




Dependency Update Latency in the Maven Software Ecosystem

Authors: Baltasar Berretta, Gus Thomas, Heather Guarnera

Context

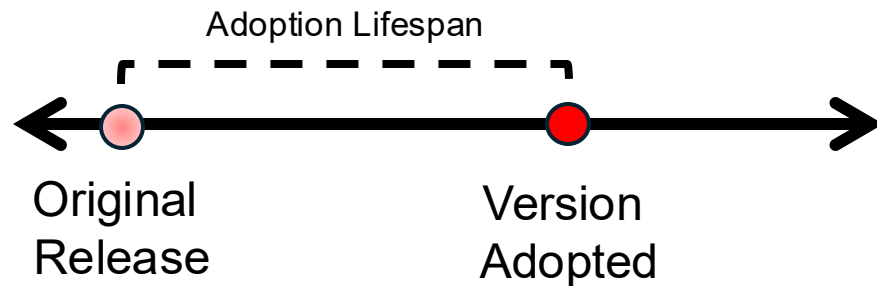
Regular dependency updates protect dependent software components

- Upstream bugs 
- Security vulnerabilities 
- Poor Code Quality 

How do we measure dependency updates?

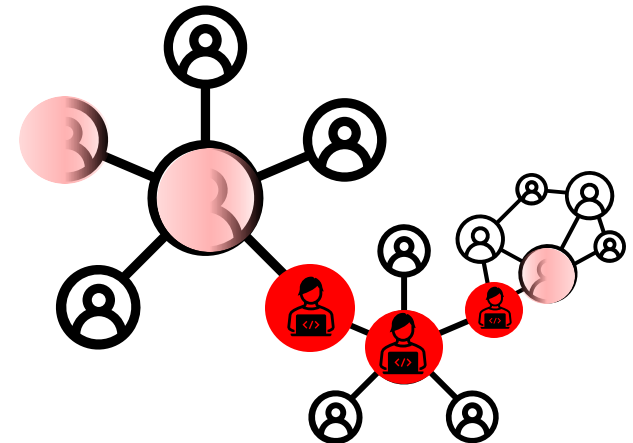
Adoption Lifespan

The time span during which a release is being newly adopted



Adoption Reach

The extent of adoption across the ecosystem



How do we examine correlations between adoption patterns?

Semantic Change

The extent of modifications affecting the meaning or behavior of the code

Version 1.0.1

```
def authenticate(username, password):  
    stored_password = get_user_password_from_db(username)  
    if password == stored_password:  
        return True  
    return False
```

How do we examine correlations between adoption patterns?

Semantic Change

The extent of modifications affecting the meaning or behavior of the code

Version 1.0.**2**

```
def authenticate(username, password):  
    stored_password = get_user_password_from_db(username)  
    if hmac.compare_digest(password, stored_password):  
        return True  
    return False
```

Adoption Patterns

We have fulfilled the criteria to state
research question 1

100%

Completed



Research Question 1

How does **Semantic Change** size correlate with
Adoption Patterns ?

Correlations between

- Magnitude of semantic change size (major, minor, patch), and
- Number of dependents and adoption lifespan

How do we examine correlations between adoption patterns?

Semantic Change

The extent of modifications affecting the meaning or behavior of the code

Version 1.0.**2**

```
def authenticate(username, password):  
    stored_password = get_user_password_from_db(username)  
    if hmac.compare_digest(password, stored_password):  
        return True  
    return False
```

Maintenance Rate

The relative *maintenance rate* of upstream packages

How do we examine correlations between adoption patterns?

Maintenance Rate

Custom metric that quantifies the number of updates relative to a release's adoption lifespan

$$\text{maintenance rate} = \frac{\# \text{ of releases}}{\text{adoption lifespan (years)}}$$

Research Question 2

How does **Maintenance Activity** correlate with
Adoption Patterns ?

Correlations between

- Frequency of maintenance activity (high, medium, low) and
- Number of dependents and adoption lifespan

Insights

	Semantic Change Size			Maintenance Rate		
	Major C.X.X	Minor X.C.X	Patch X.X.C	High $x > 0.1$	Medium $0.01 \leq x \leq 0.1$	Low $x < 0.01$
Number of packages	3,526,059 (84%)	587,676 (14%)	83,954 (2%)	1,830,042 (44%)	181,600 (4%)	2,186,047 (52%)
Average dependents	17.65	8.09	4.21	8.58	19.73	22.05
Average adoption lifespan (in days)	34.52	25.78	19.73	0.36	39.88	59.84

Insights

	Semantic Change Size			
	Major C.X.X	Minor X.C.X	Patch X.X.C	
Number of packages	3,526,059 (84%)	587,676 (14%)	83,954 (2%)	1,8
Average dependents	17.65	8.09	4.21	
Average adoption lifespan (in days)	34.52	25.78	19.73	

- Major changes are most common (**84%**)
- Have highest # of average dependents (17)
- And longest average adoption lifespan (34 days)

Insights

	Semantic Change Size			Main	
	Major C.X.X	Minor X.C.X	Patch X.X.C	High $x > 0.1$	0.
packages	3,526,059 (84%)	587,676 (14%)	83,954 (2%)	1,830,042 (44%)	1
dependents	17.65	8.09	4.21	8.58	
lifespan (in days)	34.52	25.78	19.73	0.36	

- Minor Changes are less common (**14%**)
- Have fewer average dependents (8)
- With a shorter average adoption lifespan (25 days)

Insights

	Semantic Change Size			Maintenance Rate	
	Major C.X.X	Minor X.C.X	Patch X.X.C	High $x > 0.1$	Medium $0.01 \leq x \leq 0.1$
	3,526,059 (84%)	587,676 (14%)	83,954 (2%)	1,830,042 (44%)	181,600 (4%)
	17.65	8.09	4.21	8.58	19.73
ays)	34.52	25.78	19.73	0.36	39.88

- Patch Changes are rarest (**2%**)
- Have the least amount of dependents (**4%**)
- And the shortest adoption lifespan (**19 days**)

Insights

Semantic Change Size			Maintenance Rate		
Minor X.C.X	Patch X.X.C		High $x > 0.1$	Medium $0.01 \leq x \leq 0.1$	Low $x < 0.01$
44%) 587,676 (14%)	83,954 (2%)		1,830,042 (44%)	181,600 (4%)	2,186,047 (52%)
8.09	4.21		8.58	19.73	22.05
25.78	19.73		0.36	39.88	59.84

- More than one update every 10 days
- 44% of packages are high maintenance
- Average 8.58 dependents and 0.36-day adoption lifespan.

Insights

Package Size	Maintenance Rate		
	High $x > 0.1$	Medium $0.01 \leq x \leq 0.1$	Low $x < 0.01$
4%) 83,954 (2%)	1,830,042 (44%)	181,600 (4%)	2,186,047 (52%)
4.21	8.58	19.73	22.05
19.73	0.36	39.88	59.84

- One update every 10 to 100 days.
- 4% of packages are medium maintenance.
- Average 19 dependents and 40-day adoption lifespan.

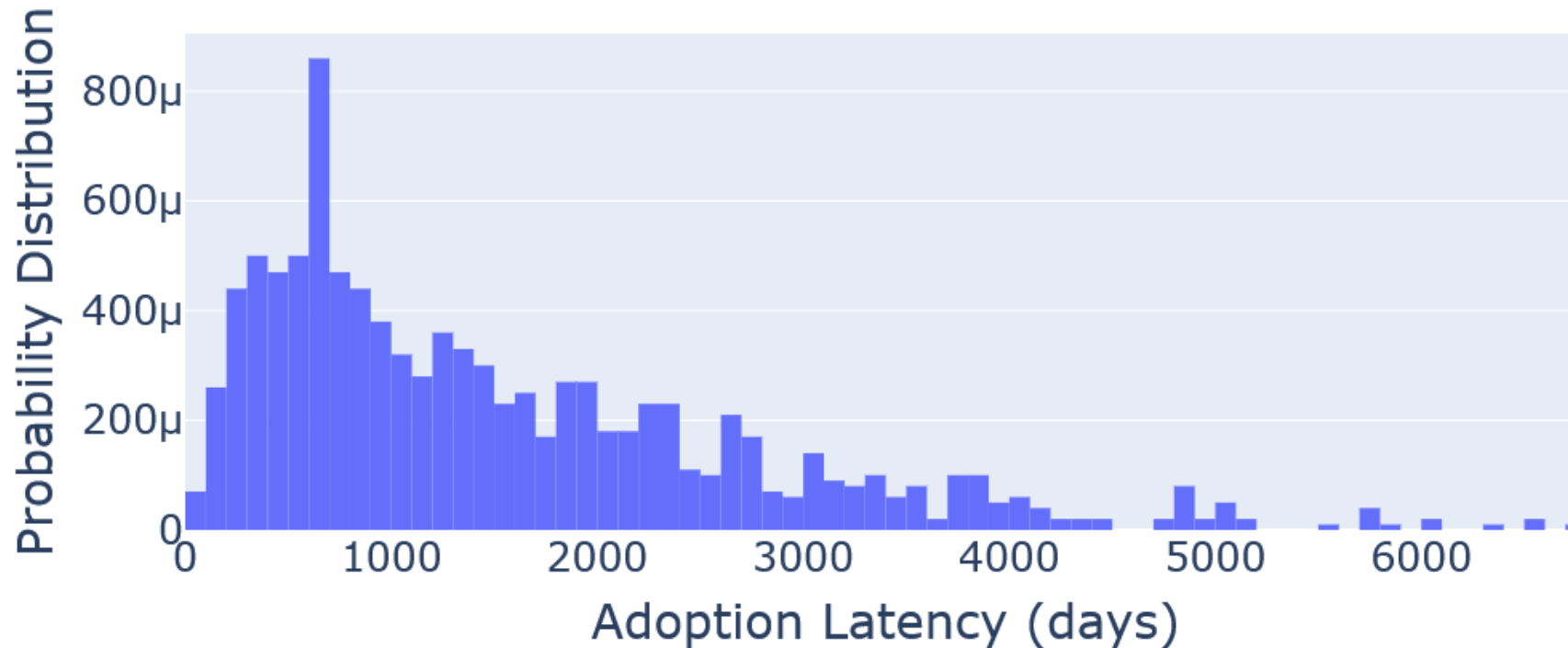
Insights

Maintenance Rate		
High $x > 0.1$	Medium $0.01 \leq x \leq 0.1$	Low $x < 0.01$
1,830,042 (44%)	181,600 (4%)	2,186,047 (52%)
8.58	19.73	22.05
0.36	39.88	59.84

- Fewer than one update every 100 days
- 52% of packages are low maintenance
- Average 22 dependents and 60-day adoption lifespan

Maven Trends

Figure 1: Probability distribution of adoption lifespan across a sample of Maven packages (N=1,000)

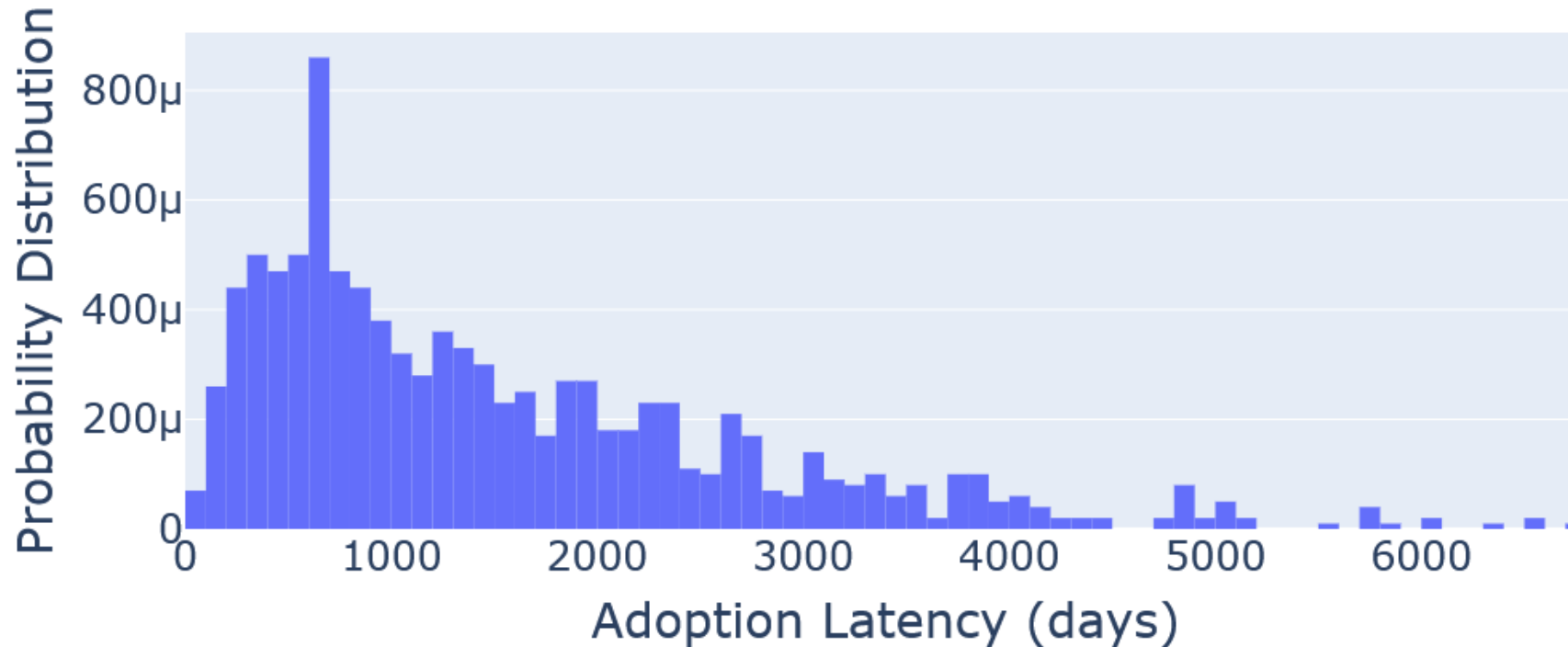


Adoption Latencies

Time between first and last adoption of a package

Maven Trends

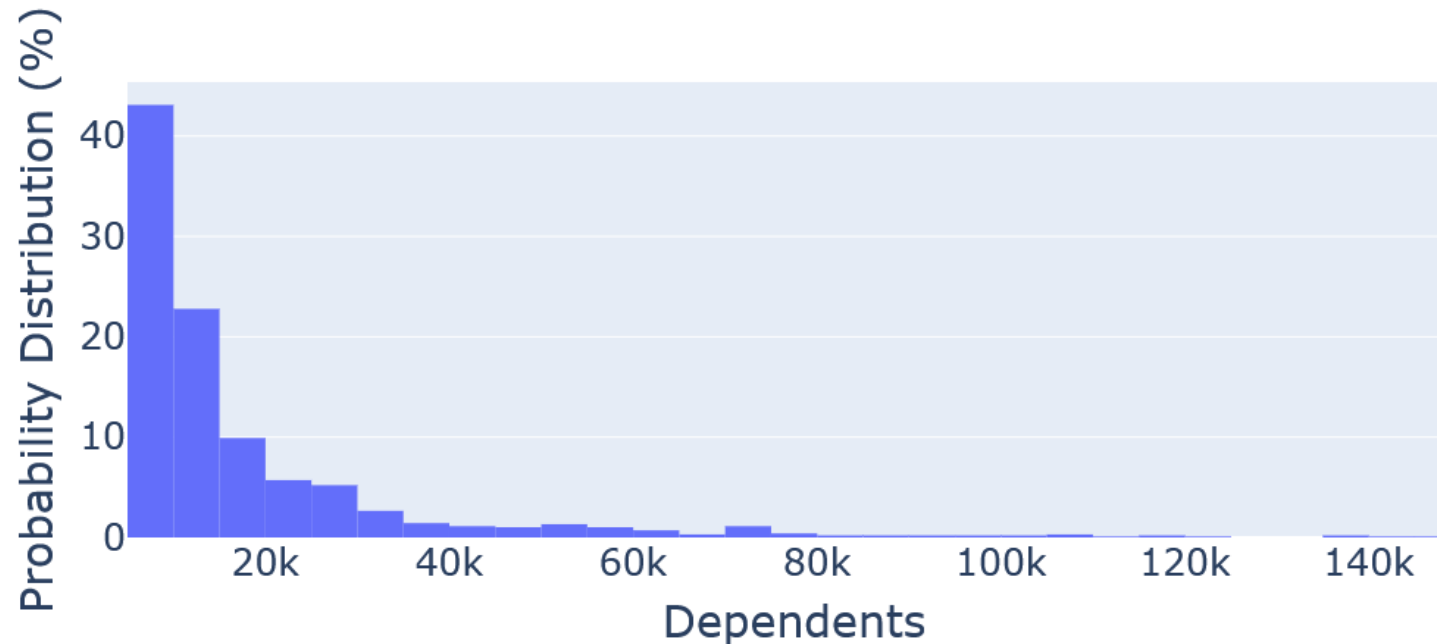
Figure 1: Probability distribution of adoption lifespan across a sample of Maven packages (N=1,000)



Modal adoption latency ~ 2 years

Maven Trends

Figure 2: Probability distribution of number of dependents across a sample of Maven packages (N=1,000)



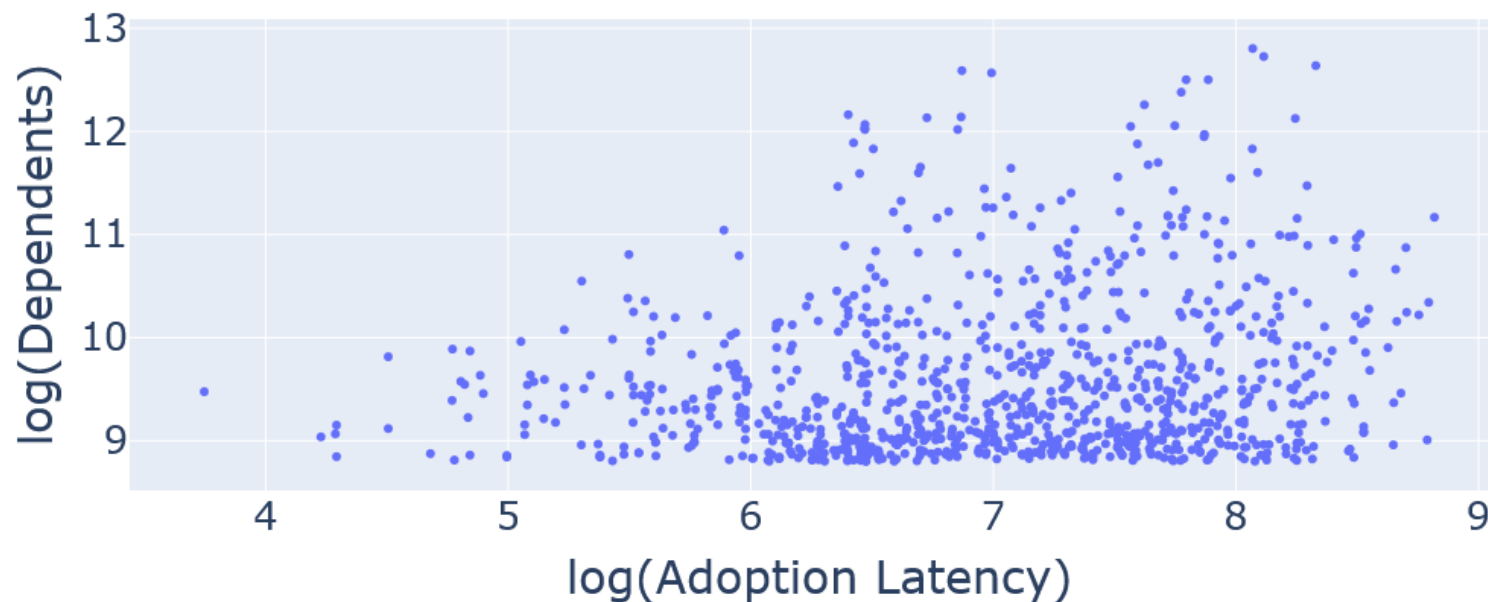
Number of dependents is exponentially distributed

Number of Dependents

Total number of releases that are dependent on a project

Maven Trends

Figure 4: Logarithm of dependent number as a function of the logarithm of adoption lifespan



Adoption latency is loosely correlated with
number of dependents

Dependent-Latency Correlation

Number of dependents as a function of adoption latency

Threats to Validity

Dataset sampling: due to infrastructure & query-time constraints for “Maven trends” we only analyzed the 1,000 packages with the highest number of dependents

Semantic versioning compliance the change caused by a release may not be represented according to the versioning convention (i.e. some “patches” may change APIs causing longer adoption times)

Conclusion & Future Work

- Maintainers of highly-depended packages should expect longer adoption periods for their major releases
- Consumers of highly maintained components should integrate automated dependency update tools
- Minor and patches are adopted at the same rate
 - Tendencies to adopt minor changes?
- Larger semantic version changes correlate to longer adoption lifespans
 - Measuring semantic change size by adoption latency

References

- D. Jaime, J. E. Haddad, and P. Poizat, “Goblin: A framework for enriching and querying the Maven Central Dependency Graph,” in Proceedings of the 21st International Conference on Mining Software Repositories, ser. MSR ’24. Association for Computing Machinery, 2024, pp. 37–41. [Online]. Available: <https://dl.acm.org/doi/10.1145/3643991.3644879>
- A. Tsakpinis, “Analyzing maintenance activities of software libraries,” in Proceedings of the 27th International Conference on Evaluation and Assessment in Software Engineering, ser. EASE ’23. New York, NY, USA: Association for Computing Machinery, 2023, p. 313–318. [Online]. Available: <https://doi.org/10.1145/3593434.3593474>
- H. He, R. He, H. Gu, and M. Zhou, “A large-scale empirical study on Java library migrations: prevalence, trends, and rationales,” in Proceedings of the 29th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering, ser. ESEC/FSE 2021. New York, NY, USA: Association for Computing Machinery, 2021, pp. 478–490. : <https://doi.org/10.1145/3468264.3468571>