**Comprehension of challenges at the level of software ecosystem and global software engineering**

Authors: basically, software ecosystem was given by “David G. Messerschmitt” and “Clemens Szyperski” which explains or defines to make a functioning unit for shared market and make a relationship with business and software. On the other hand, global software engineering means same. But this page contains the ecosystems of Facebook which was given by or written by Ralf lammel on July 2020.

**Introduction.** This page which mentions the ecosystem of software at Facebookwhich includes the infrastructure, Data infrastructure and AI Infrastructure and how Facebook makes its business in software marks and how they solve their challenges as their total business is on social media.

Summary

Software engineering and ecosystem at Facebook and its business includes the infrastructures includes version control, CI, language services, testing automation, data infrastructure includes storage engines, query engines, pipelines, metastores and AI infrastructure includes ML workflows, feature stores, online and offline prediction on GitHub

Now the page tells us about what really is a software ecosystem it’s actually a collection of software projects which are developed together in same environment like same platform as environment can be physically or virtual as projects are part of open-source community.

Now comes the challenging part for the Facebook as they have to continuously develop app, services internal tool, bug management and tracking, engineering in different time zones different employee and their degree types.

Comprehension challenges in developer workflow, ownership management, Code review automation

Scenarios of work-item prediction I/II

The ‘Incident Response’ Scenario:

• Work item: Alert for suboptimal performance

• Question: The workflow steps to follow in response

• Automation: Record steps in past instances

• Challenge: To know when someone is responding

Scenarios of work-item prediction II/II

The ‘Aggregate Performance’ Scenario:

• Work item: A diff (a system change)

• Question: Time spent on diff

• Automation: Record all activities on diff

• Challenge: To know when someone is working on the diff

These scenario shows that how scenarios are responded which are incidentally given by the employees are worked by the Alert suboptimal performance and questions asked by the costumers are responded by the workflow steps, automation records step in the past instances and the challenges which are responded by the employees 2nd scenario tells that if the work item is on change system which means on Linux or windows questions replied on different time zones record all activities on diff database and challenges are solved by different teams that’s called Aggregate performance scenario

Now the team talks about the dark natter which means the data which is not in use they efficiently try to manage it by a way they called dark matter work flow analysis, they do it by making the data go into query DB interactively and then send it to commit a version locally now the team reads the data if the data is reusable they leave it but if the data is not in use they send it to a different company or publisher and publish it from another company if the data is integrity is better than they review it or they leave it.

In this page they also mentioned that why do they have dark matter probable reasons given by the teams are that Facebook tools don`t track work items consistently and the tools aren`t fully integrated although the American government tracks all the details which Facebook is not mentioning also they tell us that the logging in is not designed with workflow analysis in mind as the developer workflow don`t have proper structured also the developers are constantly in engage in a lot of context Switching now the Facebook team consider the aspects by tools, tools which are added obsoleted and are easily removed, tools which are functionally add and removed which means new versioning, integrating with others might also help Facebook team, main problem of logging dark matter they resolve it by schema or semantics evolves in the end practice make it better

On the other page the author Ralf talks about how Facebook switches their context in a table which explains that at the end of the year different are created then by other team diff is shipped by the other team who sends to diff developers to abandoned and diff is reverted if needed and of the year the work is still in progress the next year diff is published by the developers version is created for the new version and developers mod is introduced to get review from the costumers if the diff is accepted then changes are requested and are made although the Facebooks says that they might need more time in high confidence events .on the next page Ralf showed a table that shows a system for diff prediction which explains that logging foundation is managed by the integrate all avail logs which are version control continuous integration cli, internal web-based tools . Time windows into dark matter in which team uses 10 minutes or more to test the probability of the employee on diff. candidates work items includes anything may have possible working on a diff in the end Ralf tells us about the Comprehension Challenges in Ownership Management which relates that their work in Ownesty, now what really is ownership management, Facebooks says that each asset has the most accountable owner at all the time. Basic challenges in ownership management are mentioned as Challenge Details Ownership Decay How to know whether to trust owners on file? Asset sub classing How to identify and handle specific subsets of assets? Team-level ownership How to assign teams as owners with individual signal? Ranking owner candidates What ranking to use to recommend one or more candidates? Whole/part asset relationships How to obey those relationships with recommendations? Monotonic features How to make sure that "more" means "more likely owner"? Explainable recommendations How to explain recommendations to use so that they accept? In the end Ralf writes about the entities involved in code review at Facebook which are Diff • Diff summary • Diff test plan • Commit • CI signal Miscellaneous • Task (bug or feature) • Alert • Incident • Root causing diff they are improved by knowledge graph change impact analysis traceability recovery and summaries I want to include that the author likes the cats too much .