Uncovering UX Smells in the Eclipse IDE: An LLM-Assisted Study of UXDebt in Mature Ecosystems

Carlos Carret Miranda[1], Juan Cruz Gardey[1], and Alejandra Garrido[1,2]

[1] LIFIA, Facultad de Informática, Universidad Nacional de La Plata, La Plata, Argentina
carretmiranda@gmail.com and jcgardey@lifia.info.unlp.edu.ar
[2] CONICET, La Plata, Argentina
garrido@lifia.info.unlp.edu.ar

Abstract. Improving the user experience (UX) of Integrated Development Environments (IDEs) is fundamental to enhancing developer performance and reducing the cognitive burden of modern software engineering. Unlike modern web-based IDEs, mature platforms like Eclipse often contain historical UX flaws that can obstruct daily development tasks. This paper presents an LLM-assisted study of UX smells in Eclipse by analyzing and classifying user-reported issues from its GitHub repository with the help of LLMs. Our results reveal that predominant UX smells involve poor discoverability, visual inconsistency, poor accessibility, and interface placement problems. These findings highlight how legacy software infrastructure requires specific UX maintenance strategies to remain competitive. This study contributes to understanding the evolution of developer experience in long-standing software systems.

Keywords: UX Smells, Developer Experience, UXDebt, LLM

Descubriendo UX Smells en Eclipse: Un estudio asistido por LLMs sobre la deuda de UX en entornos de desarrollo maduros

Abstract. La experiencia de usuario (UX) de los Entornos Integrados de Desarrollo (IDEs) es fundamental para potenciar el desempeño de los desarrolladores y reducir la carga cognitiva de la ingeniería de software moderna. A diferencia de los IDEs modernos basados en la web, las plataformas maduras como Eclipse acarrean problemas de UX que pueden obstruir las tareas diarias de desarrollo. Este artículo presenta un estudio de los UX smells en Eclipse asistido por Modelos de Lenguaje de Gran Escala (LLMs), mediante el análisis y la clasificación de los problemas (issues) reportados por los usuarios en su repositorio de GitHub. Nuestros resultados revelan que los UX smells predominantes involucran problemas de descubrimiento, inconsistencias visuales, barreras de accesibilidad y problemas de ubicación de elementos en la interfaz. Estos hal-

lazgos resaltan cómo la infraestructura de software heredado (legacy) requiere estrategias de mantenimiento de UX específicas para seguir siendo competitiva. Este estudio contribuye a comprender la evolución de la experiencia del desarrollador en sistemas de software de larga trayectoria.

Keywords: UX Smells, Developer Experience, Deuda de UX, LLM

1 Introduction

Integrated Development Environments (IDEs) are central to developers' daily work. As these tools have evolved into complex ecosystems, the notion of Developer Experience (DEX for short) plays a significant role in developers daily work (Fagerholm and Münch, 2012). This concept provides a framework to analyze how intuitive, efficient, and satisfying an IDE is to use. One useful way to study DEX is through *UX smells*: recurring interaction and interface problems that degrade the user experience without breaking functionality (Grigera et al., 2017). When such problems remain unresolved, they may accumulate as UXDebt (Rodriguez et al., 2023). This debt reflects the cost of suboptimal decisions in UX design that grow over time, making them increasingly expensive to fix (Baltes and Dashuber, 2024).

The original UX smell catalog was created for web applications (Grigera et al., 2017) and later extended with patterns identified in Visual Studio Code (Rodriguez et al., 2026). The extended catalog includes more than 20 UX smell types and their associated IDE quality dimensions. However, a key question remains: **are these smells generalizable to IDEs with fundamentally different architectural characteristics?**

To address this question, based on the study of UX smells in Visual Studio Code, we investigate whether a mature, long-standing IDE exhibits different UX smells. We focus on Eclipse, a mature IDE with decades of evolution, a classic UI framework, and a vast legacy plugin ecosystem. Unlike modern lightweight editors, Eclipse represents a different generation of IDE design, making it an ideal candidate for studying how architectural maturity influences UXDebt.

This paper presents preliminary results from an ongoing research line investigating UXDebt across different IDE ecosystems. Our contributions are: (1) empirical validation that the UX smells catalog generalizes to mature IDE ecosystems with different architecture, (2) the addition of a new UX smell called Visual Noise, and (3) a comparison of IDE quality attributes between mature and modern IDEs, with Eclipse showing concentration in Intuitiveness and Aesthetic design versus VS Code's focus on Informativeness and Clarity.

2 Method

We use a three-step process inspired in a previous work (Rodriguez et al., 2026) adapted to the Eclipse context, as follows: (1) data collection from the GitHub repository, (2) LLM-assisted classification using ChatGPT via web interface, and (3) synthesis and analysis. The following sections describe each stage in detail.

2.1 Data collection

We downloaded the issues from the Eclipse repository[3] using the GitHub API. Unlike VS Code, Eclipse codebase is split into multiple repositories (core platform, Java compiler, plugins, etc.), so we analyzed the one targeted for user-interface (UI) issues. There is no labeling on this repository so we extracted all the issues (as of April 2026), obtaining a total of N=1273 issues.

2.2 Categorization

The LLM-assisted classification was performed using ChatGPT (GPT-5.4) through its web interface. For each issue, we provided the title and body text with the extended UX smell catalog as context. The prompt instructed the model to classify each issue into one of four categories categories: a specific UX smell from the catalog, "bug" (for functional defects), "feature request" (for new functionality requests), or "No UX" (for issues unrelated to user experience). The model was also asked to justify each classification to validate its reasoning.

To validate the classification accuracy, each author independently analyzed a random sample of 10% of the dataset (127 issues). After independent classification, we conducted a consensus meeting to resolve discrepancies and establish ground truth labels. The overall classification accuracy reached 80.95%, with individual reviewer accuracies ranging from 76.38% to 86.61%.

3 Analysis

Among the 1,273 analyzed issues, 237 (18.6%) were classified as UX smells. The remaining issues were mostly bugs (48.8%), followed by non-UX reports (22.4%) and feature requests (10.2%). The most frequent UX smells were *Poor discoverability* (51 issues), *General UI Inconsistency* (33), *Poor accessibility* (29), *Inconsistent placement* (24), *Inconsistent Theming* (21), and *Clipped/Overlapping UI* (18).

Then, we mapped each smell to a primary IDE quality using the crosswalk between the extended smell catalog and Kuusinen's quality attributes (Kuusinen, 2015). A new smell was identified, *Visual Noise*, which was not present in VSCode. This smell captures cases where the UI contains decorative, redundant, or repetitive elements that add visual clutter without improving usability. We assigned *Aesthetic design* as its related IDE quality because it captures the impact of clutter and unnecessary visual decoration on perceived interface quality. A concrete example of this smell is Issue #114: "Remove forms default gradient and use a SWT color constant." In this issue, users reported that the default gradient effect applied to form elements creates unnecessary visual clutter that distracts from the actual configuration options. The smell *Visual Noise* accounts for 5.5% of identified UX smells in Eclipse (13 issues).

[3] https://github.com/eclipse-platform/eclipse.platform.ui

Table 1 shows that Eclipse UX smells are concentrated in *Intuitiveness* and *Aesthetic design*. This indicates that the most common experience problems are not primarily about missing functionality, but about discoverability, action placement, visual consistency, theming, and the perceived coherence of the interface.

Table 1. IDE qualities across Eclipse UX smell issues.

IDE Quality	Issues	% UX issues
Intuitiveness	83	35.0
Aesthetic design	69	29.1
Reliability	34	14.3
Clarity	33	13.9
Informativeness	10	4.2
Empowerment	4	1.7
Efficiency	4	1.7

4 Results

Table 2 presents a detailed comparison of identified UX Smells between Eclipse and VS Code, including absolute values, percentages, 95% confidence intervals, and p-values from two-proportion z-tests. The comparison reveals clearly different patterns between the two IDE ecosystems.

Table 2. UX smells distribution by IDE qualities with statistical analysis.

IDE Quality	Eclipse	%	CI 95%	VS Code	%	CI 95%	Δ pp	p-value
Intuitiveness	83	35.0	[28.9, 41.1]	194	13.3	[11.6, 15.1]	+21.7	<0.001
Aesthetic design	69	29.1	[23.3, 34.9]	72	4.9	[3.8, 6.1]	+24.2	<0.001
Reliability	34	14.3	[9.9, 18.8]	97	6.7	[5.4, 7.9]	+7.7	<0.001
Clarity	33	13.9	[9.5, 18.3]	286	19.7	[17.6, 21.7]	-5.7	0.036
Informativeness	10	4.2	[1.7, 6.8]	312	21.4	[19.3, 23.6]	-17.2	<0.001
Empowerment	4	1.7	[0.0, 3.3]	11	0.8	[0.3, 1.2]	+0.9	0.156
Efficiency	4	1.7	[0.0, 3.3]	223	15.3	[13.5, 17.2]	-13.6	<0.001

The statistical analysis confirms that most observed differences are statistically significant ($p < 0.05$). Largest gaps appear in *Aesthetic design* (Δ=+24.2pp, p<0.001) and *Intuitiveness* (Δ=+21.7pp, p<0.001), where Eclipse shows substantially higher concentrations. Conversely, VS Code exhibits significantly higher proportions in *Informativeness* (Δ=-17.2pp, p<0.001) and *Efficiency* (Δ=-13.6pp, p<0.001). The difference in *Clarity* is also significant (Δ=-5.7pp, p=0.036), though with a smaller effect size. Only *Empowerment* shows no statistically significant difference (p=0.156).

These results suggest that Eclipse accumulates relatively more UX smells in discoverability, action placement, theming, and visual coherence, whereas VS Code concentrates more on feedback, system-state communication, and workflow fluency. The confidence intervals indicate that these estimates are precise, with most intervals spanning less than 10 percentage points. Rather than indicating that one environment is uniformly better than the other, the comparison suggests that UXDebt can evolve differently depending on the maturity, architectural history, and design priorities of an IDE.

The main limitation of this study is that the validation sample (10% of issues) is relatively small, though all three reviewers achieved substantial agreement.

5 Conclusions and future work

This study presents preliminary results on UX smells in Eclipse, a mature IDE platform. Our findings suggest that the UX smell catalog generalizes to mature ecosystems, but the types of UXDebt that accumulate differ significantly from modern editors. Eclipse concentrates UXDebt in visual and perceptual dimensions (Intuitiveness and Aesthetic design), while VS Code shows higher concentrations in cognitive and workflow dimensions (Informativeness, Clarity, and Efficiency). We identified a new smell not present in VSCode: *Visual Noise*.

Future work includes: extending the analysis to other IDEs like IntelliJ IDEA and NetBeans for broader cross-IDE comparison, conducting longitudinal studies to track UX smell evolution across Eclipse releases, developing automated tooling for continuous UX smell detection in IDE development workflows, and examining the relationship between UX smells and developer productivity metrics.

References

Baltes, S., & Dashuber, V. (2024). Ux debt: Developers borrow while users pay. *IEEE/ACM 17th CHASE*, 79–84.

Fagerholm, F., & Münch, J. (2012). Developer experience: Concept and definition. *Int. Conf. on Software and System Process (ICSSP)*, 73–77.

Grigera, J., Garrido, A., Rivero, J., & Rossi, G. (2017). Automatic detection of usability smells in web applications. *Int. Journal of Human-Computer Studies*, *97*, 129–148.

Kuusinen, K. (2015). Software developers as users: Developer experience of a cross-platform integrated development environment. *Int. Conf. on Product-Focused Software Process Improvement*, 546–552.

Rodriguez, A., Gardey, J. C., & Garrido, A. (2026). Detecting ux smells in visual studio code using llms. *3rd International Workshop on Integrated Development Environments (IDE 2026)*.

Rodriguez, A., Gardey, J. C., Grigera, J., Rossi, G., & Garrido, A. (2023). Ux debt in an agile development process: Evidence and characterization. *Software Quality Journal*, *31*(4), 1467–1498.