NOVA SCHOOL OF SCIENCE & TECHNOLOGY

DEPARTMENT OF COMPUTER SCIENCE

DAVID ALEXANDRE PASSÃO PEREIRA

Bachelor in Computer Science

USABILITY EVALUATION OF OFLAT PLATFORM

MASTER IN COMPUTER SCIENCE

NOVA University Lisbon

October, 2023

NOVA SCHOOL OF
SCIENCE & TECHNOLOGY

DEPARTMENT OF
COMPUTER SCIENCE

USABILITY EVALUATION OF OFLAT PLATFORM

DAVID ALEXANDRE PASSÃO PEREIRA

Bachelor in Computer Science

Adviser: Vasco Miguel Moreira Amaral
Associate Professor, FCT - NOVA University Lisbon

Examination Committee:

Chair: Teresa Isabel Lopes Romão
Associate Professor, FCT - NOVA University Lisbon

Rapporteur: Isabel Sofia Sousa Brito
Coordinator Professor, Instituto Politécnico de Beja

Member: Vasco Miguel Moreira Amaral
Associate Professor, FCT - NOVA University Lisbon

MASTER IN COMPUTER SCIENCE

NOVA University Lisbon
October, 2023

Usability Evaluation of OFLAT Platform

This document was created with the (pdf/Xe/Lua)LaTex processor and the NOVAthesis template (v6.10.18) [30].

My "backup system" made this possible.

Acknowledgements

Firstly, I would like to thank my advisor professor Vasco Amaral, who, despite all his work, always accompanied me through almost weekly meetings, without which this work would not have been possible. My sincere thanks for giving me this opportunity. This first acknowledgment is also extended to professors António Ravara and Artur Miguel Dias who were available for this collaboration.

I would also like to express my gratitude to the FCTense community that has contributed greatly to my academic, professional, and personal development for the last 5 years. In the context of this work, a special thanks to the TC students who participated in the surveys and in the experiment.

Finally, on a more personal level, I thank my mother and father for never giving up on me and for all of their support. Aunts, grandmother, grandfather, and cousins, distant but always present, thank you for the motivation. My sister, thank you for making happier days. Thank you for making this possible.

“Al andar se hace el camino.” (Antonio Machado)

Abstract

In Formal Languages and Automata Theory (FLAT), interactive tools are critical because mathematical concepts have an algorithmic exposition that significantly benefits from presentations. Learning concepts and techniques crucially requires the understanding and visualization of algorithms.

Given these concerns, the OFLAT application was developed. To be valid, it is necessary to offer a rich set of features that is intuitive and easy to use, supporting the students' learning process.

This research aims to understand and summarize the existing similar tools and how this type of assessment was made to those tools, compare the OFLAT platform with similar existing ones, and apply similar and other methods to make the assessment and present improvement suggestions for this application.

Firstly, a systematic mapping study is made to gather information about similar existing tools and their assessments. Then, the evaluation of OFLAT is made through empirical studies (a controlled experiment and surveys to users) to systematically evaluate the platform, using techniques and methodologies of Software Engineering, in terms of its usability and, in general, its quality.

The assessment results in a qualitative and quantitative evaluation with the participation of students enrolled in the Theory of Computation (TC) course. These results show that OFLAT promotes the level of correctness of actions (is more effective), promotes good usability, and does not require much effort.

As a result of this study, guidelines for improving OFLAT user's experience were obtained. It was concluded that the current version of OFLAT being evaluated is less error-prone and provides a better user experience than the previous version. With two iterations, we observe that OFLAT is converging to a better usable tool than the previous version in comparison to the competing solutions.

Keywords: Automata Theory, Empirical Evaluation, Formal Languages, Process Mining, Software Engineering, Systematic Literature Review, Theory of Computation, Usability

Resumo

Na área de Linguagens Formais e Teoria dos Autómatos (FLAT, em inglês), é particularmente importante o uso das ferramentas interativas, porque os conceitos matemáticos têm uma exposição algorítmica que beneficia muito com as apresentações executáveis. A aprendizagem dos conceitos e técnicas requerem crucialmente a compreensão e visualização dos algoritmos.

Dadas estas preocupações, foi desenvolvida a aplicação OFLAT. Para ser útil, é necessário não só que ofereça um conjunto rico de funcionalidades mas também que seja intuitivo e fácil de usar e acompanhe o processo de aprendizagem dos alunos em FLAT.

Este estudo tem como objetivo permitir entender e resumir as funcionalidades das ferramentas semelhantes, comparar a plataforma OFLAT com outras similares, aplicar métodos semelhantes (e/ou outros) para fazer a avaliação e apresentar sugestões de melhorias para esta ferramenta.

Primeiramente, foi feito um estudo e mapeamento sistemático para reunir informações sobre ferramentas semelhantes existentes e os seus métodos de avaliação. Em seguida, a avaliação ao OFLAT é feita através de estudos empíricos (experiência controlada e inquéritos aos utilizadores) para avaliar sistematicamente a plataforma, utilizando técnicas e metodologias de Engenharia de Software, ao nível da sua usabilidade e, em geral, da sua qualidade.

O estudo resulta numa avaliação qualitativa e quantitativa, com a participação dos estudantes inscritos na unidade curricular Teoria da Computação (TC). Esses resultados mostram que o OFLAT promove o nível de correção das ações (é mais eficaz), promove uma boa usabilidade e não requer muito esforço.

Como resultado deste estudo, foram obtidas orientações para melhorar a experiência do utilizador do OFLAT. Concluiu-se que a versão atual do OFLAT em avaliação é menos propensa a erros e proporciona uma melhor experiência de utilização do que a versão anterior. Com duas iterações, observamos que o OFLAT está a convergir para uma ferramenta melhor utilizável do que a versão anterior em comparação com as soluções concorrentes.

Palavras-chave: Avaliação Empírica, Engenharia de Software, Linguagens Formais, Mineração de Processos, Revisão Sistemática da Literatura, Teoria da Computação, Teoria dos Autómatos, Usabilidade

Contents

List of Figures

List of Tables

Glossary

Systematic Literature Review A means of identifying, evaluating, and interpreting all available research relevant to a particular research question, or topic area or phenomenon of interest. [29] 17

Systematic Mapping Study A means of categorizing and summarizing the existing information about a research question in an unbiased manner. 8, 15, 17

Triangulation Taking different angles towards the studied object to provide a broader picture and increase the precision of empirical research 8, 19

Acronyms

FLAT Formal Language and Automata Theory 1, 17

SUS System Usability Scale 10, 24, 25

TC Theory of Computation 1

UEM Usability Evaluation Method 7, 19

WIMP Window, Icon, Menu, Pointing device 7

1 | Introduction

1.1 Context and Description

The area of Formal Language and Automata Theory (FLAT) contains important bases in Computer Science [2] and is taught in many faculties, in courses like Theory of Computation (TC). However, typically the students' success rate is relatively low compared to other courses. The reasons are manifold but usually converge on the fact that the introduced concepts are considered too abstract to be understood by a regular student. Therefore the learning process becomes demotivating, and "the majority of students emerge from this course with little insight, lesser interest, and no enthusiasm" [53].

There are many challenges in teaching automata theory, including outdated and ineffective material that demands a mathematical background. Mathematical concepts are frequently linked to comprehension and application issues. Knowing and visualizing algorithms is essential for understanding concepts and procedures, especially because "engineering students prefer active learning style instead of a passive lecture-driven style", according to [23]. Hence, an algorithmic presentation greatly benefits from executable presentations, and interactive tools are particularly important.

1.2 Institutional context

Given the previously described concerns, the OFLAT[1] application was developed by the FACTOR team[2] at NOVA LINCS (cooperation between researchers at Universidade da Beira Interior and NOVA School of Science and Technology). To be useful, it is necessary not only to offer a rich set of features but also to be intuitive and easy to use. However, a conclusive and extensive assessment is still to be made.

This project consists of making this assessment, comparing the OFLAT platform with other similar existing ones, and finding opportunities for improvement, so students have greater adherence to OFLAT.

[1] OFLAT: http://ctp.di.fct.unl.pt/FACTOR/OFLAT
[2] FACTOR: http://ctp.di.fct.unl.pt/FACTOR/

1.3 Motivation

As FLAT is such a universal and crucial subject in the training of Computer Engineers, it has become an area of special attention. To realize and meet the students' needs is essential to contribute to and facilitate their learning.

Despite being an issue addressed for a long time, there are still recent reports and studies that portray a lack of motivation and interest on the part of students in diverse topics of this subject: "surveys conducted among our students showed that many of them were applying some theoretical concepts mechanically rather than developing a significant learning of them, leading to a lack of motivation and interest." (2004) [9], "some students resist this material because they see it as overly theoretical, but much of it has important practical applications. In this position paper, we discuss some of these and advocate a view of the course with applications emphasized to provide motivation" (2019) [8], "challenges in teaching the pumping lemma in automata theory course" (2005) [22] and research "into the actual learning difficulties experienced by students with the different topics" (2009) [38] are proof of the persistence of these problems.

For 50 years, some applications have been developed to support the learning of FLAT. Some are complete but only work on the desktop. Others that work in web browsers are poorly designed and not always fully interactive. Sometimes the concepts are implemented in a way distant from the mathematical definitions used in classes, and the choice of the implementation language sometimes does not collaborate in this aspect. Ideally, students should understand the implementation code and its close relationship with the definitions. Also, with no usability concerns in mind (as concentrated in functional aspects), the existing tools were not followed by empirical usability studies. In the rare case they were produced, these were carried on in an ad-hoc fashion, non-systematic and non-reproducible. Therefore, their conclusions could not be considered strong enough evidence to be conclusive.

OFLAT was created to fill this gap, but an extensive usability assessment is still missing. A first pilot assessment was done before [41], but it had limited results as it was not carried on in the real context of use during the progress of a course during the semester.

Therefore, in this work, we want to answer the following question: *How to properly design and implement an empirical study to assess OFLAT in a real usage scenario, such as a Computer Theory Course, over a semester, dealing with its dynamic nature, that will allow improving its current status in terms of usability for FLAT education?*

Concluding whether learning is beneficial through this tool is more challenging to measure because it would be necessary to co-relate the use of the tool and possible improvements in the students' results in the course. Therefore, the focus is on the educational characteristics of the tool and its usability.

This work is built on previous work on the usability assessment of OFLAT [41]. That work was the first usability experiment done with the tool that consisted of a pilot study with very few subjects. Still, besides being the first, it brought several contributions,

namely a report on proposals for usability improvements of the new versions of the tool (that were taken into account in the next iterations of the tool) and the design and implementation of a complete experimental setup. However, these contributions were achieved with a limited set of users in a controlled experiment. Our contribution will address the specificities of conducting a usability study in the real context of use, having as subjects the students enrolled in a Computer Theory course during the semester.

1.4 Objectives

It is intended that this dissertation evaluates the OFLAT platform, with the techniques and methodologies of Empirical Software Engineering, as to its usability and, in general, quality. The different steps of an empirical evaluation are explored.

It is also intended to evaluate OFLAT concerning similar platforms/tools. A systematic literature review is crucial to identify gaps in this research topic.

This dissertation presents qualitative evaluation results supported by a significant number of users of different profiles and concrete and well-founded proposals for improvement. To obtain these results, qualitative and quantitative data will be collected using various methods like surveys, interviews, and tracking users' usage of OFLAT. The data is then analyzed so workflows, platform usage, usability friction points, qualities, and informal user feedback on OFLAT can improve. This, together with the design of the process and integration of tools (such as logging), will be the main contribution of this work.

1.5 Document structure

This document is organized as follows:

- Chapter 2 - Background: brief introduction, description, and direct application of essential concepts to this dissertation and developed work. OFLAT and its context are described, usability and empirical evaluation are explored, the data collection is approached;
- Chapter 3 - Related Work: presentation of some evaluations already done with OFLAT and similar tools are described as their functionalities and usability evaluation. To complement this study, a systematic mapping study is performed step by step;
- Chapter 4 - Engineering OFLAT with usability concerns: contains the usability experiment process diagram, two surveys, the results from the previous work on the usability assessment [41] and OFLAT's supported functionalities before and after the usability assessment;

- Chapter 5 - OFLAT version 2.1 and related usability assessment tools: contains information about usability assessment tools;
- Chapter 6 - Controlled Experiment: definition, design and execution of the performed experiment;
- Chapter 7 - Analysis of the controlled experiment: results analysis of the performed experiment;
- Chapter 8 - Conclusions and Future Work: final conclusions and suggestions for future work.

2 | Background

In this chapter, the concepts related to the presented work are introduced, thus generating relevant information to understand the following chapters and the work in general.

First, OFLAT is presented, then some Interaction Design concepts are explained, followed by the Empirical evaluation methods. Finally, data types and methods of analyzing data using Process Mining are mentioned.

2.1 FACTOR and OFLAT

FACTOR (Functional ApproaCh Teaching pOrtuguese couRses) is a project that aims to promote the use of OCaml in the Portuguese academic community, namely through the support of teaching approaches and tools. In particular, it seeks to broaden and consolidate the user base of software and teaching materials in OCaml in the Portuguese community for subjects in Computational Logic and Foundations of Computing in Computer Science courses [16].

The project activities are developed in the context of The NOVA Laboratory for Computer Science and Informatics (NOVA LINCS) at DI/FCT/UNL & The ReleaseLab at DI/UBI. One of those project activities includes OFLAT [32].

The mentioned work consists of an OCaml and JavaScript web application (Figure 2.1) that allowed the study of Finite Automata and Regular Expression and enabled the performing of simple exercises. The work has been continued over the last two years, leading to improvements in the application. It is now more complete and has more features such as the ones described in this document's next chapter.

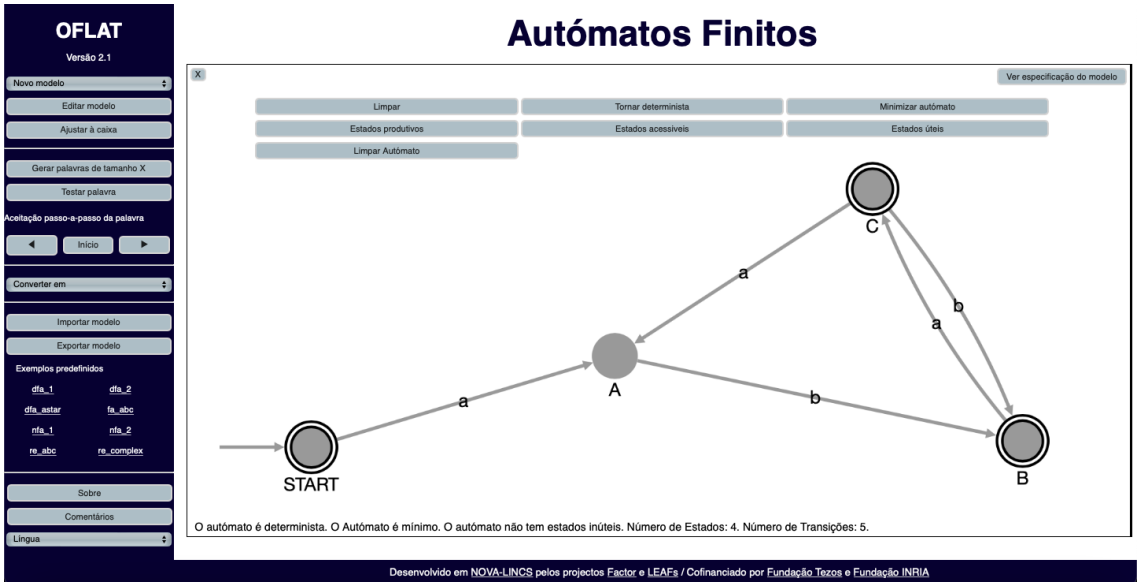


Figure 2.1: Screenshot of OFLAT layout (version 2.1)

2.2 Interaction Design

Interaction Design can be explained as the design of the interaction between users and products/systems. The products should support interaction and the way people communicate in their lives. It is about creating user experiences that enhance and augment how people work, communicate, and interact. It is fundamental to all disciplines, fields, and approaches concerned with researching and designing computer-based systems for people [13].

2.2.1 User experience

We can find the definition of UX in different sources, however, in this work, we stick to the following: The interaction design process revolves around the user experience (UX). This refers to how a product functions and is utilized by users in the real world.

More specifically, it pertains to how users feel about a product, including their contentment and joy when using, viewing, holding, and opening or closing it. It encompasses everything from their overall assessment of how enjoyable it is to use to the sensual impact minor elements have on them [43].

2.2.2 Usability

According to ISO (International Organization for Standardization), the "objective of designing and evaluating systems, products, and services for usability is to enable users to achieve goals effectively, efficiently and with satisfaction, taking account of the context

of use". Usability is relevant when designing or evaluating interactions with a system for development, review, comparison, or market research, for example, [27].

2.2.3 Usability Evaluation

Evaluation is an integral part of the design process life cycle. It aims to test the design's functionality and usability and identify and rectify any problems. It can also determine the user's attitude and response to the system.

Nielsen and Landauer, knowing by publishing [35] (1993), suggest that "usability testing with a single participant will find about a third of the usability problems, and that there is a little to be gained from testing with more than five. While this may be true of observational studies that aim to uncover usability issues, it is impossible to discover much about the extent of usability problems from such small numbers. Certainly, if the intention is to run a controlled experiment and perform statistical analysis of the results, at least twice this number is recommended" [13].

2.2.4 Usability Evaluation Methods

A Usability Evaluation Method (UEM) is a procedure composed of a set of well-defined activities for collecting usage data related to end-user interaction with a software product and/or how the specific properties of this software product contribute to achieving a certain degree of usability. UEMs were formerly developed to specifically evaluate Window, Icon, Menu, Pointing device (WIMP) interfaces, which are the most representative of desktop applications [18].

Although several taxonomies for classifying UEMs have been proposed, UEMs can, in general terms, be principally classified into two different types: empirical methods and inspection methods. Empirical methods are based on capturing and analyzing usage data from real end-users. Actual end-users employ the software product (or a prototype) to complete a predefined set of tasks while the tester (human or specific software) records the outcomes of their work. Analysis of these outcomes can provide helpful information to detect usability problems during the user's task completion. Inspection methods are performed by expert evaluators or designers (i.e., they do not require the participation of real end-users). They are based on reviewing the usability aspects of Web artifacts, which are commonly user interfaces, concerning their conformance with a set of guidelines. These guidelines can range from checking the level of achievement of specific usability attributes to heuristic evaluations concerning predictions of problems related to user interfaces.

For this work, we focus on empirical evaluation.

2.3 Empirical Evaluation

Empirical software engineering is a research area concerned with the empirical observation of software engineering artifacts and the empirical validation of software engineering theories and assumptions. It involves the software product, software development process, and software management [14].

In this case, the artifact is the usability of OFLAT, and a case study (research strategy) is defined.

A case study is an empirical method to investigate contemporary phenomena in their context. Three other major research methodologies are related to case studies [45]:

1. **Survey:** collection of standardized information from a specific population, or some sample from one, usually, but not necessarily using a questionnaire or interview;
2. **Controlled experiment:** measuring the effects of manipulating one variable on another variable;
3. **Action research:** aims to influence or change some aspect of the research focus. A case study is purely observational, while action research focuses on and involves the change process.

Research methodologies can have different purposes:

- **Exploratory:** finding out what is happening, seeking new insights, and generating ideas and hypotheses for new research;
- **Descriptive:** portraying a situation or phenomenon;
- **Explanatory:** seeking an explanation of a situation or a problem, mostly but not necessarily in the form of a causal relationship;
- **Improving:** trying to improve a certain aspect of the studied phenomenon.

It can be considered that this dissertation has as purposes all of the mentioned above. Firstly, it is relevant to find what has been done to evaluate the usability within OFLAT's similar tools (exploratory) and later describe those findings (descriptive). These purposes can be achieved by realizing a Systematic Mapping Study. Also, by evaluating the usability of OFLAT, problems, and obstacles are expected to arise, as well as their explanations to arise (explanatory). Lastly, it is important to understand if and how it is possible to improve the usability of OFLAT and the evaluation process.

The data (section 2.4) collection and analysis take into account Triangulation. More than one data source is used, as well as collecting methods (qualitative and quantitative).

Last but not least comes the reporting phase. These are specific guidelines to be followed while reporting experiments in Software Engineering, similar to other disciplines such as medicine and psychology. These guidelines try to avoid problems like insufficient

reporting or heterogeneity of reporting styles. The structure of the obtained results report in this dissertation will consider guidelines found in [48], e.g., questionnaires. For this dissertation, the questionnaires will be used to understand the students' opinions about OFLAT and its usability.

There is one interesting technique from the article [52] called *Statistical Generalization*. This type of generalization is used when the results of an educational experiment tell us something about what is typically the case. It is supported by selecting a random sample of the population of interest, which gives the strongest grounds for considering results from the trial to reflect a general pattern that would be found across the wider population.

2.4 Data

To evaluate OFLAT usability, collecting user-generated data through different methods is necessary. The collected data can be categorized as qualitative or quantitative.

2.4.1 Qualitative Data

Analyzing qualitative data is a means to answer questions about characteristics, qualities, opinions, and even feelings toward the object of study. Data of this type can be obtained through interviews, observation to investigate how a certain task is conducted or its difficulty level, questionnaires, and surveys. This type of data offers a direct assessment of the usability of a design and the detection of errors, for example.

Qualitative analysis focuses on the nature of something and can be represented by themes, patterns, and stories. For example, in describing the same population, a qualitative analysis might conclude that the average person is tall, thin, and middle-aged [43].

2.4.2 Quantitative Data

Quantitative data (primarily numerical) helps to conclude whether some tasks are easy or difficult to perform by using metrics like the time that is taken until a certain task is completed, the number of clicks needed to complete a task, the number of errors, and the success rate. This helps to measure levels of users' performance.

Quantitative analysis uses numerical methods to ascertain the magnitude, amount, or size of something; for example, the attributes, behavior, or opinions of the participants [43].

Although the surveys and questionnaires are mentioned in the previous category of data, it should be emphasized that some statistical data can be taken from them, and this becomes quantitative data, such as the number of users that noticed some problem or satisfaction ratings (if it is meaningful in the context of the study).

On that note, a specific questionnaire, System Usability Scale (SUS), was developed in 1996 by John Brooke, created for measuring usability, and consists of ten questions [7].

2.4.3 System Usability Scale

The questions are given to the users after they complete a test, before any debriefing or discussion. It is also best if the respondent responds immediately to each item rather than thinking too long. If the respondent can not answer one of the questions, then the center point on the scale should be marked, as this will result in a neutral value when calculating the final score [24].

The questions are:

1. I think that I would like to use this system frequently.
2. I found the system unnecessarily complex.
3. I thought the system was easy to use.
4. I think that I would need the support of a technical person to be able to use this system.
5. I found the various functions in this system were well integrated.
6. I thought there was too much inconsistency in this system.
7. I would imagine that most people would learn to use this system very quickly.
8. I found the system very cumbersome to use.
9. I felt very confident using the system.
10. I needed to learn a lot of things before I could get going with this system.

To avoid biased responses, the alternation between positive and negative (odd and even questions, respectively) statements causes users to require attention. The answer format is a scale from 1 (strongly disagree) to 5 (strongly agree), representing the level of agreement or disagreement with the statements.

The final SUS result is a number that represents a measure of the system's usability. Five should be subtracted from the sum of user responses to odd statements to calculate the final score. Then subtract from 25 the sum of even statements. In the end, the previous results are added and multiplied by 2.5, so the range is between 0 and 100.

- X = Sum of the points for all odd-numbered questions - 5
- Y = 25 - Sum of the points for all even-numbered questions
- SUS Score = (X + Y) x 2.5

If the final score is above 80.3, the system's usability is evaluated as "excellent". If it is between 68 and 80.3, then it is "good". If it is 68, then it is considered "OK". Between 51 and 67 is "poor", and below 51 is "awful".

It can also be interesting to generate box plots [6] to visualize results.

This questionnaire is particularly interesting if used to compare different versions of the same system, so as OFLAT is expected to change in the future, it is a method to be considered.

2.4.4 NASA TLX

The Official NASA Task Load Index (TLX) [34] assesses users' workload operating with systems/applications.

NASA TLX score is based on a weighted average of ratings on six multi-dimensional topics. Each topic has a question, and the users should attribute a value from 1 to 20. The topics are:

1. Mental Demand;
2. Physical Demand;
3. Temporal Demand;
4. Performance;
5. Effort;
6. Frustration.

The questionnaire is represented in Figure 2.2.

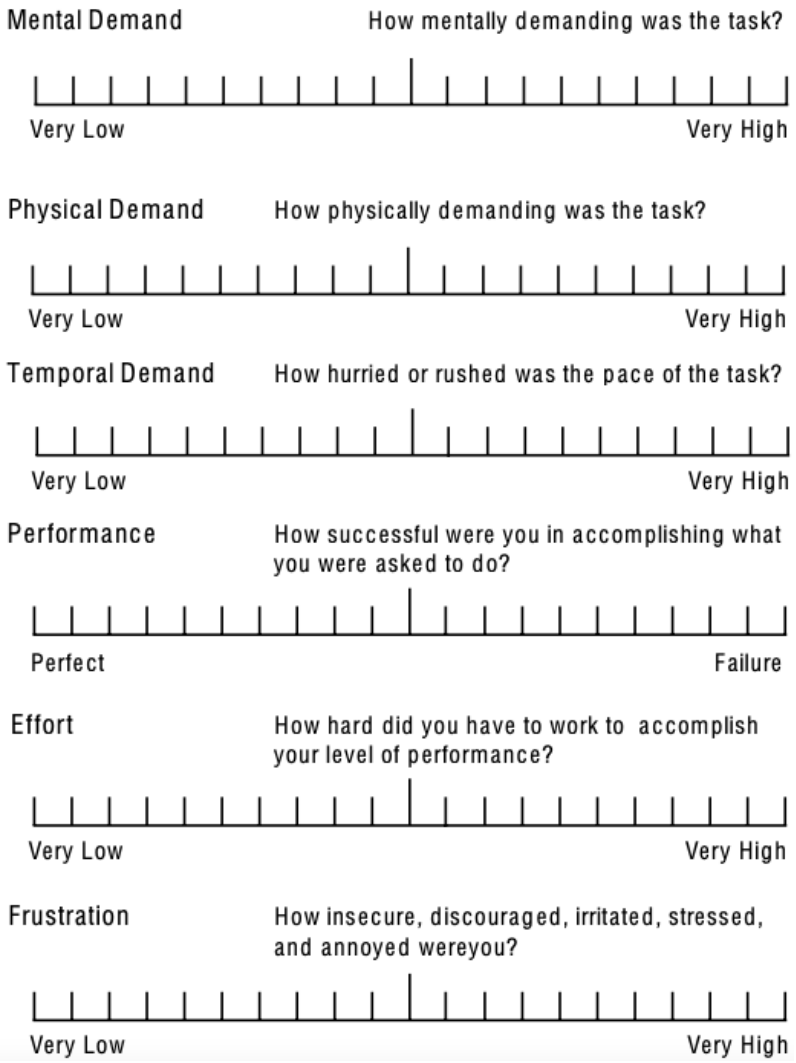


Figure 2.2: NASA TLX questionnaire

2.4.5 Event tracking and Logs

One way to obtain results through the direct use of the OFLAT platform is through implementing strategies that allow tracking of users' actions (logs). Although OFLAT does not have login options, the identification and tracking of users, consequently, their sessions can be obtained through the utilization of solutions that are made available by tools such as Segment (section 5.1).

Each generated event is associated with a user, a time when it was created, and a type that describes what was done on the platform. This is useful because it allows generating data related to metrics such as the number of clicks until the user reaches a goal, the number of clicks on each button, the average time spent on each task or total session, and so on.

Event tracking does not replace the realization of interviews and sessions made with users in person. However, it is an excellent alternative data source for circumstances

where getting volunteers to test the platform is difficult. Identifying workflows and bottlenecks during usage is more efficient when there is direct contact with testers. Still, it can also be facilitated through data analysis strategies such as Process Mining.

2.5 Process Mining

Process mining is a set of process-centric techniques that aggregate data science and process management fields to support the analysis of processes based on event logs. Event data is used to show what people, machines, and organizations are operating, addressing performance and compliance problems of operational processes and revealing bottlenecks and other areas that can be improved [1].

It helps answer the following questions [10]:

- What is the process that people follow?
- Where are the bottlenecks in the process?
- Where do people (or machines) deviate from the expected or idealized process?
- What factors are influencing a bottleneck?
- Can problems be predicted (delay, deviation, risk, etc.) for running cases?
- Can countermeasures be recommended?
- How to redesign the process/machine/organization?

There are different types of process mining (see Figure 2.3) [25]:

- "**Discovery:** Process discovery uses event log data to create a process model without outside influence. Under this classification, no previous process models would exist to inform the development of a new process model. This type of process mining is the most widely adopted;
- **Conformance:** Conformance checking confirms if the intended process model is reflected in practice. This type of process mining compares a process description to an existing process model based on its event log data, identifying any deviations from the intended model;
- **Enhancement:** This type of process mining has also been referred to as extension, organizational mining, or performance mining. In this class of process mining, additional information is used to improve an existing process model. For example, the output of conformance checking can assist in identifying bottlenecks within a process model, allowing managers to optimize an existing process."

Its importance lies in the identification of inefficiency in operational models and allows to verify that the system is running as expected. The bottlenecks discovery helps improve the process, reducing costs and driving more quality to the users.

So, to apply these techniques to OFLAT, the starting point is to generate event data and aggregate it in a table where each row is an event (id, timestamp, activity, data, for example). Then, by evaluating the cases that make sense to be accepted, a model can be created (discovery type). In the future, that model can be updated if the event log data differs from what is supposed to be according to the first defined model (enhancement type).

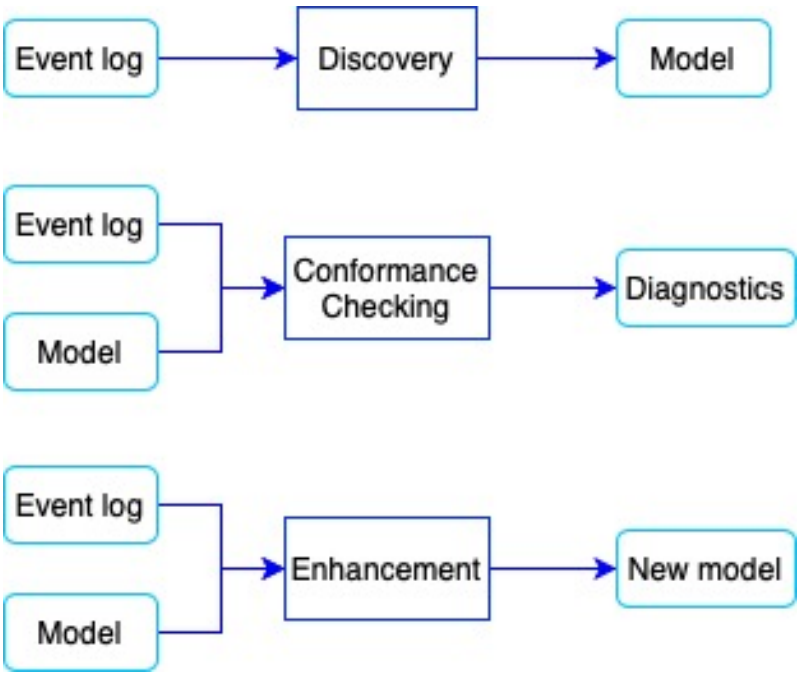


Figure 2.3: Process Mining Types - taken from IBM [25]

2.6 Summary

After reading this chapter, the main concepts are clearer, so it is possible to understand the objectives of this dissertation better.

The usability evaluation intended for this dissertation recurs to empirical evaluation methods, especially using a controlled experiment that needs to be designed and implemented. The data should be qualitative and quantitative, collected through questionnaires like SUS.

3 | Related Work

In this chapter, the main goal is to explain and summarize all the work that has been done related to the theme of this dissertation.

OFLAT and reported work on its usability is approached, a Systematic Mapping Study is conducted step by step, and the results are presented. To conclude this research, it is presented similar tools to OFLAT to better understand each tool's functionalities and eventual usability evaluation.

3.1 OFLAT description

OFLAT is a tool that was created to help students understand automata theory, algorithms, regular expressions, and context-free grammar.

This tool supports the following functionalities:

- Creation of a new model. This model can be one of the following: Finite Automata, Regular Expressions, and Context-Free Grammars;
- Edit the visible model;
- Fit visible graphics to the view box;
- Generate words with size X, i.e., gives a list of words, of size smaller or equal to a given size X, accepted by the automaton or regular expression;
- Test word, i.e., shows, with an animation, how the automaton accepts or rejects a given word; or shows, with a tree, the acceptance process of a regular expression; the input is asked through a box on the screen;
- Step-by-step word acceptance, i.e., allows the user to see the acceptance process step-by-step; the input is asked through a box on the screen;
- Convert from automaton to regular expression and vice versa;
- Import/Export model;
- Predefined examples.

Pushdown Automata, Turing machines, and Pumping Lemma are the new functionalities that will be available soon.

The work presented in [21] concerns the implementation of pushdown automata functionality in OFLAT. In this study, the "Accept" operation is the main operation of the application that allows one to see the behavior of the automaton throughout the acceptance, showing its setting at each step. With the introduction of pushdown automata, it was necessary to generalize part of the already existing functionalities to be usable by the new category of automata.

The study presented in [31] concerns the implementation of Turing machines in OFLAT. It shows how the Turing machines functionality will be integrated into OFLAT.

In terms of technologies used for the implementation, this tool is implemented in OCaml and JavaScript and uses the Ocsigen framework for the creation of its web page.

3.2 OFLAT and Usability

As previously mentioned, the work presented in [32] concerns the OFLAT web application development. It reports the testing process, both at the programming level in search of bugs and errors and at the level of user tests. Since the focus of that work was on covering functionality there was no significant systematic testing with real users. The experiment sessions encompassed a group of 16 people, some of whom were experts in the field and others not, but all with Computer Science backgrounds.

The biggest advantages of the presented evaluation sessions were that several usability problems were already detected, thus being possible to start understanding the limitations of the use of the application, as well as the presentation of suggestions by the users regarding the creation of new features that allow better and more complete usage of OFLAT.

In detail, the usability problems reported were related to the fact that some of the page's usages were not as easy as initially thought. Some users had difficulties finding the input box, and some buttons were not readable because of the text color. These two problems were solved by identifying the input box and changing the text color. Some users did not understand when there was a need to input text before clicking a button. For now, this problem has been addressed with an alert box that indicates the necessary information and format. Other users did not understand how to start working with a newly generated automaton. These sessions led to the creation of an instructions page to help users.

After the reported work, OFLAT has undergone several iterations and is currently in version 2.1. The application now covers new topics and features like creating automata, regular expressions, context-free grammar, step-by-step word acceptance, conversions, generation of words with size X, test words, import/export models, some predefined examples, and exercises.

As said, despite those reported evaluation sessions being essential for a quick jump-start on improving the implementation, it lacked a systematic approach and proper control of the variables in the experiment. Besides, there is no longer available raw data from those experiments (not a good practice), and the results do not include a presentation of gathered metrics, etc. Through an experimental evaluation and by using quantitative and qualitative metrics, more complete results are expected with our current work.

3.3 Systematic Mapping Study

Performing a Systematic Literature Review, by considering the guidelines in [29], is a great way to collect and summarize the related work and to identify gaps in this research about usability evaluation of educational FLAT platforms. Despite this intent, "if, during the initial examination of a domain before commissioning a systematic review, it is discovered that very little evidence is likely to exist or that the topic is very broad then a Systematic Mapping Study may be a more appropriate exercise than a systematic review".

After the introductory research done during the contextualization phase of the theme of this dissertation, it was expected to find little related evidence. A systematic mapping study is then more appropriate and the guidelines mentioned in [37] are still considered.

The following three stages (planning, conducting, and reporting) of this research method include the definition of the research question, the identification of the search strategy, the selection of primary studies, quality assessment, data extraction, and synthesis, and, finally, the results.

3.3.1 Planning

In the planning stage, the research question is defined. In this dissertation context, the main question is *What is the best way to evaluate OFLAT's usability?*. This main question can be divided into sub-questions like *What are the educational tools similar to OFLAT?*, *Is the usability of educational tools FLAT-related evaluated?* and *How is the usability of educational tools FLAT-related evaluated?*

To generalize and expand the context of the research, these questions need to be more comprehensive, but it is not wanted a big generalization because the Theory of Computation support tools has specific characteristics. They differ from other applications in general because they seek to cover the theory that is taught in classes and have as their mission the teaching of algorithms. They are not intended to be robust, ready for large-scale use, or allow automatons with thousands of states. The focus is on the experimental use of restricted groups of students who want support in understanding the course algorithms.

3.3.1.1 Research question

How to evaluate the usability of tools to support Theory of Computation education? can be the main research question and as sub-questions: *Are there any educational tools for teaching and training formal languages and automata theory in Theory of Computation courses?*, *Which are the educational tools for teaching and training formal languages and automata theory in Theory of Computation courses?*, *Are there any reported usability studies for tools for teaching and training formal languages and automata theory in Theory of Computation courses?*, *What techniques are used for the implantation of the reported usability studies of tools for teaching and training formal languages and automata theory in Theory of Computation courses?*.

Table 3.1 shows these research sub-questions along with their motivation.

Research sub-questions	Motivation
Q1. Are there any FLAT teaching tools?	To discover if there are similar tools to OFLAT.
Q2. Which are those tools?	To discover OFLAT similar tools and in what do they complete directly with
Q3. Do the tools have reported usability studies?	To discover if there is a gap in this kind of studies
Q4. What techniques are used in those studies?	To evaluate, find most frequently employed methods and consider the replication of those techniques applied to OFLAT

Table 3.1: Research sub-questions

According to [49], the research question was structured by following the PICOC criteria:

- Population: Educational tools FLAT-related;
- Intervention: Usability evaluation methods;
- Comparison: Different usability evaluation methods;
- Outcome: Effectiveness of the usability evaluation methods;
- Context: Research papers.

3.3.1.2 Search strategy

This stage also consists of applying the search strategy, defining criteria for including or excluding studies, quality assessment, complete data extraction, and synthesis.

During the search strategy phase, a search string is applied and includes concepts and alternative terms/synonyms from the research question and sub-questions, resulting in *("Usability") AND ("Educational" OR "Education" OR "Teaching" OR "Teach") AND*

("Automata Theory" OR "Theory of Computation") AND ("Tools"). The chosen electronic sources of relevance to Software Engineers according to [29] were IEEExplore and ACM Digital Library.

After all the phases are completed and once the studies are selected in the first iteration, it can be considered to apply a snowballing technique to expand the selected studies [56].

3.3.1.3 Selection of primary studies

Before running the search string on IEEExplore and ACM, a selection of primary studies is defined to avoid bias while selecting the relevant search results. These are the defined inclusion criteria and are in line with [17]:

- The terms are researched for their appearance in papers' abstracts. As these are the most important topics in the research, they should appear in abstracts;
- Papers presenting surveys, case studies, or experiments concerning the empirical validation of usability evaluation methods. These kinds of studies are the most representative ones to gather empirical data;
- Papers comparing the effectiveness of two or more usability evaluation methods. This kind of study is selected since comparisons among UEMs allow empirical data aggregation from different sources, avoiding Triangulation;
- Studies that are applied to the teaching domain.

It is known that the 1993 publication of Nielsen and Landauer [35] was an important milestone in the history of interface design and usability. Therefore, any source published before this year is excluded from the search. Old studies that report tools that do not exist anymore are also excluded. This is the exclusion criteria.

3.3.1.4 Quality assessment

The quality assessment usually involves the design of a questionnaire to point out the importance of each selected article and thus reach a point of agreement between different authors of a study. However, having this project only one author, it was decided to consider the number of times an article was cited by others as a way to ensure the quality of the research.

3.3.1.5 Data extraction strategy

The employed data extraction strategy was based on providing the set of possible answers for each research sub-question that had been defined. The possible answers to each research sub-question are explained in more detail as follows.

Regarding Q1 (Are there any FLAT teaching tools?), the tool of the paper can be classified into one of the following answers:

- Web/Mobile: if the tool is developed to work on web and or mobile devices;
- Desktop: if the tool is implemented to work on the desktop.

Regarding Q2 (Which are those tools?), a paper can be classified into one of the following answers:

- Similar to OFLAT: Covers at least two of these topics - create automata, regular expressions, context-free grammar, step-by-step word acceptance, exercises, and conversions;
- Not that similar: Covers only one topic of the mentioned above or does not have an interactive graphical interface.

Regarding Q3 (Do the tools have reported usability studies?), a paper can be classified into one of the following answers:

- Usability evaluated: if reports any kind of surveys or experiments;
- Did not report any study: if there are no usability evaluations.

Finally, regarding Q4 (What techniques are used in those studies?), a paper can be classified into one of the following answers:

- Empirical: if evaluation is based on capturing and analyzing usage data from real end-users;
- Inspection: if evaluation is performed by expert evaluators or designers.

3.3.1.6 Synthesis methods

The applied synthesis methods are quantitative since it was based on counting the primary studies that are classified in each answer from the research sub-questions and qualitative, for trying to include several representative studies for each sub-question, by considering the results from the quality assessment.

3.3.2 Conducting

At this stage, a table that relates the sources to the number of potential results and the number of selected studies is made. Cases in which results appear more than once in different sources are only taken into account once. See Table 3.2.

Source	Potential studies	Studies 1st iteration	Selected after snowballing
IEEExplore	1	1	1
ACM Digital Library	462	2	12
Total	463	3	13

Table 3.2: Conducting stage results

3.3.3 Reporting

Of the selected studies, there are two of them that gather information about more than one tool, which is why the questions can have more than one possible answer. This is reflected in the sum of the number of studies being different from the total number of selected studies, as well as the sums of the percentages being different from 100%, in the results shown in Table 3.3. The list of selected studies can be found in Appendix A and their mapping in Appendix B.

Research sub-questions	Possible answers	Number of studies	Percentage (%)
Q1. Are there any FLAT teaching tools?	-Web/mobile	3	23.1
	-Desktop	12	92.31
Q2. Which are those tools?	-Similar to OFLAT	11	84.62
	-Not similar	3	23.1
Q3. Do the tools have reported usability studies?	-Usability evaluated	9	69.23
	-Not reported	5	38.46
Q4. What techniques are used in those studies?	-Empirical	9	69.23
	-Inspection	0	0

Table 3.3: Systematic mapping study results

The following subsections describe the results applied to each sub-question to answer them.

3.3.3.1 Q1 - Are there any FLAT teaching tools?

All of the selected studies mention the existence of FLAT teaching tools. The study **S01** is a review of many articles that mention those tools and it ended up being a great source of studies through the snowballing research method. The studies **S05** and **S06** are also a compilation of tools. The web/mobile tools presented are described in studies **S03** and **S06**. The remaining studies include desktop tools.

3.3.3.2 Q2 - Which are those tools?

The tools selected as similar were: JFLAP (mentioned in **S02, S08** and also included in the comparison of the similar tools in section 3.4), ILSA Environment and its Finite State Machine Simulator (mentioned in **S03**), PDA - Pushdown Automata Simulator (in

S04 and **S06**), FSM Simulator (**S05** and **S11**), TM Simulator (**S05, S06** and **S11**), Finite automata simulator (**S06**), JPT (**S07**), Toolkit Language Emulator (**S09**) and SELFA (**S12**).

The jFAST finite automata simulator (**S10**) creates automata. Still, it does not include any other feature from the ones defined in the criteria, so it is not considered a similar tool as well. The tool described in study **S13** is not considered a similar tool since the Compiler-Based Toolkit does not have an interactive graphical interface.

3.3.3.3 Q3 - Do the tools have reported usability studies?

The studies that report evaluations or experiments/tests that involve users are the studies **S03, S05, S06, S09, S10, S11, S12** and **S13**.

It is essential to mention that none of those studies contain raw data relating to questionnaires or data used in experiments, such as exercises requested by users. Only results and, in some cases, the questions asked to users are reported.

3.3.3.4 Q4 - What techniques are used in those studies?

All of the mentioned studies in the previous question reported empirical evaluations which can be described as follows:

- **S03** - 36 users were asked to evaluate the system in terms of user experience and rate by their significance through questionnaires;
- **S05** - The first experiment evaluates the improvement in the students' motivation (52 students) through a questionnaire before and after visual examples are shown. An improvement in their motivation was noticed. The second experiment evaluates the effectiveness of using the tools on the students' performance (69 students). Groups of work having the same environment and conditions were challenged to complete 10 exercises, different for each group and randomly assigned. Two groups used tools and the other two groups did not use them. "The result showed a better performance for the two groups using the integrated environment. Then, the experiment was repeated by redistributing the exercises among the four groups. Again, the two groups with the integrated environment showed better performance";
- **S06** - The evaluation is very similar to the second experiment previously described for study **S05**. The two groups using the simulators showed a better performance with also one rotation of exercises. Students were then asked to evaluate the simulators based on their design, user-friendliness, functionality, usability, and on how they can help throughout the automata learning process. With the help of a 1 (worst) to 5 (best) scale, the features had an average evaluation of 4.4, 4.1, 3.9, 3.8, and 4.0, correspondingly;
- **S09** - Students evaluated the toolkit and nearly 95% of the students answered that Language Emulator was helpful in the learning process of Automata Theory. They

have also given some suggestions in order to improve the toolkit as well as its capability to help in the teaching process;

- **S10** - 18 college students enrolled in a junior-level computer science course were asked to explore the jFAST software after a brief introduction. Students then completed a questionnaire describing their prior experience with FLA, their experiences in Computer Theory classes, and their reactions to jFAST.

- **S11** - In an automata classroom, students were divided into four groups, each group containing 20 students. A set of 40 randomly selected exercises was distributed among the groups, 10 for each group. Each member of the group could collaborate inside their group but not with any other group members. No group could see the exercises of another group. Two groups were asked to answer their assigned exercises using the tools and the other two groups were without using them. An equal time period was provided to all the groups. The results showed a better performance for the two groups using the tools. Then, the experiment was repeated by redistributing the exercises among the four groups. Again, the two groups using the tools showed better performance.

- **S12** - Was created a questionnaire for the SELFA users to ascertain whether they consider the tool to be useful by them in studying the concepts and algorithms of FLAT. The ease of use of the tool was also included. Two questions were formulated to measure those aspects: "How useful do you find the tool?" and "How easy is to use?". The students who used SELFA were asked to answer anonymously. Respondents could choose among five possibilities to the first question: "Not Useful" (1), "Not very Useful" (2), "Useful" (3), "Very Useful" (4), and "Extremely Useful" (5); and other 5 to the second one: "Very Difficult" (1), "Not Easy" (2), "Easy" (3), "Quite Easy" (4) and "Very Easy" (5). Students could only choose one answer. The students found the tool very useful and quite or easy to use. On average, students rated the tool usefulness as 4.4 and 4.3 and the ease-of-use feature as 4.3 and 3.85 on a five-scale in the academic years 2005-2006 and 2006-2007;

- **S13** - supervised experiment. Fifty-three students who studied at least one course on automata theory were given exercises to compile and study automatons' behavior. Results reflect the student's feelings towards the tool. They said it was interesting as a teaching tool, and the source language is easy to learn. Results in a better understanding of finite automata.

3.3.4 Conclusions

Many of the considered similar tools are not currently available or open to general users (only to specific domains, like universities). To conclude, JFLAP is definitely the most relevant tool.

The most used technique is the inquisition of opinions and classifications through questionnaires. There are three examples of experiments that imply the observation of users by a supervisor. None of the studies used inspection techniques.

3.4 Similar Tools

There are some similar tools to OFLAT when it comes to the content covered. The four existing tools that complete most directly with OFLAT were chosen and each tool is presented below. Its main features are directly compared in Table 3.4

	OFLAT	ATv3	A. S.	Flap.js	JFLAP
Web	✓	✓	✓	✓	X
Mobile	X	✓	X	✓	X
Desktop	X	X	X	X	✓
Create automata	✓	✓	✓	✓	✓
Regular Expressions	✓	✓	X	✓	✓
Context-free grammar	✓	✓	X	X	✓
Turing Machines	✓	X	X	X	✓
Pushdown Automata	✓	X	✓	X	✓
Pumping Lemma	✓	X	X	X	✓
Step-by-step word acceptance	✓	X	✓	X	✓
Exercises	✓	✓	X	X	✓
Conversions	✓	X	X	✓	✓
Usability Evaluation	...	SUS and surveys	X	X	surveys and experiments

Table 3.4: Features comparison between similar tools. Extended table of the original found in [41]

3.4.1 Automata Tutor v3.0

Automata Tutor v3.0 [3] is an application focused on resolving exercises. It allows the registration and login of two types of users: teachers and students. Teachers are allowed to create courses and exercises in them, as well as visualize students' classifications. Students are allowed to take courses and do exercises.

The exercises focus on creating Regular Expressions, Finite Automata, and Context-Free Grammar. When a student submits a solution to an exercise, the corresponding grade and some feedback are received, allowing him to realize what was done wrong and what should be corrected. Despite being a complete application for exercise resolution, it does not allow the user to see step-by-step resolutions or conversions, which can affect the user's understanding of algorithms, for example.

Regarding usability tests, it can be read in chapter 5 of the paper "Automata Tutor v3*" [12] that the tool was mandatory to use by about 950 students during the realization of various homework. At the end of the course, an anonymous SUS survey was conducted, which allowed the conclusion of a positive result regarding the speed of learning of the tool usage, by not requiring assistance when using the tool, the confidence felt during use, ease of use, and satisfaction. The students also revealed that they understood more after using the tool, the feedback was helpful, they felt better prepared for the final exam, and that the use of the tool is their favorite way of learning when compared to theoretical class assistance, resolution of written exercises, programming, individual autonomous learning and group learning.

3.4.2 Automaton Simulator

Automaton Simulator [4] is a website that allows the creation of automata, and it tests word acceptance. It is not well-formatted to use on mobile devices. By using colors, the animations are straightforward to see the steps that are taken for the word to be accepted or not. This application does not have conversions, so it does not do any other type of animation. It does not have exercises or resolutions.

It was not reported any usability evaluation so far.

3.4.3 Flap.js

Flap.js [19] is a web application of easy access to all types of devices, including mobile devices. The user can graphically create Finite Automata, Regular Expressions, and Pushdown Automata. It allows tests and conversions (NFA to DFA, DFA to NFA, and RE to NFA) and optimizations, but none of these use animations, which would improve the learning process.

It shows if the word is accepted or not, and, in the conversions and optimizations, it gives the new transformations in substitution of the original one. This means the user cannot compare results to understand the transformation, which is a downside.

In the "Changelog" section of the Flap.js Github page, it is only possible to read about unit tests, but it is never mentioned any usability evaluation.

3.4.4 JFLAP

JFLAP is "a package of graphical tools which can aid in learning the basic concepts of Formal Languages and Automata Theory" [28]. It is the most complete application, but since it is for desktops, it does not offer easy access to all devices. It allows the study of finite automata, Turing machines, tests, conversions, and minimizations. Each state is highlighted when testing the acceptance of a word in an automaton, thus facilitating the understanding of each symbol processed step by step.

This application has support on its website, where tutorials, exercise resolutions, a book that explains the algorithms, and a lot of helpful documentation for the user are available.

However, this documentation does not contain information regarding any usability evaluation done to the application by the developers. There are some mentions of papers where studies have been done on it.

3.4.4.1 Increasing Engagement in Automata Theory With JFLAP

From the paper "Increasing Engagement in Automata Theory With JFLAP" [42], written in 2009, we know that JFLAP was used worldwide in 161 countries and has had over 64000 downloads from January 2004 to 2009. There was feedback from students that helped to improve the app. During 2005-2006 and 2006-2007, there was a formal study with 14 universities participating in evaluating JFLAP's effectiveness in learning automata theory topics.

The study was based on collecting pre- and post-knowledge test data and a survey of computer science attitudes, JFLAP implementation, and usability. The results of both years were compared, but the second year's test and survey changed, taking into account the feedback from the previous year. It is important to note that:

- During the first year, there was no control group;
- One university pre-tested the students, another one post-tested, and a third one pre and post-tested;
- The control group of the second year was not statistically significant.

The results show that "students had an easy time using JFLAP and thought it was a good tool", "one-third of the students used JFLAP 21% or more of study time for exams, and 29% used JFLAP to work additional problems", "the majority of students felt JFLAP made them more engaged and made learning concepts easier". "The attitude survey showed that the population sample was slightly less confident in their abilities. The pretest and post-test again showed that students provided significantly more correct answers on post-test".

3.4.4.2 Enhancing the learning ability using JFLAP for Theory of Computation Course

Ten years later, in 2017, another paper studies how to "Enhancing the learning ability using JFLAP for Theory of Computation Course" [57]. This study also evaluates the performance of students of two different years. The students' results are worse in the first year, and JFLAP is not used. In the second year, the results are better, and the authors associate this phenomenon with using JFLAP in that year. It should be noted that several factors might affect this change in the course results since students are different and

may have different levels of skill/interest in those topics, different test levels of difficulty might have been proposed by the teachers, etc. However, it is another reference to take into account.

It can be read in the sixth chapter about the results: "Student's feedback is also collected where students agreed that their concentration while studying this subject and interest in the subject improves. The use of the JFLAP tool improves the overall result. It also enhances learning ability by improving problem solubility and understandability among the students. To collect students' feedback on the effect of using the JFLAP tool, a survey questionnaire was designed with questions based on the efficiency of the tool and its ease of use.

Students acknowledged with the answers to the questions. This was a five-point Likert scale type with ten questions. The scales varied from strongly agree to disagree strongly. Questions were designed to determine whether the tool has helped students improve their problem-solving ability".

3.4.4.3 Making Turing Machines Accessible to Blind Students

There is another paper (2012) that reflects the need to adapt the TM (Turing Machine) simulator to be used by a blind student [11]. It can be found a brief survey that evaluates the usefulness of using the Turing Machine simulator (2007 and 2008 results). The survey checked the level of the students' agreement with the following statements:

- The simulator is useful for following the lectures;
- The simulator is useful for studying at home;
- A theoretical computer science course must use a TM simulator.

It is concluded that "almost all students thought that using a TM simulator is useful and that such a simulator should be adopted by the teacher of a theoretical computer science course".

3.5 Summary

According to the Systematic Mapping Study results of the presented features, JFLAP is the most relevant tool, but the tool considered during the following controlled experiment is OFLAT.

Before designing the controlled experiment, it is relevant to understand a little bit better the OFLAT tool from the Theory of Computation course students' point of view and its developers' concerns. As such, engineering OFLAT with usability concerns is presented in the next chapter.

4 Engineering OFLAT with usability concerns

4.1 Usability experiments process diagram

It was made the first usability study when OFLAT released its first version (version 1.3). After the usability study had been carried out, the improvements were developed for the new version of OFLAT.

The "First usability study" represents the experiments made in the previous usability study [41].

After the release of the new version of OFLAT (version 2.1), a recent usability study was made to see if there were new usability problems that were not detected in the first usability study.

The "Second usability study" represents the current usability study, and it will be explained further in chapter 6 and chapter 7.

After the conclusion of the second usability study, new improvement suggestions will be developed for the future version of OFLAT (see Figure 4.1).

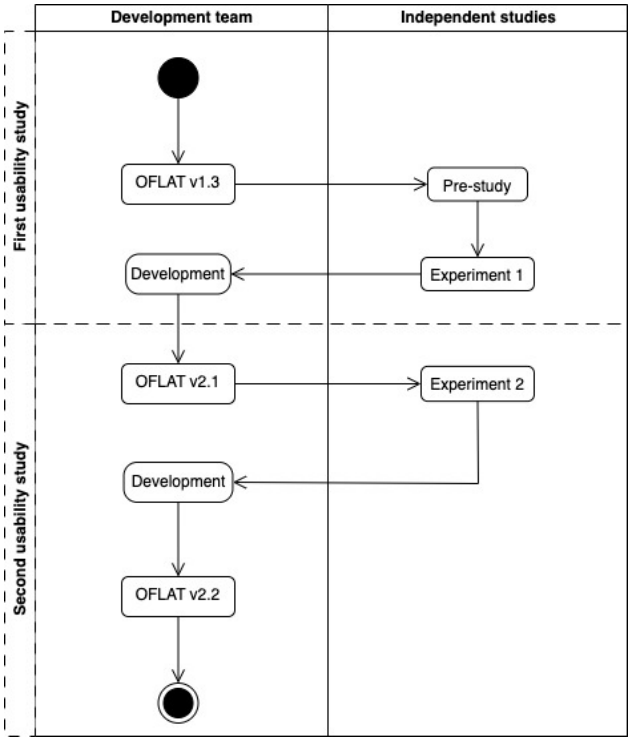


Figure 4.1: Usability experiments process diagram

4.2 OFLAT version 1.3

The following functionalities were supported in OFLAT version 1.3, before the first usability study:

- Creation of a new model. This model could be Finite Automata or Regular Expression;
- Fit visible graphics to the view box;
- Generate words with size X, i.e., give a list of words of size smaller or equal to a given size X, accepted by the automaton or regular expression;
- Test word, i.e., shows, with an animation, how the automaton accepts or rejects a given word; or shows, with a tree, the acceptance process of a regular expression; the input is asked through a box on the screen;
- Step-by-step word acceptance, i.e., allows the user to see the acceptance process step-by-step; the input is asked through a box on the screen;
- Convert from automaton to regular expression and vice versa;
- Import model;
- Predefined examples.

4.3 First Survey - OFLAT Usability Evaluation

The subject of the Theory of Computation occurs in the second semester of NOVA's School of Science and Technology Computer Science course. Therefore, to collect data on the students' impressions of the tool, it was made the first questionnaire.

The survey was developed in order to answer OFLAT developers' concerns. The topics pointed out as worrying were the identification of possible blockages or bottlenecks felt by users and trying to realize if the platform was intuitive or if it took a great initial effort to use it.

It is important to mention that OFLAT usage is not mandatory in the course. Its use in lab classes is only suggested.

The first survey was conducted.

The questions are presented in subsection 4.3.1, and the results are presented in subsection 4.3.2, respectively.

4.3.1 First Survey - Questions

The first survey questions were taken from [41].

OFLAT - Usability evaluation

OFLAT (http://ctp.di.fct.unl.pt/FACTOR/OFLAT/) is a tool that aims to extend and consolidate teaching materials of the Theory of Computation course. This tool helps to have an interactive environment supporting the definition and testing of language generators and language recognizers. (Source: https://release.di.ubi.pt/factor/oflatocsigen.html).

Your participation is very important, in order to conduct a study on OFLAT usability.
All the answers are anonymous, the data will only be used for the purpose of OFLAT evaluation and it will not be used by third parties.
This survey takes around 10 minutes to complete.

The answers can be given in Portuguese.

Thank you for your time!

*Obrigatório

Theory of Computation course

These questions help to understand the user's background related to the studied subjects.

1. In each year did you attend the Theory of Computation course? (Please select every year in which you were enrolled) *

Marcar tudo o que for aplicável.

- [ ] 2020/21
- [ ] 2019/20
- [ ] 2018/19
- [ ] 2017/18
- [ ] Other
- [ ] Does not apply

12/09/2021 OFLAT - Usability evaluation

OFLAT

About the tool (http://ctp.di.fct.unl.pt/FACTOR/OFLAT/)

2. Have you ever used OFLAT? *

Marcar apenas uma oval.

- ( ) Yes Avançar para a pergunta 3
- ( ) No Avançar para a pergunta 24

OFLAT

About the tool (http://ctp.di.fct.unl.pt/FACTOR/OFLAT/)

3. In which browser(s) did you use it? *

Marcar tudo o que for aplicável.

- [ ] Google Chrome
- [ ] Safari
- [ ] Firefox
- [ ] Microsoft Edge

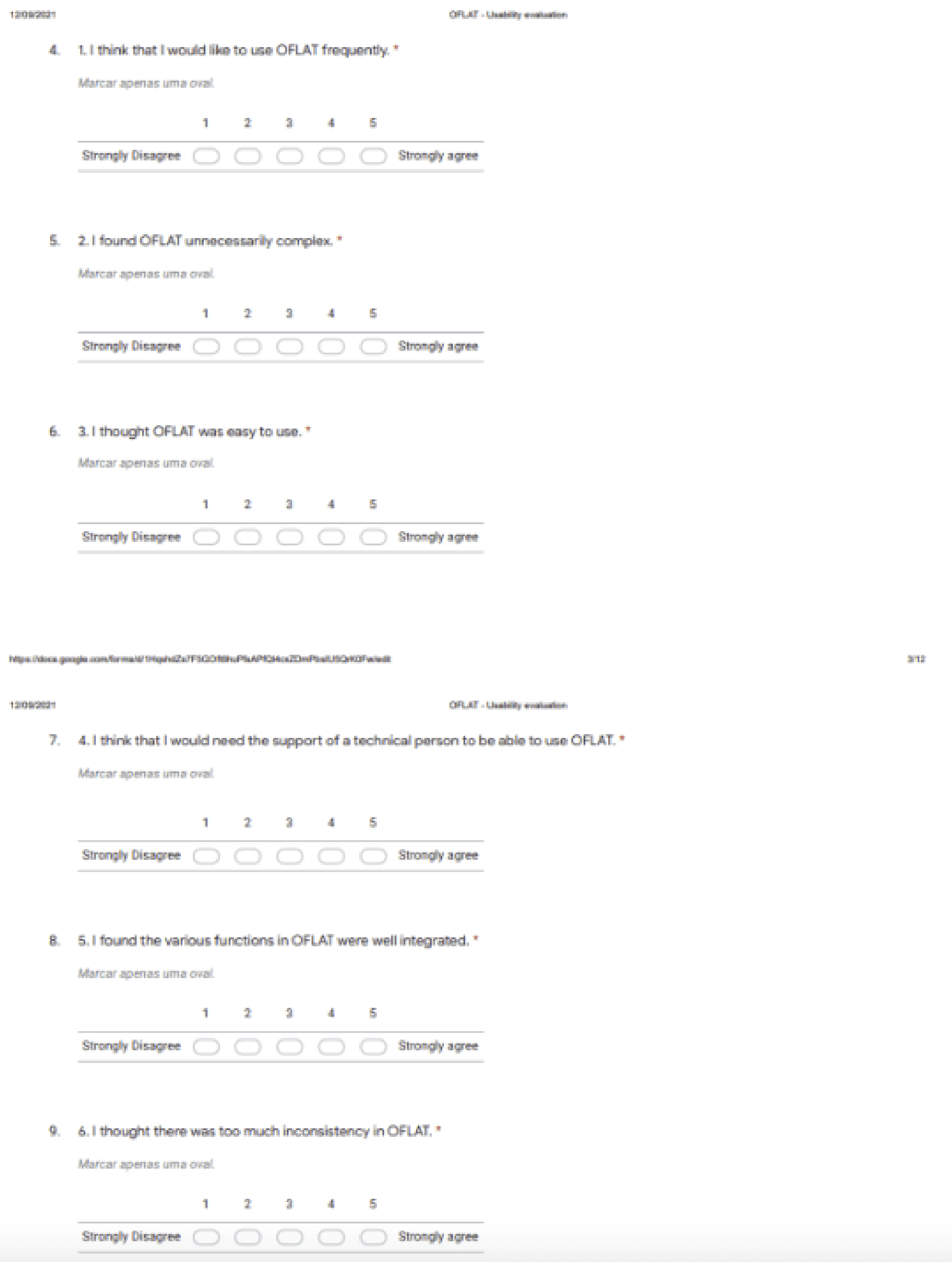
Outra: [ ]

OFLAT Usability

For the following questions, please consider the scale from 1 (strongly disagree) to 5 (strongly agree).

Figure 4.2: Survey 1 - page 1. Taken from [41]

According to the user profile, the questions in Figure 4.2 asked individuals about their participation in the Theory of Computation course, if they had ever used OFLAT, and which browser they had used.


12/09/2021 OFLAT - Usability evaluation

4. 1. I think that I would like to use OFLAT frequently. *

Marcar apenas uma oval.

	1	2	3	4	5	
Strongly Disagree	◯	◯	◯	◯	◯	Strongly agree

5. 2. I found OFLAT unnecessarily complex. *

Marcar apenas uma oval.

	1	2	3	4	5	
Strongly Disagree	◯	◯	◯	◯	◯	Strongly agree

6. 3. I thought OFLAT was easy to use. *

Marcar apenas uma oval.

	1	2	3	4	5	
Strongly Disagree	◯	◯	◯	◯	◯	Strongly agree

3/12

12/09/2021 OFLAT - Usability evaluation

7. 4. I think that I would need the support of a technical person to be able to use OFLAT. *

Marcar apenas uma oval.

	1	2	3	4	5	
Strongly Disagree	◯	◯	◯	◯	◯	Strongly agree

8. 5. I found the various functions in OFLAT were well integrated. *

Marcar apenas uma oval.

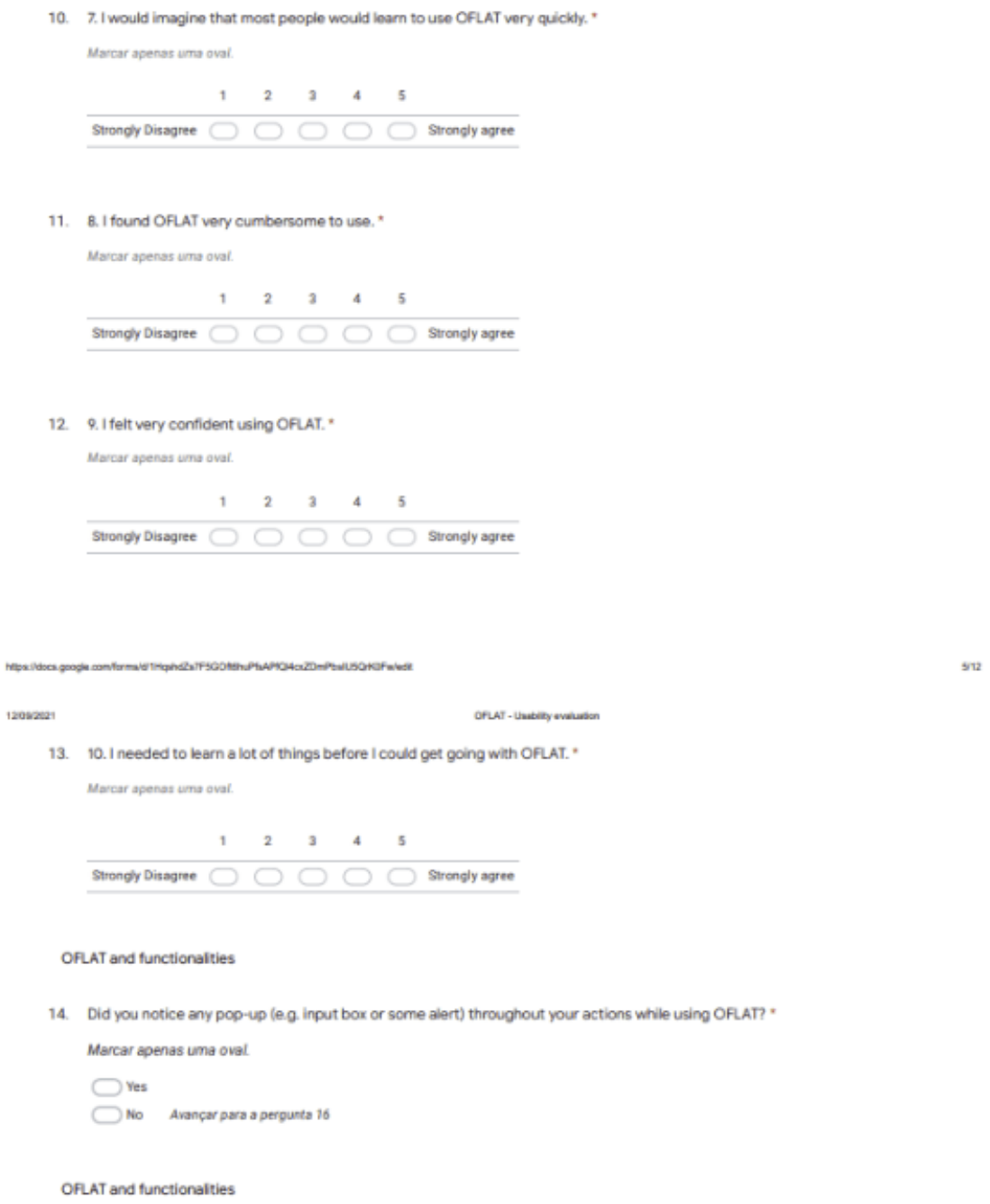
	1	2	3	4	5	
Strongly Disagree	◯	◯	◯	◯	◯	Strongly agree

9. 6. I thought there was too much inconsistency in OFLAT. *

Marcar apenas uma oval.

	1	2	3	4	5	
Strongly Disagree	◯	◯	◯	◯	◯	Strongly agree

Figure 4.3: Survey 1 - page 2. Taken from [41]



10. 7. I would imagine that most people would learn to use OFLAT very quickly. *

Marcar apenas uma oval.

	1	2	3	4	5	
Strongly Disagree	◯	◯	◯	◯	◯	Strongly agree

11. 8. I found OFLAT very cumbersome to use. *

Marcar apenas uma oval.

	1	2	3	4	5	
Strongly Disagree	◯	◯	◯	◯	◯	Strongly agree

12. 9. I felt very confident using OFLAT. *

Marcar apenas uma oval.

	1	2	3	4	5	
Strongly Disagree	◯	◯	◯	◯	◯	Strongly agree

https://docs.google.com/forms/d/1HqihdZa7F5GOft8huPfsAPfQ44cxZDmPbxlUSQrKGFw/edit 5/12

12/09/2021 OFLAT - Usability evaluation

13. 10. I needed to learn a lot of things before I could get going with OFLAT. *

Marcar apenas uma oval.

	1	2	3	4	5	
Strongly Disagree	◯	◯	◯	◯	◯	Strongly agree

OFLAT and functionalities

14. Did you notice any pop-up (e.g. input box or some alert) throughout your actions while using OFLAT? *

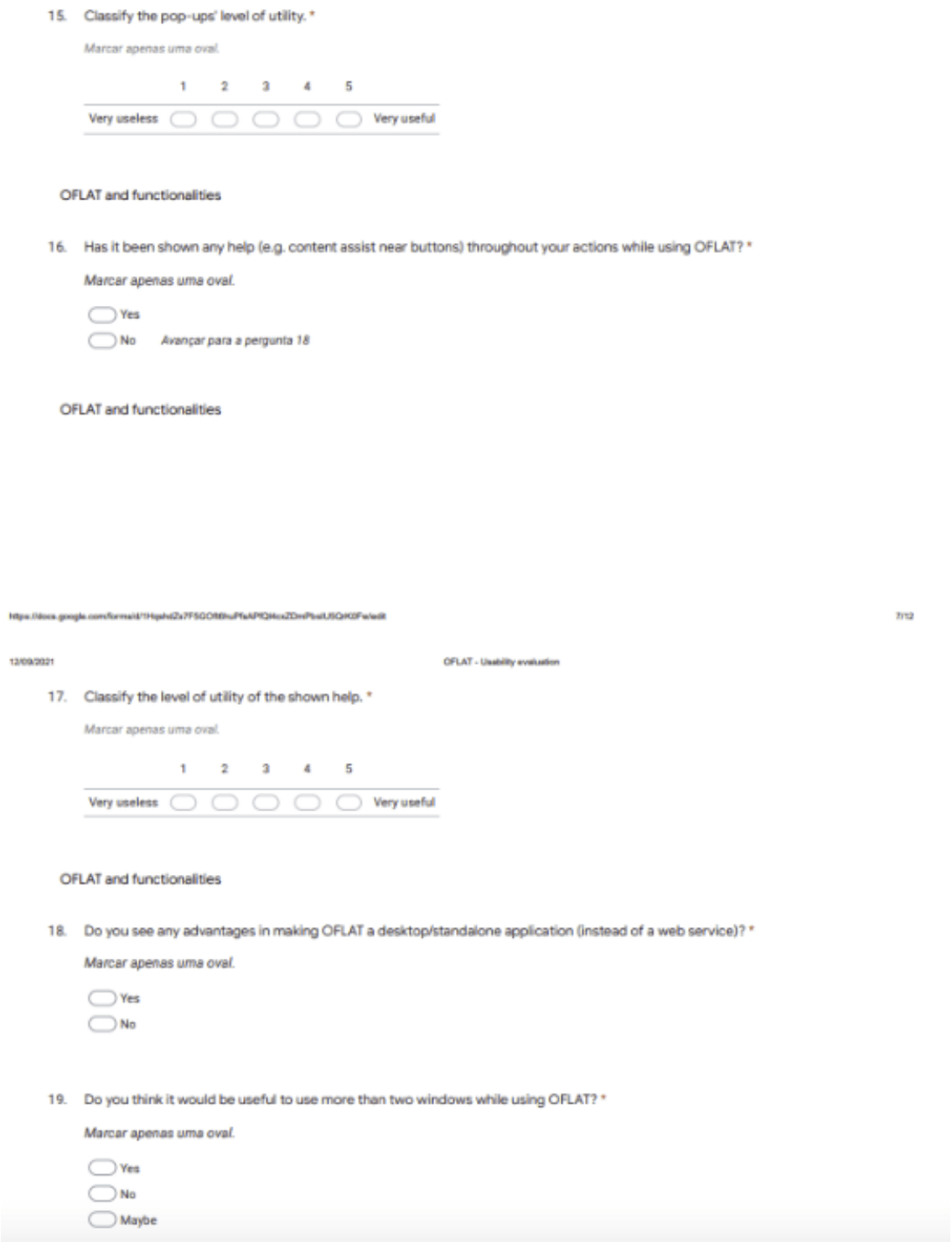
Marcar apenas uma oval.

◯ Yes
◯ No *Avançar para a pergunta 16*

OFLAT and functionalities

Figure 4.4: Survey 1 - page 3. Taken from [41]

The questions shown in Figure 4.3 and Figure 4.4 are related to SUS (from question 4 to question 13). Question 14 asked users if they noticed any pop-ups throughout their actions while using OFLAT.


15. Classify the pop-ups' level of utility. *

Marcar apenas uma oval.

	1	2	3	4	5	
Very useless	◯	◯	◯	◯	◯	Very useful

OFLAT and functionalities

16. Has it been shown any help (e.g. content assist near buttons) throughout your actions while using OFLAT? *

Marcar apenas uma oval.

◯ Yes
◯ No *Avançar para a pergunta 18*

OFLAT and functionalities

17. Classify the level of utility of the shown help. *

Marcar apenas uma oval.

	1	2	3	4	5	
Very useless	◯	◯	◯	◯	◯	Very useful

OFLAT and functionalities

18. Do you see any advantages in making OFLAT a desktop/standalone application (instead of a web service)? *

Marcar apenas uma oval.

◯ Yes
◯ No

19. Do you think it would be useful to use more than two windows while using OFLAT? *

Marcar apenas uma oval.

◯ Yes
◯ No
◯ Maybe

Figure 4.5: Survey 1 - page 4. Taken from [41]

Question 15 asked users to classify the pop-ups' level of utility, the questions 16 and 17 asked users' opinions about the shown help in OFLAT (e.g. content assist near buttons). Question 18 asked if there was any advantage in making OFLAT a desktop/standalone

application, and question 19 asked if it was useful to use more than two windows while using OFLAT (Figure 4.5).

20. How big was the initial effort when using the tool until you feel comfortable? *

Marcar apenas uma oval.

	1	2	3	4	5	
Very small	◯	◯	◯	◯	◯	Very big

21. Did you watch the demo video before using OFLAT? *

Marcar apenas uma oval.

◯ Yes
◯ No

22. Have you gone through any situations where you felt stuck/blocked? *

Marcar apenas uma oval.

◯ Yes
◯ No

https://docs.google.com/forms/d/1HqshdZa7F5GOf89huPfaAPfQi4csZDmPbxlU5QrK0Fw/edit 9/12

12/09/2021 OFLAT - Usability evaluation

23. If you answered "Yes" in the previous question, can you specify where/when?

OFLAT and similars

24. Have you ever tried to use a tool/platform that helped you during your study time for Theory of Computation course? *

Marcar apenas uma oval.

◯ Yes
◯ No

25. If yes, which one(s)?

Figure 4.6: Survey 1 - page 5. Taken from [41]

The questions in the Figure 4.6 asked users' opinions about their initial effort when using OFLAT. Questions 24 and 25 asked users if they used any tool/platform besides OFLAT.

Figure 4.7: Survey 1 - page 6. Taken from [41]

The final questions of the survey asked for final opinions and suggestions about OFLAT (Figure 4.7).

4.3.2 First Survey - Results

The first survey's results were also taken from [41].

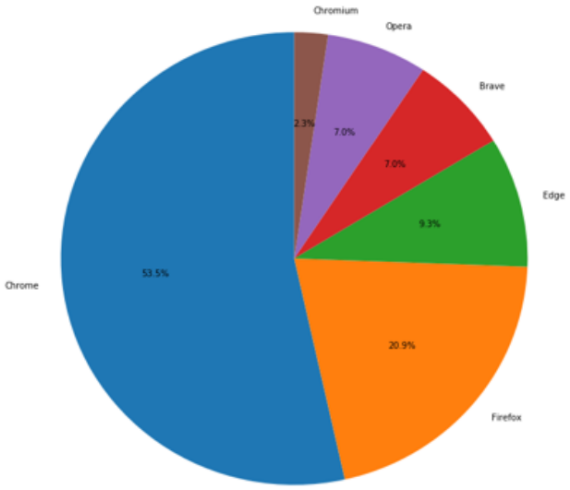


Figure 4.8: Browsers that OFLAT users use. Taken from [41]

By analyzing the pie chart presented in Figure 4.8, it is clear that Chrome was the most chosen browser to use OFLAT.

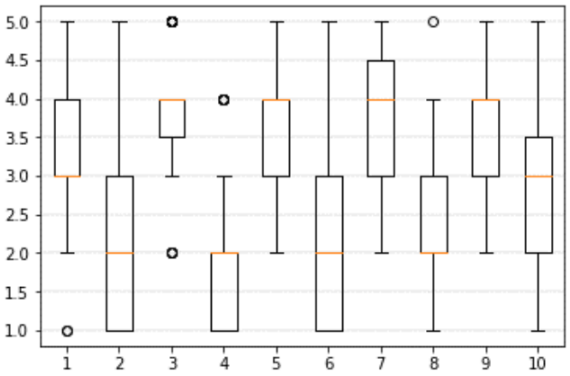
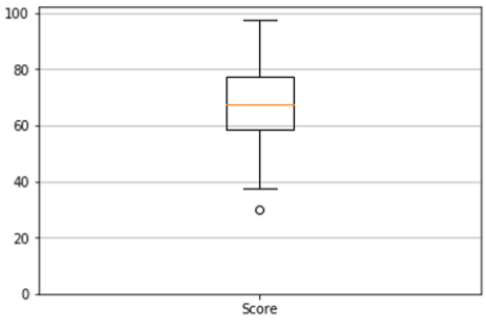


Figure 4.9: Box plot of every SUS question. Taken from [41]

By analyzing the Figure 4.9, it is shown that the boxes have a "zig-zag position" in which the odd question boxes are higher, and the even ones are lower, which is positive feedback (in general).

The median score is 68, which corresponds to an "Okay" rating and a grade C (Figure 4.10).



SUS Score	Grade	Adjective Rating
> 80.3	A	Excellent
68 – 80.3	B	Good
68	C	Okay
51 – 68	D	Poor
< 51	F	Awful

Figure 4.10: Box plot of every SUS score and adjective rating table [51]. Taken from [41]

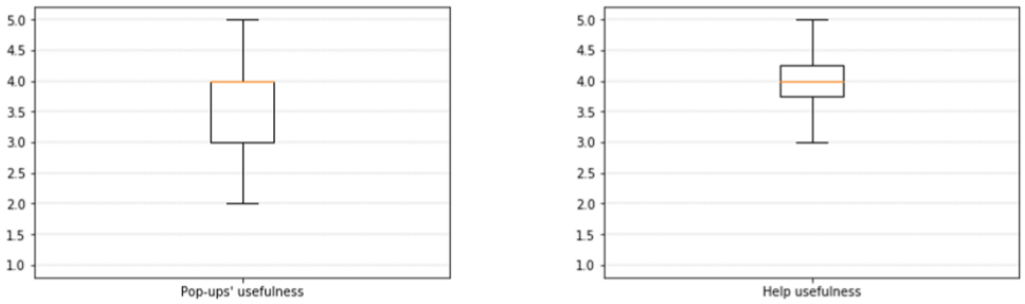


Figure 4.11: Box plot of the pop-ups and help usefulness. Taken from [41]

The box plots of the pop-ups and help usefulness are shown in Figure 4.11.

Many users are considered useful (median = 4), both pop-ups and help, but pop-ups are described as less useful for some users since a few responses are classified as 2.

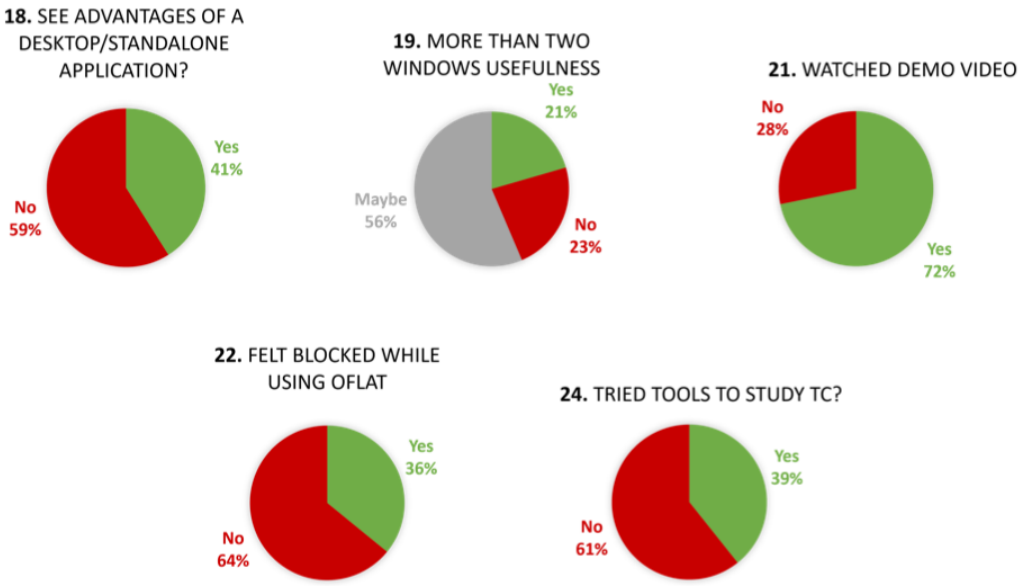


Figure 4.12: Pie charts of answers to questions 18, 19, 22, 22, and 24. Taken from [41]

The pie charts corresponding to the answers to questions 18, 19, 21, 22, and 24 are in the Figure 4.12.

By analyzing the pie charts, it is clear that most users found no advantage in making OFLAT a desktop/standalone application, the usefulness of more than two windows while using OFLAT was very questionable, 72% watched the demonstration video, only 36% felt stuck/blocked during their usage and 39% tried to use tools/platforms that helped during their study time for TC course.

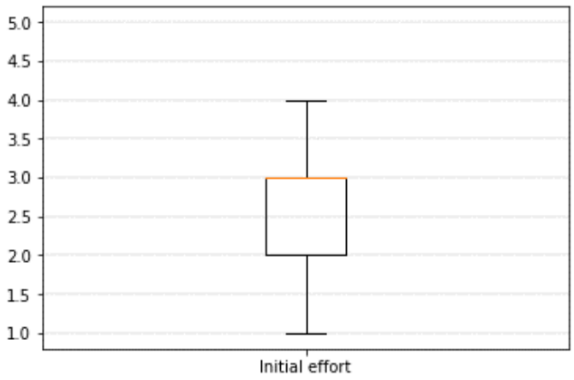


Figure 4.13: Box plot that represents the initial effort level. Taken from [41]

The box plot depicted in Figure 4.13 represents the initial effort level.

It is clear that eight answers reveal that there was a great effort (4), 17 answered average effort (3), 12 considered small (2), and two very small (1).

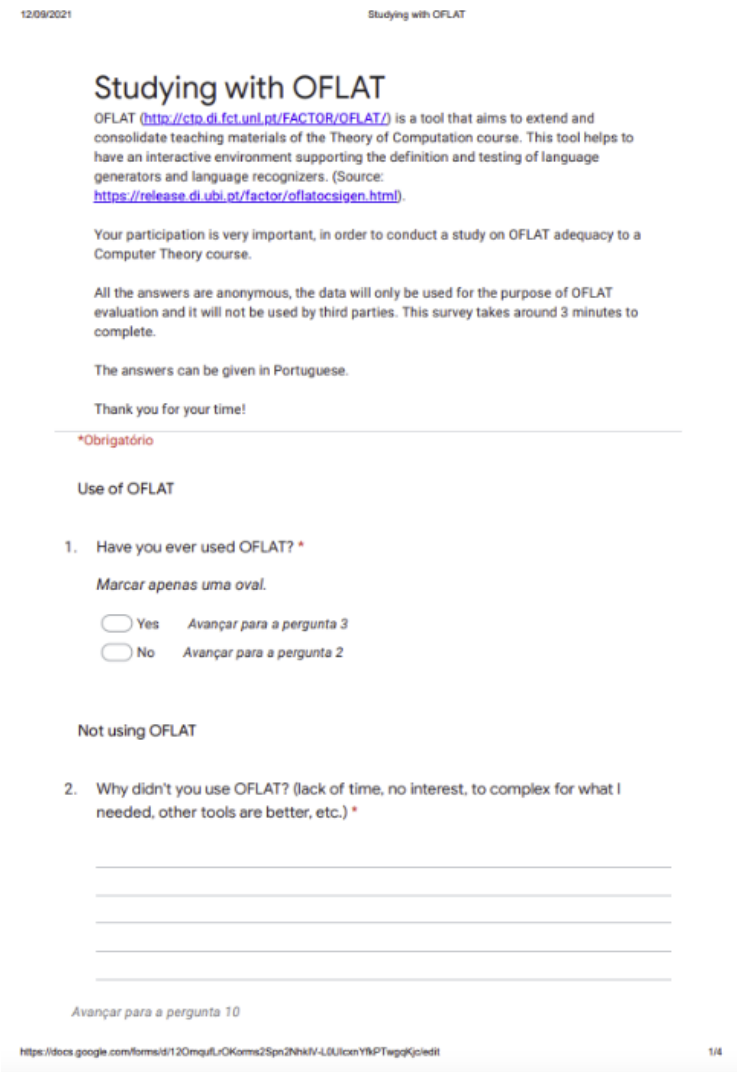
4.4 Second Survey - Studying with OFLAT

After presenting the first survey results to the OFLAT developers, other concerns arose: what is the objective of the tool usage by students, if they consider it useful for their theoretical and practical learning, and if it has been reflected in the success obtained in the course? To see these questions answered, a second survey was conducted. It had fewer answers as it was done in a delicate semester, very close to the beginning of the summer holidays.

The questions are presented in subsection 4.4.1, and the results are presented in subsection 4.4.2, respectively.

4.4.1 Second Survey - Questions

The questions of the second survey were taken from [41].



12/09/2021 Studying with OFLAT

Studying with OFLAT

OFLAT (http://ctp.di.fct.unl.pt/FACTOR/OFLAT/) is a tool that aims to extend and consolidate teaching materials of the Theory of Computation course. This tool helps to have an interactive environment supporting the definition and testing of language generators and language recognizers. (Source: https://release.di.ubi.pt/factor/oflatocsigen.html).

Your participation is very important, in order to conduct a study on OFLAT adequacy to a Computer Theory course.

All the answers are anonymous, the data will only be used for the purpose of OFLAT evaluation and it will not be used by third parties. This survey takes around 3 minutes to complete.

The answers can be given in Portuguese.

Thank you for your time!

*Obrigatório

Use of OFLAT

1. Have you ever used OFLAT? *

Marcar apenas uma oval.

Yes *Avançar para a pergunta 3*

No *Avançar para a pergunta 2*

Not using OFLAT

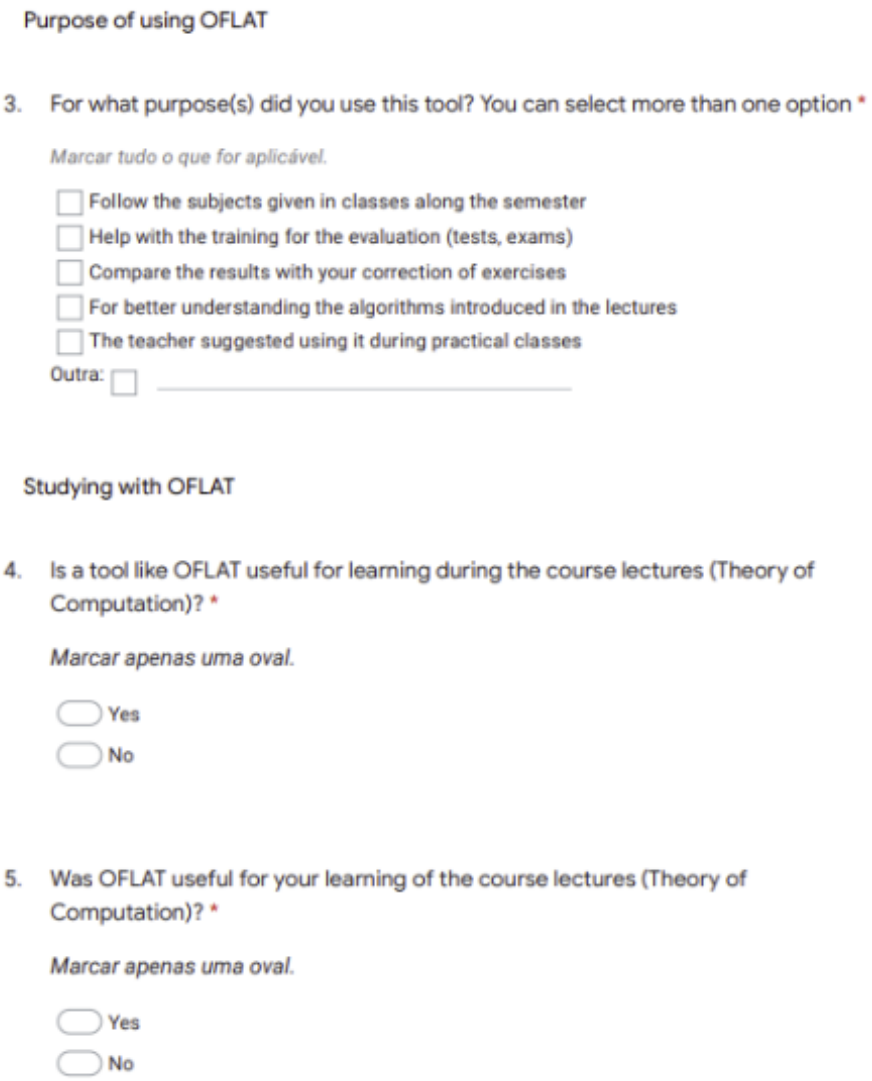
2. Why didn't you use OFLAT? (lack of time, no interest, to complex for what I needed, other tools are better, etc.) *

Avançar para a pergunta 10

https://docs.google.com/forms/d/12OmqufLrOKorms2Spn2NhklV-L0UIcxnYfkPTwgqKjc/edit 1/4

Figure 4.14: Survey 2 - page 1. Taken from [41]

The questions presented in Figure 4.14 are about the use of OFLAT during study time for TC course, i.e., if users have never used OFLAT, they had to explain why they did not use it.


Purpose of using OFLAT

3. For what purpose(s) did you use this tool? You can select more than one option *

Marcar tudo o que for aplicável.

- [ ] Follow the subjects given in classes along the semester
- [ ] Help with the training for the evaluation (tests, exams)
- [ ] Compare the results with your correction of exercises
- [ ] For better understanding the algorithms introduced in the lectures
- [ ] The teacher suggested using it during practical classes

Outra: [ ] _____

Studying with OFLAT

4. Is a tool like OFLAT useful for learning during the course lectures (Theory of Computation)? *

Marcar apenas uma oval.

- ◯ Yes
- ◯ No

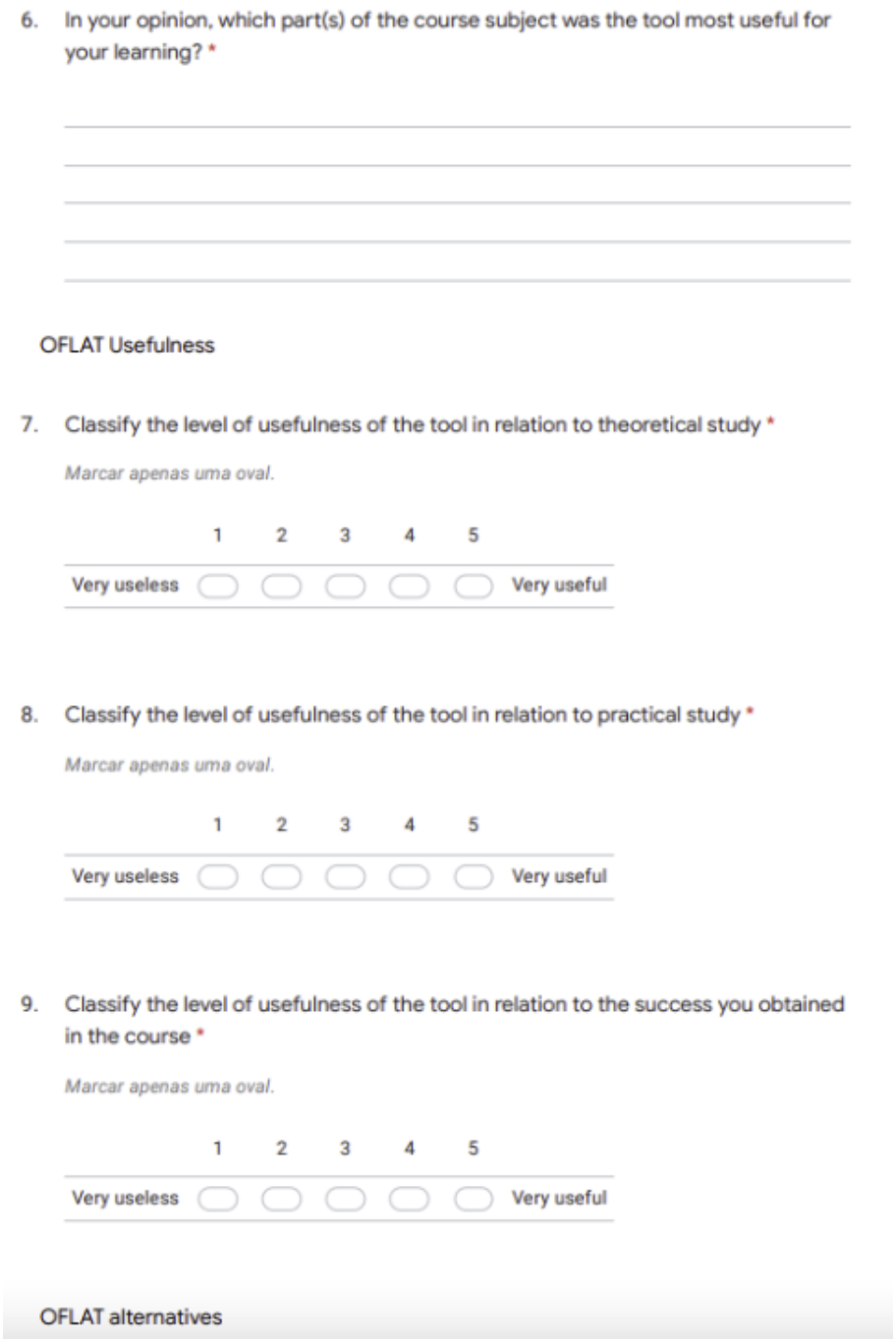
5. Was OFLAT useful for your learning of the course lectures (Theory of Computation)? *

Marcar apenas uma oval.

- ◯ Yes
- ◯ No

Figure 4.15: Survey 2 - page 2. Taken from [41]

If users had used OFLAT during their study time, they had to indicate the purposes of its usage. Questions 4 and 5 asked if the tool was helpful in learning during the lectures (Figure 4.15).


6. In your opinion, which part(s) of the course subject was the tool most useful for your learning? *

OFLAT Usefulness

7. Classify the level of usefulness of the tool in relation to theoretical study *

Marcar apenas uma oval.

	1	2	3	4	5	
Very useless						Very useful

8. Classify the level of usefulness of the tool in relation to practical study *

Marcar apenas uma oval.

	1	2	3	4	5	
Very useless						Very useful

9. Classify the level of usefulness of the tool in relation to the success you obtained in the course *

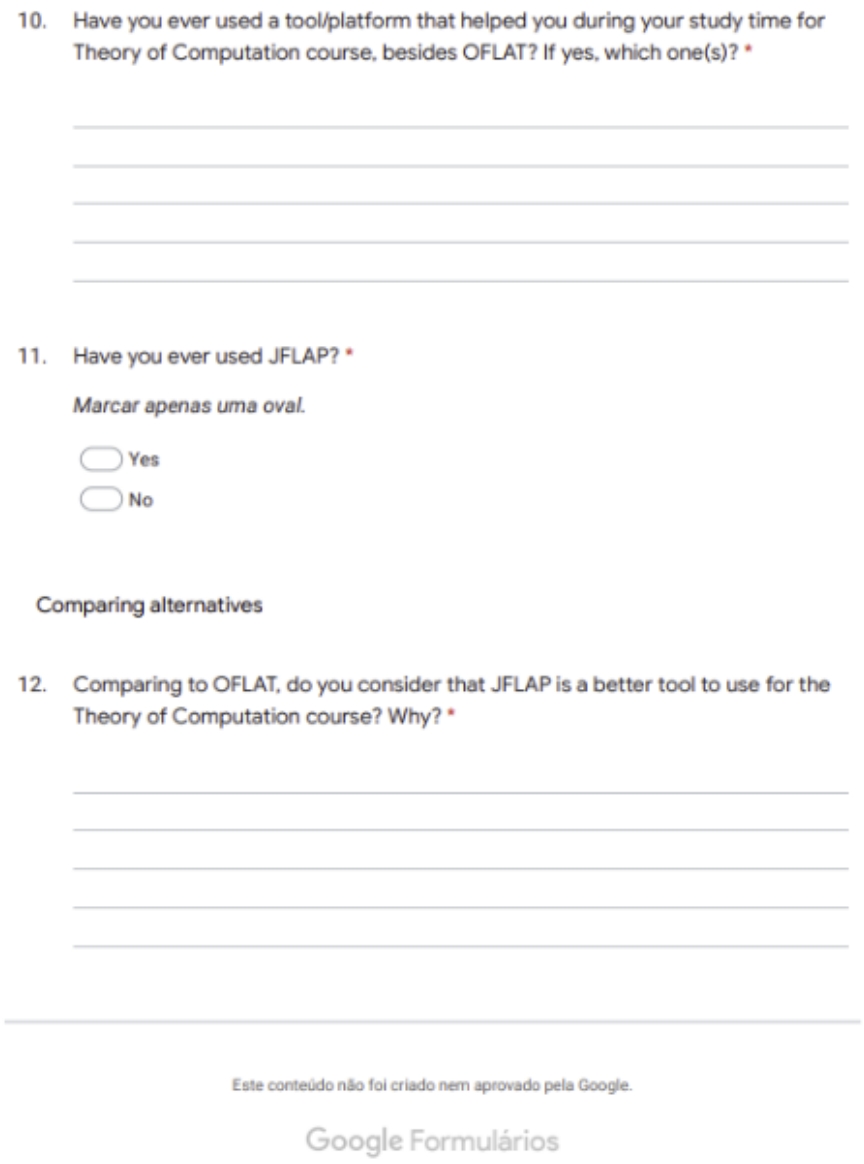
Marcar apenas uma oval.

	1	2	3	4	5	
Very useless						Very useful

OFLAT alternatives

Figure 4.16: Survey 2 - page 3. Taken from [41]

The questions presented in Figure 4.16 are about the usefulness of OFLAT in relation to theoretical study, practical study, and the success obtained in the course.


10. Have you ever used a tool/platform that helped you during your study time for Theory of Computation course, besides OFLAT? If yes, which one(s)? *

11. Have you ever used JFLAP? *

Marcar apenas uma oval.

Yes
No

Comparing alternatives

12. Comparing to OFLAT, do you consider that JFLAP is a better tool to use for the Theory of Computation course? Why? *

Este conteúdo não foi criado nem aprovado pela Google.

Google Formulários

Figure 4.17: Survey 2 - page 4. Taken from [41]

The questions presented in Figure 4.17 asked if users had used another tool/platform besides OFLAT and if users had ever used JFLAP and compared it with OFLAT.

4.4.2 Second Survey - Results

The results of the second survey were also taken from [41].

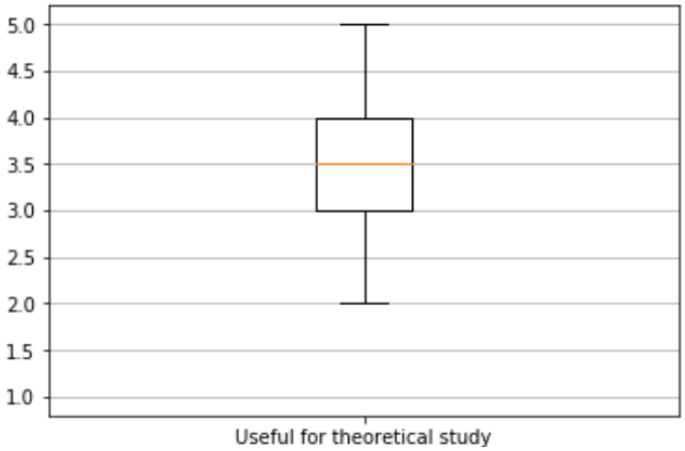


Figure 4.18: Box plot of how useful OFLAT is to theoretical study. Taken from [41]

The box plot presented in Figure 4.18 shows how useful OFLAT was to theoretical study.

The median answer was between average and some usefulness, while some were considered very useful (level 5) and others were not that useful (level 2).

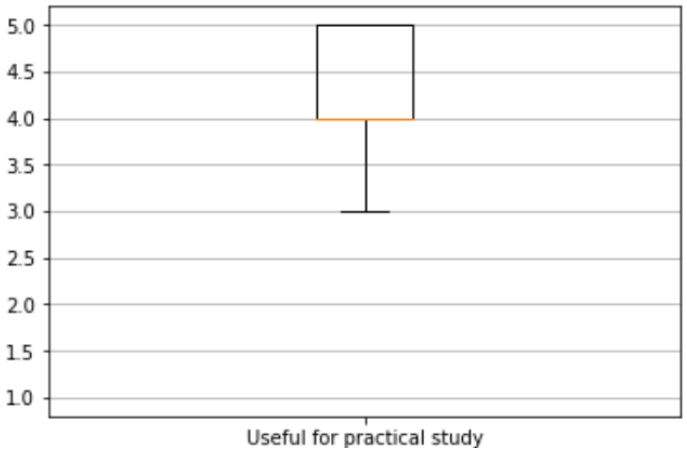


Figure 4.19: Box plot of how useful OFLAT is to practical study. Taken from [41]

The box plot presented in Figure 4.19 shows how useful OFLAT was to practical study.

Some students answered that it was very useful or somehow helpful. It is also clear that almost all students answered in between. The usefulness level is 4.

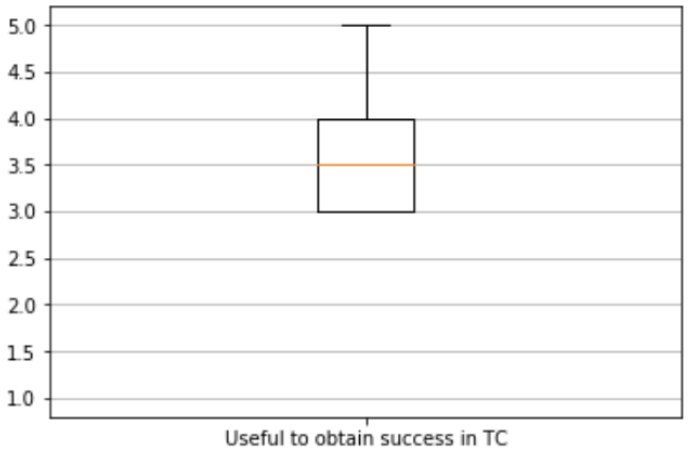


Figure 4.20: Box plot of how useful OFLAT is to obtain success in the TC course. Taken from [41]

The box plot presented in Figure 4.20 shows how useful OFLAT was to obtain success in the TC course.

OFLAT was useful in obtaining success in the TC course.

4.5 OFLAT version 2.1

After the first usability study, it was released OFLAT version 2.1. This version has the same functionalities as the previous version (see section 4.2), and the new functionalities were the following:

- Creation of Context-Free Grammar model;
- Edit the visible model;
- Export model.

In addition to the new functionalities, some improvements have been made to the existing ones, e.g., adding transitions to an automaton by dragging an arrow from one node to another, and the creation of Finite Automata, Regular Expression, or Context-Free Grammar is reserved for the "New model" button instead of one button for each model.

4.6 Summary

From these two surveys, it is possible to understand the users' opinions on the OFLAT platform based on their own experience, without any guided tasks to complete and on a perspective of regular usage of students while taking the Theory of Computation course.

It is also clear that most of the known functionalities were improved based on the pre-study and experiment of the first usability study. However new functionalities were introduced in the new release, potentially inflicting some changes in the usability of the

tool. Therefore, in a continuous effort, a new usability study should be addressed to make a more robust tool. The next chapters will present the instrumentation of the tool to deal with logging and the empirical study carried on.

5 OFLAT version 2.1 and related usability assessment tools

The architectural design of OFLAT version 2.1 is presented in the following component diagram which represents how the OFLAT application works when the new model is created, i.e., create a new automaton, regular expression, context-free grammar, pushdown automata, Turing machines, or pumping lemma (see Figure 5.1).

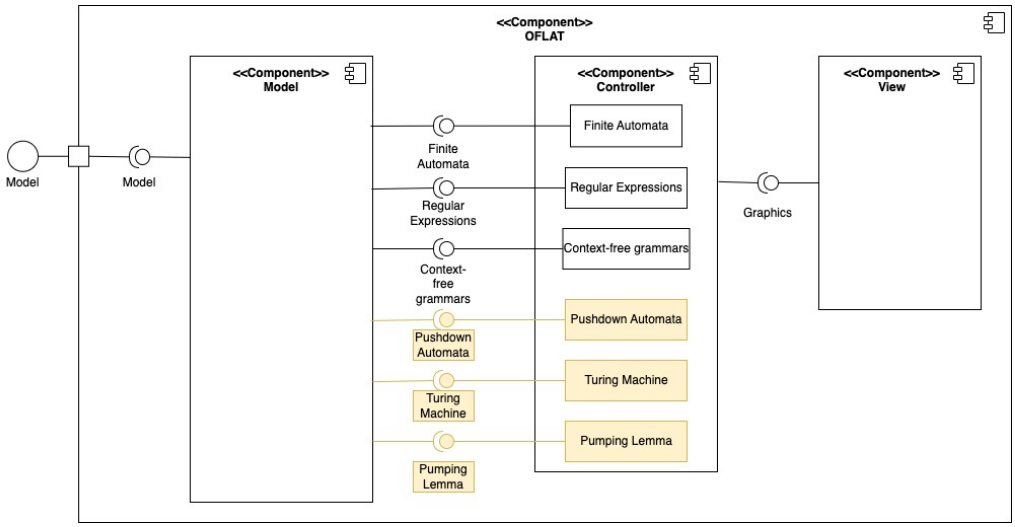


Figure 5.1: Component diagram of OFLAT

The implemented functionalities that are shown in yellow, such as the pushdown automata, Turing machines, and pumping lemma, will be integrated with the future versions of OFLAT.

5.1 Extending OFLAT with Logging

One tool is related to creating, collecting, and analyzing user-generated data while using the OFLAT website, which is worth mentioning. This tool can be very useful for deepening the study and evaluation of usability: Segment.

Segment allows the application of code snippets to the current OFLAT website and generates data through the direct usage of the site.

Segment is a customer data platform that provides a complete data toolkit and allows one to collect, transform, send, and archive data. Segment tracks events when users interact with a specified interface, such as the OFLAT website.

The Segment libraries generate messages about what happens on the chosen interface, translate those messages into different formats for use by destinations, and transmit the messages to those tools.

Several tracking API methods can be called to generate messages. The four most important methods are:

- Identify: who is the user?
- Page and Screen: what web page or app screen are they on?
- Track: what are they doing?

Every call shares the same common fields. When these methods are used as intended, it allows Segment to detect a specific type of data and correctly translate it to send it on to downstream destinations [46].

By adding a simple line of code like "analytics.track(event, [properties], [options], [callback]);" to the JavaScript code of a website button, page, or another event that one wants to track, that tracking is received by Segment and registered in a chosen data warehouse like PostgreSQL. That data can later be analyzed and managed by using a suggested visual analytics software like Google Analytics[1] (see Figure 5.2).

[1] Google Analytics: https://analytics.google.com

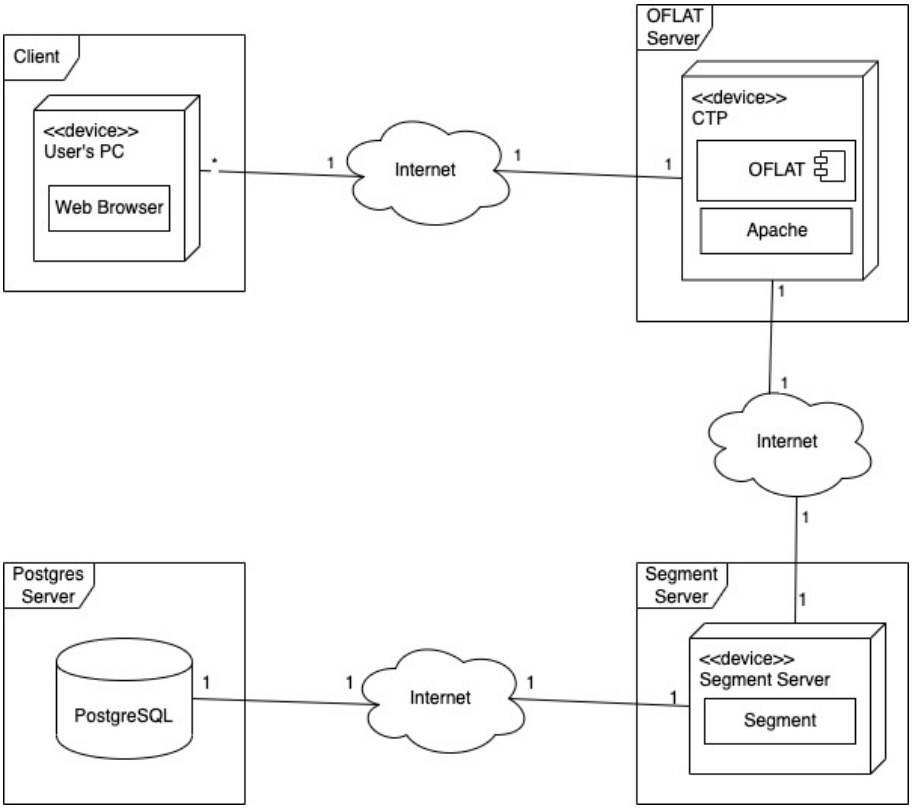


Figure 5.2: Deployment diagram of Segment Data Flow

The component diagram that follows, with the logging system depicted in blue, shows how the OFLAT application works when a new model is produced (Figure 5.3).

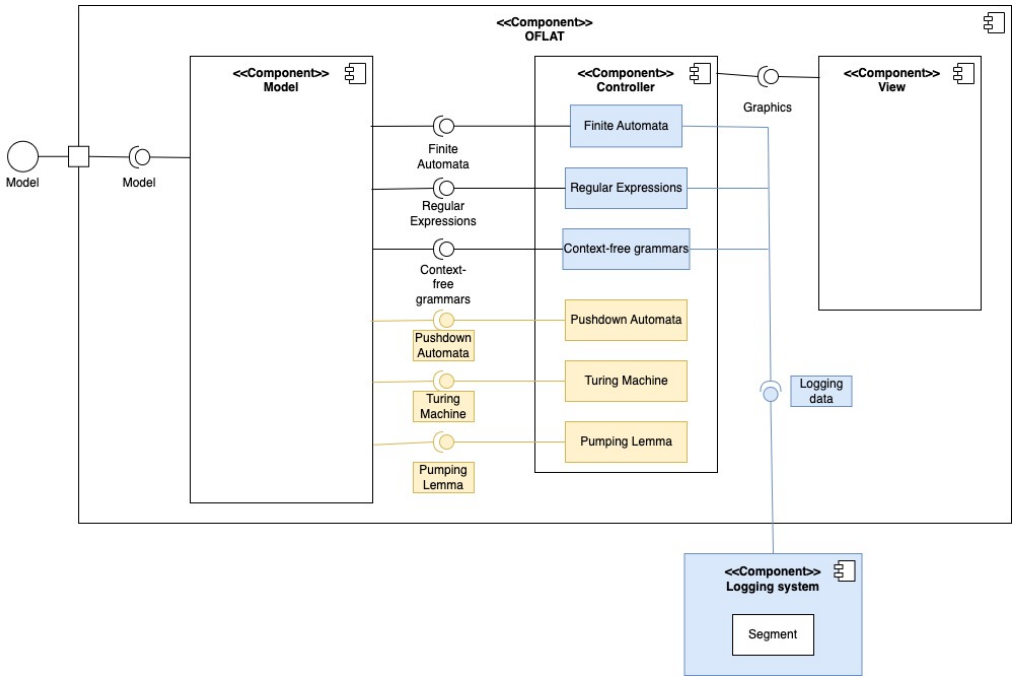


Figure 5.3: Component diagram of OFLAT with logging component

5.2 Process mining usability logging

ProM [40] is an extensible framework that supports various process mining techniques in the form of plug-ins. It is platform-independent as it is implemented in Java (Figure 5.4).

ProM is distributed in parts, which offers maximal flexibility. First, the ProM 6 core is distributed as a downloadable package using the GNU Public License (GPL) open-source license. Second, the ProM 6 plug-ins are distributed as separate packages, typically using the Lesser GNU Public License (L-GPL) open-source license.

Figure 5.4: ProM Tool

There was another process mining tool called Disco (Figure 5.5). Disco is a commercial tool that is simple, fast, and easy to use. It only supports a kind of fuzzy models and it only provides a limited number of analysis types [10].

Figure 5.5: Disco Tool [20]

ProM was chosen for process mining analysis instead of Disco for the following reasons:

- To use Disco, it is required a license acquisition;
- ProM provides many different types of analysis, and can support many different types of models;

5.2.1 ProM Example

The description of the following example can be found in the "Introduction - Running Example" section of ProM 6 tutorial.

In the "Inspecting and Cleaning an Event Log" section of the tutorial, it shows how to inspect and clean the log.

The repairExample.xes file has process instances of the running example that will be used to inspect or mine a log.

The Figure 5.6 shows the screen of ProM 6 after importing the repairExample.xes file as an event log.

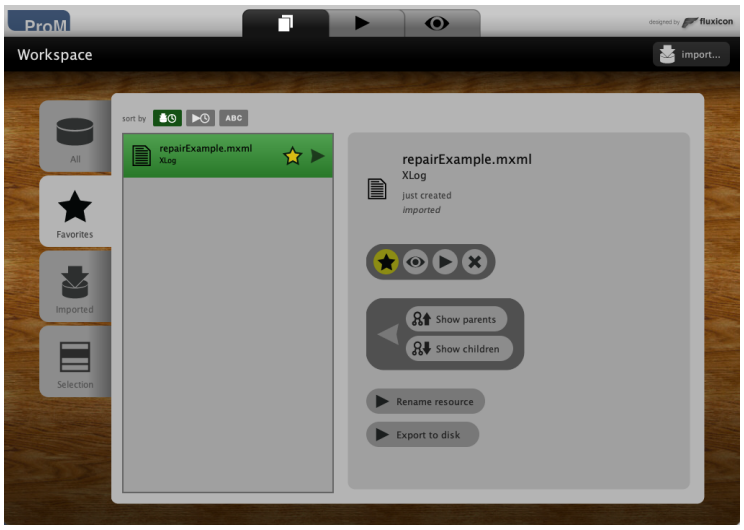


Figure 5.6: ProM 6 screen after importing the file repairExample.xes as an event log

To perform a log inspection, we want to answer the following questions:

1. How many cases (or process instances) are in the log?
2. How many tasks (or audit trail entries) are in the log?
3. How many resources are in the log?
4. Are there running cases in the log?
5. Which resources work on which tasks?

These five questions can be answered by viewing the log summary. To view this summary, we select the entry "repairExample.mxml" as shown by the last figure above, we select the "Visualize" button in the details view of this entry, and then we select the "Summary" tab.

In the "MXML Legacy Classifier", we can answer the first four questions of the list above (Figure 5.7 and Figure 5.8):

1. 1104 process instances;
2. 2 tasks (start and complete);
3. 13 resources;

4. 1104 running cases.

Total number of process instances: **1104**
Total number of events: **11855**

MXML Legacy Classifier

Event classes defined by MXML Legacy Classifier
All events

Total number of classes: **12**

Class	Occurrences (absolute)	Occurrences (relative)
Test Repair+complete	1508	12,72%
Test Repair+start	1508	12,72%
Register+complete	1104	9,313%
Analyze Defect+complete	1104	9,313%
Analyze Defect+start	1104	9,313%
Inform User+complete	1102	9,296%
Archive Repair+complete	1000	8,435%
Repair (Simple)+start	785	6,622%
Repair (Simple)+complete	785	6,622%
Repair (Complex)+start	725	6,116%
Repair (Complex)+complete	724	6,107%
Restart Repair+complete	406	3,425%

Figure 5.7: Log Summary - Part 1

Start events

Total number of classes: **1**

Class	Occurrences (absolute)	Occurrences (relative)
Register+complete	1104	100,0%

End events

Total number of classes: **5**

Class	Occurrences (absolute)	Occurrences (relative)
Archive Repair+complete	1000	90,58%
Test Repair+complete	75	6,793%
Inform User+complete	27	2,446%
Repair (Complex)+complete	1	0,091%
Repair (Complex)+start	1	0,091%

Figure 5.8: Log Summary - Part 2

The fifth question of the list above can be answered by checking out the "Event Name and Resources" section of the log summary (Figure 5.9). There are three people in each of the teams in the Repair department of the company. The employees with login "SolverC..." deal with the complex defects, while those with login "SolverS..." handle the simple defects.

Total number of classes: **22**

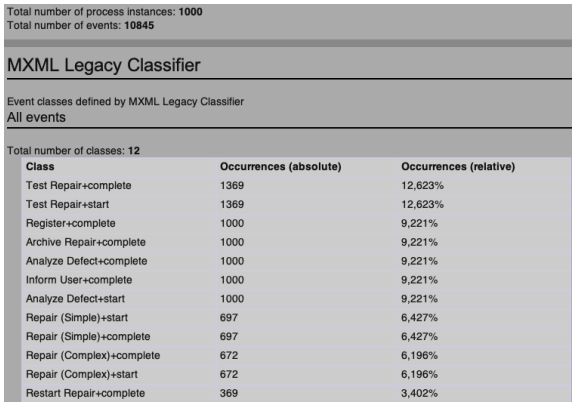
Class	Occurrences (absolute)	Occurrences (relative)
Register+System	1104	9,313%
Inform User+System	1102	9,296%
Archive Repair+System	1000	8,435%
Repair (Simple)+SolverS1	592	4,994%
Repair (Complex)+SolverC1	534	4,504%
Test Repair+Tester3	528	4,454%
Test Repair+Tester1	516	4,353%
Test Repair+Tester5	516	4,353%
Repair (Complex)+SolverC2	514	4,336%
Test Repair+Tester2	500	4,218%
Repair (Simple)+SolverS2	498	4,201%
Test Repair+Tester6	486	4,1%
Repair (Simple)+SolverS3	480	4,049%
Test Repair+Tester4	470	3,965%
Restart Repair+System	406	3,425%
Analyze Defect+Tester2	404	3,408%
Repair (Complex)+SolverC3	401	3,383%
Analyze Defect+Tester6	390	3,29%
Analyze Defect+Tester1	386	3,256%
Analyze Defect+Tester3	382	3,222%
Analyze Defect+Tester5	328	2,767%
Analyze Defect+Tester4	318	2,682%

Figure 5.9: Event Name and Resource

After inspecting the logs, the process of cleaning the logs is made.

In ProM 6, a log can be filtered by applying specific actions. From the description of the running example, we know that the completed tasks are the ones that start with a task to register the phone and end with a task to archive the instance. Thus, to filter the completed cases, the following procedure is executed:

1. Select the entry “repairExample.mxml”, and select the “Action” button;
2. Select the “Filter log using Simple Heuristics” action and select the “Start” button;
3. The first log filter to configure is the event type filter, allowing us to select the type of events we want to consider while mining the log. For the running example, the log has tasks with two event types: complete and start;
4. The second filter is the start event filter, which filters the log in such a way that only the traces that start with the indicated tasks are kept;
5. The third filter is the end event filter, which filters the log in such a way that only the traces that end with indicated tasks are kept;
6. The fourth filter is the event filter, which filters all unselected events from the log.



Total number of process instances: **1000**
Total number of events: **10845**

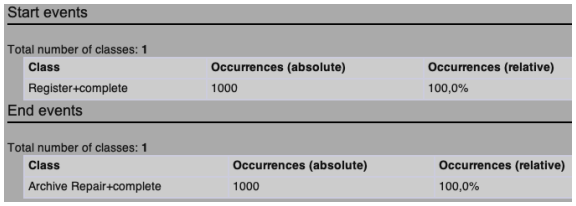
MXML Legacy Classifier

Event classes defined by MXML Legacy Classifier
All events

Total number of classes: **12**

Class	Occurrences (absolute)	Occurrences (relative)
Test Repair+complete	1369	12,623%
Test Repair+start	1369	12,623%
Register+complete	1000	9,221%
Archive Repair+complete	1000	9,221%
Analyze Defect+complete	1000	9,221%
Inform User+complete	1000	9,221%
Analyze Defect+start	1000	9,221%
Repair (Simple)+start	697	6,427%
Repair (Simple)+complete	697	6,427%
Repair (Complex)+complete	672	6,196%
Repair (Complex)+start	672	6,196%
Restart Repair+complete	369	3,402%

Figure 5.10: Log summary after applying simple heuristics - Part 1



Start events

Total number of classes: **1**

Class	Occurrences (absolute)	Occurrences (relative)
Register+complete	1000	100,0%

End events

Total number of classes: **1**

Class	Occurrences (absolute)	Occurrences (relative)
Archive Repair+complete	1000	100,0%

Figure 5.11: Log summary after applying simple heuristics - Part 2

Let us now inspect the resulting log (Figure 5.10). We notice that the log contains fewer cases (1000 cases), and all the cases indeed start with the task "Register (complete)" and finish with the task "Archive Repair (complete)" (Figure 5.11).

6 | Controlled Experiment

As mentioned in section 2.3, a controlled experiment is a research methodology related to case studies that aim to manipulate one or more variables. At the same time, everything else is held constant to measure the obtained effects.

In this chapter, the plan of the controlled experiment is presented by describing the procedure itself and its implementation, following the suggested structure in *Reporting Experiments in Software Engineering* (pp. 201-228), in [48].

6.1 Theory of Computation course

The subject of the Theory of Computation occurs in the second semester of NOVA's School of Science and Technology Computer Science course (second year of MIEI and LEI).

This subject is composed of the following topics: "Modeling with Sets and Logic", "Machines, Automata and Specifications" and "Computability".

There were students who did not have any subject before the Theory of Computation about automata theory, regular expressions, context-free grammar, and Turing machines.

6.2 Experience contact

This year, 245 students attended the Theory of Computation course. Of these 245 students, 163 were from LEI, 81 were from MIEI and one was from LM (Mathematics course).

6.2.1 Student profile

We can divide this group of 245 students into two groups: A and B. Group A comprises 165 students who made the first enrollment in the course, while Group B comprises 80 students who made two or more enrollments.

The students from group A did not have knowledge about the OFLAT tool and the FLAT topic, while the students from group B already had knowledge about the tool and the topic.

6.3 Experiment Plan

6.3.1 Goals

This experiment is expected to assess the user experience and usability of OFLAT. With this in mind, the goals of this experiment can be described using the Goal-Question-Metric [5] model.

The main goal (**G1**) is to *analyze* the usability and user experience of OFLAT, *for the purpose of* evaluation, *with respect* to the impact on the students of Theory of Computation, *from the viewpoint of* research, *in the context of* a controlled experiment conducted at NOVA School of Science and Technology.

To refine this goal to a more quantifiable way, it can then be divided into the following sub-goals that are aimed to achieve during this controlled experiment:

- **G1.1:** *Analyze* the usability and user experience of OFLAT, *for the purpose of* evaluation, *with respect to* the correctness of the solutions presented to the suggested weekly challenges, *from the viewpoint of* research, *in the context of* a controlled experiment conducted at NOVA School of Science and Technology;
- **G1.2:** *Analyze* the usability and user experience of OFLAT, *for the purpose of* evaluation, *with respect to* the ease of the tool's usage, during the resolution of the weekly challenges, *from the viewpoint of* research, *in the context of* a controlled experiment conducted at NOVA School of Science and Technology;
- **G1.3:** *Analyze* the usability and user experience of OFLAT, *for the purpose of* evaluation, *with respect to* the users' workload demand towards the tools, during the resolution of weekly challenges, *from the viewpoint of* research, *in the context of* a controlled experiment conducted at NOVA School of Science and Technology.

For each sub-goal presented, following the GQM model, questions are defined:

- **Q1.1:** Is OFLAT error-prone?
- **Q1.2:** Is OFLAT easy to use?
- **Q1.3:** Does OFLAT demand workload?

To answer these questions, it is necessary to define metrics:

- **M1.1:** score to represent the correctness of each weekly challenge;
- **M1.2:** users' perceived usability score by SUS questionnaire;
- **M1.3:** workload score by NASA TLX questionnaire;

6.3.2 Experimental units

6.3.2.1 Participants Sample

Since there are goals related to the impact on students' lives, the sample is drawn from Computer Science students who were attending the Theory of Computation course.

The invitation to participate in the experiment was shared with Computer Science students from NOVA School of Science and Technology who attended the Theory of Computation course this year. The invitation was sent through CLIP[1] with the link to Moodle[2].

As mentioned in section 6.2, 245 students from NOVA School of Science and Technology attended the Theory of Computation course. However, only 153 students were registered on Moodle (Figure 6.1).

From 153 students registered on Moodle, only 70 students volunteered to answer at least one exercise from one of the weekly challenges.

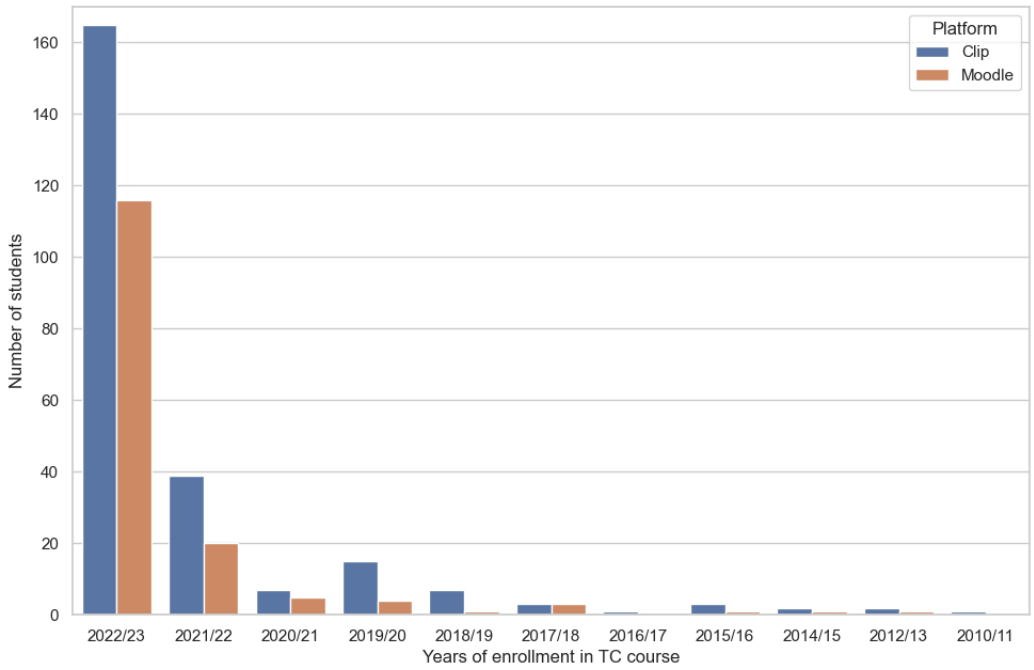


Figure 6.1: Participants' years of enrollment in TC course in Clip and Moodle

6.3.2.2 Groups

All participants are challenged to test OFLAT. By dividing the participants into two groups, it is a way to ensure no biased results.

[1] CLIP: https://clip.fct.unl.pt/utente/eu
[2] Moodle: https://moodle.fct.unl.pt/course/view.php?id=8349

With this being said, there are two groups: one consists of students who made the first enrollment in the course and another consists of students who made two or more enrollments (just as described in the previous section - subsection 6.2.1).

6.4 Experimental process

The second semester is about 14 weeks. During the first month of the semester, it is dedicated to "Modeling with Sets and Logic", i.e., Sets, inductive definitions, relations, etc. The following nine weeks are dedicated to "Machines, Automata and Specifications", and the last two weeks of the semester are dedicated to "Computability" (see Table 6.1).

Topic	Duration (weeks)
Modeling with Set and Logic	3
Machines, Automata and Specifications	9
Computability	2

Table 6.1: The duration of each topic of TC

Throughout five weeks, there were weekly challenges about automata theory for students to use OFLAT. Once the exercises of the challenge were done, those would be submitted to the Moodle platform (see Figure 6.2). After submitting the exercise to Moodle, it will analyze the solution. If the solution was wrong, then the student had to use OFLAT again to correct the solution. Otherwise, the solution would be accepted, and the student would fill out the weekly questionnaire through Moodle.

For the data part, when the student started to solve the weekly challenge in OFLAT, it was registered log data in order to understand the bottlenecks, the performance, and if there were possibilities for improvement. To extract that knowledge, it was used, the tool ProM was described in the previous chapter (section 5.2).

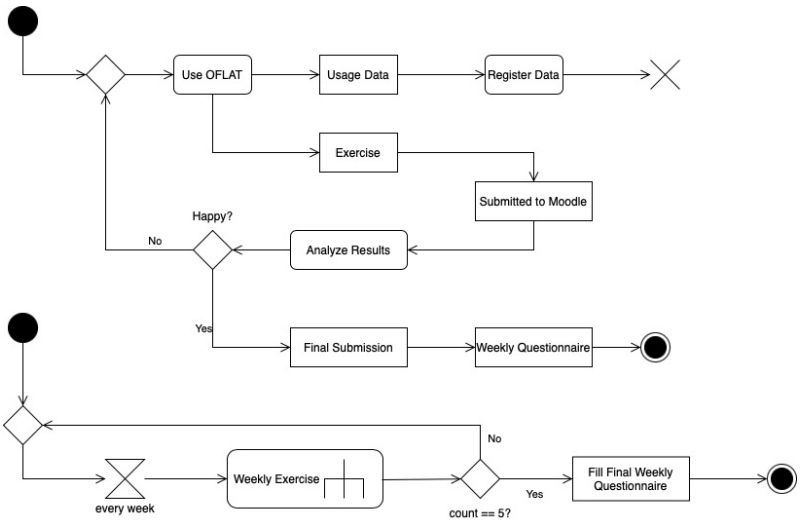


Figure 6.2: The process of the weekly challenge

Each weekly challenge corresponded to the specific topic of the course, e.g., the first weekly challenge corresponded to the topic "Deterministic Finite Automata", the second weekly challenge corresponded to the topic "Regular Expressions" and so on (see Table 6.2).

Week number	Topic
1, 2	Deterministic Finite Automata
3, 4	Regular Expressions
5	Context-free grammar

Table 6.2: Weekly challenge topics

6.4.1 Experimental material

Since the experiment was not carried out in person, participants were asked to solve the weekly challenges as a complement to their studies.

To organize all the information, it was created three usability questionnaires for each weekly challenge. Each questionnaire contains a question that asks students to introduce the student number, followed by four questions that ask the student's opinion about the difficulty of the proposed exercises of each weekly challenge, and some questions about specific functionalities used in each weekly challenge. At the end of each questionnaire, students were asked to give opinions or suggestions to improve the usability of OFLAT, grade OFLAT in terms of usability, and which browser they used to solve each weekly challenge.

The questionnaires were carried out in Moodle due to the lack of memory of Google Forms, and ease of analysis by converting the results collected into Excel (*.xlsx*) files.

The guides and questionnaires can be found in the Zenodo reference [36].

6.4.2 Weekly challenges' exercises

The exercises performed by the students covered the following:

1. First challenge - Deterministic Finite Automata:

   a) Specify a DFA over the alphabet {a,b,c} for the language of words with an odd number of occurrences of the substring "ab";

   Accepted words: ab, bab, cab, aab, ababab, aaccaabbca, aba

   Rejected words: baaa, caaa, b, c, abab, abbcabbc

   b) Specify a DFA over the alphabet {a,b,c} for the language of words in which the total number of b's and the total number of c's is two;

   Accepted words: bbcc, bcbc, cbaacba, abcbaca, cbabc

   Rejected words: aa, bb, bbc, ccac, abc, caaacb

c) Specify a DFA over the alphabet {0,1} for the language of words that represent the naturals that are multiple of 3;

Accepted words: 00000011, 00001001, 00010010

Rejected words: 00001011, 00010001, 01010000

d) Specify a DFA over the alphabet {0,1} for the language of words that represents the regular expression $(01 + 001 + 010)^*$.

Accepted words: 01, 001, 010, 0101, 01001, 010010

Rejected words: 0, 00, 0010, 0100, 01000, 01010100010

2. Second challenge - Regular Expression:

a) Define a regular expression over the alphabet {0,1} which defines the language over it of all words that do not contain the substring "000";

Accepted words: ϵ, 0, 00, 1, 01, 001, 1101001

Rejected words: 000, 01000, 1000, 11000, 001000, 011000

b) Define a regular expression over the alphabet {a,b} whose language is the set of words in which every "a" is immediately preceded or followed by "b";

Accepted words: b, baab, aba

Rejected words: aaaaa, aaabbb, bbbaaab

c) Define a regular expression over the alphabet {a,b,c} whose language is the set of words with even length that contains precisely one "a";

Accepted words: ba, ab, ca, ac, bcab, acbb

Rejected words: ϵ, b, c, bb, cc, bcbcbc, bbbc

d) Define a regular expression over the alphabet {a,b} whose language is the set of words in which the substring "aa" occurs at least twice.

Accepted words: aaa, aaaa, baabaab, baaab, bbaaa, baaaab

Rejected words: ϵ, a, aa, ba, ab, bbaa, aabb

3. Third challenge - Context-Free Grammar:

a) Define the context-free grammar for the language $\{0^m 1^n 0^{m-1} \mid m \in \mathbb{N} \wedge n \in \mathbb{N}_0\}$;

Accepted words: 0, 01, 0010, 00110, 0001100, 0011110

Rejected words: ϵ, 1, 10, 1000, 001100, 11100

b) Define the context-free grammar for strings not of the form $0^i 1^j$ where i = j and i, j $\geq$ 0;

Accepted words: 0101, 000011, 00010001

Rejected words: 01, 0011, 000111

c) Define the context-free grammar for generating all strings that are regular expressions in the alphabet {a, b, (,), +, *, ϵ};

Accepted words: ϵ, a + b, (a+b)*, (b + a + ϵ)a*, (a + b)*a

Rejected words: a+, a+b), ((b), **, a++b

d) Define the context-free grammar for $\{0^{2n}1^k2^n \mid n, k \in \mathbb{N}_0\}$.

Accepted words: 0012, 0000122, 000011122

Rejected words: 01, 0011, 0102

The question "Is it possible to get an LL1 deterministic parser for this grammar?" was an additional question for the third weekly challenge's exercises.

6.4.3 Hypotheses and Variables

All the information regarding the independent and dependent variables is presented in Table 6.3.

Type	Name	Value
Independent	Tool Experience	Previous or first usage of OFLAT
	Year(s) of enrollment in TC course	2010/11 to 2022/23
Dependent	Correctness	Score
	Perceived usability	SUS Score
	Workload	NASA TLX Score

Table 6.3: Independent and dependent variables overview

Having the dependent variables, null and alternative hypotheses (H_0 and H_1, respectively) can be defined accordingly:

$H_{0\,\text{Correctness}}$: The user correctness with the new version of OFLAT is **the same** as the correctness with the previous version.

$H_{1\,\text{Correctness}}$: The user correctness with the new version of OFLAT is **significantly different** from the correctness with the previous version.

$H_{0\,\text{Usability}}$: The user's perception of usability with the new version of OFLAT is **the same** as the perception of usability with the previous version.

$H_{1\,\text{Usability}}$: The user's perception of usability with the new version of OFLAT is **significantly different** from the perception of usability with the previous version.

$H_{0\,\text{Workload}}$: The user workload with the new version of OFLAT is **the same** as the workload with the previous version.

$H_{1\,\text{Workload}}$: The user workload with the new version of OFLAT is **significantly different** from the workload with the previous version.

6.4.4 Design

The chosen experimental design consists of a controlled experiment, recurring to qualitative and quantitative research methods, to collect qualitative and quantitative data, as mentioned in section 2.4.

The subjects that participated in this experiment were organized following the within-subjects design. This means that the same person tests OFLAT. Compared to the between-subjects design (where one person would only test some conditions of OFLAT), the within-subject design requires fewer participants since each student provides data for each level of independent variables [54].

6.5 Execution

The execution of this experiment, including preparation and procedure, is represented in an activity diagram in Figure 6.3.

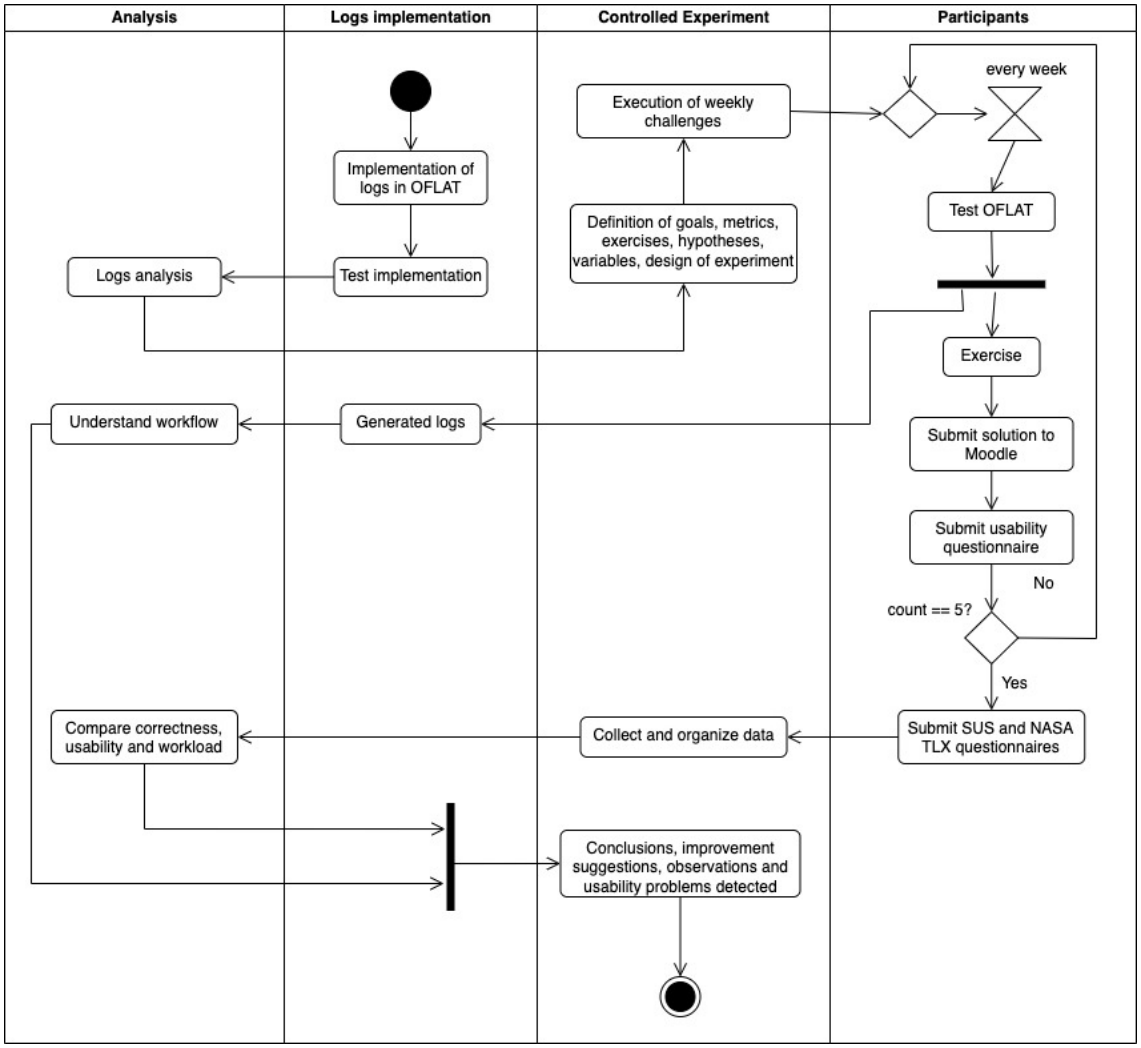


Figure 6.3: Activity diagram of the controlled experiment

6.5.1 Preparation

Logs were implemented in the OFLAT tool, as explained in the following subsection.

Then, the goals, metrics, tasks, hypotheses, variables, and this design were defined, and the three weekly challenges were developed.

6.5.1.1 Data logging

As previously mentioned, Segment (section 5.1) is a powerful tool to track the actions of users. Inside the Segment Workspace, one can set up and manage sources and destinations, manage the data schema that interfaces send, and test and monitor that data for errors.

Firstly, code snippets were directly applied to the OFLAT website and used as sources of the tracked event. Those events are sent to a PostgreSQL database and directly analyzed

using Google Analytics.

That implementation was tested by simulating different scenarios of OFLAT usage and all the buttons related to tasks performed during the controlled experiment traced data.

In Figure 6.4, it is possible to see that when an OFLAT user clicks on the button for creating a new model, i.e., a finite automaton, a regular expression or context-free grammar, a new entry is shown in the Segment Workspace (1). It is a log of type "Track", it has the name "Button Clicked" and it has a timestamp. With the "Pretty" tab open, the code that appears is the same implemented in the OFLAT code, and it helps to identify which button was clicked since each button has a different type description.

If the tab "Raw" is selected, the view is much more detailed (2). The "anonymousId" value is randomly assigned by Segment. It is the same for each user session, so if different users are using OFLAT simultaneously, it is possible to identify to whom belongs which received log.

Also, if two consecutive log entries are from the same user (by checking if it has the same "anonymousId" value), it is easier to check how much time an action took (3).

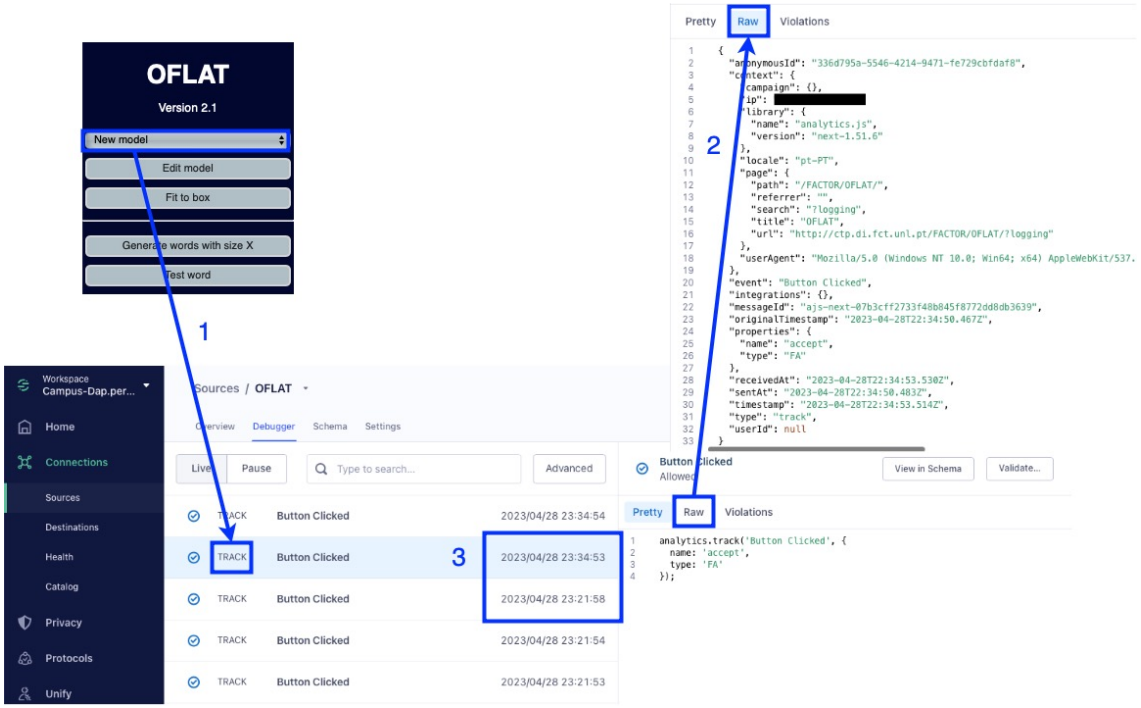


Figure 6.4: Segment workspace overview with logs example explained

6.5.2 Procedure

To publicize the need for volunteers, some direct emails were sent through CLIP to reach students who wanted to complement their studies. Those emails consisted of a brief

explanation of the experiment and a link to a Moodle page, where volunteers could solve the weekly challenges autonomously.

Participants needed to solve the suggested exercises and then submit the solutions to Moodle to collect the wanted metrics. While using OFLAT, some logs are generated throughout the experiment, so they can be then analyzed to help better understand the users' workflow.

After submitting the solutions, the students are asked to submit the usability questionnaire consisting of questions to understand better the user profile and some suggestions for improvement of this tool under study.

Then the students are asked to answer SUS and NASA TLX questionnaires.

Then, finally, the data is collected, organized, and analyzed by comparing the response correctness of each question and calculating the perceived usability and workload scores.

This should result in the ability to reach conclusions and summarize improvement suggestions.

6.5.3 Deviations

Although the ideal setting for the experiment would be a weekly challenge along the 6 sequential weeks of challenges, we have set 3 challenges due to organizational constraints organized in non-sequential weeks. This situation was primarily due to the burden of both time and effort the students would have to spend conflicting with other courses. Besides, these challenges could not be compulsory, considerably reducing the set of potential subjects. There were fewer volunteers than the registered students in the course (around 245).

7 Analysis of the controlled experiment

7.1 Dataset preparation

Each student answered four questions of each weekly challenge that consisted of simple exercises of DFAs (First challenge), Regular Expressions (Second challenge), and Context-Free Grammars (Third challenge) throughout the usage of the tool during the completion of the exercises. These answers make it possible to check if the students knew how to use OFLAT tool.

In addition to each weekly challenge, there was a questionnaire about the tool's usability where the students would give their opinions about the tool after completing each weekly challenge. In the third weekly challenge, there was one more questionnaire where the students answered the SUS and NASA TLX for OFLAT. Since the experiment data was collected with Moodle, it was extracted into an Excel sheet and analyzed.

An Excel sheet was designed to match each student with the correctness of each question.

Regarding the correctness, we consider *N* (No) if the word from Accepted words or Rejected words did not pass and *Y* (Yes) if the word from Accepted words or Rejected words passed.

The SUS and NASA TLX values were also moved to a different Excel sheet so the calculation of their scores could be analyzed individually and easily by applying some formulas for adding cells' values.

It is important to note that the NASA TLX questionnaire has been altered from the previous one in subsection 2.4.4 for practical reasons. The scale from 1 to 20 in Moodle is not noticeable, so the scale adopted was the most visible to students, and the same as the SUS questionnaire, from 1 to 5. In the original questionnaire, the scale of the question "How successful were you in accomplishing what you were asked to do?" goes from "perfect" to "failure". For the execution of this questionnaire, the scale was not changeable between questions, so the answer option was 1 (very low) to 5 (very high). This means that answers closer to 5 are expected from this question for a positive result, contrasting with the rest of the questionnaire. For the purposes of calculating the total score, students' responses to this question were converted to the inverse scale: those who answered 5

equals 1 (and vice versa), those who answered 4 equals 2 (and vice versa), and remain if the answer of the person who answered 3. In this way, lower results can be expected, corresponding to a better classification, and higher results, corresponding to a worse classification, maintaining consistency between all the questions in the questionnaire.

The results related to correctness presented in the following sections only consider the students' experiments that answered at least one exercise of one of the weekly challenges. So, it is regarded as 70 students for correctness and 12 for SUS and NASA TLX.

These results will be compared to the results related to correctness, usability and workload from the previous controlled experiment [41].

The data was then converted into the *.csv* format to be interpreted by Python code and obtain different plots. It was also organized into IBM SPSS [26] *.sav* files.

7.2 Analysis procedure

Firstly, some descriptive statistics were calculated to understand the collected data better and analyze and test the formulated hypotheses. Statistical measures like mean, standard deviation, skewness, and kurtosis are represented in Table 7.1, as well as the results from the Shapiro-Wilk, which helps to intercept data by verifying its normality or confirming its non-normality depending on the obtained value.

For data that followed a normal distribution, Welch's t-test was applied, and for data that did not follow a normal distribution, the Mann-Whitney U test was used [44].

The IBM SPSS system supported all of these decisions.

The following tables and plots summarize data and facilitate its interpretation:

- Figure 7.1: bar plot with the percentage of correct answers by the question and by the weekly challenge;
- Figure 7.2: bar plot with the number of submissions of each exercise of the weekly challenge;
- Figure 7.3: bar plot of submissions of each usability questionnaire;
- Table 7.2: table with percentages of correct answers in each exercise based on the number of submissions;
- Figure 7.4: box plot of every SUS question score by version;
- Figure 7.5: bar plot of the mean SUS score by question and by version;
- Figure 7.6: box plot of SUS score by version including adjective rating table;
- Figure 7.7: violin plot of NASA-TLX score by version;
- Figure 7.8: bar plot of NASA-TLX mean scores by question and by version.

7.3 Descriptive statistics

As explained above, the following results reflect the 70 responses from students who answered at least one question from one of the weekly challenges of the tool in terms of correction. For the SUS and TLX, all 12 responses are taken into account.

The results from the previous experiment [41] will be shown as the complement to the comparison information. These results reflect all 21 responses for correctness, SUS and TLX.

The Shapiro-Wilk test reveals results lower than 0.05 for the correctness data of version 2.1, which suggests that its distribution does not follow a normal distribution [47, 50]. SUS and TLX follow a normal distribution. This is important to define how to test hypotheses.

	Version	N	Mean	Std. Deviation	Skewness	Kurtosis	Shapiro-Wilk
Corr.	2.1	70	4.54	2.73	0.61	-0.44	**0.0001**
	1.3	21	0.81	0.20	-0.93	-0.10	0.84
SUS	2.1	12	72.71	14.87	-0.74	0.09	0.95
	1.3	21	58.21	9.26	-1.06	1.54	0.91
TLX	2.1	12	15.08	3.34	1.02	2.04	0.94
	1.3	21	12.71	3.27	0.24	1.13	0.96

Table 7.1: Descriptive statistics

Analyzing the graph referring to the percentage of correct answers in each question (Figure 7.1), it is noticed that almost all students who submitted their solutions managed to get the correct answer.

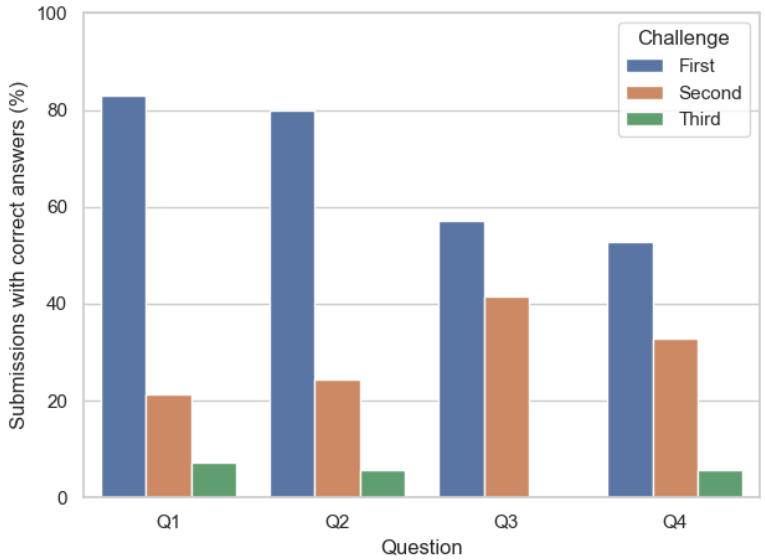


Figure 7.1: Bar plot of correct answers in each question by weekly challenge

Analyzing the graph referring to the number of submissions in each exercise (Figure 7.2), it is noticed that the number of submissions regarding the exercises of the third challenge is very low compared to the exercises of the first and second challenges. These exercises were optional, i.e., students were not forced to solve the exercises.

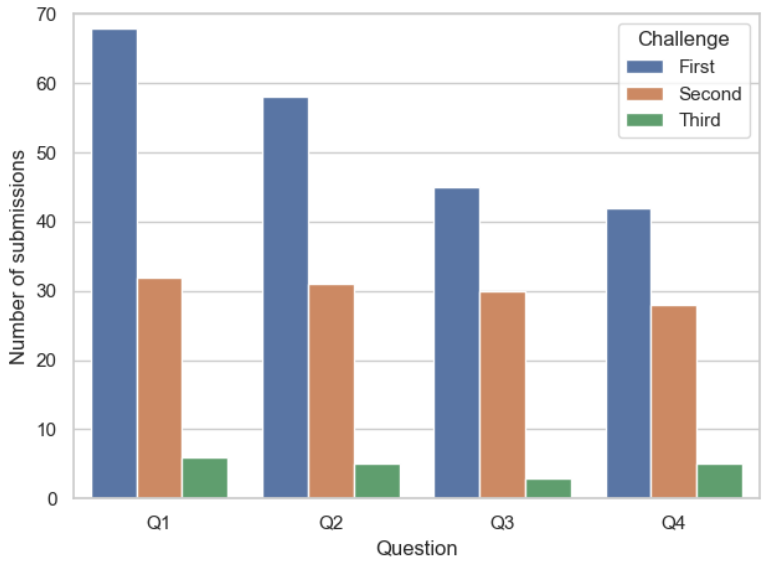


Figure 7.2: Bar plot of submissions of each exercise of the weekly challenge

Analyzing the graph referring to the number of submissions of usability questionnaire (Figure 7.3), it is noticed that the number of submissions decreased throughout each challenge. Just like the weekly challenges' exercises, these usability questionnaires were also optional.

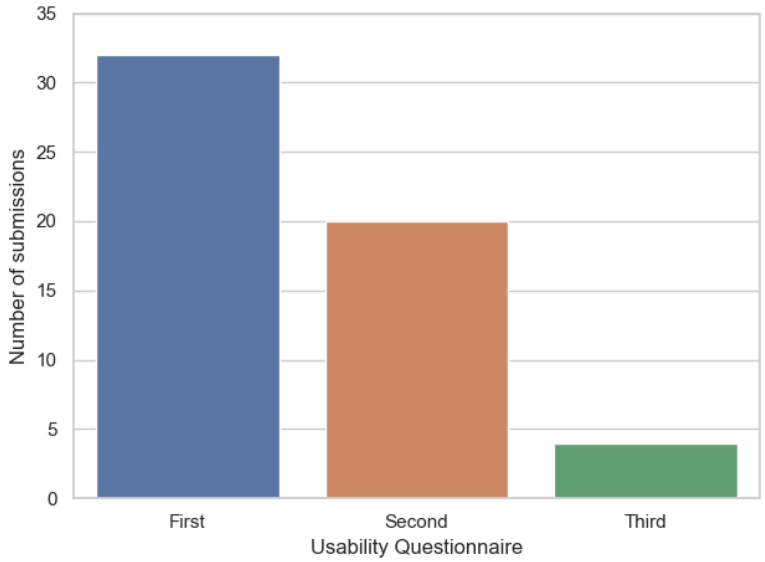


Figure 7.3: Bar plot of submissions of each usability questionnaire

Table 7.2 summarizes the information regarding the correctness of the answers given (in percentage).

Question	Correct Answers (%)
Q1.1. Specify a DFA over the alphabet {a,b,c} for the language of words with an odd number of occurrences of the substring "ab"	98.3
Q1.2. Specify a DFA over the alphabet {a,b,c} for the language of words in which the total number of b's and the total number of c's is two	99.5
Q1.3. Specify a DFA over the alphabet {0,1} for the language of words that represent the naturals that are multiple of 3	93.3
Q1.4. Specify a DFA over the alphabet {0,1} for the language of words that represents the regular expression $(01 + 001 + 010)^*$	96
Q2.1. Define a regular expression over the alphabet {0,1} which defines the language over it of all words that do not contain the substring "000"	84.1
Q2.2. Define a regular expression over the alphabet {a,b} whose language is the set of words in which every "a" is immediately preceded or followed by "b"	87.6
Q2.3. Define a regular expression over the alphabet {a,b,c} whose language is the set of words with even length that contains precisely one "a"	98.9
Q2.4. Define a regular expression over the alphabet {a,b} whose language is the set of words in which the substring "aa" occurs at least twice	94.5
Q3.1. Define the context-free grammar for the language $\{0^m 1^n 0^{m-1} \mid m \in \mathbb{N} \wedge n \in \mathbb{N}_0\}$	98.6
Q3.2. Define the context-free grammar for strings not of the form $0^i 1^j$ where i = j and i, j $\geq$ 0	80
Q3.3. Define the context-free grammar for generating all strings that are regular expressions in the alphabet {a, b, (,), +, *, ϵ}	70
Q3.4. Define the context-free grammar for $\{0^{2n} 1^k 2^n \mid n, k \in \mathbb{N}_0\}$	80

Table 7.2: Percentage of correct answers in each exercise based on the number of submissions

The following graphs present results related to the SUS questionnaire. For this, the same questions are recalled in Table 7.3.

Question
1. I think that I would like to use this system frequently.
2. I found the system unnecessarily complex.
3. I thought the system was easy to use.
4. I think that I would need the support of a technical person to be able to use this system.
5. I found the various functions in this system were well integrated.
6. I thought there was too much inconsistency in this system.
7. I would imagine that most people would learn to use this system very quickly.
8. I found the system very cumbersome to use.
9. I felt very confident using the system.
10. I needed to learn a lot of things before I could get going with this system.

Table 7.3: SUS Questions

Answers to these questions can be given on a scale of 1 to 5, where 1 represents complete disagreement, and 5 represents the entire agreement. Odd questions are supposed to have classifications near 5, contrary to the even ones that are supposed to have classifications near 1, thus if the tested system offers a good usability experience.

By the chart of Figure 7.4, the average of the answers given is significantly different from one version to the other.

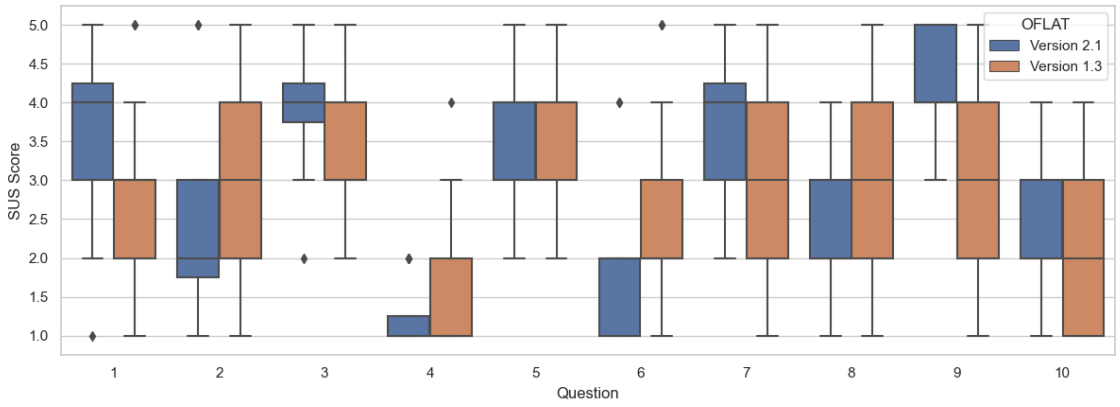


Figure 7.4: Box plot of every SUS question score by version

To complement the interpretation of these results, a more general view can be obtained with Figure 7.5. For the version 2.1, the average of the answers given in the odd questions varies between 3.5 and 4.33, and the average of the answers given in the even questions varies between 1.25 and 2.42. For the version 1.3, the average of the answers given in the odd questions varies between 2.76 and 3.57, and the average of the answers given in the even questions varies between 1.76 and 3.

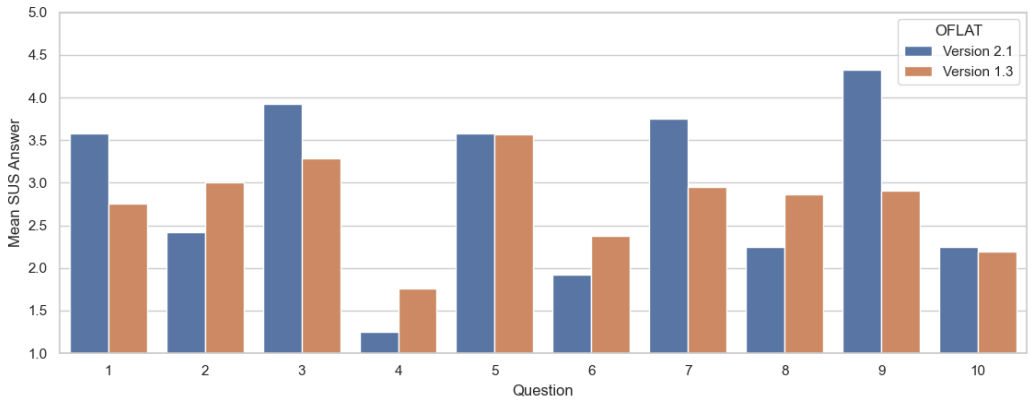
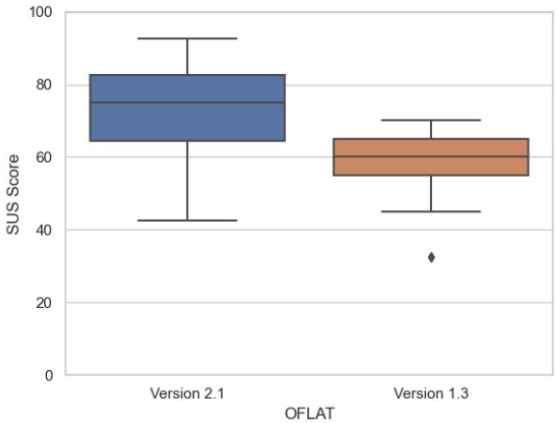


Figure 7.5: Bar plot of the mean SUS score by question and by version

To have an overview of the final score and interpret this value with a classification adjective, Figure 7.6 was generated.

The version 2.1 scores are dispersed because most of the values are higher than 50. Thus, it has a high median, resulting in a final classification of 75. It is in the range of 68-80.3, corresponding to a B grade and adjective rating "Good". As for the version 1.3, the scores are less dispersed and it has a lower median than the version 2.1, resulting in a final classification of 60. It is in the range 51-68, corresponding to grade D and adjective rating "Poor".



SUS Score	Grade	Adjective Rating
> 80.3	A	Excellent
68 - 80.3	B	Good
68	C	Okay
51 - 68	D	Poor
< 51	F	Awful

Figure 7.6: Box plot of SUS score by version including adjective rating table

As for the NASA TLX results, the vision through the violin plot in Figure 7.7 allows for observing symmetries and asymmetries of the results.

For example, students concentrated their opinions more on levels 1 and 2, resulting in agreement on a greater physical demand using OFLAT (version 2.1).

As for temporal demand, students considered average temporal effort when using OFLAT to solve the weekly challenges.

Students concentrated their opinions more on level 2, resulting in good feedback in terms of performance and frustration level.

Overall, the effort felt when using OFLAT to solve weekly challenges was similar to the mental demand.

In terms of mental demand, temporal demand, and frustration, we may observe a notable difference when compared to the results of the version 1.3.

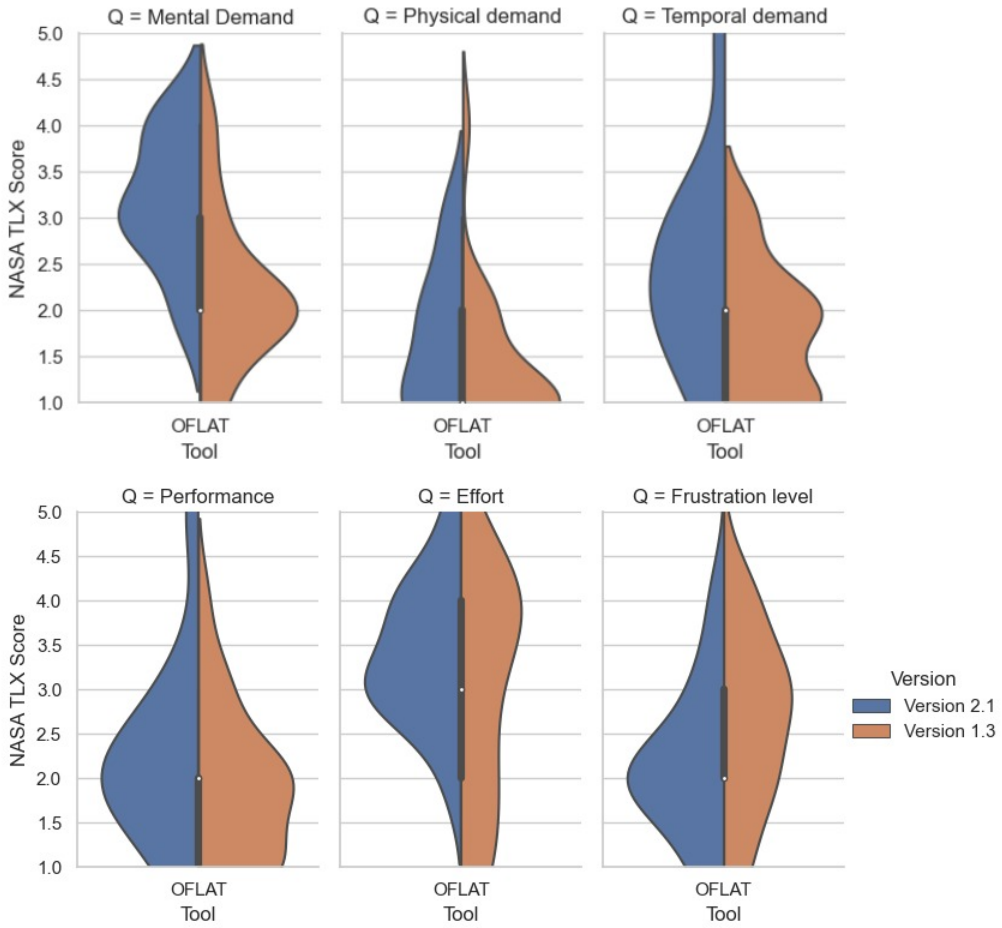


Figure 7.7: Violin plot of TLX score by version

To complement the interpretation of these results, a more general view can be obtained with Figure 7.8. The answer's average in the version 2.1 is slightly higher in questions regarding mental demand and effort. The answer's average in the frustration level is slightly lower than the version 1.3. Remember that the higher the score, the worse the feedback on the tool.

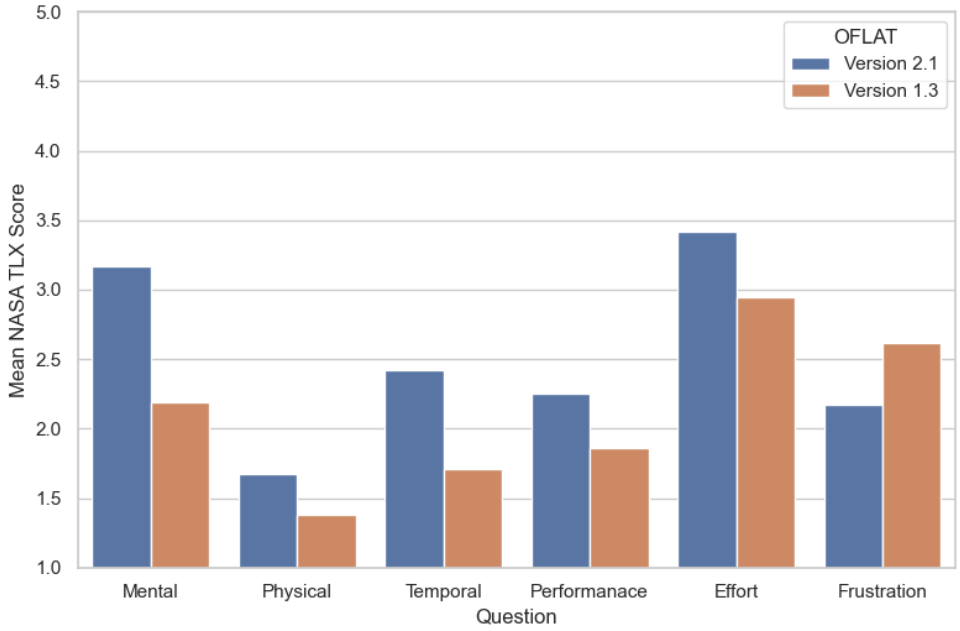


Figure 7.8: Bar plot of TLX mean scores by question and by version

7.4 Hypotheses testing

To carry out the hypothesis tests regarding usability and workload, the results of the SUS questionnaire, and the NASA TLX were submitted to Welch's test. The SUS questionnaire and NASA TLX present a Shapiro-Wilk p-value greater than 0.5 and follow a normal distribution. In contrast, the correctness presents a Shapiro-Wilk p-value smaller than 0.05 and does not follow a normal distribution.

According to Welch's test (Table 7.4), the NASA TLX p-value is more significant than 0.05, while the SUS p-value is less significant than 0.05. This leads to conclusions that, under these conditions, the previously defined null hypothesis regarding NASA TLX cannot be rejected and the previously defined null hypothesis regarding SUS can be rejected (Table 7.5).

	V2.1 Mean	V1.3 Mean	Difference	95% Dif. CI Lower	95% Dif. CI Upper	t	df	p-value
SUS	72.71	58.21	14.5	-24.55	-4.44	-3.06	15.96	**0.008**
TLX	15.08	12.71	2.37	-4.78	0.04	-2.05	20.63	**0.054**

Table 7.4: Welch's t-test scores

Accepted	$H_{1\ SUS}$: SUS means of versions 2.1 and 1.3 are not the same. $H_{0\ TLX}$: NASA TLX means of versions 2.1 and 1.3 are the same.
Rejected	$H_{0\ SUS}$: SUS means of versions 2.1 and 1.3 are the same. $H_{1\ TLX}$: NASA TLX means of versions 2.1 and 1.3 are not the same.

Table 7.5: Hypotheses acceptance and rejection for usability and workload

Unlike the usability and workload p-values, the Shapiro-Wilk correctness p-value is smaller than 0.05. Not following a normal distribution, the non-parametric Mann-Whitney U test [33] was performed (Table:7.6), since there are always two groups of samples, one referring to version 1.3 and the other to version 2.1 results.

	U	Z	p-value
Correctness	28704.00	-0.402	**0.688**

Table 7.6: Mann-Whitney U test results

Since the p-value from the Mann-Whitney U test is greater than 0.05, the null hypothesis cannot be rejected. There is no sufficient evidence that correctness is different between the two versions (Table 7.7).

Accepted	$H_{0\ Correctness}$: OFLAT version 2.1 and OFLAT version 1.3 correctness are the same.
Rejected	$H_{1\ Correctness}$: OFLAT version 2.1 and OFLAT version 1.3 correctness are not the same.

Table 7.7: Hypotheses acceptance and rejection for correctness

Given the results of the statistical tests, the proposed hypotheses for this experiment can be accepted and rejected as shown in Table 7.8.

Accepted	- $\mathbf{H_{0\ Correctness}}$: The user correctness with the new version of OFLAT is **the same** as the correctness with the previous version. - $\mathbf{H_{1\ Usability}}$: The user's perception of usability with the new version of OFLAT is **significantly different** from the perception of usability with the previous version. - $\mathbf{H_{0\ Workload}}$: The user workload with the new version of OFLAT is **the same** as the workload with the previous version.
Rejected	- $\mathbf{H_{1\ Correctness}}$: The user correctness with the new version of OFLAT is **significantly different** from the correctness with the previous version. - $\mathbf{H_{0\ Usability}}$: The user's perception of usability with the new version of OFLAT is **the same** as the perception of usability with the previous version. - $\mathbf{H_{1\ Workload}}$: The user workload with the new version of OFLAT is **significantly different** from the workload with the previous version.

Table 7.8: Hypotheses acceptance and rejection

7.5 Process Mining analysis

As previously explained in subsubsection 6.2.1.1, when using OFLAT, students' actions are stored in the Segment Workspace, and those actions are sent to a PostgreSQL database.

When those actions were sent to a PostgreSQL database, the following tables were created: button_clicked, pages, and tracks.

For the tables button_clicked, tracks, and pages, the total number of registered logs are, respectively, 11492, 11492, and 282.

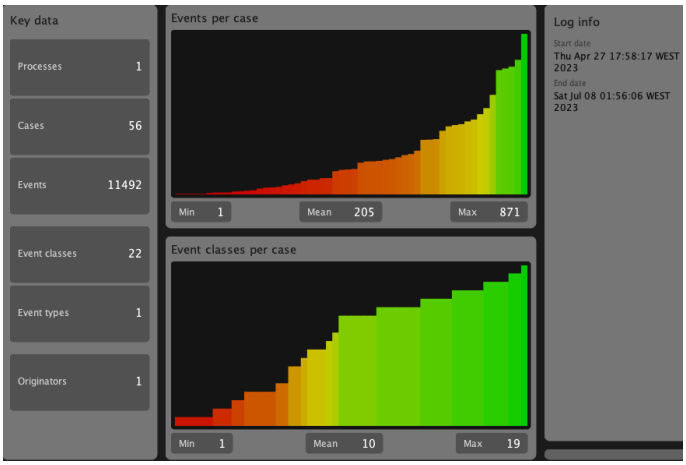
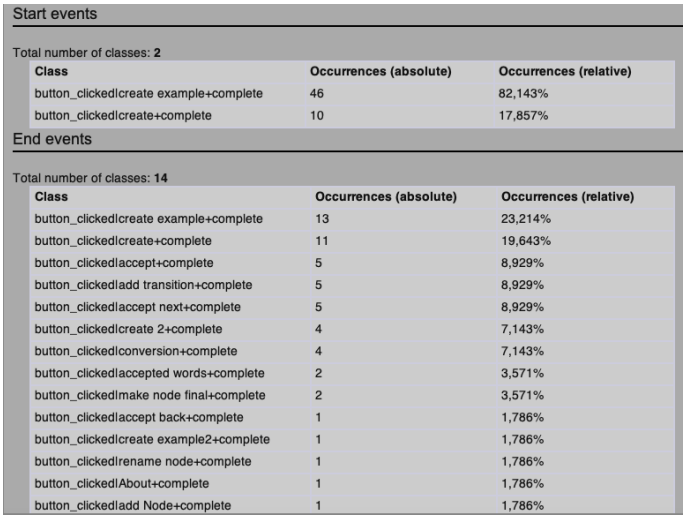


Figure 7.9: Button Clicked bar plot

In the Figure 7.9, it was selected the "anonymousId" as the case column and for the event column, it was selected "event" and "name". In this case, "event" is button_clicked, and "name" is the type of button clicked.



Start events

Total number of classes: **2**

Class	Occurrences (absolute)	Occurrences (relative)
button_clicked|create example+complete	46	82,143%
button_clicked|create+complete	10	17,857%

End events

Total number of classes: **14**

Class	Occurrences (absolute)	Occurrences (relative)
button_clicked|create example+complete	13	23,214%
button_clicked|create+complete	11	19,643%
button_clicked|accept+complete	5	8,929%
button_clicked|add transition+complete	5	8,929%
button_clicked|accept next+complete	5	8,929%
button_clicked|create 2+complete	4	7,143%
button_clicked|conversion+complete	4	7,143%
button_clicked|accepted words+complete	2	3,571%
button_clicked|make node final+complete	2	3,571%
button_clicked|accept back+complete	1	1,786%
button_clicked|create example2+complete	1	1,786%
button_clicked|rename node+complete	1	1,786%
button_clicked|About+complete	1	1,786%
button_clicked|add Node+complete	1	1,786%

Figure 7.10: Start event and End event

It is possible to infer from the Figure 7.9 and the Log Summary of the Figure 7.10 that students began the exercises in OFLAT when they clicked the "create" button or the "create example" button. They concluded those exercises when the final action was taken, such as "accept", "add transition", "make node final", "accepted words", etc.

Figure 7.11: Pages bar plot

In the Figure 7.11, it was selected the "anonymousId" as the case column, and "context_ip" as the event column.

By analyzing the Figure 7.11, it can be concluded that most students who solved the exercises in OFLAT used the suffix "*?logging*" in the following URL: `http://ctp.di.fct.unl.pt/FACTOR/OFLAT/?logging`.

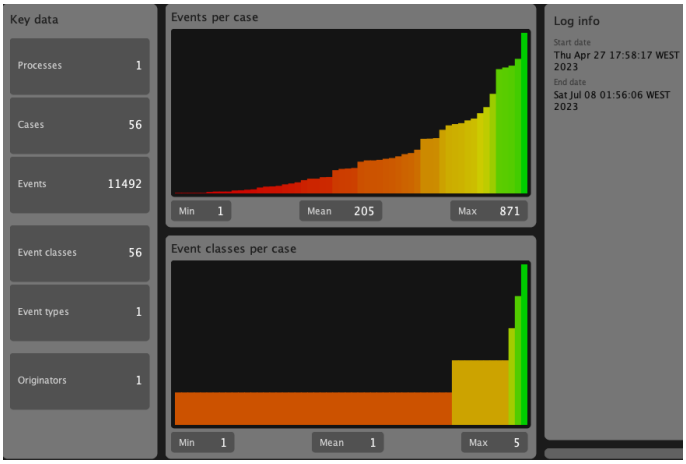


Figure 7.12: Tracks bar plot

In the Figure 7.12, it was selected the "anonymousId" as the case column and for the event column, it was selected the "event" and "context_ip".

The events contained in the table tracks and the events stored in the table button_clicked are identical, as can be inferred from the analysis of the previous figure.

Overall, the collected data from logs is significant for the experiment.

More results of Process Mining are shown in the Appendix C with different combinations.

7.6 Students' grades analysis

The majority of students who completed the weekly challenges got good grades, according to an analysis of the students' final grades of the Theory of Computation course based on the exercises of the challenges and usability questionnaires.

7.7 Discussion

7.7.1 Evaluation of results and implications

With the results presented, it is now possible to answer the questions previously defined in subsection 6.3.1.

- **Q1.1: Is OFLAT error-prone?**

  By accepting the hypothesis that using OFLAT does not influence answering correctly to the tasks and by checking the Table 7.2, it is clear that the percentage of correctness is high in OFLAT.

  These results then suggest that OFLAT is not error-prone.

- **Q1.2: Is OFLAT easy to use?**

Regarding the ease of usability, the hypothesis that using OFLAT does not influence the perceived usability of the tool was accepted. Results suggest that OFLAT is easy to use.

- **Q1.3: Does OFLAT demand workload?**

  Similarly to usability, the study on workload demand had the same outcome. Results suggest that OFLAT's workload is demanding since the null hypothesis regarding workload was accepted.

7.7.2 Other results

In addition to the results already presented, other results of the usability questionnaires are presented next.

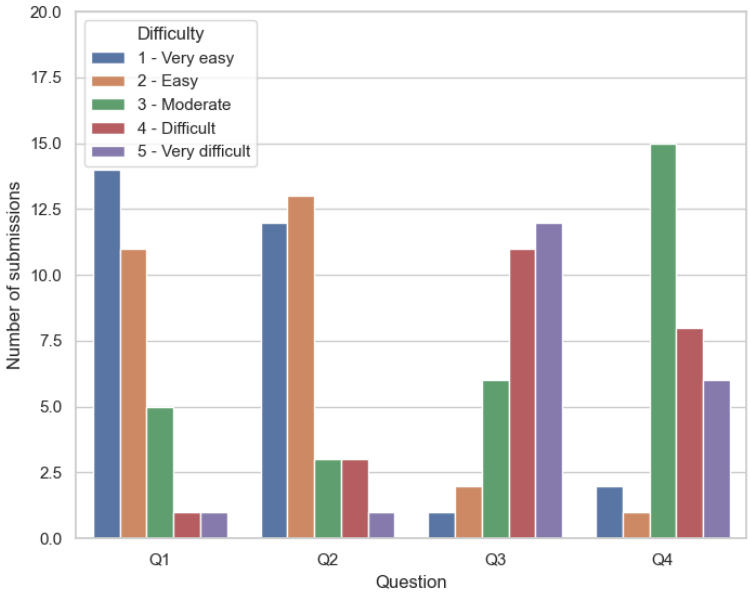


Figure 7.13: Difficulty of each exercise of the first weekly challenge

The complexity of each exercise in the first weekly challenge is depicted in a bar plot in Figure 7.13.

Most students thought the last two exercises were more complex than the first two, which they perceived as more straightforward.

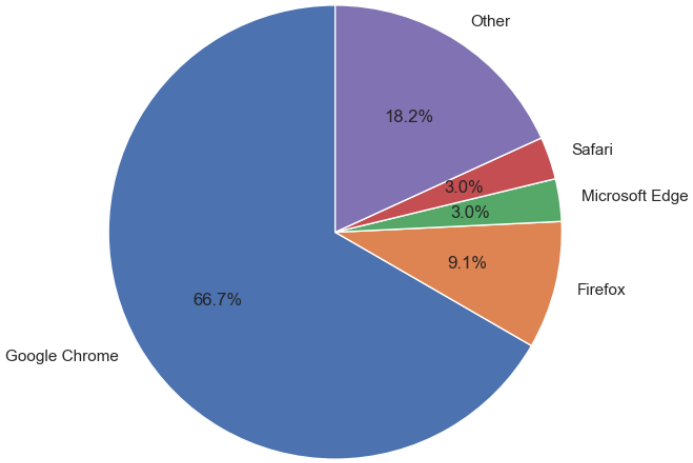


Figure 7.14: Browsers that students used for the first weekly challenge

The browser students used to complete the exercises of the first weekly challenge is depicted in the pie chart of Figure 7.14.

Most students utilized Google Chrome to complete the exercises for the first weekly challenge.

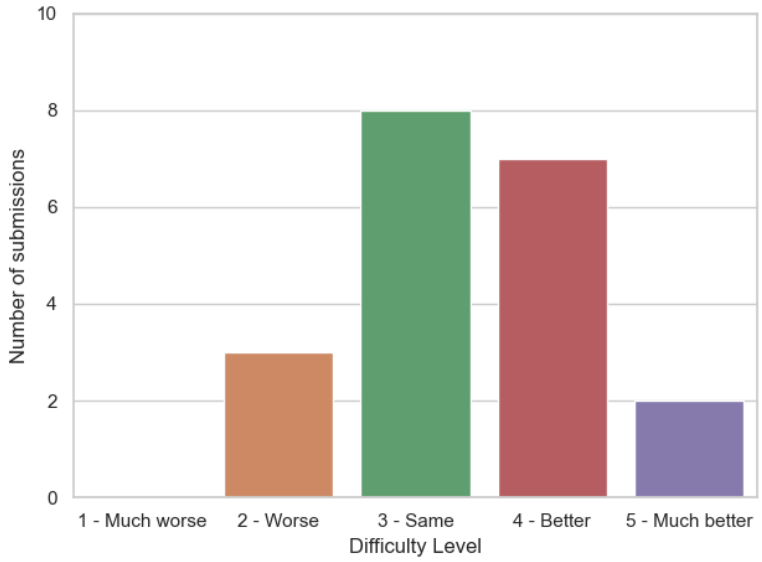


Figure 7.15: First and second challenge difficulty comparison

Students' perceptions of the difficulty of the second weekly challenge in comparison to the first are depicted in a bar plot in Figure 7.15.

Most students believed the second weekly challenge was just as tricky as the first, although some said the second was more accessible.

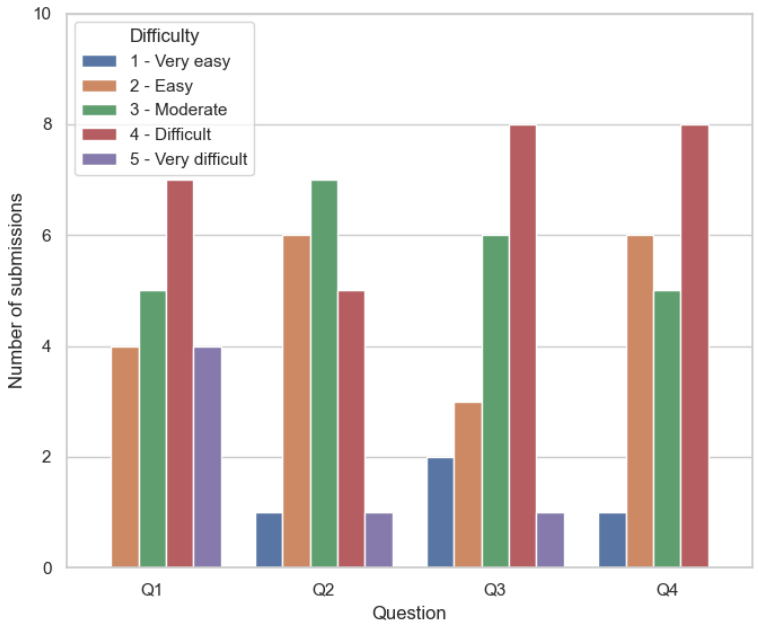


Figure 7.16: Difficulty of each exercise of the second weekly challenge

The complexity of each exercise in the second weekly challenge is depicted in a bar plot in Figure 7.16.

The majority of the students appeared to find the exercises to be challenging.

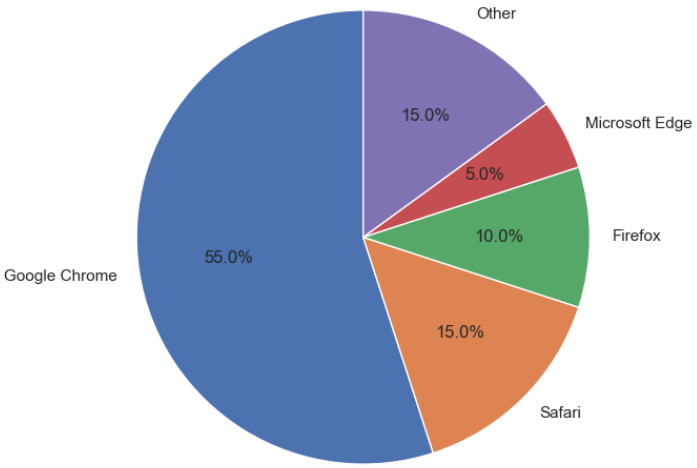


Figure 7.17: Browsers that students used for the second weekly challenge

The browser students used to complete the exercises of the second weekly challenge is depicted in the pie chart of Figure 7.17.

The majority of students utilized Google Chrome to complete the exercises for the second weekly challenge.

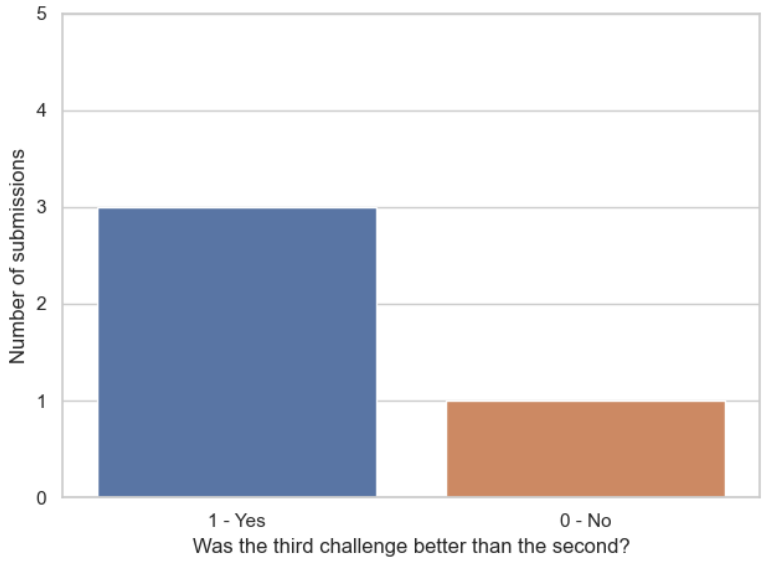


Figure 7.18: Second and third challenges difficulty comparison

Students' perceptions of the difficulty of the third weekly challenge in comparison to the second are depicted in a bar plot in Figure 7.18.

Most students believed that the third weekly challenge was better than the second.

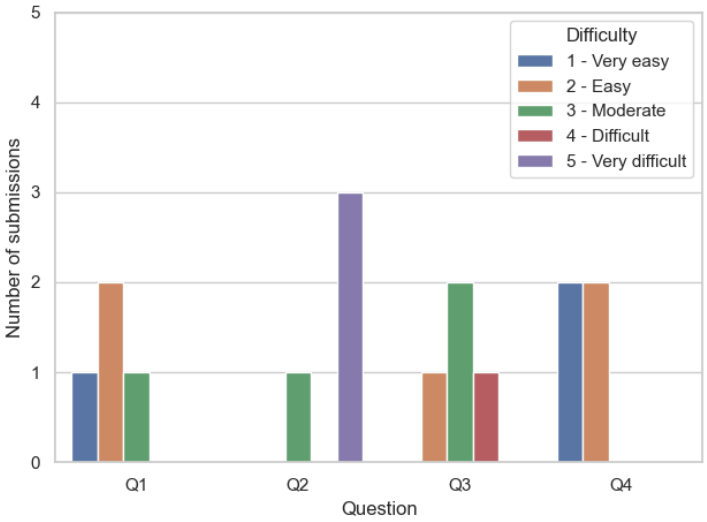


Figure 7.19: Difficulty of each exercise of the third weekly challenge

The complexity of each exercise in the third weekly challenge is depicted in a bar plot in Figure 7.19.

Most students said the second activity was the hardest, and the first and fourth exercises were more straightforward.

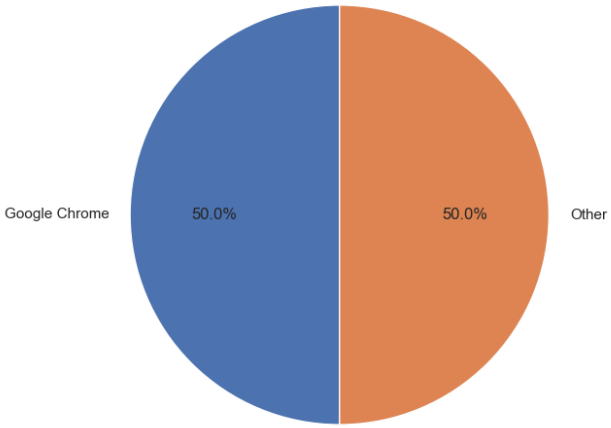


Figure 7.20: Browsers that students used for the third weekly challenge

The browser students used to complete the third weekly challenge exercises is depicted in the pie chart of Figure 7.20.

Google Chrome was used by half of the students, and another browser by the other half.

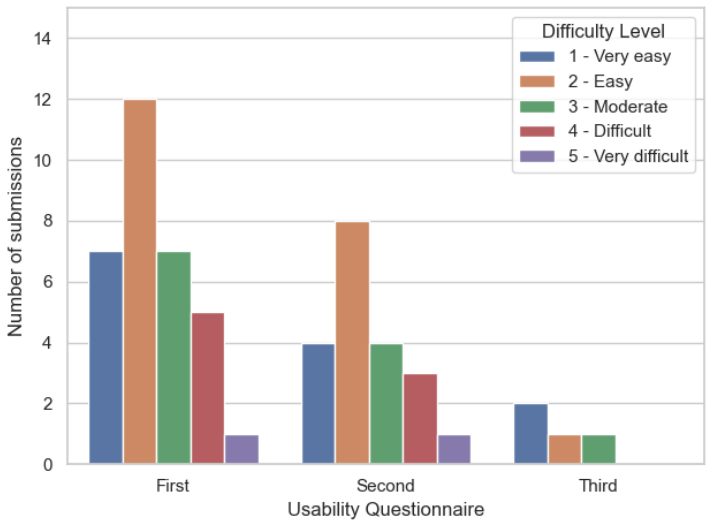


Figure 7.21: Evaluation of OFLAT in terms of usability

Students' perceptions of OFLAT's usability are depicted in a bar plot in Figure 7.21.

It is clear to see that the majority of students thought OFLAT was easy to use.

Students mentioned suggestions and improvements, and it is relevant to mention the following:

- Some transitions could automatically be better positioned;
- Being able to move the Start state;
- Remove states with backspace/delete button;
- The addition of "Finite epsilon-transition Automata";
- The addition of the "undo" button to undo the last change;
- Being able to do 1^+, instead of 11^* in regular expression.

It is also relevant to mention the following observations:

- Renaming a state makes all the existing states go all over the place;
- States being hidden from the user but remain in the model specification;
- OFLAT cannot do tests with words that are too long in regular expressions.

7.7.3 Threats to validity

An evaluation of threats to validity was carried out, according to Wohlin et al.[55], regarding internal, construction, conclusion, and external validity.

7.7.3.1 Internal validity

Internal validity has to do with examining causal relationships. Specifically, it looks at whether or not an experimental treatment or condition makes a difference and whether there is data to back up the claim.

Although the exercises of the weekly challenges were previously tested before being accessible to students, we were monitoring and ready to promptly react to any detection of problems in the provided material to avoid some misinterpretation that could put the experiment's results at stake. For instance, a minor flaw was detected in the third challenge when the exercises were presented to the students.

Since the minor flaw was solved and no more flaws were detected, there were no more significant threats to internal validity.

7.7.3.2 Conclusion validity

Conclusion Validity deals with the degree to which conclusions reached are reasonable within the data collected.

For this usability study, it was not identified threats that could affect the conclusion's validity.

7.7.3.3 Construction validity

Construction validity deals with whether or not we measure what is intended to be tested, which is how well a test or experiment lives up to its claims.

For this usability study, it was not detected threats that could affect the construction validity.

7.7.3.4 External validity

External validity refers to the extent to which the results of a study can be generalized or applied to a larger population, settings, or conditions beyond the specific context of the study. It is a measure of how well the results of a study can be considered representative of the real world [15].

For this usability study, it was not identified threats that could affect the external validity.

7.7.4 Inferences

The detected blocking situations felt by students are:

- Having to hold down the right mouse button makes it hard to create an automaton;
- Being limited in the length of words to test "medium-length" regular expressions, which forced students to convert the regular expressions into automatons;
- Some students were not able to write explicitly the ϵ (epsilon) when they wanted to create a regular expression, e.g., ϵ+1;
- When students renamed one of the states of the automaton, they had to rearrange all the states again in a more readable way.

OFLAT actions covered by the proposed exercises are less error-prone.

The usability results are good because the average SUS score is grade B and evaluated as good.

As suggestions for improvement and students' observations, the ones in Table 7.9 stand out:

Suggestions of improvement
- Some transitions could automatically be better positioned - Being able to move the Start state - Remove states with backspace/delete button - The addition of "Finite epsilon-transition Automata" - The addition of the "undo" button to undo the last change - The ideal would be to have a button to put state insertion mode - The transformation of automata from NFA to DFA should be effective in the model to be usable in the pipeline. - Being able to do 1^+, instead of 11^* in regular expression
Observations
- Renaming a state makes all the existing states go all over the place - States being hidden from the user but remain in the model specification - OFLAT is not able to do tests with words that are too long in regular expressions

Table 7.9: Suggestions of improvement for OFLAT and students' observations

In addition to the suggestions and observations given by the students, usability problems were detected throughout the experiment as shown in the Table 7.10:

Usability problems detected
- Lack of memory - The color does not show in the states with names 'AB Odd' or any other name with space on it - Reserved symbols that the user does not know - Unable to import the same model again after closing the window - Unable to Copy and Paste - Adding states should increase the state labels not to be all 'A'. It makes it uncomfortable when the user wants to model and it does not import the labels - "Some accept states do not belong to the set of all states" - "RegExpSyntax": Right-parenthesis expected - The page gets stuck when the user tries to generate words of size X in context-free grammar

Table 7.10: Usability problems detected

7.7.5 Improvement suggestions

Despite all the threats to validity presented, decisions were taken to minimize them as much as possible.

The implementation of the logs also allows the evaluation of the workflow during the semester in which the TC course takes place, so it is possible to collect more reliable results regarding the "free" experience of the users, contrary to the actions coming from the proposed weekly challenges.

8 | Conclusions and Future Work

In this chapter, conclusions and suggestions for future work are presented.

8.1 Conclusions

A Systematic Mapping Study on usability studies for FLAT tools detected a gap in this research topic. As a result of this SMS, it was decided to use JFLAP as a reference because it is the most complete and similar tool.

The empirical evaluation was carried out, and OFLAT was evaluated using a controlled experiment designed and executed. OFLAT has usability issues, and improvement proposals were presented, considering users' experiences, suggestions, and observations.

It contributed to this experiment's result, logs were implemented and are available for data collection in future studies. In this way, the actual workflow of the tool usage can be understood without usability being conditioned to a set of exercises proposed to students.

The goal of using process mining in this dissertation was to analyze the event logs generated in OFLAT to see if there were bottlenecks or inefficiencies in the functionalities.

Although the presented results are statistically significant, if it is desired to repeat the experience with even more students, ideally, they should be from other Universities (including from other countries). The design and the current results are used as a comparison baseline to assess the evolution of the tool (and further usability improvements). This present document and additional resources have provided all the information that was required [36].

Regarding usability, it is concluded that, generally, OFLAT is less error-prone. The results are satisfactory since the average SUS score is evaluated with a good grade.

Among others, one of the key suggestions pointed out by most students is to move the "Start" state of the automaton. Also, every time a state of the automaton is renamed, all states are scattered across the screen.

By implementing the presented improvement suggestions and considering the students' opinions, it is expected to obtain a more positive and beneficial user experience for the OFLAT users.

This summarizes the results obtained given the proposed objectives at the beginning of this study. The goals are then considered to have been met.

8.2 Future work

This experiment will be repeated in the future with the logs mechanism, with new functionalities implemented in OFLAT, such as Pushdown Automata and Turing Machines, and with some existing functionalities improved based on the improvement suggestions, observations, and detected usability problems mentioned in the previous chapter (subsection 7.7.4).

After the usability study, the reengineering of the tool will be made, just as described in [39]. The reengineering of OFLAT consists of improving the existing functionalities, fixing some bugs, and improving the user experience.

Bibliography

[1] W. van der Aalst. *Process Mining*. Springer-Verlag Berlin Heidelberg, 2016. ISBN: 978-3-662-49850-7 (cit. on p. 13).

[2] M. Armoni et al. *Panel Proposal: Automata Theory-Its Relevance to Computer Science Students and Course Contents*. Vol. 38. 2006, p. 197. ISBN: 1595932593. DOI: 10.1145/1124706.1121403 (cit. on p. 1).

[3] *Automata Tutor v3.0*. URL: https://automata-tutor.model.in.tum.de/index. (visited on 14/11/2022) (cit. on p. 24).

[4] *Automaton Simulator*. URL: https://automatonsimulator.com/. (visited on 14/11/2022) (cit. on p. 25).

[5] V. R. Basili, G. Caldiera, and H. D. Rombach. "The goal question metric approach". In: (1994), pp. 1–10 (cit. on p. 56).

[6] *Box plot - Khan Academy*. URL: https://www.khanacademy.org/math/statistics-probability/summarizing-quantitative-data/box-whisker-plots/a/box-plot-review. (visited on 13/01/2023) (cit. on p. 11).

[7] J. Brooke. *SUS: A quick and dirty usability scale*. Nov. 1996 (cit. on p. 10).

[8] D. P. Bunde. "Using Real Examples to Motivate Automata Theory". In: *J. Comput. Sci. Coll.* 35.5 (Oct. 2019), pp. 28–36. ISSN: 1937-4771 (cit. on p. 2).

[9] C. I. Chesñevar, M. P. González, and A. G. Maguitman. "Didactic Strategies for Promoting Significant Learning in Formal Languages and Automata Theory". In: *SIGCSE Bull* 36.3 (June 2004), pp. 7–11. ISSN: 0097-8418. DOI: 10.1145/1026487.1008002. URL: https://doi.org/10.1145/1026487.1008002 (cit. on p. 2).

[10] *Coursera*. URL: https://www.coursera.org/learn/process-mining/home/info. (visited on 11/04/2023) (cit. on pp. 13, 50).

[11] P. Crescenzi, L. Rossi, and G. Apollaro. "Making Turing Machines Accessible to Blind Students". In: *Proceedings of the 43rd ACM Technical Symposium on Computer Science Education. SIGCSE '12. Raleigh, North Carolina, USA: Association for Computing Machinery* (2012), pp. 167–172. DOI: 10.1145/2157136.2157190. URL: https://doi.org/10.1145/2157136.2157190 (cit. on p. 27).

[12] L. D'Antoni et al. "Automata Tutor v3". In: *Computer Aided Verification* (2020). Ed. by S. K. Lahiri and C. W. Cham, pp. 3–14 (cit. on p. 25).

[13] A. Dix et al. *Human-Computer Interaction*. 3rd ed. Pearson Education Limited, 2004. ISBN: 978-0-13-046109-4 (cit. on pp. 6, 7).

[14] *Empirical Software Engineering*. URL: https://www.cs.umd.edu/~basili/presentations/2006/Role%20of%20E%20in%20SE%20Irvine.pdf. (visited on 06/01/2023) (cit. on p. 8).

[15] *External Validity*. URL: https://researchmethod.net/external-validity/. (visited on 29/09/2023) (cit. on p. 86).

[16] *FACTOR*. URL: https://release.di.ubi.pt/factor/index.html. (visited on 28/12/2022) (cit. on p. 5).

[17] A. Fernandez, S. Abrahão, and E. Insfran. "A Systematic Review on the Effectiveness of Web Usability Evaluation Methods". In: *16th International Conference on Evaluation & Assessment in Software Engineering (EASE 2012)* 53.8 (2012), pp. 52–56. DOI: https://doi.org/10.1049/ic.2012.0007 (cit. on p. 19).

[18] A. Fernandez, E. Insfran, and S. Abrahão. "Usability evaluation methods for the web: A systematic mapping study". In: *Information and Software Technology* 53.8 (Aug. 2011), pp. 789–817. ISSN: 0950-5849. DOI: https://doi.org/10.1016/j.infsof.2011.02.007. URL: https://www.sciencedirect.com/science/article/pii/S0950584911000607 (cit. on p. 7).

[19] *Flap.js*. URL: https://flapjs.github.io/FLAPJS-WebApp/. (visited on 4/01/2023) (cit. on p. 25).

[20] *Fluxicon Disco*. URL: https://fluxicon.com/disco/. (visited on 11/04/2023) (cit. on p. 50).

[21] C. Freitas. "Pushdown Automata in OCAML-FLAT/OFLAT". In: (2023) (cit. on p. 16).

[22] J. Gal-Ezer and M. Trakhtenbrot. "Challenges in Teaching the Pumping Lemma in Automata Theory Course". In: *SIGCSE Bull* 37.3 (2005), p. 369. ISSN: 0097-8418. DOI: 10.1145/1151954.1067569. URL: https://doi.org/10.1145/1151954.1067569 (cit. on p. 2).

[23] M. Goyal and S. Sachdeva. "Enhancing Theory of Computation Teaching Through Integration with other Courses". In: *International Journal of Recent Trends in Engineering* 1.2 (2009) (cit. on p. 1).

[24] H. G. Henriques. "Domain Specific Language Evaluation: OutSystems' Business Process Technology". MA thesis. FCT-UNL, 2016 (cit. on p. 10).

[25] *IBM*. URL: https://www.ibm.com/topics/process-mining. (visited on 31/01/2023) (cit. on pp. 13, 14).

[26] *IBM SPSS*. URL: https://www.ibm.com/products/spss-statistics. (visited on 16/06/2023) (cit. on p. 67).

[27] *ISO - Usability*. URL: https://www.iso.org/obp/ui/#iso:std:iso:9241:-11:ed-2:v1:en. (visited on 15/01/2023) (cit. on p. 7).

[28] *JFLAP*. URL: https://www.jflap.org. (visited on 11/11/2022) (cit. on p. 25).

[29] B. Kitchenham and S. Charters. *Guidelines for performing Systematic Literature Reviews in Software Engineering*. 2007 (cit. on pp. xv, 17, 19).

[30] J. M. Lourenço. *The NOVAthesis LaTeX Template User's Manual*. NOVA University Lisbon. 2021. URL: https://github.com/joaomlourenco/novathesis/raw/main/template.pdf (cit. on p. iii).

[31] M. Lourenço. "Turing machines in OCAML-FLAT/OFLAT". In: (2023) (cit. on p. 16).

[32] R. P. Macedo. "OCaml-Flat on the Ocsigen framework". MA thesis. FCT-UNL, 2020 (cit. on pp. 5, 16).

[33] *Mann Whitney U Test - Boston University School of Public Health*. URL: https://sphweb.bumc.bu.edu/otlt/MPH-Modules/BS/BS704_Nonparametric/BS704_Nonparametric4.html. (visited on 08/01/2024) (cit. on p. 76).

[34] *NASA TLX*. URL: https://humansystems.arc.nasa.gov/groups/TLX/. (visited on 03/03/2023) (cit. on p. 11).

[35] J. Nielsen and T. K. Landauer. "A Mathematical Model of the Finding of Usability Problems". In: *Proceedings of the INTERACT '93 and CHI '93 Conference on Human Factors in Computing Systems* (1993) (cit. on pp. 7, 19).

[36] D. Pereira. *Usability Evaluation of OFLAT Platform*. Zenodo, Jan. 2024. DOI: 10.5281/zenodo.10471751. URL: https://doi.org/10.5281/zenodo.10471751 (cit. on pp. 59, 89).

[37] K. Petersen et al. *Systematic Mapping Studies in Software Engineering*. 2008. URL: www.splc.net (cit. on p. 17).

[38] N. Pillay. "Learning Difficulties Experienced by Students in a Course on Formal Languages and Automata Theory". In: *SIGCSE Bull* 41.4 (Dec. 2009), pp. 48–52. ISSN: 0097-8418. DOI: 10.1145/1709424.1709444. URL: https://doi.org/10.1145/1709424.1709444 (cit. on p. 2).

[39] J. Pinto. "Reengineering of the OFLAT platform". In: (2023) (cit. on p. 90).

[40] *ProM*. URL: https://promtools.org. (visited on 08/02/2023) (cit. on p. 50).

[41] B. D. Ramos. "Usability Evaluation of OFLAT Platform". 2022 (cit. on pp. 2, 3, 24, 28, 30–45, 67, 68).

[42] S. H. Rodger et al. "Increasing Engagement in Automata Theory With JFLAP". In: *Proceedings of the 40th ACM Technical Symposium on Computer Science Education. SIGCSE '09. Chattanooga, TN, USA: Association for Computing Machinery* (2009), pp. 403–407. DOI: 10.1145/1508865.1509011. URL: https://doi.org/10.1145/1508865.1509011 (cit. on p. 26).

[43] S. P. Rogers. *Interaction Design*. 4th ed. Wiley, 2002 (cit. on pp. 6, 9).

[44] J. Rosenberg. "Statistical Methods and Measurement". In: *Guide to Advanced Empirical Software Engineering* (2008). Ed. by F. Shull, J. Singer, and D. I. Sjoberg, pp. 155–184. DOI: 10.1007/978-1-84800-044-5_6. URL: https://doi.org/10.1007/978-1-84800-044-5_6 (cit. on p. 67).

[45] P. Runeson and M. Höst. "Guidelines for conducting and reporting case study research in software engineering". In: *Empirical Software Engineering* 14 (2 Apr. 2009), pp. 131–164. ISSN: 13823256. DOI: 10.1007/s10664-008-9102-8. URL: https://lup.lub.lu.se/search/ws/files/2838283/1276782.pdf (cit. on p. 8).

[46] *Segment*. URL: https://segment.com. (visited on 28/01/2023) (cit. on p. 48).

[47] *Shapiro-Wilk*. URL: https://www.gigacalculator.com/calculators/normality-test-calculator.php. (visited on 25/07/2023) (cit. on p. 68).

[48] F. Shull, J. Singer, and D. I. K. Sjoberg. *Guide to Advanced Empirical Software Engineering*. Springer-Verlag London, 2008. ISBN: 978-1-84800-043-8 (cit. on pp. 9, 55).

[49] E. Souza, A. Moreira, and M. Goulão. "Deriving architectural models from requirements specifications: A systematic mapping study". In: *Information and Software Technology* 109 (May 2019), pp. 26–39. ISSN: 09505849. DOI: 10.1016/j.infsof.2019.01.004 (cit. on p. 18).

[50] *Statistics Kingdom*. URL: https://www.statskingdom.com/shapiro-wilk-test-calculator.html. (visited on 25/07/2023) (cit. on p. 68).

[51] *SUS score*. URL: https://uiuxtrend.com/measuring-system-usability-scale-sus/#calculation. (visited on 16/01/2023) (cit. on p. 38).

[52] K. S. Taber. "Experimental research into teaching innovations: responding to methodological and ethical challenges". In: *Studies in Science Education* 55 (1 Jan. 2019), pp. 69–119. ISSN: 19408412. DOI: 10.1080/03057267.2019.1658058 (cit. on p. 9).

[53] R. M. Verma. "A Visual and Interactive Automata Theory Course Emphasizing Breadth of Automata". In: *Proceedings of the 10th Annual SIGCSE Conference on Innovation and Technology in Computer Science Education. ITiCSE '05*. Caparica, Portugal: Association for Computing Machinery, 2005, pp. 325–329. ISBN: 1595930248. DOI: 10.1145/1067445.1067535. URL: https://doi.org/10.1145/1067445.1067535 (cit. on p. 1).

[54] *Within subjects*. URL: https://www.nngroup.com/articles/between-within-subjects/. (visited on 26/07/2023) (cit. on p. 62).

[55] C. Wohlin et al. *Experimentation in Software Engineering*. Springer Berlin Heidelberg, 2012. ISBN: 9783642290435. URL: https://books.google.pt/books?id=Pg7PugAACAAJ (cit. on p. 86).

[56] C. Wohlin. "Guidelines for snowballing in systematic literature studies and a replication in software engineering". In: Association for Computing Machinery, 2014. ISBN: 9781450324762. DOI: 10.1145/2601248.2601268 (cit. on p. 19).

[57] P. S. Yalagi and R. K. Dixit. "Enhancing the learning ability using JFLAP for Theory of Computation Course". In: *Journal of Engineering Education Transformations, Special Issue* (2017), pp. 2394–1707 (cit. on p. 26).

A | Primary studies selected

S01	P. Chakraborty, P. C. Saxena, and C. P. Katti. "Fifty Years of Automata Simulation: A Review". 2011, p. 59. (ACM)
S02	E. Gramond and S. H. Rodger. Using JFLAP to Interact with Theorems in Automata Theory. (ACM)
S03	Cesar A. Tecson and Ma. Mercedes T. Rodrigo. "Tutoring Environment for Automata and the Users' Achievement Goal Orientations". In: 2018 IEEE International Conference on Teaching, Assessment, and Learning for Engineering (TALE). IEEE, 2018. isbn: 9781538665220. (IEEE)
S04	F. Erlacher. Pushdown Automata Simulator. 2009.
S05	M. Hamada. LNCS 5102 - Supporting Materials for Active e-Learning in Computational Models. 2008, pp. 678–686.
S06	M. Hamada. "Turing machine and automata simulators". In: Procedia Computer Science vol. 18. Elsevier B.V., 2013, pp. 1466–1474. doi: 10.1016/j.procs.2013.05.314.
S07	J. Mcdonald. Interactive Pushdown Automata Animation. 2002.
S08	M. Procopiuc, O. Procopiuc, and S. H. Rodger. Visualization and Interaction in the Computer Science Formal Languages Course with JFLAP.
S09	L. M. F. Vieira, M. M. A. Vieira, and N. J. Vieira. Language Emulator, a Helpful Toolkit in the Learning Process of Computer Theory. 2004.
S10	T. M. White and T. P. Way. jFAST: A Java Finite Automata Simulator. 2006
S11	M. Hamada. Web-based Active e-Learning Tools for Automata Theory. In: Seventh IEEE International Conference on Advanced Learning Technologies (ICALT 2007). 2007
S12	Castro-Schez, Jose J. et al. Designing and using software tools for educational purposes: FLAT, a case study. In: IEEE Transactions on Education. IEEE 2009 pp. 66-74
S13	P. Chakraborty, P. C. Saxena, and C. P. Katti. "A compiler-based toolkit to teach and learn finite automata". In: Computer Applications in Engineering Education 21 (3 Sept. 2013), pp. 467–474. issn: 10613773. doi: 10.1002/cae.20492.

Table A.1: List of primary studies selected

B Mapping of the primary studies

ID	Q1		Q2		Q3		Q4	
	Web/mobile	Desktop	Similar	Dissimilar	Usability	Unreported	Empirical	Inspection
S01	X	X	X	X	X	X	X	
S02		X	X			X		
S03	X		X		X		X	
S04		X	X			X		
S05		X	X		X		X	
S06	X	X	X		X		X	
S07		X	X			X		
S08		X	X			X		
S09		X	X		X		X	
S10		X		X	X		X	
S11		X	X		X		X	
S12		X	X		X		X	
S13		X		X	X		X	

Table B.1: Mapping of the primary studies

C Process Mining - Different Combinations

The following Process Mining results use different combinations for the tables “button_clicked”, “pages” and “tracks”.

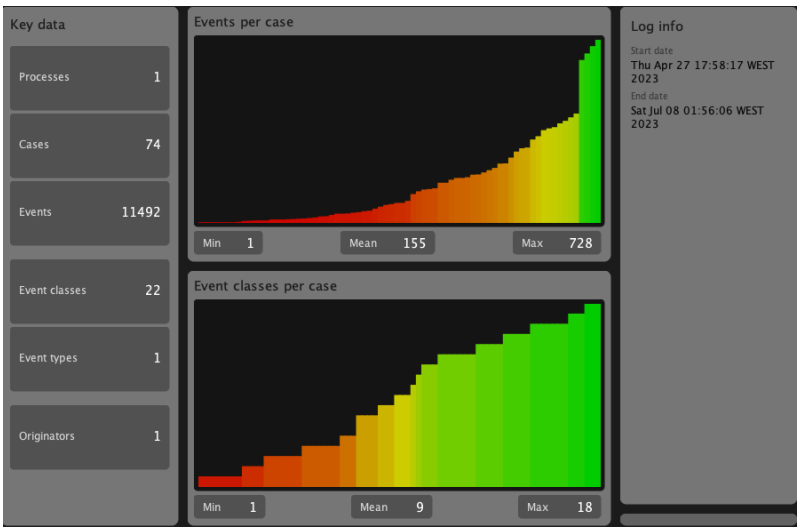


Figure C.1: Button Clicked bar plot with different combination

In the Figure C.1, it was selected the “anonymousId” and “context_ip” as the case columns and for the event column, it was selected “event” and “name”. Again, “event” is button_clicked, and “name” is the type of button that was clicked.

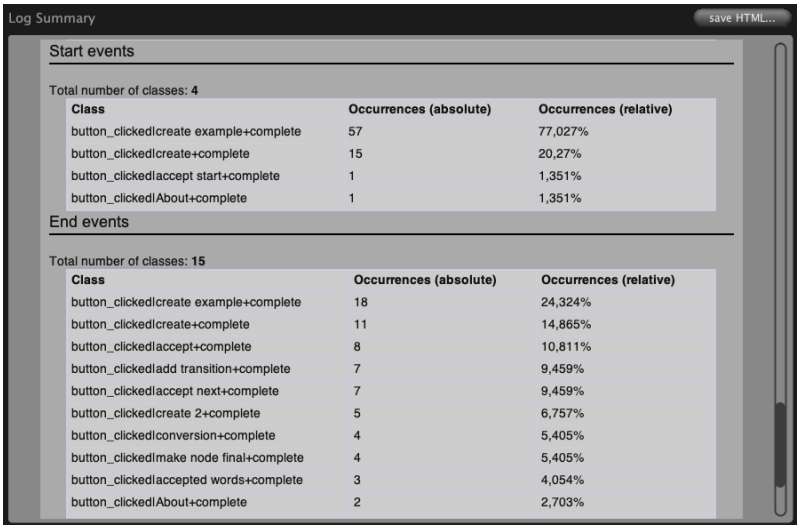


Figure C.2: Start events and End events of button clicked table

It is possible to infer from the Figure C.1 and the Log Summary of the Figure C.2 that students began the exercises in OFLAT when they clicked the "create" button, the "create example" button, or the "accept start" button, and they concluded those exercises when the final action was taken, such as "accept", "add transition", "make node final", "accepted words", etc.

These conclusions are the same as obtained in the subsection 6.3.5.

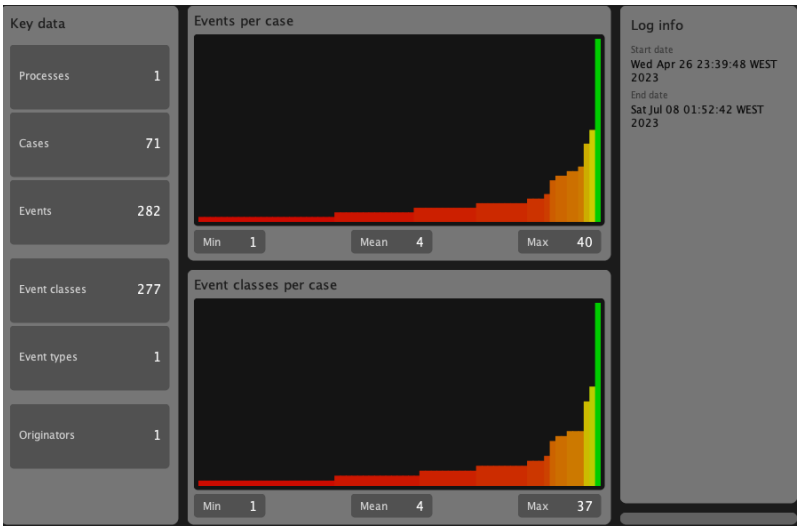


Figure C.3: Pages bar plot with different combination

In the Figure C.3, it was selected the "anonymousId" as the case column and for the event column, it was selected "context_ip" and "sent_at".

By analyzing the Figure C.3, it can be concluded that most students who solved the exercises in OFLAT used the suffix *"?logging"* in the following URL: `http://ctp.di.fct.unl.pt/FACTOR/OFLAT/?logging`.

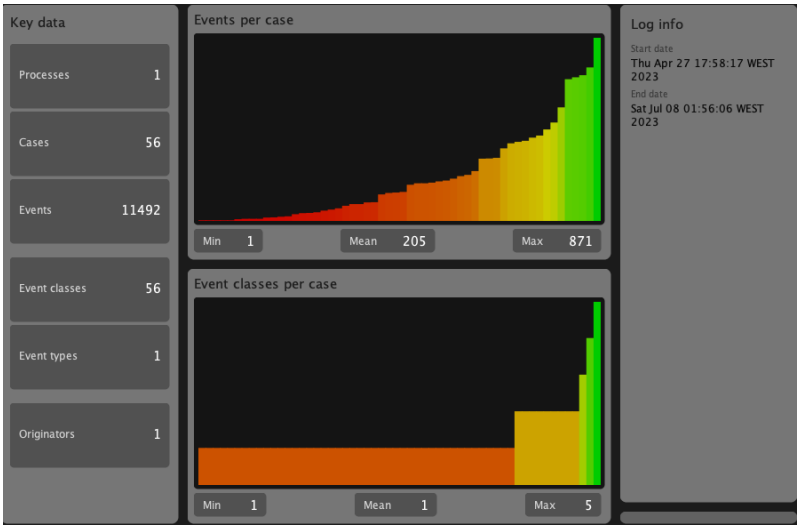


Figure C.4: Tracks bar plot with different combination

In the Figure C.4, it was selected the “anonymousId” as the case column and for the event column, it was selected “context_ip” and “event_text”.

The events contained in the table tracks and the events stored in the table button_clicked are identical, as can be inferred from the analysis of the previous figure.