Applying Prototyping and Exploratory Testing to Ensure Software Quality in an Information System for Power Tampering Detection: An Experience Report

Sabryna Araujo[1][a], Ramille Santana[1][b], Joana Silva[1][c], Arthur Passos[1,2][d], Matheus Menezes[3][e], Felipe Feyh[3][f], Carlos Moura[3][g], Lucas Pinheiro[3][h], Auriane Santos[3][i], Aristofanes Silva[1,2][j], João Dallyson[1,2][k], Italo Francyles[1,2][l] and Luis Rivero[1,2][m]

[1]*Núcleo de Computação Aplicada (NCA), Federal University of Maranhão (UFMA), São Luís, MA, Brazil*

[2]*Programa de Pós-Graduação em Ciência da Computação (PPGCC), Federal University of Maranhão (UFMA), São Luís, MA, Brazil*

[3]*Instituto de Ciência e Tecnologia Grupo Equatorial, São Luís, MA, Brazil*

{sabryna.ra, ramille.rs, silva.joana, arthur.passos}@discente.ufma.br, {matheus.menezes, felipe.feyh}@eqtlab.com.br,

Keywords: Exploratory Testing, Prototype, Software Quality, Software Engineering.

Abstract: This paper presents an experience report on the application of exploratory testing in the development of a system aimed at detecting illegal connections in the electricity supply, a critical problem that causes financial losses and compromises the safety and efficiency of power grids. The research utilized a high-fidelity prototype developed in the Figma tool as a basis for planning and executing tests, allowing the identification of functional and usability defects in an agile and collaborative manner. The adopted methodology involved the use of iterative meetings for continuous validation, ensuring alignment between requirements and implementation. During the process, 49 defects were recorded and categorized, enabling significant system improvements and ensuring higher quality in the final product. The results highlight the effectiveness of integrating prototypes and exploratory testing to reduce validation time, identify critical issues, and promote team alignment. As future work, it is proposed to expand the system's prioritization criteria and conduct user tests in real scenarios. This study contributes to the literature by reinforcing the role of agile methodologies and modern testing techniques in the development of robust and effective technological solutions.

1 INTRODUCTION

The practice of makeshift solutions, characterized by improvised solutions to technical or operational problems, is a phenomenon widely observed in Brazil and in several other developing countries. These practices generally emerge in scenarios of financial resource constraints or insufficient infrastructure and are often associated with improvisations in electrical systems, hydraulic systems, or even technological devices. Furthermore, with the increase in the number of households, energy companies face the challenge of managing a growing customer base (Vidinich and Nery, 2009). Studies indicate that these solutions, despite being creative, can pose significant risks to both the safety and performance of critical systems.

In this context, a specific module for the automatic prioritization of makeshift solutions was developed, which uses the Internal Rate of Return (IRR) as the main evaluation metric. The system combines data analysis and managerial reports to support operational

[a] https://orcid.org/0009-0001-5460-1261
[b] https://orcid.org/0009-0008-4071-8064
[c] https://orcid.org/0009-0004-2216-357X
[d] https://orcid.org/0000-0002-2823-3645
[e] https://orcid.org/0000-0001-8676-1131
[f] https://orcid.org/0009-0008-3593-9225
[g] https://orcid.org/0009-0005-8552-6136
[h] https://orcid.org/0000-0002-4641-3703
[i] https://orcid.org/0009-0003-3873-4990
[j] https://orcid.org/0000-0003-0423-2514
[k] https://orcid.org/0000-0001-7013-9700
[l] https://orcid.org/0000-0002-2041-7538
[m] https://orcid.org/0000-0001-6008-6537

Araujo, S., Santana, R., Silva, J., Passos, A., Menezes, M., Feyh, F., Moura, C., Pinheiro, L., Santos, A., Silva, A., Dallyson, J., Francyles, I. and Rivero, L.
Applying Prototyping and Exploratory Testing to Ensure Software Quality in an Information System for Power Tampering Detection: An Experience Report.
DOI: 10.5220/0013483000003929
Paper published under CC license (CC BY-NC-ND 4.0)
In *Proceedings of the 27th International Conference on Enterprise Information Systems (ICEIS 2025) - Volume 2*, pages 380-387
ISBN: 978-989-758-749-8; ISSN: 2184-4992
Proceedings Copyright © 2025 by SCITEPRESS – Science and Technology Publications, Lda.

teams in identifying critical points, allowing companies to prioritize interventions with higher economic and strategic returns. Additionally, the module helps prioritize corrective actions with the best cost-benefit ratio for the company, optimizing available resources and minimizing the negative impacts of irregularities.

The goal of this paper is to report the experience of applying exploratory testing in the development of a system aimed at detecting makeshift solutions in the electricity supply. These makeshift solutions, known as irregular electrical connections, represent a serious problem for power distributors, causing financial losses, increasing the risk of accidents, and compromising the efficiency of the electrical grid. To address this challenge, it was necessary to invest in an approach that ensured software quality, from the prototyping phase to testing, using best development and validation practices.

During the process, prototyping played an essential role, serving not only as a tool for the initial planning of the system but also as a practical documentation tool for testing. The prototype guided the development and application of exploratory tests, enabling the identification of functional and usability issues even before the final implementation. Furthermore, this approach ensured that the requirements were continuously reviewed and improved, reinforcing the alignment between the development team and the project's objectives.

2 RELATED WORK

2.1 Exploratory Testing and Prototyping

Exploratory testing has emerged as a complementary approach to automation, leveraging the tester's knowledge to identify problems and propose improvements in developing systems. Unlike automated testing, which follows predefined scripts, exploratory testing allows for the dynamic creation of test scenarios, increasing the likelihood of discovering unexpected defects.

The study (Silva et al., 2024) employed design thinking and high-fidelity prototypes developed in Figma to create a mobile application aimed at disseminating scientific knowledge. During prototyping, a panel of judges evaluated the prototype's usability using the SUS (System Usability Scale), achieving a satisfaction score above 85%. However, validation took approximately 60 days, including a second round of adjustments for items that did not meet the minimum content validity index (IVC over 0.800). This approach, although detailed, proved to be limited in agility and identifying functional defects. In contrast, the method proposed in our work prioritizes agility and direct collaboration with the team. We used the prototype as a central "oracle" to identify and classify functional and usability defects in just 2-3 hours of team meetings. This approach not only drastically reduced the time needed for validation but also enabled faster and more accurate defect detection. Additionally, our methodology included immediate defect classification, facilitating prioritization and allowing the team to promptly address the identified issues.

The study (Yu, 2018) presented an agile exploratory testing model that integrates manual and automated methods to evaluate functionalities, regression, and acceptance. The model was implemented in a university system through four iterative cycles, where tests were performed in distinct stages by two separate groups. Each cycle required preparatory tests, pre-tests, functionality, regression, and acceptance testing, resulting in an extensive and fragmented process. Dividing tasks between groups was effective in detecting defects, especially in the graphical user interface (GUI), but the time required to coordinate and integrate the results made the process longer. Our work simplifies and accelerates the process by bringing the entire team together in a single meeting, enabling defects to be identified, categorized, and prioritized collaboratively and immediately. This approach eliminates the need for long and separate cycles, significantly reducing time and improving team alignment efficiency. While Yu presents a robust model, it does not explore the possibility of rapid and direct integration that our methodology offers, highlighting the difference in focus and impact on productivity.

The paper (Fulcini and Ardito, 2022) explored the use of gamification in exploratory GUI (Graphical User Interface) testing through a prototype integrated with the Scout tool. This prototype enabled the execution of manual tests on web applications, enriching the system interface with elements such as highlighted clickable widgets, coverage metrics, and visual feedback for testers. The main objective was to assess how gamification could increase tester engagement and efficiency, promoting greater interface element coverage and better user experience. Although the prototype was essential for applying the tests, it was not used as a reference to identify or classify system defects. Moreover, the adopted method relied on separate sessions with distinct groups of testers, resulting in a more fragmented process. The study also does not address how the defects found could be prioritized or corrected collaboratively, nor does it present strategies to integrate the obtained results directly into system development.

(Azevedo and Castro, 2022) developed a prototype for a mobile application aimed at açaí shops, using Figma to structure and simulate functionalities and design. The work employed UML diagrams and validated the prototype through the SUS questionnaire, applied to 28 participants, highlighting ease of use (89.3%) and high aesthetic acceptance (82.2%). The focus was on pre-implementation validation, promoting communication between teams. However, the prototype was only used for aesthetic and functional validation, without exploring its potential as a guide for defect detection during implementation. Strategies for identifying or correcting functional failures were not addressed, nor was the use of collaborative processes to integrate the prototype into development. Differently, our work uses the prototype as a central reference to guide defect detection and prioritization in quick team meetings. This approach broadens the prototype's role, ensuring functional and technical alignment while accelerating the identification and correction of failures.

2.2 Comparison with Other Agile Testing Methods

Agile methodologies incorporate various testing approaches that aim to ensure software quality through iterative and incremental development. Among them, *Test-Driven Development (TDD)* focuses on writing tests before implementing the actual code, enforcing strict validation at an early stage (Erdogmus et al., 2005). While TDD increases code reliability, it may not capture usability or exploratory aspects effectively. *Behavior-Driven Development (BDD)*, on the other hand, emphasizes user stories and specifications in natural language, facilitating communication between stakeholders (Wynne and Hellesoy, 2012). However, BDD heavily relies on predefined scenarios, potentially missing unexpected defects. *Scrum Testing*, which integrates testing within sprints, supports continuous validation but often requires automated test cases to be effective (Crispin and Gregory, 2009).

In contrast, our approach leverages *prototyping and exploratory testing*, allowing testers to dynamically explore the system, uncovering both functional and usability issues early in the development cycle. This strategy enables rapid defect identification without being constrained by predefined test scripts, making it highly effective for agile environments requiring quick adaptability and user-driven validation.

While some studies discuss using interactive prototypes for usability and design validation, few explore their application as a basis for exploratory testing in agile projects. This paper contributes to the literature by reporting the experience of using interactive prototypes as a central reference in exploratory testing in the context of developing a prioritization system for power distribution companies. In the following sections, we present the development context, testing process, and lessons learned from this approach.

3 METHODOLOGY

3.1 Project Context

The electricity market is fundamental to the functioning of public and private services, and solving problems that impact its quality is essential. Combating irregular energy connections represents a significant challenge for companies in the sector, as these practices result in financial losses, compromise supply quality, increase accident risks, and affect the sustainability of the service. In this context, the Applied Computing Group (Núcleo de Computação Aplicada - NCA) group at Federal University of Maranhão (UFMA), Brazil, develops innovative software projects in partnership with companies. The group developed a prioritization system focused on areas without formal connection to the power grid, considering aspects such as cost, financial return, and strategic factors. The goal is to improve data analysis and facilitate decision-making regarding these areas, which, by operating outside formal contracts, generate financial impacts and risks to users' health and safety. The solution, which combines data analysis with managerial reports, enables operational teams to identify critical points and prioritize corrective actions that provide the best economic and strategic returns for the company. With this in mind, the following deployment diagram (Figure 1) illustrates how the selected technologies interact within the system's context.

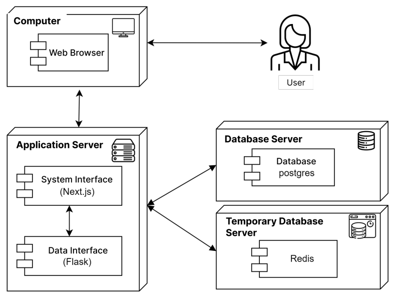


Figure 1: Deployment Diagram of the System.

3.2 Prototyping

According to (Sommerville, 2011), a prototype is an initial version of a system used to demonstrate concepts, explore design alternatives, and evaluate problems and potential solutions. The main goal of the prototyping technique is to create a tangible representation of the system, aiding in the definition and validation of the gathered requirements (Budde and Zullighoven, 1990). However, if errors are introduced during the prototyping process, the presentation of the prototype can generate a distorted perception of the system, compromising the validation of the requirements. Additionally, the prototype may include functionalities that do not belong to the scope of the system under development, contrary to the initial intention of using prototyping as a resource to ensure quality and adherence to the system's objectives.

In this context, the production of the prototype and continuous validation play fundamental roles in ensuring that the system meets user expectations and aligns with the project's objectives. For this purpose, the Figma tool was used, allowing the creation of high-fidelity prototypes that simulate the complete operation of the system. At the beginning of the project, meetings were held with clients to thoroughly understand their needs and goals, and from these interactions, it was possible to elicit the requirements necessary for system development. During the prototype creation process, new meetings were periodically held to validate the developed functionalities, make adjustments to the interface, functionalities, and navigation flows. This iterative process ensured that the final prototype was sufficiently robust and served as a solid foundation for the development team to begin system implementation.

The main flows defined in the prototype reflect the most important scenarios for the user's experience with the system, among which the following stand out:

(a) Enter Scenario Data: The scenario creation flow, was designed to collect the initial information needed about the scenario that will be configured by the user. The interface aims to capture the data that will form the basis for the next step. It is composed of three fields: the scenario name, the scenario description (optional), and the company name. Below is the initial screen of the scenario registration process created in Figma.

(b) Create Groups: In the "Create Groups" functionality, part of Step 4 of the system, the user can automatically group the irregular connections by density. The generated groups are visually displayed on a map, considering the provided parameters, such as the minimum number of points per group and the radius size used in the analysis.

(c) Validate Groups: In the "Validate Groups" functionality, part of Step 5 of the system, the user can change groups manually, making additions or changing existing ones, through unions or separations. The generated groups are visually displayed on the map, considering the actions taken, without any limit established in the analysis.

In each step, the user needs to input data, and the system must verify the possible inputs and generate a response for each one, including alternative flows, such as when the user provides optional data in the system, and exceptions for invalid input data. At the end of the design process, the high-fidelity prototype was finalized in Figma, containing all the flows and elements necessary to guide the system's development.

3.3 Application of Exploratory Testing Based on the Prototype

Exploratory testing is a software testing approach that involves actively and dynamically exploring the application or system under test. Instead of following a predefined testing script, exploratory testers interact with the software, attempting to identify flaws, defects, and unexpected behaviors in an unstructured manner. This implies a dynamic process that encompasses learning, test planning, and execution simultaneously. The task is carried out through a constant cycle, marked by alignment with the mission, which involves formulating questions about the product. Answering these questions contributes to fulfilling the mission, which in turn guides the design of tests to obtain the desired responses and the subsequent execution of the tests, as explained by (Eduardo et al., 2021).

After the system was built, an exploratory test was conducted to verify whether all previously identified requirements were being satisfactorily met before presenting the final version to the client. For this purpose, a joint meeting was organized between the development team and the analyst team, during which the tests were conducted with the participation of all involved. The meeting was recorded using an online conferencing tool, allowing the material to be revisited and analyzed later.

During the exploratory test, the responsible analyst presented the system in detail, testing all possible flows and inputs to verify whether the expected output was produced in the prototype, checking the system's navigability, and ensuring that the included business rules were correctly implemented. Each screen was compared with its respective version created in Figma, and if differences were identified, they were highlighted and discussed in real time.

Some of the main flows included: to upload files containing points that would be analyzed; editing the files; defining groups to be considered when prioritizing possible tampering locations, classifying the data and viewing the results from the classification. This approach not only allowed the identification of defects but also proposed improvements that could be incorporated into the system in an agile manner. At the end of the process, the video of the meeting was made available to the team and other stakeholders, promoting transparency and facilitating progress tracking.

From the meeting held by the team, a detailed spreadsheet was created with the purpose of centralizing and organizing all information related to the defects found during the exploratory testing process. During this meeting, an analyst, utilizing their knowledge of the system's objectives and requirements, recorded each identified defect, detailed the circumstances under which they were found, and carefully analyzed each one. Based on this analysis, it was possible to determine the severity of each defect, classifying them into categories such as high, medium, and low, according to their potential impact on system functionality and the user experience. Out of the 50 defects identified during the test, 20 were listed as priorities on the spreadsheet due to the potential impact they could cause. In this paper, we will present the defects classified as high severity, considering that these represent the most critical issues that must be addressed before the system's release. This prioritization was based on factors such as the impact on the system's core functionalities and the user experience. However, the complete Table 1, can be consulted in the Footnote[1].

The mentioned spreadsheet details the defects found during the testing process, including the expected system behavior and the severity classification (high, medium, low). As the team had limited time to implement corrections before the final delivery, this artifact became essential for prioritizing the necessary changes without having to redo entire workflows within the system to identify the defects again. This allowed the team to quickly implement the missing functionalities, enabling new tests to be conducted on the implemented changes for further validation by the company's Analyst.

Although exploratory testing is often associated with traditional software testing, in this study, it was applied as a complementary approach to previously executed validation methods. The primary goal was to enhance the verification process by leveraging a more flexible and dynamic testing strategy that allowed for rapid defect identification.

This testing phase did not replace structured validation efforts but rather supplemented them by enabling real-time validation based on the system's prototype. The prototype served as a key reference document, ensuring that the final implementation aligned with the intended requirements and expected behaviors. This approach aligns with modern agile methodologies by providing an iterative mechanism for refining the system while fostering collaboration among development and testing teams.

By integrating exploratory testing into the validation process, the team was able to identify usability and functional issues that might not have been captured through automated or scripted testing approaches. Furthermore, this strategy allowed for immediate feedback, rapid prioritization of critical issues, and iterative adjustments based on stakeholder input. Contrary to the perception that this approach reflects pre-Agile software testing methods, it enhances agility by streamlining defect detection, promoting team collaboration, and improving the overall quality of the final product.

4 RESULTS AND DISCUSSIONS

Table 1 presents examples of the process of analyzing the defects found. In these examples, we highlight two main items: (a) "The 'view scenario' option should be disabled until the scenario is finalized."; and (b) "The map must contain a default functionality when clicking on the regions, allowing initial interaction." For each item analyzed, we gathered data on where these problems occurred and how they were reported during the evaluation by the testers. We analyzed and extracted the characteristics or quality attributes associated with these issues and translated them into requirements using the language of software engineering (i.e., detailing the objects of attention and their expected functionalities or how they should be implemented). Furthermore, to support the development team in implementing the requirements related to each identified item, we analyzed and grouped all reported improvement suggestions, developing specific instructions for the improvements according to the nature of each item.

For illustrative purposes, some of the identified defects were highlighted, showcasing the process involved in their correction and the application of improvement suggestions. In Figure 2, the identified de-

[1]To access the full Table 1, click here.

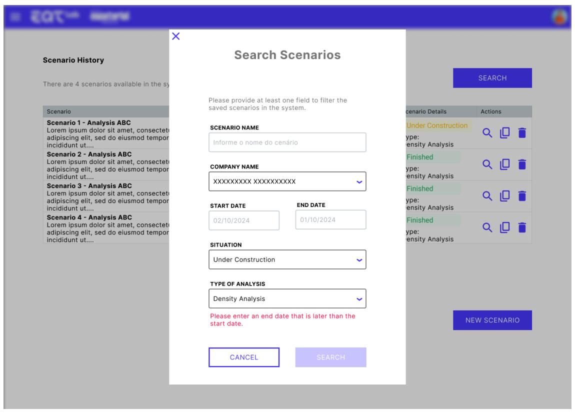
Table 1: Collection of defects found in the exploratory test.

ID	Defect	What was expected	Severity
D01	Button ”Forgot password?” does nothing	The button should redirect to a page with a password recovery form, including fields for email and a send button	High
D02	Invalid email allows login	Invalid email should not allow clicking the ”Login” button	High
D03	Allowing anyone to register in the system	Omit the ”Register” button	High
D04	When entering an end date earlier than the start date, nothing happens	It should display an error/some kind of notification	High
D05	It is not possible to view the scenario because there was no result (it was not finalized, still under construction)	Have an alternative flow to copy the scenario without reaching the finalization step (with a pop-up), allowing it to be edited	High
D06	Option to ”view scenario” enabled after deleting 1 scenario	Viewing the scenario should be disabled until it is finalized	High
D07	Clicking on ”view scenario,” when it is still under construction, loads the data	Should not return anything, as it was still under construction	High
D08	There are no ”groups” or ”marked points,” but it is marked as ”finalized”	Should remain ”under construction”	High
D09	When sending a file that should cause an error, it loaded	It should not have loaded	High
D10	Allows a file with points without data and does not display an error message	Alert screen showing what to do with this type of file (this will be the ”error message”)	High
D11	Delete all files at once	Have the functionality to delete/edit only 1 file	High
D12	Does not show replicated and empty points from the file	Should plot replicated and empty points, and create a checklist column on the right asking if that point should be deleted (displaying it in light gray) and update the message	High
D13	Logged out while working	Renew the authentication token while working, for about 10 minutes	High
D14	Even without points to fill in, it is possible to click the ”fill data” button	”Fill data” should be disabled	High
D15	Clicking the ”next” button displays a warning message	The ”next” button should be disabled	High
D16	In defining groups, only the map is shown, without any mapping	Show mapped points as if they belong to the same group (gray) before starting the interaction	High
D17	On the map showing 5008 points but only 4,999 in the file	Should show 4,999 points on the map	High
D18	Without selecting a functionality, nothing can be done on the map	A default functionality should already be pre-defined when clicking on the regions	High
D19	When selecting points, it is not possible to alter them	Have a functionality to modify points	High
D20	Whover on ”Select Module” does not make the change identifiable	Make the ”Select Module” have a hover effect that makes it bold	High

fect was: ”When setting the end date earlier than the start date, nothing happens.” To address this, an error message was implemented to indicate that the start date must be earlier than the end date. The use of the color red was intentional, as it visually signals that something is wrong. Additionally, the search button was disabled to prevent such queries from being executed. In Figure 3, the identified defect was: ”Allows files with points missing data and does not display an error message.” The solution involved adding an alert screen to inform users that the file contains unfilled points. Users were given the option to resolve the issue immediately or later, placing them at the center of the process and ensuring flexibility for future adjustments if needed. Finally, in Figure 4, the identified defect was: ”In defining groups, only the map is displayed, with no mapping.” This issue was resolved by displaying all points in the same color before selecting a group. This solution eliminated the previous ambiguity, enabling users to understand the status of the points before making a selection.

Therefore, the detailed documentation of the defects found and corrected has significantly contributed to a clear understanding of the system’s critical areas, allowing for agile and effective adjustments. The use of tools such as Figma and spreadsheets for recording and prioritizing issues reinforced the transparency and organization of the process, ensuring that corrections were implemented in a structured manner.

Despite the positive results obtained with the adopted approach, the absence of tests in a real operational environment may limit the generalization of the results. So far, evaluations have been conducted in a controlled environment, enabling the efficient iden-



Scenario History

There are 4 scenarios available in the sy

Scenario	cenario Details	Actions
Scenario 1 - Analysis ABC Lorem ipsum dolor sit amet, consectet adipiscing elit, sed do eiusmod tempor incididunt ut....	Under Construction ype: ensity Analysis	
Scenario 2 - Analysis ABC Lorem ipsum dolor sit amet, consectet adipiscing elit, sed do eiusmod tempor incididunt ut....	Finished ype: ensity Analysis	
Scenario 3 - Analysis ABC Lorem ipsum dolor sit amet, consectet adipiscing elit, sed do eiusmod tempor incididunt ut....	Finished ype: ensity Analysis	
Scenario 4 - Analysis ABC Lorem ipsum dolor sit amet, consectet adipiscing elit, sed do eiusmod tempor incididunt ut....	Finished ype: ensity Analysis	

Search Scenarios

Please provide at least one field to filter the saved scenarios in the system.

SCENARIO NAME: Informe o nome do cenário

COMPANY NAME: XXXXXXXXX XXXXXXXXXX

START DATE: 02/10/2024

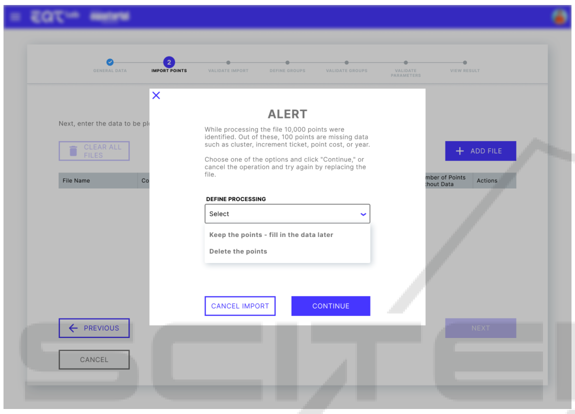
END DATE: 01/10/2024

SITUATION: Under Construction

TYPE OF ANALYSIS: Density Analysis

Please enter an end date that is later than the start date.

Figure 2: Search Scenario Functionality Screen - Error.



Next, enter the data to be pl

File Name	Co	mber of Points thout Data	Actions

ALERT

While processing the file 10,000 points were identified. Out of these, 100 points are missing data such as cluster, increment ticket, point cost, or year.

Choose one of the options and click "Continue," or cancel the operation and try again by replacing the file.

DEFINE PROCESSING: Select

- Keep the points - fill in the data later
- Delete the points

Figure 3: Search Scenario Functionality Screen.

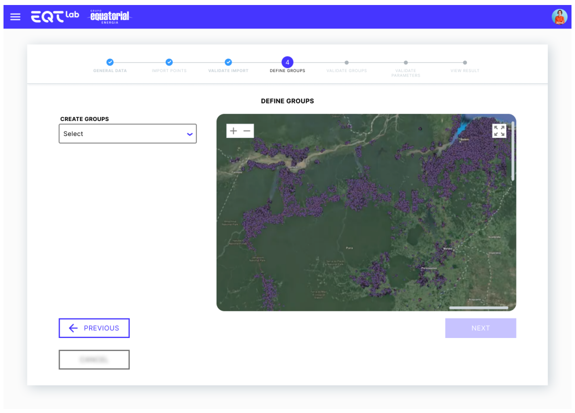


Figure 4: Search Scenario Functionality Screen.

tification of functional and usability flaws. However, this scenario does not account for external variables that may impact system performance and user experience in real-world situations. As future work, we propose conducting practical tests in the system's operational context, allowing for the analysis of factors such as variable electrical infrastructure, different user profiles, and integration with legacy systems.

Thus, introducing these tests will enable the validation of the approach's effectiveness in an environment closer to production, ensuring greater reliability of the results and identifying possible improvements to enhance the system's robustness. Additionally, considering challenges such as high data volume and adverse operational conditions will help consolidate the solution's applicability in the energy sector, ensuring it meets the real demands of companies and technical personnel involved.

5 CONCLUSIONS

The development and validation of the detection system for irregular connections through the combination of high-fidelity prototyping and exploratory testing proved to be effective strategies for ensuring software quality and meeting the needs of the energy sector. The adopted approach allowed for the identification and correction of defects in an agile and collaborative manner, integrating the development and analysis teams in an iterative and well-documented process. The use of the prototype as a central tool in the planning and execution of tests was crucial for aligning the system requirements with stakeholder expectations, ensuring that the implemented functionalities were robust and functional.

Additionally, the adopted strategy shares principles of the Minimum Viable Product (MVP), allowing for the rapid validation of essential functionalities before full implementation. The use of prototyping made it possible to test and refine the system's workflows without the need for extensive code development, reducing rework and ensuring that adjustments were made based on continuous team feedback. This iterative approach enabled a more efficient development process, focusing on the most relevant functionalities for the end user. Thus, the use of MVP combined with exploratory testing facilitated a cycle of continuous improvement, making the process more efficient and contributing to the development of a more robust software solution aligned with the demands of the energy sector.

The results obtained, including the reduction of the time required for validation and the identification of improvements in flows and functionalities, highlight the importance of integrating good development practices with techniques that can be implemented in an agile manner. The documentation of the defects found and corrected also contributed to a clear view of the system's critical areas, enabling quick and efficient adjustments. Moreover, the use of tools such as Figma and detailed spreadsheets to record problems reinforced the transparency and organization of the process, fundamental elements for the project's success.

Finally, the lessons learned from this project can serve as a basis for the application of similar methodologies in other contexts, expanding the use of prototypes and exploratory testing as pillars for developing high-quality technological solutions. As future work, we intend to expand the prioritization system by incorporating new analysis criteria, such as environmental and social indicators, which can influence decisions related to intervention areas. Moreover, the methodology used has shown promising results in improving software quality through exploratory testing and prototyping, but its application is still limited to a specific system context. However, this approach has been applied to other projects developed by the Applied Computing Center (NCA) at the Federal University of Maranhão (UFMA), covering different types of systems, such as service management, healthcare solutions, and organizational process optimization. Thus, future studies could further explore the adaptation of this methodology to other critical systems, such as applications in the financial and transportation sectors, where early defect detection is essential to ensure reliability and security. We hope that this work will inspire new initiatives in the energy sector and other areas, promoting innovation and the continuous pursuit of excellence in software development.

ACKNOWLEDGMENTS

This work was supported by the Instituto de Ciência e Tecnologia Grupo Equatorial and Grupo Equatorial through the PDI ANEEL program under grant PD-00037-0047/2022. The authors also acknowledge the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES), Brazil - Finance Code 001, Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq), Brazil, and Fundação de Amparo à Pesquisa Desenvolvimento Científico e Tecnológico do Maranhão (FAPEMA) (Brazil) for the financial support.

REFERENCES

Azevedo, S. D. B. J. d. S. and Castro, A. F. d. (2022). Development and evaluation of a prototype interface for an açaí sales application. *Undergraduate Thesis (TCC), Universidade Federal Rural do Semi-Árido (UFERSA)*.

Budde, R. and Zullighoven, H. (1990). Prototyping revisited. In *COMPEURO'90: Proceedings of the 1990 IEEE International Conference on Computer Systems and Software Engineering-Systems Engineering Aspects of Complex Computerized Systems*, pages 418–427. IEEE.

Crispin, L. and Gregory, J. (2009). *Agile Testing: A Practical Guide for Testers and Agile Teams*. Addison-Wesley.

Eduardo, J., Paiva, A., Ferreira, V., Rocha, S., Santos, Í., Rivero, L., Almeida, J., Braz Junior, G., Paiva, A., Silva, A., et al. (2021). Applying exploratory testing and ad-hoc usability inspection to improve the ease of use of a mobile power consumption registration app: An experience report. In *HCI International 2021-Late Breaking Papers: Design and User Experience: 23rd HCI International Conference, HCII 2021, Virtual Event, July 24–29, 2021, Proceedings 23*, pages 326–341. Springer.

Erdogmus, H., Morisio, M., and Torchiano, M. (2005). On the effectiveness of test-first approach to programming. *IEEE Transactions on Software Engineering*, 31(1):1–12.

Fulcini, T. and Ardito, L. (2022). Gamified exploratory gui testing of web applications: a preliminary evaluation. In *2022 IEEE International Conference on Software Testing, Verification and Validation Workshops (ICSTW)*, pages 215–222.

Silva, M., Fabrizzio, G., Ribeiro de Jesus, E., Barra, D., and Lorenzini, E. (2024). Construction and usability validation of a mobile application prototype for knowledge dissemination from the journal texto & contexto enfermagem. *Texto & Contexto - Enfermagem*, 33.

Sommerville, I. (2011). Software engineering (ed.). *America: Pearson Education Inc.*

Vidinich, R. and Nery, G. A. L. (2009). Research and development against energy theft. *Revista Pesquisa e Desenvolvimento da ANEEL–P&D*, 15.

Wynne, C. and Hellesoy, A. (2012). *The Cucumber Book: Behaviour-Driven Development for Testers and Developers*. Pragmatic Bookshelf.

Yu, J. (2018). Design and application on agile software exploratory testing model. In *2018 2nd IEEE Advanced Information Management,Communicates,Electronic and Automation Control Conference (IMCEC)*, pages 2082–2088.