Can we extract a relevant, available, and self-contained core of the Maven ecosytem?

Extracting the pillars of the community, and their dependencies.

Author
Mathijs van der Schoot

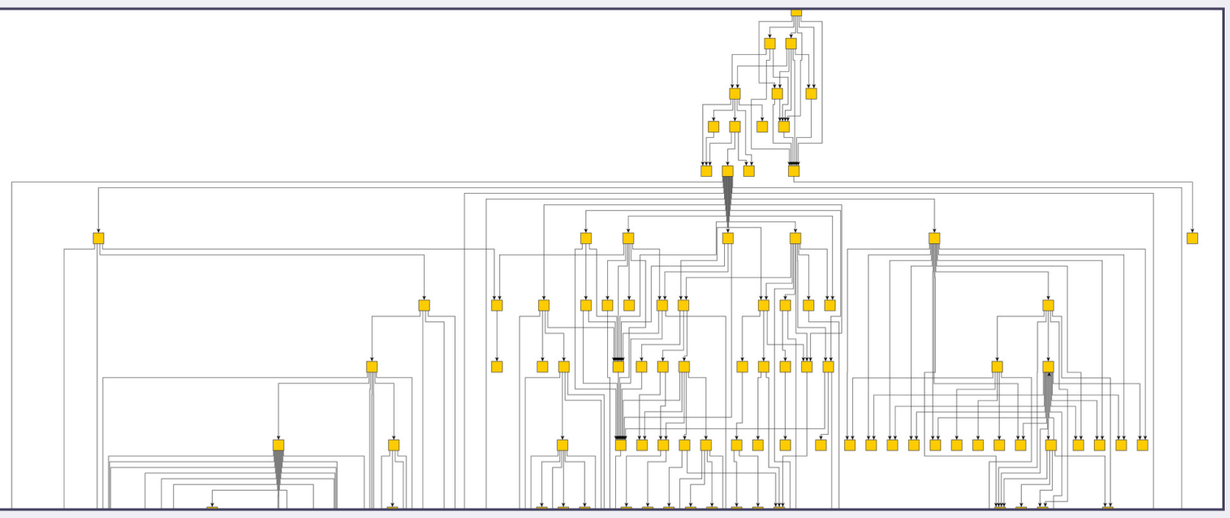
Supervisors
Supervisor & Responsible Professor:
Dr. Ing. Sebastian Proksch

Affiliations
TU Delft Software Engineering Research Group

Introduction

The Maven ecosystem is cluttered.
We extract the pillars in the community such that:

- Qualitative analysis can be performed on it, so developers have quality guarantees about their dependencies.
- Future researchers can reproduce and adapt this method to their use case, using the information in this paper.



Objective

Construct a method to create a relevant, available and self-contained core, using sub-questions about:

- Library usage count
- Libraries only referenced by other packages of the same developers
- A usage threshold filter

Methodology

Construct a core using a pipeline:

- Maven-explorer: traverses Maven Central indices & downloads packages
- Maven Dependency Plugin: constructs a list of dependencies for every package
- Graph: Use the lists to create a graph structure, and analyse it

Results

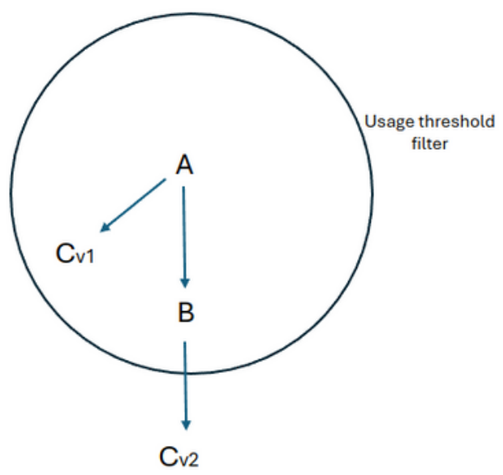
85% of libraries are not used by anyone except their own developers.

3.3% of the libraries account for **two-thirds** of all dependencies.

Analysis

This paper explores the nuances involved in the creation of a core:

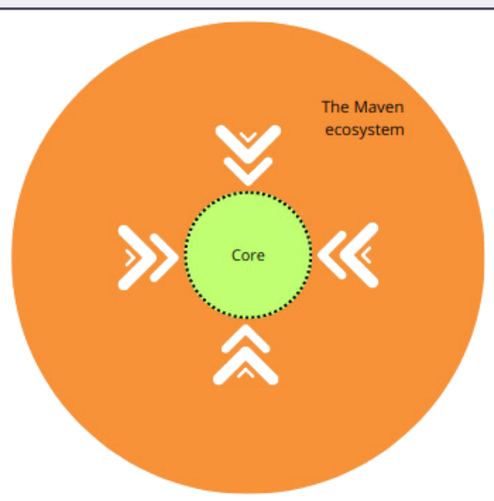
- Dependency types?
- Dependency transitivity? And with varying types?
- Multiple versions of the same library?
- What does it mean to be self-contained?
  - Library C version 2 does not pass the filter
  - Library B is now not self-contained anymore
  - And what about library A?
  - It depends on library C version 1, not 2
- What usage threshold is optimal for a particular use-case?



Library A and B pass a usage threshold filter. Library C version 1 does too, but version 2 does not. How to tackle this?

Threshold	Data set size
0	*47,152 100%*
1	*7,119 15%*
2	*4,656 10%*
5	*2,542 5%*
10	*1,527 3%*
50	*415 1%*

How the data set size is reduced by a usage threshold filter.



Conclusion

- Adapt the pipeline per use-case
- A fully self-contained core is usually not optimal
- Most packages in the Maven ecosystem are unused, adapt the usage threshold filter accordingly

Related literature

[1] Tempero, E., Anslow, C., Dietrich, J., Han, T., Li, J., Lumpe, M., Melton, H., & Noble, J. (2010). **The Qualitas Corpus: A Curated Collection of Java Code for Empirical Studies**. APSEC, 336–345

[2] Dietrich, J., Schole, H., Sui, L., & Tempero, E. (2017). **XCorpus - An executable Corpus of Java Programs**. JOURNAL OF OBJECT TECHNOLOGY, 16(4).

[3] Benelallam, A., Harrand, N., Soto-Valero, C., Baudry, B., & Barais, O. (2019). **The Maven Dependency Graph: A Temporal Graph-Based Representation of Maven Central**. 2019 IEEE/ACM 16th International Conference on Mining Software Repositories (MSR), 344–348.