A Second Look at the Dynamics of the JavaScript Package Ecosystem

Kevin de Haan, Gregory Neagu, Frederic Sauve-Hoover, Abram Hindle
Department of Computing Science
University of Alberta
Edmonton, Canada
Email: {kdehaan,neagu,rsauveho,abram.hindle}@ualberta.ca

***Abstract*—In recent years, the tools and packages most commonly involved with JavaScript development have evolved rapidly. Newer packages such as Angular and React have experienced a marked increase in popularity among developers, while frameworks such as jQuery have begun to phase out.[1] For this reason we take a second look at a 2016 paper by Wittern, Suter and Rajagopalan [1] to see what aspects of the JavaScript package ecosystem have changed, and if previously observed trends have remained constant. In the original paper the authors use the *node package manager* (npm) to gain insight into the JavaScript ecosystem as a whole, and data from projects publicly hosted on GitHub to observe an alternative measure of popularity. We adhere to the same methods of analysis, and extend the data to capture more recent information up to April 1st 2019. Ultimately, this second look aims to discover if recent years have had any significant effects on ecosystem-wide trends, and provide developers with further insight into how packages are used and evolve.**

I. Introduction

This paper is a replication of *A Look at the Dynamics of the JavaScript Package Ecosystem* [1] that performs extensive analysis of the *node package manager* (npm), a popular distributor of JavaScript-based packages. Since the publishing of the original paper, the usage and scale of npm has only grown. With more than three times as many hosted packages (over 750,000 as of April 1st 2019) and over ten times as many weekly package downloads (now over ten billion per week), the raw volume of data and the complexity of package dependency graphs has increased significantly. Additionally, the major frameworks used in JavaScript development have undergone a rapid transformation as packages such as Angular and React are adopted[1]. The core contributions we make are as follows:

- We replicate and verify the results found in the original paper for the window of October 1st 2010 to September 1st 2015.
- We extend the analysis to the time period of September 2nd 2015 to April 1st 2019, and evaluate whether patterns and trends noted in the original paper are still observable.
- We investigate whether the continued evolution of the JavaScript package ecosystem has affected the relationships between various measures of package popularity.
- We determine if the ongoing maturation of the JavaScript ecosystem has resulted in tangible changes to version numbering or adoption practices.

II. Related Work

III. Data Collection

The window of data analyzed within this paper is October 1st 2010 (as in the original paper) to April 1st 2019. We collected from three publicly available data sets. Two of these, the npm registry and the GitHub repository platform, are from the same source as in the original paper. To find repositories relying npm, we used the Google BigQuery github_repos data set, updated weekly[2]. By using this set we are able to analyze GitHub data without being constrained to the currently available window provided by the GHTorrent project [2]. The final data set encompasses 797,940 packages and **#VALUE** applications.

A. Package Metadata

this one is probably mostly for @Fred

B. Applications using npm Packages

@Greg I'm gonna need you to put something here

IV. Ecosystem Evolution

V. Package Popularity

A. Relationships between Measures

B. Distinct Package Types

C. Popularity Over Time

1) Identifying Top Packages:

2) Popular Package Dynamics:

3) Comparing Popularities:

[1] https://insights.stackoverflow.com/survey/2016#technology-most-popular-technologies, https://insights.stackoverflow.com/survey/2017#technology-_-frameworks-libraries-and-other-technologies, https://insights.stackoverflow.com/survey/2018#technology-_-frameworks-libraries-and-tools

[2] https://github.com/fhoffa/analyzing_github/

VI. Version Numbering and Package Adoption

A. Attribution of Version Numbers

B. Adoption by Version Number

VII. Conclusion

References

[1] Erik Wittern, Philippe Suter, and Shriram Rajagopalan. A look at the dynamics of the javascript package ecosystem. In *Proceedings of the 13th International Conference on Mining Software Repositories*, MSR '16, pages 351–361, New York, NY, USA, 2016. ACM.

[2] Georgios Gousios. The ghtorrent dataset and tool suite. In *Proceedings of the 10th Working Conference on Mining Software Repositories*, MSR '13, pages 233–236, Piscataway, NJ, USA, 2013. IEEE Press.