CHALMERS
UNIVERSITY OF TECHNOLOGY

UNIVERSITY OF GOTHENBURG

Design implications of an AI-assisted error categorization tool in an industrial setting

Master's thesis in Computer science and engineering

Albin Boklund
Vincent Stocks

Department of Computer Science and Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
UNIVERSITY OF GOTHENBURG
Gothenburg, Sweden 2026

Master's thesis 2026

Design implications of an AI-assisted error categorization tool in an industrial setting

Albin Boklund
Vincent Stocks

UNIVERSITY OF
GOTHENBURG

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Computer Science and Engineering
Chalmers University of Technology
University of Gothenburg
Gothenburg, Sweden 2026

Design implications of an AI-assisted error categorization tool in an industrial setting

Albin Boklund
Vincent Stocks

Supervisor: Aris Alissandrakis, Computer Science and Engineering
Advisor: Michael Sennevik, Ericsson
Examiner: Sara Ljungblad, Computer Science and Engineering

Master's Thesis 2026
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Typeset in LaTeX
Gothenburg, Sweden 2026

Design implications of an AI-assisted error categorization tool in an industrial setting

Albin Boklund
Vincent Stocks
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg

Abstract

Implementing AI-powered tools into existing workflows to increase efficiency and remain competitive is a priority among many companies and organizations, especially in the tech industry. However, there is a potential gap in the research regarding how to implement such AI systems in a way that feels natural to the users already familiar with the existing workflows. Research shows that users confronted with an AI tool that upsets their workflow are less likely to trust the tool and provide feedback to it. This loss of feedback and trust in the AI tool can lead to decreased performance over time, potentially model collapse. In this thesis we explore potential solutions to this problem from an Interaction Design and UI/UX perspective, addressing a real-world problem in an industrial setting.

The study was conducted in an industrial software development context at Ericsson, focusing on the CATegorizer, an AI tool that predicts problem types for failed software tests in the CI pipeline. A mixed-methods approach was utilized, with a substantial pre-study gathering both quantitative and qualitative data, followed by ideation, implementation and evaluation of the proposed solutions. The two implemented and evaluated solutions were a dashboard intended to increase understanding of the CATegorizer and its feedback loop, as well as an interaction mechanism for providing feedback more easily within the existing workflow.

The results suggest that existing guidelines for Human-AI interaction provide a solid foundation for applications in an industrial setting, but need to adequately take into account the affected user group and the surrounding organizational and workflow context. For successful implementation of AI tools in professional workflows, it is important that the users are adequately informed of the feedback loop and the value of their feedback, while minimizing additional efforts required to provide it.

Keywords: Interaction Design, Human-AI Interaction, UI/UX, Industrial Software Development, Continuous Integration, Error Categorization, Requirements and Guidelines, AI Feedback Loop

Acknowledgements

We would like to begin by thanking our Chalmers supervisor Dr. Aris Alissandrakis for his feedback, support and always being quick to answer our questions throughout the project. Next we would like to thank “The Michaels”, Michael Sennevik and Michael West, our Ericsson supervisors, for their guidance and support navigating the corporate landscape of the company.

We would also like to thank Anastasiia Shevchenko for her invaluable help during the implementation work and the rest of Team Suricats for letting us implement our solutions into their products. Finally we would like to express our gratitude to the Flow Guardians for their insights, the COIN organization for participating in our user studies and putting up with our mass-emails, and to Ericsson as a whole for the opportunity to write this thesis with the company.

Albin Boklund & Vincent Stocks, Gothenburg, 2026-06-16

Contents

List of Figures

List of Tables

1

Introduction

1.1 Thesis Topic

The development teams working on radio base station products at Ericsson - one of the world's largest networking and telecommunications companies - are responsible for writing and maintaining the code that operates the radios. As part of the daily workflow, automatic tests are run within a continuous integration (CI) pipeline. When a test fails, someone has to determine the cause of it. The failure can be due to a fault in the product code, a flaw in the test itself, or an issue with the test environment, such as a broken cable. This troubleshooting process can be time-consuming, as it often requires examining extensive log files to identify the root cause.

To support the troubleshooter in this task, the CI pipeline includes an AI-based error categorization tool called the CATegorizer, which processes failed test logs and predicts which of the three categories (Product, Test, or Environment) the error belongs to. This prediction helps developers narrow down their investigation and saves time otherwise spent on manual troubleshooting. After investigating the error and determining its true category, the troubleshooter is expected to provide feedback by confirming or correcting the AI's prediction. However, this feedback step is often forgotten or skipped, which can lead to the AI tool becoming less and less reliable over time, and potentially collapsing entirely.

As AI tools become more and more prevalent in all aspects of life, especially in software-related professions, the changes in workflow they introduce can have both positive and negative impacts on users. It can be difficult to foresee all such impacts and their importance, but it is certain that AI tools often require changes in workflow for the people using them. There is therefore a need among software companies, including Ericsson, for research into this area, so that they can position themselves to maximize the beneficial impacts of AI tools and limit the negative ones. This thesis topic was requested by and carried out at Ericsson.

There are several possible approaches to addressing the feedback problem described above. One alternative would be to increase the technical complexity of the AI model in order to improve its predictive robustness. Another approach could involve redesigning the system architecture to automate the categorization of frequently occurring faults. However, such technically oriented solutions do not directly address

the underlying behavioral challenge of inconsistent user feedback. This thesis therefore investigates how principles from interaction design can be applied to influence user behavior and encourage consistent feedback, thereby contributing to improved AI model reliability and longevity.

1.2 Research Question

How can the user interface of an AI-assisted error categorization tool be designed in order to encourage users to provide feedback to the AI model, without negatively impacting the users' perceived efficiency?

1.3 Aim

This thesis aims to explore how the workflow of expert users in a complex software system can be influenced through a redesign of the user interface, without impacting their perceived efficiency. The introduction of assistive AI tools in complex software systems requires changes in workflow to ensure that the models continue functioning correctly, but such changes can be met with resistance from users if introduced carelessly, especially from expert users. User cooperation and feedback is essential for the longevity of the AI models. As more and more companies deploy AI tools to aid their users, it is imperative that guidelines and frameworks be created for them to follow. Our thesis will further expand research into this area, building on established concepts and frameworks from the field of interaction design and cognitive ergonomics.

1.4 Goals

The goals of the thesis work are to evaluate the current user interface and workflow and identify barriers to successful use, redesign these aspects and create functional, high-fidelity prototypes for realistic testing, and finally validate the redesign using established methods. Two final deliverables are planned: a report presenting our work and findings, as well as the final high-fidelity, research-supported prototype that Ericsson can choose to implement and deploy.

2

Background

2.1 Interaction Design

This thesis is anchored in the field of interaction design and focuses on how to make the target group interact with the AI categorization tool to make it better. From a digital perspective, interaction design can be understood as shaping our everyday life through digital artifacts [1, p.11]. In this sense, interaction design goes beyond technical functionality and focuses on designing computer-based systems so that they become a graceful and meaningful part of everyday culture with both emotional and functional qualities. Interaction design focuses on the users of the system and emphasizes that designers should focus on users goals rather than solely on the technology that it is based on [1, p.12]. Good interaction design is centered around four main goals: clear mental models, reassuring feedback, the ability for users to navigate the software and consistency [1, p.15]. In the context of this thesis, interaction design is thus not only about improving usability of the AI categorization tool, but about designing the quality of interaction between user and system so that the tool becomes intuitive, meaningful and supportive of users' everyday work practices.

2.2 Terminology

To be able to understand the pre-existing workflow (as explained later in Section 2.3) it is important to understand some Ericsson-specific terminology and context. This section introduces Ericsson-specific terms, roles, and interfaces that are necessary to understand the studied CI workflow. The COIN organization where this thesis was conducted is also disclosed. The section first introduces the product and organizational context, followed by the main systems and interfaces referenced throughout the thesis. For additional technical description about topics covered in this thesis see Appendix A

2.2.1 Radio Base Station

The "Radio Base Station" is the collective name given to the product line of Base Stations developed and sold by Ericsson. These Base Stations allow end-users to access the RAN (Radio Access Network), which began with the introduction of 3G

and introduced the possibility of using mobile communications to transmit larger amounts of data than previously possible during the GSM era and its 1G and 2G technology. Before the introduction of 3G, the Radio Base Stations were called Base Tranceiver Stations, or BTS. According to the International Telecommunication Union, a Base Station is a "land station in the land mobile service".

2.2.2 The COIN Organization

COIN, Configuration and Infrastructure (previously known as CAT), is an organization within Ericsson with the purpose of supporting feature developers and managing product governance that enables the best possible product being delivered to Ericsson customers. The expectation is to support and adopt the introduction of System Functions, support the XFT transformation, explore support RAs with the developer in focus and guarding the CI flow. This thesis was conducted within the COIN organization but the findings can be applied to other organizations as well.

2.2.3 Flow Guardian

The Flow Guardian is a specialized developer role within the CI workflow and COIN. Their responsibilities are focused on ensuring that the CI flow functions as intended, without unnecessary stoppages or recurring errors. Flow Guardians work both reactively and proactively: reactively by tracing/troubleshooting errors and outages related to failed software validation/testing (and alerting the responsible parties) and proactively by developing, refining and improving the CI pipeline to prevent errors or mitigating their impact. There are three Flow Guardians based out of the Lindholmen office in Gothenburg, with a few more across other sites.

2.2.4 XFT Developer

Developers in cross-functional teams (XFTs) work on developing the Ericsson software products. Their work is tested, validated and deployed via the CI pipeline. Each XFT team is constructed to be able to contain and handle all roles and diciplines to be able to take a feature from idea to implementation. This includes knowledge within the system, developers, testers and documentation.

2.2.5 Follow Your Commit

Follow Your Commit (FYC) is the first interface in the CI pipeline. It is the entry point for developers and managers to watch over commits passing through each of the levels of testing. The system provides quick access to any testing results at any level, both for passing and failing tests. For UI example see Appendix C.

2.2.6 CFI Tickets

CFI tickets are written to detect errors in code that is being integrated. Each ticket is formulated as a test if the code fails to pass, the ticket is flagged as an issue. Tickets are typically written by engineers troubleshooting the CI flow, though they

can also be generated automatically by bots. Since tickets are usually created with a specific problem in mind, they can be labeled by type: Product, Test, Environment, or Undefined. These labels help troubleshooters quickly identify the root cause of a failure and resolve it more efficiently. For UI example see Appendix C.

2.2.7 Jira

Jira is a issue-tracking and project management platform developed by Atlassian used for planning, tracking and releasing software. Within the Ericsson development environment, Jira is used as the primary tool for creating agile workflows and ticket creation for issues. All tickets are stored in Jiras database and can be queried using JSQL (Jira Query Language) from its own interface or through an API call.

2.2.8 CI Flow Impediment

The data that this thesis draws conclusions from is from the project called *CI Flow Impediment* in Jira. The CI Flow Impediment (hereinafter CFI) project includes data from the whole CI/CD pipeline which allows users to collect and analyze the software test data. The CATegorizer is part of the BDC stage and therefore handles data from the CI Flow Impediment data collection. For UI example see Appendix C.

2.2.9 CFI Web

CFI Web is the primary tool for interacting with and managing CFI tickets. It allows users to create, monitor, and edit tickets, while giving troubleshooters access to key insights including a history of previous occurrences to help identify when a fault first appeared. Editing tickets in CFI Web is equivalent to editing them directly in Jira, but with a cleaner, more user-friendly interface. For UI example see Appendix C.

2.2.10 NoSA BDC Reports

The NoSA (Node System Analytics) BDC Reports interface is where the results of all BDC-level testing can be viewed. For example, if a commit fails in the BDC, the user can access the specific testing run and the related logs through this interface. It also clearly displays any CFI tickets that were tagged on that run. For UI example see Appendix C.

2.2.11 RAN CI Radiator

The RAN CI Radiator is used to monitor builds as they progress through the later stages of testing, specifically tracking builds advancing to the next deployment stage and software upgrade packages. The interface gives users a clear overview of which revisions have successfully passed through the flow and which require attention. For UI example see Appendix C.

2.3 The Pre-existing Workflow

Ericsson develops the software for their radio base stations (RBS) through many different teams within the company. Each XFT-team works on writing and updating software within and between hardware components of the RBS. To ensure code quality and system stability, all changes are handled through a continuous integration (CI) process, in which new code is continuously integrated, automatically built, and tested against existing software and hardware configurations. This code must be rigorously tested through multiple steps during the CI process to ensure that it is compatible with the existing code and hardware parts. There are also multiple different systems that the developers can use to track their code tests, with different views and information in each, resulting in a messy software ecosystem. Currently they have an artificial intelligence (AI) model that will automatically categorize the errors into *Product*, *Test* or *Environment*. This allows the Flow Guardians to find and fix the issues more efficiently, reducing both repair time and costs. Although this task is mainly for the Flow Guardians to fix, it should also be done be the XFT-developers.

Without the categorization tool, the user would only get an error response from one of the CI steps without much further explanation. This would lead to the developer having to explore the problem further to understand which part of the RBS caused the CI loop to throw the error. The categorization tool was therefore built to help developers understand the cause of the issue. The tool sets a likelihood for each category from which the issue can arise, where the categories are *Product*, *Test* and *Environment* as previously mentioned. With the system Ericsson uses today, the main issue lies in the current user interface where developers can view the various integration attempts made by different teams. Some of these attempts result in errors. In the current setup, errors are indicated by having a red line at the top of the indicated test in the sequence of tests as shown in Figure 2.1.

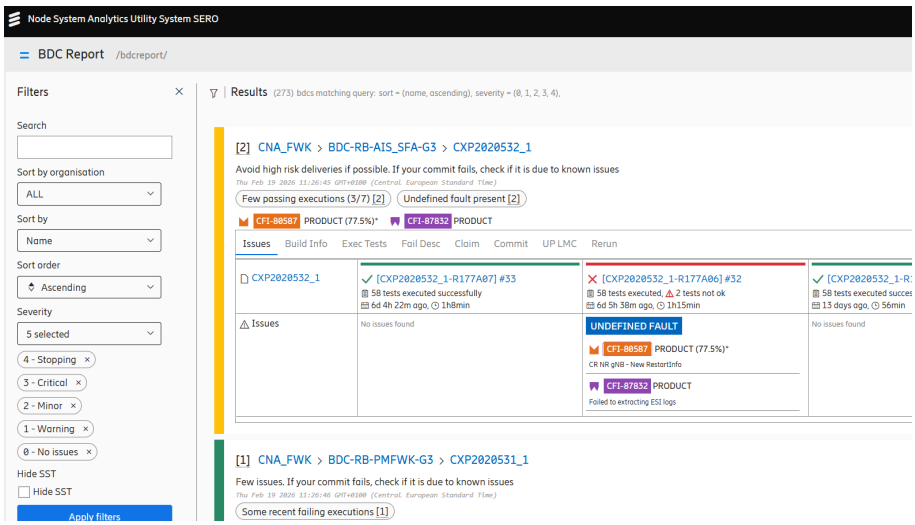


Figure 2.1: Failed test indicated by red line at the top in Node System Analysis (NoSA)

The filters that the pushed code did not pass are shown below the red line with the predicted category displayed next to it, as well as a percentage of certainty from the AI-model if the category has not been manually confirmed by a user. The intended workflow is that Flow Guardians or XFT-developers investigate and fix the error, and then report back to the categorization tool whether the assigned problem classification was correct or not. This is done through either Jira (see Section 2.2.7) as shown in Figure 2.2 or CFI Web (see Section 2.2.9) as shown in Figure 2.3.

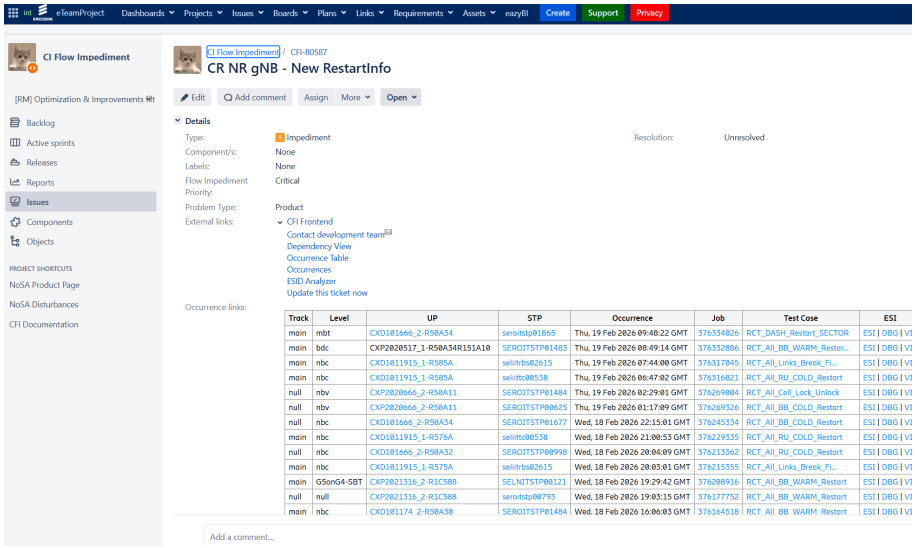


Figure 2.2: CI Flow Impediment (CFI) interface in Jira. CFI ticket data is stored in Jira, and this is one of the interfaces used for interacting with CFI tickets. The image show the CFI ticket 80587 and can be edited using the edit button below the title.

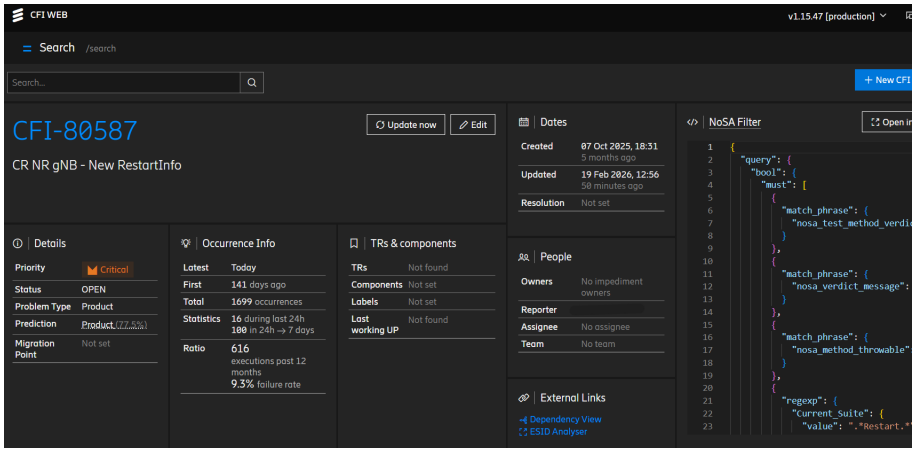


Figure 2.3: CFI WEB displaying the ticket 80587. Details and latest occurence are shown as well as the filter that catch any error. The ticket can be edited usign the edit button in the top middle.

Both interfaces require multiple clicks from the user to set the problem type; first the user must go to their CFI ticket and press *Edit*, then select the problem type on a

card that pops up, and then select save. However, this step is not always completed as the action is not mandatory or clearly indicated. In practice, Flow Guardians often fix the issue and then move on without providing feedback to the system. This missing feedback is problematic because the AI-based categorization tool relies on user input to ensure it is trained on correct and up-to-date data. Without consistent validation from users, incorrect classifications may go uncorrected, preventing the model from learning and causing it to repeat the same mistakes. Over time, this leads to a gradual decline in classification accuracy, reducing the overall effectiveness and reliability of the categorization system.

Employees at Ericsson have raised the request to make the model self-label tickets that are "easy and common". However, training an AI model on generated data can introduce a risk of inaccuracies from the data that are amplified with each training iteration. When a model is repeatedly trained on its own predictions rather than on human-verified data, small errors can accumulate and gradually skew the underlying data distribution. This phenomenon, sometimes referred to as model collapse, results in a dataset that increasingly diverges from the original real-world data it was intended to represent [2, p.755].

In the context of Ericsson's categorization tool, insufficient user feedback creates a similar risk. If users do not validate or correct the assigned error categories, the model may treat its own previous predictions as implicitly correct. Over time, this effectively means that the system is learning from partially unverified or self-generated labels rather than from confirmed human input. When the availability of new, high-quality labeled data is limited, this feedback gap becomes the primary source of degradation in model performance.

Shumarilov et al. demonstrate that self-training systems may struggle to accurately reproduce original data distributions, often introducing small variations that increase the error over successive training cycles (as demonstrated in Figure 2.4)[2, p.755]. If left unchecked, this process can significantly reduce classification accuracy and reliability. Consequently, consistent human validation is not merely a usability improvement but a critical safeguard against long-term performance decline in AI-driven systems.

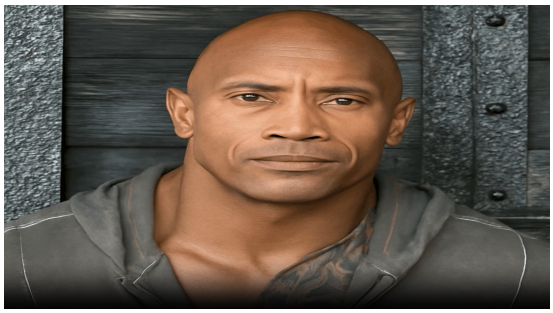


Image after one iteration.

Output image after approximately 30 iterations.

Output image after approximately 60 iterations.

Output image after approximately 100 iterations.

Figure 2.4: ChatGPT output after prompted with "Create the exact replica of this image, don't change a thing" 101 times, using as initial input an image of actor Dwayne Johnson. (Images taken from a Reddit post [3]). This exemplifies how entirely machine-based iterations become difficult for humans to interpret

3

Theory

The following chapter covers work that this thesis discuss as well as frameworks that forms the base of this thesis. For more technical theory surrounding AI, workflows and categorization see Appendix A

3.1 Related Work

Amershi et al. [4] describe Human-AI interaction as encompassing the ways users engage with AI-infused systems. This refers to systems where AI capabilities such as prediction, recognition, and recommendation are embedded into products and services. Their work focuses on how such interactions should be designed to be effective, understandable, and trustworthy. In this paper the authors present a synthesis of existing research in Human-AI interaction, based on over 150 different guidelines gathered from prior work [4]. These guidelines are compiled and interpreted into a set of 18 new guidelines that designers and developers can follow when implementing AI functionality in their systems. In Phase III, the 18 guidelines were tested in a user study consisting of a modified heuristic evaluation, with 49 participants (all from the HCI field) analyzing AI systems they were familiar with and comparing it to the guidelines. Each participant was assigned an AI-driven feature from one of 20 selected products spanning 10 categories. They graded the instances they found on a five-point scale, from "clearly violated" to "clearly applied". Applications and violations were marked along with a short explanation (and sometimes a screenshot). The participants also rated the clarity for each guideline on a five-point scale. This all resulted in participants collectively reporting 785 examples of applications/violations across the 20 products (313 applications, 277 violations, 89 neutral, and 106 "does not apply"). At least one application or violation was found for each guideline in every product, suggesting broad relevance. With the participants feedback on the clarity of the guidelines the researches redefined some of the guidelines. After this they conducted another study with 11 experts, where the experts compared the original vs. revised guidelines and assessed their clarity in terms of wording, phrasing etc. Most guidelines were deemed good, some were revised again. These guidelines provide an evidence-based starting point for designing interaction patterns that both solicit meaningful user feedback and preserve perceived efficiency. This directly aligns with our focus on feedback on AI recommendations in an error-categorization workflow.

Another interesting publication is the journal article *Interacting Meaningfully with Machine Learning Systems: Three Experiments* in which the authors investigate how end users interact with machine learning systems in other ways than just providing simple two choice feedback [5]. The authors argue that more complex forms of interaction may improve both user understanding and model performance, but that little was previously known about what types of feedback users are willing or able to provide. The study consists of three experiments, the first being a think aloud study that examines how users reason about a machine learning systems behavior and what kinds of feedback they naturally try to give when interacting with it. The results show that users often want to provide more complex feedback than simple yes or no answers, but that such feedback must be well supported by the interface. The second and third experiments explore how different forms of user feedback can be incorporated into the learning process and how this affects model accuracy and user experience. Across the experiments, the authors find that direct interactions and feedback integrated into the assessed task are both feasible and beneficial.

This work is highly relevant for this thesis, as it highlights that users are more likely to provide feedback when it is embedded naturally in their workflow rather than treated as a separate or optional task. In the context of our error categorization tool, this supports the idea that developers are unlikely to provide corrective feedback unless the interaction is seamlessly integrated with the error-resolution process itself.

In the paper *Soliciting Human-in-the-Loop User Feedback: Effects on Trust and Perceived Model Accuracy* the authors investigate the effects of requesting user feedback in human-in-the-loop machine learning systems [6]. The authors conducted a controlled user study using a simulated object detection system where participants were given opportunities to provide corrective feedback to the model. The study measured participants' trust in the system and their perception of its accuracy, both before and after interacting with the feedback mechanisms. The results show that despite the potential for feedback to improve system performance, the act of requesting feedback led to a decrease in both perceived accuracy and user trust. This effect was observed even when the systems actual performance improved as a result of the feedback. The paper demonstrates that poorly designed or intrusive feedback mechanisms may have unintended negative consequences, emphasizing the importance of carefully balancing the need for user input with the users primary goals. This is an important consideration for this thesis as the proposed UI changes that this thesis produces may influence how users perceive and interact with the AI tool. Therefore, the design changes should be implemented in a way that does not negatively impact users' trust and perceived efficiency, consistent with the aims of the research question (Section 1.2).

The CATegorizer is used within an existing professional workflow, where many users possess substantial domain knowledge about CI systems, error resolution, and the surrounding tools. It is therefore important to consider the relevant research regarding what makes someone an expert and how interaction design can account for their needs. One such paper is *Embedding expert users in the interaction design process: a case study*, where the authors explore expertise and what to consider when designing for and with expert users [7]. This is relevant for this thesis because the potential

redesign is not intended for generic end users, but rather for developers and Flow Guardians that work within an established and complex CI workflow. Designing for such users requires attention to their existing ways of working and expectations of efficiency.

3.2 Frameworks

The two main frameworks that will shape the project are Human-Computer Interaction (HCI) and Human-AI Interaction (HAI). Understanding and working within these frameworks is important to ensure that the outcome of the project will contribute something of value to the field. HCI focuses on usability, efficiency and understanding expert users and the cognitive ergonomics within the human-machine system. All these aspects will be important to consider for the entirety of our project and will form much of the basis of our work [8]. As for HAI, this framework is incredibly important in understanding and designing the relationship between the human and the AI [4]. The concept of Human-in-the-loop learning for the AI model will also be central: we will work to ensure that the feedback loop between human and AI can run as smoothly and efficiently as possible. Furthermore, better explanations and justifications could make feedback easier, while ensuring that the user possesses the appropriate level of trust in the AI system. Another important aspect is that the user retains agency, while the AI should only be assistive.

4

Methodology

To answer our RQ, "How can the user interface of an AI-assisted error categorization tool be designed in order to encourage users to provide feedback to the AI model, without negatively impacting the users' perceived efficiency?", it was necessary to understand both the system and its users from multiple perspectives. This required insight into users' needs through semi-structured interviews, which enabled detailed exploration of user experiences, and surveys, which enabled data collection from a larger number of participants. Understanding users' cognitive processes and decision-making strategies required the use of ACTA, while users' perceived workload was assessed using NASA-TLX and their emotional responses were captured using SAM. This was all measured to map which parts of the preexisting workflows that were the hardest to use and understand. In addition, measurable data was needed to establish a baseline and evaluate the effectiveness of proposed design solutions, for which usage analysis was used to examine actual user behavior and system engagement. Therefore, a suitable plan was to adopt a mixed-methods approach, combining qualitative and quantitative data collection [9, p. 4].

The resulting qualitative data was analyzed using Thematic Analysis to identify recurring patterns in user needs, motivations, and experiences (see Section 4). In parallel, quantitative methods were used to assess perceived workload, emotional responses, and behavioral changes. Specifically, usage analysis was used to measure whether the proposed solution increased the amount of manually labeled data. Together, these methods provided complementary perspectives and formed a robust basis for addressing the research question.

4.1 Usage Analysis

Usage analysis is a quantitative method for examining user interaction data in order to identify patterns of behavior and system engagement [10]. By analyzing historical interaction data stored in system databases it makes metrics such as frequency of actions, feature usage, and engagement trends over time measurable. These metrics enable quantitative comparison between different interface designs to evaluate the effect of changes. Usage analysis is a reliable method as it relies on naturally occurring data rather then self-reported perceptions, which reduces the risk of response bias. It also allows for in-depth analysis of user behavior without interrupting the workflow of the users. However, while usage data can reveal what users do, it does

not directly explain why certain behavior occur. Therefore, it is often complemented by a qualitative method to provide further understanding of user behavior.

4.2 Semi-structured Interviews

Semi-structured interviews are a qualitative data collection method commonly used in user centered studies. The format combines a predefined interview guide with open-ended questions, allowing for coverage of specific topics while still enabling flexibility during the conversation [11], [12]. This balance between structure and flexibility distinguishes semi-structured interviews from fully structured interviews, which follow a strict script, and unstructured interviews, which rely almost entirely on spontaneous dialogue. One of the primary strengths of semi-structured interviews is their ability to create in-depth data. The interviewer is allowed to ask follow-up questions which can be used to clarify uncertainties and further dig into unexpected topics that emerge during the interview. This adaptability makes the method particularly suitable for exploring users experiences, perceptions and opinions, which can help researchers discover the unknown unknown.

Furthermore, with the interviewee being able to lead the discussion, the power relationship between the two parties is equalized. Although this is good to make the interviewee feel comfortable, it can also lead to the data generated being very personal and sensitive [12]. It is therefore important that the responses are confidential after the interview and that there is an agreement on how the data may be used.

4.3 Surveys

Surveys are a widely used research method for describing and exploring human behavior, particularly in social and psychological research [13]. One of the main advantages of surveys is their ability to be distributed quickly to a large number of respondents, for example through social networks or email. They are also typically quick and convenient for participants to complete, as they do not require face-to-face interaction and can be completed independently.

Because surveys are self-administered, they reduce the risk of data entry errors introduced by an interviewer, which can simplify the processes of coding and data cleaning. Additionally, respondents may experience less pressure when answering questions, as they can complete the survey at their own pace and at a time that suits them.

However, the absence of an interviewer can also be a disadvantage. For example, open-ended responses cannot be clarified or probed further, which may limit the depth or accuracy of the data. Furthermore, distributing surveys through online channels can introduce sampling bias, as respondents are often recruited through social networks where individuals may share similar interests or perspectives. This can lead to the over-representation of certain view points. The researchers therefore need to be conscious of the risk of biases or multiple answers from the same

respondents as people with strong opinions might try to sway the results of the survey.

4.4 Likert Scale

The Likert scale is a survey tool used to gauge opinions and behaviors among the subjects of a given study [14]. An implementation of the method usually comes in the form of a number of statements (questions) that the subjects are asked to rate their agreement with. Subjects are asked to rate their agreement on a 5 or 7-point scale, such as "Strongly Disagree" to "Strongly Agree", which enables researchers to quantify qualitative data.

4.5 NASA-TLX

The NASA Task Load Index (NASA-TLX) is a widely used method for quantifying and evaluating subjective user workload. The method was developed by NASA in the 1980s as a means to gauge perceived mental workload and stress experienced by their pilots. NASA-TLX measures workload across six subscales and accounts for their relative importance to the task at hand [15, p. 146]. By weighting each subscale according to the users perceived importance, the method captures individual differences in how workload is experienced, enabling more sensitive and comparable results across users. The questionnaire assesses workload along six dimensions: mental demand, referring to the mental and perceptual activity required (e.g., reasoning and decision-making); physical demand, representing the physical effort involved (e.g., pushing or pulling); temporal demand, reflecting time pressure due to task pace; performance, the users self-assessment of success and satisfaction in achieving task goals; effort, the mental and physical exertion required to attain the reported performance level; and frustration, which captures feelings of stress, discouragement, or insecurity. To determine the relative importance of these dimensions, 15 pairwise comparisons are conducted between all combinations of the six subscales. These comparisons are used to weight the subscale scores, and the final workload score is calculated as a weighted average of the six dimensions. This ensures that the overall workload score reflects the aspects of the task that users perceived as most demanding. NASA-TLX is a reliable and well-established method for quantifying task difficulty, which might otherwise be assessed only qualitatively [16, p. 907]. Using a quantitative measure provides a numerical benchmark of perceived effort, allowing us to compare workload across interface iterations and evaluate whether the final redesign negatively impacts users perceived efficiency. However, there are potential drawbacks with using NASA-TLX that should be acknowledged. It is the most commonly used method to evaluate mental workload within the field of HCI (with good outcomes), but some researchers point out that the reason for this might have more to do with convention rather than results[17, p. 20]. One of the issues is that it was originally developed to test pilots at NASA, which is quite a different subject and context than many current HCI-applications. Further, its structure is not designed to yield actionable re-design insights[18, p. 16]. Researchers invite

HCI practitioners to also consider other scales for evaluation as these might provide different insights. For this project however, NASA-TLX remains an appropriate method as it compares the perceived workload across design iterations, while more actionable redesign insights will be gathered through other qualitative methods.

4.6 Self-Assessment Manikin (SAM)

This method is a way to gather subjective data regarding the users' feelings in a given situation [19]. Understanding the users' emotional state during certain moments of use can influence how the system should be designed, identify problem areas, etc. SAM is easy to execute since it only needs to measure three different ratings: valence, activation and dominance. This makes it well suited to be part of other methods such as during an semi-structured interview since it does not require much time or effort from the respondent. Only responding in three dimensions does however reduce the amount of details produced from the method and it should therefore not be used by it self if the main goal of the study is to capture the users emotional state. Although SAM can be somewhat abstract compared to other methods, the insights gained can be impactful towards mapping of users cognitive and emotional state.

4.7 ACTA - Applied Cognitive Task Analysis

The Applied Cognitive Task Analysis (ACTA) techniques were developed by the US Navy as a way for designers to elicit critical cognitive elements from Subject Matter Experts [20, p. 1620]. The full method comprises three techniques: the task diagram interview, the knowledge audit and the simulated interview. The first has the subject divide their interaction into sub tasks, the second dives deeper into their domain-specific knowledge and the third simulates an event within the human-machine system.

ACTA provides a practical and structured approach for understanding cognitive processes such as decision-making and problem-solving strategies, which are often hard for experts to articulate explicitly. ACTA is also quicker and easier compared to traditional cognitive task analysis which makes it more accessible for less experienced users without making the output less useful for evaluating systems. The main drawback of ACTA is that it strongly relies on the domain knowledge of the respondents meaning that the quality of the output of ACTA depends both on the respondents ability to explain their thought process and the interviewer's ability question effectively.

4.8 Thematic Analysis

Thematic Analysis is a method for understanding themes and patterns within data and is especially useful when compiling the information gathered in a user study [21]. Key concepts, phrases and ideas received from the users can be grouped according

to similarity, from which overarching themes and patterns can be deduced. The method is commonly used to process qualitative data as it is very flexible and can be applied to handle a wide range of different research questions and data types without any major change in execution. Thematic analysis is also useful for structuring large amounts of qualitative data since the coding process naturally divide the data into groups based on similarity. The method does however depend on the researchers ability to interpret the data which can be influences based on the researchers background and previous assumptions. This introduces the risk of biases in themes and it is therefore important that the researchers acknowledge their influence on the analytical process for transparency in the study.

4.9 User Profiles and Relations

The method User Profiles is very beneficial in understanding the users in any given system [22]. By establishing User Profiles based on gathered data, researchers can more accurately simulate and address the needs of the user base. There are four different classes of User Profiles, they are the Primary User, Secondary User, Side User and Co-User. The Primary User is a person who uses the product for its primary purpose, while a Secondary User is a person who uses the product but not for its primary purpose [23]. The Side User is a person who is affected by the product negatively or positively without having decided to use the product. Finally, the Co-User is a person who co-operates with a Primary or Secondary User in some way without using his or her product. Furthermore, the method also encompasses an extension in the form of User Relations, which are beneficial to understanding the relationships between different types of users.

4.10 Requirements-driven Development

Requirements-Driven Development (RDD) is a methodology designed to seamlessly integrate software into its operational environment by grounding the development process in four distinct phases [1]. The process begins by analyzing the organizational setting to identify key actors, their strategic goals, and the inter-dependencies between them. The focus then shifts to the operational environment, where the systems specific functional and quality requirements are defined. The next step is to establishes the global system structure by defining subsystems and the data flow between them. Finally, the behavior and specifications of each individual architectural component are meticulously defined. By aligning these stages, RDD effectively reduces the impedance mismatch between a system and its environment, facilitating the smoother integration of both current and future system changes. However, the methodology faces challenges with scalability; in large-scale systems, mapping the complex architectural inter-dependencies between subsystems can become highly complex and time-consuming.

4.11 ACD^3

Activity-Centered Design is a framework designed to aid in the development and design processes, especially for complex interactive systems [24]. Part of this method is to divide aspects of the product into five different levels of abstraction. The first, Effect, pertains to the intended effect that the product is supposed to have on its environment. The second, Use, considers how the human and the product can work together to achieve the Effect. The third, Architecture, relates to the product structure in parts, or the technical architecture. The fourth, Interaction, explores the interaction between the human and the product in detail. The fifth and final aspect, Element, covers the technical elements of the system. When put together, tackling a design problem through these five levels of abstraction can serve as a solid base for defining Requirements and Guidelines for the product.

4.12 Human-Centered Requirements Engineering

The work of defining Requirements and Guidelines for an interactive product can be understood as drawing on perspectives from Requirements Engineering, Human-Centered Design, and Usability Engineering [25][26][27]. Together, these perspectives support the development of a comprehensive foundation for design by considering system needs, user needs, and usability concerns. Though similar, the differences between a requirement and a guideline are important to distinguish. A requirement is, in this case, something that the prototype should support or achieve, whereas a guideline describes design-oriented principles for how those requirements can be addressed. In this project, that work is further structured through the use of ACD^3, introduced in Section 4.11, whose five levels of abstraction provide a way to organize and formulate the resulting Requirements and Guidelines.

4.13 Brainstorming

Brainstorming is a technique used to foster group creativity by spontaneously sharing thoughts and ideas to reach solutions for practical problems [28]. Brainstorming involve the stimulation of a large quantity of ideas and the cultivation of an environment where *freewheeling* and the combination of ideas are encouraged. It is built on shared thinking among the members with the goal of producing ideas and concepts. One of the key strengths of the method is the ability to spiral ideas by building on and jumping off other peoples ideas, which often leads to unusual solutions for seemingly difficult problems [29]. Brainstorming is a classic method for ideation but has been reported to produce fewer ideas from a group compared to the number of ideas from the equivalent amount of members alone. Participants may suffer from a fear of criticism which can lead to members being stuck in a mental framework.

4.14 Technical Tools and Frameworks

Below are a few of the technical tools and frameworks used during this project.

4.14.1 Python

Python is a popular high-level programming commonly used for machine learning and simple scripting thanks to its large standard library and ecosystem of third-party libraries. It uses a syntax that mimics plain english which makes it easy to read and use. Python is often used to visualize data thanks the popular plotting library *Matplotlib* which makes it easy to create static and interactive visualizations.

4.14.2 Jupyter Notebook

Jupyter Notebook is "a publishing format for reproducible computational workflows" [30]. It allows researchers to analyze and process data while simultaneously documenting their process, ensuring that their results will be easier for peers to reproduce and validate.

4.14.3 React

React is a javascript framework which makes it easy to build interfaces using a components-based approach. It was developed by facebook and is one of the most commonly used front-end frameworks as of 2025 [31].

5

Process

The work began with planning out the project chronologically. The decision was made to split everything into four distinct phases. Phase 1 would focus on the pre-study and data-gathering. During Phase 2 the prototype solution would be designed and implemented based on the research done during Phase 1. Phase 3 was designated for live-testing the solution during a number of weeks, and gathering relevant data to analyze afterwards. Phase 4 was intended for final changes to the solution based on the performance during Phase 3, but also as a buffer for potential delays at earlier stages of the project.

5.1 Phase 1 - Pre-study

Phase 1 began with gathering information to design a suitable pre-study. This was done through multiple meetings with the different stakeholders, where the purpose of the many systems and user profiles inside the continuous integration flow was explained. Domain experts within the organization were also consulted on how to best extract usage data from involved CI software systems.

5.1.1 Survey

Whereas the original plan was to only conduct a number of semi-structured interviews, after meetings with the stakeholders this was expanded to include a large scale survey. This was done to reach a wider range of user profiles within the target organization, focusing on the XFT-developers (see Section 2.2.4), and to better account for the size and complexity of the CI workflow being studied. The survey was sent out via email to the target organization (~970 Ericsson employees), excepting the Flow Guardians (see Section 2.2.3), who were advised not to fill it out since they would participate in semi-structured interviews instead. The full survey can be found in Appendix B.

The first section of the survey focused on basic information; what role the subject has at the company, how many years they have spent with the company and how many years they have spent in their current role. This was done to contextualize responses by role and experience.

The second section focused on the five relevant interfaces within the CI workflow that were identified during the meetings with the stakeholders. They were:

- Follow Your Commit (Section 2.2.5)
- NoSA BDC Reports (Section 2.2.10)
- CFI Web Interface (Section 2.2.9)
- CI Flow Impediment (Section 2.2.8)
- RAN CI Radiator (Section 2.2.11)

For each interface, the subject was shown a screenshot of it and a link to the interface itself. They were then asked how often they use that particular interface, answering either "Daily", "Weekly", "Monthly" and "Never". Following this was the question "Please elaborate - we want to know when and what you use it for!" with a corresponding free-text answer field. The reason for this was to gather data on the level of usage for each interface along with users attitudes toward and perception of them.

The third section focused on the CATegorizer and CFI tickets in general. It began with a quick summary of what the CATegorizer is and a visual explanation of where inside the interfaces it is present. The first question was about how often the CATegorizer predictions factor into their work, with the same answer options as before (Daily, Weekly, Monthly, Never) and a following free-text answer field. This was followed by a group of Likert-scale statements (as described in Section 4.4) where the subjects would rate their level of agreement with the statement, ranging from

Strongly Disagree to "Strongly Agree". The Likert group was followed by another free-text answer field for any other thoughts the subject might have regarding the statements. These questions were intended to gauge both the relevance of the CATegorizer to their work and their stance toward it. The next three questions were all regarding CFI tickets in general, asking how often the subject would open and close tickets as well as label the Problem Type for any tickets. These questions were meant to explore the subjects' current habits regarding CFI tickets.

The fourth and final section featured SAM, the Self-Assessment Manikin (see Section 4.6). The intention behind this section was to understand the users' emotional state when working with the systems in the CI flow. Subjects could rate their Valence (negative - positive), Activation (calm - excited) and Control (dominated - dominant) on a scale from 0 to 10. This information served to identify which parts of the system were the most challenging for users and which areas they preferred, offering a clearer picture of their overall user experience. The final question featured a thank you to the participants and an opt-in for allowing the researchers to contact the subject in the case of further questions.

The survey was sent out via email to the subjects and kept live for a week. During this time two friendly email reminders were sent out, encouraging the subjects to participate.

5.1.2 Semi-structured Interviews

Semi-structured interviews were carried out in-person with the three Flow Guardians based at the thesis local site (see Section 2.2.3). The Flow Guardians were considered to be domain experts, which is why the choice was made to conduct more in-depth interviews with them rather than just the survey. These interviews were structured similarly to the survey, but with added sections and complexity to extract more domain-specific knowledge than a survey could. The form used during the interviews can be found in Appendix C. The form was filled in by the researchers during the interview.

The first three sections were the same as those in the survey, with section one asking about basic info, section two about system usage, and section three about the CATegorizer and CFI tickets (see Section 5.1.1 for more). The fourth section was dedicated to completing a NASA-TLX analysis (see Section 4.5) based on the subjects BDC troubleshooting process, which they gave in answer to Question 15: *"What does your BDC troubleshooting process look like? Please explain step by step. This answer will be used in a later section!"*. The reason for using NASA-TLX was to gauge the subjects' perceived mental workload when interacting and working with the systems. Following this the subjects completed the SAM form based on the same use case as the NASA-TLX section, exploring their emotional state when working in the systems. The final section featured part of the Applied Cognitive Task Analysis method (ACTA), see Section 4.7. The featured part, Knowledge Audit, was applied to maximize the amount and depth of domain-expert data that would be extracted. The subjects were asked each of the Basic Probes in turn and encouraged to provide concrete examples. After this the subjects were thanked for their participation and relevant notes were compiled.

5.1.3 Usage Analytics

A usage analysis was conducted to evaluate the performance of the CATegorizer. The goal was to capture the tool's initial predictive accuracy and the distribution of predicted error types. This baseline would later serve as a reference point for evaluating whether the proposed interface changes resulted in a measurable increase in the volume of user-labeled tickets. As described in the methods chapter, usage analysis draws on passively collected behavioral data, making it well-suited for quantitative comparison across interface iterations without introducing observer effects.

The data for this analysis was sourced from the CI Flow Impediments project in Jira (see Section A.6), the organization's project management and issue-tracking system. Jira stores all historical ticket data, including CATegorizer predictions, making it possible to extract a comprehensive record of the tool's usage in a production environment. This made it well-suited for a retrospective usage analysis covering periodic ticket volume, prediction outcomes and categorical distributions.

To extract the relevant data for the baseline, a series of queries were formulated using Jira Query Language (JQL)(see Section A.6) and extracted the CFI ticket data from Jira. The interesting data was the total volume of tickets as well as distribution

of the different problem types. Since the goal was to capture a snapshot of the CATegorizer's initial state, the queries were restricted to tickets closed within 2 weeks , 4 weeks, 26 weeks and 52 weeks (approximately one year) as the maximum. Multiple time periods were used to identify trends and build a more complete picture of the tool's performance. As the CATegorizer is only active in the *CI Flow Impediment* project, the queries were also scoped to that project. Additionally, the baseline was restricted to human-created tickets. For example, a bot created by another organization within the CI flow called Dekrabot automatically generates a large volume of tickets that the CATegorizer does not consistently process. Extending the CATegorizer's coverage to bot-generated tickets represents a separate system-level decision and falls outside the scope of this study. The exact JQL queries used to establish the baseline are listed in Appendix D.

5.1.4 Quantitative Analysis

This section will clarify how gathered quantitative data was processed and analyzed.

Survey Data

The survey data was processed using Jupyter Notebook with the Python packages Pandas, NumPy and Matplotlib for processing and analysis (See Section 4.14.2). The data from each section was compiled into suitable graphs for easier interpretation.

A simple bar chart was chosen for the first question (in the Basic info section) to intuitively show the spread of roles among respondents. The free-text role answers were also processed to ensure that no duplicate roles would appear in the graph (for example, answers like "Software Developer", "sw developer" and "s/w developer" were grouped together under the term "Software Developer"). This was followed by two histograms for the data on how many years respondents had spent at Ericsson and how many years they had been in their current role.

For the next section, System usage, the answers were compiled into a stacked horizontal bar chart with one bar for each interface. Percentages were added inside the sections of the bars showing the exact distribution of answers (Daily–Never). The same approach was chosen for the Likert question in the third section, showing the responses for each statement (Strongly Disagree–Strongly Agree), with average responses for each statement also being calculated. For the questions regarding BDC troubleshooting and how often CATegorizer predictions factor into the work, these responses were grouped together in another stacked horizontal bar chart. Two heatmaps were then created, one for the frequency of users opening CFI tickets (see Figure 6.7) and another for the frequency of users closing them (see Figure 6.8). Responses of "Never" were excluded from these heatmaps in order to focus the comparison on respondents who had at least some experience of opening or closing CFI tickets. "Never" responses were still considered separately in the overall interpretation of the results. Both heatmaps were cross-tabulated by role.

Qualitative data from the free-text answers was also explored quantitatively through

n-gram analysis (with bigrams and trigrams). The results were visualized in heatmaps and displayed in Section 6.2.5. The aim was to provide an exploratory overview of recurring terms, phrases and possible patterns and attitudes among respondents, which was necessary due to the volume of answers. Finally, the results from the SAM-section were compiled into three histograms, one for each aspect: Valence, Activation and Control. These were then combined into one grouped bar graph to save space in the final report (See Figure 6.2.4).

Interview Data

The quantitative data from the interviews was processed in much the same way as the data from the survey. However, due to the small number of respondents (n=3), this should be viewed more as an exploratory analysis rather than inferential. This is in contrast to the survey analysis, which focused on distributions across a larger respondent group, whereas the interview analysis primarily visualized individual participant patterns. The resulting graphs were therefore used to support and contextualize the qualitative findings rather than to generalize to a larger population.

Usage Analytics

The data collected from Jira was exported as a CSV file and imported into a Jupyter Notebook for processing and visualization 4.14.2. The objective was to establish the distribution of problem types among existing closed tickets, and to compare this distribution with and without tickets created by the test generation bot dekrabot. This was achieved by loading the CSV file into a pandas dataframe and converting the occurrence count of each category into percentages, which were then visualized as stacked bar charts. To facilitate direct comparison, the two distributions were rendered side by side within the same figure. The resulting plot is shown in Figure 6.2.

Since this thesis focuses on the COIN organization, the distribution within that organization was examined separately. The same dataframe was used, filtered to include only the COIN organization subset. As dekrabot is not active within this organization, it was excluded from this analysis. The resulting plot is shown in Figure 6.1.

The primary purpose of the CATegorizer is to correctly predict the problem type of a CFI ticket, making classification accuracy a critical measure of model reliability. Predictive accuracy was therefore measured as part of the baseline evaluation, providing a reference point against which any performance changes introduced by the improvements proposed in this thesis can be assessed. Since the CATegorizer does not produce classifications of type Undefined, predictions on tickets that are later manually set to Undefined will always be interpreted as incorrect. This creates a direct disadvantage against the AI model's measured performance (when measuring accuracy) that it can not control, potentially skewing the data. To account for this, accuracy was measured both inclusive and exclusive of Undefined tickets. The results are presented in Table 6.1.

5.1.5 Qualitative Analysis

This section will clarify how gathered qualitative data was processed and analyzed, mainly through thematic analysis (see Section 4.8).

Survey Thematic Analysis

The thematic analysis of the survey data began in the Jupyter Notebook by preparing the data through sorting free-text responses by question and frequency of use. For example, the answers regarding FYC were grouped together and sorted by "Never" to "Daily". This was done to make it easier to read and extract themes. However, due to the volume of answers, computational support was used to assist the coding process. The decision was made to leverage an AI LLM model in this task and an OpenAI GPT model was chosen, accessed via the OpenAI API. All data was anonymized and striped of any sensative information to ensure that no personal data or proprietary Ericsson information was shared with the LLM.

The previously described n-gram analysis was used as an exploratory step to identify recurring terms and phrases in the material. These recurring patterns were then interpreted and used to inform the development of preliminary code families. The LLM would then match these code families to responses along with a confidence score and short reasoning, as well as a secondary code family. The confidence scores were used as indicators for which responses required closer manual review, rather than as measures of validity on their own.

The first run resulted in many responses being matched to the preliminary code families, but many responses were labeled as "Other / unclear" or had very low confidence scores. These responses were then manually analyzed with further code families being extracted and provided to the LLM for the following runs. Some of the final code families were "Manual verification / checking logs", "Workflow pressure / time constraints" and "I didn't know this existed". The full list of code families can be found in Section 6.2.5. The final code families covered all of the responses with only a few still being labeled as "Other / unclear" (mostly through misinputs/misunderstandings by the respondents). The LLM-processed data was also manually checked for inconsistencies. The code families themselves were then manually grouped together into themes, completing the analysis (see Section 6.2.5).

Interview Thematic Analysis

The thematic analysis of the interview data followed a similar overall process to the survey analysis, but manually due to the smaller amount of data. Compared to the survey free-text responses, the interview responses were generally longer and more contextual, making it important to preserve the connection between response and question. The analysis began in the Jupyter Notebook by sorting the free-text responses by question into a CSV file and importing this file into Excel.

The responses were read multiple times to become familiar with the data and to identify recurring perspectives, experiences and concerns. Initial codes were created from the content of the responses with attention to both recurring statements and

individual insights. Similar codes were grouped together into code families that were then compared across respondents and questions to identify patterns relevant to the RQ.

Unlike the survey analysis, no LLM-supported coding was used for this data, since the number of responses was considerably smaller. The manual approach was deemed the most appropriate, as the smaller dataset made it manageable while still allowing for detailed interpretation and preservation of nuance. The resulting code families were then grouped into wider themes through manual interpretation, completing the analysis. These themes were used to complement the broader XFT-developer survey findings with the perspective from the Flow Guardians.

5.1.6 Defining User Profiles, Requirements and Guidelines

This section will present how the Requirements and Guidelines were formulated and defined based on the data gathered through the survey, interviews and usage analytics, as well as how the User Profiles and Relations were defined.

User Profiles and Relations

To begin the process, User Profiles and User Relations were defined according to the framework described in Section 4.9. The profiles were defined based on the data gathered through the survey and interviews, with particular attention to the users' roles, responsibilities, usage patterns, CFI ticket interaction and relationship to CATegorizer predictions. The interview data from the Flow Guardians was especially useful for understanding expert users and their relation to other users in the CI flow, while the survey data provided a broader perspective of the XFT-developer group. After the user profiles had been defined, the relations between them were established. This step was intended to clarify how users interact with each other, the different parts of the CI flow and with the feedback loop surrounding the CATegorizer. In practice the process of defining user profiles and relations was more a synthesis rather than a new data-gathering or analysis step. Much of the relevant information had already been identified through the survey, interviews and thematic analyses, and was efficiently reorganized into user profiles and relations.

Requirements and Guidelines

The intention behind creating the Requirements and Guidelines (henceforth referred to as RGs) was to establish a foundation for the prototyping and implementation that was to be carried out in Phase 2, as described in Section 4.12. It was important to capture as many aspects of the interaction with the CI flow as possible, which is why the ACD^3 method was chosen as the basis for this process. As explained in Section 4.11, the ACD^3 method divides interaction into five levels of abstraction: Effect, Use, Architecture, Interaction and Element. By creating RGs for each of these levels the most comprehensive coverage could be achieved. Based on the data from the previous analyses, RGs were formulated and then matched to the corresponding level of abstraction. In theory there is no limit to the number of RGs that could

be created depending on the level of detail chosen by the researchers, and to keep the scope manageable the aim was to create approximately five requirements and five guidelines for each level of abstraction. A naming convention was established to label each RG, with the name of the level of abstraction followed by R for requirement and G for guideline, and a number. For example, the first requirement at the Effect level was labeled as "Effect R1".

5.2 Phase 2 - Prototyping and Implementation

During the second phase of the thesis, the aim was integrate a solution into the pre-existing system to increase the volume of tickets labeled. All proposed features in the solution were based on the results from the pre-study (See Section 5.1).

5.2.1 Brainstorming Feature Ideas

To carry the findings from Phase 1 into Phase 2, several brainstorming sessions were conducted with varying participants. The first iteration was carried out by the researchers independently, each equipped with pen and paper, under a 15-minute time constraint. The objective was to generate and record any ideas or sketches that arose without filtering or evaluation. Once the time elapsed, the outputs were shared and discussed collectively among the researchers. The resulting ideas are presented in Section 6.5. These ideas were subsequently brought to a consultation with domain experts to assess their viability. The purpose of this second session was twofold: to evaluate whether the proposed ideas were compatible with established ways of working at Ericsson and to determine their overall feasibility. The session also served to verify, with input from the experts, whether the proposed features were likely to advance the thesis objective of increasing the number of labeled tickets. All ideas deemed sufficiently concrete were documented in a spreadsheet for further processing.

5.2.2 Feature Prioritization and Implementation Cost

To prioritize the proposed features for implementation, an improvement score was calculated for each idea according to Equation 5.1. The formula was developed in collaboration with the thesis supervisor to estimate the overall value of implementing each feature. The score was derived by first estimating the number of users reached through the feature's selected interface, drawing on survey results and domain expert estimations. The calculation also incorporated traffic from the most visited interface, *Follow Your Commit*, to ensure that features with lower direct traffic but a presence in *Follow Your Commit* were not undervalued. This combined user reach was then multiplied by an estimated interaction rate reflecting how likely a user would be to engage with the feature which was itself informed by both domain expert assessments and survey results. The result was further multiplied by the estimated frequency of interaction per week, yielding a total estimated amount of weekly clicks by users. This value was finally divided by the estimated number of days required for implementation, ultimately producing the improvement score.

The following equation summarizes the calculation of a features improvement score. The score should be understood as a heuristic prioritization tool rather than an exact prediction of future usage.

$$\text{Improvement Score} = \frac{(U_I + U_F) \cdot P_I \cdot N_W}{D} \tag{5.1}$$

Where

U_I	Users reached through interface
U_F	Users reached through Follow Your Commit
P_I	Percentage of users interacting with the feature
N_W	Number of times the feature is interacted with per week
D	Number of days to implement the feature

Features were then ranked by this score to identify those offering the greatest impact relative to implementation effort. The calculations was reviewed and validated by the domain experts. Due to time constraints, only a subset of the proposed features could be selected for implementation.

The final implementation score sheet was presented to the interfaces operations team, who reviewed and discussed which features were viable to integrate into the existing software. This consultation marked a significant shift from the originally planned implementation scope.

During the course of the thesis, the operations team had in parallel developed a new tool called Pandora, consolidating *Follow Your Commit*, *CFI Web* and several other interfaces not previously considered into a single unified software. As this development occurred concurrently and was not known at the outset of the thesis, it had not been accounted for in the initial implementation planning. A deeper discussion regarding this pivot and its implications can be found in Section 7.5.

The operations team approved the features proposed for CFI Web and CFI Web Details (see Appendix G for the full implementation cost calculations), but rejected the Follow Your Commit feature on the grounds that the interface had not yet been successfully migrated to the new software. The CFI Web Details feature was approved for implementation in its original interface; however, the CFI Web dashboard was only approved as a design implementation, to be realized in Pandora rather than the original interface. A key condition set by the operations team was that all new features must be developed within the new unified software, as the legacy interfaces were scheduled for discontinuation.

5.2.3 Implementation of new features

The final step of Phase 2 was to implement the approved features into the existing software. As the operations team had restricted changes to the new platform Pandora, the researchers were granted access to its source code. To support the development process, a buddy from the operations team was assigned to the researchers to answer questions and provide guidance.

The CI workflow required developers to commit code that was automatically verified by a CI checker, ensuring that no compilation or build errors were introduced. All code changes were additionally reviewed by the operations team to confirm that the code standard matched the existing code base before being accepted.

The development process was iterative, with continuous validation conducted alongside the assigned operations team buddy. Each validation session took the form of a structured conversation between the researchers and the buddy, focusing on the current state of the interfaces and allowing both parties to exchange ideas and opinions on design choices and components.

Two domain experts were invited to two separate demo sessions, in which the researchers presented the current state of the interfaces and gathered feedback on their effectiveness in motivating users to label more unlabeled tickets. Each session followed a consistent structure: a short demonstration by the researchers, an initial impression from the domain experts, and an open discussion identifying changes needed to address any raised concerns.

After approximately three weeks of iterative implementation and validation, the interfaces were demonstrated to an XFT developer (see Section 2.2.4) for a brief sanity and quality check from a developer's perspective. The interfaces raised no issues and were shortly thereafter deemed complete and approved for deployment and live testing by the operations team. The final interfaces are presented in Section 6.7.

5.3 Phase 3 - Live Testing

With the new features fully implemented, the system required testing to evaluate the impact of the changes. The live testing was conducted over a period of 10 workdays (2 workweeks), matching the shortest interval from the measured baseline discussed in Section 6.1. During this period, the changes were live-integrated into the pre-existing system to capture real usage data reflecting how much the features contributed in practice.

At the start of the 10-workday period, an email was sent to the entire COIN organization (see Section 2.2.2 for more information) to announce the new interfaces, as they had been implemented on a platform that had not yet been publicly announced. The email described the CATegorizer dashboard, what it offered and the rationale behind its development. It also included a link to the dashboard so that recipients could explore it directly, as well as a link to a survey where they could share their opinions on the changes. A separate email containing the same survey was also sent to individuals who had participated in the pre-study and indicated they were open to being contacted, in order to gather targeted feedback from users whose input had directly informed the design (see Appendix E for more information about the survey).

The survey collected both quantitative and qualitative data. Quantitative responses took the form of Likert scale answers addressing questions about the clarity and purpose of the dashboard, as well as respondents perceptions of the impact the dash-

board would have on the CATegorizer as a tool and on the quantity of training data. Qualitative data was collected through free-text responses covering the dashboard's design and its effect on CATegorizer training.

The changes to CFI Web Details were not advertised, as they were part of an already existing interface and therefore did not require users to seek out a new platform. Consequently, these changes were not addressed in the survey but were instead raised during interviews with stakeholders and domain experts, in order to gather opinions from individuals with either deep knowledge of the software or direct experience as users of the system.

5.4 Phase 4 - Evaluation of Solution

The evaluation of the solution was conducted from three different angles. The first was to collect a broad set of opinions. This was done by sending an email to all developers in the COIN organization, as described in Section 5.3. The second was more in-depth discussion of the solution. One representative from each stakeholder group was interviewed. This included the person who initiated the study from Ericsson's side, one developer of the software into which the new interfaces were integrated, one Flow Guardian (see Section 2.2.3 for more information) and an XFT developers (see Section 2.2.4 for more information). These groups had been identified as the primary stakeholders throughout the thesis and given the considerable differences in their respective roles, it was deemed as necessary for all stakeholder groups to be included in the interviews. The interviews shared the same structure as the online survey but also included a cognitive walkthrough of the dashboard and CFI Web. The walkthrough covered different tasks to understand how well users can navigate the interface. An explanation of these tasks and some of the surrounding questions can be found in Appendix I.

Their thoughts were noted down for each task. The third source of evaluation was a usage analysis conducted in the same manner as the baseline measurement carried out during the pre-study phase, as described in Section 6.1. The purpose of this analysis was to determine whether more tickets were being labeled following the introduction of the new interfaces. That is, whether the interfaces had succeeded in addressing the problem this thesis set out to solve.

The findings from these three sources were subsequently combined and analyzed to assess whether the new interfaces had a meaningful impact on the amount of tickets labeled, both from a quantitative perspective shown by the usage analysis and from a qualitative analysis from the stakeholders opinion and expectations gathered from the interviews and survey.

6

Results

This chapter will present the results from each of the phases described in the Process chapter 5, both in the form of empirical findings and generated design output. The chapter is structured chronologically, closely following the structure from the Process chapter.

6.1 Pre-study Baseline

The following section presents the baseline results collected during the pre-study phase prior to the implementation of the proposed improvements. The purpose of this analysis was to establish an initial understanding of the existing distribution of categories in closed tickets, the influence of automatically generated ticket and the predictive performance of the CATegorizer system. The results presented in the following subsections serve as a reference point for evaluating potential changes and improvements introduced later in the study.

6.1.1 Problem Type Distribution within COIN

Figure 6.1 shows the results of the usage analysis described in Section 5.1.3. The analysis covers the 52 weeks preceding the 28th of February 2026, with that date serving as the endpoint of the investigated period. During this period, the distribution of closed ticket problem types within the COIN organization remained relatively stable at around 35%. The results also show that the number of tickets closed with a problem type of *"Undefined"* decreased by 27.8 percentage points, which corresponds to a decrease of 46.33%.

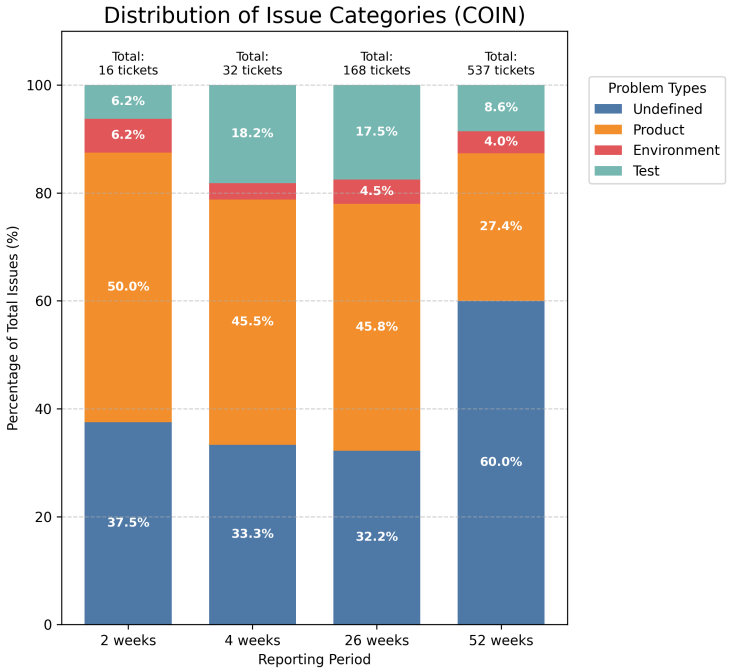


Figure 6.1: Stacked bar graph of closed tickets' problem type distribution, where the COIN organization is impediment owner, over 2, 4, 26 and 52 weeks

6.1.2 Effect of Dekrabot on Ticket Distribution

Figure 6.2 shows the distribution of issue categories across the same reporting periods (2, 4, 26, and 52 weeks), segmented by reporter type: All Reporters, dekrabot, and Not dekrabot. Across all reporting periods, tickets created by dekrabot are overwhelmingly classified as Undefined, with the proportion increasing from 85.3% at 2 weeks to 93.4% at 52 weeks. Human-written tickets, by contrast, exhibit a considerably more balanced distribution across the four problem types Undefined, Product, Environment, and Test with the Undefined category accounting for only 38.7% to 42.5% of tickets depending on the reporting period.

As the total ticket volume grows substantially over time, from 1,250 tickets at 2 weeks to 30,066 at 52 weeks, dekrabot's dominant share of Undefined tickets increasingly influences the aggregate distribution observed in the All Reporters group, where the Undefined proportion rises from 71.3% to 80.9% over the same period.

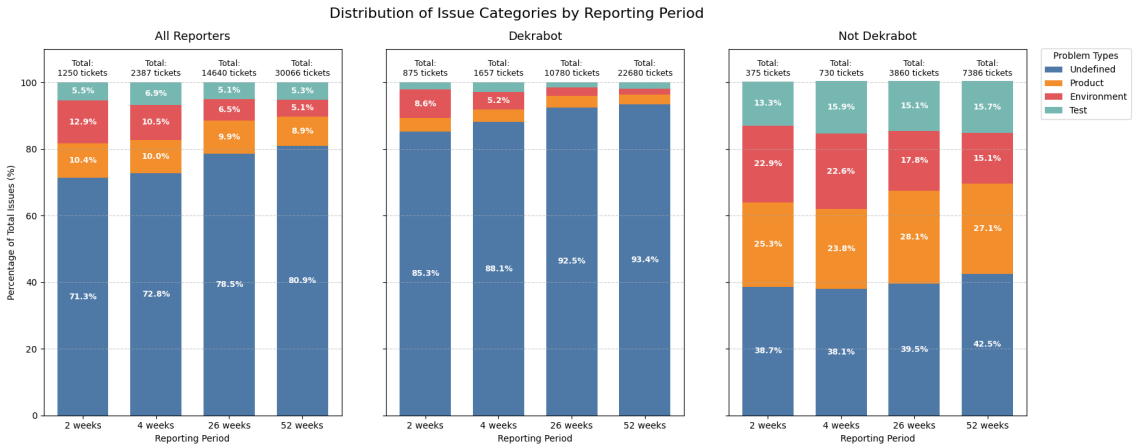


Figure 6.2: Stacked bar graph of ALL closed tickets' problem type distribution comparing all reporters, only dekrabot (automatically created tickets), and excluding dekrabot (only human-written tickets) over 2, 4, 26 and 52 weeks

6.1.3 CATegorizer Prediction Accuracy

The predictive accuracy measured during the pre-study is presented in Table 6.1. The results show that out of 264 total tickets evaluated, the CATegorizer classified 116 correctly and 148 incorrectly, yielding an overall accuracy of 43.94%.

When excluding Undefined as a manually labeled category, approximately one third of all tickets were excluded from the total count. This increased the accuracy to 65.91%, while the number of incorrectly classified tickets decreased from 148 to 60.

Metric	**Total Results**	**Excluding Undefined**
Total Count	264	176
Correct	116	116
Incorrect	148	60
Accuracy	**43.94%**	**65.91%**

Table 6.1: Accuracy Metrics Evaluation

6.2 Pre-study Survey Results

The survey from the pre-study in Phase 1 resulted in a dataset containing both qualitative and quantitative responses. For information regarding how this data was collected, processed and presented, see Section 5.1. As mentioned the survey was sent out to approximately 970 people within the organization and received 60 responses during the five days that it was live. Of the 60, 2 answers were removed (one incomplete, one from a Flow Guardian), resulting in a total of 58 responses in the dataset to analyze.

6.2.1 Respondent Background

Approximately 65.6% of respondents reported to have the roles of “Developer” or “Software Developer”, as shown in figure 6.3. Many of the other answers suggested a role similar to those two, with slightly different naming, such as “R&D Software Developer” and “5G Software Developer”. Others were more clearly distinguished from the developer role, such as “Product Guardian” and “Line Manager”, pointing to more senior roles.

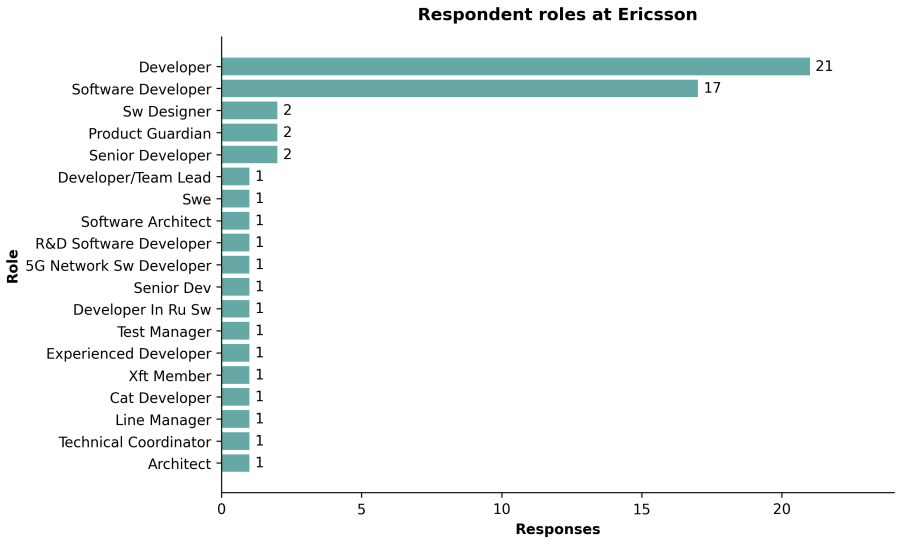


Figure 6.3: The distribution of reported roles from the pre-study survey. Majority developers (n=58)

In the histograms shown in figure 6.4 and 6.5 the responses to Time spent with Ericsson and Time spent in current role can be seen. Respondents reported a wide range of times, from less than 1 to over 38 years. However, the majority of answers fell within the time span of 0–10 years, potentially aligning with the role distribution mentioned previously. Viewed together with the role distribution, this data points to most respondents being developers with less than ten years experience at the company, while also including a smaller number of respondents in more specialized or managerial roles.

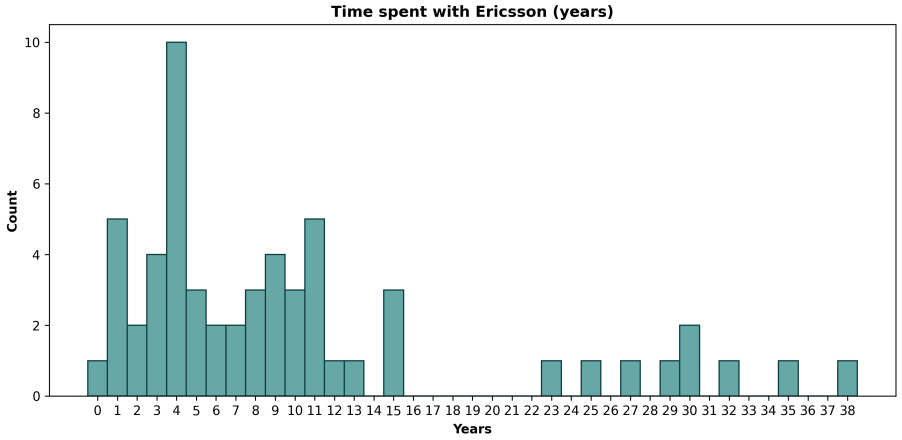


Figure 6.4: Distribution of respondents' time spent with Ericsson (n=58)

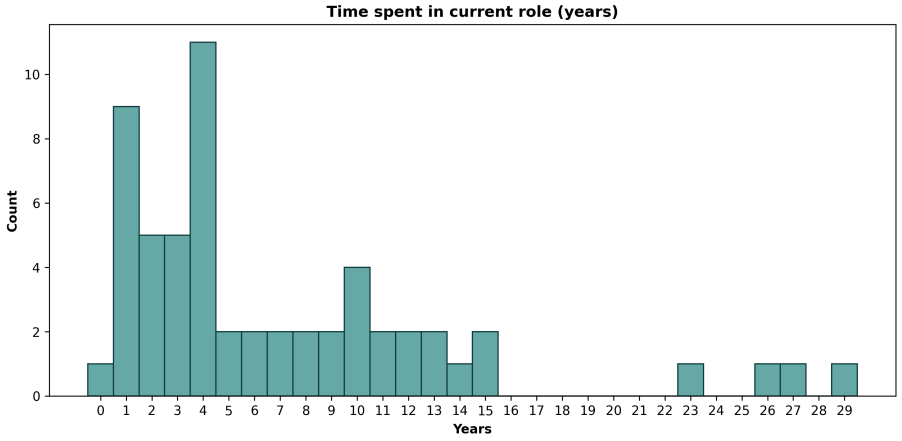


Figure 6.5: Distribution of respondents' time spent in current role (n=58)

6.2.2 System Usage

The survey revealed large differences in usage between the different systems, as shown in Figure 6.6. Follow Your Commit (FYC) was the most frequently used system among respondents by a substantial margin, followed by the RAN CI Radiator (RAN). While FYC's high usage was anticipated based on preliminary stakeholder meetings, the widespread adoption of RAN was unexpected.

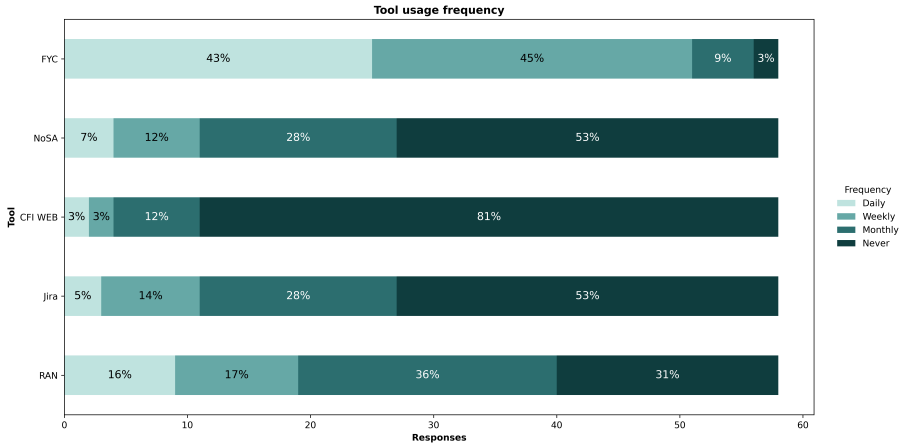


Figure 6.6: The system usage frequency (n=58)

The NoSA and Jira systems exhibited highly similar usage patterns; 53% of respondents reported never using them, 28% used them monthly, 12% and 14% used them weekly (NoSA and Jira, respectively), and 7% and 5% reported daily usage. Lastly, the least utilized tool was shown to be CFI WEB, with 81% of respondents never using it, 12% monthly, 3% weekly, and 3% daily.

Given that CFI WEB was identified as the least utilized tool, a deeper analysis was conducted regarding how users interact with its specific ticketing workflows. Figure 6.7 illustrates the frequency with which respondents create CFI tickets, revealing that the creation of CFI tickets by non-Flow Guardians rarely happens.

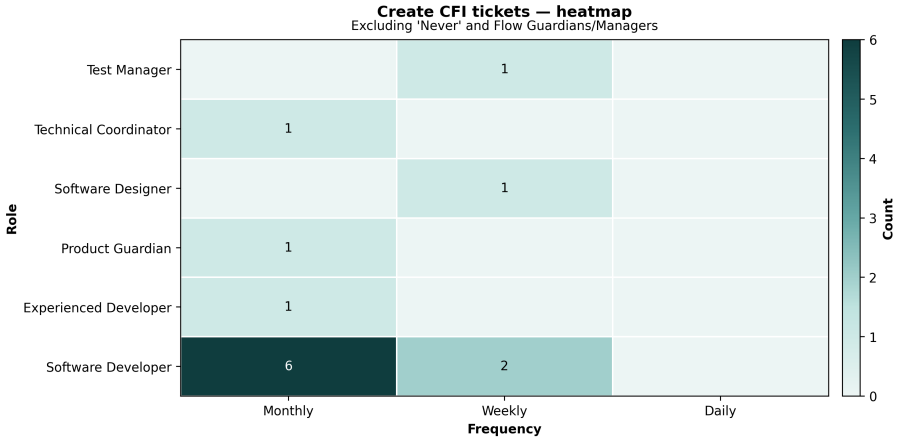


Figure 6.7: Heatmap of CFI ticket create frequency

To determine whether ticket resolution followed a similar trajectory, the frequency of closing CFI tickets was also plotted. As shown in Figure 6.8, the closure rates demonstrate a close alignment with the creation intervals in Figure 6.7.

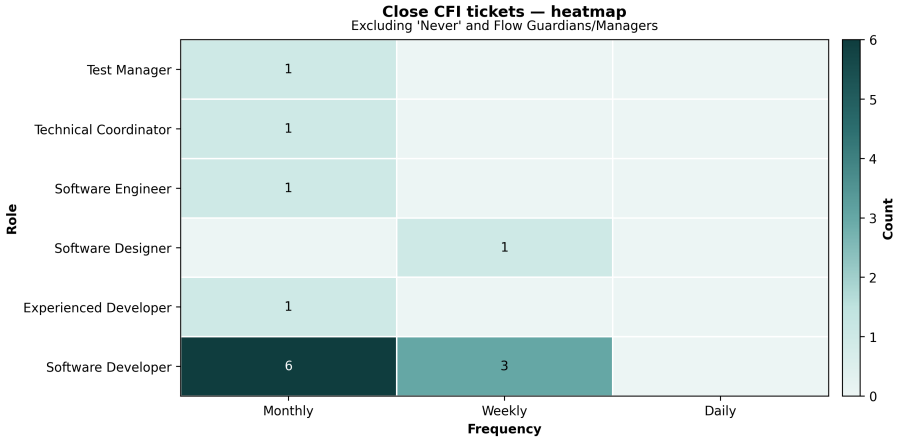


Figure 6.8: Heatmap of CFI ticket close frequency

6.2.3 CATegorizer Perception and Engagement

The results from the Likert-section of the survey (shown in figure 6.9) indicated low familiarity with the CATegorizer before the survey, with 65% of respondents answering Strongly Disagree or Disagree. Regarding predictions being valuable to their work, 60% of respondents answered Neutral. When asked about active engagement with the CATegorizer, only 4% answered positively, indicating a very low level of active engagement. Passive engagement showed slightly higher positive numbers, with 15% of respondents agreeing that they passively engage with the CATegorizer. As for wanting improvement most respondents were neutral, with a vast majority at 69% of respondents answering as such. Only 26% of respondents were positive about the CATegorizer improving. However, the percentage of negative responses was even lower at only 5%. The final statement about whether respondents would like to contribute to improving the CATegorizer received mostly negative or neutral responses.

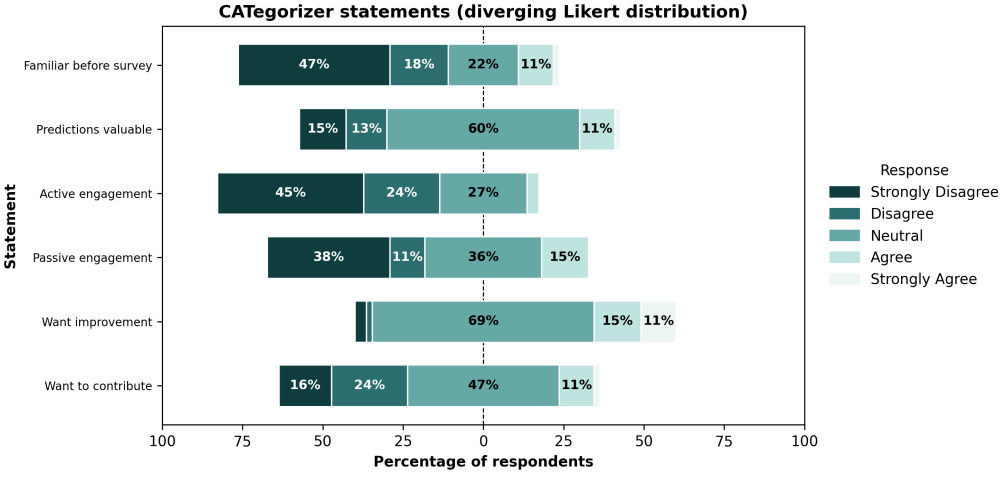


Figure 6.9: Distribution of answers to the Likert statements (n=58)

To contextualize the usage of the CATegorizer within the specific task it was designed to support, the frequency of troubleshooting BDCs was compared to how often the CATegorizer factors into the respondents' daily work. Figure 6.10 displays this comparison, which demonstrates that the CATegorizer was not utilized to its full intended extent during the troubleshooting process.

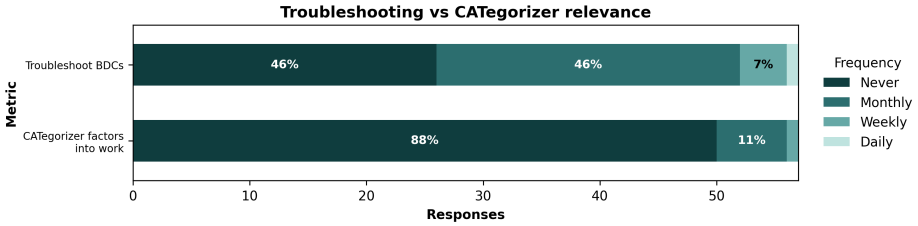


Figure 6.10: Distribution of users' troubleshooting BDCs and engaging with the CATegorizer

6.2.4 SAM Results

As part of the survey respondents filled out a SAM section, a research method that is explained further in Section 4.6. The results were visualized in a grouped bar graph, as shown in Figure 6.11. The results show a clear distribution around the middle of the scale (5), with some spread toward the higher end (6, 7). There is a notable secondary peak in the Control dimension where a spike can be seen at the value 3 on the scale. An important distinction to remember here is that the scale carries different meanings for the different dimensions. For example, a high Valence score could indicate that users feel very happy when working in the system, but the same score in the Activation dimension could instead point to the respondents feeling high levels of stress. Still, the results being grouped around the center as they are could indicate that most users find the emotional experience of interacting with the systems of the CI workflow to be somewhat balanced.

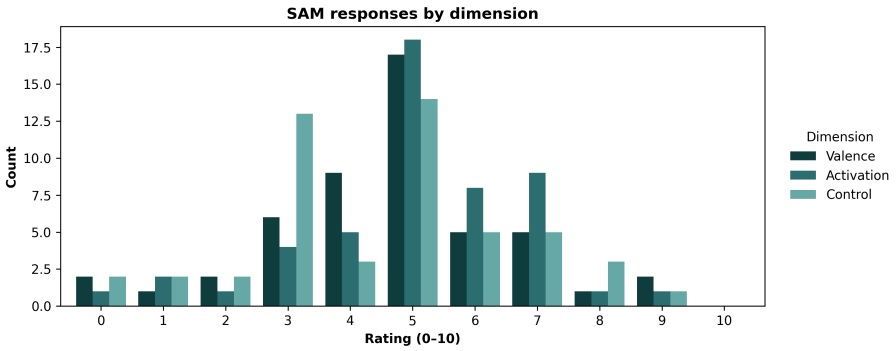


Figure 6.11: The compiled results of the SAM-section in the survey (n=58)

6.2.5 Survey Thematic Analysis

The initial n-gram analysis produced multiple heatmaps showing the most common bigrams and trigrams grouped by respondents' reported frequency of use. One of

the heatmaps can be seen in Figure 6.12 and in that example we can see that no bigrams appear in a frequency other than “Never”. Two of the found bigrams said “didn’t know” and “not familiar”, suggesting a lack of awareness regarding CFI Web among users. As mentioned in Section 5.1.5, these were only used to guide the creation of the initial code families for the full thematic analysis. The rest of the heatmaps can be found in Appendix H.

The thematic analysis of the survey free-text responses produced four main themes that highlight recurring aspects of the respondents’ experiences and attitudes related to the CI workflow. The themes were defined based on the extracted code families, which are also presented in the following sections. To avoid potential confusion, throughout this section the term “response” is used to refer to individual free-text answers provided by the respondents. Since each respondent answered multiple questions, there are many more individual responses than there are respondents.

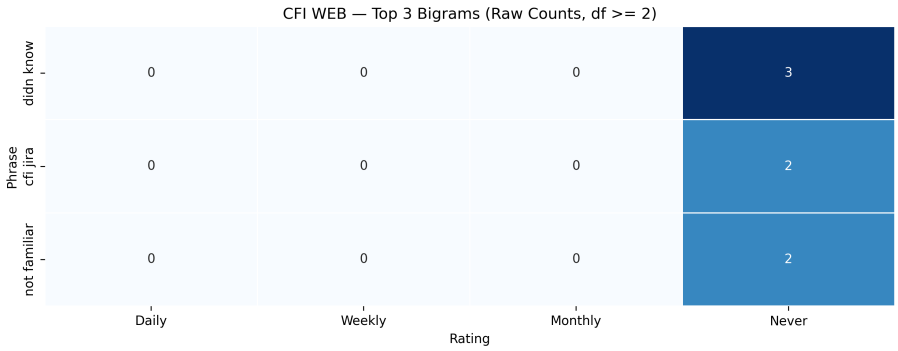


Figure 6.12: The top bigrams from the free-text answers to the CFI Web usage question

Survey Theme 1: More information/education regarding the system is needed

This first theme was based on the following code families:

- "Explanation / transparency needed"
- "I didn’t know this existed"
- "I lack knowledge to use this"

A total of 52 free-text responses were categorized as belonging to these code families and by extension Theme 1. The theme focuses on the lack of information and knowledge regarding the system among respondents. It points to an area of the CI flow in need of improvement that could lead to a meaningful improvement of the CATegorizers performance.

Survey Theme 2: Users retain control through manual investigation and monitoring

The second theme was based on the following code families:

- "Monitoring the system and/or my commits"
- "Manual verification / checking logs"
- "Low trust / manual confirmation needed"
- "Multi-system troubleshooting"
- "Problem identification / root cause finding"
- "Routine / habitual use"
- "Step-by-step troubleshooting workflow"

The 160 free-text responses belonging to these code families make it the broadest, most prominent of the four themes. This is in part due to the theme encompassing 7 different code families, 4 more than the other themes have. It is also worth noting that splitting this theme in two (one for monitoring, one for troubleshooting) was also debated, but ultimately decided against. The reasoning behind this decision was that monitoring and manual investigation were closely related in the respondents' answers. The theme covers how users control the systems and what they use them for. This is important to consider when making changes or adding features to the systems. Added features and changes should not impact the usability of the systems for these goals.

Survey Theme 3: The current predictions are useful, but mainly as a supporting tool

For the third theme, these were the code families:

- "Prediction supports prioritization / overview"
- "Prediction influences work / decision-making"
- "Prediction has limited role"

These code families encoded 18 different responses, making it one of the smaller themes but still an important one to consider. It focuses on the current attitudes towards the predictions made by the CATegorizer and the impact that it has on the users' everyday work.

Survey Theme 4: Practical constraints limit engagement

The fourth and final theme was based on these three code families:

- "Workflow pressure / time constraints"
- "Feedback seen as extra effort"
- "This is not my job or responsibility"

The 12 responses encoded in this theme make it the smallest, but also point to some of the more important aspects of the CI workflow environment and cross-organizational communication. Addressing concerns such as feedback being seen as

extra effort through making the feedback-process simpler could lead to increased feedback, which in turn could lead to an improvement of the CATegorizer's performance.

Together, these four themes suggest that engagement with the CATegorizer is influenced not only through the perceived usefulness of the predictions, but also through the users' understanding of the system, their existing troubleshooting and monitoring routines, and the practical constraints of the CI workflow.

6.3 Pre-study Interview Results

The answers from the interviews with the three Flow Guardians (See Section 2.2.3 for description) contain both quantitative and qualitative data. The interview was divided into different sections and the results are presented in the same manner.

6.3.1 Usage Patterns

During the interviews with the Flow Guardians, their usage of the various workflow tools and suggested tasks were mapped against the frequency of use reported by each Flow Guardian. The results shown in Figure 6.13 indicate that *Follow Your Commit*, *NoSA BDC Reports* and the *CI Flow Impediment* project in Jira were all part of the daily work of all three Flow Guardians. These were closely followed by *CFI Web* and *RAN CI Radiator*, which were used daily by two out of three. The results also show that one of their primary tasks was troubleshooting BDCs, which was unsurprising given that this is a core responsibility of the role.

Furthermore, the results indicate that Flow Guardians create and close CFI tickets on a regular basis. However, the CATegorizer rarely factors into their workflow, and they do not consistently label the problem type of tickets upon closing them.

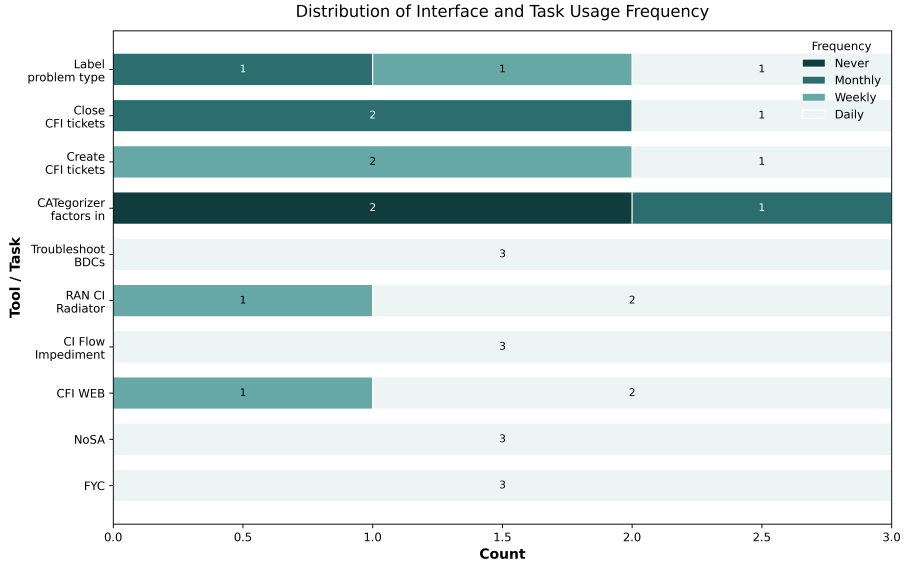


Figure 6.13: Usage patterns of Flow Guardians

6.3.2 Participants attitude towards the CATegorizer

To complement the questions about interface usage and task completion habits, the Flow Guardians were also asked about their attitudes toward the CATegorizer as a tool. The responses were collected using a Likert scale and are displayed in Figure 6.14. The results clearly show that all Flow Guardians are aware of the CATegorizer and unanimously agree that they would like the tool to improve. However, they also indicated that they do not actively engage with its predictions or incorporate them into their decision making, which is further reflected in their neutral view on the value of the predictions.

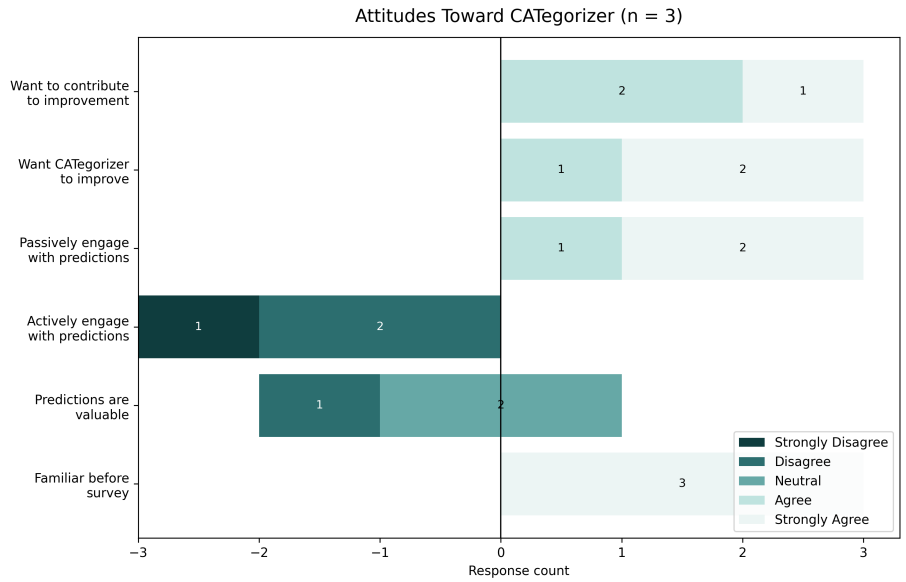


Figure 6.14: Likert scale answers about the CATegorizer from Flow Guardian interviews

6.3.3 Interview Thematic Analysis

The thematic analysis performed on the data gathered during the interviews resulted in four different themes. These themes were based on code families used to encode the interview responses, the same way they were used in the survey thematic analysis (Section 6.2.5).

Interview Theme 1: Flow Guardians rely on experience and manual investigation to stay in control

Theme 1 was based on the following code families:

- **Experience** - Experience matters greatly in the work, and is often more important than what the tools say.
- **Monitoring** - Much of the system is based on monitoring, being able to monitor existing issues for new occurrences, and monitor the CI flow for potential issues.

- **Context** - Gather information through the tools to gain an understanding of the context.
- **Efficiency of resources** - Use tools and allocate resources in the most efficient way possible. This is largely based on experience.

This theme focuses on how the interviewees interact with the systems in the CI workflow related to their specific tasks, and how their experience and knowledge allows them to leverage these systems in the best, most efficient ways. For example, one of the respondents told of how they created their own automated system to monitor for a very specific problem in a special way, pointing to their extensive knowledge and experience with the CI flow.

Interview Theme 2: CATegorizer predictions are not trusted enough to guide work directly

Theme 2 was based on the following code families:

- **Tools** - Tools are sometimes not accurate. This is especially true for the CATegorizer predictions, which are wrong often enough that they are normally disregarded.
- **Ideas** - Suggested improvements regarding expanded functionality of the CATegorizer.

The second theme focuses on the interviewees trust in the different tools and specifically the CATegorizer. The answers encoded with these families suggested a low sense of trust in the CATegorizer's predictions, but also shone a light on their ideas for potential improvements in different areas.

Interview Theme 3: CFI tickets are central to the workflow, but labeling is not consistently prioritized

Theme 3 was based on the following code families:

- **CFI Tickets** - Working and interacting with CFI tickets as part of the CI flow.
- **Labeling** - How and when the users label the problem type of a ticket.
- **Frequency of actions** - How often the users take certain actions in the systems. Varies greatly depending on the tool in question.

Theme 3 focuses on the role of CFI tickets in the interviewees' work, and highlights their central role in ensuring the CI workflow continues unimpeded. The answers with these codes focus on how the interviewees work with the CFI system, how often they label the problem types of tickets, and how often they take many other actions.

Interview Theme 4: Engagement is shaped by workload, communication, and cognitive effort

Theme 4 was based on the following code families:

- **Communication** - Parts of the system feature in-system communication options (the comments in Jira for example). This is an important avenue of communication for the users.

- **Fatigue** - How mental fatigue can affect the experience working in the system.

The final theme focuses on the human factor; how mental workload, fatigue and stress can negatively impact the performance of the users, but also the importance of clear channels of communication. It is therefore important that any changes made to increase user feedback should not increase any of the negative aspects, and not impact communication in a negative way.

Together, these four themes suggest that the Flow Guardians' relationship with the CI systems and CATegorizer is heavily influenced by their experience and domain knowledge. On average, their knowledge appears to be deeper and more detailed than the XFT-developers working in the same CI flow. This is important to consider when designing a solution to the RQ.

6.3.4 NASA-TLX Results

The NASA-TLX results from the interviews are presented as bar charts in Figure 6.15, displaying each respondent's score per category alongside the average of the three responses. The results reveal variation across categories. *Mental Demand* shows a wide spread among respondents, with a range of 40 points on a scale of 0 to 100, suggesting that Flow Guardians experience the cognitive load of troubleshooting very differently. *Physical Demand* is consistently low across all three respondents, which is expected given that the task is situated in front of the computer. Regarding the *Temporal Demand*, Participants 2 and 3 reported low time pressure, while Participant 1 felt noticeably more pressed for time. All respondents were aligned in perceiving their own performance as satisfactory and the effort required for troubleshooting as moderate. *Frustration* was yet another polarizing category, with Participants 1 and 2 reporting a calm experience during troubleshooting, while Participant 3 described themselves as relatively frustrated.

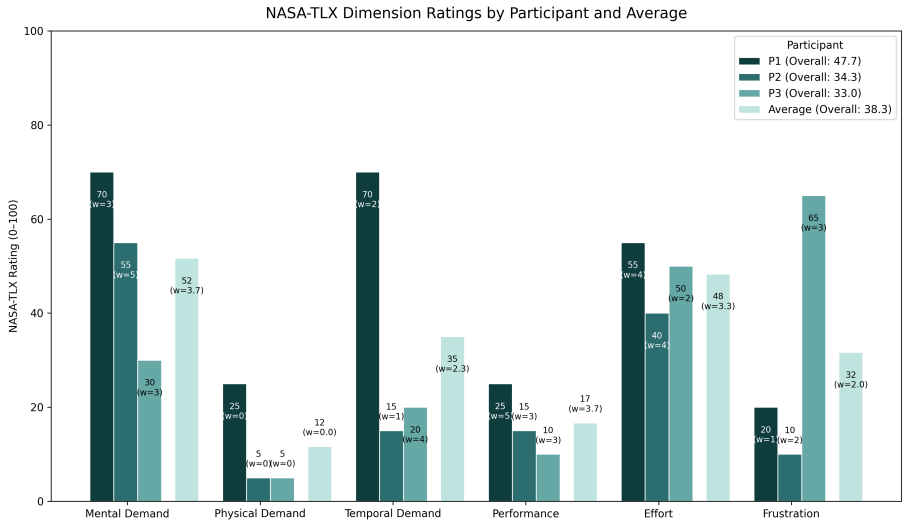


Figure 6.15: Results of the NASA-TLX performed during interviews three Flow Guardians

6.3.5 SAM Results

The Self-Assessment Manikin performed during the interview show certain themes from the respondents. The bars in Figure 6.16 show that the participants feel in the lower values when asked about activation when troubleshooting BDCs. This outcome was anticipated, as participants interpreted the activation measurement literally as physical movement, whereas the experimental tasks were performed entirely on a computer. The results also shows that the respondents generally feel in control when troubleshooting which is also expected given their experience with the task. The respondents also indicate that they are moderately to positive when troubleshooting.

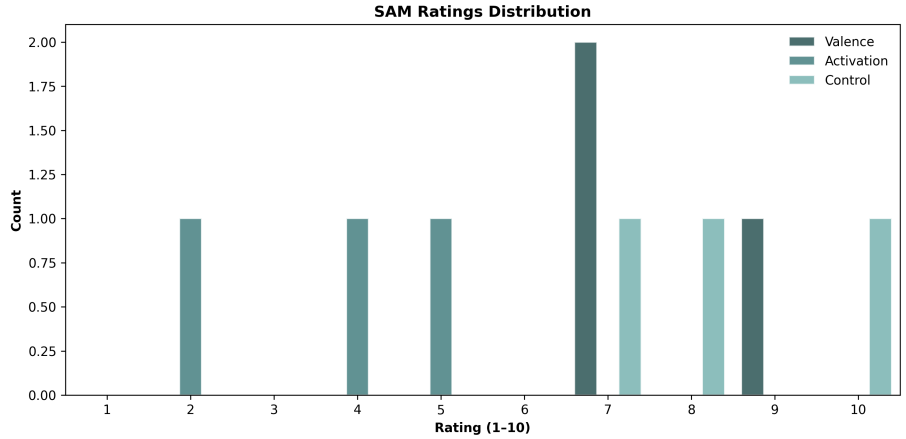


Figure 6.16: Self-Assessment Manikin scores from Flow Guardian interviews

6.4 Resulting User Profiles, Requirements and Guidelines

Requirements and Guidelines

The final requirements and guidelines were derived based on all the findings of the pre-study and the process of defining them are described in Section 5.1.6 and displayed in Table 6.2. The requirements and guidelines are divided into the different levels Effect, Use, Architecture, Interaction and Element for clarity.

Table 6.2: Requirements and Guidelines by Abstraction Level

Level	Requirements	Guidelines
Effect	**ER1.** The prototype shall encourage increased engagement with the CATegorizer.	**EG1.** Design for adoption within real workflow constraints, not idealized use.
	ER2. The prototype shall inform users of its value to their work.	**EG2.** Design appropriate channels of communication for the CATegorizer.
	ER3. The prototype shall integrate with existing CI workflows without disrupting them.	**EG3.** Aim to improve quality and frequency of user feedback to the CATegorizer.
	ER4. The prototype shall support more frequent and informed feedback without reducing perceived efficiency.	
Use	**UR1.** The prototype shall support users in quickly understanding what the CATegorizer says regarding the current issue they are handling.	**UG1.** Design the prototype for both fast-scanning and deeper investigation, since users and use cases may vary.
	UR2. The prototype shall allow users to provide feedback at a moment that aligns with their workflow and responsibility.	**UG2.** Place feedback opportunities where the users are already interacting with the CATegorizer.
	UR3. The prototype shall accommodate the diverse roles and responsibilities of its users.	**UG3.** Avoid assuming that all users will interact with the prototype in the same way.
	UR4. The prototype shall support users who want to verify the predictions against other sources.	**UG4.** Design so that interaction with the prototype feels like part of the troubleshooting process, not a separate step.

Continued on next page

Table 6.2 – *continued from previous page*

Level	Requirements	Guidelines
Architecture	**AR1.** The prototype shall include a clear and concise prediction presentation.	**AG1.** Structure the prototype into separate but connected parts: overview, explanation, verification/context, feedback.
	AR2. The prototype shall include a deeper explanation component to support users who want to understand why a certain prediction was made.	**AG2.** Design the system to reflect the observed user process.
	AR3. The prototype shall include a feedback component that allows for quick and easy feedback from users.	**AG3.** Keep components modular so that they can be surfaced at different stages.
	AR4. The prototype shall include a workflow interaction component that connects the CATegorizer interaction to the user's troubleshooting context.	**AG4.** Ensure the feedback component captures relevant information.
	AR5. The prototype shall differentiate between information needed for fast-scanning and deeper investigation.	
Interaction	**IR1.** The prototype should present CATegorizer predictions in a way that is immediately interpretable.	**IG1.** Design with progressive disclosure in mind: show essential meaning first and allow expansion for further explanation.
	IR2. The prototype should communicate confidence, uncertainty, or limitation in a way that helps users calibrate trust.	**IG2.** Make system reasoning legible in user-centered terms, not only model- or back-end-centered terms.
	IR3. The prototype should reduce the effort required to submit feedback.	**IG3.** Use interaction patterns that allow quick responses.
	IR4. The prototype should make clear what kind of feedback is wanted and why it matters.	**IG4.** Frame feedback as beneficial to the immediate task as well as future system quality.

Continued on next page

Table 6.2 – *continued from previous page*

Level	Requirements	Guidelines
	IR5. The prototype should avoid overloading the user with unnecessary information at the first point of contact.	**IG5.** Design for low-friction interaction within the workflow.
		IG6. Support verification and validation through other sources.
Element	**ElR1.** The prototype shall include visual elements for predicted category, confidence/uncertainty, short rationale, access to more details, and feedback action.	**ElG1.** Use labels and wording that are immediately understandable to users in the CI context.
	ElR2. The prototype shall include navigational or linking elements that connect the prediction to relevant evidence sources or related tools.	**ElG2.** Use concise microcopy to explain what the prediction means and what the user can do next.
	ElR3. The prototype shall visually distinguish explanation, evidence, and feedback from one another.	**ElG3.** Prefer interface elements that reduce interaction cost, such as prefilled context, selectable reasons, or one-click acknowledgments.
	ElR4. The prototype shall include cues that support discoverability for first-time or infrequent users.	**ElG4.** Include onboarding or contextual help elements for users with low awareness or low familiarity.
	ElR5. The prototype shall minimize unnecessary manual input when feedback is submitted.	**ElG5.** Ensure the visual hierarchy emphasizes what users need first: issue relevance, predicted cause, and next possible action.

User Profiles and Relations

The user profiles derived in Section 5.1.6 are displayed in Table 6.3. The profiles show three different user groups with varying degrees of direct engagement with the CATegorizer. No secondary user was identified within the scope of the this thesis but the category is displayed in the table to complete the framework.

Table 6.3: User Profiles

User Type	Profile in this project	General Profile
Primary User	The Flow Guardians within the COIN organization.	• Higher education (BSc most common) • Experience working in a CI flow • In-depth knowledge of systems deployed through the COIN CI pipeline • Highly experienced in troubleshooting at the BDC-level • Highly comfortable using the CFI ticket system
Secondary User	No secondary user identified in this project.	
Side User	Line Managers, who do not directly engage with the technical aspects but are influenced by the performance of the CATegorizer and its effect on their teams.	• Higher education (Masters standard, some Bachelors) • Many years of experience at Ericsson in various roles • Limited recent hands-on experience with the systems
Co-User	The XFT developers within the COIN organization.	• Higher education (commonly engineers from Chalmers) • Less CI flow experience than the Primary User • Detailed knowledge of their own system • Skilled programmers

6.5 Ideation of New Features

The brainstorming session described in Section 5.2.1 yielded multiple design concepts. Figure 6.17 presents proposed features, which were grounded in the insight that users lacked awareness of the tool, and that no dedicated interface existed to surface it. The sketch also addressed the disconnect between CFI Web, CFI Web Details, and Follow Your Commit. The pre-study had revealed a clear gap in user base across these interfaces, making it counterproductive to develop disconnected features in isolation. Introducing a visual and navigational red line between them was intended to reduce this separation and create a sense of unified workflow rather

than isolated, disconnected tools.

As shown in Figure 6.17, the proposed changes included adding a CATegorizer prediction to Follow Your Commit, along with a link to the corresponding CFI Web ticket page. The CFI Web ticket would in turn include a link to the CATegorizer dashboard, providing an overview of the tool's current diagnostic state. Additionally, the dashboard would be accessible directly through the CFI Web navigation panel, making it available to a broader range of users regardless of their typical entry point.

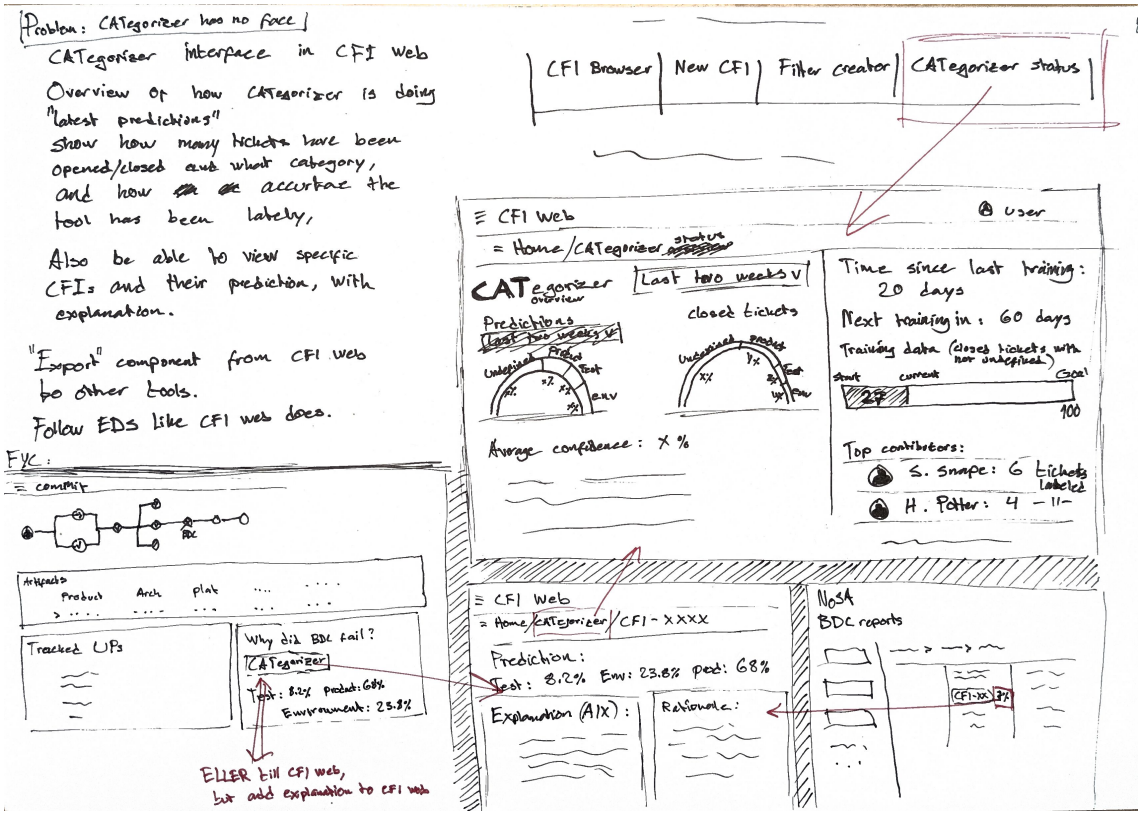


Figure 6.17: Brainstorming session ideas of dashboard after refinement

Other ideas focused on smaller, targeted features aimed at addressing specific issues identified during the pre-study. The overarching theme was improving communication through subtle visual signals within the interfaces. One of the key issues identified was that the tickets in CFI Web that did not have a category assigned or lacked one completely displayed no indication of being incomplete. To the user, such tickets appeared completely valid and finished. The proposed solution was therefore to add a yellow warning triangle next to the details field, along with a yellow underline beneath the problem type label, to clearly signal to the user that the ticket was incomplete.

The sketch also proposed a predicted problem type confirmation button, allowing users who spend little time in the interface to still make a meaningful contribution, or enabling Flow Guardians to quickly confirm a problem type when it aligns with

their experience.

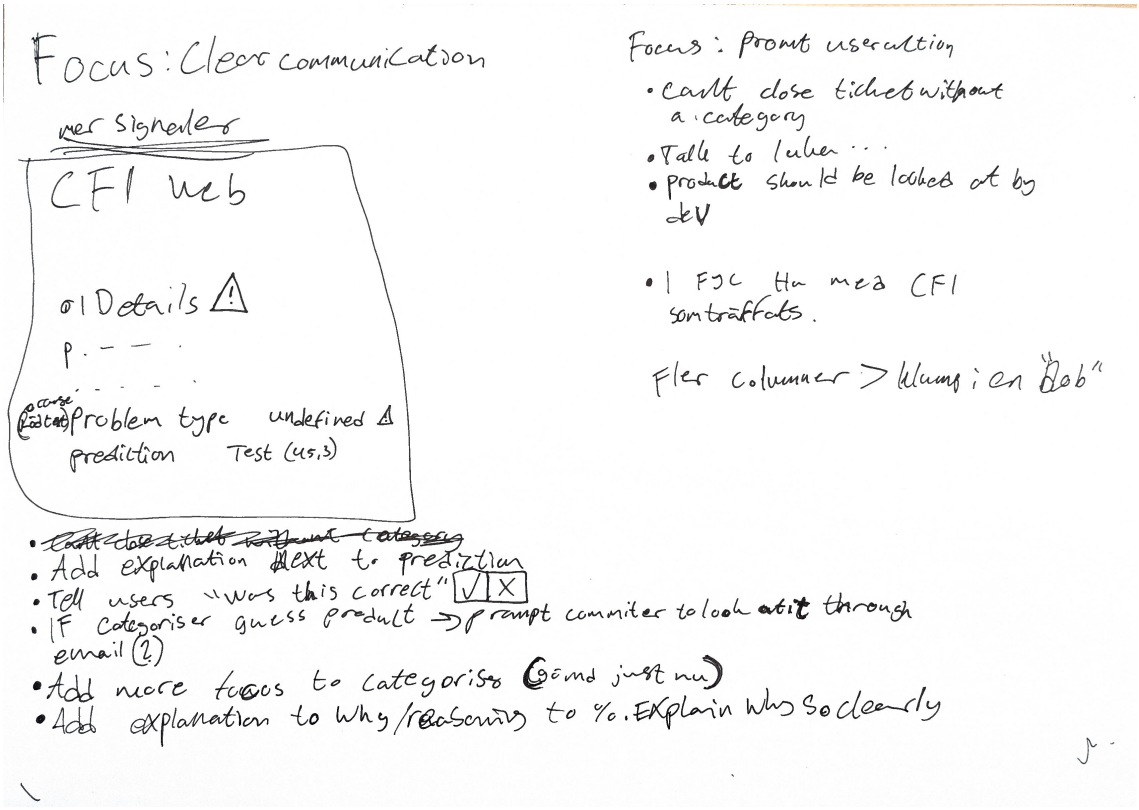


Figure 6.18: Brainstorming session ideas of smaller signals with the standout features being added error signaling in CFI WEB details and a confimation button of prediction

6.6 Implementation Cost for Prototype

The top-ranking features resulting from the improvement score calculation are presented below in Table 6.4. These represent the features identified as offering the highest impact relative to their implementation cost, and form the basis for the implementations carried out in Phase 2. Features that were otherwise viable but could not be included were excluded solely due to time constraints. See Appendix G for the calculation of all features.

Interface	Feature	Reqs	Guidelines	Improvement Score
FYC	Add CATegorizer prediction	ER1, ER3, ER4, UR1, AR1, AR4, IR1, ElR2	EG1, EG2, UG4, AG1, AG2, IG4, ElG4	137.14
CFI Web Details	Yellow warning triangle, yellow underline	ER1, ER3, UR2, AR1, IR1, IR5, ElR1, ElR4	EG1, UG2, UG4, AG1, IG2, IG4, ElG1, ElG3	78.00
CFI Web Details	Prediction confirmation button	IR3, ER4, AR3	EG1, EG3, UG2, UG3, UG4, AG2, AG4, IG2, IG3, IG4, IG6, ElG1, ElG2	65.33
CFI Web Dashboard	Percentage correct predictions	ER2, ElR1	EG2, EG3, UG1, AG1, AG3, IG3, IG5	38.67
CFI Web Dashboard	Average confidence of predictions	IR2, AR1, ElR1	EG2, UG1, IG1, IG6, ElG1, ElG4	38.67
CFI Web Dashboard	Distribution of latest predictions	ER1, UR3, IR2, ElR1	EG1, EG2, UG1, AG2, AG3, IG2, IG5, ElG4	19.33

Table 6.4: Top six features based on improvement score, including covered requirements and guidelines

6.7 Final Interfaces

The final interfaces consist of a dashboard highlighting the CATegorizer performance and the distribution of data used to train the AI model, as well as additional error signaling within the CFI Web interface (see Section 2.2.9 for more information). Figure 6.19 shows the final CATegorizer dashboard, which comprises four different visualizations of CFI ticket data (see Section 2.2.6 for more information on CFI tickets), along with a header.

Figure 6.19: Final version of the CATegorizer dashboard

The top-left card displays a donut chart showing the distribution of labeled versus unlabeled or undefined CFI tickets. The purpose of this card is to inform users of how many tickets remain unlabeled, conveying that a significant portion of the data is still incomplete. As shown in the figure, 47% of tickets are currently untagged, meaning that nearly half of all CFI ticket patterns are excluded from the AI model's training data. The donut chart was chosen because it is well suited to displaying a single whole distribution across a small number of categories. With only three categories, the chart remains uncluttered and easy to interpret.

To illustrate how the model makes predictions, a confusion matrix is included to show which problem types the model tends to confuse and where it performs most reliably. As shown in Figure 6.19, the most common problem type is Product, and the CATegorizer performs best when predicting this category. The confusion matrix also reveals that the prevalence of Product tickets in the dataset leads the model to default toward predicting Product even when the correct label is different. A complementary visualization displays the distribution of labeled problem types alongside the distribution of predicted types. This allows users to identify which problem types are over- or under-represented in the model's predictions relative to the actual composition of the training data. The visualization also serves as a record of historical label distributions, providing insight into the patterns present in the dataset over time.

The bottom graph illustrates the divergence between the model's confidence and its actual accuracy. Ideally, these two measures should align closely, as this indicates that the model's predictions are well-calibrated. The line chart displays historical data, with data points sampled at intervals of a few days, automatically adjusted to best fit the width of the graph. As shown, the model's average confidence over the last three months was 81%, while its actual accuracy fluctuated considerably ranging between 0% and 100% with an average of only 36%, which is a cause for concern.

All information displayed on the dashboard can be filtered based on three filters. The first is a time period selector located in the top right, which allows users to define a time window stretching from the current date backwards. The second filter allows users to narrow the data by impediment owner - the project responsible for the tickets enabling them to focus on their own project, where they are most likely to be familiar with the CFI tickets. The third filter controls whether Dekrabot-generated tickets are included in the data (see Section 5.1.4 for more information about Dekrabot). This filter was added because Dekrabot submits CFI tickets at a high volume without the same level of follow-up as human-written tickets, which skews the data heavily toward undefined labels a pattern that is not representative of human-written tickets, which are the primary focus of this thesis. As these tickets are nonetheless part of the dataset, users are given the option to include or exclude them as needed.

The header acts as a summary of the dashboard as a whole, briefly describing what the dashboard presents and prompting users to label CFI tickets in Jira (see Section 2.2.7) or CFI Web (see Section 2.2.9). On the opposite side, the header also displays the current status of the CATegorizer. This status is shown as *Poor* if the accuracy over the last 30 days falls below 50%, *Degraded* if it falls between 50% and 85%, and *Healthy* if it exceeds 85%. The purpose of this indicator is to give users an immediate sense of model performance without requiring them to interpret the charts in detail. The average accuracy and confidence of the model over the last 30 days are also displayed for quick reference.

In addition to the visualized data, users can hover over any element on the dashboard to reveal a tool tip with supplementary information. This allows the dashboard to remain uncluttered for users who want a high-level overview, while still providing deeper detail for those who wish to explore further.

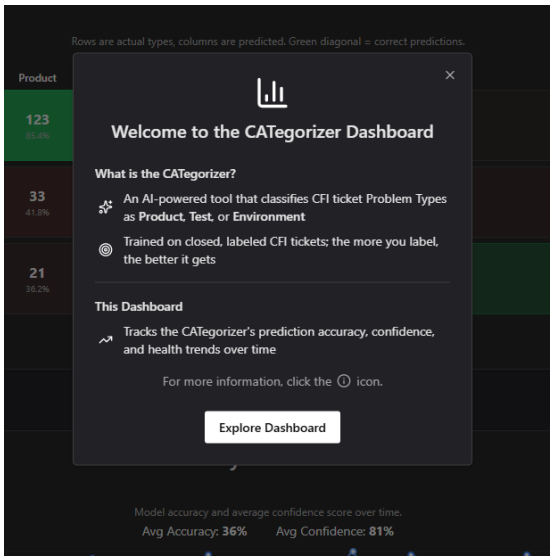


Figure 6.20: Informational card appears in the middle of the screen when a user first visits the dashboard

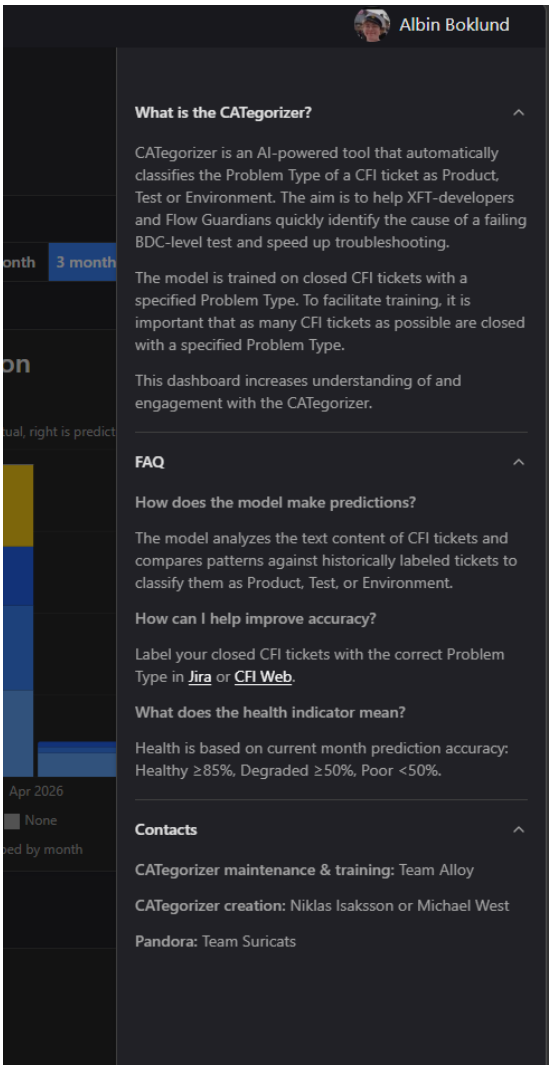


Figure 6.21: Informational side panel with expandable sections

When a user first visits the dashboard a welcome card shown in Figure 6.20 is included as an overlay to introduce the CATegorizer and explain the purpose of the interface. Additional information is accessible via an information panel shown in Figure 6.21, which slides out from the right of the dashboard when the information button on the top right is pressed and includes answers to expected user questions as well as contact information for future support.

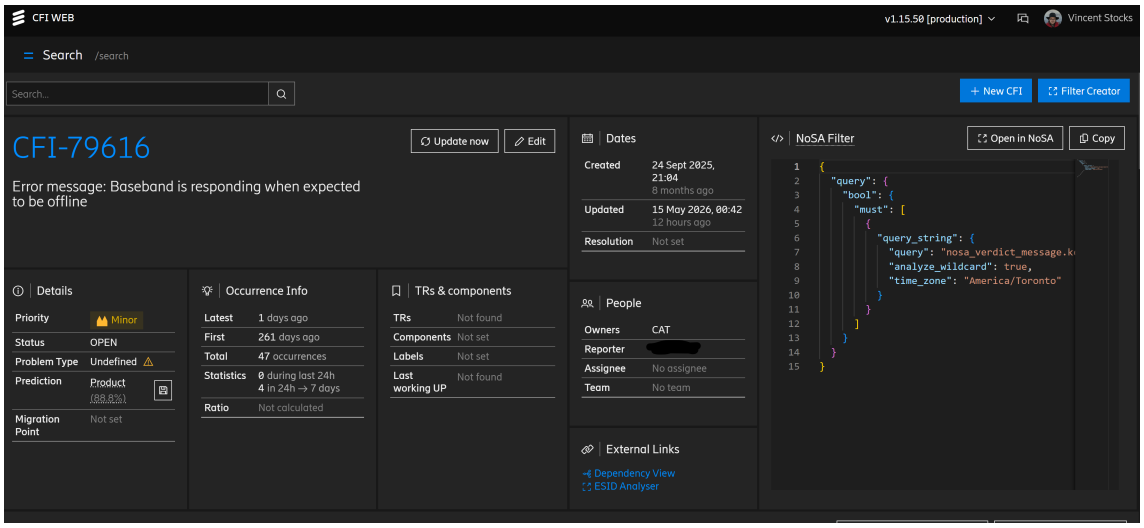


Figure 6.22: CFI Web with additional features in the Details section

Figure 6.22 shows the complete CFI Web interface for ticket *CFI-79616*. The interface remains unchanged except for the *Details* section.

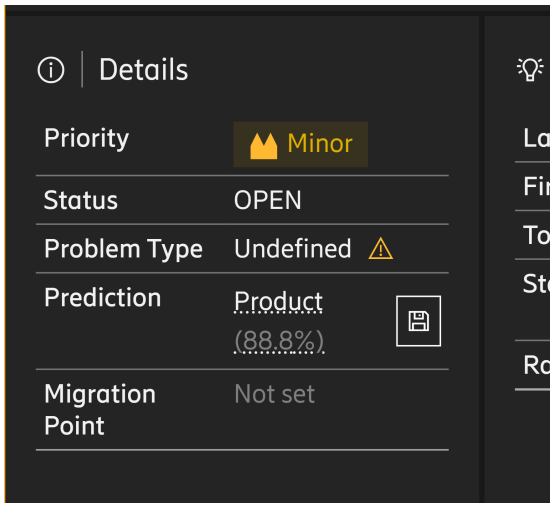


Figure 6.23: Final solution to CFI Web details section

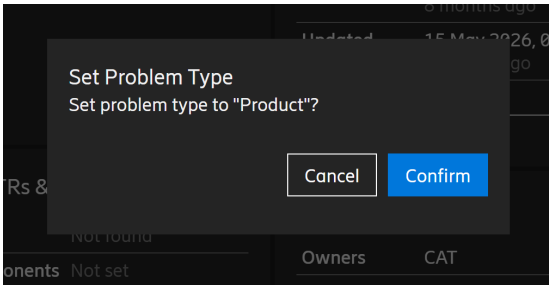


Figure 6.24: Confirmation section after clicking on the save button next to prediction

6.8 Live-testing Survey Results

The final survey sent out to the COIN organization yielded 6 responses. 5 of the respondents reported having a software developer role, and 1 respondent reported being a Technical Coordinator. Their time spent exploring the dashboard varied, ranging from 2 to 5 minutes (3 respondents), 5 to 10 minutes (2 respondents) and over 10 minutes (1 respondent).

6.8.1 Live-testing Likert Results

Responses to the Likert-section of the survey can be seen in Figure 6.25. The results are generally positive, with a large majority of respondents reporting that they Agree or Strongly Agree with the statement. The general perception was that it is easy to navigate and interpret the content of the dashboard. The dashboard was also unanimously viewed as successful in clarifying the performance of the CATegorizer.

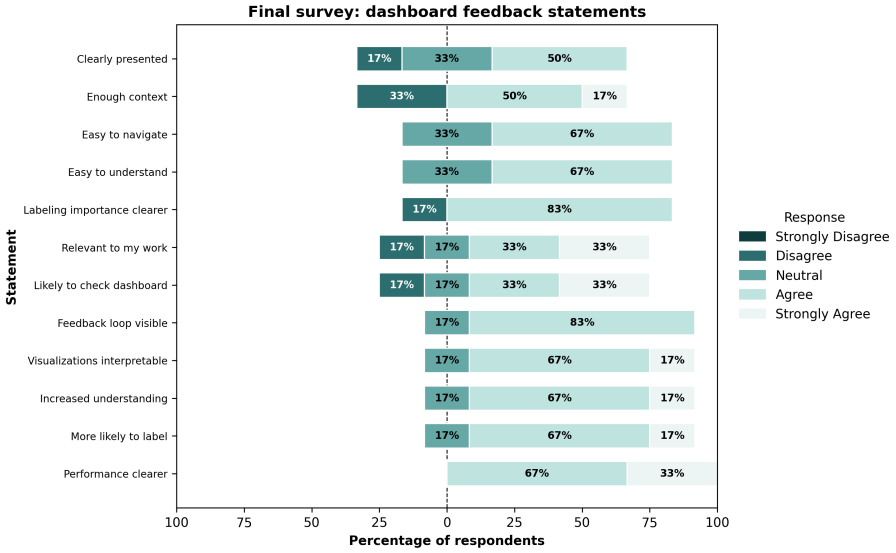


Figure 6.25: The distribution of answers to the likert-statements in the final survey (n=6)

A second theme was that the dashboard appeared to improve users' understanding of the relationship between their labeling efforts and the CATegorizer. Most respondents agreed that the dashboard increased their understanding of the system, made the feedback loop more visible and communicated the importance of labeling CFI tickets.

The most mixed responses concerned whether the dashboard provided sufficient context for understanding the presented data and its relevance to the respondents' daily work. While these statements still received mostly positive responses, they also accounted for the majority of neutral and negative answers, indicating that some users may require additional contextual information or project-specific views.

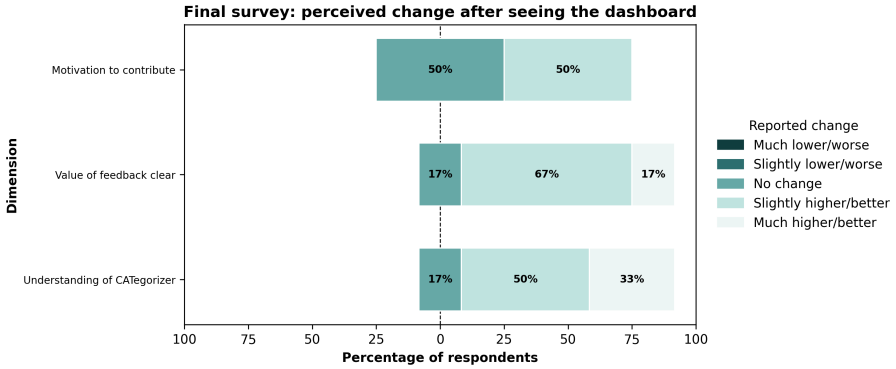


Figure 6.26: The distribution of answers to the perceived change section of the final survey (n=6)

As shown in Figure 6.26, respondents also reported positive perceived changes following exposure to the dashboard. The strongest effect concerned the perceived value of feedback, where nearly all respondents reported either a slightly or much better understanding of its value. Similarly, most respondents reported an increased understanding of the CATegorizer.

The effect on motivation to contribute was more modest. While half of the respondents reported slightly increased motivation, the remaining respondents reported no change. No respondent reported a decrease in motivation, suggesting that the dashboard had either a positive or neutral effect on willingness to contribute.

6.8.2 Free-text Responses by Question

In the free-text part of the survey only three of the respondents actually wrote down their thoughts, with the other three leaving the fields blank. When asked about what part of the dashboard best supported their understanding of the CATegorizer, two answers pointed out the Impediment Owner filter as very beneficial, saying *"CI Flow impediments"* and *"Flows separations and view"*. A third answer said *"Confusion Matrix, showing how predictions and actual problem types relate"*. The second question, asking whether anything reduced the dashboard's usefulness, all answers pointed to more explanation regarding certain areas of the interface, for example an explanation as to what "dekrabot" is. When asked if they believe this dashboard could increase user engagement with and feedback to the CATegorizer, two respondents said they believed so but also that the increased understanding in general would be the biggest benefit. The third answer said that they were not sure. For the final question, asking about potential improvements of the CATegorizer were it developed further, one answer focused on increased filtering options, writing *"Possibility to filter on different targets, G3, G4, G5, COTS. AI-button where we can ask/search for something trough all CATegorizer-page"*. The second suggested adding more context and information, with the third suggesting further flow visualizations, stating *"flow visualization, maybe some way to go even deeper or some links to go directly and check status of the failed job, it is at least not easy to find it if it is there"*.

6.9 Post Implementation Stakeholder Interviews

The following section presents the thoughts and answers from the interviews conducted with the stakeholders after the implementation of the new features went live. As explained in Section 5.4, four people were interviewed, each holding a different role in the CI workflow.

6.9.1 Cognitive Walkthrough Results

During the cognitive walkthrough described in Section 5.3, the XFT developer noted on the first task that the overall goal of the dashboard felt somewhat unclear, and that they would typically only be interested in data from their own organization rather than a global view. The product owner acknowledged that the confusion matrix required some "cognitive energy" to interpret, but considered the dashboard's overall intent to be clearly communicated. The Flow Guardian similarly found the dashboard clear, but noted that they would only visit it with a specific goal already in mind, such as reviewing the performance of the CATegorizer.

During the second task, all interviewees immediately looked to the *Current Status* section in the top right corner and collectively interpreted the model's performance as relatively poor at the time of testing. Several interviewees found the distinction between *Accuracy* and *Confidence* insufficiently explained and suggested that clearer descriptions would be beneficial. However, none of the interviewees could propose a concrete solution that would improve clarity without adding visual clutter to the dashboard.

During the third task, the XFT developer and interface developer both looked first at the bar charts, guided by the *Problem Type Distribution* label. The interface developer attributed this to familiarity, stating "The bar chart because it is used the most at Ericsson to display those kinds of things". The product owner and Flow Guardian instead applauded the confusion matrix, describing it as "the only one that clearly shows the accuracy across different problem types". The Flow Guardian elaborated on this, stating that "It is very important for me that the CATegorizer can distinguish between environment and test or product, since I do not care about environment tickets. That is the lab's job to fix."

The fourth task revealed mixed responses regarding how easily users could find information about contributing to the CATegorizer. The interface developer was uncertain whether this was sufficiently prominent and suggested adding a banner, while acknowledging that this risked making the interface feel cluttered. The remaining three interviewees agreed that the dashboard subtitle communicated the contribution pathway clearly, but noted that the information icon was not immediately visible and required some exploration of the dashboard before being discovered.

The fifth task, which asked participants to filter the dashboard by Impediment Owner and Time Period, was completed with ease by all four interviewees. The primary concern, raised by the Flow Guardian and XFT developer, was that the term *Impediment Owner* was unfamiliar in its current context, as it is used only as

a legacy label within Ericsson. Both suggested renaming it to *Product Area* as a more intuitive alternative.

The final task asked interviewees whether they believed the dashboard would support the process of closing tickets. All four agreed that the dashboard would not directly assist during the ticket-closing workflow itself, but that it could serve as a motivational tool encouraging users to label tickets. The Flow Guardian highlighted the confusion matrix as particularly useful for identifying outliers, while the XFT developer noted that the dashboard would increase their inclination to label tickets more consistently.

6.9.2 Likert Scale Answers

When asked how the CATegorizer factors into their daily work, three of the four interviewees indicated they interact with it at least weekly as shown in Figure 6.27, though their use cases varied considerably. The Flow Guardian noted that while the tool is visible daily, it does not consistently inform decision-making. The product owner of the CFI-ticket system (and supervisor of this thesis project), described the CATegorizer as having a significant impact on their work. The XFT developer expressed a more targeted interest, primarily wanting to know whether they had contributed to any issues during a major release. The sole exception was the interface developer, who selected *Never*, explaining that, as the person responsible for developing the software in which the CATegorizer operates, its predictions has no impact on their own work.

Regarding the opening and closing of CFI tickets, the patterns again reflected differences in role and responsibility. The XFT developer reported opening tickets on a monthly basis, typically upon identifying a problem worth investigating, though acknowledged that closure is largely reactive prompted by automated warning emails rather than active follow-up, noting that keeping track of every ticket is not feasible. The Flow Guardian, by contrast, described ticket creation as part of daily operations, with closure occurring weekly or monthly during periodic reviews of incoming emails. The remaining respondents selected *Never* for both actions, consistent with their roles not involving CFI ticket handling.

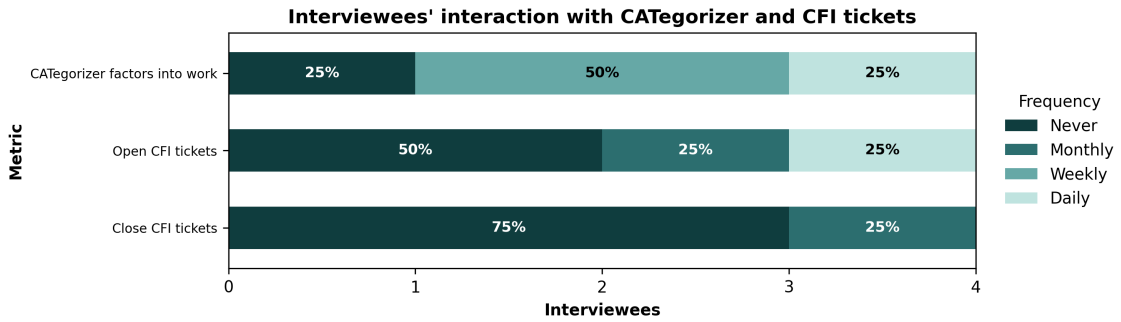


Figure 6.27: Stakeholder answers from interviews on the topic of CFI tickets usage patterns

When asked about their opinions regarding how the dashboard presents data, its usability and potential to encourage users to label more tickets, stakeholders responded mostly positively, as shown in Figure 6.28. The most notable responses are the *Strongly Disagree* ratings across the statements. It is noteworthy that the *Strongly Disagree* responses to the questions *Relevant to my work* and *Likely to check dashboard* were provided by the same respondent, as was the *Strongly Disagree* response for *Feedback loop*.

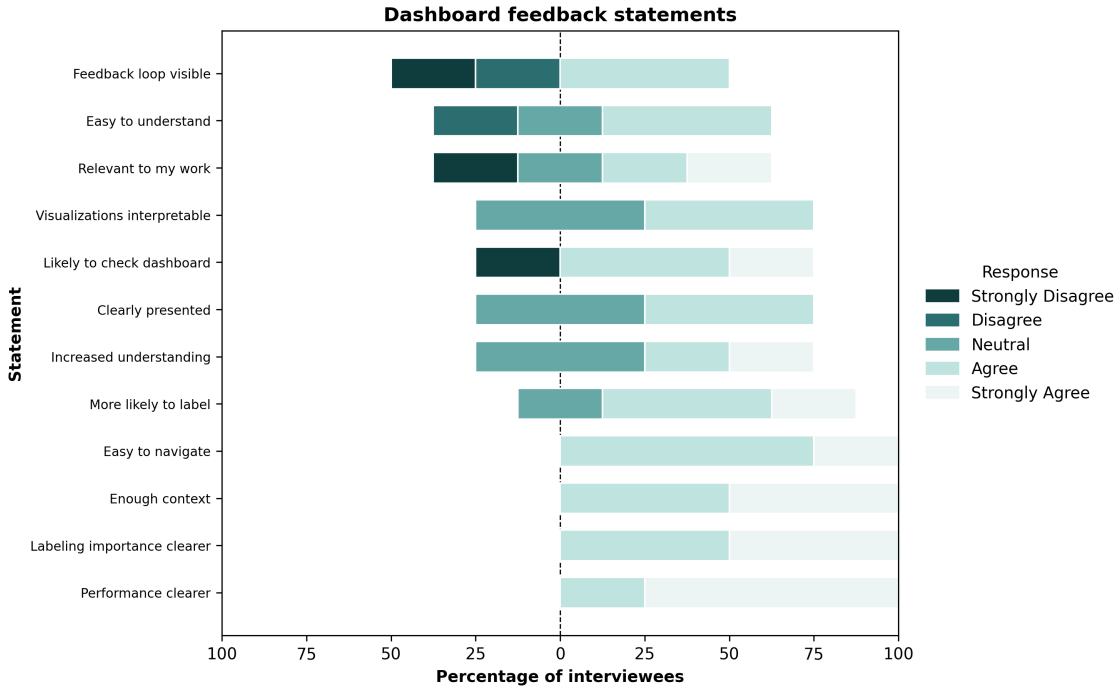


Figure 6.28: Stakeholder opinions of the dashboard

When asked about the primary goal of the dashboard, which is to motivate users to label more tickets by conveying the value of their contributions and explaining the purpose of the CATegorizer, stakeholders responded positively. The majority agreed that the dashboard succeeds in motivating users to label tickets and provides a clearer understanding of the CATegorizers function and its reliance on labeled data.

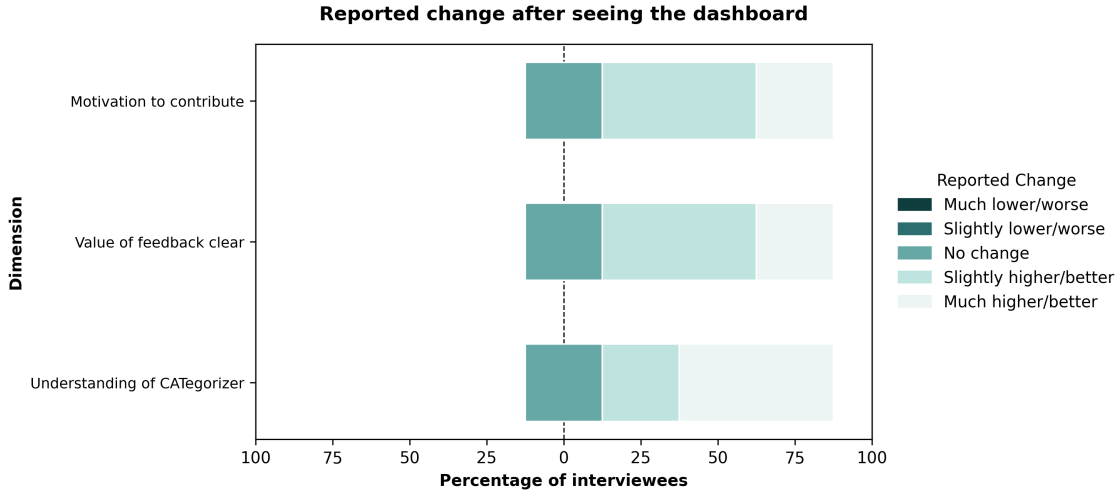


Figure 6.29: Perceived impact of the dashboard

The responses gathered during the *CFI Web* cognitive walkthrough were predominantly positive. Stakeholders agreed that reducing the required interactions to set a Problem Type from four clicks to two represented a meaningful usability improvement. The perception of the warning triangle was only positive, and many saw it as the best addition to the interface. These sentiments were reflected in comments such as "...the warning triangle is brilliant..." and "...the button will make CFI Web better...". The main piece of negative feedback was that the button was considered too discreet and, according to one interviewee, was not immediately noticeable without being pointed out. Another suggestion was to show the warning triangle when no problem type is selected at all, not only when Undefined is selected.

6.10 Post Implementation Usage Analysis

The following section displays the usage analysis performed after the two-week live testing period, compared to the baseline measured in the pre-study, as described in Section 6.1. As Dekrabot was not considered a reporter relevant to this specific thesis, as described in Section 6.1.2, the evaluation includes only datasets excluding Dekrabot. When comparing all reporters, the results show a larger volume of tickets post-implementation compared to the pre-study period, as shown in Figure 6.30. The distribution of tickets, however, shows no major difference between the two periods.

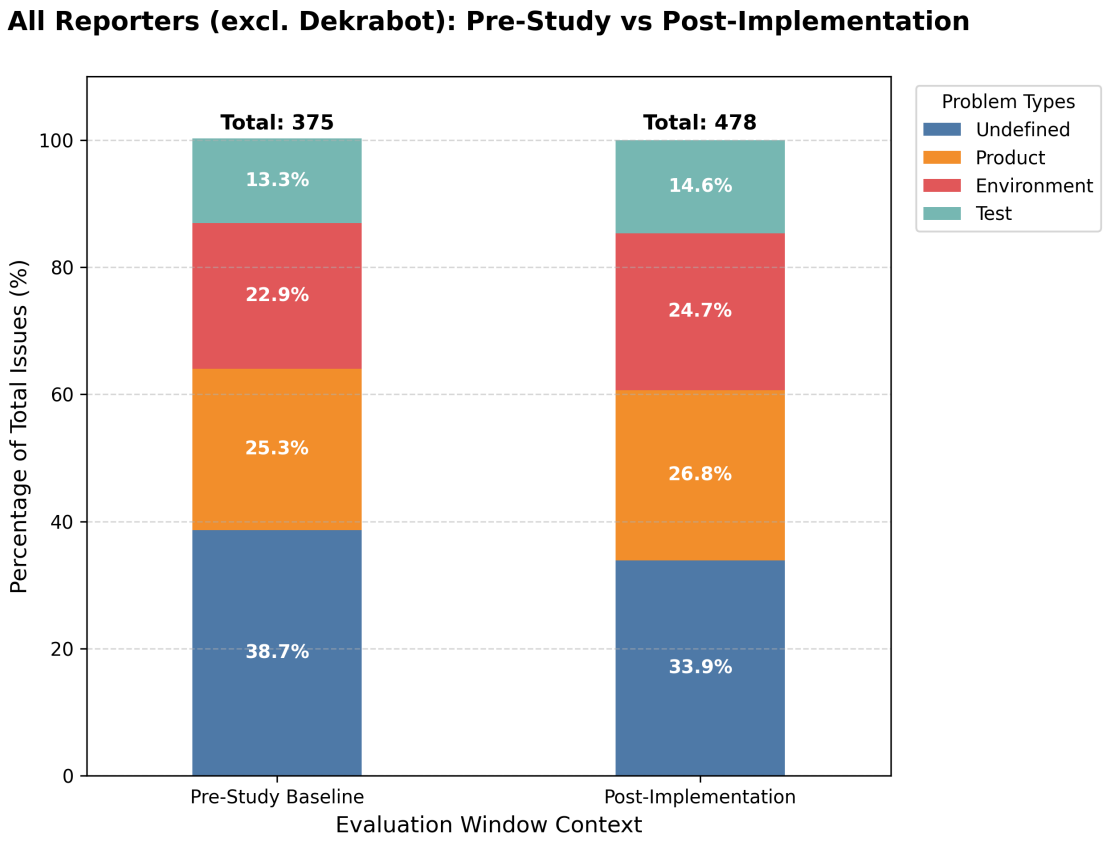


Figure 6.30: Comparison of baseline and post-implementation usage analysis when excluding Dekrabot

When comparing only the distributions of tickets created within the COIN organization, a visible difference is observed before and after the implementation of the new interfaces as illustrated in Figure 6.31. Both periods show a nearly identical total number of tickets, with 16 tickets created during the 2 week period of the pre-study baseline and 17 during the 2 week post-implementation period. The distributions shown in the figure indicate that the number of tickets labeled as *Undefined* decreased from 6 to 2, representing a proportional decrease of 68.8%. During the same period, the number of tickets labeled as *Product* instead increased from 8 to 14, representing a proportional increase of 75%.

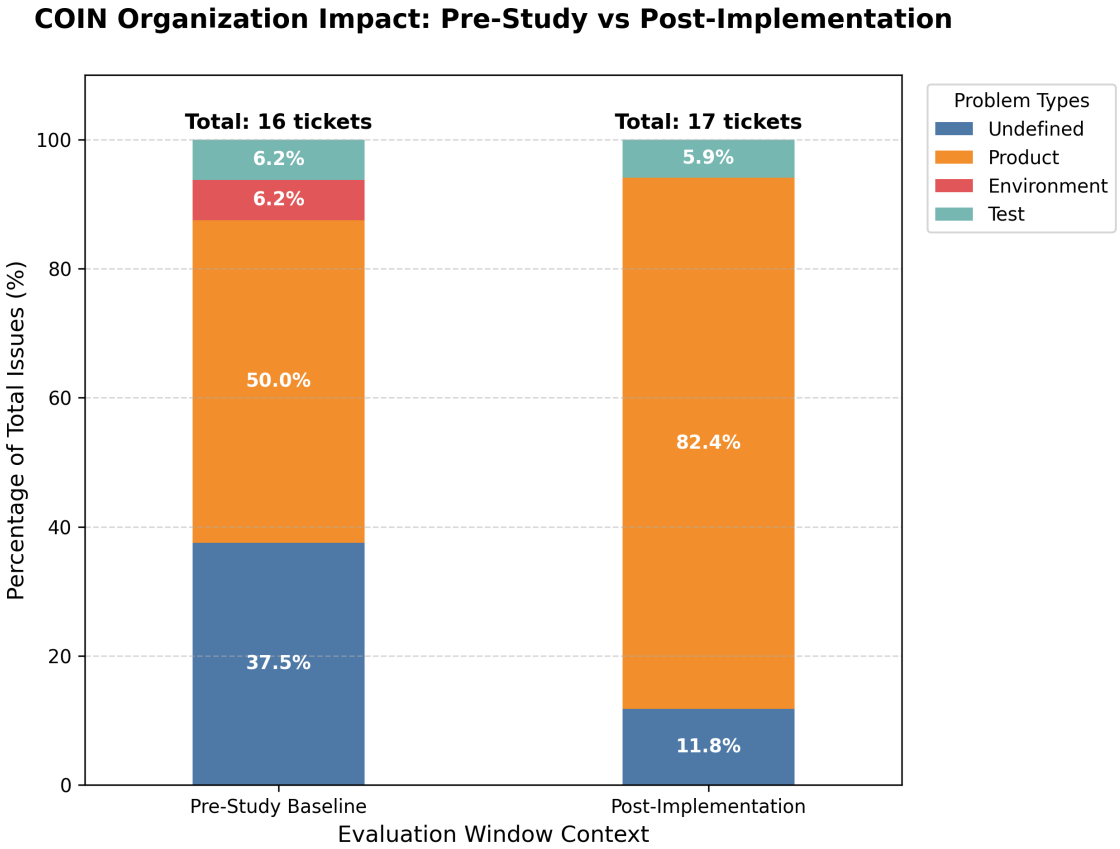


Figure 6.31: Distribution of problem types in the COIN organization before and after interface changes. No tickets were labeled as Environment during the two week period post-implementation

When projecting the two-week results onto the same time periods used for the pre-study baseline namely 2, 4, 26, and 52-weeks the differences in distribution become more pronounced. The projected distributions are shown in Figure 6.32, with dashed bars representing the projections. If tickets in COIN were to follow the distribution observed during the two-week period, there would be 470 tickets labeled as *Product*, 34 labeled as *Test*, and 67 labeled as *Undefined*. This would represent an approximate decrease of 80% in *Undefined* tickets, while *Product* tickets would increase from 150 to 470, corresponding to an increase of approximately 213%.

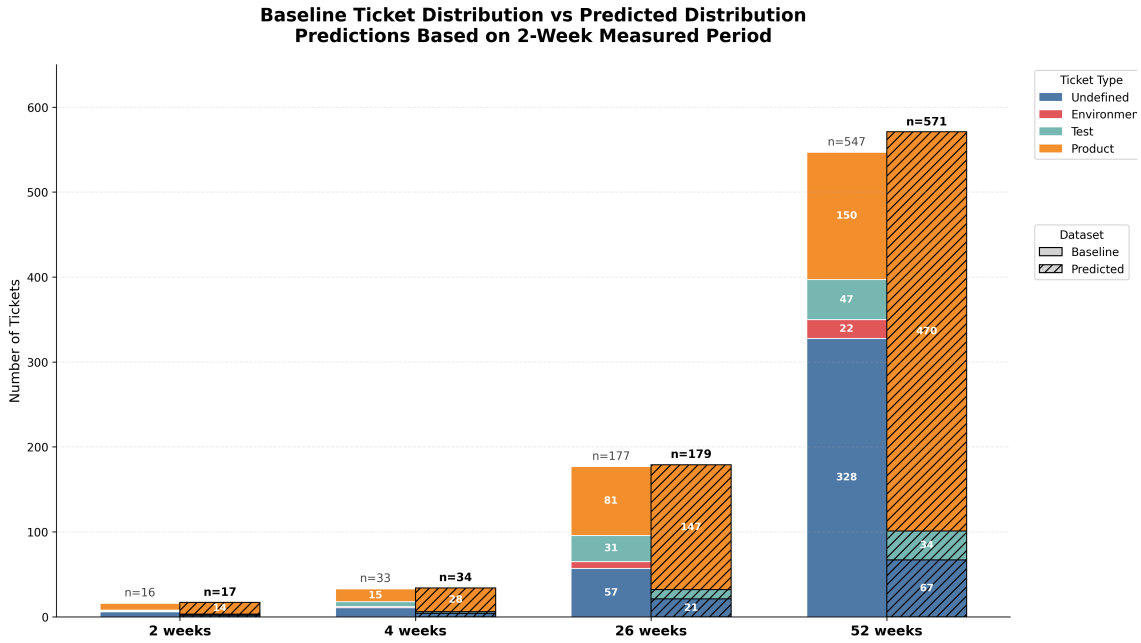


Figure 6.32: Predicted ticket type distribution in COIN based on the distribution measured during the two-week live test period, compared to the pre-study baseline

Regarding the accuracy of the CATegorizer over the same time period, it correctly predicted 46 out of 139 tickets, resulting in an accuracy of 33.09% when the problem type *Undefined* is included, as shown in Table 6.5. When tickets labeled as *Undefined* are excluded so that only the types that the CATegorizer can predict are measured, the CATegorizer correctly predicted 46 out of 92 tickets, resulting in an accuracy of exactly 50%.

Metric	**Total Results**	**Excluding Undefined**
Total Count	139	92
Correct	46	46
Incorrect	93	46
Accuracy	**33.09%**	**50.0%**

Table 6.5: Accuracy Metrics Results

When comparing the predictive accuracy of the CATegorizer before and after the implementation of the new interfaces, a decrease in accuracy is observed both with and without the *Undefined* type, as shown in Table 6.6.

Table 6.6: Comparison of Accuracy Metrics: Pre-Study Baseline vs. Post-Implementation

	Pre-Study Baseline		**Post-Implementation**	
Metric	**Total Results**	**Excluding Undefined**	**Total Results**	**Excluding Undefined**
Total Count	264	176	139	92
Correct	116	116	46	46
Incorrect	148	60	93	46
Accuracy	**43.94%**	**65.91%**	**33.09%**	**50.00%**

7

Discussion

This chapter contains the discussion regarding the thesis process, results, experiences and findings. It also contains important reasoning regarding ethical and social concerns, especially regarding AI and the social, economical and environmental issues this new technology could introduce.

7.1 Answering the Research Question

The research question, as specified in Section 1.2, was defined as "How can the user interface of an AI-assisted error categorization tool be designed in order to encourage users to provide feedback to the AI model, without negatively impacting the users' perceived efficiency?". To answer this, we must consider multiple angles of evaluation.

First of all, the pre-study revealed that most users either did not possess or had very limited knowledge of and motivation surrounding the CATegorizer. We believe this to have been one of the main causes of the low engagement with the model. This is one reason why the dashboard was designed, to increase the knowledge of the tool among users. The evaluation of the dashboard suggested a positive impact on the users' knowledge of the CATegorizer, making it clear to them that their work had a direct impact on the model's performance and making them more likely to provide feedback. A broader interpretation of this finding could be that any AI-tool that relies on a healthy user feedback loop will benefit from increased understanding in the user base and that this *can* be achieved through UI design.

Another important aspect to consider is found in the users' motivation. As mentioned above, the pre-study revealed the CATegorizer-related motivation to be lacking among users in general. This is in part because of the low levels of knowledge surrounding the system, but could also be attributed to a "why should I care?" attitude. This brings us to the next reason for designing the dashboard; attempting to create a sense of shared ownership. If the users feel that the tool is something that they "own", something that is directly affected by their work, then that could lead to an increase in motivation and engagement. This can also be generalized into a broader interpretation where by creating a sense of shared ownership surrounding an AI model users may become more motivated to engage with it.

The second part of the RQ, "...without negatively impacting the users' perceived

efficiency?", is important to consider thoroughly as well. A reason that the aforementioned solutions work could be that they are based on an implementation placed just outside the actual CI workflow. This in turn limits the impact that the extra interface can have on the users' perceived efficiency in their everyday tasks. However, changes were also made to the CFI Web interface, as explained in Section 6.7 and further discussed in Section 7.6.2. These changes were made carefully as to not negatively impact the perceived efficiency of the system, instead aiming to *increase* efficiency while *also* increasing the ease for users to provide feedback. The evaluation suggested that the changes supported this goal, with many users reporting a higher likelihood of providing feedback while also experiencing the system as more effective. To generalize this finding, we theorize that feedback mechanisms within AI-assisted tools should be designed to fit into users' existing workflows rather than as additional, separate tasks. If feedback can be provided at a stage where the user is already engaging with the relevant information, this helps mitigate the negative impact on perceived efficiency.

Thus, our answer to the RQ is not a single feature or interface element, but instead a combination of awareness-building outside the workflow and efficient feedback opportunities inside the workflow itself.

7.2 Discussion of Process

At the outset of the project, the company stakeholders and supervisors had left the project relatively open in terms of how it should be carried out and how success should be measured. Their goal was to increase the performance and longevity of the CATegorizer model, and little weight was put on *how* this was to be achieved in the end. This open approach was valuable to us as we could shape the thesis and project to suit the needs of both Chalmers and Ericsson. However it also resulted in the pre-study (Phase 1, see Section 5.1) becoming much larger and taking more time than anticipated and planned for. Still, the very thorough pre-study not only laid a solid foundation for the rest of the project, it also produced data that in itself turned out to be useful to the company. The data highlighted usage patterns within the different systems of the CI flow and some areas did show minor discrepancies between what different organizations expected from each other, suggesting a need for small alignments and synchronization across organizations. Overall, we consider the pre-study to be one of the strongest aspects of this thesis and the use of multiple different methods for both data gathering and analysis to have been appropriate for the exploratory nature of the work that was positive toward the results.

During Phase 2 the focus on ideation and implementation based on the results from the pre-study allowed for an efficient and smooth process. The help and guidance from the operations team behind Pandora proved to be very valuable during the implementation part of the work. However, there were some drawbacks and obstacles that are worth mentioning. For example, one of the features calculated to be the most impactful could not be realized due to the constraints and ways of working imposed by the operations team. An alternative could have been to implement our solutions in a testing environment outside the actual CI workflow. That way all the

features could have been implemented and tested in a controlled setting. However, the decision was made to implement the solution in the production environment instead as real usage data was deemed to be much more valuable than controlled test data.

Phase 3 deviated significantly from the initial project plan, a shift that occurred as a direct consequence of the pivot to Pandora (further discussed in Section 7.5). Instead of the originally planned multi-week live testing phase and subsequent quantitative data gathering, the evaluation relied on a combination of live testing, interviews and a survey. Consequently, the resulting dataset was more qualitative than initially anticipated. However, this shift in focus provided valuable contextual insights. While a substantial decrease in the volume of tickets labeled as "Undefined" would have rendered the quantitative impact of this thesis more visible, the perspectives collected through the qualitative methods showed a positive attitude toward the changes and proved sufficient for addressing the research questions. This insight would not have been possible to collect from quantitative data as Pandora did not have a persistent user base. While it still would have been optimal to have a multi-week live testing phase, due to the time constraints of the project this was not feasible. However, since the change in approach produced usable data that supported the analysis of how well the research questions had been addressed, we would argue that this was the correct decision.

Although a prolonged live testing phase remained the ideal scenario, it was rendered unfeasible by the project's time constraints. Nevertheless, because the modified approach produced viable data that robustly supported the analysis, this pivot represents a pragmatic and effective methodological adjustment.

7.3 Discussion of Methods

A wide selection of methods were used in this thesis, especially in the pre-study. This was deemed necessary due to the exploratory nature of the project, and worked well to elucidate some important theorized problem areas while clarifying others as being less influential. Still, some methods were more impactful and others less so. For example, the methods NASA-TLX and SAM were used during the pre-study and produced results that had a relatively limited impact on the outcome of the project. However, this could be due to the findings not revealing any strong or unexpected patterns. Had the methods uncovered more significant patterns among the respondents they would likely have been more impactful. Therefore it was still a justified choice to carry out these methods as part of the study. The most impactful method used was the survey during the pre-study. This resulted in a large dataset to analyze, and the findings substantially influenced the final designs and implementations.

7.4 The Initial Status of the CATegorizer

The results from the baseline usage analysis confirmed that the CATegorizer is not performing at an optimal level, showing an accuracy of 43.94% (increased to 65.91% when excluding Undefined tickets) as seen in Table 6.1. The data regarding ticket labeling indicates that in general, 30%-40% of labeled tickets are labeled as Undefined, as shown in Figures 6.1 and 6.2. This means that when considering the baseline, there is a large portion of tickets that the CATegorizer cannot train on. This further confirms the explanation of the poor pre-thesis system status as given by the main stakeholders.

As for the pre-study survey, this provided usable data covering many aspects of the experiences and sentiments of developers working in the CI flow. The tool usage frequency proved to be very influential on the rest of the project, guiding the efforts toward designing the most impactful yet least disruptive solution. The survey also revealed the general lack of knowledge among developers regarding the CATegorizer, which would become one of the main focal points during the ideation and implementation process.

7.5 The Pivot to Pandora

The revelation during the implementation phase of the thesis work that a new system, Pandora, had been in active development by another operations team at the company did cause some issues. Pandora was a new interface designed to combine many of the standalone systems that had been examined during the pre-study, systems that we had designed our new features around. Information regarding this new interface had not been withheld intentionally, instead it was a simple case of miscommunication between the different stakeholders along with shifting timelines of other projects and launch windows.

Rather than redesigning the proposed features entirely, they were instead adapted for implementation in Pandora directly. The main reason this shift was possible was that since Pandora was an amalgamation of the previously existing systems, many of its features and functions were very similar to the old systems. This along with the support of the operations team allowed for a relatively painless transition and adaptation. Still, this was a contributing factor to some of the features not being implemented. For example, the feature designed for the Follow Your Commit interface could not be realized in time despite being calculated as the most impactful of the proposed features. Whether this feature could have been implemented were it not for this pivot is not clear given that the operational team of Pandora could have been against it. Furthermore, implementing our solutions directly into the new interface could also be seen as a very good thing for their longevity. The alternative, implementing the solutions into systems that will be discontinued in the near future, could have lead to the solutions quickly becoming outdated or obsolete as well. From this perspective, the pivot increased the likelihood that the implemented features would remain relevant beyond the thesis period.

The main drawback of the pivot was the lack of users in Pandora. As it is a brand new system that has not officially been launched, the comparatively small user base means that we could not expect the amount of usage data that had originally been estimated. This is the reason for the change in approach to using both a survey, interviews and usage analytics regarding CFI tickets. This trade-off produced sufficient data to support an assessment of the solution.

7.6 Discussing the Final Implementation

This section discusses the different aspects of the final implementations, including both the dashboard and the smaller changes made to the CFI Web interface. It will also discuss how well the implementations align with the defined requirements and guidelines (henceforth referred to as RGs).

7.6.1 Dashboard

The final version of the dashboard received generally positive feedback from the respondents in both the final survey and final interviews, which could be interpreted as an indication that some of the RGs had been met. These RGs are mainly those that can be measured by the users' perception and experience working in the system, since that is where the data comes from. For example, the implementation could be seen to align with both UR3 and UG3 (see Table 6.2) based on the feedback provided by the users, and is especially supported by the survey response quotes regarding the filtering functionality as seen in Section 6.8.2. Both quotes suggest that the "filter Impediment Owner" functionality allows users to view the dashboard from a perspective that is most aligned to their personal use case. However, one answer suggests the implementation could be improved even more through further expansion of the filtering functionality, which would theoretically make the dashboard even more aligned with UR3 and UG3. Still, we argue that the final implementation fulfills UR3 and UG3.

In contrast, the RGs UR4 and UG4 could be viewed as less covered by the final implementation (see Table 6.2). They focus on the prototype supporting users who want to verify predictions against other sources, and a design that makes the prototype feel part of the troubleshooting process, not a separate step. The dashboard currently provides very little in terms of confirmation against other sources, although it could also be argued that the dashboard should not provide this functionality since it is not focused on individual predictions. Regarding the prototype feeling like a natural part of the troubleshooting process, the dashboard only partially satisfies this aspect. The dashboard is a separate interface from the troubleshooting process; it is an overview, not part of the actual troubleshooting flow. However, it could be argued that the dashboard being hosted in the Pandora system makes at least a secondary part of the troubleshooting process even when not directly involved.

There are more examples of the dashboard aligning and not aligning with the RGs, but they are mainly minor discrepancies and do not warrant further detailed discussion. The key takeaway from the post-implementation testing and data gathering is

that the dashboard aligns with most of the RGs, and never directly contradicts any of them.

From a UX perspective, the dashboard has multiple strengths. One of them is the way that it presents complex model-related information through multiple different visualizations, another is the way it leverages filtering to support role- and assignment-specific interpretation. It also provides easy-to-understand onboarding and contextual help for users with low prior knowledge and/or experience. The aspects align with the results from the pre-study suggesting that many users lacked sufficient knowledge about the CATegorizer and were in need of clear explanations regarding the system and its value. Still, some parts require further refinement. In particular, the distinction between accuracy and confidence should be addressed more clearly, as this was something that many users indicated to be problematic for their understanding of the system. Overall, this suggests that the dashboard succeeds in making the CATegorizer and its feedback loop more visible, but that further work is needed to make all model-related visualizations and concepts more immediately understandable.

7.6.2 CFI Web

Compared to the scope of the dashboard, the features implemented in CFI Web were few as the changes only included a button and a waring triangle. Still, during the interviews most interviewees were very positive to the changes made. By adding the confirmation button next to the prediction and reducing the amount of clicks required to set a Problem Type from four to two, many respondents stated that they believed this would lead to a positive change in the amount of CFI tickets being labeled. The addition of the button could also be said to align the implementation with RGs focused on making user feedback easier (such as ER3, EG3, ER4, UR2, UG2 and AR3 to name a few, for more see Table 6.2). Even better was the addition of the warning triangle, which most respondents really supported and said would lead to the most engagement. Regarding the button itself being a bit obscure in the interface, this is something that can be further refined with future work. Also, showing the warning triangle when “None” is selected was an incredibly useful idea and is something that will be added as soon as possible. It should probably have been that way from the start, but was overlooked by us. Overall, the positive perception among users regarding the changes suggest that these small changes will have a positive impact on the system and feedback loop over time.

Viewed together, the CFI Web changes and the dashboard address different aspects of the same problem. The dashboard primarily focuses on awareness, understanding and motivation. It achieves this by making the performance and feedback loop visible. In contrast, the CFI Web changes support direct action by simplifying the ticket labeling and feedback process for the user. Furthermore, the inclusion of a visual warning triangle effectively highlights exactly when and where this feedback is required. The combination of these are important as the results suggest increasing feedback requires both making it easier to provide while explaining why it is needed.

7.7 Final Usage Analysis, Interviews and Survey

The evaluation of the performance was measured using quantitative data through usage analysis and Likert scale responses from the interviews and surveys, as well qualitative free-text responses from the interviews and surveys. The results of the usage analysis are shown in Section 6.10 and interviews in Section 6.9.

7.7.1 Usage analysis

Figures 6.30 and 6.31 together reveal an important contrast. At the organizational level, ticket label distributions across all product areas (excluding dekrabot) show no meaningful shift between baseline and post-implementation periods, suggesting the CFI Web feature and CATegorizer dashboard had insignificant impact. With this result taken into isolation, this null result would suggest that the changes in the interfaces was ineffective towards encouraging users to label more tickets. Several factors could possibly account for this null result. First, the pivot to Pandora (Section 7.5) significantly narrowed the interventions reach. Pandora lacked the established user base and workflow integration of the original system, leading to the implemented changes going unnoticed in most users day-to-day activity. Second, a two-week live testing period is likely insufficient to produce measurable behavioral change for features embedded in a newly adopted system, where usage patterns have not yet stabilized. Third, and arguably most consequential, the Follow Your Commit feature, identified in Appendix G as the touch point with the largest daily user group was left unimplemented. By not implementing this feature the population exposed to the intervention was significantly decreased, which directly limits the interpretive weight that can be placed on the null result.

The COIN organization, however, shows a clear decrease in tickets labeled as *Undefined* and a corresponding increase in tickets labeled as *Product*. This shows a pattern that requires further investigation before concluding the new features as the mutual exclusive reason behind it. Three alternative explanations are plausible. First, the pre-study exposed poor labeling practices within COIN and managers may have subsequently prompted teams to label tickets more carefully, which would be a behavioral correction independent of the new tooling. This interpretation is supported by Theme 3 from the thematic analysis of the Flow Guardian interviews (Section 6.3.3), in which Flow Guardians acknowledged that labeling was not consistently prioritized, a pattern also visible in Figure 6.13. Second, since tickets labeled as *Product* are the most straightforward for non-Flow-Guardian developers to classify, as they map directly to commits in product code, individual team managers may have begun asking developers to take ownership of labeling their own commits. Third, it remains possible that the pivot to Pandora had a more limited negative effect than anticipated and the implemented features produced a stronger than expected behavioral change among developers. Given the available data, the first explanation is considered most likely, though these accounts are not mutually exclusive and the data cannot definitively distinguish between them.

Projecting the two-week COIN distribution across a 52-week period to match the

pre-study baseline (Figure 6.32) amplifies the trend observed in the shorter window. Tickets labeled as *Product* increase by approximately 213%. This figure should not be read as a realistic forecast as the projection assumes trend continuity, which cannot be verified from two weeks of data. However, it does illustrate the scale of additional training data the CATegorizer would accumulate if the labeling improvement were sustained. One notable gap in the projection is the complete absence of tickets labeled as *Environment*, which is a direct consequence of no such tickets being recorded during the two-week period. This is not treated as a critical limitation, given that a Flow Guardian indicated during Task 3 of the cognitive walkthrough (Section 6.9.1) that the CATegorizers predictive value is concentrated in the *Test* and *Product* categories as these are the ticket types that system operators can act on. Whereas *Environment* tickets are routed to lab environments and fall outside the tools primary operational scope.

A notable observation from the usage analysis is the decline in CATegorizer predictive accuracy shown in Table 6.6. Predictive accuracy dropped from 65% at baseline to 50% during the two-week period which correspond to a 15 percentage point decrease. While this reduction may appear concerning, we do not interpret it as a direct consequence of the implemented changes. As discussed in Section 7.5, the pivot to Pandora substantially reduced user coverage compared to the interfaces evaluated in the pre-study, which lowers the risk that our changes cause users to label incorrectly. Furthermore, we believe that a two-week window should be considered insufficient to produce a shift of this magnitude through organic labeling behavior alone. Taken together, these factors suggest the accuracy decline is not a consequence of our changes to the system, but instead some outside factor. This uncertainty further motivates the decision to weight the qualitative findings from the survey and interviews more heavily in the overall evaluation.

7.7.2 Interviews and Survey

Although the usage analysis across all organizations shows no notable change in labeling patterns compared to the pre-study baseline, the qualitative data gathered from interviews and surveys reveals a consistently positive attitude toward the implemented changes.

The survey distributed to the COIN organization captured broadly favorable opinions regarding the dashboard's ability to increase knowledge and understanding of the CATegorizer, as illustrated in Figure 6.25. As shown in Figure 6.26, the majority of respondents agreed that the dashboard helped them better understand what the CATegorizer is and why it requires continuous feedback to ensure its longevity. While respondents were uncertain whether the dashboard would motivate them to actively contribute to the training process, no negative sentiment was expressed. This positive trend is further supported by the stakeholder interviews presented in Figure 6.28.

We attribute the disconnect between the positive survey and interview responses and the absence of measurable change in the usage analysis to two factors: the limited time the implementation was live, and the fact that Pandora was not integrated into

users day-to-day workflows. Nevertheless, we interpret these results as a meaningful indicator of the potential value in surfacing and acknowledging internal tools that might otherwise go unnoticed. This addresses one of the key findings from the pre-study as identified in Theme 1 in Section 6.2.5 which was that the CATegorizer suffered from low visibility within the organization. The results presented here suggest that the dashboard has successfully addressed this issue.

7.8 Relating the Project Requirements and Guidelines to Human-AI Interaction Guidelines

The paper by Amershi et al. presented in the beginning of the thesis where they compiled 18 guidelines for human-AI interaction. Their work was focused around a wide range of consumer products that have AI integrated into them such as email filters, music recommenders and navigation systems. We agree that many of their guidelines overlap with the requirements and guidelines formulated from the findings of this thesis' pre-study presented in Table 6.2. The main overlaps can be found surrounding the topics of clear explanations, feedback and contextual relevance. However, Amershi et al.'s guidelines are formulated from the perspective of a generic user when interacting with a standalone AI-infused product. With the findings of this thesis we argue that while this perspective is important it is also insufficient when the AI tool is not the primary software, but instead embedded within an existing professional workflow. An example of this is the NoSA BDC Report (see Section 2.2.10), where the primary tool is the test overview and the CATegorizer functions as an embedded component within it, as illustrated in Figure 2.1. In this context, the Human-AI interaction with the CATegorizer should not be treated the same way as interaction with a standalone AI product such as a chatbot. For instance, a guideline around communicating that the model can be wrong may be less important here, given that this could divert attention from the main use case of the NoSA BDC Report interface.

Based on our work, three aspects of Human-AI interaction appear underrepresented in Amershi et al.s guidelines. The first aspect is when the AI is not the primary product. Unlike consumer AI products where the users actively choose to engage with it the CATegorizer is encountered as part of an existing workflow. This means that the design cannot assume that the user will be motivated to engage with it. Instead, the interaction must feel like a natural part of the process instead of something that interrupts it. Second, Amershi et al.'s guidelines assume that the user is already using the system, but in an industrial context where an AI tool is introduced into an established workflow it has to adapt to it and make users want to use it. The tool must clearly communicate its value to users who did not choose it and must integrate without disrupting existing practices. Third and one of the most important aspects when incorporating AI tools into existing professional workflows is the consideration of different user profiles. Amershi et al's guidelines treat all users as a homogeneous user profile which reflects the consumer product context which is the context of their work. In the context of our work we believe that the same AI output and presentation will be encountered by users of widely different levels of domain

knowledge and goals with the interaction. Designing for a broad heterogeneity in an organizational context may therefore lead to a mismatch between what the tool offers and what different users actually want and need from it, ultimately leading to a lack of interest in engaging with the AI tool.

We believe that these findings suggest that while Amershi et al's guidelines provide a strong foundation for designing successful interaction between a user and an AI system, it does not fully account for a professional and organizational context. The importance of designing for organizational and role context of their users should be at the center of the design process.

7.9 The importance of AI embedded naturally in workflow

The interviews and results consistently indicated that users were reluctant to perform any additional actions beyond their primary workflow in order to support the CATegorizer's training. This aligns with the finding of Stumpf et al. that users are significantly more willing to provide feedback when it is embedded naturally within their existing tasks rather than presented as a separate activity [5]. This principle guided the redesign of CFI Web, where a save button was placed directly adjacent to the model's prediction, allowing users to confirm it with minimal effort. Stakeholders responded positively to this approach, describing it as something that "could not have been done in any better way.". Similarly, the warning triangle used to signal when labeling required attention was well received, with one stakeholder calling it "brilliant." While broader behavioral impact cannot yet be confirmed pending further testing, these reactions suggest the design successfully reduced perceived friction.

7.10 Validity and Reliability

This study used a mixed methods approach to collect both quantitative and qualitative data. This was to make sure that as many aspects of the problem and solution was captured to gain a more complete understanding of the problem from multiple perspectives. However, there are several issues regarding the validity and reliability of this study that needs to be acknowledged.

The first issue is that this work had a rather small sample size compared to the total number of developers. The pre-study survey gathered 60 answers from a population of approximately 970 developers. This is a great amount of people and we believe that it most likely allows this study to capture a great overview of the opinions of the user group. However the final survey only received 6 responses. While these users could share the opinions of the general user, it could still be a threat to the overall validity and reliability of the generalization of their feedback. In addition, as discussed in Section 7.5, the much smaller reach of the implementation and shorter live-testing period as compared to the original plan could also affect the final usage

analysis, but this has been taken into account.

During each step of the study the respondents of each interview and survey was aware that it was part of a study. The participants may have been inclined to provide answers they believed were expected or desired, rather than reflecting their genuine opinion or experience. This social desirability bias poses a threat to the internal validity of the data gathered from the survey and interviews. To mitigate this the participants were assured of anonymity and were reminded that there was no right or wrong answers to any of the questions. Regardless, the possibility of a positivity bias cannot be entirely excluded from the results.

One of the main threats to the validity of the results comes from the natural environment that this study has been conducted within. We as researchers cannot control and account for all external factors, such as influence from teammates or managers, ongoing organizational changes or broader company culture. As a result, it cannot be argued with certainty that the changes proposed in this thesis are the exclusive reason for any change in the behavior of the users. Therefore, the findings of this study should be interpreted as indicative rather than definitively causal and future research in a more controlled setting would help validate these results further.

7.11 Ethical and Social Considerations

This thesis primarily aimed to benefit the direct users of the CI pipeline by improving the reliability of AI-assisted fault estimations. More accurate error categorization was expected to reduce the time spent troubleshooting failed integrations, thereby improving development efficiency and overall team delivery speed. In addition, providing guidance on how to support and maintain model performance can contribute to the development of more stable AI systems and increase their longevity.

The focus of the project was on understanding how to design more efficient, reliable, and sustainable AI tools for internal use at Ericsson. As the CATegorizer was an entirely internal tool, the primary ethical and social considerations relate to employees within the organization. According to our company supervisor, approximately 1000 employees interact with the tool either actively or passively. It was therefore important that any proposed changes did not negatively affect users' everyday work or disrupt established workflows. The large and diverse user base also introduced challenges related to inclusion. Users differ in their working styles, preferences, and needs in order to function optimally. While the proposed solution followed established industry design guidelines in an effort to accommodate as many users as possible, it cannot be guaranteed that all users felt equally supported. This risk of exclusion was an important consideration during the prototyping and evaluation phases of the work. There could potentially be risks involved with nudging user behavior, especially with users who were very comfortable in the current state of their workflow. Introducing changes can cause frustration, which in turn could lead to decreased productivity and a perceived drop in efficiency, something that goes completely against our research question. This is why it was so important to conduct a thorough user study to ensure that our changes would not create any such

frustrations.

There could also risks be inherent to increasing trust in AI systems. As users begin to trust the AI more and more, they could start to become over-reliant on the tool, forgetting to double check results and predictions [32]. This could lead to erroneous use, which in turn can lead to decreased efficiency in the workflow. Over time, over-reliance could also lead to the AI learning behavior that is incorrect and false because of it not being double check, leading to it becoming more and more unreliable. As this thesis was conducted at a private company the project had to follow company policies and some of the content in this thesis is therefore redacted.

7.11.1 AI Ethics

As of 2024, artificial intelligence systems deployed within the European Union are subject to the EU AI Act. The Act is a legal framework with the primary objective is to ensure the safety, trustworthiness and protection of fundamental rights in AI systems [33]. The CATegorizer is an internal tool used to increase the productivity in the troubleshooting of the CI pipeline. The AI tool can therefore not cause any harm apart from decreasing the efficiency of troubleshooting if the output of the model is wrong. However there are some aspects of the AI act that should be discussed surrounding the CATegorizer.

A central concern in our work addressed by the Act is the presence of bias in AI models. Training data must accurately reflect the distribution of the underlying data and must not systematically favor any particular category. In the context of the CATegorizer, human-written labels carry a risk of bias, as some issue categories may be more intuitive or straightforward for users to identify than others. While such cognitive bias is to some extent unavoidable as they are rooted in human judgment, it becomes a concern when the user interface steers users toward particular categorizations. Even without a direct intent, subtle design choices can introduce systematic biases into the training data over time, degrading the quality and fairness of the model.

Transparency is another important requirement under the Act. Article 50 discusses that AI systems must be transparent in their operation and in how they produce outcomes. Applied to our work, this means that users should be clearly informed that their interactions with the CATegorizer contribute to its training. Transparency of this kind is foundational to building user trust, which in turn supports sustained and meaningful engagement with the tool.

The AI Act further discusses that AI systems must incorporate mechanisms for human oversight, both to continuously monitor model behavior and to ensure that someone can be held accountable if an issue were to occur. Article 14 of the Act specifically requires that high-risk AI systems need to be designed to allow natural persons to oversee, intervene and correct the system outputs where necessary. Although the CATegorizer should not be seen as a high-risk AI system, it still incorporates direct human feedback as a part of the training loop, directly addressing this requirement. By enabling users to actively participate in labeling and correc-

tion, the system ensures that a human remains in the loop at each stage of model development, thereby preserving both accountability and the ability to identify and remediate unintended model behavior before it propagates.

Finally, the Act prohibits AI systems from manipulating user behavior in harmful ways or exploiting user vulnerabilities. This is relevant to our work, given that our interface designs are intended to influence how users provide input. It was therefore essential that all such design choices are made with user well-being in mind, and that any behavioral nudges remain within the ethical boundaries set out by the Act.

7.12 Future Work

Although the results of this thesis answers the research question (see Section 1.2) in many ways, there are aspects that requires more development to further improve the CATegorizer as a tool.

One area for future work concerns the underlying model on which the CATegorizer is built. The tool was developed approximately six years prior to this thesis, during a period when AI model performance has been rapidly evolving and a more modern alternative would likely improve its generalization capabilities and predictive accuracy. One feature of the CATegorizer that was explicitly excluded from the scope of this thesis is its free-text explanatory output, where the model generates a natural language description of the reasoning behind each prediction. This feature was excluded because the output was consistently too difficult to interpret, not only for developers with limited knowledge of the area, but sometimes even for experienced users. It was determined that this could not be resolved through changes to how it was displayed alone. However, replacing the underlying model with a more modern alternative would likely produce clearer and more informative explanations, potentially making this feature viable in the future. While the benefits of such an upgrade are evident, it fell outside the scope of this thesis and was therefore not pursued.

A second area that deserves further investigation is user trust in the CATegorizer. This was raised repeatedly as a concern during the pre-study, and while the prediction accuracy improvements addressed in this thesis partially mitigate the issue, the topic of trust was not explored in sufficient depth to draw firm conclusions. Notably, the paper *Soliciting Human-in-the-Loop User Feedback: Effects on Trust and Perceived Model Accuracy*, presented in Section 3.1, demonstrates that soliciting user feedback on an AI model's output can paradoxically lower users' perceived accuracy of the model. This is a highly relevant finding in the context of the CATegorizer and represents a promising direction for future research.

8

Conclusion

This thesis set out to explore how the user interface of an AI-assisted error categorization tool can be designed to encourage users to provide feedback to the underlying AI model in an industrial setting. The model in question, called the CATegorizer, is part of the CI pipeline at Ericsson. A mixed-methods pre-study, including interviews, a survey, and usage analytics, examined users' interactions with the CI workflow.

The results suggested that knowledge of the CATegorizer and how it could be used was generally low among most respondents, although a specialized user group (the Flow Guardians) were much more familiar with it. The pre-study also suggested some discrepancies between what different user groups expected of each other and of their own responsibilities while collaborating in the CI flow. The gathered data was processed and analyzed using established methods common in the field of interaction design, such as thematic analysis. From these analyses a set of requirements and guidelines were defined that would guide the ideation and implementation of the potential solutions.

Through a generative ideation phase anchored in the pre-study, a set of potential solutions were created in the form of features that could be implemented in the different systems of the CI workflow. At this point in the project the existence of Pandora was revealed by the operations team; a new interface still in development that would incorporate the functionality and purpose of many of the standalone systems that had been assessed during the pre-study. Rather than redoing the ideation process, the decision was made to adapt the features to fit the Pandora environment instead. Since the interfaces of Pandora were based on the old systems and thus similar in design and functionality, it was reasoned that the features could be adapted without much extra work and still have a solid foundation in the pre-study.

Three of the features pitched to the operations team were accepted for implementation. However, due to the time limit for the project, the development plans of the operations team and certain system limitations, only two of the features were possible to realize within the time frame. The first was a dashboard in Pandora intended to display information regarding the current and historic status of the CATegorizer, allowing for users to gain an understanding of how the model's performance changes over time. It also showed how their work can impact the performance of the model, making clearer the feedback loop that the pre-study suggested to be largely unfa-

miliar to the users. The second feature consisted of two minor changes to the CFI Web interface, where a small warning triangle was added and displayed when the Problem Type of a given ticket was set to Undefined. The intention was to nudge users toward setting the Problem Type to something other than Undefined whenever possible, thus increasing the amount of data for the model to be trained on. A simple confirmation button was also added next to the CATegorizer prediction, allowing users to quickly set the Problem Type to the one predicted. Together, the implementation of these two features addressed different parts of the research question. The dashboard aimed to increase users' understanding and motivation, while the CFI Web changes aimed to reduce the effort required for users to provide feedback.

The final user study included semi-structured interviews, a survey and usage analytics, following the same principles as the pre-study. The results suggested a generally positive attitude toward the dashboard and the CFI Web changes, with many respondents and interviewees indicating that they believed that these solutions could have a positive impact on their engagement with CATegorizer and its feedback loop. Though the final usage analytics did not show a clear overall improvement in the feedback received by the model, this could possibly be attributed to external factors such as the much smaller user base in Pandora and the relatively short live testing time period. Furthermore, the decrease in tickets labeled as Undefined within the COIN organization is interesting but requires deeper investigation before any conclusions can be drawn.

When analyzing how this project compares to related research and literature, such as the Human-AI Guidelines proposed by Amershi et al. discussed in Section 7.8, the results can potentially be considered as generally in alignment with current research while still delivering something new and of value to the conversation [4]. From this perspective, the thesis provides a situated example of how Human-AI interaction guidelines can be translated into project-specific requirements, guidelines and interfaces in an industrial CI context. The findings also suggest that when embedding AI into workflows it is important that this is done in a way that feels natural to the user. This further supports the findings by Stumpf et al. that feedback is most effective when embedded as on-the-spot interaction within the user's existing task, rather than presented as a separate corrective activity [5].

When considering the research question at the center of the project, the findings suggest that user engagement and feedback can be encouraged through appropriate UI design choices that increase awareness and decrease system friction. In this project, the dashboard served to increase awareness, understanding and motivation among users by making the CATegorizer's performance and feedback loop visible. The changes made to the CFI Web interface served to simplify and encourage the process of providing feedback to the model, while decreasing system friction. Although the limited user reach of the Pandora platform and short live testing period prevent stronger claims regarding long-term behavioral change, the results do indicate that this approach is a promising way to encourage user feedback without negatively affecting perceived efficiency.

Bibliography

[1] G. C. Smith, "What is interaction design," *Designing interactions*, pp. 8–19, 2007.

[2] I. Shumailov, Z. Shumaylov, Y. Zhao, N. Papernot, R. Anderson, and Y. Gal, "Ai models collapse when trained on recursively generated data," *Nature*, vol. 631, no. 8022, pp. 755–759, 2024. DOI: 10.1038/s41586-024-07566-y. [Online]. Available: https://doi.org/10.1038/s41586-024-07566-y.

[3] u/Foreign_Builder_2238, *I tried the "create the exact replica of this image, don't change a thing" 101 times, but with Dwayne Johnson*, Reddit (r/ChatGPT), Accessed: 2024-05-22, May 2024. [Online]. Available: https://www.reddit.com/r/ChatGPT/comments/1kbj71z/i_tried_the_create_the_exact_replica_of_this/.

[4] S. Amershi et al., "Guidelines for human-ai interaction," in *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems (CHI '19)*, Glasgow, Scotland, UK: Association for Computing Machinery, May 2019, pp. 1–13, ISBN: 978-1-4503-5970-2. DOI: 10.1145/3290605.3300233. [Online]. Available: https://doi.org/10.1145/3290605.3300233.

[5] S. Stumpf et al., "Interacting meaningfully with machine learning systems: Three experiments," *International Journal of Human-Computer Studies*, vol. 67, no. 8, pp. 639–662, 2009, ISSN: 1071-5819. DOI: https://doi.org/10.1016/j.ijhcs.2009.03.004. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1071581909000457.

[6] D. Honeycutt, M. Nourani, and E. Ragan, "Soliciting human-in-the-loop user feedback for interactive machine learning reduces user trust and impressions of model accuracy," in *Proceedings of the AAAI Conference on Human Computation and Crowdsourcing*, vol. 8, Oct. 2020, pp. 63–72. DOI: 10.1609/hcomp.v8i1.7464. [Online]. Available: https://doi.org/10.1609/hcomp.v8i1.7464.

[7] T. Humphreys, L. Leung, and A. Weakley, "Embedding expert users in the interaction design process: A case study," *Design Studies*, vol. 29, no. 6, pp. 603–622, 2008. DOI: 10.1016/j.destud.2008.07.006.

[8] Y. Rogers, H. Sharp, and J. Preece, *Interaction Design: Beyond Human-Computer Interaction*, 6th ed. John Wiley & Sons, Inc., Apr. 2023, ISBN: 9781119901099.

[9] J. W. Creswell, *Research Design: Qualitative, Quantitative, and Mixed Methods Approaches*, 3rd. Thousand Oaks, California: SAGE Publications, Inc.,

2009, Accessed via ResearchGate upload by Rulinawaty Kasmad, ISBN: 978-1-4129-6557-6. [Online]. Available: https://www.researchgate.net/publication/342223523_THIRD_EDITION.

[10] Avahi. "Usage analytics." Glossary, Avahi, Accessed: Feb. 11, 2026. [Online]. Available: https://avahi.ai/glossary/usage-analytics/.

[11] S. Brinkmann and S. Kvale, *Doing Interviews*, 2nd ed. SAGE Publications Ltd, 2018. DOI: 10.4135/9781529716665.

[12] M. S. Lewis-Beck, A. Bryman, and T. Futing Liao, "Semistructured interview," in *The SAGE Encyclopedia of Social Science Research Methods*, M. S. Lewis-Beck, A. Bryman, and T. Futing Liao, Eds., Sage Publications, Inc., 2004, pp. 1021–1021. DOI: 10.4135/9781412950589.n909.

[13] H. L. Ball, "Conducting online surveys," *Journal of Human Lactation*, vol. 35, no. 3, pp. 413–417, 2019, PMID: 31084575. DOI: 10.1177/0890334419848734. eprint: https://doi.org/10.1177/0890334419848734. [Online]. Available: https://doi.org/10.1177/0890334419848734.

[14] R. Likert, "A technique for the measurement of attitudes," *Archives of Psychology*, vol. 22, no. 140, pp. 1–55, 1932.

[15] S. G. Hart and L. E. Staveland, "Development of NASA-TLX (Task Load Index): Results of empirical and theoretical research," in *Human Mental Workload*, P. A. Hancock and N. Meshkati, Eds., Amsterdam, Netherlands: North-Holland, 1988, pp. 139–183. DOI: 10.1016/S0166-4115(08)62386-9. [Online]. Available: https://human-factors.arc.nasa.gov/groups/TLX/downloads/Hart_Staveland_ORIGINAL_1.pdf.

[16] S. G. Hart, "Nasa-task load index (nasa-tlx); 20 years later," *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, vol. 50, no. 9, pp. 904–908, 2006. DOI: 10.1177/154193120605000909. eprint: https://doi.org/10.1177/154193120605000909. [Online]. Available: https://doi.org/10.1177/154193120605000909.

[17] T. Kosch, J. Karolus, J. Zagermann, H. Reiterer, A. Schmidt, and P. W. Woniak, "A survey on measuring cognitive workload in human-computer interaction," *ACM Computing Surveys*, vol. 55, no. 13s, pp. 1–39, Jul. 2023, ISSN: 0360-0300. DOI: 10.1145/3582272. [Online]. Available: https://doi.org/10.1145/3582272.

[18] E. Babaei, T. Dingler, B. Tag, and E. Velloso, "Should we use the nasa-tlx in hci? a review of theoretical and methodological issues around mental workload measurement," *International Journal of Human-Computer Studies*, vol. 201, p. 103 515, 2025, ISSN: 1071-5819. DOI: https://doi.org/10.1016/j.ijhcs.2025.103515. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1071581925000722.

[19] M. M. Bradley and P. J. Lang, "Measuring emotion: The self-assessment manikin and the semantic differential," *Journal of Behavior Therapy and Experimental Psychiatry*, vol. 25, no. 1, pp. 49–59, Mar. 1994. DOI: 10.1016/0005-7916(94)90063-9.

[20] L. Militello and R. Hutton, "Applied cognitive task analysis (acta): A practitioner's toolkit for understanding cognitive task demands," *Ergonomics*, vol. 41, pp. 1618–41, Dec. 1998. DOI: 10.1080/001401398186108.

[21] V. Braun and V. Clarke, "Using thematic analysis in psychology," *Qualitative Research in Psychology*, vol. 3, no. 2, pp. 77–101, 2006. DOI: 10.1191/1478088706qp063oa. eprint: https://doi.org/10.1191/1478088706qp063oa. [Online]. Available: https://doi.org/10.1191/1478088706qp063oa.

[22] J. Janhager, "User consideration in early stages of product development – theories and methods," Doctoral thesis, Royal Institute of Technology (KTH), Stockholm, Sweden, 2005.

[23] J. Janhager, "Classification of users: Due to their relation to the product," in *Proceedings of the International Conference on Engineering Design (ICED 03)*, Stockholm, Sweden, Aug. 2003.

[24] L.-O. Bligård. "ACD3-processen," Accessed: Jan. 13, 2026. [Online]. Available: http://acd3.com/about.html.

[25] I. Sommerville and P. Sawyer, *Requirements Engineering: A Good Practice Guide*. John Wiley & Sons, 1997, ISBN: 9780471974444.

[26] J. Nielsen, *Usability Engineering*. Boston: Academic Press, 1993, ISBN: 9780125184069.

[27] International Organization for Standardization, *ISO 9241-210:2019 ergonomics of human-system interaction — part 210: Human-centred design for interactive systems*, https://www.iso.org/standard/77520.html, Accessed: 2026-04-10, 2019.

[28] H. Al-Samarraie and S. Hurmuzan, "A review of brainstorming techniques in higher education," *Thinking Skills and Creativity*, vol. 27, pp. 78–91, 2018, ISSN: 1871-1871. DOI: https://doi.org/10.1016/j.tsc.2017.12.002. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1871187117302729.

[29] T. Kelly, "The art of innovation," in Doubleday, 2000, ch. 4: The Perfect Brainstorm.

[30] T. Kluyver et al., "Jupyter notebooks – a publishing format for reproducible computational workflows," in *Positioning and Power in Academic Publishing: Players, Agents and Agendas*, IOS Press, 2016, pp. 87–90. DOI: 10.3233/978-1-61499-649-1-87.

[31] T. Krotoff, *Git cheat sheet*, https://gist.github.com/tkrotoff/b1caa4c3a185629299ec234d2314e190, GitHub Gist, 2016. Accessed: May 22, 2024.

[32] A. Klingbeil, C. Grützner, and P. Schreck, "Trust and reliance on ai an experimental study on the extent and costs of overreliance on ai," *Computers in Human Behavior*, vol. 160, p. 108 352, 2024, ISSN: 0747-5632. DOI: https://doi.org/10.1016/j.chb.2024.108352. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0747563224002206.

[33] European Parliament and Council of the European Union, *Regulation (eu) 2024/1689 of the european parliament and of the council...* Jun. 13, 2024. Accessed: May 4, 2024. [Online]. Available: http://data.europa.eu/eli/reg/2024/1689/oj.

[34] Y. Geng, Q. Li, G. Yang, and W. Qiu, "Logistic regression," in *Practical Machine Learning Illustrated with KNIME*. Singapore: Springer Nature Singapore, 2024, pp. 99–132, ISBN: 978-981-97-3954-7. DOI: 10.1007/978-981-97-

3954-7_4. [Online]. Available: https://doi.org/10.1007/978-981-97-3954-7_4.

[35] S. S. Skiena, *The Data Science Design Manual* (Texts in Computer Science). Cham, Switzerland: Springer International Publishing, 2017, ISBN: 978-3-319-55444-0. DOI: 10.1007/978-3-319-55444-0. [Online]. Available: https://link.springer.com/book/10.1007/978-3-319-55444-0.

[36] R. Chillarege, I. S. Bhandari, J. K. Chaar, M. J. Halliday, B. K. Ray, and D. S. Moebus, "Orthogonal defect classification—a concept for in-process measurements," *IEEE Transactions on Software Engineering*, vol. 18, no. 11, pp. 943–956, 1992. DOI: 10.1109/32.177364.

[37] S. Dalal and R. S. Chhillar, "Empirical study of root cause analysis of software failure," *ACM SIGSOFT Software Engineering Notes*, vol. 38, no. 4, pp. 1–7, 2013. DOI: 10.1145/2492248.2492263.

[38] M. Shahin, M. A. Babar, and L. Zhu, "Continuous integration, delivery and deployment: A systematic review on approaches, tools, challenges and practices," *IEEE Access*, vol. 5, pp. 3909–3943, 2017. DOI: 10.1109/ACCESS.2017.2685629.

[39] E. Laukkanen, M. Paasivaara, and T. Arvonen, "Stakeholder perceptions of the adoption of continuous integration: A case study," in *Proceedings of the 2015 Agile Conference*, IEEE, 2015, pp. 11–20. DOI: 10.1109/Agile.2015.15.

A

Terminology

This appendix presents some general terminology about AI and Machine Learning, as well as some Ericsson-specific context and terminology.

A.1 Artificial intelligence

Artificial intelligence (AI) is an umbrella term that refers to machines designed to perform tasks that typically require human intelligence [34, p. 6]. The concept can be defined from different perspectives such as if the machine can act human or think like a human. In general, AI encompasses computational systems capable of reasoning, learning and problem solving in ways that resemble human cognitive processes. Today, AI is applied across a wide range of domains, from image recognition to code generation, all of which rely on the ability of machines to process information and identify patterns to produce meaningful outputs in response to given inputs.

A.2 Machine learning

Machine Learning (ML) is the concept of computers adjusting and refining parameters in calculations in order to improve its own capabilities through iterative repetition of tasks [34, p. 8]. This enables the system to outperform its current state and become more efficient in the next execution of similar or identical tasks. The task of machine learning is to create systems that can summarize experiences, discover patterns, learn rules and with this predict the future. Machine learning models can be trained using different paradigms, including supervised learning where models learn from labeled data, unsupervised learning which involves discovering patterns in unlabeled data and reinforcement learning where models are trained through a reward-based system that reinforces correct actions. By training the machine on existing data the model can learn to classify new data or predict a value through regression. Classification aims to identify similarities between data points in order to assign them to predefined classes. By learning these similarities, the model can map new data points to the most similar existing class. Regression, in contrast, focuses on modeling how a dependent variable changes in relation to one or more independent variables, for example over time, in order to capture underlying patterns and predict future values.

A.3 Logistic Regression

Logistic regression is a widely used method for binary classification. In contrast to linear regression, which models continuous outcomes, logistic regression estimates the probability that a given input belongs to a particular class. While linear models can separate data using a straight decision boundary, this assumption is often insufficient in practice. Logistic regression addresses this limitation by applying a non-linear transformation to a linear combination of the input features [35, p. 292].

Given an input vector $\mathbf{x} \in \mathbb{R}^d$, the model estimates the probability that the corresponding label $y \in \{0, 1\}$ belongs to class 1 as:

$$P(y = 1 \mid \mathbf{x}) = \sigma(\mathbf{w}^\top \mathbf{x} + b) \tag{A.1}$$

where $\mathbf{w} \in \mathbb{R}^d$ is the weight vector, $b \in \mathbb{R}$ is the bias term, and $\sigma(\cdot)$ is the sigmoid function defined as:

$$\sigma(z) = \frac{1}{1 + e^{-z}} \tag{A.2}$$

The sigmoid function maps any real-valued input $z \in (-\infty, \infty)$ to the interval $(0, 1)$, allowing the model output to be interpreted as a probability. Consequently, the probability of class 0 is given by:

$$P(y = 0 \mid \mathbf{x}) = 1 - P(y = 1 \mid \mathbf{x}) \tag{A.3}$$

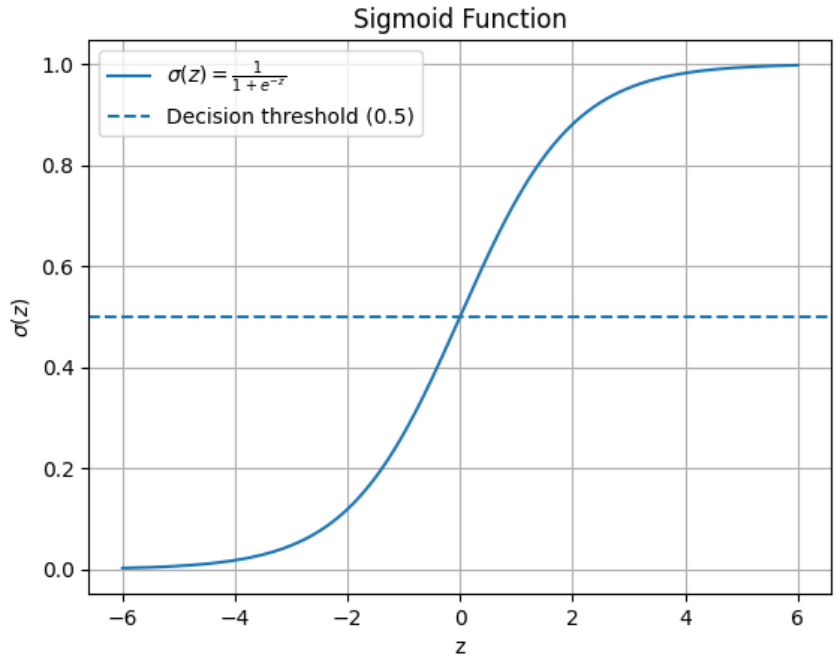


Figure A.1: Sigmoid function used in logistic regression.

To obtain a class prediction, a threshold (commonly 0.5) is applied to the predicted probability. If $P(y = 1 \mid \mathbf{x}) \geq 0.5$, the input is classified as class 1; otherwise, it is classified as class 0, see Figure A.1.

A.4 Training machine learning models

Training a machine learning model involves estimating model parameters such that the predicted outputs closely match the true labels in the predefined training data. This is achieved by defining a loss function, which quantifies the difference between predicted and actual values [34, p. 134]. Different types of models use different loss functions to calculate this discrepancy. Binary logistic regression models use the cross-entropy loss function, which is defined as

$$L(w, b) = -\frac{1}{N} \sum_{i=1}^{N} [y_i \log(\hat{y}_i) + (1 - y_i) \log(1 - \hat{y}_i)] \tag{A.4}$$

where L is the loss, y is the true value, and $\hat{y}$ is the predicted value.

The objective during training is to minimize the loss function by adjusting the model parameters. This is typically done using iterative optimization algorithms such as gradient descent, which computes the gradient of the loss function with respect to the model parameters. The gradient indicates the direction of steepest increase, and by updating the parameters in the opposite direction, the algorithm moves towards a minimum of the loss function. By repeatedly applying this procedure, the model parameters are refined, resulting in improved classification performance.

The goal of minimizing the loss function is not only to achieve good performance on the training data, but also to enable the model to generalize to unseen data to be able to correctly classify it. The amount of training data is important to ensure that the model can generalize effectively. With limited data, the model may learn patterns that are not representative of underlying patterns. By increasing the size and diversity of the training dataset, the model can learn better decision boundaries. For logistic regression, this leads to more reliable estimates of the parameters w and b, which improves the models ability to classify new data accurately.

To evaluate the performance of a classification model, a confusion matrix is commonly used, see Figure A.2. The confusion matrix compares predicted and actual class labels and provides insight into different types of classification outcomes:

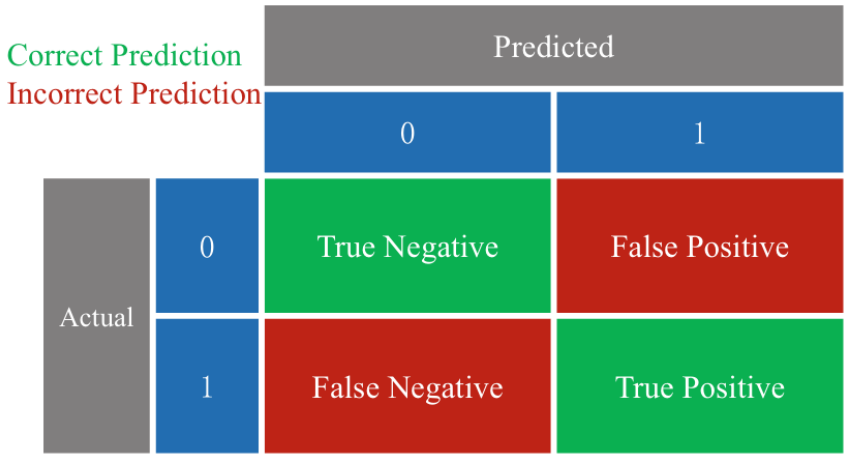


Figure A.2: Confusion matrix

- TP (True Positive) = the problem is detected, and it is indeed a real issue that needs to be addressed.
- FP (False Positive) = the problem is detected, but in reality, there was no problem, causing a false alarm.
- FN (False Negative) = the problem goes undetected, posing a latent risk despite its actual existence.
- TN (True Negative) = no problem is detected, and there was actually no problem, resulting in a satisfactory outcome.

Based on the confusion matrix, several evaluation metrics can be computed. Accuracy measures the overall proportion of correctly classified instances:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{A.5}$$

Precision measures how many of the predicted positive instances are actually correct:

$$\text{Precision} = \frac{TP}{TP + FP} \tag{A.6}$$

Recall measures how many of the actual positive instances are correctly identified:

$$\text{Recall} = \frac{TP}{TP + FN} \tag{A.7}$$

These metrics provide complementary perspectives on model performance and are particularly useful when evaluating classification models.

A.5 Categorization

In the context of software troubleshooting, categorization can be understood as the classification of software errors, defects or failures into predefined categories in order to support diagnosis, communication and correction[36], [37]. Rather than functioning only as a descriptive label, categorizing can help narrow the search for likely causes and provide feedback about recurring issues in the development and testing processes.

A.6 Workflows

The software development workflows at Ericsson follow principles associated with Continuous Integration and Continuous Deployment (CI/CD). Continuous Integration refers to the practice of frequently integrating code changes into a shared codebase, where each integration is verified through automated build and test activities [38], [39]. By continuously improving upon and delivering their software to the customers, the process of adopting new functionality and adapting to new constraints, technologies and regulations becomes smoother. Instead of shipping one large update every few months, CI has the advantage that it turns integration and verification into a continuous, small size activity instead of a rare, high-risk event. In practice, this usually gives the developers faster feedback with earlier error detection and lower integration risk.

B

Survey Questions

The following pages contain the survey questions used in the pre-study. Note that this is a conversion from the Microsoft Forms-platform to a normal PDF.

CATegorizer & BDC Reports Survey

The data collected in this survey will be used to explore potential solutions to issues with the CATegorizer AI model and also to get a better understanding of the overall system/flow surrounding BDC tests. The survey is carried out as part of a Master Thesis by Albin Boklund and Vincent Stocks in collaboration with Michael West. We expect the survey to take around 5-10 minutes to complete.

Any personal information will not be shared in any way, and any potential quotes used in the final report will be anonymized.

Basic info

1 **Role at Ericsson**

Enter your answer

2 **Time spent with Ericsson (years)**

The value must be a number

3 **Time spent in current role (years)**

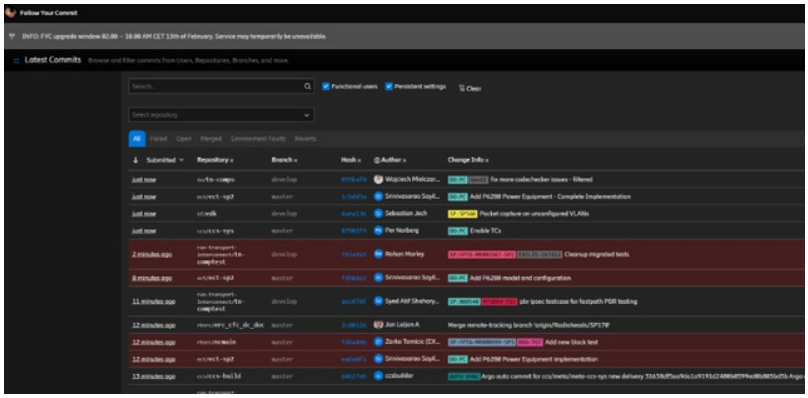
The value must be a number

CATegorizer & BDC Reports Survey

System Usage

4 **How often do you use the Follow Your Commit tool?**

Example interface screenshot:
Link: <link>



- ☐ Daily
- ☐ Weekly
- ☐ Monthly
- ☐ Never

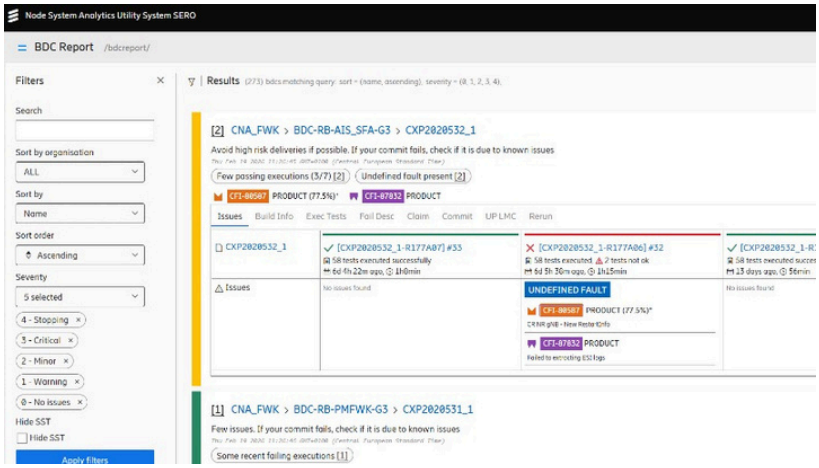
5 **Please elaborate - we want to know when and what you use it for!**

Enter your answer

System Usage

6 **How often do you use the NoSA BDC report interface?**

Example interface screenshot:
Link: <link>



☐ Daily ☐ Weekly ☐ Monthly ☐ Never

7 **Please elaborate - we want to know when and what you use it for!**

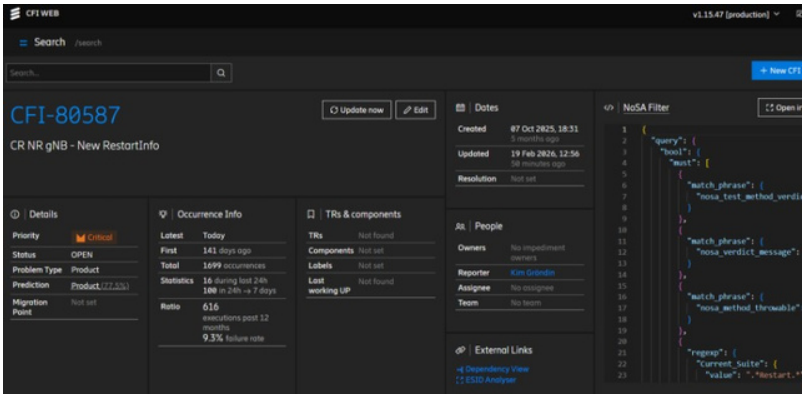
Enter your answer

CATegorizer & BDC Reports Survey

System Usage

8 **How often do you use the CFI WEB interface?**

Example interface screenshot:
Link: <link>



- ☐ Daily
- ☐ Weekly
- ☐ Monthly
- ☐ Never

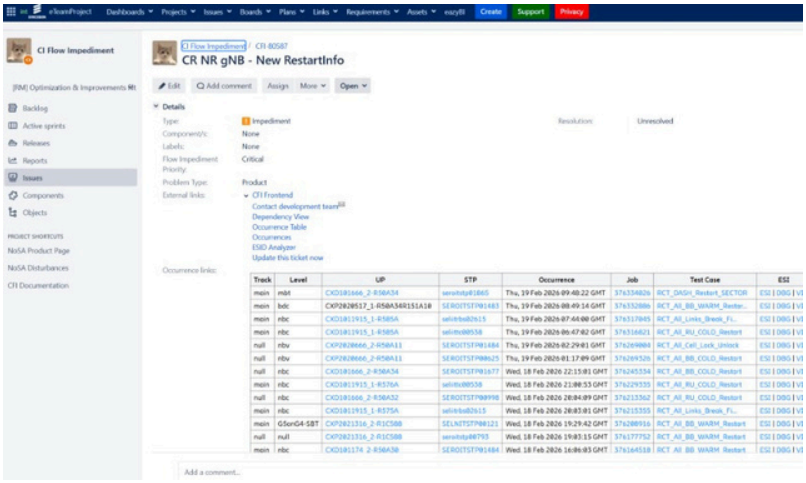
9 **Please elaborate - we want to know when and what you use it for!**

Enter your answer

System Usage

10 **How often do you use the CI Flow Impediment interface?**

Example interface screenshot:
Link: <link>



- [ ] Daily
- [ ] Weekly
- [ ] Monthly
- [ ] Never

11 **Please elaborate - we want to know when and what you use it for!**

Enter your answer

CATegorizer & BDC Reports Survey

System Usage

12 **How often do you use the RAN CI Radiator?**

Example interface screenshot:
Link: <link>

- [ ] Daily
- [ ] Weekly
- [ ] Monthly
- [ ] Never

13 **Please elaborate - we want to know when and what you use it for!**

Enter your answer

14 **How often do you troubleshoot BDCs?**

- [ ] Daily
- [ ] Weekly
- [ ] Monthly
- [ ] Never

15 **What does your BDC troubleshooting process look like? Please explain step by step.**

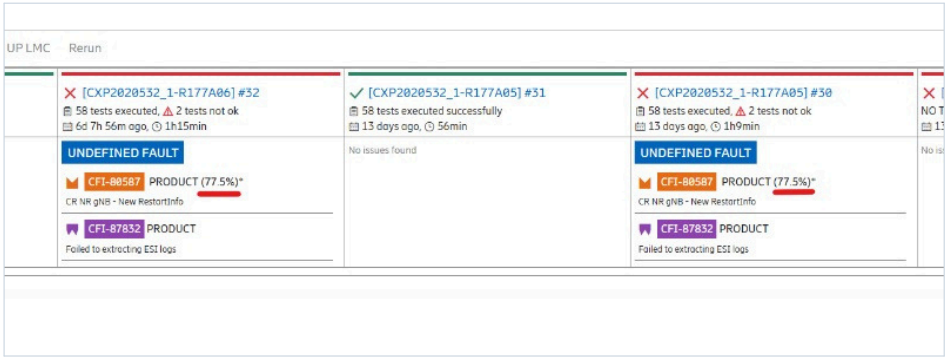
Enter your answer

The CATegorizer & CFI tickets

You may or may not be familiar with the CATegorizer, but it is present in all the systems and interfaces mentioned in the previous section.

When a CFI ticket has no set **Problem Type**, the CATegorizer analyzes the **ESI debug logs** from the failed **BDC-level test** associated with that CFI. Based on this analysis, it predicts whether the underlying issue is a **Product error** (software) or an **Environment error** (lab/hardware).

In NoSA, the prediction is shown next to the CFI label as a Problem Type with a percentage denoted by an asterisk (*) next to it (*see image below)*.
In the CI Flow Impediment interface, it is shown as prediction percentages together with a more detailed explanation.

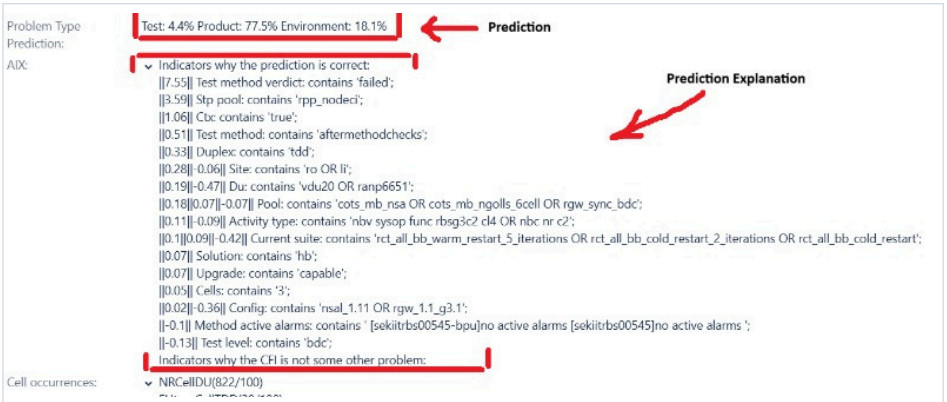


CATegorizer & BDC Reports Survey

The CATegorizer & CFI tickets

16 **How often do the CATegorizer predictions factor into your work?**

Below is an example of how the prediction is presented in the CI Flow Impediment interface.



☐ Daily ☐ Weekly ☐ Monthly ☐ Never

17 **Please elaborate!**

Enter your answer

The CATegorizer & CFI tickets

18 **The following statements concern the CATegorizer and your experience using it. Please select the option that best matches your level of agreement for each statement.**

	Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree
Before starting this survey, I was already familiar with the CATegorizer and its purpose	☐	☐	☐	☐	☐
The CATegorizer predictions are valuable to my workflow	☐	☐	☐	☐	☐
I actively/directly engage with the CATegorizer	☐	☐	☐	☐	☐
I passively/indirectly engage with the CATegorizer	☐	☐	☐	☐	☐
I want the CATegorizer to improve	☐	☐	☐	☐	☐
I want to contribute to improving the CATegorizer	☐	☐	☐	☐	☐

The CATegorizer & CFI tickets

19 **If you have other thoughts related to any of the statements above, please write them here.**

Enter your answer

20 **How often do you create CFI tickets?**

☐ Daily	☐ Weekly	☐ Monthly	☐ Never

21 **Please elaborate!**

Enter your answer

22 **How often do you close CFI tickets?**

☐ Daily	☐ Weekly	☐ Monthly	☐ Never

23 **Please elaborate!**

Enter your answer

The CATegorizer & CFI tickets

24 **How often do you label the problem type of a CFI ticket?**

☐ Daily ☐ Weekly ☐ Monthly ☐ Never

25 **Please elaborate!**

Enter your answer

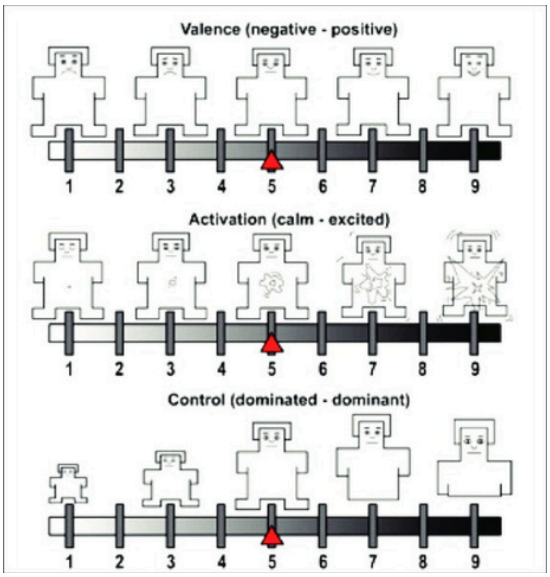
SAM, the Self-Assessment Manikin

To gauge your emotional state when using or interacting with the system flow, we will employ the Self-Assessment Manikin scale. Please consider a use case where you are troubleshooting one of your own commits that has failed at the BDC level.

Valence - how negative/positive (sad/happy) do you feel when interacting with the system?

Activation - how calm/excited do you feel when interacting with the system?

Control - when interacting with the system, do you feel dominated by it, or do you feel that you are dominant over it?



26 **Valence**

1	2	3	4	5	6	7	8	9

27 **Activation**

1	2	3	4	5	6	7	8	9

28 **Control**

1	2	3	4	5	6	7	8	9

Final notes

Thank you so much for your participation! We hope to be able to contribute something of value to the CI workflow, and your help means a lot. Feel free to reach out to us on Teams with any questions regarding this survey or our work.

BR,
Albin Boklund and Vincent Stocks

29 **If we find one of your answers and/or insights especially interesting, may we contact you on Teams for more questions? If yes, please enter your name below (as shown in Teams). If no, please leave the text field blank.**

Enter your answer

C

Interview Questions

The following pages contain the interview questions used in the pre-study during the interviews with the Flow Guardians. Note that this is a conversion from the Microsoft Forms-platform to a normal PDF.

CATegorizer FlowGuardian Interview Response Sheet

The data collected in this interview will be used to explore potential solutions to issues with the CATegorizer AI model and also to get a better understanding of the overall system/flow surrounding BDC tests. The survey is carried out as part of a Master Thesis by Albin Boklund and Vincent Stocks in collaboration with Michael West. We expect the survey to take around 5-10 minutes to complete.

Any personal information will not be shared in any way, and any potential quotes used in the final report will be anonymized.

* This form will record your name, please fill your name.

Basic info

1

Role at Ericsson

Enter your answer

2

Time spent with Ericsson (years)

The value must be a number

The value must be a number

3

Time spent in current role (years)

The value must be a number

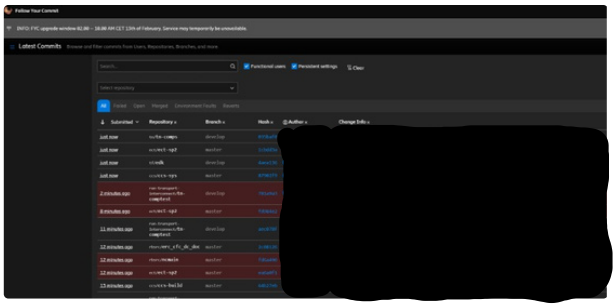
The value must be a number

System Usage

4

How often do you use the Follow Your Commit tool?

Example: <LINK REMOVED>



- ◯ Daily
- ◯ Weekly
- ◯ Monthly
- ◯ Never

5

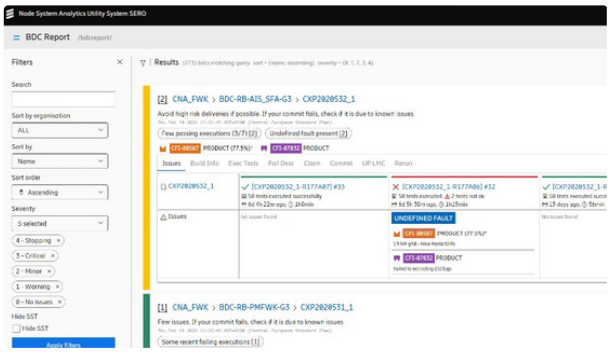
Please elaborate - we want to know when and what you use it for!

Enter your answer

6

How often do you use the NoSA BDC report interface?

Example: <LINK REMOVED>



- ◯ Daily
- ◯ Weekly
- ◯ Monthly
- ◯ Never

7

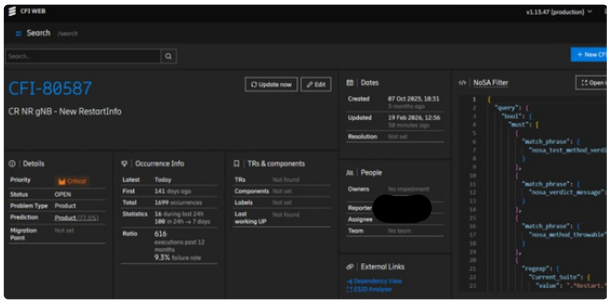
Please elaborate - we want to know when and what you use it for!

Enter your answer

8

How often do you use the CFI WEB interface?

Example: <LINK REMOVED>



- ◯ Daily
- ◯ Weekly
- ◯ Monthly
- ◯ Never

9

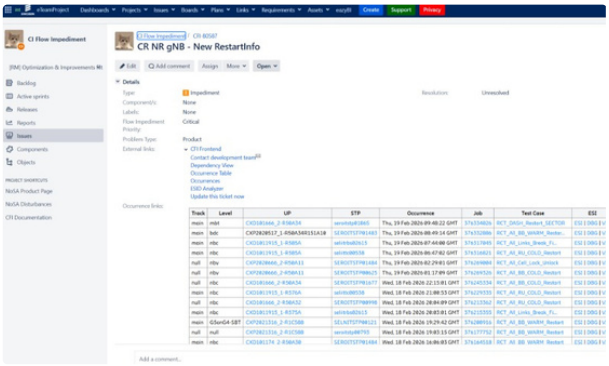
Please elaborate - we want to know when and what you use it for!

Enter your answer

10

How often do you use the CI Flow Impediment interface?

Example: <LINK REMOVED>



- ◯ Daily
- ◯ Weekly
- ◯ Monthly
- ◯ Never

11

Please elaborate - we want to know when and what you use it for!

Enter your answer

12

How often do you use the RAN CI Radiator?

<LINK REMOVED>

- ◯ Daily
- ◯ Weekly
- ◯ Monthly
- ◯ Never

13

Please elaborate - we want to know when and what you use it for!

Enter your answer

14

How often do you troubleshoot BDCs?

- ◯ Daily
- ◯ Weekly
- ◯ Monthly
- ◯ Never

15

What does your BDC troubleshooting process look like? Please explain step by step. This answer will be used in a later section!

Enter your answer

The CATegorizer & CFI tickets

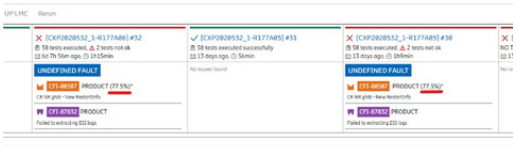
You may or may not be familiar what the CATegorizer is, but it is present in all the systems and interfaces mentioned in the previous section.

When a CFI ticket has no set **Problem Type**, the CATegorizer analyzes the **ESI debug logs** from the failed **BDC-level test** associated with that CFI. Based on this analysis, it predicts whether the underlying issue is a **Product error** (software) or an **Environment error** (lab/hardware).

In NoSA, the prediction is shown next to the CFI label as a Problem Type with a percentage denoted by an asterisk (*) next to it *(see image below)*.

In the CI Flow Impediment interface it is shown as prediction percentages along with a more in-depth explanation of why it landed at that prediction *(see image in the question below)*.

It is also shown in CFI WEB, and in the RAN CI Radiator (helps with color-coding).



16

How often do the CATegorizer predictions **factor into your work**?
Below is an example of how the prediction is presented in the CI Flow Impediment interface.

- ◯ Daily
- ◯ Weekly
- ◯ Monthly
- ◯ Never

17

Please elaborate!

Enter your answer

18

The following statements concern the CATegorizer and your experience using it. Please select the option that best matches your level of agreement for each statement.

	Strongly Disagree	Disagree	Neutral	Agree
Before starting this survey, I was already familiar with the CATegorizer and its purpose	◯	◯	◯	◯
The CATegorizer predictions are valuable to my workflow	◯	◯	◯	◯
I actively/directly engage with the CATegorizer	◯	◯	◯	◯
I passively/indirectly engage with the CATegorizer	◯	◯	◯	◯
I want the CATegorizer to improve	◯	◯	◯	◯
I want to contribute to improving the CATegorizer	◯	◯	◯	◯

19

If you have other thoughts related to any of the statements above, please write them here.

Enter your answer

20

How often do you create CFI tickets?

- ◯ Daily
- ◯ Weekly
- ◯ Monthly
- ◯ Never

21

Please elaborate!

Enter your answer

22

How often do you close CFI tickets?

- ◯ Daily
- ◯ Weekly
- ◯ Monthly
- ◯ Never

23

Please elaborate!

Enter your answer

24

How often do you label the problem type of a CFI ticket?

- ◯ Daily
- ◯ Weekly
- ◯ Monthly
- ◯ Never

25

Please elaborate!

Enter your answer

NASA Task Load Index

In this step we will measure perceived mental workload and stress experienced when working in the CI flow by examining load during a specific Use Case. To do this we will employ the NASA-TLX method, which investigates and compares six different aspects of the users' (your) mental workload. To make it simple for you, please use the tool linked here: https://www.keithv.com/software/nasatlx/nasatlx.html

Use case for NASA-TLX:
Please consider the workflow you outlined in the answer to question 15

When rating workload: include the effort of reading, interpreting the prediction + explanation, deciding what to do, and finally updating the Problem Type. Do not include the effort spent troubleshooting the code or making changes to it.

After completing the input, please copy your whole result table and paste it in the answer box further down.

You can read more about NASA-TLX here:
https://en.wikipedia.org/wiki/NASA-TLX

26

Please paste your whole result below. (Example of a result, copy and paste all of this) - >

	Rating	Tally	Weight
Mental Demand	65	3	0.2
Physical Demand	45	3	0.2
Temporal Demand	30	3	0.2
Performance	70	0	0
Effort	50	4	0.26666666666666666
Frustration	70	2	0.13333333333333333

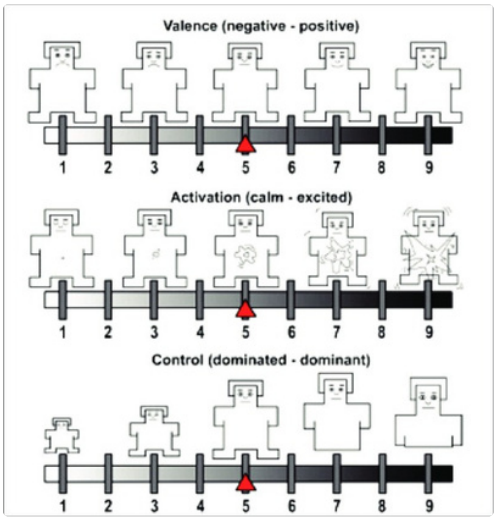
Overall = 50.66666666666667

Enter your answer

SAM, the Self-Assessment Manikin

To gauge the users' (your) emotional state when using/interacting with the system flow (through all or some of the systems mentioned in section 2), we will employ the Self-Assessment Manikin scale. Please consider a Use Case where you are troubleshooting one of your own commits that has failed at the BDC level.

Explanations:
Valence - how negative/positive (sad/happy) do you feel when interacting with the system?
Activation - how calm/excited do you feel when interacting with the system?
Control - when interacting with the system, do you feel dominated by it, or do you feel that you are dominant over it? Who is in control - you or the system?

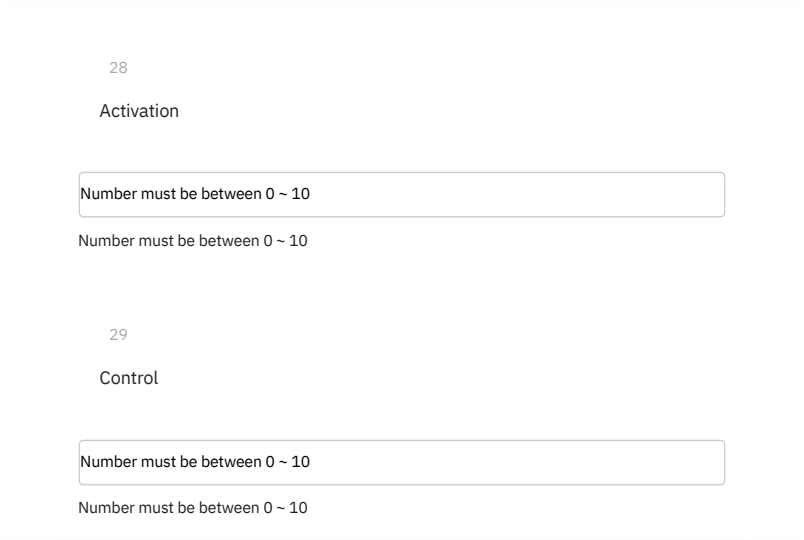


27

Valence

Number must be between 0 ~ 10

Number must be between 0 ~ 10


28
Activation
Number must be between 0 ~ 10
Number must be between 0 ~ 10
29
Control
Number must be between 0 ~ 10
Number must be between 0 ~ 10

ACTA - Applied Cognitive Task Analysis - Knowledge Audit

BASIC PROBES:

Past & Future. Experts can figure out how a situation developed, and they can think into the future to see where the situation is going. Among other things, this can allow experts to head off problems before they develop.
Is there a time when you walked into the middle of a situation and knew exactly how things got there and where they were headed?

Big Picture. Novices may only see bits and pieces. Experts are able to quickly build an understanding of the whole situation—the Big Picture view. This allows the expert to think about how different elements fit together and affect each other.
Can you give me an example of what is important about the Big Picture for this task?
What are the major elements you have to know and keep track of?

Noticing. Experts are able to detect cues and see meaningful patterns that less-experienced personnel may miss altogether.
Have you had experiences where part of a situation just "popped" out at you; where you noticed things going on that others didn't catch? What is an example?

Job Smarts. Experts learn how to combine procedures and work the task in the most efficient way possible. They don't cut corners, but they don't waste time and resources either.
When you do this task, are there ways of working smart or accomplishing more with less — that you have found especially useful?

Opportunities/Improvising. Experts are comfortable improvising—seeing what will work in this particular situation; they are able to shift directions to take advantage of opportunities.
Can you think of an example when you have improvised in this task or noticed an opportunity to do something better?

Self Monitoring. Experts are aware of their performance; they check how they are doing and make adjustments. Experts notice when their performance is not what it should be (this could be due to stress, fatigue, high workload, etc.) and are able to adjust so that the job gets done.
Can you think of a time when you realized that you would need to change the way you were performing in order to get the job done?

OPTIONAL PROBES:

Anomalies. Novices don't know what is typical, so they have a hard time identifying what is atypical. Experts can quickly spot unusual events and detect deviations. And, they are able to notice when something that ought to happen, doesn't.
Can you describe an instance when you spotted a deviation from the norm, or knew something was amiss?

Equipment Difficulties. Equipment can sometimes mislead. Novices usually believe whatever the equipment tells them; they don't know when to be skeptical.
Have there been times when the equipment pointed in one direction, but your own judgment told you to do something else? Or when you had to rely on experience to avoid be- ing led astray by the equipment?

Final notes

Thank you so much for your participation! We hope to be able to contribute something of value to the CI workflow, and your help means a lot. Feel free to reach out to us on Teams with any questions regarding this interview or our work.

BR,
Albin Boklund and Vincent Stocks

D

JIRA Queries Used for Data Collection

This appendix lists all JIRA Query Language (JQL) queries used to retrieve data from the *CI Flow Impediment* project.

Time Period Definitions

The queries were executed multiple times with different values for `START_DATE`. The end date for all queries was `2026-02-28`.

- 2 weeks: `START_DATE = "2026-02-13"`
- 4 weeks: `START_DATE = "2026-01-30"`
- 26 weeks: `START_DATE = "2025-08-29"`
- 52 weeks: `START_DATE = "2025-02-28"`

Queries for Human-Reported Tickets

All closed CFI tickets reported by someone other than dekrabot (i.e., likely human-created tickets) within the selected time period.

```
reporter != dekrabot AND project = CFI AND status = Closed
AND resolved >= "START_DATE" AND resolved < "2026-02-28"
```

All closed human-reported CFI tickets where Problem Type is still set to Undefined within the selected time period.

```
reporter != dekrabot AND project = CFI AND status = Closed
AND "Problem Type" = Undefined
AND resolved >= "START_DATE" AND resolved < "2026-02-28"
```

All closed CFI issues reported by non-dekrabot users where Problem Type = Product within the selected time period.

```
project = CFI AND status = Closed
AND "Problem Type" = Product AND reporter != dekrabot
AND resolved >= "START_DATE" AND resolved < "2026-02-28"
```

All closed CFI issues reported by non-dekrabot users where Problem Type = Environment within the selected time period.

```
project = CFI AND status = Closed
AND "Problem Type" = Environment AND reporter != dekrabot
AND resolved >= "START_DATE" AND resolved < "2026-02-28"
```

All closed CFI issues reported by non-dekrabot users where Problem Type = Test within the selected time period.

```
project = CFI AND status = Closed
AND "Problem Type" = Test AND reporter != dekrabot
AND resolved >= "START_DATE" AND resolved < "2026-02-28"
```

Queries for Dekrabot-Reported Tickets

All closed CFI tickets reported by dekrabot within the selected time period.

```
reporter = dekrabot AND project = CFI AND status = Closed
AND resolved >= "START_DATE" AND resolved < "2026-02-28"
```

All closed dekrabot-reported CFI tickets where Problem Type is Undefined within the selected time period.

```
reporter = dekrabot AND project = CFI AND status = Closed
AND "Problem Type" = Undefined
AND resolved >= "START_DATE" AND resolved < "2026-02-28"
```

All closed CFI issues reported by dekrabot where Problem Type = Product within the selected time period.

```
project = CFI AND status = Closed
AND "Problem Type" = Product AND reporter = dekrabot
AND resolved >= "START_DATE" AND resolved < "2026-02-28"
```

All closed CFI issues reported by dekrabot where Problem Type = Environment within the selected time period.

```
project = CFI AND status = Closed
AND "Problem Type" = Environment AND reporter = dekrabot
AND resolved >= "START_DATE" AND resolved < "2026-02-28"
```

All closed CFI issues reported by dekrabot where Problem Type = Test within the selected time period.

```
project = CFI AND status = Closed
AND "Problem Type" = Test AND reporter = dekrabot
AND resolved >= "START_DATE" AND resolved < "2026-02-28"
```

Queries for All Tickets

All closed issues in the CFI project, regardless of reporter or Problem Type, within the selected time period.

```
project = CFI AND status = Closed
AND resolved >= "START_DATE" AND resolved < "2026-02-28"
```

All closed CFI issues where Problem Type = Undefined within the selected time period.

```
project = CFI AND status = Closed
AND "Problem Type" = Undefined
AND resolved >= "START_DATE" AND resolved < "2026-02-28"
```

All closed CFI issues where Problem Type = Product, regardless of reporter, within the selected time period.

```
project = CFI AND status = Closed
AND "Problem Type" = Product
AND resolved >= "START_DATE" AND resolved < "2026-02-28"
```

All closed CFI issues where Problem Type = Environment, regardless of reporter, within the selected time period.

```
project = CFI AND status = Closed
AND "Problem Type" = Environment
AND resolved >= "START_DATE" AND resolved < "2026-02-28"
```

All closed CFI issues where Problem Type = Test, regardless of reporter, within the selected time period.

```
project = CFI AND status = Closed
AND "Problem Type" = Test
AND resolved >= "START_DATE" AND resolved < "2026-02-28"
```

E

Thematic Analysis of COIN Survey

The following pages contain the thematic analysis conducted on the answers from the survey that was sent out to the COIN organization. The analysis was conducted in Microsoft excel and exported to PDF before being added in this paper.

primary_family	secondary_family	confidence	rationale
Explanation / transparency needed	Workflow pressure / time constraints	0,89	Says it's too complicated and hard to find information, needing clearer guidance.
Explanation / transparency needed		0,93	Says it is not clear what the tool does, indicating need for explanation.
Explanation / transparency needed		0,94	Finds the UI too complicated and hard to understand, implying need for clearer explanation.
Explanation / transparency needed		0,82	Requests the tool be more user friendly, implying usability and clarity improvements.
I didn't know this existed	I lack knowledge to use this	0,95	Used only a few times, implying very limited exposure and familiarity.
I didn't know this existed	I lack knowledge to use this	0,96	Asking what it is indicates not knowing the tool exists and lacking familiarity.
I didn't know this existed	I lack knowledge to use this	0,97	Very direct statement that they have never used it.
I didn't know this existed	I lack knowledge to use this	0,99	Directly states uncertainty about what the tool is.
I didn't know this existed	Routine / habitual use	0,97	Explicitly says they didn't know it existed and have not checked it again.
I didn't know this existed	Workflow pressure / time constraints	0,78	Mentions missed information and suggests it should be part of the process.
I didn't know this existed		0,99	Explicitly says they've never heard of it before.
I didn't know this existed		0,99	States they didn't know the tool existed.
I didn't know this existed		0,96	Question indicates unfamiliarity with the tool.
I didn't know this existed		0,95	Says they had not noticed it before, implying lack of awareness.
I didn't know this existed		0,99	Explicitly says they never heard of it.
I didn't know this existed		0,99	Directly says they did not know about it.
I didn't know this existed		0,99	Explicitly says they didn't know it existed.
I didn't know this existed		0,99	Explicitly says they didn't know it existed.
I didn't know this existed		0,99	Directly says they didn't know the tool existed.
I didn't know this existed		0,95	Asks 'what's this?', showing no awareness of the tool.
I didn't know this existed		0,98	Explicitly says they never heard of it.
I didn't know this existed		0,97	Asks 'What is it?', indicating no prior awareness.
I didn't know this existed		0,98	States they have never seen it, implying no awareness.
I didn't know this existed		0,99	Explicitly says they didn't know about its existence.
I didn't know this existed		0,94	Directly states never used it; likely unfamiliar with it.
I didn't know this existed		1	Directly says they didn't know it existed.
I didn't know this existed		0,98	Asking what it is indicates no prior awareness.
I didn't know this existed		0,91	No idea implies lack of awareness of the tool.
I didn't know this existed		0,97	Never seen it clearly signals unfamiliarity.
I didn't know this existed		0,99	Explicitly states they did not know it existed.
I didn't know this existed		0,99	Explicitly says they were unaware of the tool.
I didn't know this existed		0,98	States they don't know anything about the tool, indicating lack of awareness.
I didn't know this existed		0,93	Says they didn't need it, implying no use and likely little awareness.
I didn't know this existed		0,99	Directly states they didn't know the tool existed.
I didn't know this existed		0,98	'what's this?' clearly shows unfamiliarity with the tool.
I didn't know this existed		0,95	Simply says never used, indicating non-use; no other context.
I didn't know this existed		0,98	Directly asks 'what's this?', indicating lack of awareness of the tool.
I didn't know this existed		0,95	Asks whether they should use it, suggesting unfamiliarity with the tool's existence or purpose.
I lack knowledge to use this	Explanation / transparency needed	0,84	Says the tool is confusing, indicating insufficient understanding and need for clarity.
I lack knowledge to use this	I didn't know this existed	0,88	Not familiar enough and unsure what it can be used for.
I lack knowledge to use this	I didn't know this existed	0,95	Directly says they are not familiar with the tool, implying lack of knowledge.
I lack knowledge to use this	I didn't know this existed	0,96	States unfamiliarity with the tool and not seeing what it can be used for.

Final Code

THEME 1

More information/education regarding the system is needed

- Explanation / transparency needed
- I didn't know this existed
- I lack knowledge to use this

THEME 2

Users retain control through manual investigation and monitoring

- Monitoring the system and/or my commits
- Manual verification / checking logs
- Low trust / manual confirmation needed
- Multi-system troubleshooting
- Problem identification / root cause finding
- Routine / habitual use
- Step-by-step troubleshooting workflow

THEME 3

The current predictions are useful, but mainly as a supporting tool

- Prediction supports prioritization / overview
- Prediction influences work / decision-making
- Prediction has limited role

THEME 4

Practical constraints limit engagement

- Workflow pressure / time constraints
- Feedback seen as extra effort
- This is not my job or responsibility

Other / unclear

I lack knowledge to use this	I didn't know this existed	0,97	Explicitly says they are not familiar enough with the tool and have not seen it used enough to know its purpose.
I lack knowledge to use this	Other / unclear	0,89	Mentions not learning how to configure a filter, indicating limited knowledge.
I lack knowledge to use this	Other / unclear	0,89	Says it is messy and not sure what can be done, suggesting unclear understanding of the tool.
I lack knowledge to use this		0,97	Says they never used it because they are new, indicating lack of familiarity/experience.
I lack knowledge to use this		0,93	Indicates they need to first learn or try using it, showing lack of knowledge or experience.
I lack knowledge to use this		0,95	Says they are not familiar enough with the tool and its use cases.
I lack knowledge to use this		0,97	States they are not familiar with the tool.
I lack knowledge to use this		0,99	Explicitly says they don't know how and when to use it or find it.
I lack knowledge to use this		0,98	Directly says they are not familiar with the tool.
Low trust / manual confirmation needed	Problem identification / root cause finding	0,95	Does root-cause checking instead because the prediction is not very accurate.
Low trust / manual confirmation needed	Problem identification / root cause finding	0,89	Relies on the tool to judge whether failures are real faults versus normal behavior.
Manual verification / checking logs	Low trust / manual confirmation needed	0,84	Looks at logs for the same fault and notes false positives, indicating verification.
Manual verification / checking logs	Monitoring the system and/or my commits	0,98	Directly describes checking Jenkins output logs for troubleshooting.
Manual verification / checking logs	Monitoring the system and/or my commits	0,95	Checks logs specifically to see what went wrong with their commit.
Manual verification / checking logs	Monitoring the system and/or my commits	0,93	Verifying passes and delivery success reflects checking outcomes manually while monitoring commit progress.
Manual verification / checking logs	Monitoring the system and/or my commits	0,9	Looks for errors in test suites and codechecker to confirm code quality.
Manual verification / checking logs	Monitoring the system and/or my commits	0,93	Checks commit test results and SST when needed, mainly verifying outcomes and monitoring own commits.
Manual verification / checking logs	Multi-system troubleshooting	0,91	Uses multiple tools, inspects logs, and rebuilds jobs to diagnose unstable or failing BDCs.
Manual verification / checking logs	Multi-system troubleshooting	0,96	Explicitly mentions going through several logs and tools to find the failure time before troubleshooting.
Manual verification / checking logs	Multi-system troubleshooting	0,95	Downloads logs and checks multiple evidence sources like dbis and traces.
Manual verification / checking logs	Multi-system troubleshooting	0,95	Uses multiple diagnostic sources: dcgm, moshell, dbi, and trace logs.
Manual verification / checking logs	Other / unclear	0,97	Explicitly says checking logs for failures.
Manual verification / checking logs	Other / unclear	0,96	Checks logs and reruns locally to troubleshoot.
Manual verification / checking logs	Prediction has limited role	0,87	They still do manual prescreening first, with the prediction only sometimes glanced at.
Manual verification / checking logs	Problem identification / root cause finding	0,96	Describes retrieving logs and checking why a test failed, plus checking frequency of occurrence.

Manual verification / checking logs	Problem identification / root cause finding	0,97	Downloads and opens logs to check whether the commit caused the issue.
Manual verification / checking logs	Problem identification / root cause finding	0,94	Checks logs and diffs to determine when failure started and probable cause.
Manual verification / checking logs	Problem identification / root cause finding	0,98	Describes checking failed test case, errors, commands, and logs to diagnose the issue.
Manual verification / checking logs	Problem identification / root cause finding	0,92	Explicitly mentions finding logs for a specific issue.
Manual verification / checking logs	Problem identification / root cause finding	0,95	On-demand use to review failed jobs and get logs.
Manual verification / checking logs	Problem identification / root cause finding	0,87	Seeks additional logs or information while working on an issue.
Manual verification / checking logs	Problem identification / root cause finding	0,78	Uses history to inspect failing test cases and possibly feature usage.
Manual verification / checking logs	Problem identification / root cause finding	0,93	Checks why a commit failed, likely by inspecting results/logs to diagnose the failure.
Manual verification / checking logs	Problem identification / root cause finding	0,97	Uses failure logs for troubleshooting, combining log checking with diagnosis.
Manual verification / checking logs	Step-by-step troubleshooting workflow	0,96	Checks test runs and then looks for logs and results when something fails.
Manual verification / checking logs	Workflow pressure / time constraints	0,84	Checks failing BDCs and test links manually; access problems made the investigation harder.
Manual verification / checking logs		0,98	Simply says to look at collected logs.
Manual verification / checking logs		0,85	Checking an occurrence table is manual verification of events.
Monitoring the system and/or my commits	Low trust / manual confirmation needed	0,95	Watches for merge failures, clearly monitoring commit status.
Monitoring the system and/or my commits	Manual verification / checking logs	0,88	Troubleshooting is tied to their own commit and related errors.
Monitoring the system and/or my commits	Manual verification / checking logs	0,94	Centers on whether the failure came from their commit, then checks logs and dumps.
Monitoring the system and/or my commits	Manual verification / checking logs	0,92	Checks if the issue is theirs, then inspects logs for the failure.
Monitoring the system and/or my commits	Manual verification / checking logs	0,94	Checking whether a commit passed the CI pipeline is monitoring, with some verification.
Monitoring the system and/or my commits	Manual verification / checking logs	0,97	Checks test passing before and after submit and verifies promotion to CL3.
Monitoring the system and/or my commits	Manual verification / checking logs	0,98	Explicitly says following commits and checking SDC results when something fails.
Monitoring the system and/or my commits	Manual verification / checking logs	0,93	Tracks commits to see whether they pass all tests.
Monitoring the system and/or my commits	Manual verification / checking logs	0,96	Checks commit status and whether tests fail.
Monitoring the system and/or my commits	Manual verification / checking logs	0,97	Daily checking whether own or colleagues' commits pass all tests.

Monitoring the system and/or my commits	Manual verification / checking logs	0,98	Focuses on tracking delivered commits, checking logs, and troubleshooting failures.
Monitoring the system and/or my commits	Manual verification / checking logs	0,9	Checks CFI status and logs from other jobs.
Monitoring the system and/or my commits	Manual verification / checking logs	0,86	Describes tracking failure frequency and checking CI flow status after commits.
Monitoring the system and/or my commits	Manual verification / checking logs	0,89	Checking the latest run status is classic monitoring and verification.
Monitoring the system and/or my commits	Multi-system troubleshooting	0,92	Regularly checks BDCs, test cases, and states across flows/modules; primarily monitoring with cross-system scope.
Monitoring the system and/or my commits	Multi-system troubleshooting	0,88	Tracks jobs and suites across products and test types, indicating monitoring with cross-system scope.
Monitoring the system and/or my commits	Other / unclear	0,91	Checks whether a bad delivery came from their own work.
Monitoring the system and/or my commits	Other / unclear	0,81	Checking status and version/relationship details is mainly monitoring.
Monitoring the system and/or my commits	Other / unclear	0,72	Used to check product state; general monitoring purpose is most fitting.
Monitoring the system and/or my commits	Other / unclear	0,88	Describes checking the status of BDCs or test loops for a given LMC, which is monitoring system status.
Monitoring the system and/or my commits	Other / unclear	0,67	Finding a revision in a release sounds like checking status/version information rather than troubleshooting.
Monitoring the system and/or my commits	Prediction influences work / decision-making	0,89	Uses CI/baseline status to determine whether the issue remains after a fix.
Monitoring the system and/or my commits	Prediction supports prioritization / overview	0,9	Checks BDC status across products and before/after submitting, mainly for status monitoring and overview.
Monitoring the system and/or my commits	Prediction supports prioritization / overview	0,78	Uses the tool to follow commits and see status, aiding oversight.
Monitoring the system and/or my commits	Prediction supports prioritization / overview	0,86	Checks CI stability and chooses what to test next based on it.
Monitoring the system and/or my commits	Prediction supports prioritization / overview	0,88	Checks flow status and finds the last passing flows for reference.
Monitoring the system and/or my commits	Prediction supports prioritization / overview	0,9	Checks BDC status and uses it to choose the latest UP to use.
Monitoring the system and/or my commits	Prediction supports prioritization / overview	0,82	Uses it to find good UPs and as a starting point for BDC/NBV, supporting overview and tracking.
Monitoring the system and/or my commits	Problem identification / root cause finding	0,9	Tracks merged commits and also looks for the commit that broke the pipeline.
Monitoring the system and/or my commits	Problem identification / root cause finding	0,92	Checks whether deliveries broke something and what package/version includes the delivery.
Monitoring the system and/or my commits	Problem identification / root cause finding	0,93	Tracks commit status and investigates where it fails after delivery.
Monitoring the system and/or my commits	Problem identification / root cause finding	0,93	Monitors commit CI flow and identifies issues in the product flow.

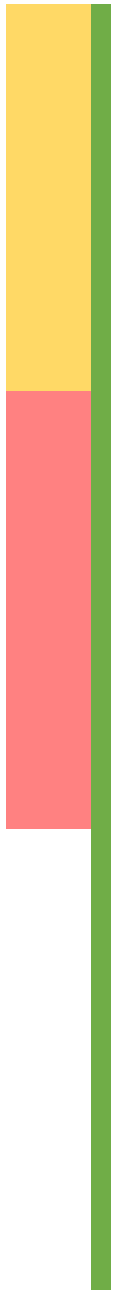
Monitoring the system and/or my commits	Problem identification / root cause finding	0,92	Tracks post-delivery gating failures and whether the commit caused problems.
Monitoring the system and/or my commits	Problem identification / root cause finding	0,78	Following up issues, tracking occurrence frequency, and checking whether problems persist.
Monitoring the system and/or my commits	Problem identification / root cause finding	0,9	Attached to a failed run involving their commit, implying monitoring commit outcomes.
Monitoring the system and/or my commits	Problem identification / root cause finding	0,85	Checks a commit when it gets stuck, implying status monitoring and issue investigation.
Monitoring the system and/or my commits	Problem identification / root cause finding	0,9	Uses it only when something goes wrong with the commit.
Monitoring the system and/or my commits	Problem identification / root cause finding	0,78	Mostly used to follow commits; also mentions finding a specific test loop.
Monitoring the system and/or my commits	Problem identification / root cause finding	0,9	Checks whether changes are included and navigates to job results, mainly monitoring and investigation.
Monitoring the system and/or my commits	Problem identification / root cause finding	0,78	Using it to investigate whether things are down and find issues indicates monitoring plus basic fault investigation.
Monitoring the system and/or my commits	Routine / habitual use	0,92	Uses it to check product state and does so daily as part of regular team meetings.
Monitoring the system and/or my commits	Routine / habitual use	0,9	Describes participation in daily error correction/product care meetings and likely ongoing monitoring work.
Monitoring the system and/or my commits	Step-by-step troubleshooting workflow	0,91	Follows commits through verification levels after review and push.
Monitoring the system and/or my commits	Step-by-step troubleshooting workflow	0,97	Follows commit submission and checks that all gates pass.
Monitoring the system and/or my commits	Step-by-step troubleshooting workflow	0,92	Checks commit passes and retrieves UPs, indicating post-commit tracking workflow.
Monitoring the system and/or my commits	Step-by-step troubleshooting workflow	0,9	Uses it after pushing to track progress and remember what still fails.
Monitoring the system and/or my commits	Workflow pressure / time constraints	0,84	Focuses on their commit and mentions debugging with lower pressure.
Monitoring the system and/or my commits		0,82	Only mentions checking a CI status dashboard.
Monitoring the system and/or my commits		0,98	Tracks commit progress and status across repos, clearly describing ongoing monitoring of commits.
Monitoring the system and/or my commits		0,97	Following changes in CI is direct monitoring of commit flow and status.
Monitoring the system and/or my commits		0,97	Explicitly says monitor commit delivery and flow, fitting commit monitoring.
Monitoring the system and/or my commits		0,95	Following up on commit progress in the flow is ongoing monitoring.
Monitoring the system and/or my commits		0,97	Checking whether a commit reached CL3 is commit tracking/monitoring.
Monitoring the system and/or my commits		0,96	Following up on submitted commits indicates monitoring their status.
Monitoring the system and/or my commits		0,98	Following CI flow is straightforward monitoring of the system.
Monitoring the system and/or my commits		0,97	Following commits to ensure passing and locating their destination version is monitoring.
Monitoring the system and/or my commits		0,96	Checking the commit flow through CI machinery is monitoring commit progress.

Monitoring the system and/or my commits		0,88	Wants to see commit impact and destination date/version, fitting monitoring.
Monitoring the system and/or my commits		0,95	Checks failing loops and which UP contains the commit, indicating monitoring.
Monitoring the system and/or my commits		0,95	Describes following a commit after merging and identifying its upgrade/package location.
Monitoring the system and/or my commits		0,97	Very short statement about tracking a commit.
Monitoring the system and/or my commits		0,98	Explicitly says it is used to check CI flow status during commits.
Monitoring the system and/or my commits		0,96	Repeatedly follows ongoing commits and fixes.
Monitoring the system and/or my commits		0,95	Following released code through CI is direct commit monitoring.
Monitoring the system and/or my commits		0,97	Checks results and package/version numbers, consistent with monitoring commits.
Monitoring the system and/or my commits		0,95	Simply states checking CI job results for uploaded patchsets.
Monitoring the system and/or my commits		0,91	Uses it to look at own commit and follow up progress.
Monitoring the system and/or my commits		0,94	Directly says it is used to track delivery status.
Monitoring the system and/or my commits		0,93	Tracking CFIs on their jobs is monitoring commit-related issues.
Monitoring the system and/or my commits		0,96	Checking assigned CFI is routine monitoring of work items.
Monitoring the system and/or my commits		0,82	Checks BDC status through history, indicating monitoring.
Monitoring the system and/or my commits		0,95	Checks the status of the environment in the repo, a monitoring activity.
Monitoring the system and/or my commits		0,88	Uses it to check job status, which is straightforward monitoring.
Monitoring the system and/or my commits		0,82	Single-word 'commit' likely refers to checking commit status, but is somewhat vague.
Monitoring the system and/or my commits		0,91	Simply says they see BDC status, which is direct monitoring.
Monitoring the system and/or my commits		0,82	Used to monitor status and progress of flows.
Monitoring the system and/or my commits		0,84	Focuses on understanding current stability and inventory status.
Monitoring the system and/or my commits		0,78	Used for finding flow states, which is monitoring system behavior.
Monitoring the system and/or my commits		0,89	Directly states checking the status of BDC flows.
Monitoring the system and/or my commits		0,88	Checks CI flow status, which is straightforward monitoring.
Monitoring the system and/or my commits		0,81	Tracks BDC status, a monitoring activity.
Monitoring the system and/or my commits		0,95	Tracking the status of changes is direct monitoring of work or commits.
Multi-system troubleshooting	Explanation / transparency needed	0,76	Mentions CFI Jira and Kibana together for use across tools, though not detailed.
Multi-system troubleshooting	Manual verification / checking logs	0,97	Uses several tools and logs across systems to diagnose what went wrong.
Multi-system troubleshooting	Manual verification / checking logs	0,9	Investigates logs and the responsible component to locate and fix the issue.
Multi-system troubleshooting	Prediction supports prioritization / overview	0,89	Plans to combine tools for more insights and relate ongoing issues to open TRs.
Multi-system troubleshooting	Prediction supports prioritization / overview	0,67	Comparing legacy behavior to a feature suggests cross-system comparison for troubleshooting.

Monitoring the system and/or my commits	Other / unclear	0,36	Mentions merging or checking another patch, but purpose is too vague to code confidently.
Prediction has limited role	Low trust / manual confirmation needed	0,86	They skip it without deep use, though acknowledge it has some value.
Prediction has limited role	Manual verification / checking logs	0,72	They barely look at it and instead infer from the issue name.
Prediction has limited role	Other / unclear	0,71	Says BDC failures are usually found before reaching BDC, implying limited need to use this process.
Prediction has limited role	Other / unclear	0,74	Uses it only for a specific handover-related task, suggesting a narrow, limited role.
Prediction has limited role	Problem identification / root cause finding	0,88	Someone else already identified the problem, so they usually skip the prediction.
Prediction has limited role		0,94	Says they have not looked at the predictions at all.
Prediction has limited role		0,94	Simply says they did not analyze it.
Prediction has limited role		0,63	Says they already have everything needed, implying no need for the tool.
Prediction has limited role		0,81	Used only once for a specific need, indicating very limited use.
Prediction has limited role		0,82	Usage dropped because another portal exists, implying limited current role.
Prediction has limited role		0,62	Uses a different report instead, suggesting this tool has little or no role in their workflow.
Prediction influences work / decision-making	Low trust / manual confirmation needed	0,82	Prediction trust is stated, and use depends on troubleshooting needs, suggesting it informs decisions.
Prediction influences work / decision-making	Prediction has limited role	0,91	The category changes how far they continue troubleshooting and can lead them to stop early.
Prediction influences work / decision-making	Prediction supports prioritization / overview	0,74	Uses the tool to decide test suites and subflows before risky commits, influencing delivery decisions.
Prediction influences work / decision-making	Problem identification / root cause finding	0,9	Says the prediction gives a hint where to look and helps guide deeper investigation.
Prediction supports prioritization / overview	Monitoring the system and/or my commits	0,7	Emphasizes a good presentation of merged and in-progress commits, suggesting overview use.
Prediction supports prioritization / overview	Monitoring the system and/or my commits	0,88	Seeks status of flow steps and overview of features in the pipeline.
Prediction supports prioritization / overview	Monitoring the system and/or my commits	0,79	Used to see which STPs BDC tests run on, providing an overview for prioritization.
Problem identification / root cause finding	Manual verification / checking logs	0,95	Focuses on identifying the commit that broke a test and checking root cause with downloaded data.
Problem identification / root cause finding	Manual verification / checking logs	0,93	Looks for failing tests, related product impact, and pinpoints root cause.
Problem identification / root cause finding	Manual verification / checking logs	0,93	Checks for issues in tests, a diagnostic and verification activity.
Problem identification / root cause finding	Manual verification / checking logs	0,81	Seeks fault type and responsible party, supporting diagnosis and checking details.

Problem identification / root cause finding	Manual verification / checking logs	0,86	Uses it when failures occur, likely to inspect related logs and issues.
Problem identification / root cause finding	Manual verification / checking logs	0,79	Checking failed test cases is part of diagnosing what went wrong.
Problem identification / root cause finding	Manual verification / checking logs	0,86	Looks for problems and accesses test execution data to investigate issues.
Problem identification / root cause finding	Manual verification / checking logs	0,94	Looks for the exact failing test loop and compares good versus bad cases to diagnose issues.
Problem identification / root cause finding	Manual verification / checking logs	0,95	Analyzing error traces is classic root-cause investigation using diagnostic evidence.
Problem identification / root cause finding	Monitoring the system and/or my commits	0,88	Checks which tests failed after code changes to find issues.
Problem identification / root cause finding	Monitoring the system and/or my commits	0,84	Tracking issues in NBV suggests issue monitoring and diagnosis.
Problem identification / root cause finding	Multi-system troubleshooting	0,94	Mentions troubleshooting, suspected culprit commits, and searching commit ranges for causes.
Problem identification / root cause finding	Multi-system troubleshooting	0,9	Uses it during troubleshooting to understand which revisions were delivered.
Problem identification / root cause finding	Multi-system troubleshooting	0,79	Used when helping investigate an identified issue in their area.
Problem identification / root cause finding	Other / unclear	0,74	Looking for duplicate TRs suggests investigating issues and identifying related problems.
Problem identification / root cause finding	Other / unclear	0,74	Uses it to find obscure parts of the CI system, suggesting investigative lookup.
Problem identification / root cause finding	Prediction has limited role	0,87	Used to gather more data about why something is stuck in CI, indicating diagnosis and cause finding.
Problem identification / root cause finding	Step-by-step troubleshooting workflow	0,92	Looks for failing flows and identifies last passing flows to troubleshoot issues.
Problem identification / root cause finding	Step-by-step troubleshooting workflow	0,86	Reporting issues and resolving them is investigation and troubleshooting work.
Problem identification / root cause finding		0,86	States they try to understand the problem with their commit, which is root-cause oriented.
Problem identification / root cause finding		0,72	Says TRs have been found using CFI, suggesting issue discovery.
Problem identification / root cause finding		0,95	Uses it to quickly identify what a failure may be related to.
Problem identification / root cause finding		0,9	Looking for example failures is aimed at identifying problems.
Problem identification / root cause finding		0,95	Searching for known issues is direct problem identification.
Problem identification / root cause finding		0,88	Only used when asked to investigate a specific CFI.
Problem identification / root cause finding		0,9	Uses it when BDC fails, indicating failure investigation and issue identification.
Routine / habitual use	Monitoring the system and/or my commits	0,93	Explicitly says monitoring is part of the team's daily routine.
Routine / habitual use	Other / unclear	0,55	Mentions seeing it but not using it; unclear whether this is a routine-related non-use.
Routine / habitual use	Prediction influences work / decision-making	0,78	Describes frequent past use for adapting tests during legacy updates, part of regular work.
Routine / habitual use	Problem identification / root cause finding	0,83	Uses it when writing ESID rules to track an issue, a recurring workflow.
Routine / habitual use	Workflow pressure / time constraints	0,86	Frequency depends on development phase, suggesting regular operational use that varies with workload.
Routine / habitual use		0,85	Describes daily use for writing CFIs, indicating routine work.
Routine / habitual use		0,78	Mentions using it once for a specific task; limited but actual use.
Routine / habitual use		0,98	Brief, generic usage statement about submitting code; no clear prediction or troubleshooting context.
Routine / habitual use		0,72	Directly states they simply do not use it.
Step-by-step troubleshooting workflow	Manual verification / checking logs	0,93	Sequential process: check FYC, identify failing test, inspect results, then fix.
Step-by-step troubleshooting workflow	Manual verification / checking logs	0,97	Gives a clear troubleshooting sequence: check CI, logs, fetch ESI, then inspect.

Step-by-step troubleshooting workflow	Prediction supports prioritization / overview	0,77	Describes using automatic test flow to ensure needed tests run and support development workflow.
Step-by-step troubleshooting workflow	Problem identification / root cause finding	0,98	Clearly lists a troubleshooting sequence from radiator to logs, similar issues, rerun, and repeat.
Step-by-step troubleshooting workflow	Problem identification / root cause finding	0,97	Provides a numbered troubleshooting process ending in writing a CFI.
Step-by-step troubleshooting workflow	Problem identification / root cause finding	0,95	Lists a clear sequence for diagnosing the failure and isolating the cause.
Step-by-step troubleshooting workflow	Problem identification / root cause finding	0,87	Describes using it mainly for troubleshooting and investigating failures.
Step-by-step troubleshooting workflow		0,95	Explicitly says it is used when troubleshooting.
This is not my job or responsibility	Other / unclear	0,86	Says other team members track NOSA, implying it is handled by others.
This is not my job or responsibility	Prediction has limited role	0,91	Says it is not directly necessary for their role and a meeting summary is sufficient.
This is not my job or responsibility	Routine / habitual use	0,82	Being lead developer explains role-based involvement more than tool usage.
This is not my job or responsibility	Routine / habitual use	0,9	Says it is not directly necessary for their role.
This is not my job or responsibility		0,98	Directly states the task is not theirs.
This is not my job or responsibility		0,99	States they do not work with CFIs, so the tool is outside their responsibility.
This is not my job or responsibility		0,73	Usage is by other team members, implying it is not their responsibility.
This is not my job or responsibility		0,88	Simply notes their team develops the system, not tool usage.
This is not my job or responsibility		0,74	Says other team members use it, implying it is handled by others.
This is not my job or responsibility		0,96	States they are the lead developer; no indication of using the tool beyond role description.
Workflow pressure / time constraints		0,97	Explicitly cites lack of time to learn or use it properly.
Workflow pressure / time constraints		0,84	Rare use due to slow loading suggests performance-related workflow friction.
Other / unclear	Prediction has limited role	0,58	Uses it only when another source fails; limited fallback use.
Other / unclear	Prediction has limited role	0,62	Compares versions used together, but the exact use is too vague to map confidently to a family.
Other / unclear	Prediction influences work / decision-making	0,74	Says they are not using it yet but may start based on recommendation; current usage is unclear.
Other / unclear	This is not my job or responsibility	0,72	Dismisses use with a rhetorical question; likely irrelevant to their role.
Other / unclear		0,99	No substantive content provided.
Other / unclear		0,98	States they do not use it; no further context about why or how.
Other / unclear		0,61	Uses it only when given a link during issues; usage context remains vague.
Other / unclear		0,99	Single-word nonresponse with no meaningful content.
Other / unclear		0,87	Says they used a different interface, not why or how they use this one.
Other / unclear		0,31	Only states they develop it; no clear use case or behavior is described.
Other / unclear		0,88	Only states usage frequency varies by project phase; no task or purpose described.
Other / unclear		0,86	Describes only variation in frequency by project phase, not the actual use.



Other / unclear	0,99	Phrase is too vague to infer a meaningful code family.
Other / unclear	0,35	Too vague; only says usage depends on number of commits pushed.
Other / unclear	0,2	No substantive text beyond reference to prior answer.
Other / unclear	0,99	Nil provides no substantive information.
Other / unclear	0,72	Generic statement about using it as needed; not specific enough for a clearer family.
Other / unclear	0,62	Mentions unreadable UI and interest in one product, but intent is unclear.
Other / unclear	0,55	Vague statement about needing it for a specific run, without clear usage purpose.
Other / unclear	0,9	Very vague 'Need basis' indicates occasional, unspecified use.
Other / unclear	0,97	Only states usage is very rare, without a clearer purpose.
Other / unclear	0,52	Too vague; only says it is used sometimes when working with a TR.

F

Thematic Analysis of Interview with Flow Guardians

The following pages contain the thematic analysis conducted on the answers from the interviews with three different flow guardians.

FlowGuardian Interviews

Participant	Role	Years at Ericsson	Years in current role	Source	Section	Excerpt	Code 1	Code 2	Code 3	Notes
P1	COIN, Flow Guardian, Flow Management	22	4	ACTA	Anomalies — Expertise	No time. It is not common to have a problem or bug in the problem. Always in the CFI ticket filter	Anomalies not common			
P2	Flow Guardian, Technical coordinator in the flow service program	12	8	ACTA	Anomalies — Expertise	Can notice when something is new. Knows from experience	Notice new things			
P3	Flow Guardian	8	2	ACTA	Anomalies — Expertise	When an environmental is very clearly that novices can't see. BDC took 40 minutes which is not correct.	Experience important			
P1	COIN, Flow Guardian, Flow Management	22	4	Interview	BDC troubleshooting process	Depending on the length of the problem time wise. A bit dependent on the organization. He knows which one it is. Is it an environment or product? Start from BDC report How many times in a row has it failed. If multiple: has it failed in the same way or different. If so restart the BDC if environment: create a cfi ticket for it. If product: Which R state was introduced in it. Will see it in the occurance table	Many different pathways, choice depends on experience			
P2	Flow Guardian, Technical coordinator in the flow service program	12	8	Interview	BDC troubleshooting process	Finding something that is failing vertically or horizontally. Look at the test cases to create new ones for the fails that is not taken care of. Help narrow down where the problem. If we have not seen it before create a new. If it is old then he will check with the design team of create a trouble report to track it on that level. Not a problem hunting tool to solve an issue. It is a trouble shoot tool. Flow guardians pin point the problem so that everyone is not trouble shooting.	Many different pathways, choice depends on experience			
P3	Flow Guardian	8	2	Interview	BDC troubleshooting process	Core of his work. Not the things that are in the SP solution package then there are WP work package. SP becomes somethings for ex shutting down the radios when they are to warm Workflow: Mix of the BDC report and see what the stats say. Try to see if there are any chains in the logs. Check the JCAT link to see the test. Is this a CFI that we are tracking right now, is it old, do we know it. 1. Environment: check in the BDC report to see the solution and see the cloud stuff. Check the BDC one by one and then to dumpire. If there is a fault he look at dumpire which show the test logs. Check the config. What environment, settings, hardware etc.	Core of work. Many different pathways, choice depends on experience			
P1	COIN, Flow Guardian, Flow Management	22	4	ACTA	Big Picture — Cues & strategies	By experience. Recognize aspects of the problem. Can look in MIA in nexus to find the team name	Experience important			
P2	Flow Guardian, Technical coordinator in the flow service program	12	8	ACTA	Big Picture — Cues & strategies	Using CFI prompter or the NoSA database to find a root cause. To find where it was introduced	Find root cause			
P3	Flow Guardian	8	2	ACTA	Big Picture — Cues & strategies	Can recognize the cues in the BDC.	Experience important			
P1	COIN, Flow Guardian, Flow Management	22	4	ACTA	Big Picture — Expertise	Who does the BDC belong to? If this is understood it creates a context	Understand context			
P2	Flow Guardian, Technical coordinator in the flow service program	12	8	ACTA	Big Picture — Expertise	Very important to understand the big picture. It is not important to understand specific things, to solve the problem you have to found the root cause of issues	Find root cause			
P3	Flow Guardian	8	2	ACTA	Big Picture — Expertise	Understanding on what is needed to act on and what is not needed to act on. Knows what BDCs are needed to care of.	Allocate resources depending on importance	Experience matters		
P1	COIN, Flow Guardian, Flow Management	22	4	ACTA	Big Picture — Why difficult	Time consuming if the recognition is not happening	Lack of experience leads to more time spent			
P2	Flow Guardian, Technical coordinator in the flow service program	12	8	ACTA	Big Picture — Why difficult	It takes time to understand what is important. The database is very big and it takes time to understand it	Lack of experience leads to more time spent			
P3	Flow Guardian	8	2	ACTA	Big Picture — Why difficult	Hard to know which BDCs are important and not	Lack of experience leads to more time spent			
P1	COIN, Flow Guardian, Flow Management	22	4	Survey	CATegorizer factors-into-work elaboration	Does look at it but it does not factor into the work. If he sees a problem he can see what category that the problem is. Domain knowledge. Has kinda look away from environment and product. Wants to get what the R state from occurrence instead of the category!!!!! All the information is already in CFI ticket. No % is really needed...	Experience outweighs prediction			
P2	Flow Guardian, Technical coordinator in the flow service program	12	8	Survey	CATegorizer factors-into-work elaboration	Other indicators are more useful. If it is only happening on one STP (equipment) then it is probably an environment fault. That is better	Experience outweighs prediction			
P3	Flow Guardian	8	2	Survey	CATegorizer factors-into-work elaboration	Not to often. Historically it is not reliable. Therefore they have stopped relying on it. It is currently just a gimmic.	Prediction not accurate			
P1	COIN, Flow Guardian, Flow Management	22	4	Survey	CFI WEB usage elaboration	Use it mainly to see what occurrences that is there. When did this first occur. When was the last occurrence.	Gather context info			
P2	Flow Guardian, Technical coordinator in the flow service program	12	8	Survey	CFI WEB usage elaboration	Same as JIRA and used to create filters rather then to find info.	To write CFI tickets			
P3	Flow Guardian	8	2	Survey	CFI WEB usage elaboration	For writing CFI and monitoring existing CFI. To make sure that they can track the issues in the CI flow.	To write CFI tickets	Monitoring		
P1	COIN, Flow Guardian, Flow Management	22	4	Survey	CI Flow Impediment usage elaboration	Use it in the same way as the last it just depends on where he navigates from. The CFI interface is quicker and therefore better. The cfi interface is better. He prefers the occurence graph in the jira board. Would like it in the CFI interface.	Gather context info			
P2	Flow Guardian, Technical coordinator in the flow service program	12	8	Survey	CI Flow Impediment usage elaboration	Creating tickets, finding where things have been seen. Comment to developers to communicate the status	To write CFI tickets	Communication		
P3	Flow Guardian	8	2	Survey	CI Flow Impediment usage elaboration	Same purposes. For monitoring to make sure that he can correlate. Try to use the CFI web more since this is the new one. But they kinda offer the same. The difference is about familiarity, new functionallity. Trying to mimic the older one as much as possible	Communication			
P1	COIN, Flow Guardian, Flow Management	22	4	Survey	Close CFI tickets elaboration	When i get the first reminder. Weekly or monthly. Tries to be better but you forget. Gets notifications from JIRA. Ones in a while.	Reminders needed			
P2	Flow Guardian, Technical coordinator in the flow service program	12	8	Survey	Close CFI tickets elaboration	couple a week	A few a week			
P3	Flow Guardian	8	2	Survey	Close CFI tickets elaboration	Every 2 weeks maybe. or one month it differs.	Every few weeks			
P1	COIN, Flow Guardian, Flow Management	22	4	Survey	Create CFI tickets elaboration	Only create CFI if the hits in the database is spread out or old.	When necessary			
P2	Flow Guardian, Technical coordinator in the flow service program	12	8	Survey	Create CFI tickets elaboration	couple a week	A few a week			
P3	Flow Guardian	8	2	Survey	Create CFI tickets elaboration	Couple a times a week. Mostly because to track different fault that he find that match BDC faults. Make one CFI ticket per fault.	A few a week	When necessary		
P1	COIN, Flow Guardian, Flow Management	22	4	ACTA	Equipment Difficulties — Expertise	No time				
P2	Flow Guardian, Technical coordinator in the flow service program	12	8	ACTA	Equipment Difficulties — Expertise	If there is a bug then they report it.	Report bugs			
P3	Flow Guardian	8	2	ACTA	Equipment Difficulties — Expertise	Judgement can tell when tools say that something is good. That does not always mean that nothing is wrong with it.	Tools can be wrong	Experience matters		
P1	COIN, Flow Guardian, Flow Management	22	4	Survey	FYC usage elaboration	When I see problems in BDC and NBC. Then i need to see what has been delivered. I manually monitor the SBC state and see what the problem has ended up as. This is where the commit get the R state (revision)	To track faulty deliveries	Monitoring		
P2	Flow Guardian, Technical coordinator in the flow service program	12	8	Survey	FYC usage elaboration	Someware between daily and weekly. Follow up on changes. Either his own or others. Share what to follow	A few a week	Monitoring		
P3	Flow Guardian	8	2	Survey	FYC usage elaboration	Monitoring commits when they go through the whole flow. To look at CL2 and CL3. Monitor the whole chain. For his own commits and others commits	Monitoring	To track faulty deliveries		
P1	COIN, Flow Guardian, Flow Management	22	4	ACTA	Job Smarts — Cues & strategies	If things connect in a clear flow from the log and from the commit message and the problem matches the commit message.	If things match	Experience matters		
P1	COIN, Flow Guardian, Flow Management	22	4	ACTA	Job Smarts — Expertise	If he sees that the problem is easy to solve then there is no need to create a CFI ticket. Just solve it rather then creating a ticket as it is super easy to solve. Unnecessary to create ticket as it takes more time then it gives good.	If easy fix, don't create CFI ticket for it	Experience matters		
P2	Flow Guardian, Technical coordinator in the flow service program	12	8	ACTA	Job Smarts — Expertise	Use the tools that are available for example CFI ticket creator. No clear shortcuts.	Use available tools			
P3	Flow Guardian	8	2	ACTA	Job Smarts — Expertise	The programs are open enough so no shortcuts. Using another tool to speed up the process. Dumpire is slow but useful, Use other tools like integrating with TGF to directly interact with the notebooks. Harder but if you know it then it can be faster	Use available tools	Experience matters		
P1	COIN, Flow Guardian, Flow Management	22	4	ACTA	Job Smarts — Why difficult	Experience for investigation	Experience matters			
P3	Flow Guardian	8	2	ACTA	Job Smarts — Why difficult	TGF is not as straight forward as dumpire	Tools complicated			

THEMES	NR OF CODES IN THEME
Experience - Experience matters greatly in the work, and is often more important than what the tools say.	33
Monitoring - Much of the system is based on monitoring, being able to monitor existing issues for new occurences, and monitor the CI flow for potential issues.	10
Tools - Tools are sometimes not accurate. This is especially true for the CATegorizer predictions, which are wrong often enough that they are normally disregarded.	3
Ideas - Suggested improvements regarding expanded functionality of the CATegorizer	5
Context - Gather information through the tools to gain an understanding of the context	8
Frequency of actions - How often the users take certain actions in the systems. Varies greatly depending on the tool in question.	8
Communication - parts of the system feature in-system communication options (the comments in Jira for example). This is an important avenue of communication for the users.	3
Efficiency of resources - use tools and allocate resources in the most efficient way possible. This is largely based off experience.	6
Fatigue - how mental fatigue can affect the experience working in the system	4
Working with CFI tickets	4
Labeling	2

P1	COIN, Flow Guardian, Flow Management	22	4	Survey	Label problem type elaboration	Always but sometimes selects multiple. Never leave it empty. Environment is easy to say but test or product is harder to determine. Sometimes click both. He does it when writing it. Does not know if it is important	Never leave empty	
P2	Flow Guardian, Technical coordinator in the flow service program	12	8	Survey	Label problem type elaboration	couple a week (always)	A few a week	Never leave empty
P3	Flow Guardian	8	2	Survey	Label problem type elaboration	There are a lot of CFI that you create where you don't know what fault it is when creating it. If he find out what it is he does not go back and label it even if he think that it would be good and should do it. When closing it he always try to have a link of what has tried to fix it, ex TR etc.	Doesn't go back to label retroactively	
P1	COIN, Flow Guardian, Flow Management	22	4	Survey	NoSA usage elaboration	More then FYC. This is the main tool that he uses. Look into the current state. No history. Get a real time view of what fails.	Monitoring	The main tool
P2	Flow Guardian, Technical coordinator in the flow service program	12	8	Survey	NoSA usage elaboration	Getting an overall status view and find out where problems are. Looked at each morning meeting daily.	Monitoring	
P3	Flow Guardian	8	2	Survey	NoSA usage elaboration	Monitoring BDCs that COIN take care of. Fault finding. To make sure that BDC work correctly	Monitoring	
P1	COIN, Flow Guardian, Flow Management	22	4	ACTA	Noticing — Cues & strategies	Sees recurring names and problems and ID's	See recurring names	Experience matters
P3	Flow Guardian	8	2	ACTA	Noticing — Cues & strategies	Knows the tests	Experience matters	
P1	COIN, Flow Guardian, Flow Management	22	4	ACTA	Noticing — Expertise	Where does a BDC belong? Same as before.		
P2	Flow Guardian, Technical coordinator in the flow service program	12	8	ACTA	Noticing — Expertise	Does happen. There was a feature called hardware sensitive install where software was only downloaded for the certain execution. If the test was not written in the right way then there would be an error even if the problem was not incorrect. It just lacked the consideration of the HSI. He can see this directly when he see it.	Know common issues	Experience matters
P3	Flow Guardian	8	2	ACTA	Noticing — Expertise	Focusing on different things and the FG might have worked with a type of test before and knows what tests are causing different issues	Experience matters	
P1	COIN, Flow Guardian, Flow Management	22	4	ACTA	Noticing — Why difficult	Experience	Experience matters	
P2	Flow Guardian, Technical coordinator in the flow service program	12	8	ACTA	Noticing — Why difficult	It was not clear that this was the problem. He has worked with it and solved it.	Problem solving	
P3	Flow Guardian	8	2	ACTA	Noticing — Why difficult	New people don't know which ones cause which issue. Knowledge based on experience	Experience matters	
P1	COIN, Flow Guardian, Flow Management	22	4	ACTA	Opportunities / Improvising — Cues & strategies	Sees that CFI ticket is close to what he wants to do.	Knows similar CFI tickets	Experience matters
P1	COIN, Flow Guardian, Flow Management	22	4	ACTA	Opportunities / Improvising — Expertise	If you have a CFI that someone else has created: sometimes change someone else CFI instead of a new one if they are close to what he wants it to do. Only if they have time. FG would like to do this	Knows similar CFI tickets	Change existing ticket if close to what he wants
P2	Flow Guardian, Technical coordinator in the flow service program	12	8	ACTA	Opportunities / Improvising — Expertise	There was one case where there was a lot of things connected to a service. When the service was down it would give an error. The problem was that the router was bad.	Single point of failure outside the scope of the system	
P3	Flow Guardian	8	2	ACTA	Opportunities / Improvising — Expertise	Setting up his own testing flows since he knows the system and what to use. Novinces don't know what resources to use without causing issues. Appoligieses and then keep going. They did not like doing a certain tool to do a test. The tool was used for CL3 and upwards. Instead find tools that are not in much demand. Knows which config that can run the test. Know he knows that through "test and fail". Check TG web to check the test failure	Personalized testing flows	Experience matters
P3	Flow Guardian	8	2	ACTA	Opportunities / Improvising — Why difficult	Has to do the dirty work and get screamed at to learn which resources that could be used	Has to learn what resources are available to use	Experience matters
P1	COIN, Flow Guardian, Flow Management	22	4	Survey	Other CATegorizer thoughts	Dont want to have extra steps. But if it is seamless to make it better and contribute then there is no problem. It is easy to forget to close it, everything should be automated will be better.	No added steps or workload	Automation is better	Change what the CATegorizer does
P2	Flow Guardian, Technical coordinator in the flow service program	12	8	Survey	Other CATegorizer thoughts	It would be better to have it say if it is gateable or not. Meaning if the CI can throw out a particilar thing to recover it self. To improve it should say gateble or not instead of what type it is. Instead of CATegorizer there could be something that pinpoint the problem	Gateable or not	CATegorizer does
P3	Flow Guardian	8	2	Survey	Other CATegorizer thoughts	Gateing: could be something that would add values. Would be more valueable then just showing the issue type. Typically then can see it fairly easy what type it is when looking in logs.	Gateable or not	CATegorizer does
P1	COIN, Flow Guardian, Flow Management	22	4	ACTA	Past / Future — Cues & strategies	Where was the problem found. Is it BDC? Or NBC?	Find point of failure	
P2	Flow Guardian, Technical coordinator in the flow service program	12	8	ACTA	Past / Future — Cues & strategies	Many different things. Error log, test case execution log, NoSA database. Often NoSA database.	Find point of failure	
P3	Flow Guardian	8	2	ACTA	Past / Future — Cues & strategies	Looking at the tags CFIs. Knows which one is one can recognize it. CFI tickets are numbered incrementally.	Experience matters	
P1	COIN, Flow Guardian, Flow Management	22	4	ACTA	Past / Future — Expertise	Gut feeling from experience. Kinda lika xrays that doctors see	Experience matters	
P2	Flow Guardian, Technical coordinator in the flow service program	12	8	ACTA	Past / Future — Expertise	Knows from previous instances to understand a problem.	Experience matters	
P3	Flow Guardian	8	2	ACTA	Past / Future — Expertise	Know how to fix an issue based on familiar CFI tickets which have been seen before.	Experience matters	
P1	COIN, Flow Guardian, Flow Management	22	4	ACTA	Past / Future — Why difficult	Hard to translate problem into CFI ticket. Identify what the problem is that should be filtered away	Hard to translate into filters	
P2	Flow Guardian, Technical coordinator in the flow service program	12	8	ACTA	Past / Future — Why difficult	It is just about experience. Your have to work with it to recognize it.	Experience matters	
P3	Flow Guardian	8	2	ACTA	Past / Future — Why difficult	Is completely based on recognizeability which takes time	Experience matters	
P1	COIN, Flow Guardian, Flow Management	22	4	Survey	RAN CI Radiator usage elaboration	I dont chech BDC here. I check NBC here.	Check NBC in Radiator	
P2	Flow Guardian, Technical coordinator in the flow service program	12	8	Survey	RAN CI Radiator usage elaboration	Only for checking UP builds. No flow state of anything.	Check UP builds	
P3	Flow Guardian	8	2	Survey	RAN CI Radiator usage elaboration	Monitoring the CL3 and UPs. Can see what revisions has made it to the next step in CI flow. Also able to check the BDC when the NoSA is down. Not that often though.	Monitoring	Check UP builds
P1	COIN, Flow Guardian, Flow Management	22	4	ACTA	Self Monitoring — Cues & strategies	If he is mentally tired. Might come from having to explain the problem and sometimes they are not compliant.	Mental fatigue matters	
P1	COIN, Flow Guardian, Flow Management	22	4	ACTA	Self Monitoring — Expertise	Yes they are aware and will reduce time for long term performance if he is tired.	Mental fatigue matters	
P2	Flow Guardian, Technical coordinator in the flow service program	12	8	ACTA	Self Monitoring — Expertise	Would take a break if tired. If he cant solve a problem then he would just not care about it.	Mental fatigue matters	
P3	Flow Guardian	8	2	ACTA	Self Monitoring — Expertise	There was one time when he has a specific TR that had to be quick. Went through the whole system to make sure that everything was working correctly. Felt like he was in a circle. Nothing got solved first. Had to stop using resources for individual issues. Started running them parallel which fastened the work. Well-being: If stressed then hand of keyboard and walk away to take time. Hard to see when stressed. Worked together on similar problems which helps to handle stress since they can tell each other when they need a pause.	Mental fatigue matters	

G

Estimated Implementation Cost and Value of New Features

The pages below show an estimation of the cost and value of implementing different features into the existing system. The features was imagined during different brainstorming sessions and the cost estimated by the researchers and stakeholders.

Interface	UI Element	Feature	Description	Desired effect	Req ID	Guideline ID	Improvement score (Total clickthroughs per implementation day spent)	Users reached weekly through Interface* *estimation based on COIN survey responses (total 60), extrapolated to full dev team (total ~900)	Users reached from FYC (If FYC change is implemented)	% Users interacting (0-1)	Clickthroughs (Users reached / % users interacting)	Frequency weekly	Total clickthroughs per week	Cost (Days)
CFI Web	Dashboard	Distribution of latest predictions	A graph showing the distribution of the latest predictions - Product, Environment, Test - for the selected time period	To give users a better understanding of the fact that the CATegorizer is an active tool that they all contribute to collectively, which will lead to increased engagement and interaction with it across all interfaces	Effect R1, Use R3, Interaction R2, Elem R1	Effect G1, Effect G2, Use G1, Arch G2, Arch G3, Interaction G2, Interaction G5, Elem G4	19,33	72	160	0,25	58	1	58	3
CFI Web	Dashboard	Percentage correct predictions	A graph showing the percentage of correct guesses for the selected time period	-||-	Effect R2, Elem R1	Effect G2, Effect G3, Use G1, Arch G1, Arch G3, Interaction G3, Interaction G5	38,67	72	160	0,5	116	1	116	3
CFI Web	Dashboard	Average confidence of predictions	Percentage showing average confidence in predictions	-||-	Interaction R2, Arch R1, Elem R1	Effect G2, Use G1, Interaction G1, Interaction G6, Elem G1, Elem G4	38,67	72	160	0,5	116	1	116	3
CFI Web	Dashboard	Progress bar; How many new training data points (labeled closed tickets)	Part of the "Community Ownership" idea, shows that the users have direct effect on the performance of the model through their	-||-	Effect R1, Effect R4, Interaction R4	Effect G3, Use G2, Arch G3, Interaction G3, Interaction G4	7,67	72	160	0,2	46	0,5	23	3
CFI Web	CFI Web Details	Yellow warning triangle, yellow underline	Shown when Problem Type is set to Undefined. When hovered, tooltip says "Please change from Undefined before closing the ticket"	To visually indicate that setting/leaving the problem type as "Undefined" is not good practice, while leaving it up to the developer to decide (sometimes the problem type is	Effect R1, Effect R3, Use R2, Arch R1, Interaction R1, Interaction R5, Elem R1, Elem R4	Effect G1, Use G2, Use G4, Arch G1, Interaction G2, Interaction G4, Elem G1, Elem G3	78,00	72	320	0,2	78	2	156	2
CFI Web	CFI Web Details	CATegorizer AIX	Add the CATegorizer explanation (as shown in Jira board) to the information displayed in the CFI Web details	Shows that the CATegorizer produces more than just predictions. Can potentially provide insight for troubleshooting as	Use R1, Arch R2, Arch R4, Arch R5, Elem R2, Elem R3	Effect G2, Use G1, Use G4, Arch G1, Interaction G1, Elem G4	13,00	72	320	0,1	39	1	39	3
CFI Web	CFI Web Details	Prediction confirmation button	Add a simple yes/no button next to the prediction percentage in the details view, when pressed it sets the problem type to that of the prediction.	By making it easier to provide feedback, engagement will increase	Interaction R3, Effect R4, Arch R3	Effect G1, Effect G3, Use G2, Use G3, Use G4, Arch G2, Arch G4, Interaction G2, Interaction G3, Interaction G4, Interaction G6, Elem G1, Elem G2	65,33	72	320	0,25	98	2	196	3
CFI Web	CFI Web Homepage	Link to the CATegorizer dashboard	Add a simple link to where the CATegorizer dashboard is located (like the tools in the current homepage of CFI Web)	Will increase traffic to the new dashboard by providing another entry point	Effect R3, Use R4, Arch R5, Elem R2, Elem R4	Effect G1, Effect G2, Use G1, Arch G1, Arch G3, Interaction G4, Elem G3	14,00	72	0	0,2	14	1	14	1

Follow Your Commit	Additional Link field	Add CATegorizer prediction	Show CFIs tagged at a failed BDC-level test, with CATegorizer predictions.	This will help the user to get a quick view of why the commit did not pass the BDC.	Effect R1, Effect R3, Effect R4, Use R1, Arch R1, Arch R4, Interaction R1, Elem R2	Effect G1, Effect G2, Use G4, Arch G1, Arch G2, Interaction G4, Elem G4	137,14	800	0	0,4	320	3	960	7
NoSA	Expanding CFI label	Add expanding element to CFI label	In Nosa, when a prediction is made by the CATegorizer, the prediction percentage is shown next to the CFI label. This would make that percentage clickable, which would expand a box below with the full CATegorizer prediction and explanation. Maybe even the previous occurences.	Providing more CFI info directly in can increase engagement with CFIs, and by extension the CATegorizer	Use R1, Use R2, Arch R1, Arch R2, Arch R3, Arch R5, Interaction R1, Interaction R5, Elem R1, Elem R3, Elem R5	Effect G1, Effect G2, Effect G3, Use G1, Use G2, Use G3, Use G4, Arch G1, Arch G2, Arch G4, Interaction G1, Interaction G2, Interaction G4, Interaction G5, Interaction G6, Elem G1, Elem G2, Elem G4	11,33	189	40	0,15	34	1	34	3

H

N-gram and Frequency Heatmaps

These are the bigram and trigram heatmaps produced as the initial step of the thematic analysis as explained in Section 5.1.5.

Figure H.1: Raw and normalized bigram heatmaps for CFI Web and Follow Your Commit

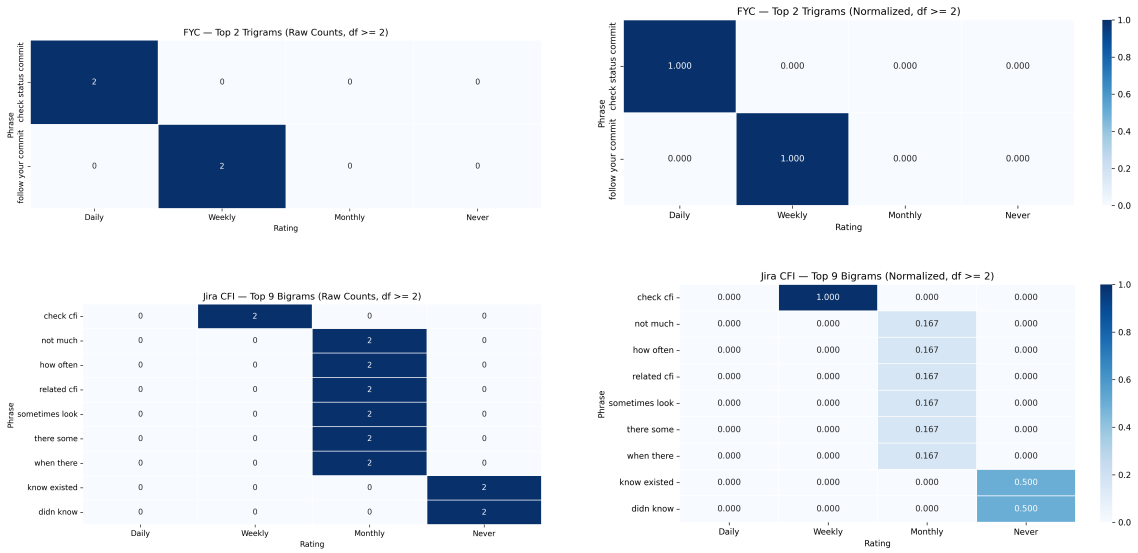


Figure H.2: Raw and normalized n-gram heatmaps for Follow Your Commit trigrams and Jira CFI bigrams

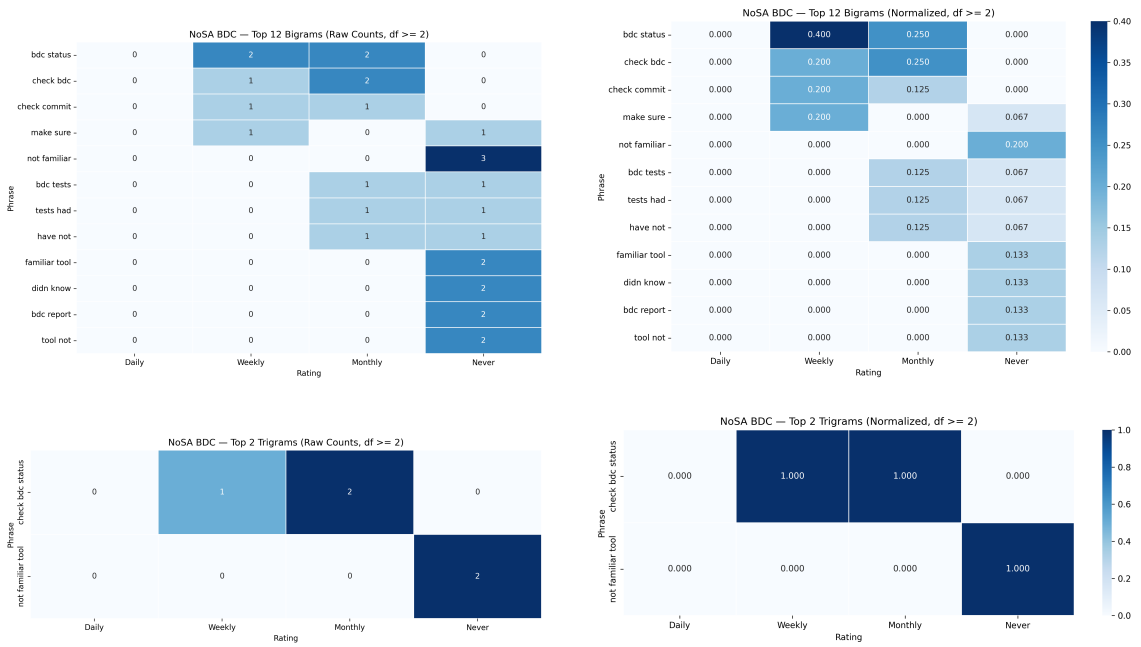


Figure H.3: Raw and normalized bigram and trigram heatmaps for NoSA BDC Report

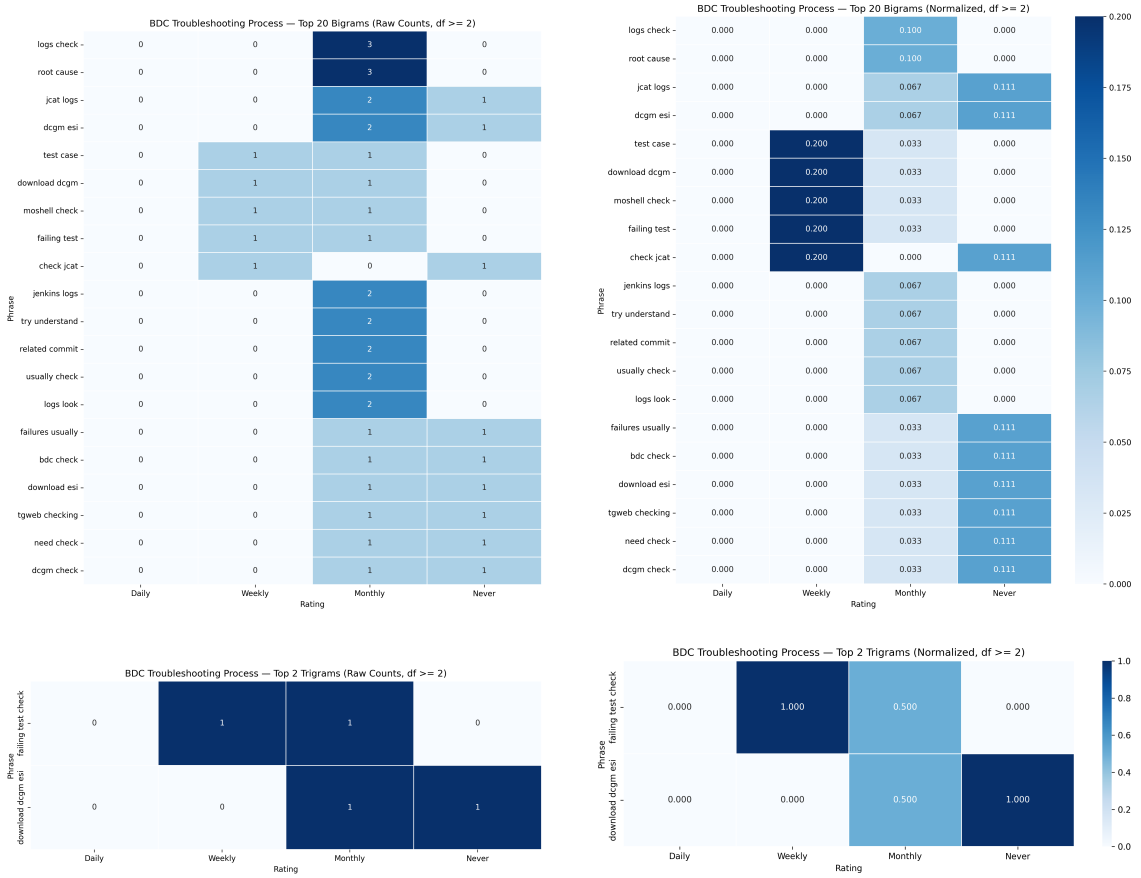


Figure H.4: Raw and normalized bigram and trigram heatmaps for grouped survey questions Q15 from Q14

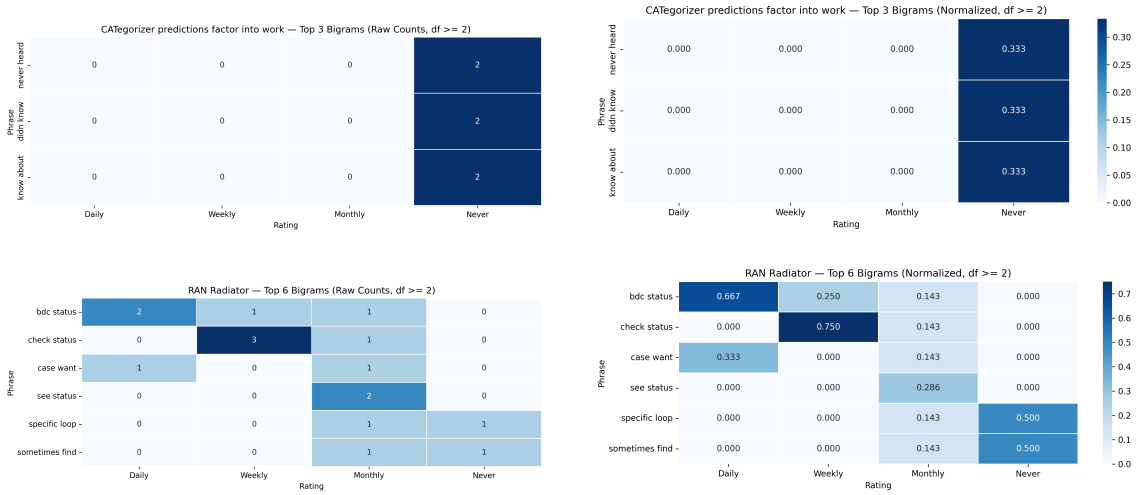


Figure H.5: Raw and normalized bigram heatmaps for grouped survey questions Q17 from Q16 and RAN CI Radiator

I

Cognitive Walkthrough Questions

The cognitive walkthrough performed during the final evaluation consisted of six tasks regarding the CATegorizer dashboard and one regarding the CFI Web changes.

The six tasks regarding the dashboard were:

1. Task 1: Open the dashboard. What do you think its purpose is?
   Follow-up:
   - What catches your attention first?
   - Is the purpose clear?
   - Is anything confusing?

2. Task 2: Find information about the CATegorizer's current performance.
   Follow-up:
   - Where did you look first?
   - How do you interpret this information?
   - Is it enough to understand if the model is performing well?

3. Task 3: Find information about the CATegorizer's accuracy across different Problem Types.
   Follow-up:
   - Where did you look first?
   - How do you interpret this information?

4. Task 4: Find out how users can contribute to improving the CATegorizer.
   Follow-up:
   - Was the information where you expected to find it?
   - Was it a clear answer?

5. Task 5: Find the accuracy of ticket-predictions related to your own organization, within the last year.
   Follow-up:
   - Was it easy/hard to find?
   - Would this view feel more relevant to you than the previous one?

6. Task 6: Imagine that you are working with or closing a CFI ticket. Use the dashboard to decide whether anything here would help you in that situation. Follow-up:
   - At what point in your workflow would you use this dashboard, if at all?
   - Would it feel like support or extra work?
   - Would it make you more likely to label tickets?

The CFI Web task was

1. Task 1: Open CFI Web. Find CFI-79616. Explore the details-panel. Follow-up:
   - What do you think about the Save button?
   - Change it to Undefined. What do you think about the warning triangle?