Tool Window Usage Analysis in IDEs

Author: *Marina Vračarić*
Date: *8.11.2025.*

1. Introduction

The goal

This analysis investigates whether there is a significant difference in the duration that tool windows remain open in an IDE depending on whether they were opened manually by the user or automatically triggered by events (e.g., debugging).

Dataset origin and **context**

The dataset consists of IDE tool window events recorded over time, containing anonymized user IDs, timestamps, event types ("open"/"close"), and an indicator for whether the window was opened manually or automatically.

2. Data Understanding and Assumptions

Dataset Columns

- `user_id`: anonymized identifier

- `timestamp`: event time (epoch milliseconds)

- `event_id`: `"open"` or `"close"`

- `open_type`: `"manual"` or `"auto"` (only for open events)

Assumptions

- Only one tool window can be open at a time for one tool. Multiple open tool window actions, manual or automatic, do not open new window but put existing opened window into focus.

- Consecutive open or close events of the same type are possible due to logging issues.

- Missing open/close pairs are possible due to logging issues.

3. Data Cleaning Strategy

During the initial inspection of the event log, several inconsistencies were identified such as consecutive "open" or "close" events and unmatched records (events without a corresponding pair).
These issues likely result from system logging artifacts or rapid user actions that trigger multiple entries in a short time span.
To ensure reliable duration measurements, a structured data cleaning procedure was applied.

Handling Consecutive Events

For consecutive **"closed"** events, only the first one in each sequence was retained, assuming that repeated "close" logs do not represent distinct user actions or that we are missing open event between them. For consecutive **"opened"** events, two alternative strategies were developed to explore different interpretations of user intent:

- **Strategy 1 – Keep the first "opened" event**
  This strategy assumes that the first "opened" represents the true start of an interaction, while subsequent consecutive "opened" entries are redundant.
  It typically results in longer measured durations, as the opening time is anchored to the earliest recorded event in the sequence.

- **Strategy 2 – Keep the last "opened" event**
  This strategy assumes that the last "opened" event in a sequence marks the actual activation of the tool window before it remains open.
  It intuitively produces shorter durations, providing a more conservative estimate of active usage time.
  This approach is particularly useful if multiple "opened" logs are believed to originate from repeated system triggers or quick reactivations rather than distinct user actions.

By comparing both strategies, it was possible to evaluate whether these duplicated "opened" events significantly influence the results and to assess the robustness of the findings under different data-cleaning assumptions.

Handling Unmatched Events

Events that could not be properly paired were excluded from the analysis:

- "Opened" events without a corresponding "closed" (before the dataset ended) were discarded.

- "Closed" events without a prior "opened" were ignored.

This ensured that only valid open–close pairs were used in calculating tool window durations, providing a consistent and interpretable foundation for further analysis.

Duration Calculation

For each open–close pair:

$$Duration = (timestamp(closed) - timestamp(opened))/1000$$

The division by 1000 converts the duration from milliseconds to seconds.

Outlier Filtering

Durations above the 95th percentile were excluded to remove unrealistic usage times and potential logging artifacts.

It was assumed that tool window sessions exceeding approximately 14000 seconds (almost four hours) are unlikely to represent active user interaction.
Such long durations probably correspond to situations where a user left the IDE open without active work — for example, by closing the laptop or leaving the session idle — while the system continued recording the window as "opened."

Therefore, these extreme cases were considered **outliers** and were excluded from further analysis to ensure that the results reflect genuine, active tool window usage.

- **Strategy 2** → 95th percentile = 14100 s

- **Strategy 1** → 95th percentile = 58000 s (with additional analysis also capped at 14100 s for consistency and comparison)

4. Analysis and Statistics

Descriptive Statistics

```
manual.describe()

count      611.000000
mean       418.192823
std       1499.597388
min          0.015000
25%          2.159500
50%         12.284000
75%        137.120500
max      14100.989000
Name: duration2, dtype: float64
```

```
manual1.describe()

count      620.000000
mean       844.700900
std       4085.009624
min          0.015000
25%          2.215750
50%         13.115500
75%        145.698500
max      52725.404000
Name: duration2, dtype: float64
```

```
manual2.describe()

count      610.000000
mean       360.855334
std       1378.423189
min          0.015000
25%          2.078750
50%         11.090000
75%        114.358750
max      13917.197000
Name: duration, dtype: float64
```

```
auto.describe()

count      881.000000
mean      1002.656932
std       2279.787073
min          0.154000
25%         31.659000
50%        156.569000
75%        777.707000
max      14207.539000
Name: duration2, dtype: float64
```

```
auto1.describe()

count      920.000000
mean      2343.336459
std       7325.284067
min          0.154000
25%         35.195500
50%        172.863000
75%        926.878250
max      57743.124000
Name: duration2, dtype: float64
```

```
auto2.describe()

count      930.000000
mean       984.585308
std       2188.814105
min          0.154000
25%         30.236500
50%        151.655500
75%        794.200000
max      14093.312000
Name: duration, dtype: float64
```

| Strategy1 | Strategy1(95th percentile) | Strategy2 |

Across all strategies, the median and upper quartile (75th percentile) values remain remarkably consistent between manual and automatic openings.

This suggests that the typical user behavior - the usual duration for which a tool window remains open - is largely stable and not sensitive to the specific data-cleaning approach applied.

However, noticeable differences appear in the mean and standard deviation, particularly for automatically opened windows.

These statistics are strongly affected by long-duration outliers, which explains why the mean decreases substantially after applying percentile filtering (Strategy 1, 95th percentile).

Strategy 2, which keeps the last "open" event in each sequence, produces slightly lower average durations overall, reflecting its more conservative interpretation of active window usage.

In summary, while the choice of cleaning strategy influences the mean duration due to how outliers are treated, the central tendency and distribution shape (as captured by the median and quartiles) remain highly similar - indicating that the core user interaction pattern is robust across different preprocessing approaches.

Statistical Comparison

To evaluate whether there is a statistically significant difference in the duration of tool window usage between manual and automatic openings, the Mann-Whitney U test was applied.

This non-parametric test was chosen because the duration distributions were not normally distributed, as indicated by prior exploratory analysis

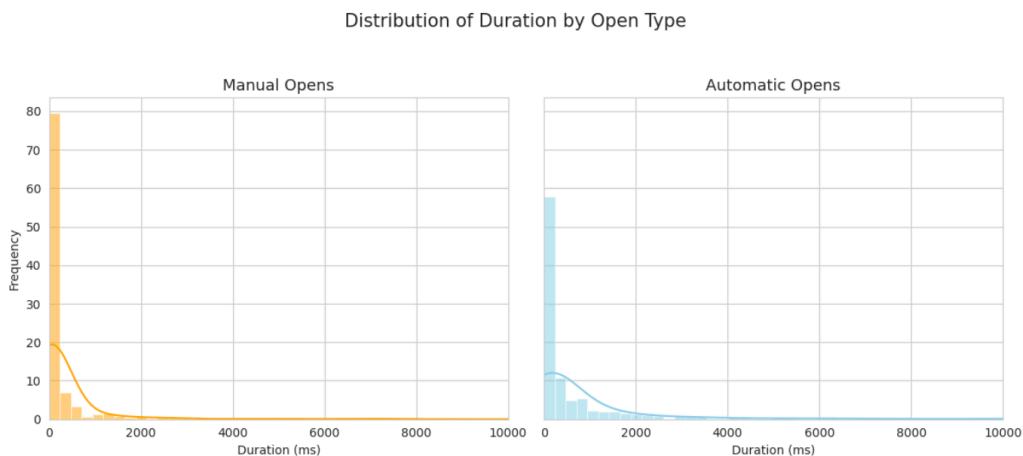| Comparison	p- value	Interpretation
Strategy 1	`2.72435196159057e-53`	Significant difference
Strategy 1 (95th percentile)	`4.932066879745845e-55`	Significant difference
Strategy 2	`4.373743990382247e-59`	Significant difference

The p-values obtained are far below the conventional significance threshold ($p < 0.05$), confirming that the difference between manual and automatic durations is highly statistically significant in both cases.

This supports the conclusion that manually opened tool windows tend to remain open for a shorter period of time compared to those opened automatically, regardless of the data-cleaning strategy applied.
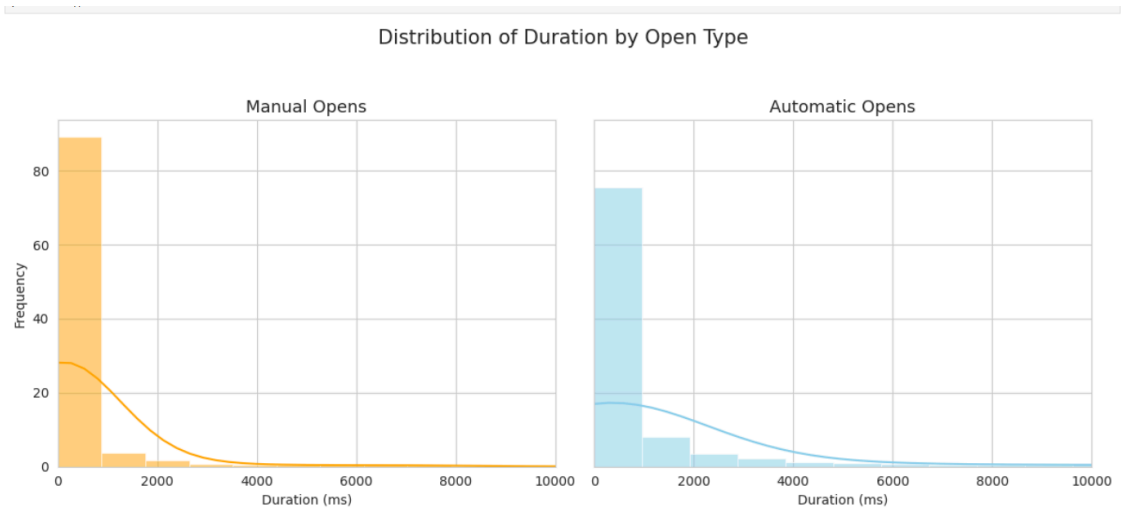
5. Results Summary

To better understand how different data-cleaning strategies affect the observed window durations, the results are visualized using histograms of duration distributions for each open type (manual and automatic).
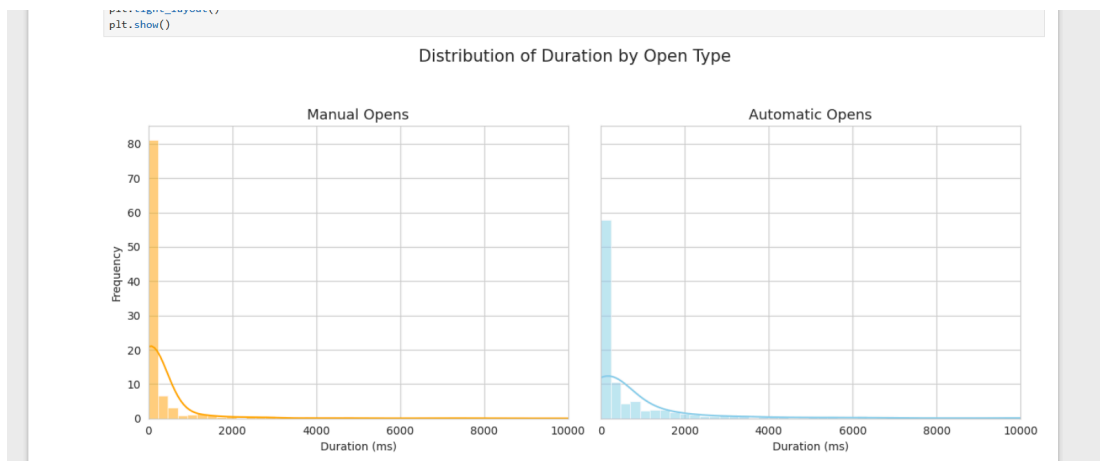
This approach provides an intuitive way to observe both the shape and spread of the data, highlighting differences that might not be immediately visible from descriptive statistics alone.

Strategy 1 - 92% percentile

Strategy 1 - 95% percentile

Strategy 2

Across all three strategies, the histograms consistently show that manual openings are typically shorter-lived, while automatic openings tend to remain active for longer durations.

Although the absolute values vary slightly between strategies due to different cleaning and filtering rules, the overall shape of the distributions remains stable — highly right-skewed, with most events concentrated near zero and a long tail of infrequent, extended sessions.

These findings clearly demonstrate that the two usage patterns differ fundamentally - manual openings represent deliberate, focused user interactions, while automatic openings reflect prolonged, system-driven processes within the IDE environment.

6. Limitations and Future Work

The dataset includes only one tool window type, and user behavior may differ across other windows or IDE contexts.

In addition, durations may be affected by factors not captured in the logs (e.g., user idle time, debugging sessions, or background tasks).

Future work could extend this analysis to multiple tool windows, incorporate user activity tracking, and explore clustering of usage patterns to better understand different user profiles.

7. Conclusion

The study demonstrates a clear and statistically significant difference in how users interact with tool windows depending on whether they are opened manually or automatically.
Manual openings correspond to shorter,more focused interactions, while automatic openings remain active for extended periods.
These findings are consistent across all data-cleaning strategies and support the idea that user intent strongly influences tool window behavior.
The methodology developed here can serve as a foundation for broader analyses of IDE interaction data.