**CS 5128/6028 Large Scale Software Engineering**

**Graduate Student Project**

Austin Conn ([connat@mail.uc.edu](mailto:connat@mail.uc.edu))

John Brown ([brow2jp@mail.uc.edu](mailto:brow2jp@mail.uc.edu))

The graduate student project for Large Scale Software Engineering called for a continuation of the key focuses present throughout the semester. Our group, consisting of Austin Conn and John Brown, chose to focus on the popular chat platform Discord for our main focus for the project. As part of the definition for the project, we compared our findings of the Discord changelog against the datasets previously identified in the course, the WebEx and Zoom datasets. Our analysis of the Discord dataset does extend back to 2017 whereas the datasets from the course are limited to the year of 2022. Our dataset was directly derived from the Discord changelog found on the Discord Developer Portal at https://discord.com/developers/docs/change-log.

Our project consisted of two parts: an Exploratory Analysis of our dataset, and a comparison between our dataset and the collective datasets analyzed during the duration of the course, the Zoom and WebEx datasets. During our exploratory analysis, we derived a method of collating and organizing our dataset for analysis, as well as defining several conclusions we arrived at while analyzing the dataset. During our comparison phase of the project, we compare our conclusions from our analysis and compare them to the conclusions found during the course.

Exploratory Analysis:

In order to compile a dataset for the Discord changelog, our first step was to scrape the webpage found on the developer portal into a file which we could manually manipulate into a format that could be analyzed. Previous attempts were made at scripting the scraping and manipulation of the webpage. However, it was found that due to a lack of standardized approach Discord took in publishing their changes, the nature in which the webpage data was formatted, and the lack of structured versioning, the level of effort required to implement these irregularities into organizing the data outweighed the level of effort required to manually manipulate the data into an analyzable format.

We do note, as opposed to the datasets from the course, our dataset does extend back to 2017. After compiling the information from the Discord changelog into an analyzable format, we found there were a total of 259 feature entries (Figure 1). Most of these entries were linked together, implying these feature releases were release alongside each other. This implication led to a classification of 85 unique release days over the more than five years spanning from 2017 through 2023. A key data point lacking in the Discord changelog were changes related to bug fixes or issues resolved as a result of user feedback. Due to the lack of granularity in bug fixes being reported, no features were removed or deprecated prior to exploratory analysis. We took the liberty in this decision during further analysis in the hopes that our ability to compare the Discord data with the Zoom and WebEx data.

**Figure 1: Discord Feature Count by Year (2017-2023)**

When looking at the magnitude in which changes were implemented throughout a calendar year, we found a rather even distribution across the year (Figure 2). We did note a spike in November of the calendar year, which appeared to be counterintuitive considering Thanksgiving falls at the end of the month. Many corporate calendars involve a short week surrounding Thanksgiving (in the case of our employer, Thanksgiving Thursday and the following Black Friday are given off) which would have normally led us to believe the week surrounding Thanksgiving should be rather unproductive. Given our analysis, however, we found this to be a surprising result.

**Figure 2: Discord Feature Count by Month (2017-2023)**

To get a further understanding of our dataset, we next analyzed it by grouping number of release days by year to grasp how many noteworthy changes were made by year. By analyzing the dataset in this manner, we do see, as the product matures, more changes occurring as each year passes. This is likely directly attributable to several key factors, including the age of the product, the number of users on the platform, and the number of developers working on the platform.

As a product ages, it appears logical that the product would grow in width; the product would expand to cover more than the initial target for which it was developed. The breadth of customer base that a product cover could be assumed to also account for the increase in feature release a product experience each year.

Further, as a product ages, it could be expected that the number of users would also grow. We believe that this is represented in the growth seen from year to year regarding number of feature releases per year growing.

Lastly, we observed that an addition of developers could further increase the number of features introduced throughout the year. As is aligned with the previous two observations, it is logical that as a product grows and its userbase grows, additional resources would be needed in terms of developmental ability which could account for an increased number of developers helping to contribute to the overall total of features released at any given time.

**Figure 3: Discord Release Day Count by Year (2017 -2023)**

NOTE: NOTHING BELOW HERE HAS BEEN COVERED OR INTERPRETED

**Figure 4: Discord Release Day Count by Month (2017 -2023)**

**Figure 5: Discord Feature Count Depreciating Old Features by Year (2017 -2023)**

**Figure 6: Discord Feature Count by Month (2022)**

**Figure 7: Zoom Feature Count by Month (2022)**

**Figure 8: WebEx Feature Count by Month (2022)**

Conclusions:

… discuss conclusions …

\*talk about workflow of each company

\*\*consistent for Webex, zoom and discord have months with nothing

\*talk about release windows for discord/zoom

\*\* talk about quarters (of the year with like a business angle)

Figure 9: Combined Feature Count by Month (2022)

TODO:

• Formalize the spreadsheet

o submit

• Write the report

o Link to definition: https://en.wikipedia.org/wiki/Exploratory\_data\_analysis

o Throw as much garbage as possible!

o Submit

• Effort/peer evaluation

o Create/submit

“Curate the features” -> organize

Exploratory analysis:

• Total # of features

• Total/average/mean/max min # of features per month

• # of tokens (voc size)

• # of sentences per feature

• Sentence length

• Feature length …

Draw meaningful comparisons with Zoom and WebEx:

• When are released happening (month)

o Compare by year (2022 for zoom data)

o “clean boundaries”

• Number of releases per year

• Sentence length