OSSPREY: AI-Driven Forecasting and Intervention for OSS Project Sustainability

Nafiz Imtiaz Khan, Priyal Soni, Arjun Ashok, Vladimir Filkov
Department of Computer Science, University of California, Davis
{nikhan, pdsoni, arjashok, vfilkov}@ucdavis.edu

Abstract—**Open source software (OSS) underpins modern software infrastructure, yet many projects struggle with long-term sustainability. We introduce OSSPREY, an AI-powered platform that can predict the sustainability of any GitHub-hosted project. OSSPREY collects longitudinal socio-technical data, such as: commits, issues, and contributor interactions, and uses a transformer-based model to generate month-by-month sustainability forecasts. When project downturns are detected, it recommends evidence-based interventions drawn from published software engineering studies. OSSPREY integrates scraping, forecasting, and actionable guidance into an interactive dashboard, enabling maintainers to monitor project health, anticipate decline, and respond with targeted strategies. By connecting real-time project data with research-backed insights, OSSPREY offers a practical tool for sustaining OSS projects at scale.**

The codebase is linked to the project website at: https://oss-prey.github.io/OSSPREY-Website/
The screencast is available at: https://www.youtube.com/watch?v=N7a0v4hPylU

I. Introduction

Open Source Software (OSS) has become the invisible backbone of modern computing, powering critical infrastructures and enabling innovation across sectors from academia to Fortune 500 companies [1], [2]. Despite their high importance, approximately 90% of OSS projects experience stagnation or total abandonment [3], often without warning, creating maintenance issues and even critical vulnerabilities in the software supply chain, since most modern systems are heavily dependent on OSS.

At the heart of this challenge lies the question of *sustainability* [4]. How can we measure project is on a sustainable trajectory, detect early signals of decline, and provide actionable insights to help projects thrive?

Available tools deliver only limited solutions. **Apache Clutch**[1] provides real-time status updates for Apache Incubator projects[2] but in a one-size-fits-all manner, and lacks historical trends or diagnostic feedback. **APEX**[3], developed by Ramchandran et al. [5] takes a data-driven approach using sociotechnical networks to forecast the sustainability of Apache projects with LSTM-based forecasting model. However, APEX has limitations: (1) It is based only on Apache foundation projects and (2) it provides no actionable recommendations for projects dropping off of sustainable trajectories.

Currently, there exists no open source tool, which attempts to *forecast* the evolution of sustainability for general GitHub projects. Let alone, they provide future trends based on past activity of a project. This future trend is essential for proactive intervention. Furthermore, when issues are detected, maintainers lack precise remediation strategies/actionable recommendations based on measurable properties and scientific literature findings to help them make their projects sustainable in the long run.

To address these identified gaps, we introduce **OSSPREY** (*Open Source Software PRoject sustainabilitY tracker*), which extends prior work by offering: 1)

1) **Real-time monitoring:** Given a GitHub repository link, OSSPREY can scrape project commits, issues, and related metadata in real time.
2) **Forecasting capabilities:** The tool predicts the month-wise (current and future) probability of sustainability for any given project.
3) **Actionable recommendations:** When a downturn (i.e., low sustainability probability) is predicted, the tool suggests Researched ACTionables (ReACT) [6]—evidence-based interventions drawn from prior literature—to support maintainers.

Thus, OSSPREY bridges the gap between researched actionables and practical implementation, equipping maintainers with data-driven decision support.

II. Background & Theory

This section outlines the foundational theories guiding OSSPREY's development. We employ sociotechnical networks to model the evolving OSS dynamics and leverage Researched Actionables (ReACTs) to generate actionable recommendations.

A. Socio-Technical Networks

OSS contributions are classified into technical and social categories [7]. Technical contributions include code commits, pull request reviews, and issue resolutions, while social contributions include mailing list participation, issue commenting, and community forum participation. These can be modeled through socio-technical networks, structured graphs capturing collaborative dynamics.

[1]https://incubator.apache.org/clutch/
[2]https://incubator.apache.org/projects/
[3]https://ossustain.github.io/APEX/

The technical network can be modeled as a bipartite graph, where one set of nodes represents developers and the other set represents the files they have contributed to. Similarly, the social network can also be represented as a bipartite graph, where nodes correspond to developers, and edges capture interactions between them, such as: communication through issue comments, mailing lists, or code reviews.

By extracting data from issue trackers, mailing lists, and commit histories, researchers construct these networks to analyze patterns that distinguish sustainable projects from those that fail. Previous research demonstrates their utility in examining how collaboration structures impact project outcomes [8], [9].

B. Researched Actionables (ReACTs)

ReACTs, pioneered by Khan and Filkov [6], address the gap between sustainability research and practical guidance for OSS maintainers. Motivated by the challenge that many key project metrics such as contributor activity or commit frequency are not directly controllable by project teams, the authors proposed this concept and introduced 105 empirically derived actionable recommendations that can positively influence these metrics. The recommendations were synthesized from 186 peer-reviewed studies spanning two decades, and their mapping to sustainability-related features was conducted through a holistic framework proposed by the authors.

III. System Modules

OSSPREY's architecture comprises four integrated modules that enable real-time sustainability forecasting and actionable recommendation generation.

A. OSS Scraper Module

The scraper module, implemented in Rust, fetches granular monthly activity data for the given GitHub repository using the GraphQL API [4]. The scraper extracts commit-level and issue-level metadata, including commit SHAs, timestamps, lines added and deleted, modified files, issue URLs, and comments. The Scraper module's output is passed to the Network Generator module for further processing.

B. Network Generator Module

This Python module converts scraped data into structured socio-technical networks by constructing two distinct graphs: 1) **Social Network**: It captures communication between contributors. Each node represents a developer, with directed edges from node A to node B created when B responds to A's message (e.g., in GitHub issues or mailing lists). 2) **technical network**: It maps developers to file types, indicating which types of files each developer has contributed to at a given point in time. This network allows us to infer which developers are working on the same modules or components within the project.

Monthly network snapshots generated by the module are stored in structured JSON format in an intermediate file system. These graph networks serve as input to the forecasting module and are also used for generating interactive visualizations, such as Sankey diagrams, within the tool.

[4]https://graphql.org/

C. Forecaster Module

The forecaster module takes socio-technical network graphs as input and outputs the month-wise sustainability probability of a given project. It computes project metrics and features from the networks following the approach proposed by Yin et al. [10], who extracted 16 features from socio-technical network data. We incorporated all ten of their socio-technical features and added three additional characteristics based on theoretical considerations: `s_net_overlap`, `t_net_overlap`, and `st_num_dev`. Here, social features are prefixed with `s_`, technical features with `t_`, and hybrid features with `st_`. A full list of features is available at https://oss-prey.github.io/OSSPREY-Website/#Features.

To make sustainability predictions, we leveraged a transformer-based model developed by Khan et al. [11] (currently under submission), which was trained on the same feature set and achieves a 94% F1-score in predicting GitHub project sustainability. In addition to final classification, the model provides month-wise sustainability probabilities for any given GitHub project.

D. ReACT-Recommender Module

This Python-based recommendation engine implements the ReACTs framework [11], which consists of 105 curated actionables derived from 186 empirical software engineering studies. The module takes a project's socio-technical feature data as input and identifies features that are underperforming, meaning their values fall below the historical average for that project in a given month. For example, if the feature Number of Active Developers (`st_num_dev`) is lower than the project's average for that metric (calculated across all previous months), the module recommends specific actionables known to positively influence `st_num_dev`.

Each recommendation is tagged with an importance level: *Critical*, *Medium*, or *Low*, based on the number of features it is known to impact. An actionable affecting more than five features is labeled as *Critical*; one that impacts between two and five features is tagged as *Medium*; and one that affects fewer than two features is marked as *Low*. The system also links each recommendation to the original research publication from which it was derived.

IV. System Architecture

The system follows a modular pipeline architecture where data flows sequentially from GitHub scraping to sustainability forecasting and actionable recommendation delivery. Raw data collected by the Scraper module is written to an intermediate file system in structured format. This data is then passed to the Network Generator module, which constructs monthly socio-technical networks based on developer communication and collaboration patterns. These network snapshots are used to compute project features that serve as input to the Forecaster

Module, which estimates month-wise sustainability probabilities, and to the ReACT Recommender, which generates tailored, evidence-based interventions.

All frontend–backend interactions are managed through a Flask-based API layer that coordinates execution and logic across modules. This API layer exposes endpoints for data ingestion, feature access, forecast generation, and recommendation delivery, supporting modularity, asynchronous execution, and real-time updates to the user dashboard. A complete list of endpoints and their functionalities can be found at https://oss-prey.github.io/OSSPREY-Website/#API.

The system avoids reliance on a centralized database. Instead, each module writes and reads intermediate outputs from the file system, allowing for loose coupling and greater transparency in the processing pipeline. The backend is containerized using Docker and served behind an NGINX reverse proxy, which ensures secure, scalable, and low-latency access.

V. Dashboard Elements

The dashboard elements of the tool are described as follows:
Project Selector. This component allows users to process any publicly available GitHub project and retrieve month-wise sustainability forecasts, actionable recommendations, and visualizations of the project's socio-technical network, metadata, and features. All other components dynamically update based on the month selected by the user.

Probability of Sustainability. This component displays the month-wise predicted sustainability probability, as computed by the Forecaster module. The line chart visualizes the sustainability trajectory of a selected project, with values ranging from 0 (unsustainable) to 1 (sustainable). The observed sustainability trend up to the selected month is shown in blue. From the selected month onward, the chart branches into three forecasted paths: a green curve for the positive scenario (if the project's health improves), a red curve for the negative scenario (if conditions worsen), and a gray curve for the neutral scenario (if the project continues on its current trajectory). These projections help users understand the potential future paths the project may take based on its socio-technical dynamics.

Project Details. This component shows descriptive metadata, including project name, sponsor organization, official GitHub URL, and key statistics such as stargazers, forks, and watch count. These details contextualize the technical and social indicators shown elsewhere in the dashboard.

Researched Actionables. This element shows the evidence-based interventions suggested based on longitudinal deviations in socio-technical metrics. When sustainability indicators fall below baselines, OSSPREY surfaces relevant recommendations (e.g. increasing documentation efforts, onboarding new contributors) using a filtered subset of peer-reviewed interventions, ranked by relevance and visualized with reference links.

Social Network. This element visualizes developer interactions through issue discussions. Nodes represent contributors; directed edges denote message flow or replies. Edge weight and node size scale with interaction volume. Interactive features include node hovering and filtering, helping identify bottlenecks in communication structures over time.

Technical Network. This element displays commit activity between developers and file types using a bipartite Sankey layout where one side represents developers and the other shows file types (e.g., '.py', '.java', '.md'). Edges reflect contribution patterns, and hovering reveals specific developer-file mappings and relative contributions, helping the user understand code ownership, specialization, and technical load distribution.

VI. Technology Used

OSSPREY adopts a modular, full-stack architecture with technologies chosen for performance and extensibility.

The *frontend* uses *Vue.js* with Composition API for modular and reactive design. *Vite* powers the build system for faster hot-module reloading. Styling uses *Vuetify 3* and SCSS with light/dark mode support. *Pinia* handles state management, *Vue Router* manages routing, and *Vercel* provides deployment with global CDN and CI/CD.

The *backend* employs *Flask* for routing, data orchestration, and API exposure, serving as middleware connecting frontend to backend modules. *Gunicorn* provides asynchronous WSGI HTTP serving, deployed in *Docker* containers with *nginx* reverse proxy and *systemd* resource management.

The *scraping layer* uses *Rust* for execution speed, memory safety, and zero-cost abstractions. It uses GitHub GraphQL API with *Rayon* library enabling parallel scraping for hundreds of repositories.

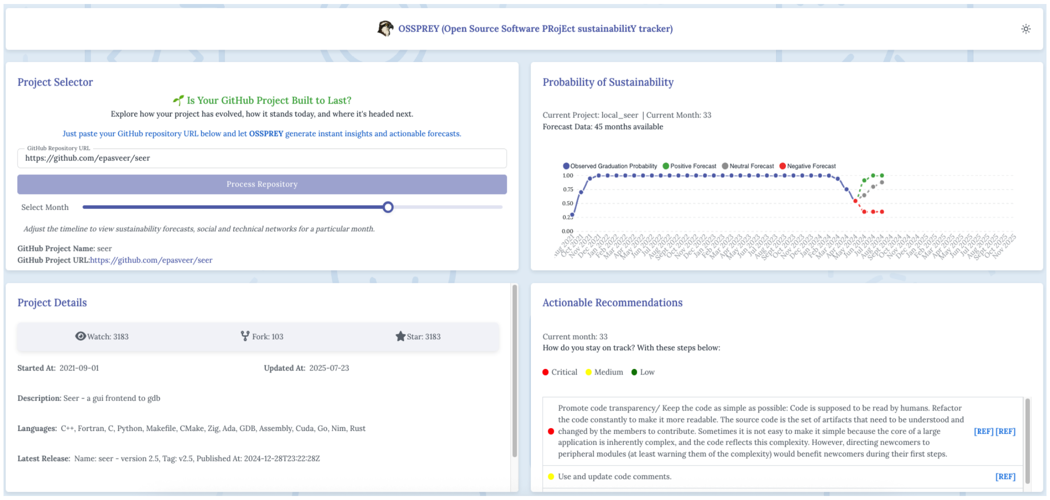
Analytics and forecasting employ *Python* with *PyTorch* for sustainability forecasting module. The *ReACT-Recommender* operates as a standalone Python service that processes monthly data to return project-specific actionables based on evidence in JSON format.

VII. Use Cases

OSSPREY addresses critical challenges in OSS sustainability through two core applications:

Real Time Monitoring. OSSPREY enables continuous, month-by-month tracking of project metrics and sustainability probabilities, allowing maintainers to observe progress over time. The tool also visualizes future sustainability forecasts, helping developers anticipate potential downturns or improvements and plan timely interventions. Maintainers can also use OSSPREY to monitor evolving socio-technical patterns, such as changes in developer activity, communication dynamics, and collaboration structures.

Evidence Based Decision Support. OSSPREY provides maintainers with actionable recommendations that are grounded in empirical software engineering research. Each recommendation is linked to the original research publication to ensure transparency and traceability. These evidence-based actionables can guide maintainers in adopting effective strategies to improve long-term project sustainability.


OSSPREY (Open Source Software PRojEct sustainabilitY tracker)

Project Selector

Is Your GitHub Project Built to Last?

Explore how your project has evolved, how it stands today, and where it's headed next.

Just paste your GitHub repository URL below and let **OSSPREY** generate instant insights and actionable forecasts.

GitHub Repository URL: https://github.com/epasveer/seer

Adjust the timeline to view sustainability forecasts, social and technical networks for a particular month.

GitHub Project Name: seer
GitHub Project URL: https://github.com/epasveer/seer

Probability of Sustainability

Current Project: local_seer | Current Month: 33
Forecast Data: 45 months available

Process Repository
Select Month
● Observed Graduation Probability
● Positive Forecast
● Neutral Forecast
● Negative Forecast
1.00
0.75
0.50
0.25
0.00
Aug 2021 Oct 2021 Nov 2021 Dec 2021 Jan 2022 Feb 2022 Mar 2022 Apr 2022 May 2022 Jun 2022 Jul 2022 Aug 2022 Sept 2022 Oct 2022 Nov 2022 Dec 2022 Jan 2023 Feb 2023 Mar 2023 Apr 2023 May 2023 Jun 2023 Jul 2023 Aug 2023 Sept 2023 Oct 2023 Nov 2023 Dec 2023 Jan 2024 Feb 2024 Mar 2024 Apr 2024 May 2024 Jun 2024 Jul 2024 Aug 2024 Sept 2024 Oct 2024 Nov 2024 Dec 2024 Jan 2025 Feb 2025 Mar 2025 Apr 2025 May 2025 Jun 2025 Jul 2025 Aug 2025 Sept 2025 Oct 2025 Nov 2025

Project Details

Watch: 3183 Fork: 103 Star: 3183

Started At: 2021-09-01 **Updated At:** 2025-07-23

Description: Seer - a gui frontend to gdb

Languages: C++, Fortran, C, Python, Makefile, CMake, Zig, Ada, GDB, Assembly, Cuda, Go, Nim, Rust

Latest Release: Name: seer - version 2.5, Tag: v2.5, Published At: 2024-12-28T23:22:28Z

Actionable Recommendations

Current month: 33
How do you stay on track? With these steps below:

● Critical ● Medium ● Low

● Promote code transparency/ Keep the code as simple as possible: Code is supposed to be read by humans. Refactor the code constantly to make it more readable. The source code is the set of artifacts that need to be understood and changed by the members to contribute. Sometimes it is not easy to make it simple because the core of a large application is inherently complex, and the code reflects this complexity. However, directing newcomers to peripheral modules (at least warning them of the complexity) would benefit newcomers during their first steps. [REF] [REF]

● Use and update code comments. [REF]

Fig. 1. Screenshot of OSSPREY Dashboard

VIII. Discussion & Conclusion

While OSSPREY presents a comprehensive framework for forecasting and improving the sustainability of OSS projects, several limitations remain that offer opportunities for future enhancement. First, the system relies on a fixed set of socio-technical features derived from prior studies, which may not capture the unique dynamics of all GitHub-hosted repositories, especially those with unconventional governance structures or sparse contributor activity. Incorporating adaptive modeling techniques or allowing user-defined features could improve flexibility and generalizability. Second, the ReACT Recommender uses static mappings between features and interventions. Although grounded in empirical evidence, these mappings do not consider project-specific factors such as team size, technology stack, or historical context. Personalized, context-aware recommendations powered by large language models or retrieval-augmented generation could provide more nuanced guidance. Third, infrastructure performance may degrade for large repositories (e.g., with more than 5,000 commits or 300 contributors) due to threading and memory constraints. This bottleneck could be addressed through improved parallelization and cloud-based task management. Finally, OSSPREY analyzes projects independently and does not consider dependencies between them. Extending the system to capture upstream and downstream relationships could offer deeper insights into ecosystem-wide risks and sustainability.

Despite these limitations, OSSPREY makes significant strides in bridging the gap between research-driven sustainability metrics and practical decision-making for OSS maintainers. By integrating real-time GitHub scraping, socio-technical network analysis, machine learning-based forecasting, and empirically validated actionable recommendations, OSSPREY enables proactive governance and targeted intervention. Unlike prior efforts, it delivers forward-looking sustainability insights and links them to concrete, evidence-based strategies tailored to project needs. Its modular and extensible architecture supports diverse use cases, from monitoring project health trajectories to empowering foundation stakeholders with governance tools. OSSPREY thus lays a critical foundation for sustainable software analytics and the development of resilient, community-driven open source ecosystems.

Acknowledgments

This research was supported by the National Science Foundation under Grant No. 2020751, as well as by the Alfred P. Sloan Foundation through the OSPO for UC initiative (Award No. 2024-22424).

References

[1] P. Hunter and S. Walli, "The rise and evolution of the open source software foundation," *IFOSS L. Rev.*, vol. 5, p. 31, 2013.

[2] Apache Software Foundation, "Incubator clutch status," https://incubator.apache.org/clutch/, 2025, accessed: 2025-05-09.

[3] C. M. Schweik and R. C. English, *Internet success: a study of open-source software commons*. MIT Press, 2012.

[4] L. Yin, Z. Chen, Q. Xuan, and V. Filkov, "Sustainability forecasting for apache incubator projects," in *Proceedings of the 29th ACM joint meeting on european software engineering conference and symposium on the foundations of software engineering*, 2021, pp. 1056–1067.

[5] A. Ramchandran, L. Yin, and V. Filkov, "Exploring apache incubator project trajectories with apex," in *Proceedings of the 19th international conference on mining software repositories*, 2022, pp. 333–337.

[6] N. I. Khan and V. Filkov, "From models to practice: Enhancing oss project sustainability with evidence-based advice," in *Companion Proceedings of the 32nd ACM International Conference on the Foundations of Software Engineering*, 2024, pp. 457–461.

[7] J. Wu, K.-Y. Goh, and Q. Tang, "Investigating success of open source software projects: A social network perspective," *ICIS 2007 Proceedings*, p. 105, 2007.

[8] W. Scacchi, "Socio-technical interaction networks in free/open source software development processes," in *Software process modeling*. Springer, 2005, pp. 1–27.

[9] L. Yin, M. Chakraborti, Y. Yan, C. Schweik, S. Frey, and V. Filkov, "Open source software sustainability: Combining institutional analysis and socio-technical networks," *Proceedings of the ACM on Human-Computer Interaction*, vol. 6, no. CSCW2, pp. 1–23, 2022.

[10] L. Yin, X. Zhang, and V. Filkov, "On the self-governance and episodic changes in apache incubator projects: An empirical study," in *2023 IEEE/ACM 45th International Conference on Software Engineering (ICSE)*. IEEE, 2023, pp. 678–689.

[11] N. I. Khan, A. Ashok, S. Singhvi, Ștefan Stănciulescu, and V. Filkov, "Socio-technical networks-based modeling of open source software project sustainability," 2025, under review.