A DESKTOP APPLICATION FOR MANAGING SOFTWARE LICENSES AND INDEXING INSTALLED PROGRAMS

Mario FUKUOKA, Aleksander BOBIŃSKI,
Maciej GRZELCZAK, Patryk CHODOROWSKI
dr inż. Krzysztof GRUDZIEŃ, dr inż. Zbigniew CHANIECKI

Computer Science
Lodz University of Technology, International Faculty of Engineering

ABSTRACT

This project tackles the problem of keeping track of lawful use of software. Due to how ubiquitous the internet is, downloading and installing new software is very easy, yet keeping track of its licensing can be hard. It is difficult to tell when a license permits the usage of a program in a certain situation. Additionally, licenses can be confusing and easy to misunderstand. During the project, over 370 people are interviewed and surveyed and it is found that users who claim that they care about lawful use of software do not read licenses and do not know in entirety what they have installed on their computers. The proposed solution is a desktop application meant to help users search for programs which are installed but not known by them, organize them along with finding their corresponding licenses, and also analyze the licenses to summarize them for faster and easier understanding of their content.

1. Introduction

1.1. Background information

Nowadays, the installation of computer software is not a big issue, especially from the installation process point of view. It is easy to find software with a set of desired features. We can choose based on looking up benchmarks or seeing advertisements, and easily download a program. However, the problem arises when we want to know, according to the license, if we can use this software in some arbitrary way. What is the difference between the paid and trial version? The misunderstanding of license can lead unlawful use. Even if we install trial type software in a proper way, what should we do after the trial period is finished? Does it have to be uninstalled or just not used any more? Is leaving it installed considered misuse? The users want to avoid the situation where the software is used not in accordance to the license, and yet license conditions and agreements can be difficult to understand. The problem is magnified when a non-technically skilled person installs computer software. Additionally, we may find ourselves in a situation where we are using a system with previously installed programs. How to check if the installed software is legal or not? Another issue is when several different people work on the same computer. Users are looking for a way to manage owned or newly installed software. They want to be sure, so that if they have to submit the computer to the legal software audit, that everything is in line with the license rules. Today's world is ever changing, in a constant hurry. More and more people use computers, but not everyone is an expert in the computer science field.

How can we be sure that our computer software is legal? How to manage computer systems in such a way?

1.2. Problem finding

To outline the process, we tackled the problem by first creating questions relating to the purpose of licenses and their use. Based on these questions, a list of interview questions was constructed. 15 interviews were then performed with people of different levels of familiarity with computers & IT. The interview answers were then analyzed and we used the findings and

data trends to construct a survey in order to confirm them with a larger sample size. More than 360 people answered the survey. To help with analyzing the survey data, a program for associating responses was written, and used to formulate the final definition of our problem.

More in-depth, the first step of finding the problem was asking ourselves basic questions about end user licenses, such as who benefits from their existence, and conversely who would benefit from their absence. This was done in order to gain deeper insight about their role and purpose. This insight was used to construct open interview questions.

The form of an interview was chosen due to the ability of the interviewer and the interviewee to freely talk to each other during it. The interviewee could clarify their responses if necessary, and additionally mention their own thoughts or problems with licenses and installation of software, which we had not thought of during our first step. In total, 15 people were interviewed, ranging from very basic users to teachers and professors of computer science courses.

Multiple answers included things such as not having enough time to read end user licenses, finding them hard to understand, and also being unable to recognize an installed program. Trends such as resorting to piracy due to lack of money were deemed to be a limiting factor beyond the scope of this project, and we decided not to pursue them further.

In order to confirm the trends and common complaints we found during the interviews, a survey was constructed with questions based on our findings. The survey included 10 questions, and received more than 360 responses. Due to the large volume of data from the survey which needed to be processed, a program was created which was used to associate one kind of question response with other question responses.

The program would take the spreadsheet of questions and answers from the survey as the input. Using the program, it is possible to select a specific group of respondents who answered a question in a particular way, and then show how they answered all other questions. For example, by selecting the people who answered that they greatly care about the legality of their computer, the program generates a new spreadsheet, where it is possible to directly look up how many of these respondents said that they have found unknown software on their computers in the past. Employing this program in our analysis has greatly helped in definition of the problem, as it made certain relationships in the way people answered various questions much easier to study.

After analysis of the data, the problem definition was decided upon:

People care about owning legal software, but do not really know what they have installed on their computers, and do not bother reading the licenses.

This was concluded from several data relations:

- More than half of our respondents said that they find having legal software quite important or very important, but of the same group more than half also said that they either barely care about licenses, do not care about them at all, or never thought about it. This means that there are people who care about software legality, but don't care about software licenses. (Fig. 1, 2)
- Over half of respondents found software of unknown origin on their machines. Of those, about 45% said that software legality is either quite or very important to them. (Fig. 3)
- 48% of people said that they have no time to read licenses, and 27% said that they are too long or complicated. Within this combined group, 43% either thought of them as quite important or very important. (Fig. 4, 5)

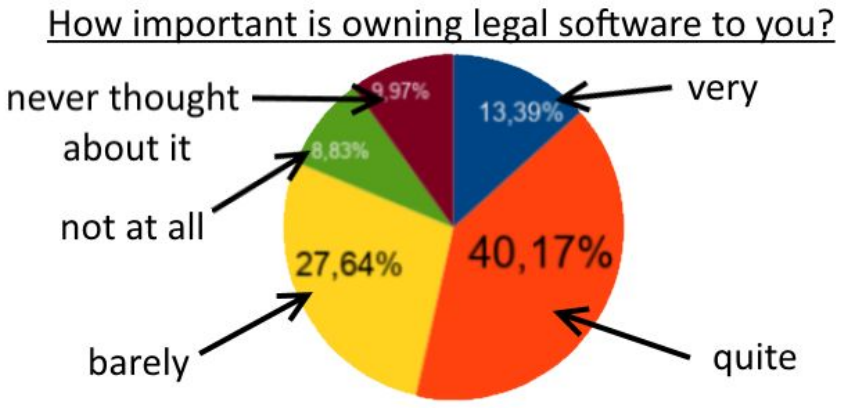


Fig. 1
source: own research survey

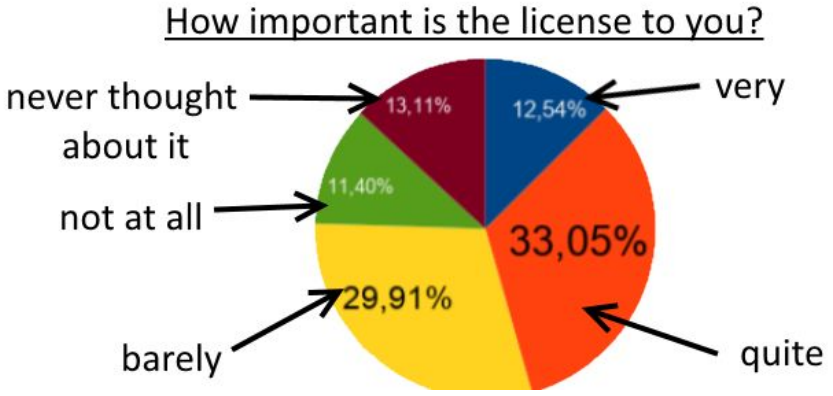


Fig. 2: Responses from the same group as in Fig. 1
source: own research survey

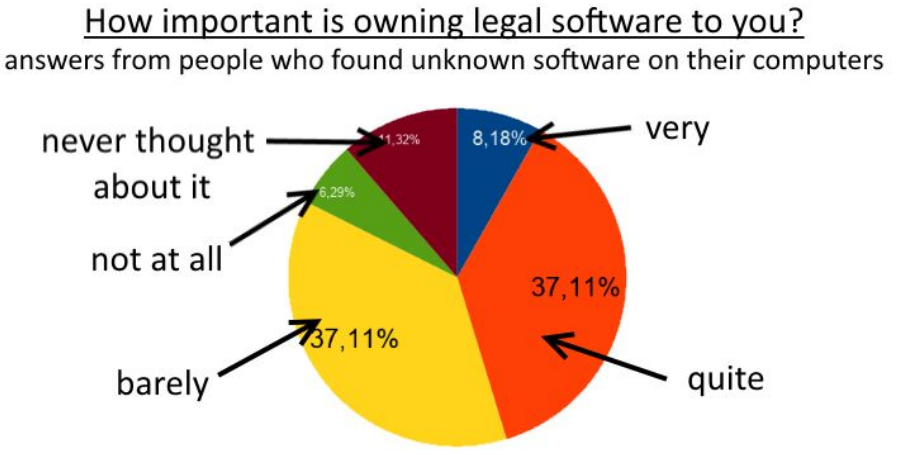


Fig. 3
source: own research survey

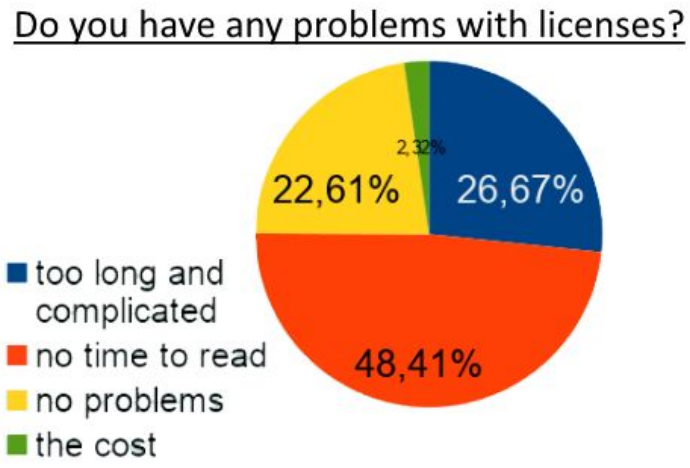


Fig. 4
source: own research survey

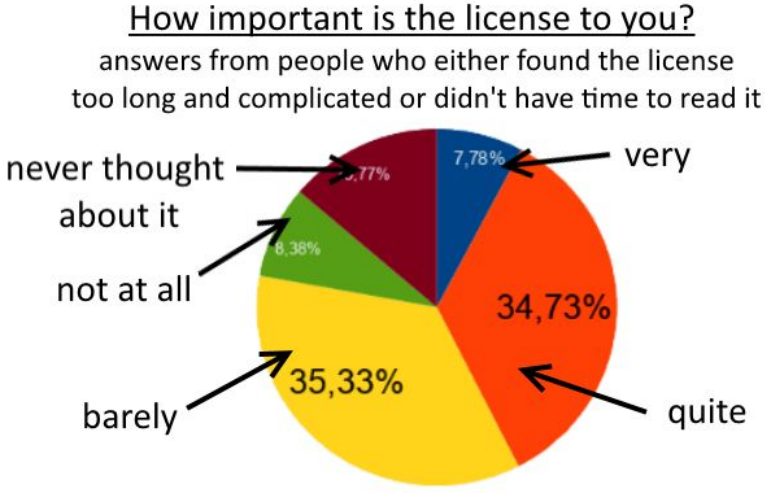


Fig. 5
source: own research survey

2. Idea finding

2.1. State of the art

An internet search may reveal that there already are potential partial solutions to this problem. It seems, however, that none of them addresses the whole problem as it is defined here. Software-based solutions of interest to us can be grouped into a few main categories:

License Analyzers

License Analyzers take the text content of a license as input, search for words and phrases that it deems to be of interest to the user, and point them out, or otherwise make the meaning and consequences of a license easier to understand. For example, an analyzer may find and mark what it interprets to be mentions of tracking or using personal data. This approach tackles the license length & complexity problem, however an effective implementation is difficult and fairly rare due to the complexity of having to interpret natural language combined with the difficult law terminology.
Examples: EULAlyzer[1], SpywareGuide license analyzer[2]

License Enforcers

License Enforcers, as their name suggests, enforce the correct usage of the license. They can be added by developers to their software so that users have to use it according to the license. This type of software tackles the problem of not knowing what you can lawfully do with a program, because it enforces lawful usage of the software. It is notably a solution on the vendor/developer side, instead of the user side.
Examples: Reprise License Manager[3], Protection! licensing toolkit[4]

Command line programs

Many advanced users enjoy using command line programs to perform complex file and data manipulations by simply typing the operation into a text window, without the need of a visual interface. This is no different in the case of finding installed programs. There are many programs and commands available even by default on Windows which can be used to list all installed programs, which could help people find out about the programs that were not installed by them directly, or programs that they have forgotten about. The

main downside of this method is the relatively high barrier of entry. Presumably, most of the basic Windows users barely even know about the existence of such a tool as the command line, and would likely be uncomfortable with using it at their technical skill level.
Examples: WMIC "product get (...)" query[5]

File managers

File managers help you manage files you have installed on your computer. They make it easier to stay organized. One of the secondary problems that was mentioned by interviewees and respondents was the tediousness of having to keep track of license files, which this type of program is designed to counteract.
Examples: Total Commander[6]

2.2. **Innovative ideas**

It was decided that before we would commit to a single idea, a more variable number of solutions should be explored. At the suggestion of our supervisors, we employed an impact/difficulty matrix. First, it was agreed upon that no matter how far-fetched an idea seemed, it could be added to the matrix.

After a sufficient number of ideas was reached, each idea was evaluated by assigning impact and difficulty values to them - the former describing how much of an impact the solution would have on the problem and the problem related environment in general, and the latter describing its difficulty in implementation. Based on this, the best candidates for implementation were chosen.

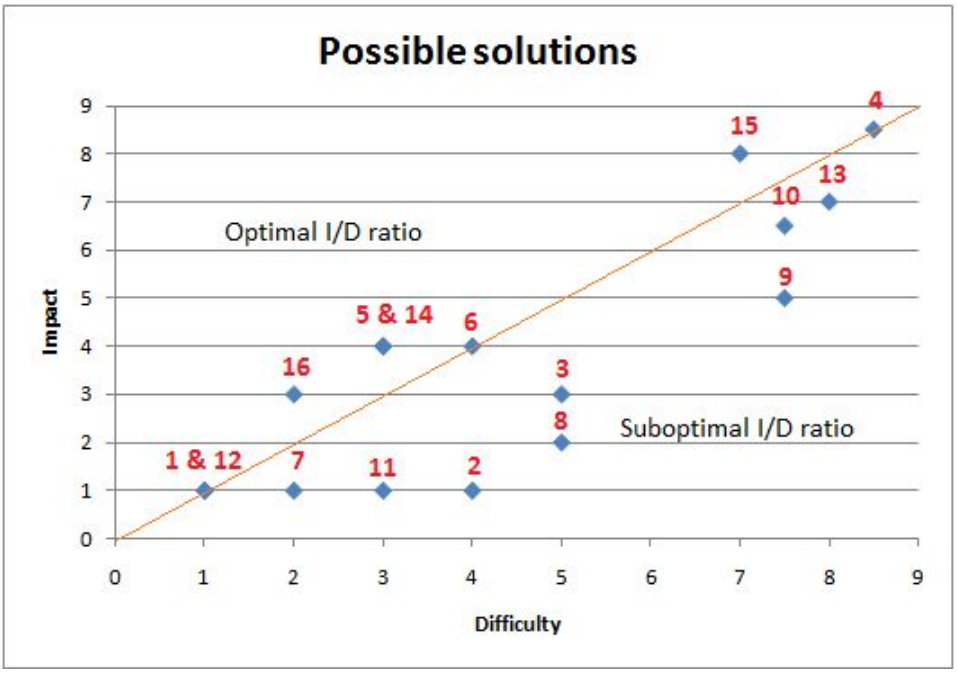


Fig. 6: Plot of potential solutions with their impact on the problem/environment vs. difficulty of implementation. We considered only the potential solutions above the orange line, as they have a high ratio of impact to difficulty.
source: our own materials

Nr.	Solution	Difficulty	Impact
1	A poster, to show people that they do not know as much as they think they do about licenses and that it is worth reading them	1	1
2	A poster, to show people that they do not know as much as they think they do about licenses and that it is worth reading them	4	1
3	Video game which will raise awareness that writing software is not an easy task and that people should not pirate them.	5	3

4	A website where people can read a summary/explanation of a software license, written by other users.	8,5	8,5
5	Software which reminds the user about expiration dates of his trials and subscriptions.	3	4
6	Motivational program which would keep track of how much software is legal, and then record progress done towards having a fully lawful computer.	4	4
7	Tool for organizing time that will help the user find the time to read licenses.	2	1
8	License reading course organized by a lawyer.	5	2
9	Virtual space where user can install try installing software in order to see whether it also installs other software with it.	7,5	5
10	Docker for windows (operating-system-level virtualization)	7,5	6,5
11	Software which will tell you what licenses, or what parts of a license are interesting to you depending on who you are e.q. probably your expertise/technical familiarity, profession, etc.	3	1
12	A quiz meant to show people how little they know about licenses, thus encouraging them to be better informed.	1	1
13	Q&A service with lawyers for answering license-related questions	8	7
14	Software & license manager/database	3	4

15	License analyzer/summarizer	7	8
16	Software lister to list hard to find installed software by searching the registry	2	3

2.3. **Main idea selection and justification**

Ultimately, it was decided that our solution would involve the license analyzer, file manager and a software lister, which would list programs installed on a given computer. Individually, these solutions have been tried already, as is mentioned in the State of Art section. As is also mentioned, however, none of the individual types of programs fully covered our problem definition. Thus, our planned application would integrate these components together in order to tailor the solution to our defined problem. The license analyzer allows easier and quicker comprehension of a license, the software lister can provide an in-depth list of installed software, and the file manager part of the program would serve to store found programs, as well as bind license files to them for quick access and analysis using the analyzer.

Initially, implementing a software analyzer was deemed to be a difficult project just on its own. Not only would the program have to parse natural language, it would also have to be adapted to the law terminology used in the license, which is a legally binding document after all. This meant that we had to choose between a potentially lackluster software lister and organizer combination, and a risky and difficult endeavour of implementing our own license analyzer. Fortunately, while researching already existing approaches to this problem, we found an analyzer program that was web-based instead of being a standalone app. It was possible to put the license text to be analyzed on a textbox within the website, then click a button to navigate to a page with the results. This meant that we could utilize the web interface that the website already used to communicate with a web browser. Using this interface, it is possible to mimic exactly what a web browser does after pasting the license into the textbox and clicking ‘analyze’. We can then intercept the analysis page that is sent in response, and display it in-app or save it to a file.

This solution seemed to be the most convenient, as in theory it covered all of the problems mentioned in the problem definition, and so it is the one that was chosen to be implemented.

3. Solution implementation

3.1. Detailed solution description

Due to the nature of our program, a desktop app would be the most fitting type of solution to create in order to facilitate the functioning of our program. We decided to implement it using Python 3.6, due to its relatively simple syntax, which makes it easier for us to create complex systems faster, and also its very large support of various libraries, in particular http method & web scraping libraries which we could use for the license analyzer part of the program.

As of the time of writing of this article, these are the main libraries currently being used in our application:

- **appJar** - a GUI library which lets us create the visual interface and layout of the application. Chosen for simplicity and speed.
- **requests** - a library which allows python to communicate with various website APIs through the usage of HTTP methods. Essential for the web-based license analyzer to work.
- **webbrowser** - at the moment, this library is used only to open the license analysis html response file in the web browser, though the analysis result is planned to be viewed purely through the application, so this library may become obsolete in future versions.
- **os, glob** - other less significant but still crucial libraries.

The program is divided into 5 separate parts:

- GUI
- Storage
- Controller/Organizer
- License Analyzer
- Software Lister

The GUI is connected to the Licence Analyzer and the Storage. Requests for parsing a license are fed straight into the License Analyzer as a String that contains the license text. The analyzer then sends a request to an external tool that analyzes the license. The obtained HTML response is saved to a file and opened for the user inside his default web browser.
The request to search for installed software is passed from the GUI into the controller. The controller then requests a scan from the Software Lister. The lister performs a quick scan searching for .exe files. In the quick scan method

the results of the scan are returned as a list of items into the controller. The controller then adds the found software entities to the storage.. The storage saves the entries inside a binary file. It allows to easily add, get, delete or update any of the saved entries.

The original concept was to look for installed software using the windows registry system. This approach is obsolete due to the fact that the scanning process produces less accurate results than the quick scan and takes significantly longer to complete.

License Analyzer

The web-based license analyzer itself that has been mentioned throughout this document is located over at http://www.spywareguide.com/analyze/analyzer.php. In order to operate it remotely from within python, the 'requests' library is used to send HTTP requests with the license text as one of the parameters. Currently, the HTTP response which contains the analysis HTML file is saved to a new file called 'response.html' and then opened using the 'webbrowser' library in a new tab in the default browser. The analysis itself includes assigning various readability scores such as through Flesch-Kincaid test, or the Gunning fog index. It also searches for mentions of specific phrases, such as tracking, and marks them in the text.

Software Lister

The first thing the user sees is the main window (Fig. 7). In it there is a list of software, a list of licenses, license text area with contents of the selected license, 2 search boxes for searching software and licenses in their corresponding lists and 3 buttons: one for adding software, one for deleting and one for parsing the selected license.

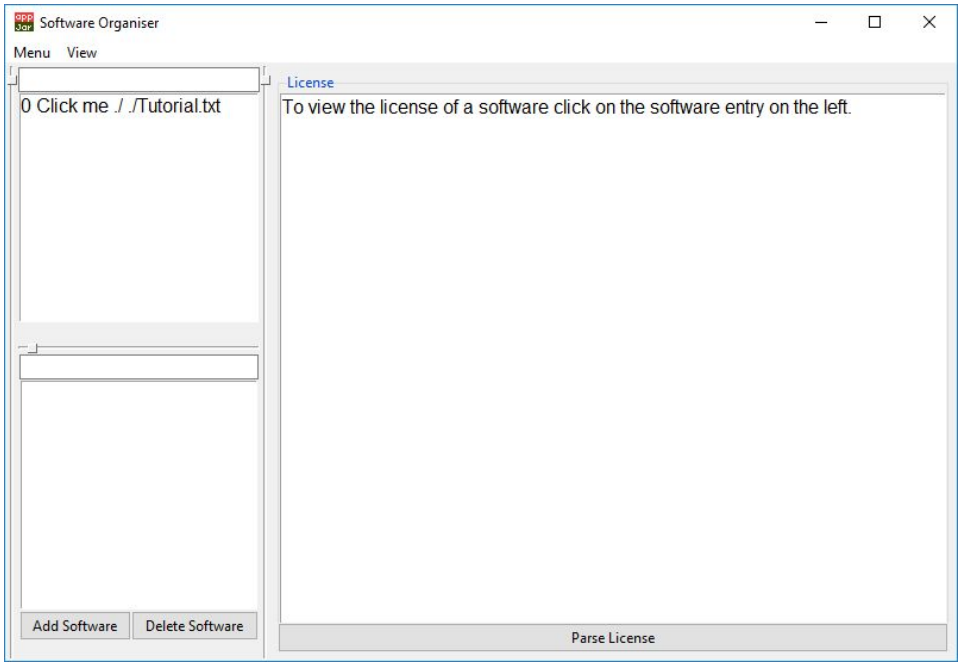


Fig. 7: Main Window
source: own app

Using “Menu” menu one can start scanning the computer for software (Fig. 8). It takes around a minute (Fig. 9) and finds every executable file that is not a windows file and is not used for installation or uninstallation of software. In this example it found around 2000 results (Fig. 10).

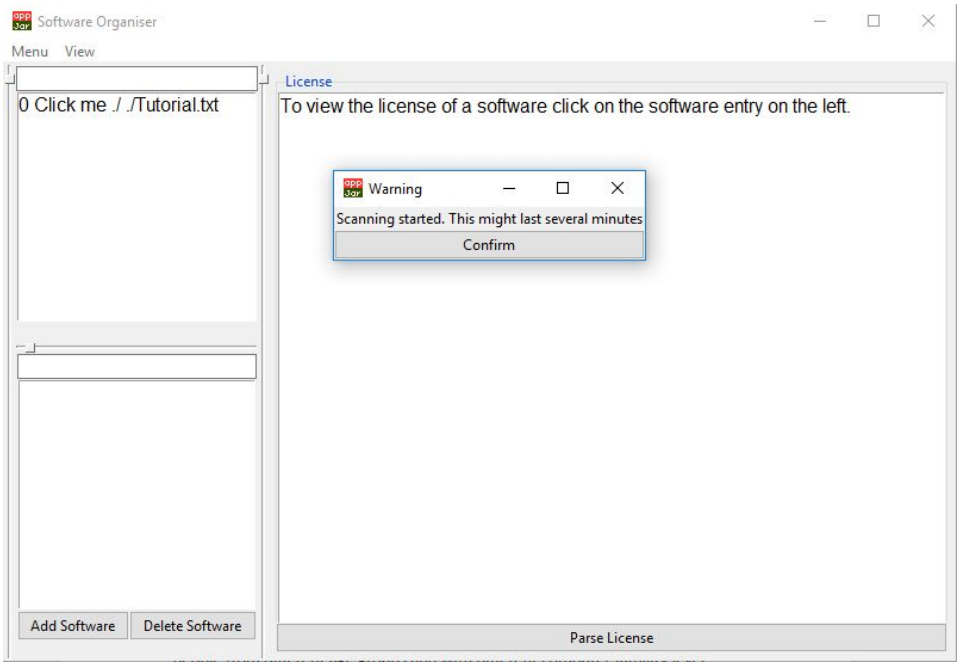


Fig. 8: Scan Start

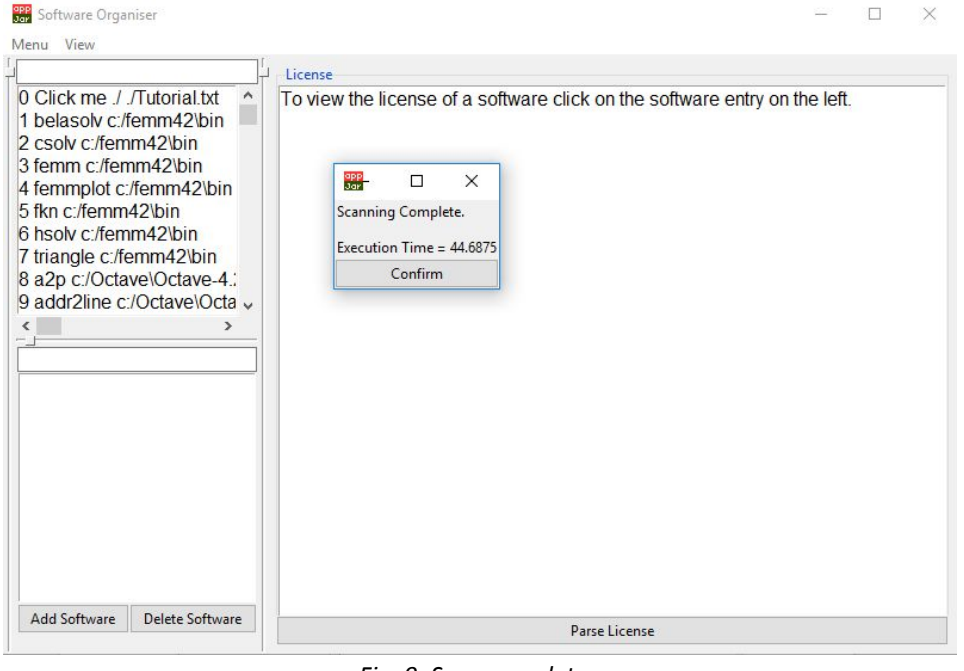


Fig. 9: Scan complete

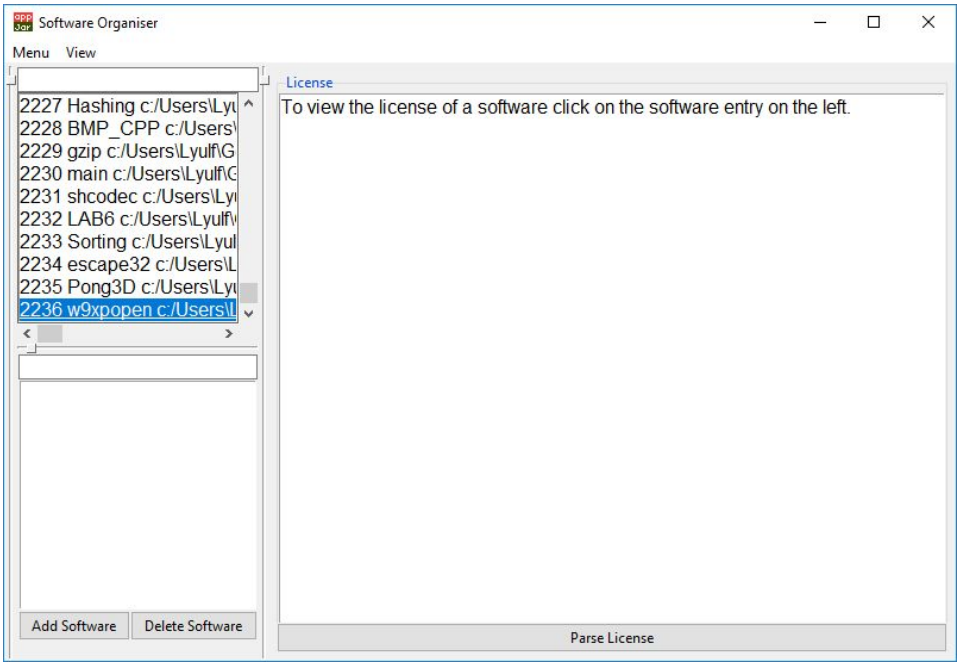


Fig. 10: 2236 results

If a software is not present on the list or is present, but it does not have every information, one can edit its values using the edit window (Fig. 11).

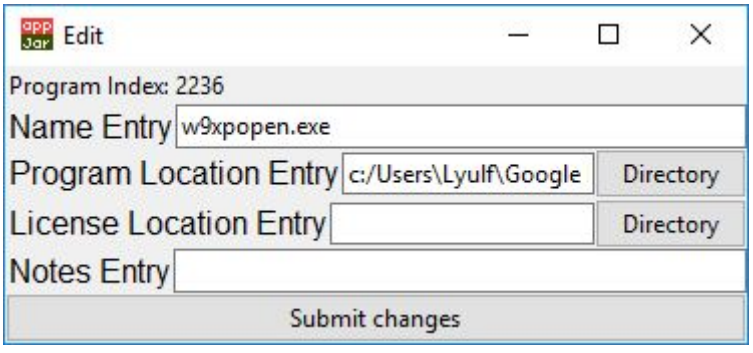


Fig. 11: Window for adding/editing software

After selecting software, the license list automatically searches for licenses in softwares license location. Selecting license displays its contents in the license text area (Fig. 12).

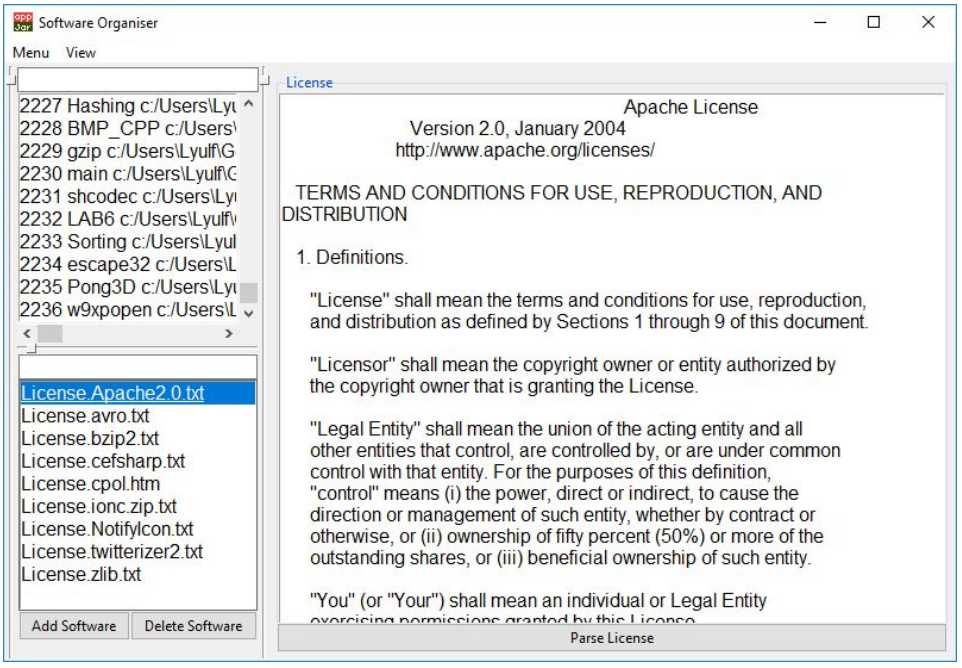
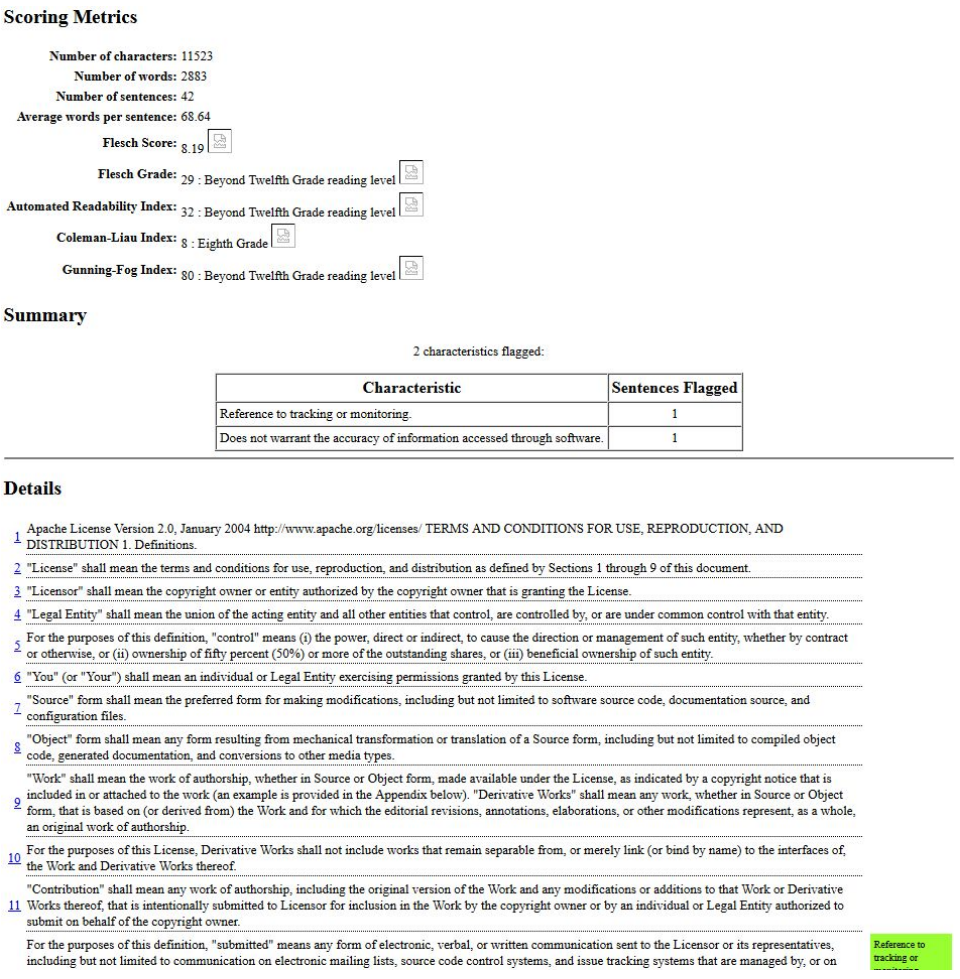


Fig. 12: License View

Pressing parse license button sends the contents of the license to http://www.spywareguide.com/analyze/analyzer.php in the body of a HTTP GET method, and saves the response text to response.html (Fig. 13)



Scoring Metrics

Number of characters: 11523
Number of words: 2883
Number of sentences: 42
Average words per sentence: 68.64
Flesch Score: 8.19
Flesch Grade: 29 : Beyond Twelfth Grade reading level
Automated Readability Index: 32 : Beyond Twelfth Grade reading level
Coleman-Liau Index: 8 : Eighth Grade
Gunning-Fog Index: 80 : Beyond Twelfth Grade reading level

Summary

2 characteristics flagged:

Characteristic	Sentences Flagged
Reference to tracking or monitoring.	1
Does not warrant the accuracy of information accessed through software.	1

Details

1 Apache License Version 2.0, January 2004 http://www.apache.org/licenses/ TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION 1. Definitions.

2 "License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document.

3 "Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License.

4 "Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity.

5 For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity.

6 "You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License.

7 "Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files.

8 "Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types.

9 "Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below). "Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship.

10 For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof.

11 "Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner.

For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on

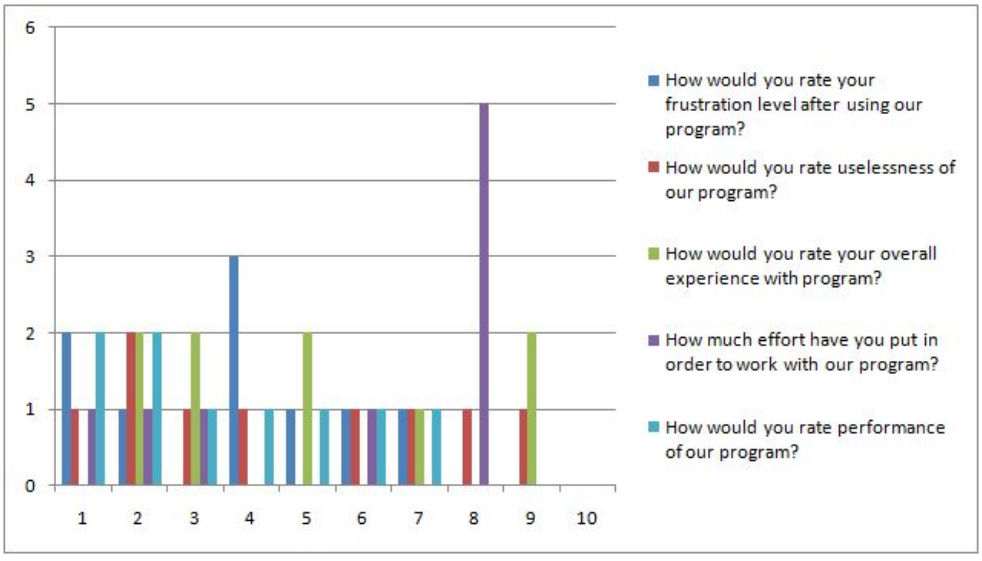
Reference to tracking or

Fig. 13: Analysis results

source: raw HTML response from http://www.spywareguide.com/analyze/analyzer.php

3.2. **Ways of verification**

In order to gather feedback from our clients group, we gathered about 10 people from different age groups and with different computer familiarity level, and let them use our program. After that, we discussed the user experience with every person involved in our test. In order to normalise and then compare the opinions, we created a user experience survey, which we then distributed to all testers. The survey structure is simple. It is divided in 4 parts, 3 for every module of our program and last for general experience. The questions were mostly in NASA-TLX [7] style, questioning about the frustration level of difficulties in usage of our program. The survey results came out to be drastically different from each other. There was not a single survey which would give a similar answer in at least one module. That answers were not actually satisfying, due to the fact that they did not point us out any particular problem, with which our beta-users were struggling, giving us ideas of areas to improve. However the fact, that the feedback were mostly positive, ensured us in fact that the direction we have chosen is the right one. Several ideas have been already suggested, such as having the license field have 'tabs' like web browsers do, and having the analysis of a licence pop up as another tab, so that it is possible to freely switch between the analysis and the original text. Another suggestion was a drop-down showing attributes of a program to increase the program clarity, as opposed to the current list of programs which does not have rows divided into columns, making the attributes blend together.



4. **Conclusions and perspectives**

The initial research was rather slow and required a lot of inquiry due to the specialized nature of the problem. We managed to research the way the users felt about licenses and software legality in general, and extract raw data that could lead us to the problem definition. It was difficult to establish any correlations within the resultant data, but we had a breakthrough when we created the custom data analysis program and performed calculations on various survey data. It became much easier and faster to see relationships and correlations within the survey responses. From this improved analysis, the problem definition was derived.

Since not many of the solutions thought up during the impact/difficulty matrix that placed within the 'optimal' zone on the graph were feasible (and also partly because we are computer scientists), we opted to implement a desktop application. Despite some of its shortcomings and mixed feedback in some areas, the general user response was positive. Additionally, the solution does address the main problems and if given more resources and time, it could prove to be a quite useful tool.

The shortcomings it does exhibit are in part because of development tool choices. For example, some time into developing the app, we found out that appJar unfortunately does not support events in tables. This is why the current version uses a simple list. It has greatly affected the visual clarity of the presented software list. This means that if we tried to implement a table of programs (so that its related attributes such as 'path' or 'license' can be separated into different columns in the same row) we could not make the table interactive, as it will not react to things such as mouse clicks. We decided it was too late to switch and start developing from the beginning in another GUI library, and now the list of software on the left hand of the program does not have separated software name, path, license path and notes, and instead these parameters are stuck in the same line, only separated by a space. Given more time, perhaps we could find a suitable workaround, or even switch to a more advanced GUI environment like Qt.

Another shortcoming turned out to be the service which we used for license analysis. The analysis tool itself is rather buggy. On the analysis page, it will sometimes display hundreds of lines of the same error message for no reason, regardless of whether the input changed or not. Coincidentally, the error message does not overwrite nor influence the rest of the analysis in any

way, so as a workaround we created a function that would catch the error strings and replace them with empty strings, so that in the end there was no difference between a correct response and 'erroneous' response. In addition, the analysis itself is rather basic. It is however the only free online analysis tool accessible with an API, so in the end we incorporated it in our code out of necessity for any such tool at all.

Given more time and resources, the priorities in development would surely be to develop our own analysis tool, and improving the UI design to make it look less dated, in the worst case through a switch to a different GUI library.

References

[1] EULAlyzer: https://www.brightfort.com/eulalyzer.html
[2] SpywareGuide analyzer: http://www.spywareguide.com/analyze/analyzer.php
[3] Reprise License Manager: http://www.reprisesoftware.com/index.php
[4] Protection! license toolkit:
http://www.jproductivity.com/products/protection/protection.htm
[5] Using WMIC to see installed programs:
https://community.spiceworks.com/how_to/111076-using-wmic-to-retrieve-a-list-of-all-installed-programs
[6] Total Commander: https://www.ghisler.com/
[7] Wikipedia: NASA-TLX https://en.wikipedia.org/wiki/NASA-TLX