Designing Abandabot:
When Does Open Source Dependency Abandonment Matter?

Wednesday, April 15th, 2026

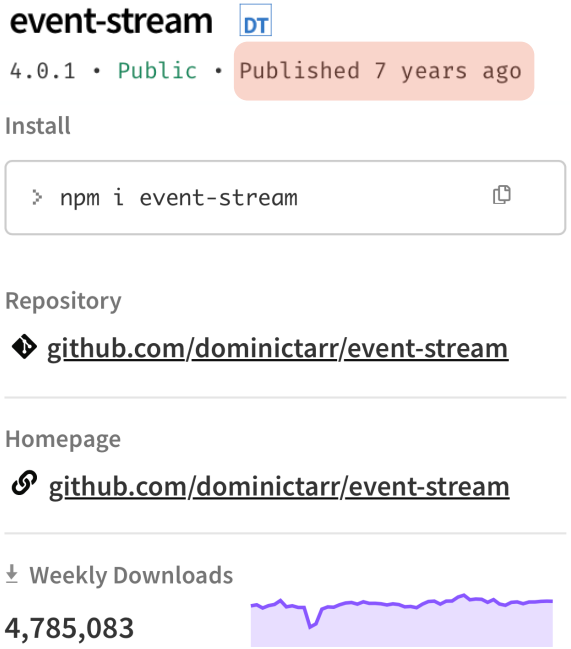
Courtney Miller*, Hao He*, Weigen Chen, Elizabeth Lin, Chenyang Yang, Bogdan Vasilescu, Christian Kästner
* Both authors contributed equally to this research

Paper

Package Abandonment Can Disrupt Supply Chain Integrity For Downstream Users

Abandonment Exposes Users to Risk of:

- Unpatched vulnerabilities
- Costly unplanned migrations
- Targeted supply chain attacks



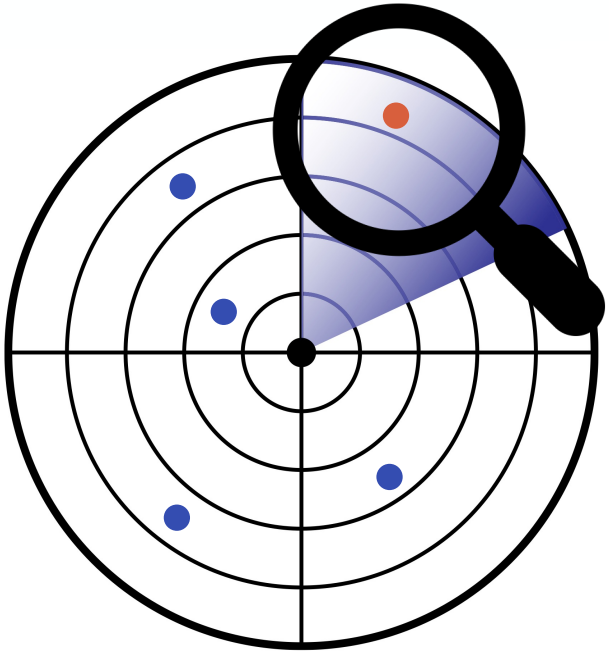
Identifying & Remediating Abandoned Dependencies is Considered Best Practice*

Mitigate Complacency in Maintenance:

- Implement policies that enforce regular reviews and updates of all open source dependencies, particularly those neglected for over a year. This approach will combat latent risks and reduce the chances of introducing vulnerabilities into your software.

* 2024 Sonatype State of the Software Supply Chain Report

There Is An Unmet Need For Automated Abandonment Detection Tooling At Scale



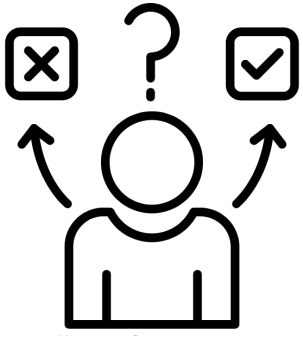
Abandabot

~~Dependabot~~

Maintained Dependency Abandoned Dependency

Reflection On Lessons From Broader Challenges With Automated Tooling For Software Engineering

Too many notifications, especially ones considered spurious, can lead to notification fatigue and tool disengagement



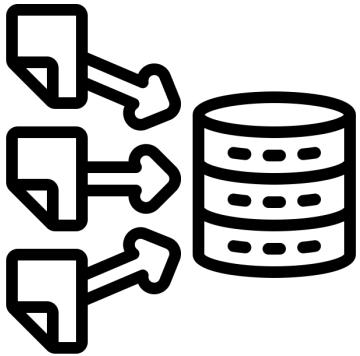
Manual configuration is possible but often impractical due to the overhead required

Goal: Develop intelligent pre-configuration that predicts which abandonment will be disruptive to a project, given the context of their dependency usage

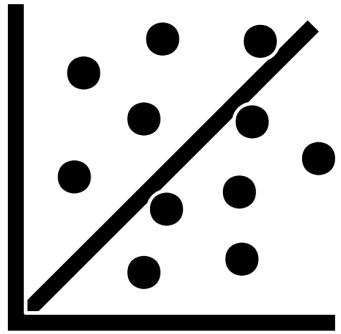
But how?

Idea #1: Traditional Approach

Develop theory & operationalize factors



Collect extensive training data



Build conventional machine learning model

Downsides:
Difficulty operationalizing nuanced context-specific factors at scale
Extremely time intensive

Idea #2: Naive LLM Approach

0.58 Macro-F1 score

Zero-shot LLM Baseline

Our Solution:
Prediction Through A Hybrid Approach

0.58 Macro-F1 score

Zero-shot LLM Baseline

0.75 Macro-F1 score

Theory-Driven Reasoning + Context Approach

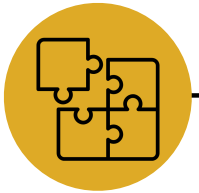
Hypothesis: Large language models (LLMs), equipped with theory-driven reasoning and contextual information, can accurately predict the impact of abandonment disruptions better than LLMs alone to support developer decision-making

Method

- Extract expert knowledge through need finding interviews & develop theoretical framework
- Operationalize & extract theory-driven context
- Develop theory-driven structured reasoning prompt
- Build classifier
- Perform independent user study evaluating classifier performance

Extracting Expert Knowledge And Developing Theoretical Framework

- **Step 1**. Performed semi-structured *need-finding interviews* with 22 developers
- **Step 2**. Analyzed data using iterative thematic analysis to develop theoretical framework, which informed theory-driven reasoning & context developed

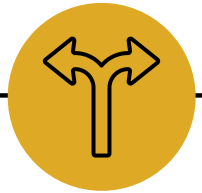


Importance of Functionality

How important is the functionality provided by the dependency to the project?

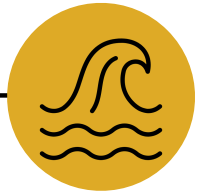
Depth of Integration

How difficult is it to replace the dependency, considering the depth of its integration in the project's code base?



Availability of Alternatives

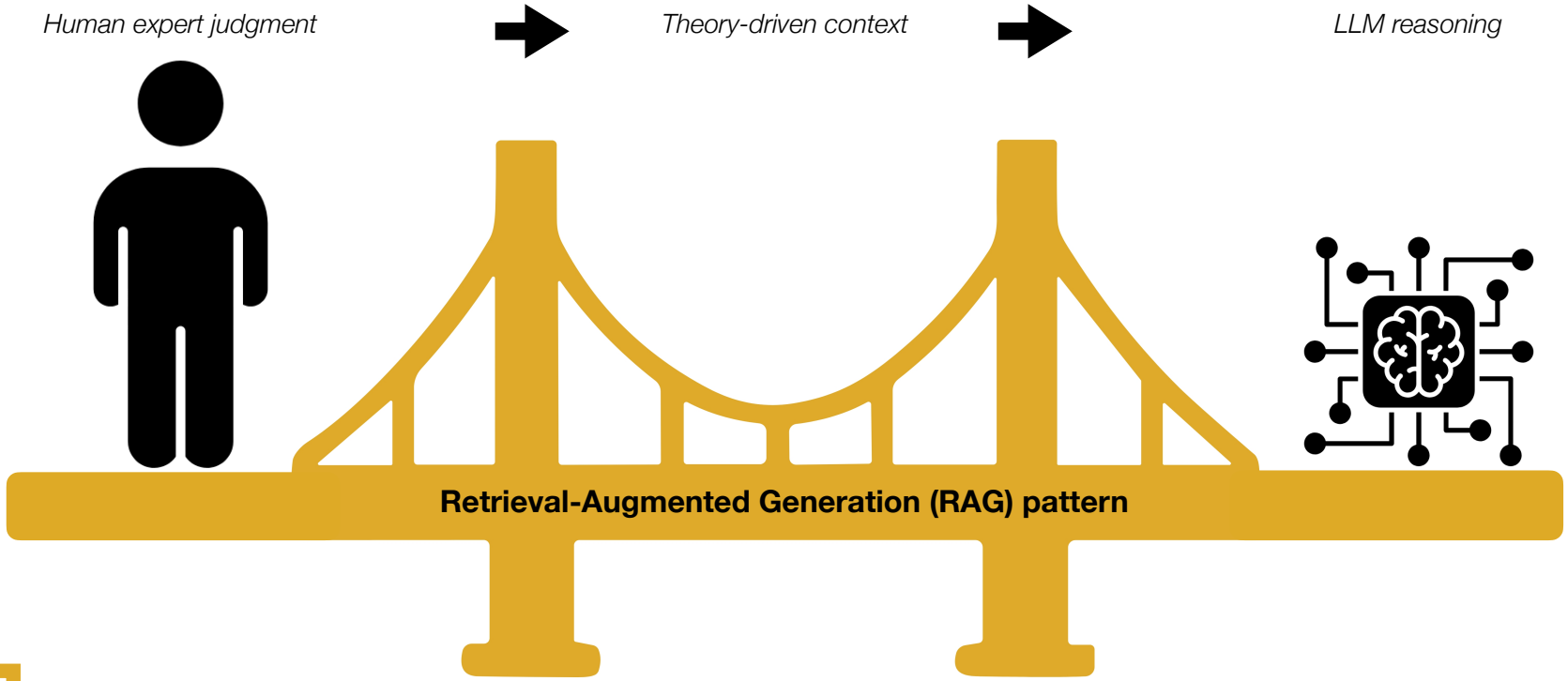
How difficult is it to replace the dependency, considering the availability of suitable alternatives?



External Environmental Pressures

How likely is it that external environmental changes will force the project to act on the dependency's abandonment?

Operationalize And Extract Theory-Driven Context



Operationalize And Extract Theory-Driven Context

Theoretical Framework

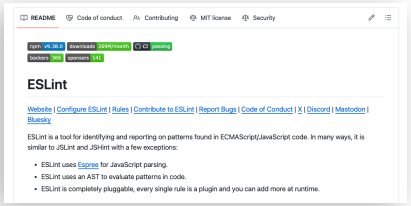
Importance of Functionality

External Environmental Pressures

Availability of Alternatives

Depth of Integration

Project README

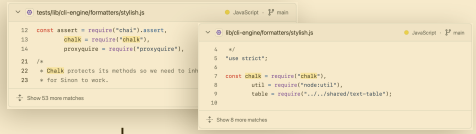


Project package.json file

Dependency README

LLM knowledge

Relevant usage code snippets



static code analysis via CodeQL (keyword search & global dataflow analysis)

Theory-Driven Prediction Architecture

Project: nasa/juggling-club
Dependency: puppeteer

Extract Contextual Information

Project README

 # Web specifications

This repository contains a curated list of technical Web specifications. The list is used in a variety of ways...

Dependency README

Puppeteer is a JavaScript library which provides a high-level API to control Chrome or Firefox over the ...

package.json

```
"devDependencies": { "mocha": "^11.1.0",
  "puppeteer": "^24.2.1", "rimraf": "^6.0.1”}
```

Usage Context

```
[start of src/find-specs.js]
3 import puppeteer from 'puppeteer';
...
133   const browser = await puppeteer.launch();
134   try {
135     const page = await browser.newPage();
136     proposals = await page.evaluate(...);
...
145   finally {
146     await browser.close();
[end of src/find-specs.js]
```

Theory-Based Prompting

You are an expert JavaScript developer building a tool to notify a project's maintainers when one of the project's dependencies becomes abandoned. However, instead of notifying them when any of their dependencies are abandoned, you want to only notify them about the abandonment of dependencies that are likely important and impactful to the project given the context of their dependency usage, so as to minimize notification fatigue.

I am going to ask you important dependency management questions regarding whether a dependency's future hypothetical abandonment is likely to be impactful and therefore noteworthy to the project. For context, I will provide you with [context] I want you to answer the following four
questions based on the information I provided:

1. How important is the functionality provided by the dependency to the project?
2. How difficult is it to replace the dependency, considering the depth of its integration in the project's code base?
3. How difficult is it to replace the dependency, considering the availability of alternative packages that could serve as suitable replacements and provide the same functionality?
4. How likely is it that external environmental changes will force the project to act on the dependency's abandonment?

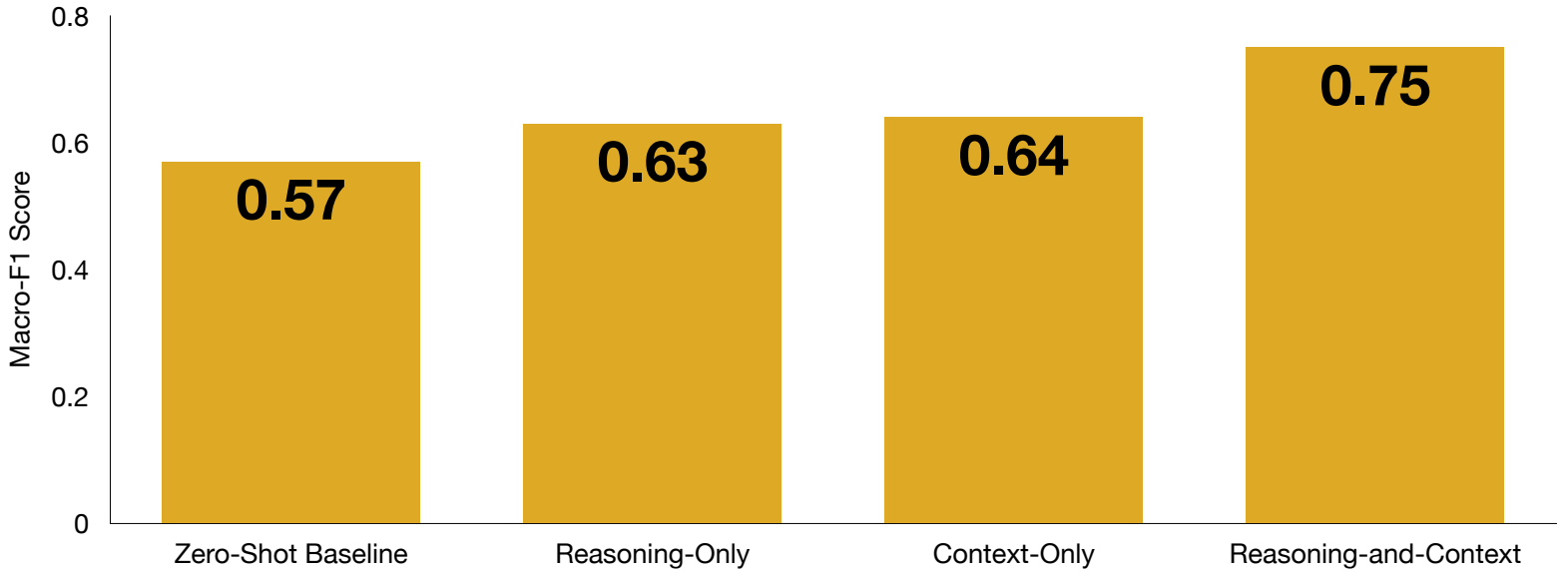
For each question, I want you to provide a score on a scale from 1 to 5, where 1 is the least important, difficult, or likely, and 5 is the most important, difficult, or likely. Along with the score, please provide detailed, specific reasoning behind the score. Your response for each question should be placed in the top-level "importance", "integration", "alternatives", and "likelihood" fields of your JSON response, respectively. For each of these fields, you should provide a "reasoning" field with your reasoning.

..... [rest of prompt omitted]

Evaluation & Prediction

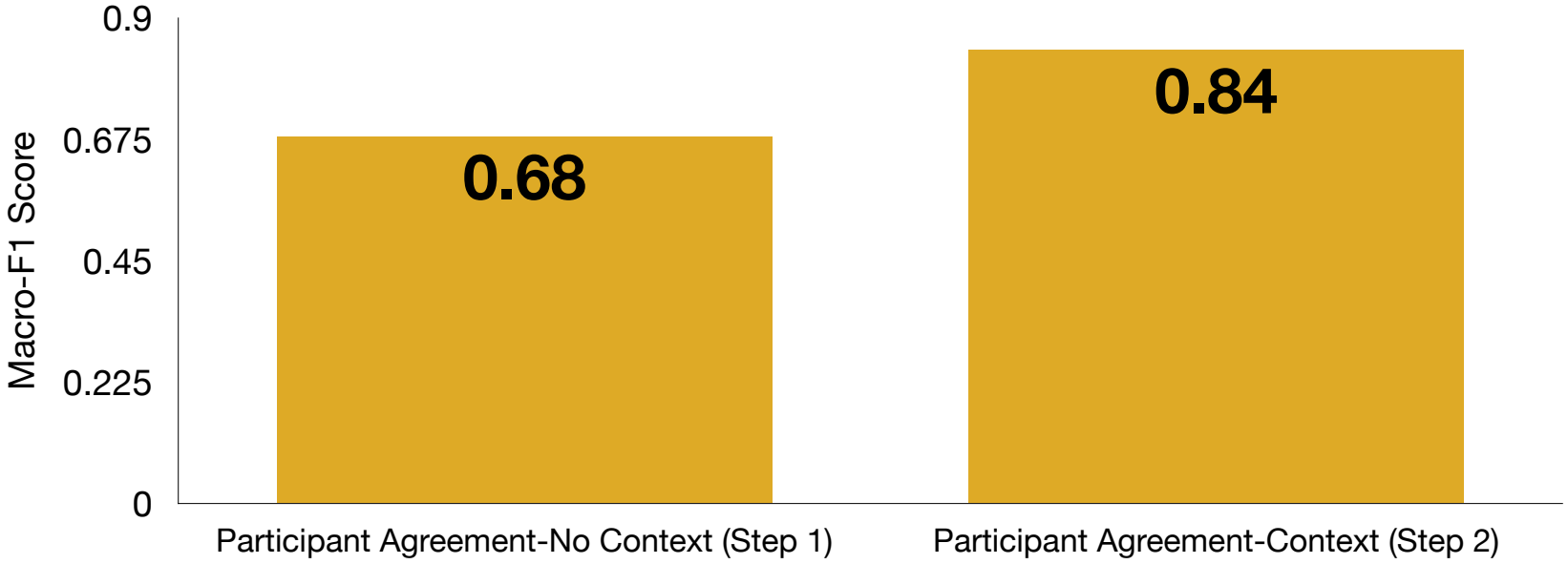
```
{
  "importance": {
    "reasoning": "Puppeteer is used in multiple critical
parts of the project...",
    "score": 5
  },
  "integration": {
    "reasoning": "Puppeteer is deeply integrated into
the project's codebase...",
    "score": 5
  },
  "alternatives": {
    "reasoning": "While alternative such as Playwright
or Selenium exist, each has its own set of features and
limitations...",
    "score": 3
  },
  "likelihood": {
    "reasoning": ”Shifts in browser automation
technologies...",
    "score": 2
  },
  "reasoning": "Considering the critical role Puppeteer
plays in the project's operations, its deep integration
into the codebase, the challenges associated with
finding and transitioning to a suitable alternative, the
abandonment of Puppeteer would likely be directly
impactful to the project.",
  "impactful": true
}
```

Combined Theory-Driven Reasoning And Context Approach Performance



Key Finding:
Abandabot's Predictions Are Effective!

Ran online survey (n=124)



In Summary

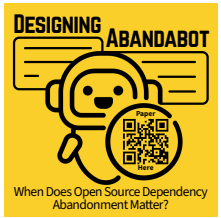
Designing Abandabot:

When Does Open Source Dependency Abandonment Matter?

Courtney Miller*, Hao He*, Weigen Chen, Elizabeth Lin, Chenyang Yang, Bogdan Vasilescu, Christian Kästner
* Both authors contributed equally to this research

- Contextual factors influence impact of abandonment: depth of integration, availability of alternatives, importance of functionality, and external environmental pressures
- Developers wanted intelligent pre-configuration that predicts which abandonment is impactful
- LLM-based classifier using theory-driven reasoning and project-specific context can effectively approximate developer judgments of impact better than zero-shot baseline

Paper



We have stickers!
←

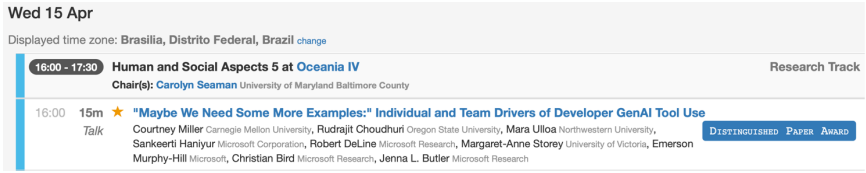
In Summary

Designing Abandabot:

When Does Open Source Dependency Abandonment Matter?

Courtney Miller*, Hao He*, Weigen Chen, Elizabeth Lin, Chenyang Yang, Bogdan Vasilescu, Christian Kästner
* Both authors contributed equally to this research

- Contextual factors influence impact of abandonment: depth of integration, availability of alternatives, importance of functionality, and external environmental pressures
- Developers wanted intelligent pre-configuration that predicts which abandonment is impactful
- LLM-based classifier using theory-driven reasoning and project-specific context can effectively approximate developer judgments of impact better than zero-shot baseline


Wed 15 Apr

Displayed time zone: Brasilia, Distrito Federal, Brazil change

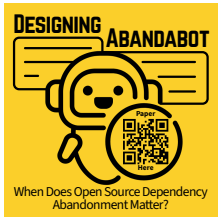
16:00 - 17:30 Human and Social Aspects 5 at Oceania IV — Research Track

Chair(s): Carolyn Seaman University of Maryland Baltimore County

16:00 15m Talk ★ "Maybe We Need Some More Examples:" Individual and Team Drivers of Developer GenAI Tool Use — Distinguished Paper Award

Courtney Miller Carnegie Mellon University, Rudrajit Choudhuri Oregon State University, Mara Ulloa Northwestern University, Sankeerti Haniyur Microsoft Corporation, Robert DeLine Microsoft Research, Margaret-Anne Storey University of Victoria, Emerson Murphy-Hill Microsoft, Christian Bird Microsoft Research, Jenna L. Butler Microsoft Research

Paper



We have stickers! ←

courtney-e-miller.github.io @CourtneyElta

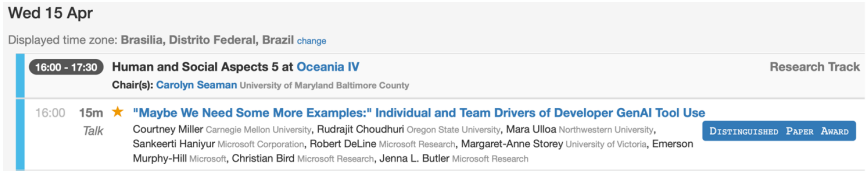
In Summary

Designing Abandabot:

When Does Open Source Dependency Abandonment Matter?

Courtney Miller*, Hao He*, Weigen Chen, Elizabeth Lin, Chenyang Yang, Bogdan Vasilescu, Christian Kästner
* Both authors contributed equally to this research

- Contextual factors influence impact of abandonment: depth of integration, availability of alternatives, importance of functionality, and external environmental pressures
- Developers wanted intelligent pre-configuration that predicts which abandonment is impactful
- LLM-based classifier using theory-driven reasoning and project-specific context can effectively approximate developer judgments of impact better than zero-shot baseline


Wed 15 Apr

Displayed time zone: Brasilia, Distrito Federal, Brazil change

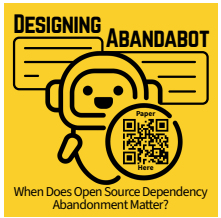
16:00 - 17:30 Human and Social Aspects 5 at Oceania IV — Research Track

Chair(s): Carolyn Seaman University of Maryland Baltimore County

16:00 15m Talk ★ "Maybe We Need Some More Examples:" Individual and Team Drivers of Developer GenAI Tool Use — Distinguished Paper Award

Courtney Miller Carnegie Mellon University, Rudrajit Choudhuri Oregon State University, Mara Ulloa Northwestern University, Sankeerti Haniyur Microsoft Corporation, Robert DeLine Microsoft Research, Margaret-Anne Storey University of Victoria, Emerson Murphy-Hill Microsoft, Christian Bird Microsoft Research, Jenna L. Butler Microsoft Research

Paper



We have stickers! ←

Life Update:

Carnegie Mellon University → THE GEORGE WASHINGTON UNIVERSITY WASHINGTON, DC

courtney-e-miller.github.io @CourtneyElta