# Relatório da Entrega 4 de Estudos Avançados em Sistemas de Software (08/05/2025)

Nome: David Tadokoro
NUSP: 10300507
Orientador: Paulo Meirelles

## 1) Contexto

No primeiro semestre de 2024, houve o oferecimento da disciplina de *Desenvolvimento em Software Livre* (MAC0470/5856), que envolveu mentorar os alunos no ecossistema do Linux, que inclui o próprio projeto do kernel Linux, ferramentas que suportam o desenvolvimento, distribuições GNU/Linux, entre outras. O programa da disciplina foi dividido em três fases, onde os alunos imergiram de forma prática em cada uma das camadas do ecossistema.

O projeto proposto para a disciplina envolve a escrita de um artigo científico sobre como treinar novos contribuidores de uma forma eficiente e prática com intuito de prepará-los com as habilidades necessárias para se tornarem reais desenvolvedores do kernel, baseado nas experiências da disciplina. Vale notar que o objetivo deste treinamento é de munir novos contribuidores (com pouca ou nenhuma experiência prévia em software livre) com a base essencial de habilidades e conhecimentos para irem além de serem os chamados *one-time contributors*.

Nesta quarta entrega, o objetivo era fazer primeiras versões da seção de materiais e métodos e da seção de resultados. No entanto, conversando com meu orientador (Paulo), decidimos abandonar a ideia de agregar os dados do oferecimento da disciplina neste ano, ao menos para o artigo que será submetido ao final do semestre (meu projeto para a disciplina). Percebemos que os dados do primeiro oferecimento de 2024 já são satisfatórios para trazermos a discussão desejada, mas também que não temos tempo hábil para coletarmos os dados deste ano, além do plano inicial de coletá-los no meio do semestre (ao invés do final, como foi feito ano passado) ser improdutivo.

Uma consequência disso é que tivemos que reajustar o cronograma das entregas, removendo os entregáveis relacionadas à coleta, análise e incorporação dos dados deste ano. Portanto, para esta quarta entrega, decidimos nos ater apenas à seção de materiais e métodos e mover a seção de resultados para a próxima entrega.

Tanto o arquivo fonte em Markdown (`entrega-4/relatorio.md`) deste relatório, quanto o arquivo em Latex puro da seção de materiais e métodos (`entrega-4/method.tex`) se encontram no **meu repositório de artefatos para a disciplina**.

Disponibilizamos o **documento Latex no overleaf para consulta** e ele deve estar visível sem necessidade de liberação de acesso. Ele está um pouco mais

organizado, mas ainda como um "dump" dos artefatos para a escrita do artigo. De toda forma, a seção está a seguir na íntegra para consulta fácil.

## 2) Seção de Materiais e Métodos

# Method

The main goal of this work is to design efficient and reproducible guidelines to train several inexperienced contributors to the Linux kernel ecosystem, providing them with a solid base of hard and soft skills necessary to become long-lasting assets in Linux kernel development. The guidelines can be characterized by answering our two main research questions:

- **RQ1.** What teaching techniques are effective in introducing newcomers to the Linux ecosystem (in-loco workshops, tutorials, lectures from practitioners)?
- **RQ2.** What hard and soft skills are crucial to success in the Linux ecosystem as a contributor?

Our research work was divided into two parts:

1. Training a group of computer science students which was implemented in a four-month university course.
2. Collection and analysis of qualitative data representing the perspectives from students and mentors about the experience of the course.

### Training Contributors to the Linux ecosystem

The course was offered at a Computer Science higher education institution and was attended by undergraduate, postgraduate, and other students. Of all the students enrolled, 24 completed the course in full.

The course structure focused on practical and interactive activities rather than lectures or expository classes, even though the latter were seldom used. The course was divided into the following three phases. Phases one to three occurred chronologically, and went from the lowest level of the Linux ecosystem, passing through the supporting and downstream projects, and ending in an arbitrary FLOSS project.

For most steps of each phase, report-like activities tied to grades were required to keep track of the student's progress and ensure that the whole class was roughly on the same page.

#### Phase 1: Linux kernel contribution

In the first phase, students were exposed to the development of the kernel component of GNU/Linux operating systems. The Linux kernel is the lowest

layer of the Linux ecosystem.

Initially, students spent five weeks building a development setup and learning the fundamental processes and practices (workflows) of Linux kernel development using tutorials devised by a veteran Linux practitioner. Each week, there were two separate in-loco workshop sessions where all students had to do the tutorials with the support of the mentors. The mentors were present throughout all sessions to help any student individually, not just by troubleshooting the several problems encountered over the execution of the tutorials, but also by answering conceptual doubts and curiosities.

After setting up the necessary environment and knowledge, students spent three weeks planning and developing a patch destined for a Linux subsystem, culminating in sending the patch and participating in the review process. A comprehensive list of patch suggestions was produced by the same veteran Linux practitioner who authored the tutorials used in part one of the first phase in the context of the Industrial I/O (IIO) subsystem, but students were not obliged to follow the suggestions or contribute to IIO. During these three weeks, the two separate in-loco workshop sessions continued, where mentors helped the students from understanding a patch suggestion to replying to maintainer feedback. A guide on how to use git to create, send, and update a patch for email-based development models (the case of the Linux kernel) made by the mentors was introduced to the students.

For this phase, complex changes of deep impact were not demanded, as the focus was to present students with the overall dynamics and skills necessary to contribute to the Linux kernel, beyond technical knowledge of device drivers or low-level programming. Nevertheless, naive and straightforward contributions (like fixing typos or coding style violations) were discouraged as they are considered spam in most Linux communities and do not bring any challenge from the perspective of development and reviewing.

**Phase 2: GNU/Linux supporting ecosystem contribution**

In the second phase, students moved from the base layer of the Linux ecosystem to the tools and projects that support the ecosystem, i.e., projects that directly help Linux kernel developers in their daily tasks in the form of tools or similar, and projects that use the Linux kernel as a platform, which in this context we call *downstream* projects, as they feed of the *upstream* project (that is the Linux kernel).

Students started this phase collaborating with the *kworkflow* project, which is a set of tools that automate and streamline the daily tasks of Linux developers in a unified interface. Unlike the Linux kernel, kworkflow is a project hosted on GitHub, so mentors had to introduce students to how Web-based projects operate and the fundamental concepts to contribute to those. This part took two weeks, one of which had two workshop sessions (the other week, students were on a mid-semester break), where mentors helped students in the same fashion as

in the part of sending contributions to the Linux kernel, but adapting it to the context of the kworkflow project.

After collaborating with kworkflow, students had a workshop conducted by a veteran Debian developer about Debian packaging, using Perl projects as context. This part lasted only one week, with two workshop sessions composed of two hands-on tutorials that introduced students to the overall workflow of Debian packaging, whilst they attempted to package a Perl project. In this part, mentors did not have considerable knowledge or experience in software packaging, so most of the mentoring was done by the guest Debian developer.

Even though contributing to kworkflow and Debian package is arguably simpler and has less setup overhead, contributions in this phase were still not required to be of impact.

#### Phase 3: Arbitrary FLOSS project contribution

In the last phase, students chose to contribute to an arbitrary FLOSS project. Continuing to contribute to the Linux kernel, kworkflow, or Debian packaging was possible (and even encouraged), but students were not obliged to do so.

In the last 5 weeks, students clustered around a handful of FLOSS projects, with some repeating projects from the other phases. Workshop sessions continued. However, the form of those was more unrestrained as there was a significant variability in the development contexts. In this sense, mentors adopted a more passive consulting role to help and guide students with any issues relating to their contributions and projects. At this point, we envisioned that, after experiencing the considered harder layer of the Linux ecosystem, then a more understandable layer of the ecosystem, students were more than equipped to handle entering new FLOSS projects on their own (in case they did not continue in a project of the previous phases).

Unlike the contributions of the other two phases, students were expected to contribute more meaningfully to the respective projects.

## Collecting and analyzing perspectives of students and mentors

Qualitative data that encodes the perspectives of students and mentors was collected and analysed to extract the necessary insights for us to answer our research questions and determine where we got it right and wrong to devise solid guidelines.

To accomplish this, we used three different sources of informations:

1. Students blog posts
2. Students surveys
3. Mentors interviews

#### Students blog posts

Throughout the course, students had to present report-like deliverables in the form of blog posts documenting their experiences in each step of the phases. The only direction provided to students about how to produce the blog posts was to aim to do a *board log*. A non-obligatory post summarizing the whole course experience was asked to be written.

#### Students surveys

We conceived a survey with 47 questions to query for different perspectives of the students. This survey was released to students at the end of the course, and they were asked to answer voluntarily.

Questions covered a wide range of topics, from the class characterization (education level, professional background, familiarity with development tools, among others) to their feedback on the course activities, materials, and personnel (were the workshops helpful? was the mentor's assistance useful? which steps were improductive?) to their individual perspective on their skills improvement as result of the training (how confident are you in continuing contributing to the Linux kernel?). Most questions used the Likert scale (ratings from 1 to 5 in different forms), but some were boolean (yes or no), and even free form.

#### Mentors interviews

Through direct conversation, the mentor's perspectives were collected. In the interviews, we did not follow a script, just overall questions about how they perceived the course experience

#### Analyzing the different sources of data

We surveyed students using the Limesurvey platform and had a full report of the answers out of the box. We selected meaningful answers and clustered replies to questions that used the Likert scale as positive, neutral, and negative.

Regarding the mentor's interviews, as they represented a smaller volume of data than the students' blog posts and surveys, we did not filter or aggregate the data from this source, taking it at face value.

## 3) Conclusão

Após uma correção de curso, acreditamos que a direção do artigo finalmente está bem solidificada e, com uma base para a seção de materiais e métodos e com os resultados do oferecimento de 2024 que já foram analisados nas duas primeiras entregas, podemos partir para (1) começar a fechar o artigo com as "macro" seções remanescentes e (2) refiná-lo incluindo o embasamento teórico necessário e partes remanescentes (trabalhos relacionados, limitações, e tudo mais).