CSE 563 Project Individual Report Number 2

Sai Ganesh Nindra

Table of Contents

Table of Contents

1. Customer Problem (Updated)

1.1. Lack of Agile-centric effort tracking

1.1.1. Ineffective effort logging for agile teams

- To make EffortLogger a useful tool in modern day businesses it needs to cater to modern-day practices followed by businesses like the Agile methodology. As stated in 'Agile Processes and Methodologies: A Conceptual Study'[1] agile methodology follows an iterative and incremental process, where requirements can be changed with respect to the customer needs
- Effort Logger is not ideal for a fast-paced environment that is around in the present day. EffortLogger doesn't provide any estimation tools either which are widely used in agile teams such as planning poker. Integration of estimation techniques such as planning poker would be good.
- The ideal approach to estimation techniques to include would be machine learning. Using data-based estimation over an expert-based estimation method would prove to be more useful. Using metrics such as Median of Magnitude of relative error shows more accuracy in data-based estimation.[2]

1.1.2. Role-Specific Features

- EffortLogger treats everyone in the same manner, as in, if the person in question is a developer, project manager, product owner, tester or a scrum master, effort logger doesn't provide any features for different members of a team
- There needs to be a certain structure wherein only users with certain access can use certain features like accessing historical data, reports and dashboards

1.2. No integration with agile tools

1.2.1. Inefficient method of tracking effort and defects

- Effort and defect are currently tracked using user input, when the task is started and when it is complete. This can be inaccurate and misleading, which can give a wrong picture of how much effort has been put and what defects occurred.
- Integration with tools like IDE's, Jira and GitHub can help track efforts based on user activity, commits and if any updates have been made by the user. This eliminates guesswork keeping a data-driven approach in the future more accurate.

[1] Sharma, Sheetal & Sarkar, Darothi & Gupta, Divya. (2012). Agile Processes and Methodologies: A Conceptual Study. International Journal on Computer Science and Engineering. 4.

[2] Fernández-Diego, Marta & Mendez, Erwin & L. Guevara, Fernando González & Abrahão, Silvia & Insfran, Emilio. (2020). An Update on Effort Estimation in Agile Software Development: A Systematic Literature Review. IEEE Access. 8. 10.1109/ACCESS.2020.3021664. DOI:10.1109/ACCESS.2020.3021664

1.2.2. No method to track current efforts for future use

- The current data which is stored may have inaccuracies and are not used to compare with historical data. Agile teams need insights into how much effort is spent on completing tasks.
- With dashboards and visualization tools, agile teams can compare historical effort data with current effort data which will help in improving estimation strategies for the future

1.3. Privacy and Security of data

1.3.1. Security of Data

- EffortLogger was developed in the 1990's and there are multiple new standards the industry has adopted to safeguard themselves from attacks and breaches which need to be adopted by EffortLogger.
- Old software is very vulnerable to attacks which makes it susceptible to data breaches and hacks.
- The system needs to be equipped with robust security features like multi-factor authentication and data encryption.[3]
- The data that is being used as historical data must be done by the employees themselves, they cannot view past user stories worked on by other employees ensuring employee data privacy

1.3.2. Compliance with rules and regulations

- Rules and regulations have been ramped up since the 1990's when EffortLogger was developed and will need updates to comply with the current regulations.
- It will lack basic encryption and will be following different data policies.
- We will need to ensure that it is certified so that it is following various government regulations with respect to data privacy and security.

1.4. No integration with cloud

1.4.1. Data stored locally

- The current version of EffortLogger stores all data locally which is in turn stored on a floppy disk. This can cause loss of data and inefficiencies in usage of such data.

[3] What is Data Protection and Privacy? n.d, https://cloudian.com/guides/data-protection/data-protection-and-privacy-7-ways-to-protect-user-data/

- There are tracking delays as well and any updates require a lot of process as accessing previously logged data becomes a time-consuming process.
- Integration with cloud opens many doors such as no inconsistency in data, convenience in updating, automatic backups which result in no loss of data and this data can be accessed from anywhere.

~~1.4.2. Multiplatform compatibility~~

- ~~EffortLogger was just an application on the computer which causes inconvenience in modern day as users are on the go always. Not everyone has access to a computer at all times.~~
- ~~Making EffortLogger accessible on mobile devices as well would help users update their work from anywhere.~~

2. Stakeholders (Updated)

2.1. Members of an Agile Team

2.1.1. Developers

- In an agile team, the developers are the fundamental unit who work and log their work to measure their effort and help them keep a track of their productivity and give them a sense of where they stand with respect to the work being done.
- Without integration of agile tools with an EffortLogger, developers face difficulty in tracking their work which leads to inefficiency and mistakes.
- Developers need to be able to understand historical data accurately since its their own data to ensure accurate data driven estimations.

2.1.2. Product Owners

- The product owner is the one responsible for establishing a stakeholder standpoint with the team and focusing on the customer's needs[4]. If data being logged is inaccurate it becomes a problem in looking over progress.
- If historical data isn't present, then they will have to rely completely on guesswork making it harder to make data-driven decisions to maximize productivity and efficiency.
- Product Owners have privileges to access team reports which give insights on how a team is performing and how accurate are the estimations since adopting a data driven approach

Formatted: Font: 12 pt

2.2. Business Executives

2.2.1. Owner (Financial Stake)

- Stakeholders can be investors who put money into the product as well[5]. They expect that the tools that are being built will meet the end user's needs and will be a good market fit for them to gain financial benefit.
- As the EffortLogger of the 1990s is outdated based on security features, regulatory compliances and adaptability in the current IT environment it will not help with increasing sales or profit.
- Integrating the application with cloud and having multiple features that support agile principles will increase its acceptance in the industry, lifting its sales which in turn make it profitable.

[4] Langholf, Valentin & Wilkens,. (2021). Agile Project Management, New Leadership Roles and Dynamic Capabilities - Insight from a Case Study Analysis. 11. 1-18. Doi: 10.25437/jcsm-vol11-17.

[5] Dragos, Paul. (2021). THE IMPACT OF STAKEHOLDERS IN AGILE SOFTWARE DEVELOPMEN. THE ANNALS OF THE UNIVERSITY OF ORADEA. ECONOMIC SCIENCES. 30. 353-362. Doi: 10.47535/1991AUOES30(2)037.

2.2.2. Executives

- Different executives of companies look after the strategies, decision making process, profitability and engineering. They expect the tool to work with the necessary features.
- The data driven solutions that the new version brings to the table help execute strategic decisions in resource optimization , maximize productivity and profitability.
- Using dashboards and the metrics provided they can form strategies to reduce delays in projects and allocate necessary budgets as well.

2.3. QA and DevOps Team

2.3.1. QA Team

- The QA team is responsible for ensuring the quality of the product. They ensure that the operation of the software happens seamlessly and find new ways to improve quality. In agile they ensure quality through communication, testing, refactoring and integration[6].
- Without accurate information of defect logging and effort logging the team faces problems in understanding, what requires more extensive testing and how much progress has taken place.
- Integrating the application with agile tools can help the team identify the progress as well as if there have been certain hinderances or bottlenecks in the testing and can improvise in future stages.

2.3.2. DevOps Team

- DevOps team look after the automation and integration of systems. They ensure that EffortLogger is easily deployable and works as it should on the user end.
- As there is no integration with agile tools as of now, it doesn't allow the team to efficiently manage continuous integration, continuous deployment pipelines which will affect the quality of the product at the customers' end.
- With the software being integrated with the cloud they can ensure no inaccuracies and delay in delivering their work. Using version control systems helps in maintaining a good workflow as well. They can ensure scalability as well (catering to many users).

[6] Abdullahi Wakili, Almustapha & Alhassan, Lawan & Kamagata Hamisu, Abubakar. (2024). Quality Assurance Practices in Agile Methodology. Doi: 10.48550/arXiv.2411.05134.

2.4. GRC Team

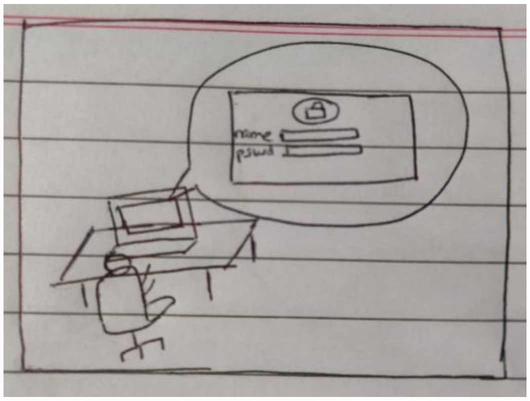
2.4.1. Governance, Risk and Compliance Team[7]

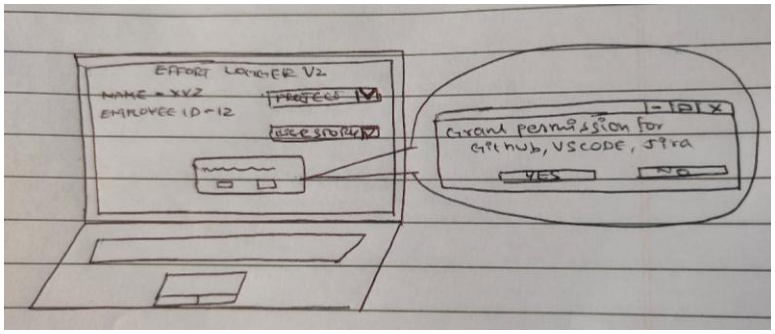
- They ensure that the tool that is being built complies with all necessary rules, regulations and standards.
- The EffortLogger being outdated was never going to comply with the current policies. It would have remained in a very susceptible environment to data breaches, which don't uphold the current policies and standards.
- By updating the necessary security features of the tool, the GRC team can ensure the safety and security of the users and their data, which in turn help the tool be available to users in the modern day as they comply with government regulations and are safe to work with.

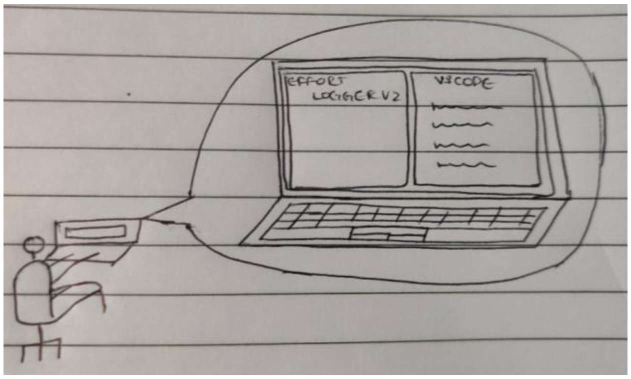
[7] "Regulatory Compliance in Agile Projects: Challenges and Solutions", n.d, https://transformation.agiledigest.com/regulatory-compliance-in-agile/

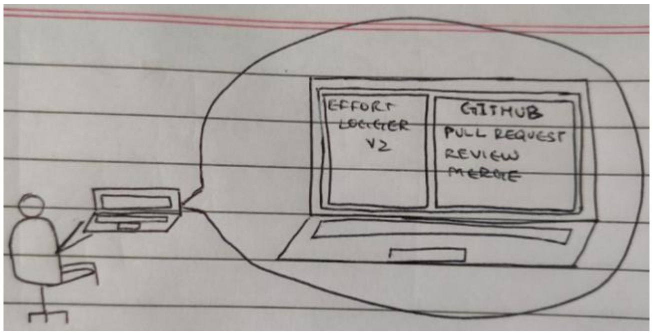
3. Prioritized Operations Concepts

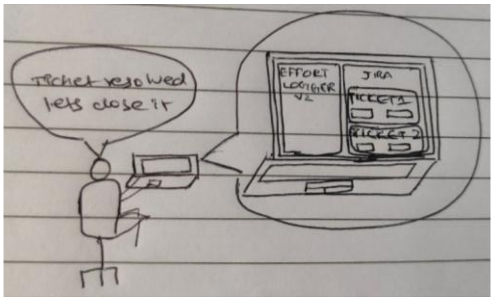
3.1. Storyboard

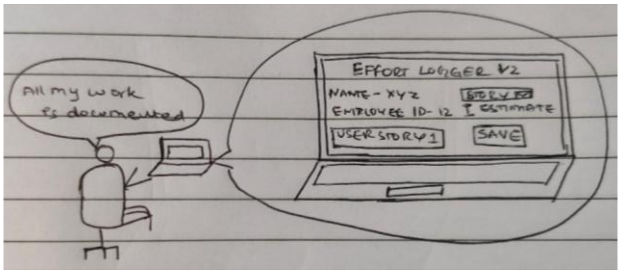












3.1.1. User Authentication

- Developer logs into the EffortLogger using their credentials.
- These credentials are private data and unique for every employee.
- After logging in, a dialog box pops up asking to grant permissions to connect to agile tools.

3.1.2. Developer starts working

- Developer opens IDE and starts working. (writing code)
- The storyboard shows EffortLogger and IDE on a split screen, but it can run in the background as well.

3.1.3. Code merge

- Developer creates a pull request which is reviewed by a team member and is merged.
- EffortLogger running in the background tracks commits, pull requests and merges.

3.1.4. Task solved

- Once a task is solved, a ticket can be closed on tools like Jira.
- Developer manually closes tickets on an agile tool which will be automatically detected by EffortLogger which is running side by side.

3.1.5. Review and Save

- Developer goes through the user story data, reviews it and saves it which nevertheless automatically gets saved.

3.2. Introduction

3.2.1. Project Description

- The latest EffortLogger system is a product that aims to make a shift in software estimation from opinion based to a data driven approach. The new system will collect and store historical data, which will be used[8] while incorporating agile methodologies to improve the accuracy of story point estimation.

3.2.2. Background

- The conventional way of effort estimation is carried out in a very subjective manner, often leading to inaccurate estimates which in turn cause project delays or just make the entire process more lengthy than usual. The original EffortLogger was developed in the late 20th century, which doesn't support any of the modern-day methodologies[9]. This newer version of EffortLogger will incorporate modern day practices, improve estimations with a shift to a more data driven approach, project planning and all in all the way software development process takes place.

3.2.3. Assumptions and Constraints

- Assumptions
  - The users must be willing to adopt this new version.
  - Historical data must be available and relevant.[8]
- Constraints
  - The new version of EffortLogger must be able to integrate with the tools being used in the industry currently. (For example, Jira, IDE's).
  - The software must be in compliance with all rules and regulations be it related to data, security or laws.

[8] Internal document: EffortLogger Business Opportunity (Document V1.2) Pg.5
[9] Internal document: EffortLogger Business Opportunity (Document V1.2) Pg.6

Conclusion

3.3. Overview of the Envisioned System

3.3.1. Overview

- The focus with this new version of EffortLogger is tracking developer effort in a better and more accurate manner which can further help in story points estimation. When integrated with planning poker it should provide historical data to assist and help developers make data driven decisions.

3.3.2. System Scope

- User authentication.
- Tracking effort of users.
- Logging of defects associated with a user story.
- Integration with agile tools.
- Provide historical data during planning poker sessions. (Integration of Planning Poker with EffortLogger).
- Data analysis and visualization of various metrics for higher officials.
- Security and privacy measures.

3.4. Needs, Goals and Objectives of Envisioned System

- Accurate effort tracking using metrics apart from time and lines of code.
- Store data of user story to serve as historical data for future use.
- Improve estimation accuracy by using historical data.
- Provide insights and help to developers, managers and executives based on previous work, team efforts and business progress.
- Ensure data privacy and security.

3.5. Overview of System and Key Elements

- Integration Layer: Connection between EffortLogger and agile tools to capture effort metrics.
- Tracking and Logging Console: Logs effort and defects.[10]
- Suggestive Algorithms: Provides data-driven suggestions during Planning Poker.
- Data Analysis Engine: Processes estimate data to provide insights through dashboards and visuals.
- Database: Stores effort data and user story data.

[10] Internal document: EffortLogger V1-11 User Guide V1.1 Pg.2

3.6. Proposed Capabilities

3.6.1. Effort calculation operation

- Integrates with agile tools to track effort.
- Analyzes time spent on different activities within a user story, such as coding, testing, and debugging.
- Effort can be estimated looking into the variety of skills required, number of tickets being assigned on a regular basis, time taken to resolve tickets and how many are still pending.
- Number of defects logged and resolved can also depict effort put into working on a task.
- Number of modifications in code can show how much effort has been put into developing something.
- Analyzing pull requests and reviews while merging code.
- Measuring the amount of work completed in each sprint can provide insight into team effort over time, which in other words would be sprint velocity.

3.6.2. Description and Tagging Operation

- Allow developers to add a description to user stories which will help in tagging user stories using the keywords used in the description.
- Developers add skills and tools used for the user story while working on it.
- Tags are assigned automatically using algorithms (NLP) which pick the keywords in description.
- Historical data includes both descriptions and skill tags for future references.

3.6.3. Integration with agile tools Operation

- Provides a data-driven approach only for the initial phase for discussions during estimation.
- Historical data can be accessed of only the employee who is trying to access the data. No other data should be visible.[11]
- During Planning Poker sessions, the system automatically (Machine Learning) suggests similar historical user stories based on tags with skills used and initial estimate of story points and final story points.

3.6.4. Analytical Operations

- Generate reports on team performance, estimation accuracy, and productivity trends.
- Restricts access to detailed team analytics to Product Owners or Managers only.
- Provides customizable dashboards for different management levels. (Developer level/Executive Level).

[11] Internal document: EffortLogger Supervisor Input V1-2 2024-01-06 Pg.1

3.7. Operational Scenarios

3.7.1. Nominal Conditions

- Developer Logging Effort upon completion of user story
    - Developer logs into the EffortLogger system.
    - Developer selects the current user story they are working on.
    - Developer writes code and does their work.
    - Upon completion of work, they close all open tickets and initiate a pull request.
    - EffortLogger tracks all the effort and stores the data automatically which can be reviewed by the developer.
    - They add tags for skills used (e.g., "Java", "RESTful").
    - User story gets added into database with its tags of user story description and skills used.
- Product owner (PO) prepares sprint planning using Planning Poker (Online).
    - PO initiates a new Planning Poker session in the system.
    - PO inputs the new user stories for estimation.
    - For each story, the system shows each member similar past stories based on user story tags.
    - The team discusses and votes on estimates using historical data as a reference.
    - Upon completion of sprint, PO investigates dashboards to see teams' efforts over the duration of the sprint. (Provides insights for future).

3.7.2. Off-Nominal Conditions (TBD)

3.8. Risks and Potential Issues

- Employee data privacy concerns.[12]
- Unacceptance by users due to productivity reasons.
- Over reliance on historical data for estimations of new user stories.
- Integrating with each tool used in agile methodology might pose challenges.

[12] Internal document: EffortLogger Customer Need V2-0 Document V1-2 Pg.2