Dead Links and Lost Code: Investigating the State of Source Code Repositories in Maven Central Packages

Author: Tudor-Gabriel Velican
t.velican@student.tudelft.nl

Supervisor: Mehdi Keshani
Responsible Professor: Sebastian Proksch

(1) Introduction

With ~11M indexed packages, Maven Central is the de-facto source of free and open-source Java libraries for many developers.

Reproducible source code is crucial to maintaining trust and accountability on Maven Central [1]. Moreover, developers need to have the guarantee that a library is in active development and has a thriving community before integrating it into their projects [2]. This can only be achieved if developers:

- Provide a publicly accessible source code repository URL;
- Make explicit which published version corresponds to which commit;
- Configure the build process to be reproducible.

We explore the extent to which developers follow these practices when publishing packages to Maven Central.

(2) Research Questions

RQ1: How reliable are the repository links?

RQ2: Where are the repositories hosted and how does this change over time in the ecosystem?

RQ3: Can the commit pertaining to a specific release be pinpointed?

RQ4: How reproducible are the packages? Can one rebuild the packages with the same checksum?

(3) Data Selection

Out of all ~11M package releases in the Maven Central index, we randomly sampled a single version from each package, so as to ensure adequate representation from each year.

Selecting a single version from each package yielded a sample size of 473,352 packages.

(4) Methodology

RQ1: Extracted the source code repositories provided by library authors and verified whether they were publicly accessible.
RQ2: Extracted hostnames from URL fields, aggregating this data to analyze the market share of various repository hosting services over time.
RQ3: Checked whether the artefact versioning on Maven Central was consistent with the releases/tags in the repository.
RQ4: Compared the checksum of the published artefacts with the checksums obtained after building from source.

(5) Results

- 98.7% of packages specified a repository URL. At the same time, only 80.28% of all packages had at least 1 valid Git repository URL.
- Github was the most popular repository host, with a market share exceeding 90% most years. Alternative hosts such as Gitee and Apache Gitbox have been gradually increasing in popularity in the last few years. Figure 3 shows the market share of repository hosts defined in the correct URL field (*scm.url*). Note however, that we found 3 other fields in which developers often defined the repository URL, namely (*url, scm.connection* & *scm.devConnection*)
- Out of the 360,086 packages with a valid Github repository, 74.35% were found to have a release/tag closely resembling the version name. 24.45% had a release/tag with the same name as the version.

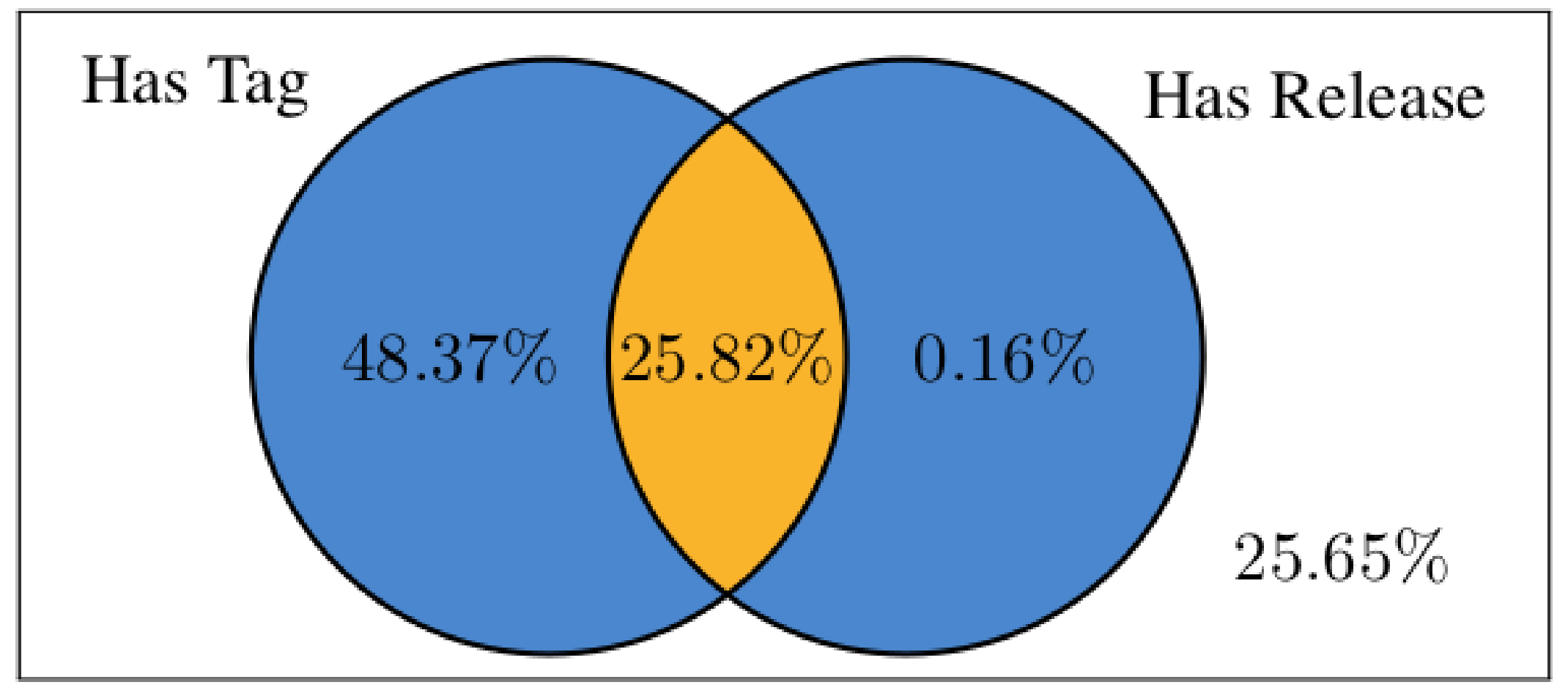

Fig. 1 - packages with releases/tags

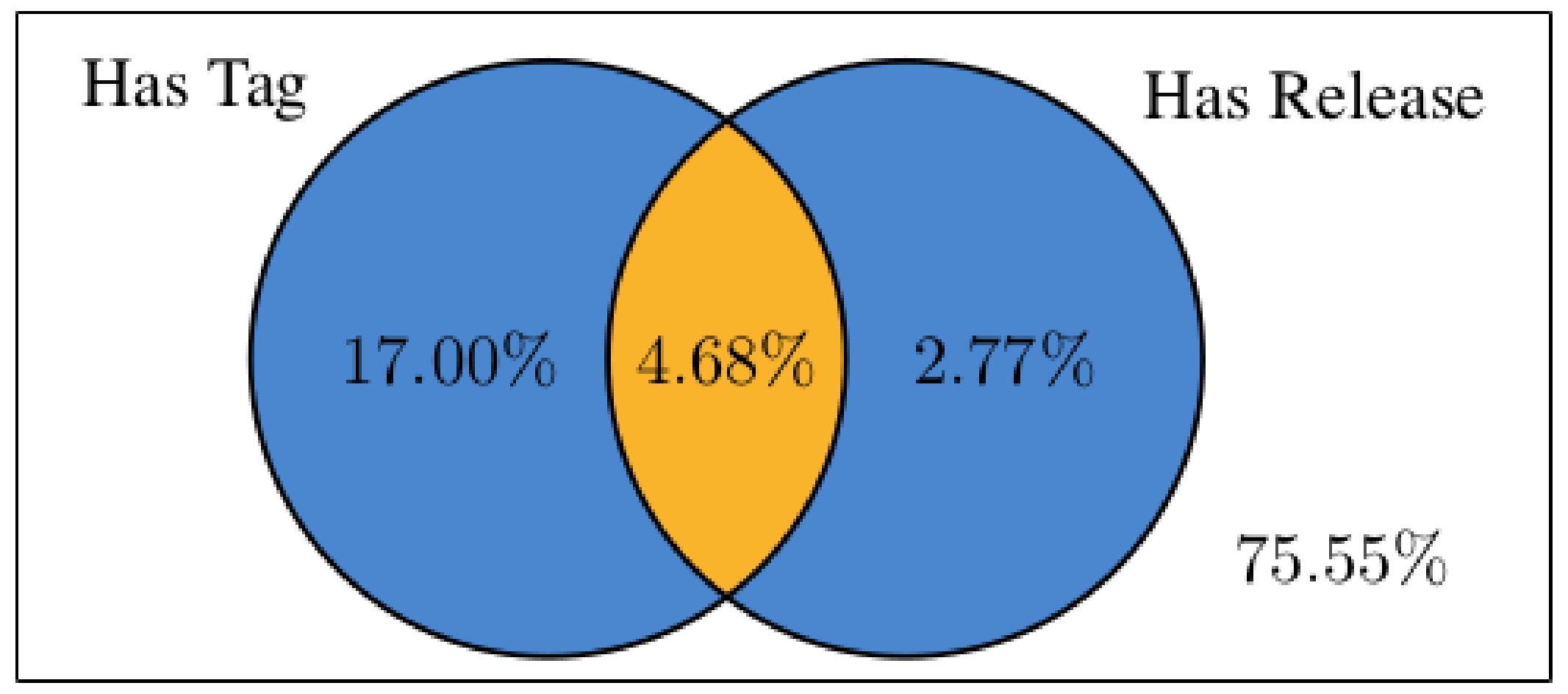

Fig. 2 - packages with releases/tags with the same name as the version

Figure 1 shows the proportion of tags and releases found, whilst Figure 2 shows it only for packages where the version name exactly matched the release/tag name.

- Out of the packages with a corresponding tag/release, only 9603 packages were configured to enable reproducibility. We attempted to build 481 packages, of which 230 could be built successfully using Maven's default build configuration. 37 packages were completely reproducible, 191 only partially reproducible and 2 were not reproducible at all.

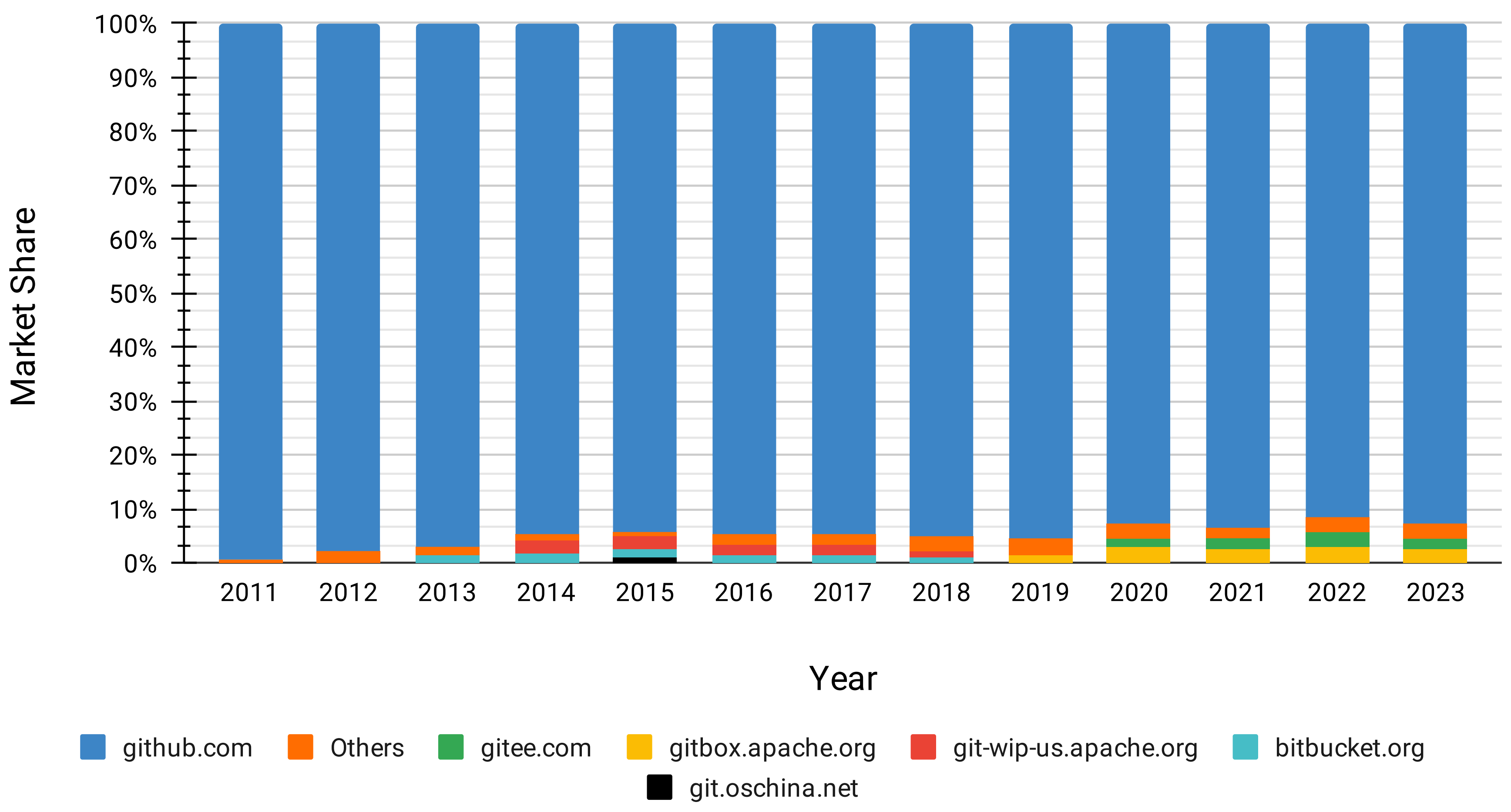

Fig. 3 - market share of repository hosts defined in scm.url field

(6) Conclusions

- There was inconsistency in how project URLs were provided, and some packages even had missing mandatory fields, raising concerns about the strictness of Maven Central's submission guidelines.
- Github emerged as the dominant repository host, with an overwhelming 90% market share.
- About 74% of packages used tags or releases, but naming conventions differed between Maven Central versions and source code repositories, posing difficulties for developers in locating correct versions and assessing reproducibility.
- Reproducibility was a significant concern: only 3% of packages had the necessary configuration for ensuring reproducibility, and of those that were buildable, only 16% achieved full reproducibility, indicating a lack of knowledge or consideration by developers.

The study suggests a need for stricter submission guidelines and validation mechanisms for repository links in Maven Central, and increased attention to reproducibility and consistent naming conventions.

References

[1] C. Lamb and S. Zacchiroli, Reproducible builds: Increasing the integrity of software supply chains," IEEE Software, vol. 39, no. 2, pp. 62–70, 2022.

[2] N. Haenni, M. Lungu, N. Schwarz, and O. Nierstrasz, "Categorizing developer information needs in software ecosystems," in Proceedings of the 2013 international workshop on ecosystem architectures, 2013, pp. 1–5.