al.*, 2017).

The Usability Goals Achievement Metric (UGAM) is a method of measuring the degree the design of an artefact accomplishes, its user experience aims to quantify the system efficiency and user satisfaction (Joshi et *al.*, 2010; Mifsud, 2015). Some of these metrics were used in the testing phase of development as can be seen in the corresponding chapter: these include calculating the completion rate, time-based efficiency and satisfaction levels. The significance of an application's usability is fundamental to its functionality as a toolkit and it was a priority to ensure the system's compatibility to current accessibility and usability standards.

4.3 OPEN SOURCE

4.3.1 FOSS

A key focus was to develop an artefact using free open source software (FOSS) that would also be freely licensed to allow for others to use, modify and share the project as part of its extended development. LLTT is based on the Free Open Source Software (FOSS) principle as the purpose of the artefact is to be a collection of third-party tools and thus legally it needs to use freely alterable and distributable software. Additionally, there are many positives to using open source rather than proprietary software that are referenced in academia. This is outlined by Ebert (2008) who states that "*today's most innovative products and solutions"* are open source and include a variety of software developments, from OSs like Linux to application servers like Apache. The open source movement is praised by many for improving software development and boosting competition in what was a market dominated by big software developers. So much so that the trend is proposed to be changing the market from supplier-driven to focussing on the needs of the user, which can be seen in the variety of applications available in modern times (Robles, 2006; Kratov, 2011; Yingkui et *al.*, 2010).

Despite this, FOSS is not entirely free to the developer as even if there is no financial cost the life cycle of the software development it will undoubtedly cost hundreds of hours of work, as demonstrated by this project. Robles (2006) expresses that while the FOSS development model can be less efficient than traditional approaches, it allows for greater flexibility as external contributors can assist in the development process. These external contributors are often users of the application, so can offer their experience to aid the development and better integrate testing. In fact, Wei et *al.* (2017) found that while independently developed FOSS projects may take longer to

be released, they are often better maintained and offer greater user support, typically through community-based forums. While the artefact in its current state has not been publicly released, a website for downloading the source files, developers blog, and a support forum has been proposed as part of future project development.

The GNU Project is the main source of public licensing in FOSS, namely the General Public License (GPL) (currently in version 3), but they offer several other licenses for different cases. These licenses aim to avoid restrictions from copyright laws and are evaluated by the project to review the software to ensure it conforms to the chosen license. The GNU Project is funded by the Free Software Foundation (FSF), a non-profit organisation with a focus on making proprietary software redundant by providing free software for every computer need (Smith, 2007). As demonstrated in the list of software used, the majority of tools use the GPL license system, demonstrating its significance in current software development (see Table 6).

4.3.2 Lubuntu & LiveCDs

The foundation of the artefact is the open source operating system called Lubuntu, which is a lighter version of the popular Debian based OS, Ubuntu. Lubuntu is referred to as being "lighter, less resource hungry and more energy-efficient" and runs off LXDE desktop environment rather than Ubuntu's Unity desktop, so it is better optimised to lower performance systems (Paul, 2018). As they are based on the same family, it supports a huge range of applications and is frequently updated. As can be seen in Table 5, Lubuntu requires nearly half of the resources as Ubuntu which makes it suitable as a liveCD.

Table 5. Ubuntu vs Lubuntu Minimum System Requirements

	Ubuntu 18.04	**Lubuntu 18.04**
CPU	2GHz dual core	2GHz single core
RAM	2GB	512 MB to 1GB
ISO size	1.9GB *(Cooke, 2018)*	1GB *(Paul, 2018)*

As the system is being run in a live environment mode, all data is transferred from the media (USB or CD) to the system memory (using *initial ramdisk)*, which is often a limiting factor in a system. The 64-bit version was chosen over the 32-bit as LLTT is targeted at the majority of system users which can operate 64-bit architectures and therefore compatibility issues should not occur. Additionally, it ensures the support for the current and acclaimed tools used in LLTT.

The liveCD operates without data persistence, so no data can be written to the media, however, changes can be made during a session for downloading antivirus database files or extra drivers. Data persistence is included in some liveCDs to allow for customisation and changes to the system, such as for installing additional software (see 'Open discovery' in the Literature Review). While this could extend the CD's functionality, to minimise media size and simplify the operation of the toolkit (Vetrivel & Pilla, 2008).

4.3.3 Software used

Table 6. List of Applications

Name	Description	Developer	License
Boabab	GNOME graphical disk usage analyser	Marzocca (GNOME team)	GPL
Boot Repair	Repairs frequent boot issues with Windows or Linux OSs	Yannubuntu https://sourceforge.net/p/boot-repair-cd/	GPL
ClamTK	GUI for ClamAV antivirus	Dave M (Github: @dave-theunsub)	GPL
Clonezilla	A disk and partition imaging and cloning tool	NCHC Free Software Labs	GPL
Firefox	Popular free and open source web browser	The Mozilla Foundation	MPL 2.0
Forensics Tools Meta-package	An all-inclusive Debian package of forensics tools	Debian Security Tools Team	BSD-3-Clause
GEdit	GNOME text editor	GNOME	GPL
GNOME Disks	Comprehensive storage management tool	David Zeuthen (GNOME team)	LGPL
Gnome MultiWriter	USB writing tool	GNOME	GPL
GParted	GNOME partition editing tool	GNOME	GPL
HARDiNFO	Default system profiler and benchmark tool	Ultimate Systems	GPL
Kate Text Editor	Advanced text editor	KDE	GPL
Lubuntu 18.04	Lightweight OS based on Ubuntu	Lubuntu Community	GPL
lxtask	Default task manager	LXDE	GPL
LXTerminal	Default terminal	LXDE	GPL
PCMANFM	Default file manager in Lubuntu	Hong Jen Yee	GPL
Phoronix Test Suite	Free benchmarking software with over 220 tests	Phoronix Media	GPL
PhotoRec	CLI data recovery tool	Christophe Grenier	GPL
Psensor	Graphical temperature monitor based on lm-sensor	Jean-Philippe Orsini	GPL
Stegosuite	Steganography tool	Tobias https://dev.stegosuite.org	GPL
TestDisk	CLI based disk tool for recovering partitions	Christophe Grenier https://www.cgsecurity.org/wiki/TestDisk	GPL
Wireshark	Network packet analyser for network management	The Wireshark team	GPL
Zenmap	GUI for Nmap network scanner	Gordon Lyon	GPL

5 SYSTEM DEVELOPMENT

5.1 INTRODUCTION

5.1.1 Purpose

The purpose of a diagnostic tool is to identify faults in the hardware and software of a computer system. Firstly, the test is chosen and then the test is executed, followed by interpretation of the results. For example, the user may think they have a faulty hard drive, so they choose a relevant test, run the test and analyse the results to decide on what to do next. Difficulties occur when there are multiple or complex faults, for example, a memory test can diagnose faulty memory, but it could be the RAM stick or DIMM slot that is faulty. Therefore, the diagnosis process is often more complex than initially imagined and may involve multiple stages with the use of multiple tests. Hence, the artefact includes some similar applications that overlap in function, such as two rootkit scanners, *chkrootkit,* and *rkhunter*.

5.1.2 Interpretation of results

As stated above, the results from tests may not provide a clear method to resolve the fault. Shubin & Ulrich (1982) highlight the importance of intelligible results to prevent the unnecessary rerun of tests. Therefore, as well as tutorial guides being included in the toolkit, they provide guidance on how to interpret results and solutions for repair. This is not currently a standard for diagnostic tools like the existing solutions which demonstrate the need for locally stored resource guides.

5.1.3 Choosing the tool

As a key feature of the solution is making these tools more accessible to less technically skilled users, help systems will be provided to help users chose which tests may be appropriate based on their system faults/symptoms. For example, if the user is noticing their system is running slower than normal, they can look this symptom up in the guide and review the suggested diagnostic tests. Additionally, guides will provide recommended solutions outside of the toolkit and suggest useful webpages, articles or videos that may help them resolve their issues. Haugsted (2008) discusses the benefit of script automation in simplifying and speeding up the diagnosis process as background scripts can run common software. This is the purpose of the ‘Help System’, which is essentially a computerised flowchart to help users decide what software to run.

5.2 LLTT Development

5.2.1 Method

The first stage of the development process was to determine which method for creating the LiveCD would be used. There are a few choices depending on the level of configuration required and the complexity of configuring the system (see Table 7). In the end, imaging a configured installation was used as it allowed for the greatest level of system customisability while still being a fairly simple process and it could be performed through the GUI.

Table 7. LiveCD Development Methods

Title	Description	Examples	Pros	Cons
From Scratch	Building the system from the ground up using source code.	• Linux from Scratch • Ubuntu Customisation Kit	• High customisability • No bloatware	• Complex configuration
Creation Tool	Using an automated creator to add modules to a system	• CUBIC • Slax	• Easy configuration • Quick to finish	• Can't customise desktop environment
Current Installation Imaging	Imaging a system that has been configured	• PinguyBuilder • Linux Live Kit	• High customisability • Simple	• Some bloatware

5.2.2 The Foundation

The artefact is built on OS Lubuntu 18.04 (named Bionic Beaver) with LXDE desktop environment which was selected due to its low resource usage making it suitable to be used as a live environment. Bionic Beaver uses the Linux 4.15 x86_64 kernel, which was the latest stable version at the time of initial development in 2018. Once installed, the first stage was to clear the system of redundant packages which were removed through the Software Store and Synaptic Package Manager which are both included in Lubuntu, before installing the desired packages through the same methods (see ID 1 in Fig. 4). (*Please see Appendix: 4. System Development for more*).

ID	Purpose/Story	Difficulty	Priority	Success
1	Remove uncessary files	1	3	Yes
2	Install desired packages	2	5	Yes
3	Test packages	4	4	Yes
4	Configure Conky	3	3	Yes
5	Remove password authentication	3	2	Yes
6	Develop custom scripts	5	3	Yes
7	Design desktop and GUI	3	4	Yes
8	Implement accessibility requirements	3	4	Yes
9	Test current state	4	4	Yes
10	Convert to liveCD ISO	2	3	Yes
11	Test ISO	5	3	Yes

Figure 4. Scrum Product Backlog

5.2.3 Applications

Some of the packages included in Lubuntu were kept due to use in the toolkit (see the full list in Design). Other packages were preferably sourced using the Software Store to ensure compatibility, or through Synaptic Package Manager (see Fig. 5). The applications were tested once installed to verify correct function and, in some cases, older versions of the packages were sourced to ensure compatibility with the OS. All of the tools can be accessed through the menu and some can also be accessed through custom scripts, like the Symptom Checker tool.

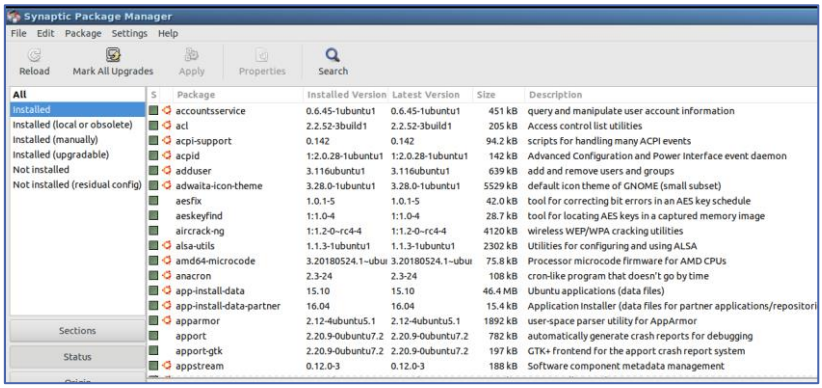


Figure 5. Synaptic Package Manager

5.2.4 Scripts

The Lubuntu environment functions with bash shell scripts which allows simple configuration by modifying inbuilt scripts or creating new ones. The first script developed was “Philip’s Conky” script, that is a lightweight system monitor used to display in the text the hardware of the system and key specifications, including temperatures and resource usage (Ruan et *al.*, 2009). The conky package is highly configurable and gives more purpose to the desktop space. Additionally, scripts were used to create a help system and symptom checker that were graphical methods for the user to select the best suited tool to improve the usability of the product. This was performed using Zenity, a program that executes GTK+ dialog boxes in shell scripts, which was integrated into the scripts to change them from a CLI to a GUI. Finally, some scripts were used to simplify the use of included command line-based tools including a stress script that uses the stress-ng package to force the CPU to 100% utilisation (see Appendix: 4. System Development).

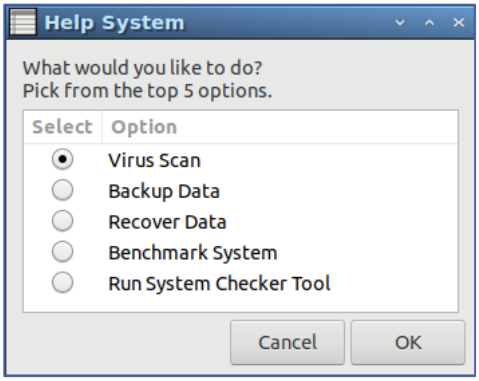


Figure 6. Help System with Zenity

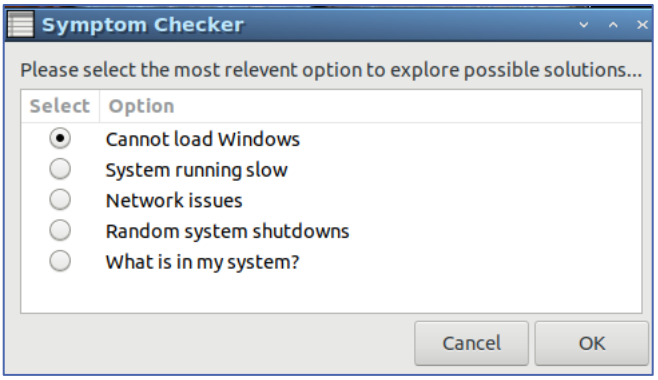


Figure 7. Symptom Checker with Zenity

5.2.5 Environment Customisation & Accessibility

The most common method of making applications accessible is through the menu system which is why it was created with accessibility features including tooltips and graphics. There are three menu sections separating the applications by 'basic' or 'advanced' category based on the tool's ease of use, the likelihood of use, and potential damage if operated incorrectly. It can be accessed using mouse or keyboard navigation and is organised in alphabetical order to simplify access. The preferences section allows the user to customise their experience, install any additional drivers they may need for graphics card support, or change accessibility settings. The menu was created using the program MenuLibre, where the categories and launchers could be configured. In version 1, the applications could be accessed through shortcuts on the desktop such as in other solutions like AIO SRT; however, it quickly became cluttered and made organising the applications.

Figure 8. Menu System

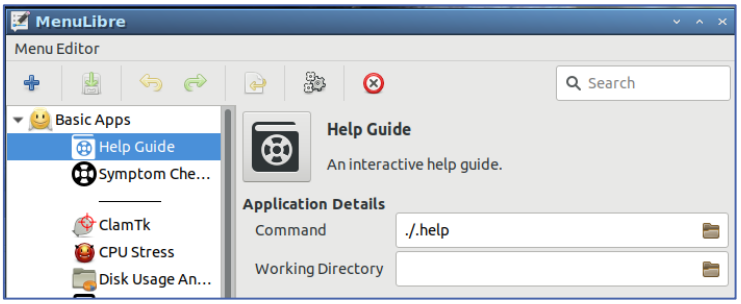


Figure 9. MenuLibre

A common system feature is the requirement of a user password before running certain privileged commands and applications such as Zenmap. Thus, the *sudoers* system file was edited to remove this requirement.

The main criterion for the design was to be accessible while maintaining functionality. This led the design to be Microsoft Windows-esque, as the majority of users would be more familiar with using a WOS. For example, the Mikachu window theme, ThinIce widget, GNOME Icon theme, and terminal theme were chosen for resembling past Windows OS design styles. As part of the accessibility requirements, the logo and wallpapers were designed to be clear and visually appealing. In total, 6 wallpaper images were included to allow the user to customise their experience or alternatively, they can change the desktop to a solid colour to aid their user experience. The logo and icons were developed with Inkscape while the images were edited with FastStone Image Viewer.

5.2.6 ISO generation & Live CD Distribution

The final stage is to image the operating system for the liveCD, which was performed using the free and open source (DBAD public license) application PinguyBuilder, a fork of Remastersys tool (SK, 2016). The user can customise the boot menu image, included directory and liveCD name. It generates an ISO file using packages like Ubiquity (the default Ubuntu installer) and Casper file system, which took 1 hour 32 minutes to complete and resulted in a 1.99GB ISO file. The output can be viewed live to follow the progress of the operation in the Output tab and once finished it produces a hash value for file validation (see Appendix: 4. System Development).

5.3 Development Issues

Unfortunately, due to the complexity of the system and magnitude of applications included, during testing multiple bugs were found affecting scripts, the tools, and the overall system. A particular issue occurred with Zenity due to it being outdated, as the exit code output from a user closing the dialogue window is the same as when the OK button is used. Consequently, it was problematic to program logic that could progress as desired or be closed. Another issue that frequently occurred was configuration incompatibilities, such as with the SleuthKit which was missing dependencies that were not included in the Debian Forensics meta-package. Additionally, Wireshark network packet capture is disabled as it is run as super-user, making it redundant in the current state. Therefore, this would be easily rectified in a bugfix update.

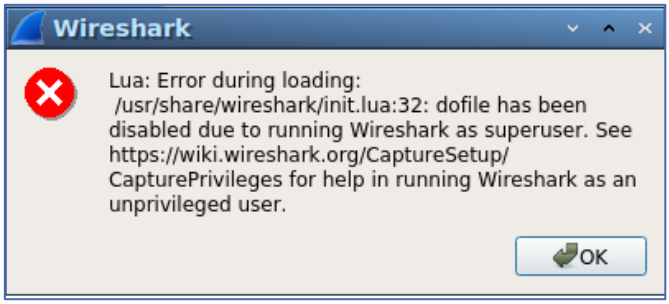


Figure 10. Wireshark Lua Error

While Conky is a useful and lightweight system monitor that is easy to configure, the script for accessing network card and graphics card data could not be resolved as the host system' s interface location could be different. The network command could be fixed by adding additional commands to support multiple network cards in a bugfix update. The latter issue was due to difficulty supporting both Nvidia and AMD dedicated graphics cards, as preloading drivers for all modern graphics cards would require lots of additional storage.

PinguyBuilder offers multiple offers for imaging the system, including making a distributable copy, however, this did not save the user account data required to run the artefact properly. This issue may be due to the user account being root where it should have been a guest account. Consequently, the Backup option was used to image the full system (see Appendix: 4 System Development).

As prior discussed, Version 2 of the artefact was initially developed with the latest version of Lubuntu (18.10), however, there were stability issues with some of the desired applications and problems with VMware (see Development Logs in Appendix). Instead, version 1 on Lubuntu 18.04 was reworked with few consequences; thus, the issue only resulted in work and time loss which was recorded in the risk register (see Appendix: 2. Project Management).

6 TESTING

6.1 INTRODUCTION

Testing methodologies are strategies used to test a product to verify it is fully functional, meets its requirements, and functions within its design parameters without abnormalities. It is important to plan software testing within the design phase as modern software development can be highly complex with applications and systems incorporating many functions and operate on multiple platforms. Testing software as part of the development stage incorporates various parts, from testing individual modules to the entire system, from debugging to accessibility checking, and from measuring performance to validating security. Additionally, it demonstrates the quality of the product to its shareholders, such as the development team, publishers and investors.

6.2 METHODOLOGY

To test an artefact as complex as an operating system requires separating the system's logical parts based on their functions: such as scripts, packages, and settings components. Therefore, an approach based on frequent testing of each component within iterations was required. Test Driven Development (TDD) is an agile methodology that focuses on testing; each development stage of the software is coordinated with a correlated testing stage within each cycle, which can be seen in the sprint backlogs. Romano et *al.* (2017) describe TDD as an iterative development approach that defines unit tests in the design phase to implement a stricter code development that aims to reduce the overall testing time. It is generally considered to improve code quality and improve testing time across iterations (Besson et *al.*, 2015; Maier et *al.*, 2018). A study at Microsoft found that despite an increase of 35% in the initial development time by using TDD, this reduced to a 15% faster development in the second iteration and 68% less defect complexity (Bhat & Nagappan, 2006). However, a meta-analysis by Karac & Turhan (2018) suggested that the effects of TDD were mainly from anecdotal evidence rather than empirical sources; and found that TDD can overcomplicate the development process which goes against the aim of rapid cyclical development in agile methodologies. Suitably relevant to this dissertation, Maier et al (2018) used TDD for fault injection in firmware to test its ability to detect and mitigate a variety of faults. They found that using the agile approach is highly appropriate for this situation as it requires quick response times for feedback in test development which is also needed in firmware fault detection. Therefore, the DTT methodology assists in achieving a key specification in diagnostic style systems, namely fast mitigation, which makes it an apt strategy for this dissertation.

6.3 TESTING

Primary white box testing was used during the alpha development stages within the product development cycle, and black box testing was used pre-deployment beta tests. The main beta testers were chosen with at least 2 years of experience in programming and comprehensive knowledge of computer system operation to thoroughly test the functionality of the artefact; while laypeople were additionally used to measure the artefact for usability and accessibility. This was done because the artefact is aimed to be accessible to both users with low and high technical skills.

6.3.1 Public Artefact Testing

Participants were invited to test the artefact that was being run on an old laptop with a failing hard drive, to allow testers to use the system in a more real-world situation. Afterwards they were asked

to complete a short survey to quantify their experience and gain their opinions on the artefact's design, functionality, and usability (see Appendix: 6. Testing).

6.3.1.1 Results & Analysis

The overall feedback was positive and included comments that are useful for furthering the artefact development, with bug reports and tool suggestions.

- The artefact was rated highly, with 100% scoring 4/5 or 5/5 for usability, functionality, and design.
- Most people (55%) would rate 5/5 for the likelihood of reusing the artefact, with comments indicating other opinions were due to hoping they would not need to use the toolkit.
- 100% of people would be highly likely to recommend it.

6.3.2 Usability Testing

To measure the usability of the artefact and quantify the data to allow evaluation in relevance to the current solutions, a time-based efficiency test was devised. The tester would be timed for how long it took them to locate a certain tool in each of the systems, which could be averaged across multiple testers and used to rank the solutions for usability (see Appendix: 6. Testing).

Table 8. Usability Test Results

Time (seconds)	***Systems***				
	AIO SRT	**HB CD PE**	**UB CD**	**KNOPPIX**	**LLTT**
T1	0.78	12.30	12.50	8.13	4.56
T2	3.56	9.30	16.54	93.52	19.20
T3	4.79	15.99	14.32	20.11	8.46
T4	33.23	6.98	22.26	46.33	7.35
T5	8.81	36.16	13.36	89.39	52.67
T6	9.49	60.59	24.10	23.56	20.66
AVERAGE	**10.11**	**23.54**	**17.18**	**46.84**	**18.82**
RANK	**1**	**4**	**2**	**5**	**3**

AIO SRT had the fastest user response time due to the applications being accessible via shortcuts on the desktop, making them immediately visible when booted. This design style was considered for the development of the artefact, however, due to the number of applications the result was cluttered (see Appendix: System Design). The artefact, LLTT, came third overall which is a credit to the menu system and when testers utilised the Help System tool the time was faster (see *T3 & T4*). However, a search tool could be implemented in the menu in the future development to improve the usability of the artefact.

7 AUTHOR'S DISCUSSION

7.1 EVALUATION

In relation to the current solutions, the artefact has demonstrated a method for making LiveCD repair tools more accessible to a wider range of users. The usability test showed that the Help System tool improved access to the commonly used tools but overall the artefact has slower time-based efficiency due to lacking a search function. Despite this, two-thirds of testers rated it 5/5 for usability and the artefact was stated as "*easy to understand and use*". Additionally, while the artefact contains similar types of tools to the other solutions, it often only has one application for each type of tool. This makes it more concise and easier to navigate but lacks tool redundancy and limits the choice for the user to try different applications that may perform better. Furthermore, as identified through testing feedback it was recognised that more categories of applications could be added, such as for BIOS management and drive diagnosis tools, which are supported in current solutions like Ultimate Boot CD. However, in public testing, 77% of respondents rated the artefact 5/5 for functionality, with comments including "*good range*", "*well-rounded*", and "*every tool I'd need*". The positive feedback indicates that the artefact has achieved its aims and the project has resulted in a product that can compete with the current, popular solutions.

Despite this, as a LiveCD computer repair solution, the method for operating the system may feature difficulties to less skilled users as the BIOS will most likely need to be configured. Specifically, the USB or CD will need to be selected as the boot device, which may be too complex for some users, as indicated by the preliminary questionnaire. However, these all-in-one solutions are praised by many as improving functionality, usability, and convenience over traditional individual application installation on the host machine:

> *"These applications are available without requiring configuration, installation, or administration by the end user"* - (Diesburg et *al.*, 2005).

Furthermore, the preliminary research suggests that people would use the artefact even if they have not used a similar solution before (89% of respondents). As the scope of the project covers issues including e-waste, software accessibility, the R2R and FOSS trends, the impact of any solution that enables consumers to have more control over their devices; such as diagnosing common computer problems, is a possibility for growth in the industry.

The use of an agile methodology proved suitable and successful due to the relaxed development strategy that was the result of having little experience developing an artefact of this size. The TDD approach ensured that regular and scheduled tests of components aided in the successful down-up method to warrant proper operation of the system as a whole; despite the tests being performed informally. The approach combined with the scrum management techniques works well to produce an overall strategy that aided the development of the artefact and helped in documenting the development for the purpose of this dissertation; plus, for project collaboration purposes. Ultimately, the project management methodology has proved to be successful as the artefact and project were completed on or before the deadline.

7.2 Future Developments

The main work will be in continual development and updating the artefact for compatibility, updating, and expansion, with ideas expressed in the evaluation for areas of growth. As seen with KNOPPIX/Maths, there is a potential to shape systems to a specific purpose, suggesting that the same could be applied to LLTT. Continuing from one of the aspects of LLTT, namely security, a specialised version with a more varied and advanced range of anti-malware could be produced. Similarly, after using Hiren's Boot CD PE that operates in Windows 10 preinstallation environment (PE), a solution offering both Windows and Linux based systems could expand the functionality. Moreover, in public testing, the most common comment for why they would be averse to using the tool was because it was Linux based, suggesting they would be more likely to use it if it was Windows based. Alternatively, it could prove valuable to use a different method to create the liveCD, such as Linux from scratch. This could improve the conciseness of the product by having greater control of the OS foundation; however, the learning curve for this dissertation was considered too great.

In addition to artefact development, to improve the overall project a website with a support forum, development blog and download access the toolkit could be developed to progress the project and create a development network. Furthermore, 89.80% of respondents in the preliminary research stated that a support website with a forum would encourage them to use the toolkit, highlighting its significance to the project.

7.3 Conclusion

The proposed solution is an all-in-one toolkit based on a Lubuntu LiveCD with an array of highly reputable technician tools, custom scripts and a carefully designed environment with a focus on accessibility and usability features. The artefact is needed as the current solutions tend to focus on functionality over usability, and next to none account for accessibility features. LLTT maintains the functionality by providing similar and effective applications while improving usability and accessibility through help systems and locally stored tool guides that describe the purpose and how to use the tools.

The artefact has been proven in multiple testing stages and associated feedback to be a functional, accessible and usable toolkit for a range of purposes. Furthermore, it has been found to be suitable for users with a variety of IT skillsets, which makes the tools accessible to an audience who may have been unable or uncomfortable using such tools before. Thus, it enables typical, non-technical users to take greater control over the performance of their technology. Despite its incompletion, it has met the desired aims and objectives, while providing plenty of opportunity for future development by the author or a third party. As the artefact was developed using open source software and an aim of the dissertation was to improve awareness of the importance of FOSS, the project is helping to contribute to the movement. When the artefact has been revised and improved with the future developments, it would be released publicly with a GPL license to allow and promote its use, replication, and development.

The potential benefits of the artefact are varied; however, they demonstrate the principle that an open source development can have impacts wider than the intended industry. The knock-on effects of users being able to repair their own devices will have economic, social and environmental implications. From consumers not having to pay for the tools to repair, to them not requiring technical support regarding problems they could easily resolve themselves when provided the access and knowledge to do so. From users having more choice regarding the safety and

performance of their systems, to the environmental benefit due to less e-waste and less electrical component production. The Live Linux Technician Toolkit demonstrates how this can be done.

8 REFERENCES

Alves, D. & Cagnin, M. (September 2014), 'Accessibility in development of free software projects', *2014 XL Latin American Computing Conference (CLEI),* IEEE, Uruguay.

Alzghoul, A.; Backe, B.; Löfstrand, M.; Byström, A.; Liljedahl, B. (October 2014), 'Comparing a knowledge-based and a data-driven method in querying data streams for system fault detection: A hydraulic drive system application', *Computers in Industry*, Vol.65, Issue 8, pp.1126-1135.

Awotar, M. & Sungkur, RK. (May 2018), 'Optimization of Software', *Procedia Computer Science*, Vol.132, pp.1804-1814.

BECTA (January 2009), 'Standards and guidelines for making accessible software'. Available at: www.becta.org.uk (Accessed: 14/11/2018)

Bennett, S. & Hollander, C. (1981), 'DART: An Expert System for Computer Fault Diagnosis', *International Joint Conferences on Artificial Intelligence*, USA. Available at: https://www.ijcai.org/Proceedings/81-2/Papers/050.pdf

Besson, F; Moura, P.; Kon, F. & Milojicic, D. (January 2015), 'Bringing Test-Driven Development to web service choreographies', *The Journal of Systems & Software*, Vol.99, pp.135-154.

Bhat, T. & Nagappan, N. (2006), 'Evaluating the efficacy of test-driven development: industrial case studies', *ISESE'06 – Proceedings of the 5th ACM-IEEE International Symposium on Empirical Software Engineering*, pp.356-363.

Card, S. (2014), 'Designing with the Mind in Mind (Second Edition)', *Simple Guide to Understanding User Interface Design Guidelines*, pp.9-11.

Cooke, W. (August 2018), *Installation/SystemRequirements*. Available at: https://help.ubuntu.com/community/Installation/SystemRequirements (Accessed: 13/5/2019)

Diesburg, S.; Gray, P. & Joiner, D. (April 2005), 'High performance computing environments without the fuss: the Bootable Cluster CD', *19th IEEE International Parallel and Distributed Processing Symposium 2005*, pp.8, IEEE, USA.

Ebert, C. (May 2008), 'Open Source Software in Industry', *IEEE Software*, Vol.25, pp.52-53, IEEE.

Gao, Z; Cecati, C. & Ding, S. (June 2015), 'A Survey of Fault Diagnosis and Fault-Tolerant Techniques—Part II: Fault Diagnosis With Knowledge-Based and Hybrid/Active Approaches', *IEEE Transactions on Industrial Electronics*, Vol.62, Issue 6, pp 3768 – 3774.

Gibson, G. & Schroeder, B. (2007a), 'Disk Failures in the Real World', *in Proceedings of FAST'07 5th USENIX Conference on File and Storage Technologies*, pp. 1-16.

Gibson, G. & Schroeder, B. (2007b), 'Understanding failures in petascale computers', *Journal of Physics: Conference Series*, Vol.78, IOP Publishing LTD.

Gibson, G. & Schroeder, B. (2009), "Computer Failure Data Repository (CFDR)". Available at: https://www.usenix.org/legacy/event/osdi06/posters/schroeder.pdf (Accessed: 1/1/2019)

GNU (2019), 'GNU Accessibility Statement - GNU Project - Free Software Foundation'. Available at: https://www.gnu.org/accessibility/accessibility (Accessed: 1/1/2019)

Goncalves de Branco, R.; Cagnin, M.; Barroso P. & Paiva, D. (June 2014), 'AccTrace: Accessibility in Phases of Requirements Engineering, Design, and Coding Software', *14th International Conference on Computational Science and Its Applications*, pp.225-228.

Hamada, T. (6 February 2009), 'KNOPPIX/Math: a live system for enjoying mathematics with computer', *ACM Communications in Computer Algebra*, Vol.42(3), pp.175-176.

Haugsted, L. (May 2008), ''Scripted' Solving', *Multichannel News*, May 12, 2008, Vol.29(19), p.32.

Hongxia, P.; Jinying, H. & Guangmin, L. (December 2008), 'Fault diagnosis of circuit board based on fault tree', *2008 10th International Conference on Control, Automation, Robotics and Vision*, IEEE, Vietnam.

HSE (2019), 'Waste Electrical and Electronic Equipment recycling (WEEE)', Health and Safety Executive. Available at: http://www.hse.gov.uk/waste/waste-electrical.htm (Accessed: 14/03/2019)

Hubbard, J. (1999), *FreeBSD Handbook*, The FreeBSD Documentation Project. Available at: https://docs.freebsd.org/doc/3.3-RELEASE/usr/share/doc/handbook/index.html (Accessed: 27/03/2019)

ISO (2018), 'ISO/DIS 9241-11.2(en) Ergonomics of human-system interaction — Part 11: Usability: Definitions and concepts', Online Browsing Platform. Available at: https://www.iso.org/obp/ui/#iso:std:iso:9241:-11:dis:ed-2:v2:en (Accessed: 11/3/2019)

ISO (2019), 'ISO 9241-171:2008'. Available at: https://www.iso.org/standard/39080.html (Accessed: 1/1/2019)

Joshi, A; Sarda, N.L. & Tripathi, S. (2010), 'Measuring effectiveness of HCI integration in software development processes', *The Journal of Systems & Software*, Vol.83(11), pp.2045-2058.

Karac, I. & Turhan, B. (July 2018), 'What Do We (Really) Know about Test-Driven Development?', *IEEE Software*, Vol.35(4), pp.81-85.

Kratov, S. V. (August 2011), 'The free software demonstration platform', *Proceedings of 2011 6th International Forum on Strategic Technology,* IEEE, China.

Kruchten, P. (April 2013), 'Contextualizing agile software development'*, Journal of Software: Evolution and Process*, Vol.25(4), pp.351-361.

Kshirsagar, R. & Patrikar, R. (December 2008), 'A novel fault tolerant design and an algorithm for tolerating faults in digital circuits', *2008 3rd International Design and Test Workshop*, IEEE, Tunisia.

Lazar, J.; Feng, J. & Hochheiser, H. (2017), 'Chapter 14 - Online and ubiquitous HCI research', *Research Methods in Human Computer Interaction (Second Edition*), pp.411-453.

Li, x.; Huang, M. C.; Shen, K. & Chu, L., (2007), "An Empirical Study of Memory Hardware Errors in a Server Farm", *HOTDEP Workshop*. Available at: http://ftp.cs.rochester.edu/u/kshen/papers/hotdep2007.pdf (Accessed: 1/1/2019)

Maier, P.; Kleeberger, V.; Mueller-Gritschneder, D. & Schlichtmann, U. (January 2018), 'Fault Injection for Test-Driven Development of Robust SoC Firmware', *ACM Transactions on Embedded Computing Systems (TECS),* Germany, Vol.17, pp.1-26.

May, J. (2001), 'Human–Computer Interaction', *International Encyclopedia of the Social & Behavioral Sciences,* pp.7031-7035.

Mifsud, J. (22 June 2015), 'Usability Metrics – A Guide To Quantify The Usability Of Any System'. Available at: https://usabilitygeek.com/usability-metrics-a-guide-to-quantify-system-usability/ (Accessed: 11/03/2019)

Miller, S. A. (31 March 2015), 'EXPLAINING THE LACK OF LARGE SCALE STUDIES IN IT'. Available at: https://smbitjournal.com/2015/03/explaining-the-lack-of-large-scale-studies-in-it/ (Accessed 1/1/2019)

Moll, R.; Prokop, M. & Morgenstern, H. (September 2009), 'Workshop: Digital Discovery with Bootable CDs', *2009 Fifth International Conference on IT Security Incident Management and IT Forensics,* IEEE, Germany.

Moreno, A.; Seffah, A; Capilla, R & Sanchez-Segura, M. (April 2013), 'HCI Practices for Building Usable Software', Computer, Vol.46(4), pp.100-102.

Nightingale, E. B.; Douceur, J. & Orgovan, V. (April 2011), 'Cycles, Cells and Platters: An Empirical Analysis of Hardware Failures on a Million Consumer PCs', Proceedings of EuroSys 2011, ACM, USA.

OPA (2019), 'About', Open Repair Alliance. Available at: https://openrepair.org/about/ (Accessed: 24/03/2019)

Paul, J. (15 July 2018), *Lubuntu 18.04 Review: Stable and Dependable As Always*. Available at: https://itsfoss.com/lubuntu-review/ (Accessed: 13/5/2019)

PMI (2019), *What is Project Management?*. Available at: https://www.pmi.org/about/learn-about-pmi/what-is-project-management (Accessed: 27/03/2019)

Powell, D. (2001), *A Generic Fault-Tolerant Architecture for Real-Time Dependable Systems*. Springer US.

Randell, B.; Laprie, J-C. & Avizienis, A. (May 2001), 'Fundamental Concepts of Computer System Dependability'*, IARP/IEEE-RAS Workshop on Robot Dependability: Technological Challenge of Dependable, Robots in Human Environments – Seoul*, Korea.

Robles, G. (September 2006), 'Empirical Software Engineering Research on Free/Libre/Open Source Software', *2006 22nd IEEE International Conference on Software Maintenance*, IEEE, USA.

Romano, S.; Fucci, D.; Scanniello, G.; Turhan, B. & Juristo, N. (September 2017), 'Findings from a multi-method study on test-driven development'*, Information and Software Technology*, Vol.89, pp.64-77.

Ruan, X.; Manzanares, A.; Yin, S.; Nijim, M. & Qin, X. (July 2009), 'Can We Improve Energy Efficiency of Secure Disk Systems without Modifying Security Mechanisms?', *2009 IEEE International Conference on Networking, Architecture, and Storage*, IEEE, China.

Sanker, S. & Gurumurthi S. (August 2013), 'Soft Failures in Large Datacenters', *IEEE Computer Architecture Letters,* Vol.13, Issue.2, pp.105 – 108.

Shubin, H. & Ulrich, J. (1982), 'IDT: AN INTELLIGENT DIAGNOSTIC TOOL', AAAI-82 Proceedings. Available at: https://www.aaai.org/Papers/AAAI/1982/AAAI82-069.pdf (Accessed: 1/1/2019)

SK (28 March 2016), *Pinguy Builder – Build your own, custom Ubuntu OS*. Available at: https://www.ostechnix.com/pinguy-builder-build-custom-ubuntu-os/ (Accessed: 8/5/2019)

Smith, B. (2007), 'A Quick Guide to GPLv3', Free Software Foundation, Inc. Available at: https://www.gnu.org/licenses/quick-guide-gplv3.html (Accessed: 24/02/2019)

Špundak, M. (19 March 2014), 'Mixed Agile/Traditional Project Management Methodology – Reality or Illusion?', *Procedia - Social and Behavioral Sciences*, Vol.119, pp.939-948

Statcounter (March 2019), *Desktop Operating System Market Share Worldwide*. Available at: http://gs.statcounter.com/os-market-share/desktop/worldwide/ (Accessed: 04/04/2019)

Statista (2019), 'UK households: ownership of home computers 1985-2018'. Available at: https://www.statista.com/statistics/289191/household-penetration-of-home-computers-in-the-uk/ (Accessed: 24/03/2019)

Suzak, K.; Iijima, K.; Yagi, T. ; Tan, H. & Goto, K. (2005), 'SFS-KNOPPIX', *Fourth IEEE International Symposium on Network Computing and Applications 2005*, pp.247-250, IEEE, USA.

Sztandera, M. (December 2010), 'Optimal Inventory of Computer Repair Parts: A Fuzzy Systems Approach', *2010 IEEE 16th Pacific Rim International Symposium on Dependable Computing*, Japan, pp.225-226.

Tian, H. & Jia, Li. (December 2018), 'Dynamic process fault diagnosis using improved Fisher discriminant analysis – An approach towards IoT', *Cognitive Systems Research*, December 2018, Vol.52, pp.261-266.

Vetrivel, U. & Pilla, K. (2008), 'Open discovery: An integrated live Linux platform of Bioinformatics tools', *Bioinformation*, 2008, Vol.3(4), p.144-146. Available at: https://www-ncbi-nlm-nih-gov.plymouth.idm.oclc.org/pmc/articles/PMC2637960/ (Accessed: 16/10/2018)

Wei, K.; Crowston, K.; Eseryel, U. Y. & Heckman, R. (July 2017), 'Roles and politeness behavior in community-based free/libre open source software development', *Information & Management*, July 2017, Vol.54(5), pp.573-582.

Williams, T. (February 1994), 'Using a risk register to integrate risk management in project definition', *International Journal of Project Management*, Vol.12, pp.17-22.

World Bank Group (WBG) (24 July 2018), 'World Bank Group Announces New Commitments on Disability Inclusion'. Available at: https://www.worldbank.org/en/news/press-release/2018/07/24/world-bank-group-announces-new-commitments-on-disability-inclusion (Accessed: 1/1/2019)

W3C (5 June 2018), 'Web Content Accessibility Guidelines (WCAG) 2.1', W3C Recommendation 05 June 2018. Available at: https://www.w3.org/TR/2018/REC-WCAG21-20180605/ (Accessed: 1/1/2019)

Yingkui, Z.; Jing, Z. & Liye, W. (December 2010), 'Justification of Free Software and its Enlightenment', *2010 Second World Congress on Software Engineering*, IEEE, China.

Yu, G.; Wang, L.; Meng, X. & He, Q. (01 July 2012), 'LXtoo: an integrated live Linux distribution for the bioinformatics community', *BMC Research Notes*, Vol.5(1), pp.360. Available at: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3461469/ (Accessed: 16/10/18)

9 APPENDIX

1) **Research**
    a. Existing solutions review
    b. Preliminary Questionnaire - Questions
    c. Preliminary Questionnaire - Results

2) **Project Management**
    a. Project Proposal
    b. GANTT
    c. Risk Register
    d. Feasibility
    e. Scrum Product Backlog
    f. Scrum Sprint Log
    g. Logs & meetings

3) **System Design**
    a. Storyboard

4) **System Development**
    a. Desktop
    b. Tool Guides
    c. Tool Development
    d. Menu Development
    e. Lab reports

5) **Testing**
    a. Solution Testing
    b. Public Testing Questionnaire

6) **Poster**

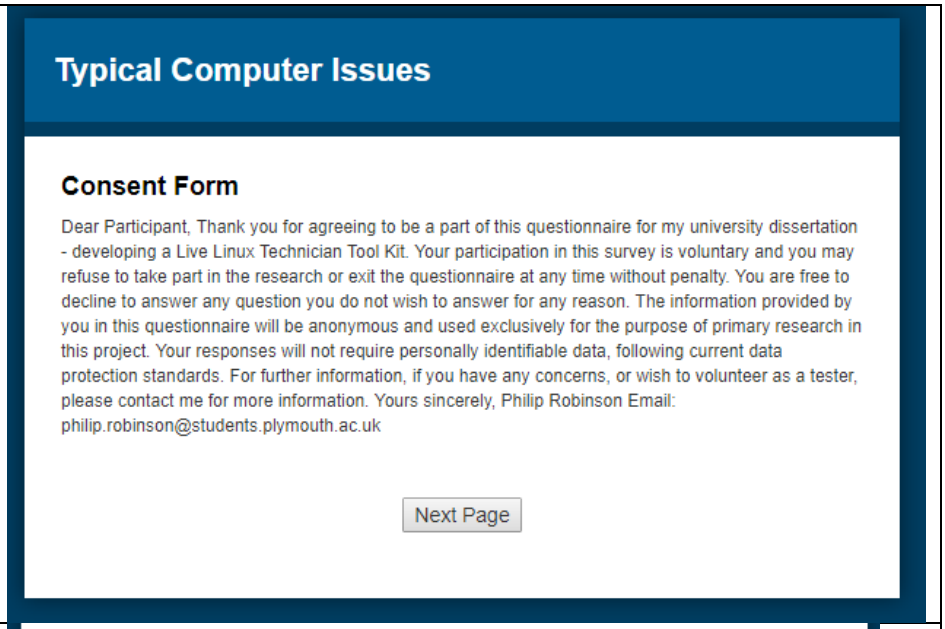
1. Research

Review of Existing Solutions

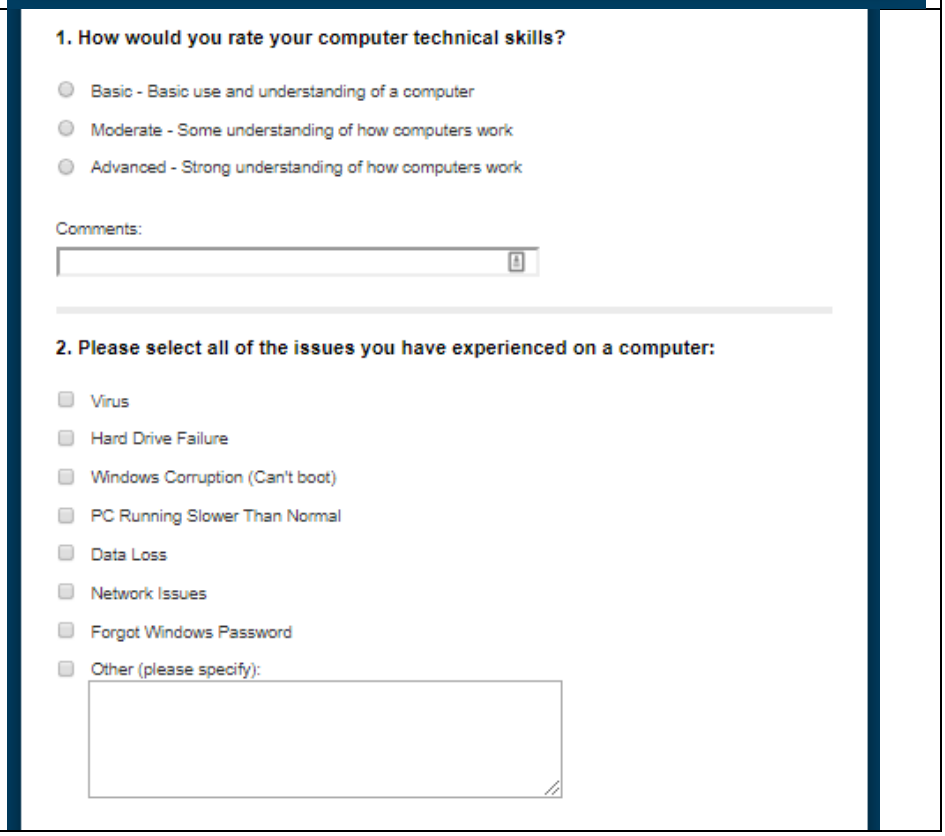
Name	All in One System Rescue Toolkit	Hiren's BootCD PE	[redacted]	KNOPPIX
Overview	https://paulis-a-geek.org/aio-srt/ A Lubuntu GUI based live CD for system administrators + Windows .exe application with over 30 tools. Under 700MB file size.	https://www.hirensbootcd.org/ Windows 10 PE x64 environment 1292MB ISO file size.	http://www.ultimatebootcd.com/ Unix CLI based system plus PartedMagic Unix GUI OS. 696MB ISO file size.	www. Knopper.net Debian with a choice of LXDE, KDE or GNOME desktop i686 4.4GB ISO file size.
Functionality	Light and streamlined kit for most repair situations with 15 tools.	Comprehensive with 68 Tools	140 Tools	Full desktop
Usability	Simple layout using the desktop as the main application interface	Windows 10 based so very familiar operating procedure. However, so many tools that it can be difficult to sort.	Simple command line interface for keyboard navigation. Plus access to PartedMagic Unix OS with more tools.	A typical Unix-based desktop with many menu categories
Applications	•Boot Repair – simple tool to recover access to your Operating Systems •ClamAV – open source antivirus engine •Xfburn – simple CD/DVD burning tool •Clonezilla – partition and disk imaging/cloning program •Disks – disk utility to manage SMART and disk partitions •Disk Usage Analyzer – graphical application to analyse disk usage •Galculator – a GTK 2 / GTK 3 algebraic and RPN calculator •GParted – partition editor for graphically managing your disk partitions •LBreakout2 – breakout-style paddle ball game •Mprime – Mersenne prime CPU torture test •NT Password Reset – Windows password removal utility •Nwipe – secure disk •PhotoRec – file data recovery software •System Profiler and Benchmark – displays information about your hardware •Terminal – Lubuntu Command Line Interface (CLI) •TestDisk – data recovery software designed to help recover lost partitions •Web Browser – a lightweight, fast, and free web browser	•2 Boot Repair Tools (Bootice) •16 Drive Tools (Recuva, Defraggler, HDTune) •4 Drive Explorer Tools (Runtime Captain Nemo) •5 Drive Imager Tools (Acronis TrueImage 2018, Macrium Reflect PE) •2 Partition Tools •6 Security Tools (Malwarebytes, ProduKey, NT Password Edit) •14 System Tools (CPU-Z, GPU-Z, CCleaner) •7 Network Tools (Aero Admin, Teamviewer and PENetwork) •12 Misc Tools (Rufus, 7zip, Notepad++, HDD format)	•7 BIOS Tools (BIOS, CMOSPWD, WipeCMOS) •14 CPU Tools (CPUstress, CPUinfo, x86test) •14 Boot Tools (BOOTMGR, EditBINI, Super Grub Disk) •6 Data Recovery Tools (PCLoginNow, PhotoRec, TestDisk) •15 Drive Info Tools (DISKINFO, GSmartControl, DiskCheck) •22 Drive Diagnosis Tools (ES-Tool, GWSCAN, VIVARD) •9 Drive Cloning Tools (Clonezilla, EaseUS Disk Copy, partimage) •4 Drive Edit Tools (Disk Editor) •11 Drive Wiping Tools (CopyWipe, Darik's Boot and Nuke) •6 Drive Installation Tools (DiscWizard) •13 Partition Manger Tools (GParted) •6 Memory Tools (Memtest86, Windows Memry Diagnostic) •9 System Tools (AIDA16, HWiNFO, System Speed Test 32) •8 Misc Tools (ClamScan AV, Parted Magic) •6 Peripheral Tools (Video Memory Test,	Full desktop experience with productivity and media software. Features default GNOme system tools plus additional tools, such as Disk Usage Analyzer, File Manager, System Profiler and Benchmark, Network monitoring tools, Disk Analysers (Gdisk, GuyManager, ClamTK, etc.

Preliminary Questionnaire - Questions

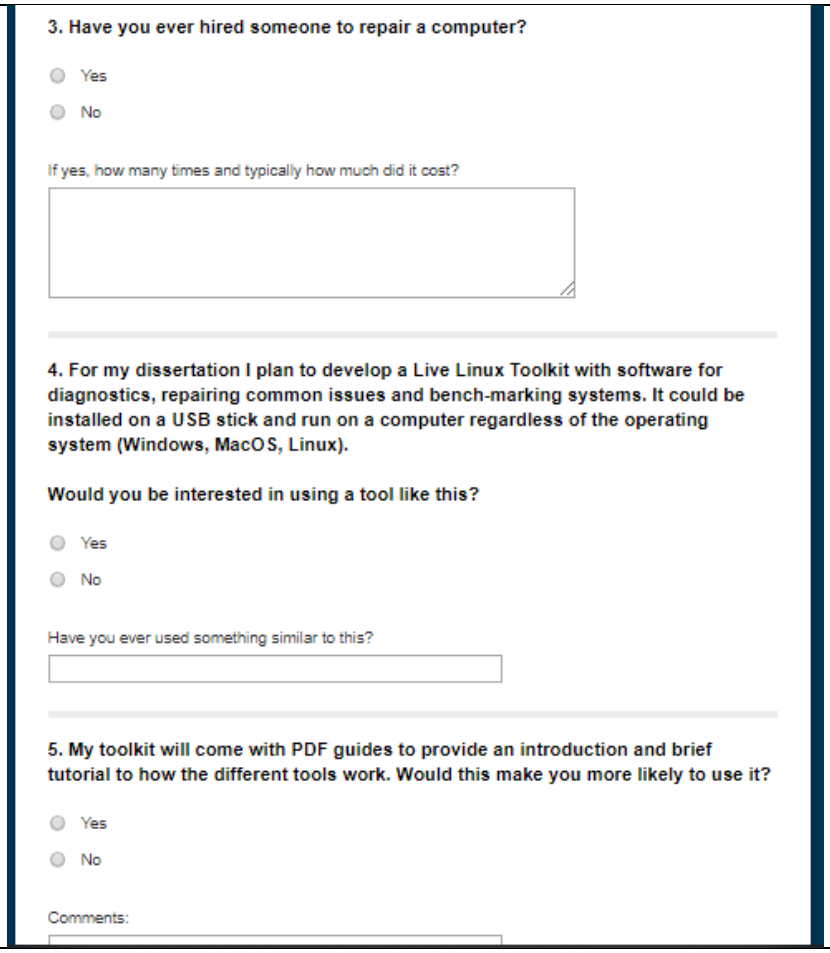
1.



2.



3.



3. Have you ever hired someone to repair a computer?

- Yes
- No

If yes, how many times and typically how much did it cost?

4. For my dissertation I plan to develop a Live Linux Toolkit with software for diagnostics, repairing common issues and bench-marking systems. It could be installed on a USB stick and run on a computer regardless of the operating system (Windows, MacOS, Linux).

Would you be interested in using a tool like this?

- Yes
- No

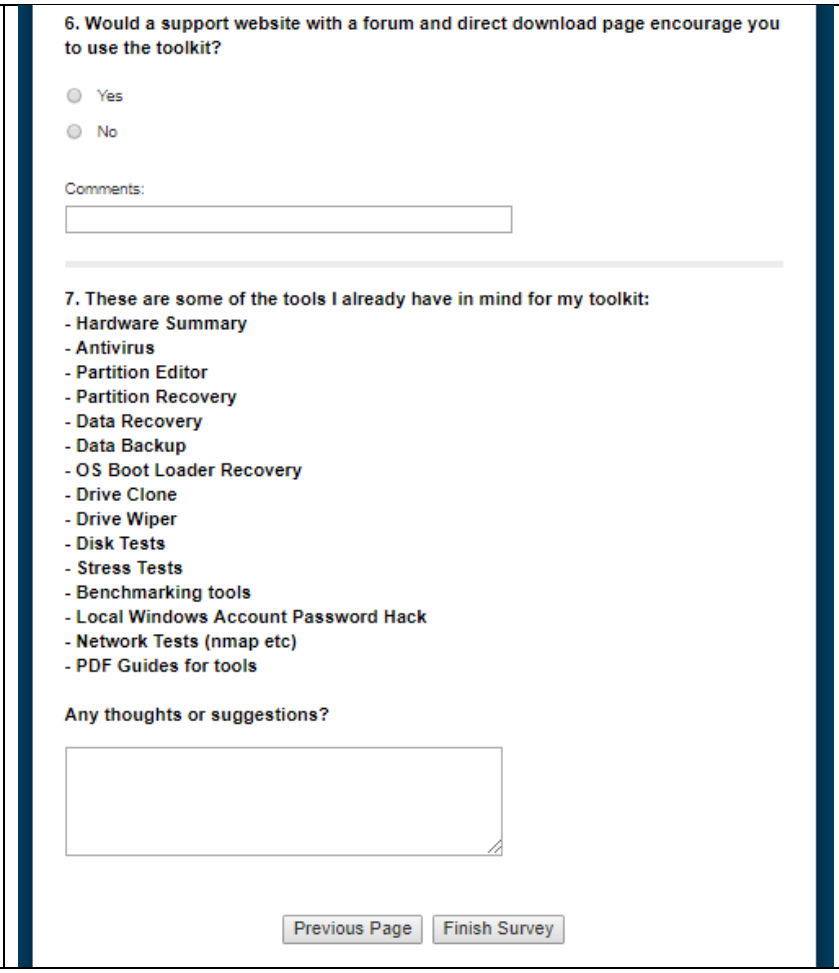
Have you ever used something similar to this?

5. My toolkit will come with PDF guides to provide an introduction and brief tutorial to how the different tools work. Would this make you more likely to use it?

- Yes
- No

Comments:

4.



6. Would a support website with a forum and direct download page encourage you to use the toolkit?

- Yes
- No

Comments:

7. These are some of the tools I already have in mind for my toolkit:
- Hardware Summary
- Antivirus
- Partition Editor
- Partition Recovery
- Data Recovery
- Data Backup
- OS Boot Loader Recovery
- Drive Clone
- Drive Wiper
- Disk Tests
- Stress Tests
- Benchmarking tools
- Local Windows Account Password Hack
- Network Tests (nmap etc)
- PDF Guides for tools

Any thoughts or suggestions?

Previous Page | Finish Survey

Preliminary Questionnaire – Results

Q1.

1. How would you rate your computer technical skills?

		Response Percent	Response Total
1	Basic - Basic use and understanding of a computer	33.33%	17
2	Moderate - Some understanding of how computers work	45.10%	23
3	Advanced - Strong understanding of how computers work	21.57%	11

Analysis							
	Mean:	1.88	Std. Deviation:	0.73	Satisfaction Rate:	44.12	answered 51
	Variance:	0.54	Std. Error:	0.1			skipped 0

Comments: (6)

1	Probably somewhere between basic and moderate?
2	Some level of computer systems knowledge but little hands-on experience
3	Parallel programming in C/C++
4	I've replaced a fan and cd drive!
5	Advanced for Year 9
6	I'm easily frustrated by computers too.

Q2.

2. Please select all of the issues you have experienced on a computer:

		Response Percent	Response Total
1	Virus	74.51%	38
2	Hard Drive Failure	43.14%	22
3	Windows Corruption (Can't boot)	33.33%	17
4	PC Running Slower Than Normal	86.27%	44
5	Data Loss	37.25%	19
6	Network Issues	78.43%	40
7	Forgot Windows Password	29.41%	15
8	Other (please specify):	11.76%	6

Analysis							
	Mean:	15.63	Std. Deviation:	23.51	Satisfaction Rate:	166.95	answered 51
	Variance:	552.83	Std. Error:	3.29			skipped 0

Other (please specify): (6)

1	Mechanical failure - ball-bearing in laptop fan loose, caused overheating
2	Overheating - CPU
3	Shelf full of books fell on open laptop
4	Malware

Q3.

3. Have you ever hired someone to repair a computer?

		Response Percent	Response Total
1	Yes	35.29%	18
2	No	64.71%	33

Analysis							
	Mean:	1.65	Std. Deviation:	0.48	Satisfaction Rate:	64.71	answered 51
	Variance:	0.23	Std. Error:	0.07			skipped 0

If s, how many times and typically how much did it cost? (16)

1	Once, cost £35 to wipe and reinstall OS
2	Once, was to remove a virus and repair hard drive. It was a bit pricey around 60 to 70 pounds.
3	It was a friend who did it free but he had to take it away to sort out x that was about 15 years ago x
4	Hardware repairs - under insurance - 2 times
5	Once, to replace a hard drive. £50
6	Free
7	$200
8	Once, for a hard drive. About £80
9	2 £100
10	12 ish times. And he did it as a hobby so around £10 per hour
11	Twice, I think £50 each time but I can't be sure.
12	6/8 From £25 —£60
13	If dad counts then yes. He's my go to, don't know what I'd do without him!
14	Only once and as it was a family friend just £30 I believe
15	Twice - covered by applecare
16	Over the last 20 years probably 6 times cost varying from £50 to £150

Q4.

4. For my dissertation I plan to develop a Live Linux Toolkit with software for diagnostics, repairing common issues and bench-marking systems. It could be installed on a USB stick and run on a computer regardless of the operating system (Windows, MacOS, Linux).Would you be interested in using a tool like this?

		Response Percent	Response Total
1	Yes	95.92%	47
2	No	4.08%	2

Analysis							
	Mean:	1.04	Std. Deviation:	0.2	Satisfaction Rate:	4.08	answered 49
	Variance:	0.04	Std. Error:	0.03			skipped 2

Have you ever used something similar to this? (28)

1	No
2	No
3	No but I like the concept, I haven't heard of something similar before
4	No
5	No
6	I would be keen to use it, so long as it thoroughly shows me what to do - step by step.
7	Eh?
8	No
9	I have not used a like this before.
10	Have used bench marking before, a tool kit would be nice.
11	No
12	No
13	No
14	No
15	No
16	No
17	No
18	Not to my knowledge - only downloads/apps that claim t perform similar skills plus functions from my security provider.
19	Yes.... Rescatux
20	Possibly. If I understood what what it meant
21	No - seems like lots of overhead for trying to cover 3+ OSs
22	No
23	No
24	Hirens BootCD PE
25	Not
26	No
27	I usually use YouTube to solve the problems
28	HIRENS Boot CD

Q5.

5. My toolkit will come with PDF guides to provide an introduction and brief tutorial to how the different tools work. Would this make you more likely to use it?

		Response Percent	Response Total
1	Yes	87.76%	43
2	No	12.24%	6

Analysis							
	Mean:	1.12	Std. Deviation:	0.33	Satisfaction Rate:	12.24	
	Variance:	0.11	Std. Error:	0.05			

answered	49
skipped	2

Comments: (10)

1	Additional videos could be useful
2	As long as my computer is working enough to open the PDF! Make it available on mobile too?
3	Umm?
4	Helpful to have step by step guides to work alongside with
5	No one, including me reads instructions, especially since mobile apps
6	Keep it simple
7	Maybe. If they were super easy
8	If they were simple and easy to follow
9	Lots of these tools are lacking in documentation
10	Not sure

Q6.

6. Would a support website with a forum and direct download page encourage you to use the toolkit?

		Response Percent	Response Total
1	Yes	89.80%	44
2	No	10.20%	5

answered	49
skipped	2

Comments: (6)

1	Extra support is always a good thing when more support is available.
2	Confused!
3	Sounds like it would be for more technically capable than me.
4	Being able to share solutions and problems easily
5	A website would definitely make it easier to access, would allow user to user engagement and would make it easier for you to develop your product by using user feedback
6	Forums are very useful for suggestions, troubleshooting and the like, as long as it isn't a mess.

Q7.

7. These are some of the tools I already have in mind for my toolkit:- Hardware Summary- Antivirus- Partition Editor- Partition Recovery- Data Recovery - Data Backup- OS Boot Loader Recovery- Drive Clone- Drive Wiper- Disk Tests- Stress Tests- Benchmarking tools- Local Windows Account Password Hack- Network Tests (nmap etc)- PDF Guides for toolsAny thoughts or suggestions?

		Response Percent	Response Total
1	Open-Ended Question	100.00%	22

1	Fix it with minimal knowledge on my (the user) part - aka accessible for all stages of understanding.
2	USB Viewer (what was the last USB device connected date and time) Rootkit scanner (counts as antivirus) Data recovery is that include mobile ISO/Android? Maybe have open sorce and paid and give people the option for what they wish to use?
3	Disk Defragmentation?
4	These all sound great, but an accessible, easy to read guide on what these things are and how to use it. Otherwise brilliant.
5	A simple and easy to use bottlenecking tool would be handy. If it could show what parts of your pc are weakest and suggest upgrades with performance improvement forecasts that would be something I would be very interested in
6	Dictionary to tell you what some words mean in order to help diagnose what is wrong in order to be able to fix it
7	Sounds good but it needs to be basic enough for the average older user to understsnd.
8	If it makes the process of detecting and identifying problems easier at a reasonable cost then it would be a very handy toolkit. As computer shops tend to be very expensive.
9	Hides head in shame… I'm before basic on a computer!
10	Local password hack sounds like a bad idea. I'm not sure how liable you'd be if some were to use it for nefarious purposes/knacker thief pc… Similar for disk recovery of someone has deleted data they want to hide - this sort of stuff shouldn't be too accessible…
11	Sounds great
12	Can't think of anything else to add. Sounds great!
13	Can you list them again please. In English
14	Seems really handy but there's quite a range here and not all will be useful for everyone. i.e. Local Windows Account Password Hack isn't useful for *nix users. Similarly, nmap seems like a bit of an odd one out. I think the idea is really cool, though!
15	Nope, that sounds pretty good to me!
16	BCD/MBR editor(unless that is included under OS boot loader recovery) Hardware temperature monitor
17	Seems comprehensive
18	I'll be honest with you, I have no idea what most of this means but sounds good from what I do understand :)
19	If you want to charge for the software, may be worth not including the anti virus as it's something most people have already, you could sell it as a separate product
20	Something introductory on 'typical reasons your computer might be running slowly' with links to the relevant helpful toolkits
21	Certainly would be useful
22	Forensic Capabilities, SSH server on live CD (for headless systems)

2. Project Management

BSc (Hons) Applied Computing

Final Project & Ethical Approval Form
Project Proposal

1. Research details	
Student Name/s: **Philip Robinson**	**Supervisor:** Clint Washington
Programme title: **BSc (Hons) Applied Computing Technologies**	**Module Title:** **TRUR3044 Computing Project**
Project Title: Live Linux Technician Toolkit – "LLTT"	

Please complete this form along with Form A, B or C depending upon the subject of your research. Failure to give sufficient information under each section will delay the approval process. Any additional information may be provided on additional sheets of paper.

2. Aims and objectives

Aims

1. Research existing products and identify current problems and limitations
2. Quantify the usability and performance of these products
3. Design and develop a cross-platform AIO tool kit with both custom and open source software
4. Quantitative evaluation on usability and performance of developed artefact using research and testing

Objectives

1. Perform primary research using questionnaires to assess current opinion of products
2. Develop the product using current procedures for live OS development
3. Customise the operating system, consisting of tools for PC diagnostics, repair and benchmarking and include locally stored guides/ tutorial PDFs for guidance
4. Testing developed artefact using current software testing methodologies
5. Use findings from research to evaluate the developed artefact and make improvements

3. Design

1. A Traditional Project Management method for overall project to meet specific deadlines using standard project management tools

2. An Agile Project Management method for software product development

3. Summary of methodology (including participants and procedures)

- Qualitive and quantitative data research of existing products
- Use questionnaires to research what people with different IT skills would want from the product
- Design Linux ISO with all the desired software
- Make the working ISO (USB bootable)
- Testing – developer and third party
- Improve the ISO – changing/ updating software, redesigning layout etc
- Research deployment options

Research & References Sources

Primary

- Investigating and using existing products
- Interview peers
- Questionnaires (qualitive & quantitative) to general public and peers
- Software testing using Software Testing standards including ISO 29119

Secondary

- Research into functionality of software used and hardware
- Researching existing products

Sources

- Peer reviewed journals, conference proceedings, reports, reviews, books, thesis's etc.

4. Higher level clearance

Please identify whether your research intends to include any of the following aspects:

	Yes	No
• Active research with children	☐	☐
• Ingestion	☐	☐
• Invasive studies (ie. blood samples)	☐	☐
• Active research with vertebrates	☐	☐
• Research involving non-living human participants	☐	☐

If any of these aspects are being studied please refer to the *Notes for Guidance* in order to identify what supplementary paperwork must be included.

5. Signatures

6.a Supervisor

I have discussed the design of this research project with the applicant.

Lecturer Signed:_____ dated: _____

Supervisor Signed:_____ dated: _____

6.b Programme Level Ethics Review Panel Approval

We agree that this project has been:

☐ referred to College Ethics Board
☐ approved at Programme Level and the research may continue
☐ approved at Programme Level and the research may continue subject to completion of the agreed action points and/or clarification*

Signed:_____ Programme/Module Leader

Signed:_____ Critical friend

6.c College Ethics Board Approval

We agree that this project has been:

- ☐ approved at the College Ethics Board and the research may continue
- ☐ approved at the College Ethics Board and the research may continue subject to completion of the agreed action points*
- ☐ declined at the College Ethics Board. Review of methodology recommended.

Signed:_____ Ethics Board Chair or nominee

Date of Board:_____

* Attach agreed action points to the signed form

Form A – Use of Human Participants

Ethical procedures
Please indicate how you will ensure that your research conforms with each clause of the University of Plymouth's *Ethical Principles for Research Involving Human Participants.*

Informed consent

Participants will be asked to answer questionnaires confidentiality.

Please see attached consent form.

Openness and honesty

Participants will be made aware of the purpose of the research in consent form.

Right to withdraw

Participants will have the option to withdraw their data.

Protection from harm

Follow current Health & Safety standards.
Participants be made aware of possible dangers of using the product and recommended to use ISO virtually.

Debriefing

Participants will be made aware of the purpose of the research in consent form.

Confidentiality

Questionnaires will be anonymous and not require personally identifiable data, following current data protection standards.

Other professional bodies

N/A

Form B – Use of Non-Human Participants

Ethical procedures Please indicate <u>how</u> you will ensure that your research conforms with each clause of the University of Plymouth's *Ethical Principles for Research Involving Non-Human Participants.*
Legal and moral responsibilities
Avoidance of unnecessary study
Environmental impacts
Monitoring
Justifiable benefits

N/A

Form C – Use of Non-living Human Participants

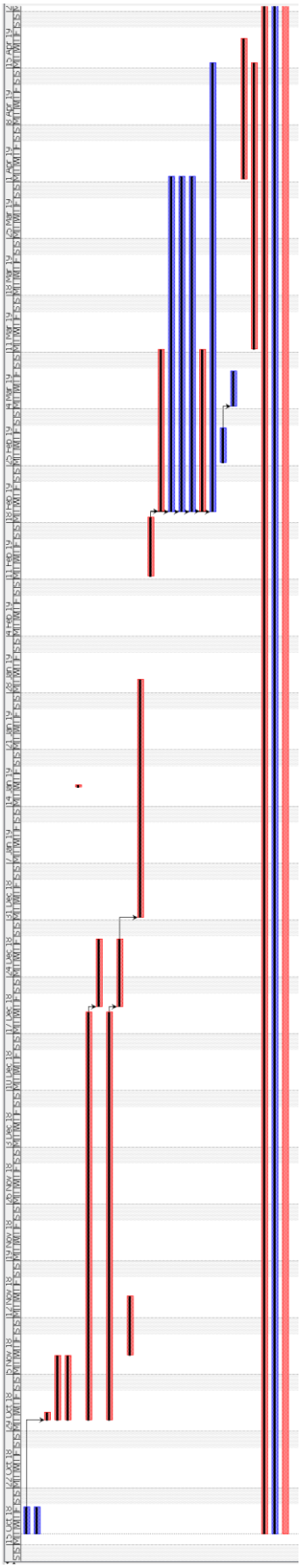
Ethical procedures Please indicate <u>how</u> you will ensure that your research conforms with each clause of the University of Plymouth's *Ethical Principles for Research Involving Non-Living Human Participants.*
Respect for the dead
Permission from guardians of the dead
Scientific considerations
Negotiated agreements

N/A

GANTT Chart – Tasks

1		Project Proposal	16/10/18 08:00	19/10/18 17:00			✓	
2		Consent Form Draft	16/10/18 08:00	19/10/18 17:00			✓	
3	✔	Gannt Chart Draft	30/10/18 09:00	31/10/18 09:00	1		✓	
4	✔	Feasibility Study	30/10/18 09:00	07/11/18 09:00			✓	
5	✔	Risk Register	30/10/18 09:00	07/11/18 09:00			✓	
6	✔	Mile Stone Review	16/01/19 09:00	16/01/19 17:00			✓	
7	✔	Research Alternatives	30/10/18 09:00	19/12/18 17:00			✓	
8	✔	Document Reseach	20/12/18 08:00	28/12/18 17:00	7		✓	
9	✔	Research Design Methods	30/10/18 09:00	19/12/18 17:00			✓	
10	✔	Document Research	20/12/18 08:00	28/12/18 17:00	9		✓	
11	✔	Questionnaire Development	07/11/18 09:00	14/11/18 17:00			✓	
12	✔	Write Literature Review	31/12/18 08:00	29/01/19 17:00	10		✓	
13	✔	Dissertation Plan	11/02/19 09:00	18/02/19 17:00			✓	
14	✔	Testing Section	19/02/19 09:00	11/03/19 09:00	13		✓	
15		Design Section	19/02/19 08:00	01/04/19 17:00	13		✓	
16	✔	Problem Section	19/02/19 08:00	01/04/19 17:00	13		✓	
17		Methodology Section	19/02/19 08:00	01/04/19 17:00	13		✓	
18	✔	Introduction Section	19/02/19 09:00	11/03/19 09:00	13		✓	
19		Evaluation & Conclusion Section	19/02/19 08:00	15/04/19 17:00	13		✓	
20	✔	Artefact Testing Plan	25/02/19 09:00	01/03/19 17:00			✓	
21		Testing	04/03/19 08:00	08/03/19 17:00	20		✓	
22	✔	Update Documentation	01/04/19 08:00	18/04/19 17:00			✓	
23	✔	Poster development	11/03/19 09:00	15/04/19 17:00			✓	
24		Deadline - Project Submission	16/10/18 08:00	24/05/19 17:00			✓	
25		Deadline - Poster Hand-in	16/10/18 08:00	26/04/19 17:00			✓	
26		Deadline - Project Submission	16/10/18 08:00	24/05/19 17:00				

GANTT Chart – Graph



Risk Register 6/11/19

Risk Title	Category	Description	Impact	Likelihood /5	Consequence /5	Rank /10	Trigger
example	*time/cost*	*example*	*on project*	1	4	5	*when to trigger contigency plan*
Absence of member	Workload, Time	If member is off sick etc.	Missing work/ meeting, getting behind	2	3	5	Member misses 2 sessions
Behind progress/Missed deadline	Time, Progress	Missed deadline	Missing deadline, catching up, project delay	2	4	6	Deadline missed
Miss meeting with Supervisor	Time, Progress	Failure to have weekly meeting	Progress not assessed, issues not discussed	2	4	6	Meeting missed
Missed Final Deadline	Time, Progress	Project is not completed	Failed Project	1	5	6	Deadlines are not met
Work Loss	Time, Progress	Work lost due to data loss/ computer failure etc	Loss of Progress, getting behind	1	5	6	Software/hardware issues
Design Fault	Time, Progress	Initial idea cannot be met due to unforseen issue	Delaying progress, getting behind	3	4	7	Repeated occurance of issues
Workload Issue	Workload, Time	Other university work pressure time management	Delaying progress, getting behind	2	2	4	Not meeting weekly time quota
Hardware/Resource Issues	Time, Cost	Laptop or resource failure eg. VM issues	Loss of Progress and works	1	5	6	Instance of issue
Licensing Issues	Workload, Progress	Software in solution have to be removed due to liscensing	Loss of Progress	2	2	4	Instance of issue
Software Incopatabilities	Workload, Progress	OS doesn't support programs or tools don't work correctly	Delaying progress, getting behind	4	3	7	Instance of issue
OS Restrictions	Time, Progress	OS choice isn't sufficient to meet project aims	Delaying progress, getting behind	2	5	7	Instance of issue
Unsufficient Questionnaire Results	Time, Progress	Questionnaire results are not conclusive/ useful	Delaying progress, getting behind	3	2	5	Instance of issue
Lack of Testers	Time, Progress	Sufficient amount of testers cannot be found	Delaying progress, getting behind	2	3	5	Instance of issue
Inadequate Knowledge	Time, Progress	Insufficient knowledge for software development	Delaying progress, getting behind	1	4	5	Instance of issue

Prevention Plan	Contingency Plan	Owner	Event Date & Resolved	Residual Risk	Comments
Mitigation	*Action if occurs*	*Team*	*Yes (date)*	*After treatment level /5*	*Example*
Planning, Redundancy	Workload shared to other member,	ME			
Planning, Redundancy	Workload shared to other member, catch up	ME	10/2/19 - Network Assignment Priority	3/5 (2nd Semester Assignments)	Missed 3 weeks
Planning, online discussion	Email Project Supervisor	ME		1/5 - low	
Deadlines and Redundancy set	Repeat module	ME			
Backup data and frequently save	Redo work	ME	19/1/19 - v2 VM error, all data lost	2/5	Virtual machine corruption due to Unix error.
Frequent set meetings with Supervisor and discussing progress	Urgent team meeting and w/ client	ME			
Mitigation with Planning and Redundancy	Meeting with Supervisor, time management	ME			
Backup of data and alternative plans	Redo work and find alternative resources	ME			
Ensure only freely distributable software is used	Find alternative	ME	Software package included propriety fonts		
Trial tools first and check for guides	Find alternative	ME	19/1/19 -during v1 & v2 development	3/5	Version2 in 18.10 incompatabilities
Research OS in- depth prior to development	Change OS based on available resources	ME			
Reviewing before wide distribution	Distribute further / extrapolate data	ME	Received 50+ responses	0/5	Great response amount with good results
Using peers as testers	Find testers through online groups	ME			
Research prerequiset knowledge required	Find easier alternative	ME			

Scrum Backlog

ID	Purpose/Story	Difficulty	Priority	Success
1	Remove uncessary files	1	3	Yes
2	Install desired packages	2	5	Yes
3	Test packages	4	4	Yes
4	Configure Conky	3	3	Yes
5	Remove password authentication	3	2	Yes
6	Develop custom scripts	5	3	Yes
7	Design desktop and GUI	3	4	Yes
8	Implement accessibility requirements	3	4	Yes
9	Test current state	4	4	Yes
10	Convert to liveCD ISO	2	3	Yes
11	Test ISO	5	3	Yes

Scrum Sprint Log

Backlog ID	Backlog Story	Sprint Task ID	Sprint Story
1	Remove unnecessary files	1.1	Install system updates
		1.2	Uninstall packages in Package manager
		1.3	Uninstall software in Software Store
		1.4	Manual file system configuration
2	Install desired packages	2.1	Install packages in Software Store
		2.2	Install packages in Package Manager
		2.3	Directly from source
3	Test packages	3.1	Execute /run programs to test if working
		3.2	Configure if not working
		3.3	Retest
		3.4	If not working - Install past version
		3.5	Test
		3.6	If not working - Remove nonfunctioning packages
4	Configure Conky	4.1	Execute Conky
		4.2	Configure Conky script
		4.3	Test
5	Remove password authentication	5.1	Configure Visudo script
		5.2	Test script
6	Develop custom scripts	*6.1.1*	Install *stress* command package
		6.1.2	Develop script using command
		6.1.3	Integrate Zenity dialogue box as GUI
		6.1.4	Test script
		6.2.1	Plan logic for Help script
		6.2.2	Write script
		6.2.3	Implement Zenity
		6.2.4	Test script
7	Design desktop and GUI	7.1	Add logos, icons, wallpapers
		7.2	Set taskbar settings
		7.3	Configure widgets
		7.4	Configure desktop environment to GNOME\LXDE
8	Implement accessibility requirements	8.1	Use accessibility checklist
		8.2	Navigation by keyboard
		8.3	Accessibility options menu
9	Test current state	9.1	Test each application
		9.2	Test custom scripts
		9.3	Test acessibility options
	Convert to liveCD ISO	10.1	Clean system
		10.2	Configure Pinguybuilder
		10.3	Use Pinguybuilder
11	Test ISO	11.1	Burn ISO to USB
		11.2	Predeployment test
		11.3	External testers

Feasibility Study

Feasibility Study

A description of the problem

Many common computer issues such as malware, non-bootable OS, and misdiagnosis. Lots of these can be fixed using free software tools, rather than paying for a technician to fix it – typically for £45 to £90 per hour (Which, 2018). However, the majority of current solutions are difficult to use and offer little local guidance on how to use the tools for system recovery and repair so are not suitable for less skilled or experienced users.

Project Plan

To design and develop a more advanced all-round technician toolkit that features locally stored guides for how to use the tools.

Identification of the probable costs and benefits

Benefits	Costs
Saving money by performing simple repairs	Time to research solution
Saving time	Time to develop solution
More control and security of data	Cost of USB flash drive
Increasing IT Skills	

Analysis of the feasibility of the project

Type of feasibility	Analysis
Technical	Access to design software – virtual machine
Economic	All software will be free or self-written. Eg. Linux.
Legal	Software should all be FOSS, and freely distributable to prevent copyright licensing issues.
Operational	Project management will viably achieve the result.
Scheduling	Time restriction of June 2019 deadline; however, for opportunities to iteratively develop the artefact further in any end is viable.

Logs & Meetings

Final Artefact Development Logs

<table>
<tr>
<td>Week 1:
14/1/19</td>
<td>Objectives:
Develop version 2 of the artefact in updated software and Lubuntu 18.10</td>
<td>Tasks Performed

Recreating LLTT in Lubuntu 18.10 i368 ISO

Remove redundant software
Downloading tools

ClamAV
Stress command
Conky
Gdisk
Gpart
Nwipe
Forensics
Network

Configure environment

Remove password
Accessibility
Design: wallpaper, icons
Menu

Designed logos

<u>Progress lost - infinite load loop due to faulty USB in VM</u>


https://lists.debian.org/debian-kernel/2018/10/msg00016.html
https://bugs.launchpad.net/ubuntu/+source/linux/+bug/1794922
</td>
</tr>
<tr>
<td>Week:
21/01/19</td>
<td>Objectives:
Develop version 2 in the original OS version.</td>
<td>Tasks Performed

Recreating LLTT in original Lubuntu 18.4 ISO

Update software
Remove redundant software
Add new tools
Customise the menu
Customise the environment

Develop tool guides

Disks
ClamAV

</td>
</tr>
</table>

Week: 28/01/19	**Objectives:** Continue development of V2.	#

Week: 4/02/19	**Objectives:** Continue development of V2.	**Tasks Performed**
		• Development of custom scripts
		  ○ Stress Script for stress testing CPU with Zenity
		  ○ Help System script for easy access to common tools
		  ○ Symptom Checker script for common issues and link to tools

Week: 11/02/19	**Objectives:** Continue development of V2.	**Tasks Performed**
		• Implementing Environmental Design & Accessibility Features
		  ○ Menu design with MenuLibre
		  ○ Import image backgrounds and logos
		  ○ Configure desktop and toolbar

Week: 18/02/19	**Objectives:** Continue development of V2.	**Tasks Performed**
		• Secondary testing
		  ○ Menu
		  ○ Custom scripts
		  ○ Tools

Week: 25/02/19	**Objectives:** Continue development of V2.	**Tasks Performed** • Final check of system • Creating live system ISO using PinguyBuilder • Using Rufus to burn ISO to USB

Meetings

Date: 06/11/2018	With: Mike	Agenda: Informal discussion of idea	Discussed 1. Audience a. Low skilled – type of software needed (maintenance, information) – easier to make b. Skilled – advanced tools for diagnostics and repair – more difficult to make as my skill level dependant 2. Distro a. Most appropriate for task 3. Automatic Detection Script a. Interface to find issues b. House keeping based c. Summary of system d. Password checker etc.	To do • Questionnaire to assess skill levels and required/desired tools • Research housekeeping/ automatic script idea as replacement or additional to main proposal
Date: 07/11/2018	With: Clint	Agenda: Update Check	Discussed 1. GANTT chart check – all up to date 2. Questionnaire drafts 3. Documentation review – Feasibility study and Risk Register	To do • Finish Questionnaire and distribute • Clint will check questionnaires • Update Documentation
Date: 05/12/2019	With: Clint	Agenda: Update Check	Discussed 1. GANTT chart check – all up to date 2. Review of Questionnaire results 3. Review of product development progress	To do • Continue Literature Review

			4. Review of Literature Review	• Continue product development • Think about options for testing and how to quantify results
Date: 08/01/2019	With: Clint	Agenda: Update Check	Discussed 1. GANTT chart check – all up to date 2. Review of product development progress 3. Discussion of Literature Review difficulties – should be used in the write up and used to express the change in fault diagnosis in non-IT fields	To do • Continue Literature Review • Continue product development and start testing
Date: 27/02/19	With: Clint	Agenda: Review of Progress	Discussed 1. Finished artefact – arrange testing for 6/03/19 with access students 2. Review of current dissertation write up	To do • Develop testing sheet for next week • Finish a chapter for review next week
Proceeding Meetings were used for reviewing chapter development				

Milestone Meeting

Milestone Meeting

16/01/19

Up-to-date Documentation

Gantt chart, risk register

Research Undertaken

Researched current solutions and evaluated them.

Interesting Points Discovered

Accessibility standards for software development and how to apply them.

Project Management Strategy and Current Position

Traditional Project Management method for overall project to meet specific deadlines using standard project management tools and an Agile Project Management method for software product development.

Project Write Up

Started writing the first section of the dissertation and writing the literature review.

Artefact Development

Finished version 1 of artefact, currently evaluating and planning for a next version.

Thoughts about Dissertation Document

Used the given dissertation layout and sections, currently writing the introduction.

Thoughts about Poster

Planning on basing it on standard scientific layouts and using best bits from last years' posters.

Future Work

Planning testing once the next version of artefact is complete.

AOB

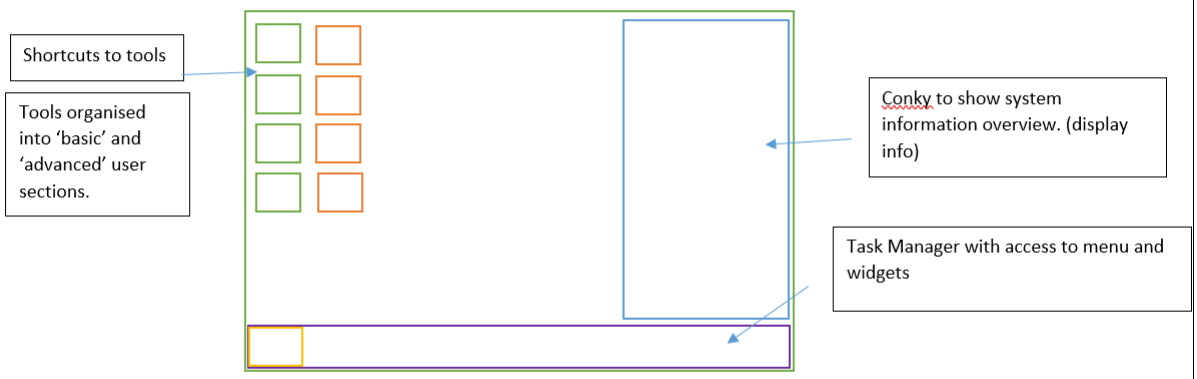
Literature review difficulties

3. System Design

Story Board

Program Title:	LLTT
Purpose of Program:	Collection of software for diagnostics, repair, recovery and benchmarking.
Program Designer:	10572891
Screen Reference/Title:	Desktop v1.1

Screen Design

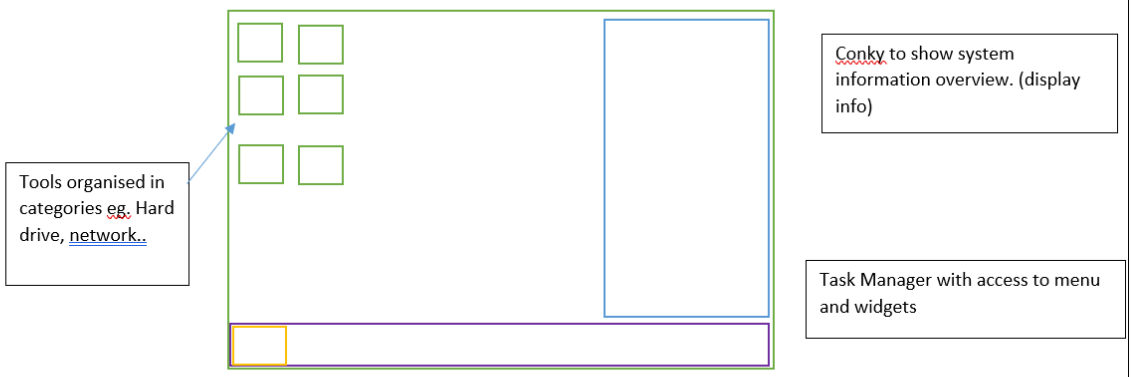


Details:

- Design features
- Icons sizes
- Mouse or keyboard navigation
- Display, icons and accessibility settings are changeable.

Screen Reference/Title:	Desktop v1.2

Screen Design



Details:

- Design features
- Icons sizes
- Mouse or keyboard navigation
- Display, icons and accessibility settings are changeable.

Screen Reference/Title:	Desktop v2

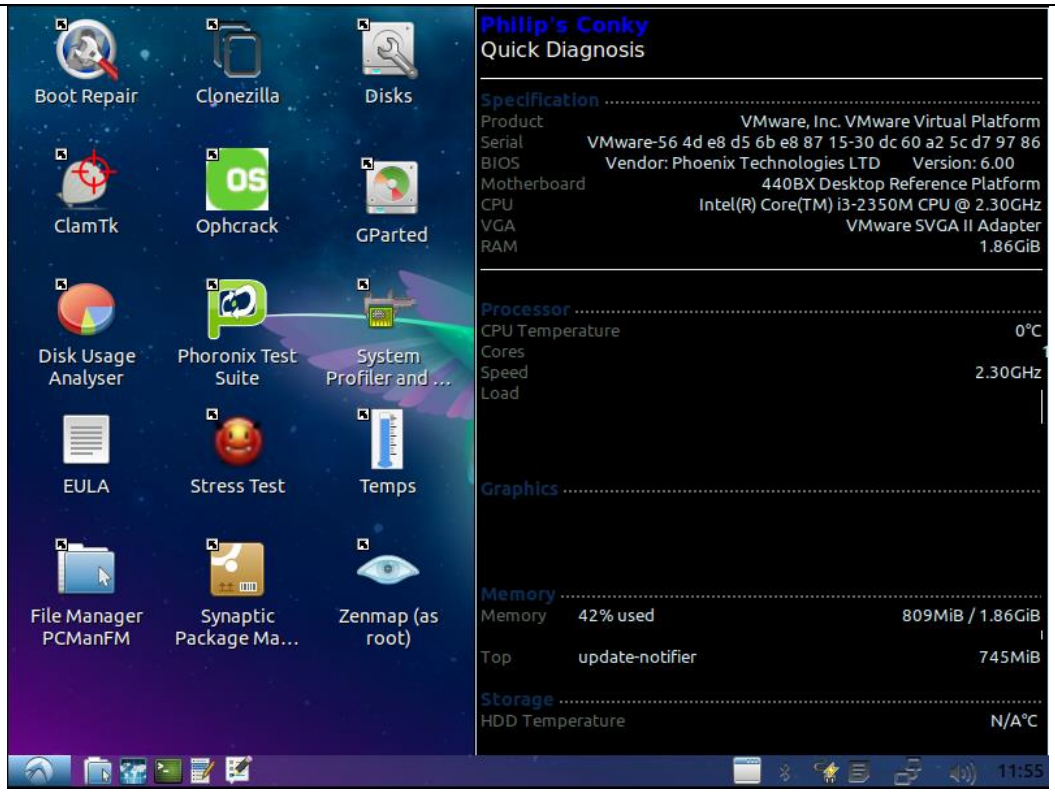
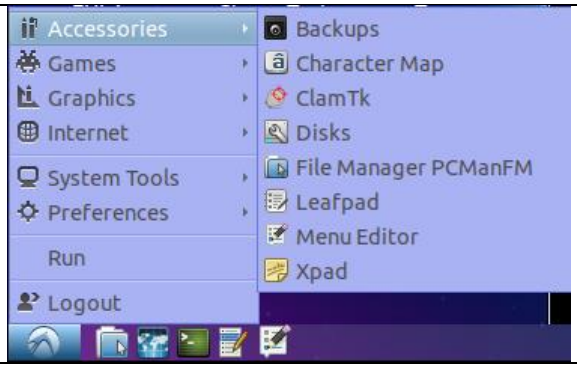
Screen Design

Details:

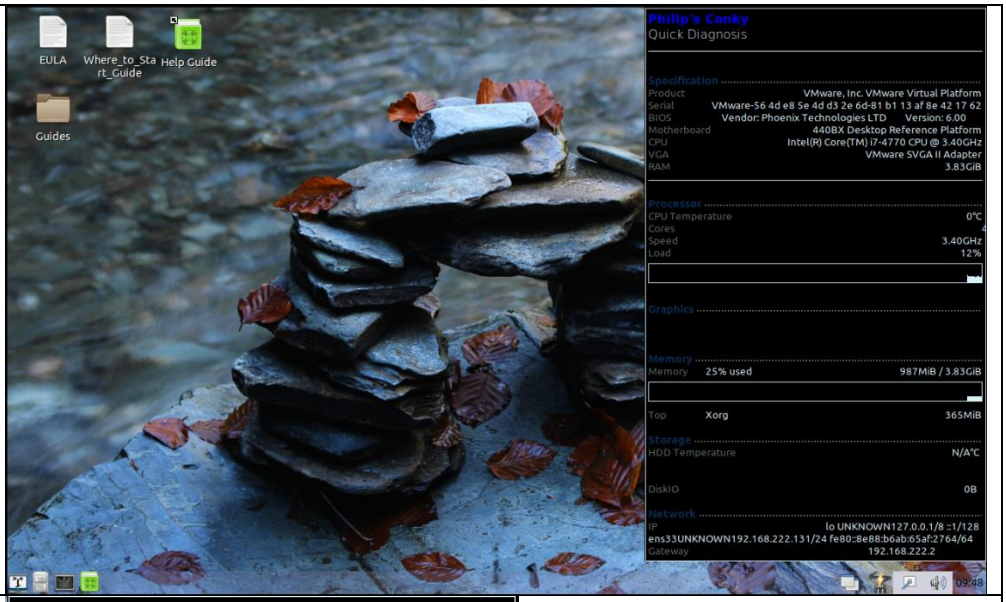
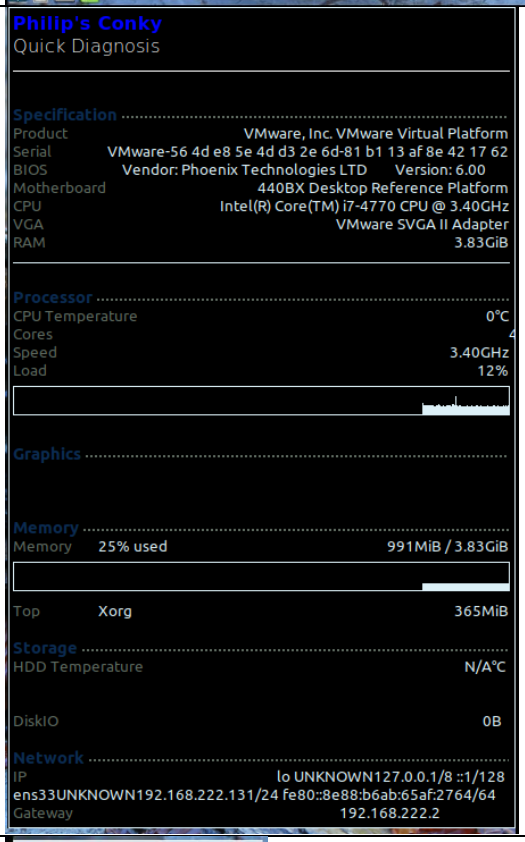
- Tools are instead accessed through two directories in the menu, for 'Basic' and 'Advanced' users.
- Icons sizes 33%, Task bar 35%.
- Mouse or keyboard navigation
- Display, icons and accessibility settings are changeable.

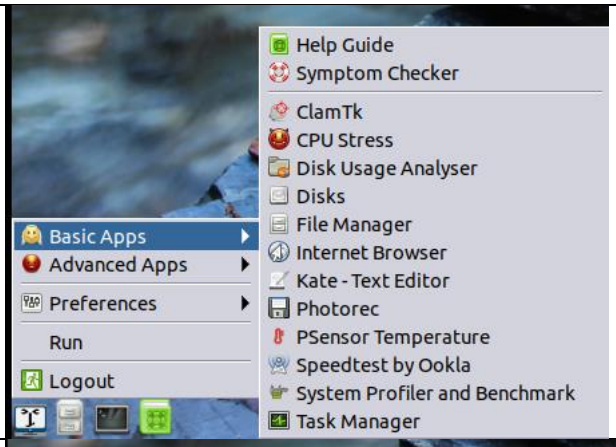
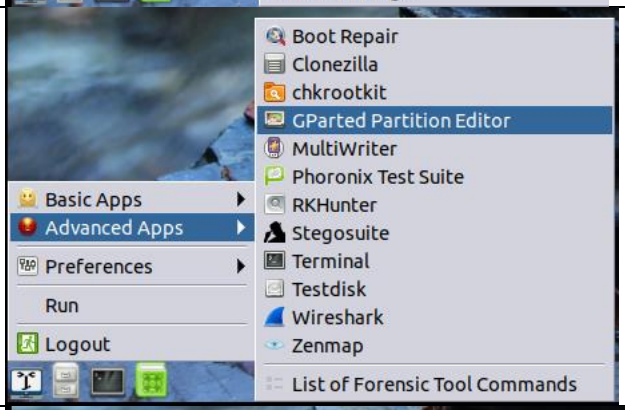
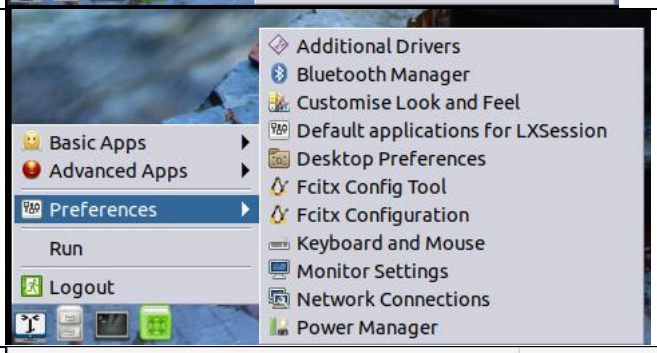
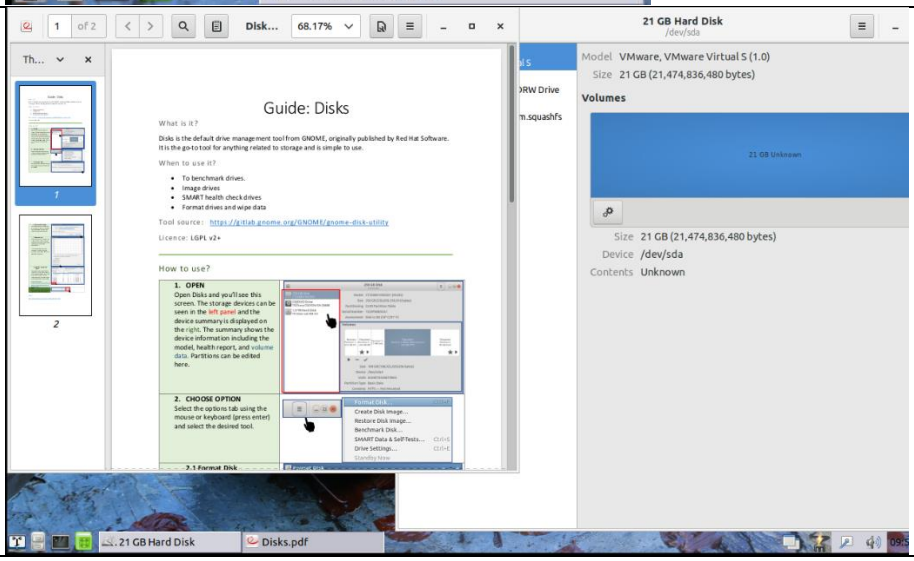
4. System Development

Desktop

Version 1 Screen captures	
Desktop	
Menu	

Version 2 Screen Captures

Desktop	
Conky	
Menu	

Basic Apps Menu	
Advanced Apps Menu	
Preferences Menu	
Disk tool guide opens with Disk tool	

Tool Guides

ClamTK

Guide: ClamTK

What is it?

ClamTK is a graphical user interface for ClamAV, a highly praised free and open source antivirus (AV) that scans a target file or folder for malware and removes them from the system. It is developed by Cisco Systems and is regularly updated.

When to use it?

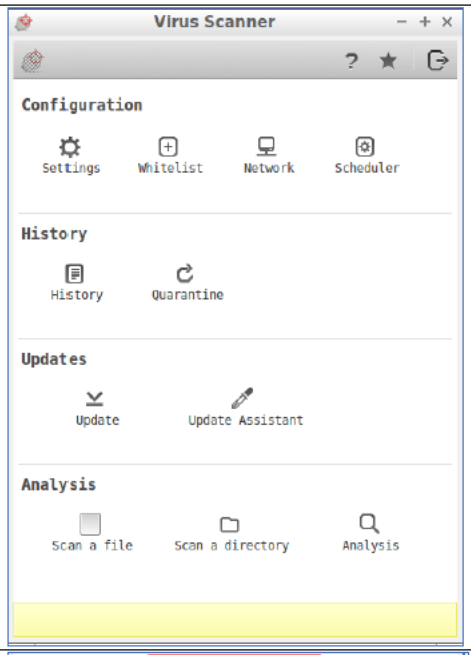
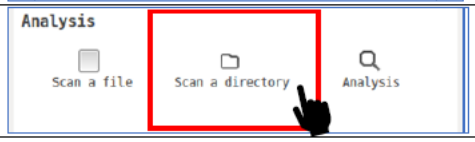
- If you suspect malware then this the first tool to run.
- Before you transfer data to other media.
- It is generally recommended as a regular maintenance tool, so it is worth using regardless of your issue.

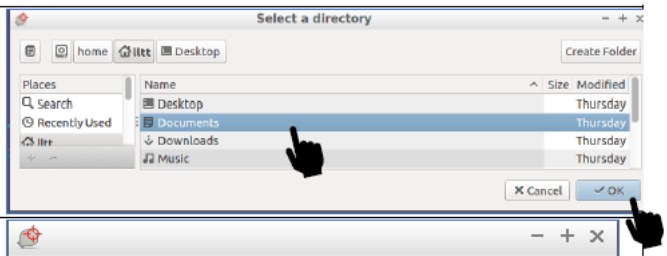
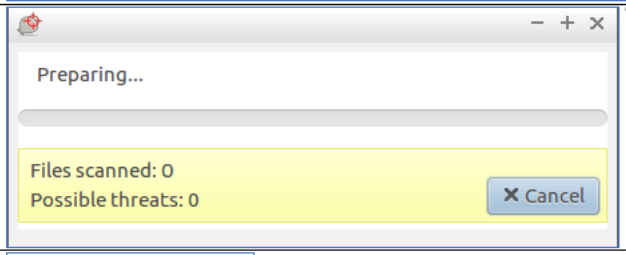
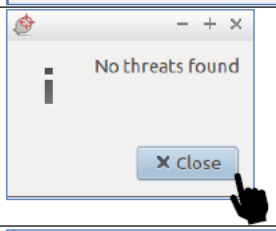
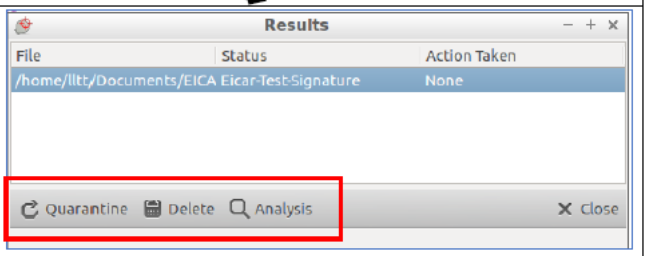
Tool source: ClamAV: www.clamav.net

ClamTK: https://github.com/dave-theunsub/clamtk

Licence: GNU General Public License

How to use?

1. OPEN Open ClamTK and you'll see this screen. If you're connected to the internet it may say that there will be updates available, which it should automatically install.	
2. CHOOSE OPTION To select an entire drive or folder, select (double left click) 'Scan a directory'	

3. CHOOSE DIRECTORY Highlight the drive or folder you wish to scan and select 'OK'	
4. SCAN This will initiate the scan so wait for the result. This may take some time if a large directory is chosen.	
5. RESULTS – 0 THREATS A results window will pop-up, hopefully with no threats detected. **Go to Step 7.**	
6. RESULTS - THREAT FOUND If a threat is found the file you can choose for Quarantine, Delete or Analysis. For most cases it is best to **Quarantine** or **Delete** the file. It worth **running the scan again** to make sure removal was successful.	
7. END You may wish to scan another directory or close the program.	

Extra

For a video tutorial see here: https://www.youtube.com/watch?v=yvvK1-7GcbM

Note: You'll need to have internet access to view it.

Credit: (Youtube (27 Jun 2015). 'Clamtk Gui for Clam Antivirus')

Disks

Guide: Disks

What is it?

Disks is the default drive management tool from GNOME, originally published by Red Hat Software. It is the go-to tool for anything related to storage and is simple to use.

When to use it?

- To benchmark drives.
- Image drives
- SMART health check drives
- Format drives and wipe data

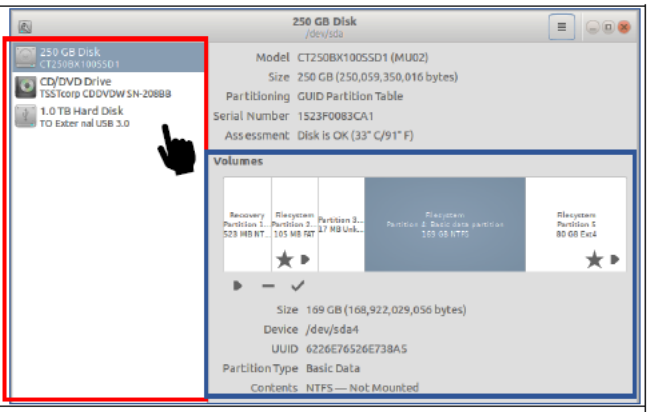
Tool source: https://gitlab.gnome.org/GNOME/gnome-disk-utility

Licence: LGPL v2+

How to use?

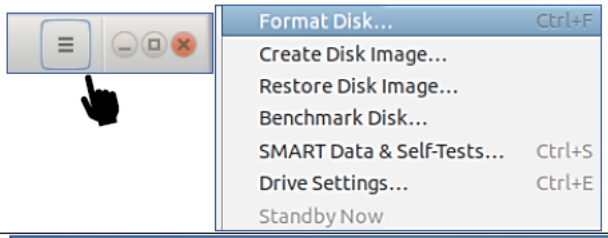
1. OPEN

Open Disks and you'll see this screen. The storage devices can be seen in the left panel and the device summary is displayed on the right. The summary shows the device information including the model, health report, and volume data. Partitions can be edited here.



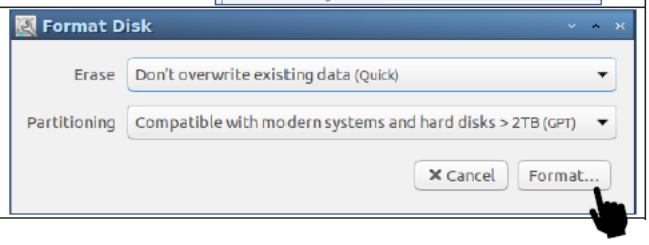
2. CHOOSE OPTION

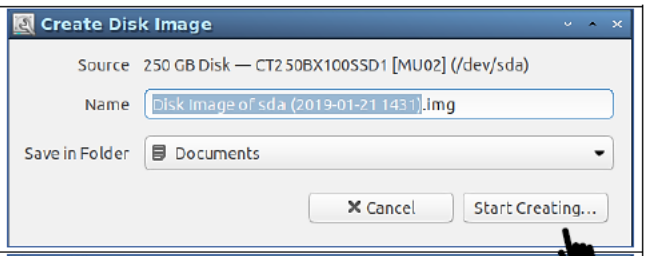
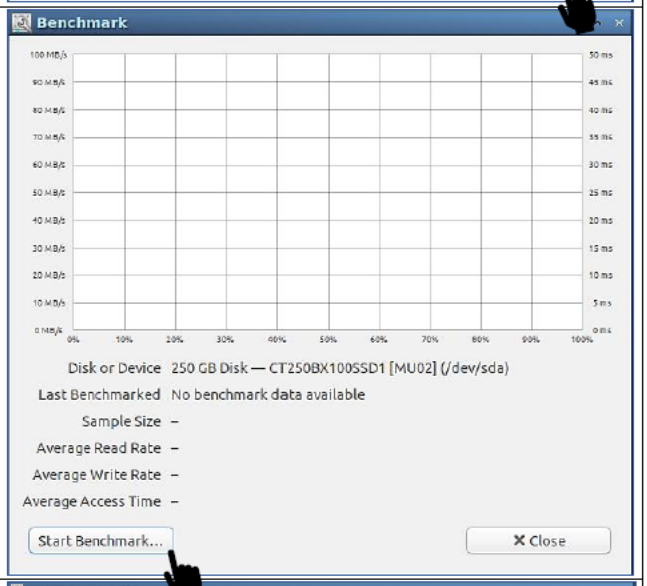
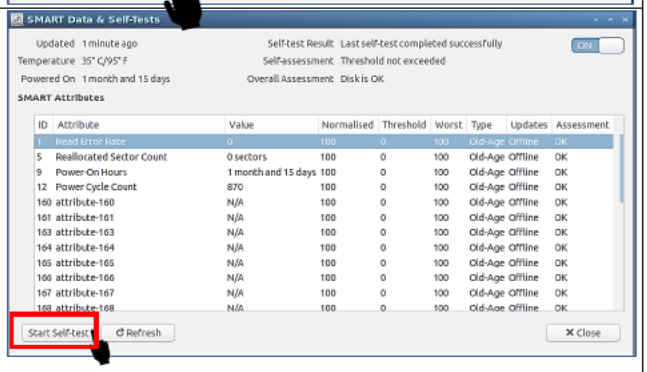
Select the options tab using the mouse or keyboard (press enter) and select the desired tool.



2.1 Format Disk

Here you can format disks to wipe data and/or prepare a disk for OS installation.



2.2 Create Disk Image This is where you can create an img image of a device or partition for drive backup. You can also restore images using this tool.	
2.3 Benchmark With the benchmark tool you can measure a disk's read and write speeds, as well as the access time to get a quick view of the performance. If it is performing lower than expected, it could be that the interface is slow (USB| SATA) or a sign the disk is aging.	
2.4 SMART Data & Self Tests These tests assess the health of the disk using multiple measures and is a good indicator of a failing drive. If you suspect a drive is failing then this is a useful tool to use before backing up your data. More information: https://www.backblaze.com/blog/what-smart-stats-indicate-hard-drive-failures/	
3 END You may wish to perform another operation or close the program.	

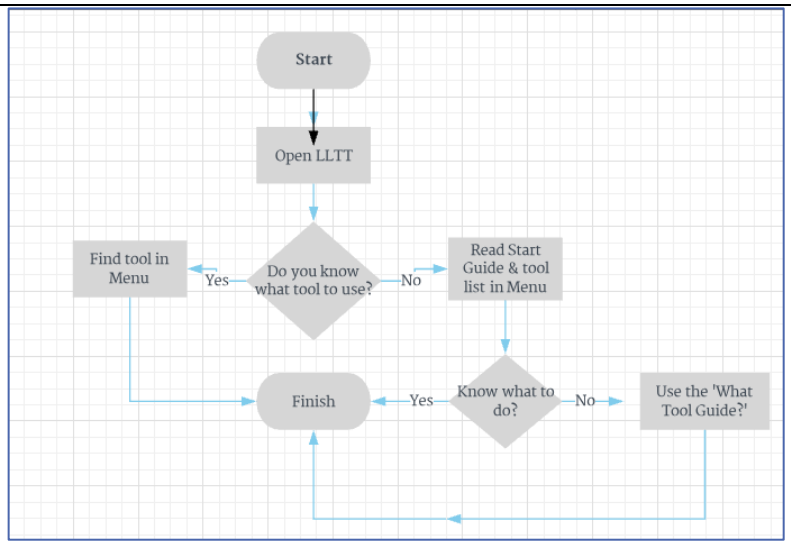
Extra

https://help.gnome.org/users/gnome-help/stable/disk.html

5. Tool Development

Help System

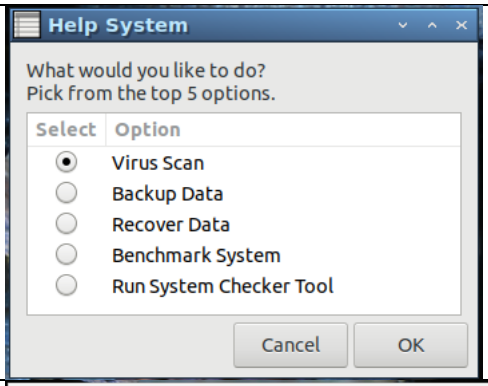
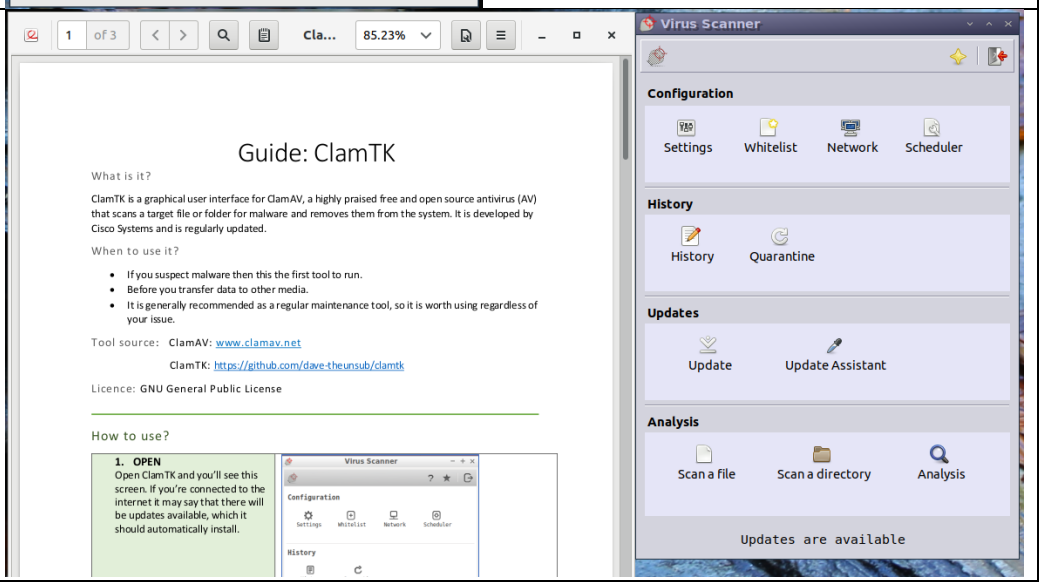
Flowchart



Script

```
#!/bin/bash
### Help System
set -x
ans1=$(zenity --list --title="Help System" --text="What would you like to do? \nPick from the top 5 options." \
--radiolist --column="Select" --column="Option" --height=250 --width=350 \
TRUE "Virus Scan" \
FALSE "Backup Data" \
FALSE "Recover Data" \
FALSE "Benchmark System" \
FALSE "Run System Checker Tool")

if [ "$ans1" == "Virus Scan" ]
    then
        ans1="1"

elif [ "$ans1" == "Backup Data" ]
    then
        ans1="2"

elif [ "$ans1" == "Recover Data" ]
    then
        ans1="3"

elif [ "$ans1" == "Benchmark System" ]
    then
        ans1="4"

elif [ "$ans1" == "Run System Checker Tool" ]
    then
        ans1="5"
else
    zenity --warning --title="ERROR" --text="There's been an error... Closing the program!" --height=250 --width=350; sleep 1; exit
fi

case $ans1 in
    1)
        clamtk %F &  evince ~/Desktop/Guides/ClamTK.pdf


    ;; ###end

    2)
        pcmanfm
    ;;

    3)
        lxterminal --command=photorec
    ;;

    4)
        lxterminal --command='phoronix-test-suite interactive'
    ;;

    5)
    ./.symptom_checker
    ;;


    *)
        zenity --warning --text="ERROR!" --height=250 --width=350;
        sleep 1;
    zenity --warning --text="CLOSING!" --height=250 --width=350; exit

    ;;
esac
```

Output

Window 1	
Window 1.1	

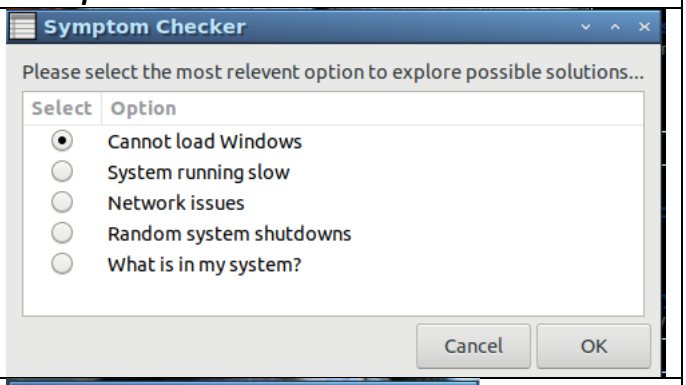
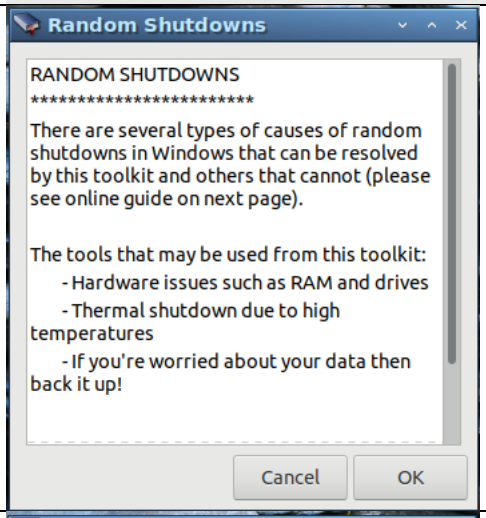
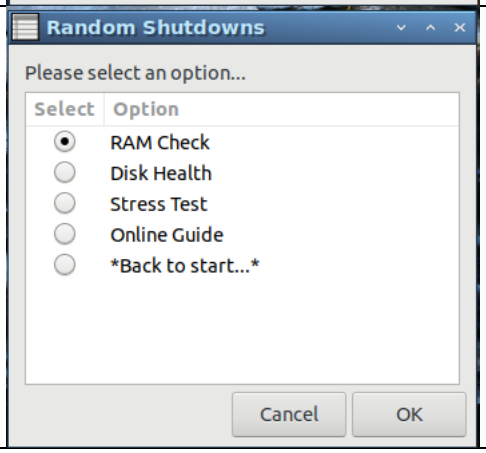
Symptom Checker

Script

```
#!/bin/bash
### Symptom Checker System
set -x
ans=$(zenity --list --title="Symptom Checker" --text="Please select the most relevent option to explore possible solutions..." \
--radiolist --column="Select" --column="Option" --height=250 --width=350 \
TRUE "Cannot load Windows" \
FALSE "System running slow" \
FALSE "Network issues" \
FALSE "Random system shutdowns" \
FALSE "What is in my system?")

if [ "$ans" == "Cannot load Windows" ]
    then
        ans="1"

elif [ "$ans" == "System running slow" ]
    then
        ans="2"

elif [ "$ans" == "Network issues" ]
    then
        ans="3"

elif [ "$ans" == "Random system shutdowns" ]
    then
        ans="4"

elif [ "$ans" == "What is in my system?" ]
    then
        ans="5"
else
    zenity --warning --title="ERROR" --text="There's been an error... Closing the program!" --height=200 --width=350; sleep 1; exit
fi

case $ans in
    1)
        sudo cat ~/Documents/Windows_Boot_Issues | zenity --text-info --title="Windows Boot Issues" --height=450 --width=350
    ans2=$(zenity --list --title="Windows Boot Issues" --text="Please select an option..." \
        --radiolist --column="Select" --column="Option" --height=250 --width=350 \
    TRUE "Run Boot Repair" \
    FALSE "Windows Installation USB Guide" \
    FALSE "Backup data" \
    FALSE "*Back to start...*")

    if [ "$ans2" == "Run Boot Repair" ]
        then
            sudo boot-repair-pkexec
    elif [ "$ans2" == "Windows Installation USB Guide" ]
        then
            xdg-open 'https://www.neosmart.net/wiki/windows-error-recovery-failed-start/'
    elif [ "$ans2" == "Backup data" ]
        then
            pcmanfm
    else
        ./.symptom_checker
    fi


    ;; ###end

    2)
        sudo cat ~/Documents/System_slow | zenity --text-info --title="System Running Slow" --height=350 --width=350
    ans2=$(zenity --list --title="System Running Slow" --text="Please select an option..." \
        --radiolist --column="Select" --column="Option" --height=300 --width=350 \
    TRUE "System Speed Up Guide" \
    FALSE "Temperature Check" \
    FALSE "Virus Scan" \
    FALSE "Drive Health Check" \
    FALSE "*Back to start...*")

    if [ "$ans2" == "System Speed Up Guide" ]
        then
            xdg-open 'https://support.microsoft.com/en-gb/help/15055/windows-7-optimize-windows-better-performance/'
    elif [ "$ans2" == "Temperature Check" ]
        then
            lxterminal --command=./.stress_script
    elif [ "$ans2" == "Virus Scan" ]
        then
            clamtk %F & evince ~/Desktop/Guides/ClamTK.pdf

    elif [ "$ans2" == "Drive Health Check" ]
        then
            gnome-disks & evince ~/Desktop/Guides/Disks.pdf

    else
```

```
else
    ./.symptom_checker
fi
;;

3)
    sudo cat ~/Documents/Network_issues | zenity --text-info --title="Network Issues" --height=350 --width=350
ans2=$(zenity --list --title="Network Issues" --text="Please select an option..." \
    --radiolist --column="Select" --column="Option" --height=300 --width=350 \
TRUE "Internet Speed Test" \
FALSE "Wireshark" \
FALSE "Zenmap" \
FALSE "Online Guide to Networks" \
FALSE "*Back to start...*")

if [ "$ans2" == "Internet Speed Test" ]
    then
        xdg-open 'https://speedtest.net/'
elif [ "$ans2" == "Wireshark" ]
    then
        sudo wireshark
elif [ "$ans2" == "Zenmap" ]
    then
        su-to-root -X -c zenmap

elif [ "$ans2" == "Online Guide to Networks" ]
    then
        xdg-open 'https://www.drivethelife.com/windows-10/fix-windows-10-network-connection-issues.html'

else
    ./.symptom_checker
fi

;;

4)
    sudo cat ~/Documents/Random_shutdown | zenity --text-info --title="Random Shutdowns" --height=350 --width=350
ans2=$(zenity --list --title="Random Shutdowns" --text="Please select an option..." \
    --radiolist --column="Select" --column="Option" --height=300 --width=350 \
TRUE "RAM Check" \
FALSE "Disk Health" \
FALSE "Stress Test" \
FALSE "Online Guide" \
FALSE "*Back to start...*")

if [ "$ans2" == "RAM Check" ]
    then
        zenity --warning --title="RAM Check" --text="The RAM check is available on the LLTT load menu - please RESTART."
        exit
elif [ "$ans2" == "Disk Health" ]
    then
        gnome-disks & evince ~/Desktop/Guides/Disks.pdf
elif [ "$ans2" == "Stress Test" ]
    then
        ./.stress_script

elif [ "$ans2" == "Online Guide to Networks" ]
    then
        xdg-open 'https://www.windowsreport.com/random-shutdown-windows-10/'

else
    ./.symptom_checker
fi

;;

5)
sudo cat ~/Documents/System_info | zenity --text-info --title="System Information" --height=350 --width=350
ans2=$(zenity --list --title="System Information" --text="Please select an option..." \
    --radiolist --column="Select" --column="Option" --height=300 --width=350 \
TRUE "Hardware Summary" \
FALSE "Disk Information" \
FALSE "*Back to start...*")

if [ "$ans2" == "Hardware Summary" ]
    then
        /usr/bin/hardinfo
elif [ "$ans2" == "Disk Information" ]
    then
        gnome-disks & evince ~/Desktop/Guides/Disks.pdf
else
    ./.symptom_checker
fi

;;
```

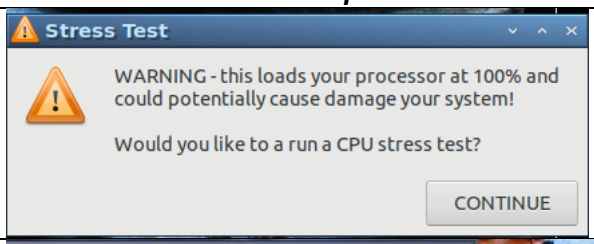
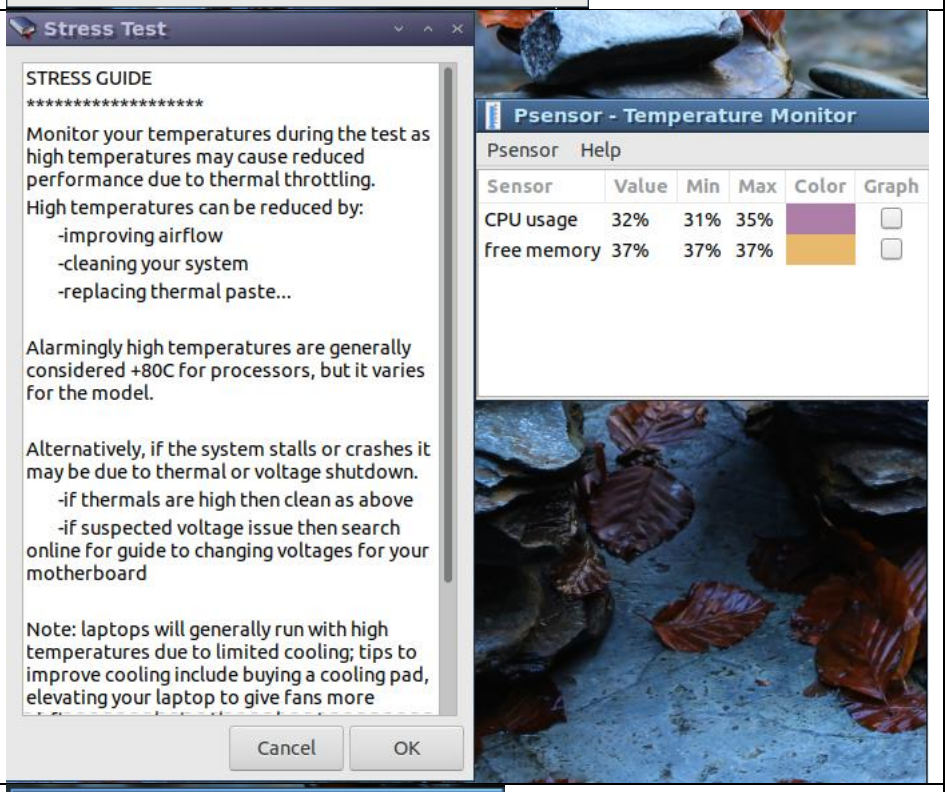
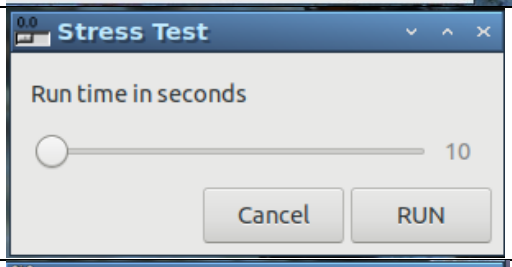
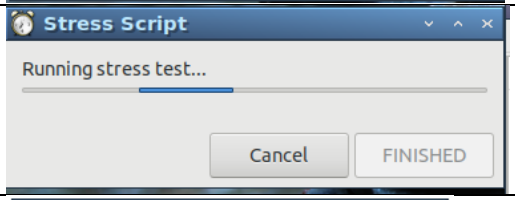
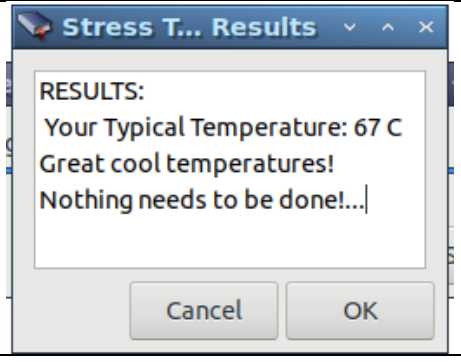
```
   *)
       zenity --warning --text="ERROR!";
       sleep 1;
   zenity --warning --text="CLOSING!" ; exit

   ;;
esac
```

Output	
Window 1	
Window 4.1 Information Page	
Window 4.2 List of options	

Stress Script

Script

```
#!/bin/bash
### Creating a script to use the command Stress for stress testing the CPU and opens temps
###Find no. of CPU cores
###set -x
CC=$(grep 'siblings /proc/cpuinfo | uniq | awk '{print $3}')

###Start Window
zenity --warning --title="Stress Test" --text="WARNING - this loads your processor at 100% and could potentially cause damage your system! \n\nWould you like to a run a CPU stress test?" --width=350 --ok-label="CONTINUE"

###Start Temperature Tool Psensor
psensor &

sudo cat ~/Documents/Stress | zenity --text-info --title="Stress Test" --height=550 --width=350 --ok-label="OK"

ans=$(zenity --scale --title="Stress Test" --text="Run time in seconds" --min-value=10 --max-value=600 --value=10 --step=5 --height=350 --width=350 --ok-label="RUN")

###Start test and Progress Bar
x=1
while [ $x -lt $ans ]; do echo $x & sleep 1; let x=x+1
done | zenity --progress --title="Stress Script" --text="Running stress test..." --percentage=$x --width=350 --pulsate --ok-label="FINISHED" & stress --cpu $CC --timeout $ans

###Display Results
sensors > ~/temp.txt
grep '^Package id 0: ' ~/temp.txt | uniq | awk '{print $4}' > ~/Documents/temp2.txt
ans2=$(cat ~/Documents/temp2.txt | cut -c 2-3)
if [ $ans2 -lt 60 ]
    then
        printf "RESULTS: \nYour Typical Temperature: $ans2 C\nGreat cool temperatures! Nothing needs to be done!..." > ~/Documents/temp2.txt
    elif [ $ans2 -nt 75 ]
    then
        printf "RESULTS: \nYour Typical Temperature: $ans2 C\nThings are running a little hot! Try to reduce temps and run again!" > ~/Documents/temp2.txt
    else
        printf "RESULTS: \nYour Typical Temperature: $ans2 C\nOK temperature!..." > ~/Documents/temp2.txt
fi
cat ~/Documents/temp2.txt | zenity --text-info --title="Stress Test Results" --height=175 --width=250
exit
```

	Output
Window 1 Warning	
Window 2 Information guide & temperature monitor (screencapture does not have temperature sensor due to running as VM)	
Window 3 Run time selection	
Window 4 During operation	
Window 5 Results	

Conky Script

Script

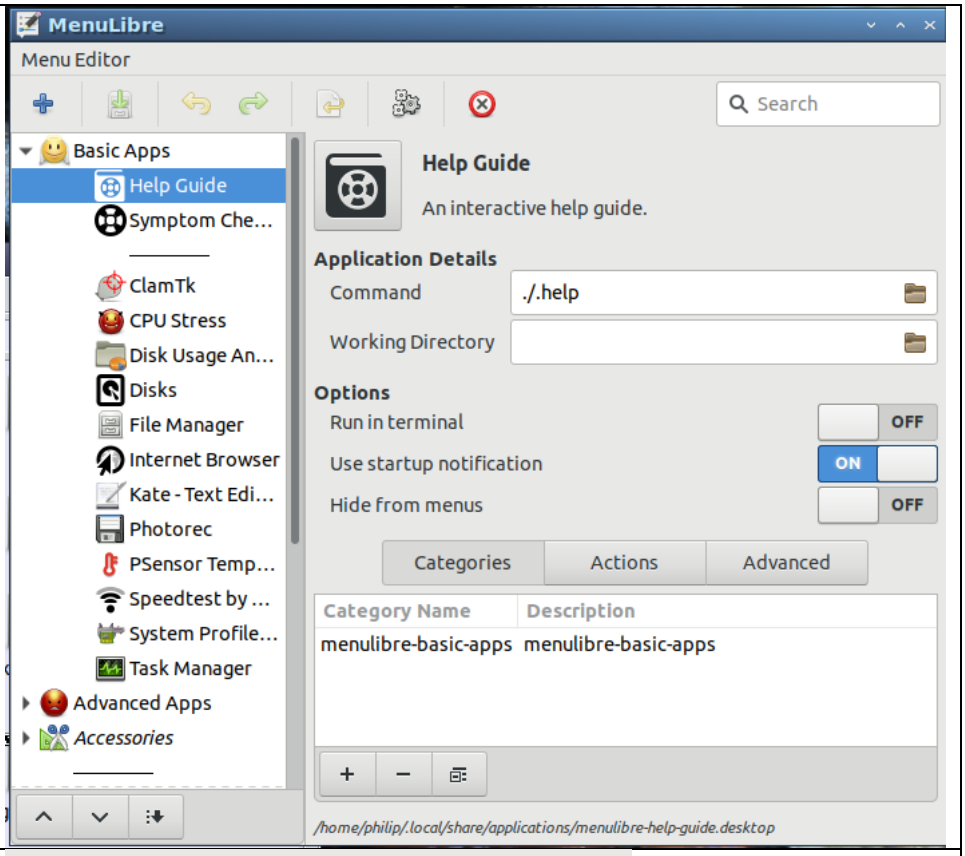
```
##############################################################
#Conky by Philip Robinson
#Inspired by Paul Vreeland @ https://paul.is-a-geek.org/aio-srt/
#& Andreas Ghohr @ https://www.splitbrain.org/blog/2016-11/20-simple_conky_setup
##############################################################

#CONFIGURATION
background yes
use_xft yes
xftalpha 0.8
update_interval 0.75
own_window yes
own_window_type normal
own_window_class conky
own_window_transparent no ##transparency (yes|no)
own_window_argb_visual yes
own_window_argb_value 50  # semi-transparent
own_window_hints undecorated,below,sticky,skip_taskbar,skip_pager
double_buffer yes
no_buffers yes
uppercase no
cpu_avg_samples 1
override_utf8_locale
format_human_readable

#DRAWING
draw_shades no
draw_outline no
draw_borders yes
draw_graph_borders yes
default_bar_width = 150, default_bar_height 5
default_graph_width = 150, default_graph_height 12
default_gauge_width = 20, default_gauge_height 20

#FONT
font Ubuntu:size=10
default_color DCF0F7 #pale blue
color1 DCF0F7
color2 0C2C52 #dark blue
color3 5F6B61 #light blue
color4 blue
color5 white
default_outline_color black
outline_color1 white

#LOCATION
alignment top_right
gap_x 5
gap_y 5
minimum_size 250 5
maximum_width 500

#TEMPLATES
template0 ${font Ubuntu:bold:size=11}${color2}\1 ${color3}${stippled_hr 2}${font}
template1 ${color3}\1
template2 ${goto 80}${color}
template3 ${color}${alignr}

TEXT
${color4}${font :weight=bold}Philip's Conky
${color5}${font :weight=}Quick Diagnosis
${hr}

${template0 Specification}
${template1 Product}                ${template3}${exec sudo dmidecode -s system-manufacturer} ${exec sudo dmidecode -s system-product-name| sed 's/  //g'}
${template1 Serial}                 ${template3}${exec sudo dmidecode -s system-serial-number}
${template1 BIOS}                       ${template2}${exec sudo dmidecode -t bios | egrep "Vendor"} ${exec sudo dmidecode -t bios | egrep "Version: "}
${template1 Motherboard}                ${template3}${exec sudo dmidecode -s baseboard-product-name}
${template1 CPU}                    ${template3}${exec cat /proc/cpuinfo | grep 'model name' | sed -e 's/model name.*: //'| uniq}
${template1 VGA}                        ${template3} ${exec lspci | grep VGA | cut -d ':' -f3}
${template1 RAM}                    ${template3}${memmax}
${hr}

${template0 Processor}
${template1 CPU\ Temperature}                       ${template3}${acpitemp}°C
${template1 Cores}                  ${template3}${exec cat /proc/cpuinfo | grep 'processor' | wc -l}
${template1 Speed}                                      ${template3}${freq_g}GHz
${template1 Load}                               ${template3}${cpu cpu1}%
${color1}${cpugraph cpu0}

${template0 Graphics Card}
${template Temperature}             ${template3} ${exec sudo nvidia-smi}
                                    ${template3} ${exec sudo intel_gpu_tools}
                                    ${template3} ${exec sudo aticonfig --odgc --odgt}
${template0 Memory}
${template1 Memory}${template2}${memperc}% used  ${template3}${mem} / ${memmax}
${memgraph}
${template1 Top}${template2}${top_mem name 1}           ${template3}${top_mem mem_vsize 1}

${template0 Storage}
${template1 HDD\ Temperature}                       ${template3}${hddtemp}°C
                                    ${template3}${exec cat /dev/sda}

${template1 DiskIO}                 ${template3}${diskio}

${template0 Network}
${template1 IP}                     ${template3}${execi 5 ip -br addr | sed 's/  //g'}
${template1 Gateway}        ${template3}${gw_ip}
```

Output

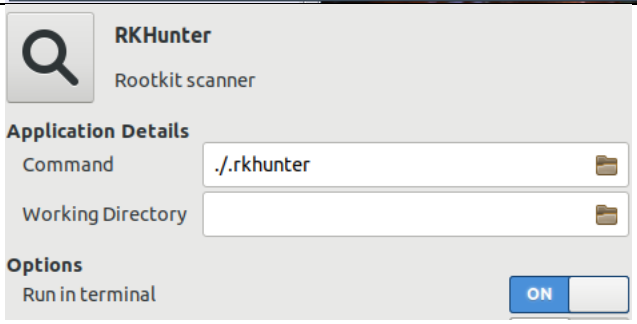
Please see 4. System Development > Version 2 Screen captures

Menu Development

MenuLibre
In MenuLibre the Basic Apps and Advanced Apps directories were created. In these the application launchers were created using the console commands to run them.



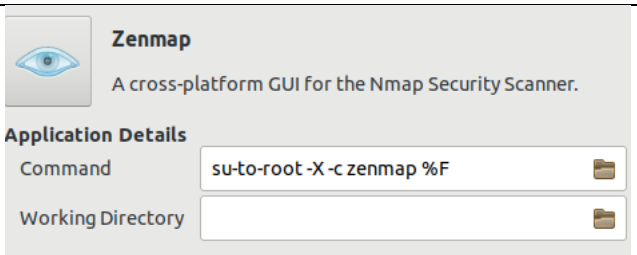
Some commands required using custom executable scripts to run them with operators, such as RKHunter.



The simple script runs the command with the required operators.

```
#!/bin/bash
sudo rkhunter --check
kate /var/log/rkhunter.log
```

Other applications could be adequately initiated using just the MenuLibre command input.



LAB REPORT: PINGUY

Introduction

Testing methods for converting the current system into a live ISO that can be used for installation as a bootable USB.

Methodology

Download updates in Ubuntu:

```
sudo apt-get update
```

Download from Source Forge:

```
Wget http://freefr.dl.sourceforge.net/project/pinguy-
os/ISO_Builder/pinguybuilder_5.1-8_all.deb
```

Install package:

```
sudo dpkg -i pinguybuilder_5.1-8_all.deb && sudo apt-get install -f
```

Run Pinguy and test options to produce live ISO

Results

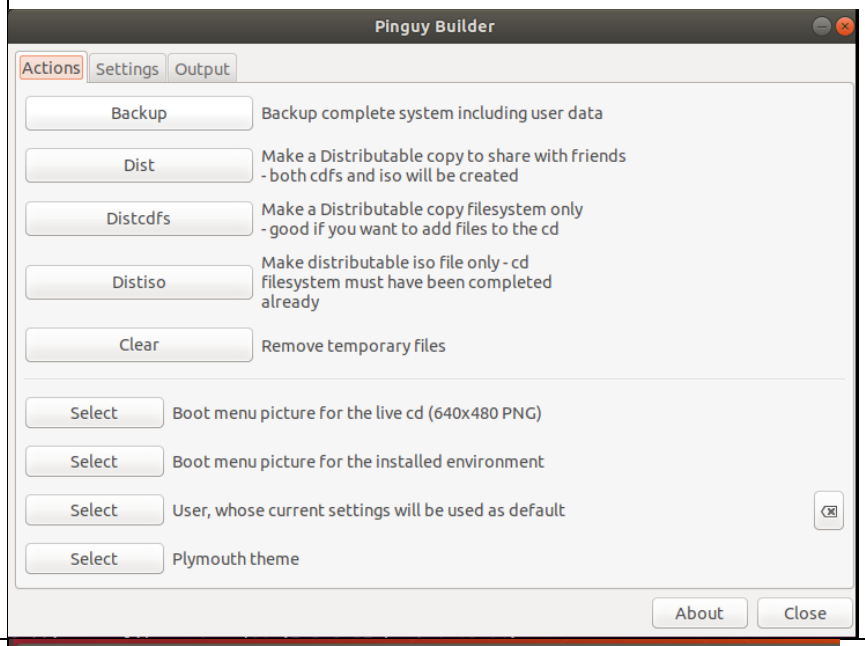
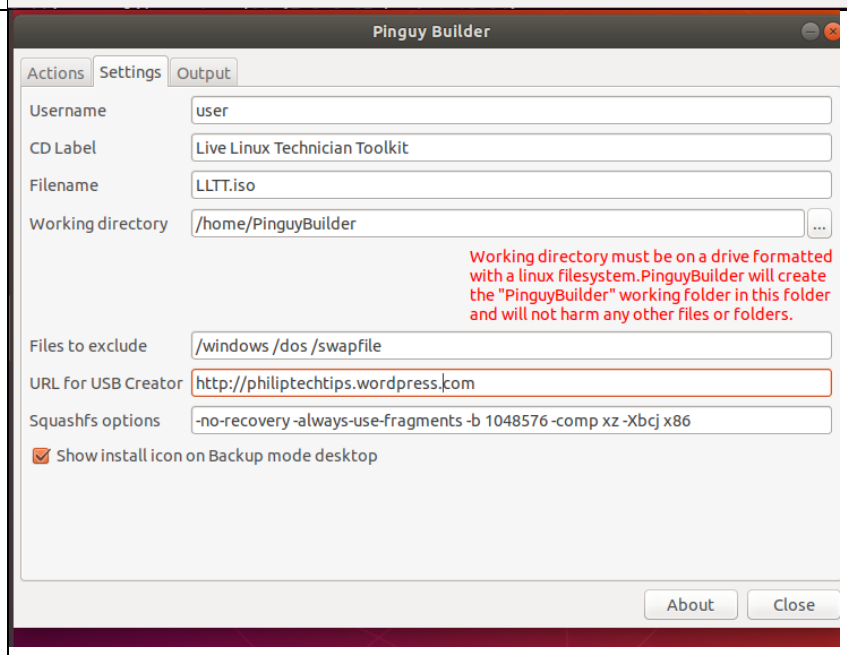
Using the backup option, the user profile to copy can be selected to use. – seems like the best option, as the others create a new user with a different desktop but still access to programs.

Therefore, the it is important to ensure the password is not needed, or if that cannot be performed, a simple password for the end user is needed. This could be used as a level of authentication for end users.

To do

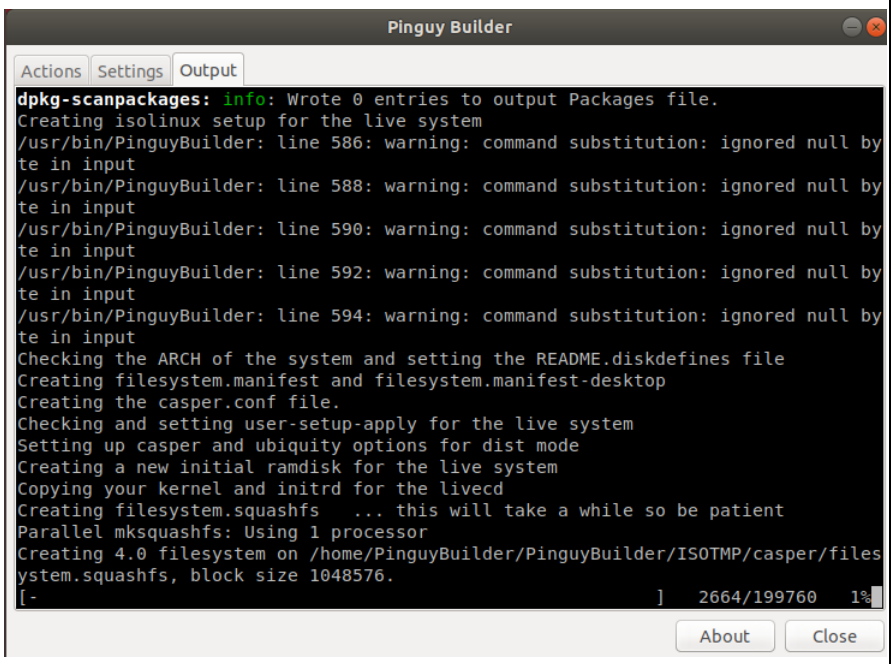
Develop a new Lubuntu system as a virtual machine, with plain naming system (eg. Computer and user name as user/tech). Before, too much time spent on developing, test with Pinguy sudo to see if it can develop a live ISO from a VM.

Screenshots

Pinguy Home Page The home page shows the options available for creation depending on the circumstance.	
Settings In settings, the user can customise the configuration of the generated ISO, such as the user name, ISO title and the files to include.	

Output

The output tab shows the code generated by the program to produce the ISO. It can be seen to use the commands for Casper and LiveCD that are system standards for ISO creation. Once finished, the ISO file can be found in the designated folder and then it can be burned to a USB or CD, or used in a VM.



6. Testing

Solution Testing

Solution Testing

Testing plan for creating quantitative results that can be used for comparison and evaluation.

Time Based Efficiency

The measure of how much time it takes for a user to locate a desired application.

Methodology

1. The tester (*t*) is told to close their eyes
2. They are told the application or tool type to search for (e.g. find an antivirus tool)
3. The system is opened on the desktop
4. When told the *t* opens their eyes and the timer is started
5. Time stops when the user has the mouse over a correct selection
6. Time is recorded
7. *Repeat for other systems*

Results

Time (seconds)	*Systems*				
	AIO SRT	HB CD PE	UB CD	KNOPPIX	LLTT
T1	0.78	12.30	12.50	8.13	4.56
T2	3.56	9.30	16.54	93.52	19.20
T3	4.79	15.99	14.32	20.11	8.46
T4	33.23	6.98	22.26	46.33	7.35
T5	8.81	36.16	13.36	89.39	52.67
T6	9.49	60.59	24.10	23.56	20.66
AVERAGE	**10.11**	**23.54**	**17.18**	**46.84**	**18.82**
RANK	**1**	**4**	**2**	**5**	**3**

Summary

- AIO SRT had the fastest user response time due to the applications being accessible via shortcuts on the desktop, making them immediately visible when booted. This design style was considered for the artefacts development, however due to the amount of applications the result was cluttered (see Design).
- Ultimate Boot CD came second due to its simplistic design; however, the categories were ambiguous they were quick to navigate.
- Hirens Boot CD PE had the second fasted results when users took advantage of the *search* tool to quickly locate applications. As part of the future artefact development a menu search tool will be include as it is an available feature in Ubuntu.
- The artefact, *LLTT,* came third overall which is a credit to the menu system. When testers utilised the *Help System* tool the time was faster. However, a search tool could be implemented in the menu.
- Knoppix came last due to the sheer amount of applications available in the OS, however as Knoppix is a full desktop not a repair kit, this is not surprising.

Public Testing Questionnaire

Q1.

1. Rate the usability of the toolkit as a whole.(How easy it is to use)

		Response Percent	Response Total
1	1 (low)	0.00%	0
2	2	0.00%	0
3	3	0.00%	0
4	4	33.33%	3
5	5 (high)	66.67%	6

Analysis							
	Mean:	4.67	Std. Deviation:	0.47	Satisfaction Rate:	91.67	answered 9
	Variance:	0.22	Std. Error:	0.16			skipped 0

Comments: (7)

1	easy to understand and use
2	Good help system
3	i like the guides
4	nice layout
5	good guides and help tool
6	very good
7	Good for beginners with guides.

Q2.

2. Rate the functionality of the toolkit as a whole.(How suited to being a diagnostic, repair and benchmarking tool)

		Response Percent	Response Total
1	1 (low)	0.00%	0
2	2	0.00%	0
3	3	0.00%	0
4	4	22.22%	2
5	5 (high)	77.78%	7

Analysis							
	Mean:	4.78	Std. Deviation:	0.42	Satisfaction Rate:	94.44	answered 9
	Variance:	0.17	Std. Error:	0.14			skipped 0

Comments: (8)

1	Good range of tools
2	Well-rounded
3	Every tool I'd need
4	suits most needs
5	good range
6	lots of tools
7	nice list of apps
8	Haven't tried anyothers but seems comprehensive.

Q3.

3. Rate the design of the toolkit.(Colours, fonts, windows, panels etc)

		Response Percent	Response Total
1	1 (low)	0.00%	0
2	2	0.00%	0
3	3	0.00%	0
4	4	44.44%	4
5	5 (high)	55.56%	5

Analysis							
	Mean:	4.56	Std. Deviation:	0.5	Satisfaction Rate:	88.89	answered 9
	Variance:	0.25	Std. Error:	0.17			skipped 0

Comments: (7)

1	easy to read
2	Some text unclear with background
3	bigger icons would be better
4	similar to other Linux
5	clean layout
6	clear
7	Everything looks good.

Q4.

4. Which were your favourite tools and why?

		Response Percent	Response Total
1	Open-Ended Question	100.00%	9

1	the quick diagnosis kit
2	disk cleaner and benchmark tester
3	Top five boxes - due to level of knowledge required for a less advanced user
4	System profiler as easy to use
5	Multiple anti-malware tools
6	Help System was a good idea
7	Antivirus
8	symptom checker is a good idea for less skilled people
9	The Help System and Symptom Checker as they are good for beginners.

Q5.

5. Which were your least favourite tools and why?

		Response Percent	Response Total
1	Open-Ended Question	100.00%	6

1	Conky as couldn't close
2	Command line based tools like Clone
3	Wireshark wouldn't work
4	N/A
5	don't like the terminal colours
6	Conky window as it is distracting

answered	6
skipped	3

Q6.

6. BUG REPORT: Did you experience any problems during the test?

		Response Percent	Response Total
1	Open-Ended Question	100.00%	7
	1 no		
	2 Closing the help guide caused an unknown error		
	3 Help system had an error when closed		
	4 conky		
	5 Wireshark can't work		
	6 conky doesn't fit		
	7 N/A		
		answered	7
		skipped	2

Q7.

7. What changes/ improvements would you make?eg. to the design or tools included

		Response Percent	Response Total
1	Open-Ended Question	100.00%	9
	1 Wireshark not accessible		
	2 Use different forensic tools that have been certified to use and think about using Recuva for the data recovery		
	3 Would give conky a close option		
	4 Fix conky		
	5 more tools and updated		
	6 Update to new Lubuntu		
	7 more forensics/security tools		
	8 OPTION TO REMOVE CONKY		
	9 Maybe have videos to show what to do?		
		answered	9
		skipped	0

Q8.

8. How likely would you reuse LLTT?

		Response Percent	Response Total
1	1 (Low)	0.00%	0
2	2	0.00%	0
3	3	11.11%	1
4	4	33.33%	3
5	5 (High)	55.56%	5

Analysis							
	Mean:	4.44	Std. Deviation:	0.68	Satisfaction Rate:	86.11	answered 9
	Variance:	0.47	Std. Error:	0.23			skipped 0

Comments: (6)

1	not a linux person myself but it seemed pretty intuitive
2	definatly
3	If needed
4	hopefully wouldn't need to
5	would like to try again
6	If I knew what to do!

Q9.

9. How likely would you be to recommend LLTT?

		Response Percent	Response Total
1	1 (Low)	0.00%	0
2	2	0.00%	0
3	3	0.00%	0
4	4	0.00%	0
5	5 (High)	100.00%	9

Analysis							
	Mean:	5	Std. Deviation:	0	Satisfaction Rate:	100	answered 9
	Variance:	0	Std. Error:	0			skipped 0

Comments: (4)

1	fun little system
2	I liked it
3	only to friends with decent IT skills
4	I don't have any friends...

Q10.

10. Finally, what one word would you use to describe LLTT?

		Response Percent	Response Total
1	Open-Ended Question	100.00%	9
	1 technical		
	2 sweet		
	3 Complete		
	4 cool		
	5 Good		
	6 clever		
	7 nice		
	8 well-done		
	9 Cool!		
		answered	9
		skipped	0

LLTT
Live Linux Technician Toolkit

Philip Robinson
ID: 10572891
Supervisor: Clint Washington

IN PARTNERSHIP WITH PLYMOUTH UNIVERSITY

List of applications

- ✓ Data recovery
- ✓ Virus removal
- ✓ Stress test
- ✓ Partition editor
- ✓ Forensic tools
- ✓ Boot repair
- ✓ Disk cloner
- ✓ System profiler
- ✓ Benchmarks
- ✓ Network tools

Problem

Since the adoption of home desktop computers their availability and affordability has increased dramatically resulting in 88% of people in the UK owning at least one home computer (Statista, 2019). This means that there are over 58 million people with computers that are probably going to experience a hardware or software fault at some point in their life which will need to be resolved. Many common computer hardware and software issues can be diagnosed and potentially resolved by an average home user with minimal IT skills; by using the huge range of free software available online, thus saving them time and money.

However, there are issues surrounding a lack of knowledge and confidence for sourcing and using appropriate and reputable tools, as well as difficulties diagnosing common computer issues and how to best resolve them. Based on current trends and statistics, plus experience working as an IT technician, the study investigates the problem, evaluates current solutions, and presents a proposed artefact as a better alternative. The project links to current trends including Right to Repair, e-waste reduction, and open source software movements.

Current Solutions

There are current and well-founded live CD repair solutions like Knoppix, Ultimate Boot CD & Hiren's Boot CD; however, these tend to be outdated, command line focused, lack local support, and are designed for functionality rather than usability. As part of the study research these solutions were tested and evaluated, the results inspiring the content and features of the developed artefact.

Proposed Solution

This project is focusses upon creating a comprehensive and convenient toolkit for PC diagnostics, repair and benchmarking that is accessible to all.

- **Featuring popular industry tools**
- **Custom scripts and adjustable environment**
- **Diagnostic wizard & guides for tools**
- **OS independent & USB bootable**

Development

LLTT is based on open source methodology as the purpose of the artefact is to be a collection of third-party tools and thus legally it needs to use freely alterable and distributable software. The foundation of the artefact is the operating system, Lubuntu, which is a lighter version of the popular Debian based OS, Ubuntu. The low resource requirements allows for it to run well as a live CD which operates from a USB to system memory. Packages were sourced through the Ubuntu Software Store and Synaptic Package Manager to ensure compatibility before testing and adaption.

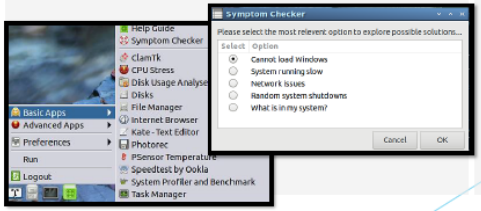
Design

As the key aim of the project was to develop a more usable solution, the design was focussed on being intuitive to use and to allow the user to customise their experience with accessibility settings. Additionally, the resources were to be accessible and provide clear and meaningful content. The interface was designed based on industry standard frameworks by W3C and ISO, plus Human Computer Interface (HCI) considerations.

Methodology

An agile approach, namely adapted Test Driven Development, was used for the software development using scrum backlog and sprint sheets were used for management. A traditional method for the project methodology with linear progression steps recorded with a Gantt chart.

Primary research included questionnaires and evaluating current solutions. Secondary research encompassed reading current thinking by practitioners and an academic literature review.



Future Work

1. Continual development and updating LLTT for compatibility and expansion.
2. Design a website for a support forum, development log and to download the toolkit.

Conclusion

The proposed solution has been proven in testing to be a functional, accessible and usable toolkit for a range of purposes. It has met the its desired aims and provides opportunity for future development - Initial feedback rated it 5/5 for functionality and 100% would recommend it to a friend. The implications on users include saving money, having more control of their data, being more time efficient, and producing less e-waste.

Primary research found that the typical price paid by questionnaire respondents was typically £50 per fix. As some operations like recovering data from a hard drive can take multiple hours, the financial impact of IT repair can be substantial for individuals and more so for businesses. Thus, the tool has the potential to save users a lot of time and money. As the aim of the artefact is to be accessible by most computer users and offer more comprehensive array of diagnostic, repair and benchmarking tools, it is applicable to a wide scope of audiences and use cases; from a lay person using it once a year to a fully qualified technician using it as part of their organisation's IT maintenance.

References
Statista (2019), *UK households: ownership of home computers 1985-2018*. Available at: https://www.statista.com/statistics/289191/household-penetration-of-home-computers-in-the-uk/ (Accessed: 24/03/2019)
Gibson, G. & Schroeder, B. (2009), "Computer Failure Data Repository (CFDR)". Available at: https://www.usenix.org/legacy/event/osdi06/posters/schroeder.pdf (Accessed: 1/1/2019)

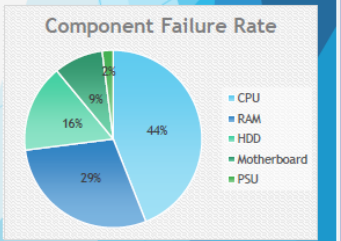


Fig. 1 Component Failure Rate (Gibson & Schroeder, 2007)

GPL Free Software Free as in Freedom
GNOME
lubuntu

UNIVERSITY OF PLYMOUTH