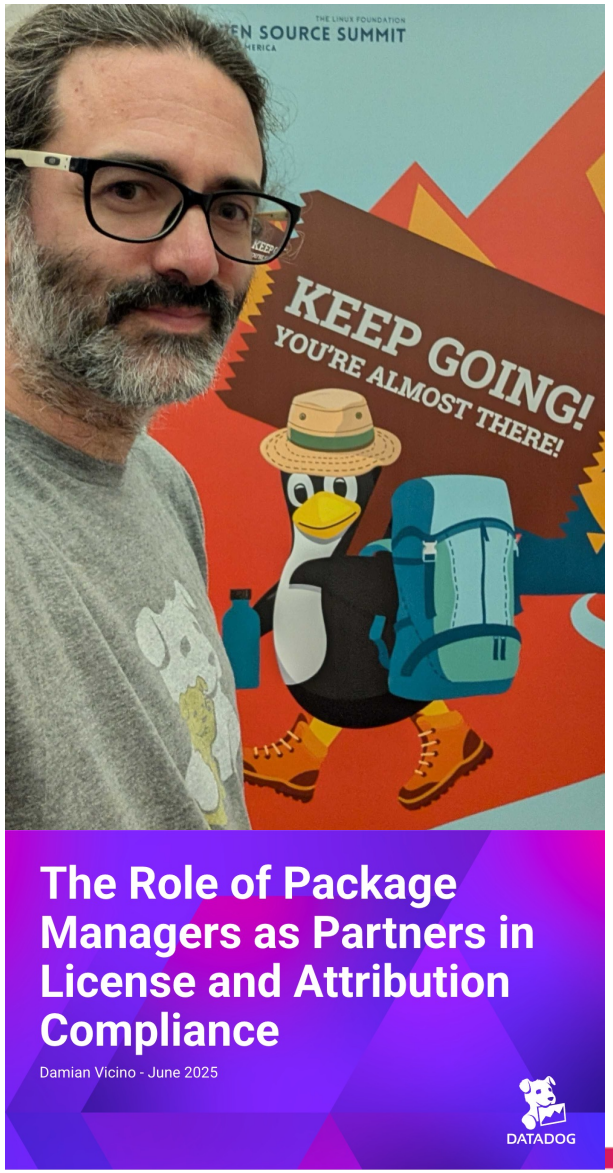
THE LINUX FOUNDATION
OPEN SOURCE SUMMIT
NORTH AMERICA

Embedded Linux Conference

PACKAGE MANAGERS METADATA AND CROSS-ECOSYSTEM PROJECTS IN THE ERA OF SBOMS

Damián Vicino
CHAOSS Package Metadata WG
Datadog OSPO
May 2026



Package Managers Metadata and Cross-Ecosystem Projects in the Era of SBOMs

Damián Vicino

CHAOSS Package Metadata WG chair

Senior OSS Program Developer, Datadog OSPO

CHAOSS

What package managers metadata?

pyproject.toml

```
[project]
name = "dd-license-attribution"
version = "0.5.0"
readme = "README.md"
requires-python = ">=3.11"
license = "Apache-2.0"
...
dependencies = [
    "agithub",
    "scancode-toolkit",
    "typer",
    "requests",
    "license-expression",

...
```

package.json

```
{
  "name": "lodash",
  "version": "4.18.1",
  "license" = ”MIT”,
  "private": true,
  "main": "lodash.js",
  "engines": {
    "node": ">=4.0.0"
  },
  "scripts": {
    "build": "npm run
build:main && npm run
build:fp",

...
```

FreeBSD ports

```
PORTNAME=    ohmyzsh
PORTVERSION=
             20241128
CATEGORIES= shells

MAINTAINER= skozlov@FreeBSD.org
COMMENT=    Community-driven...
WWW=        https://ohmyz.sh/

LICENSE=    MIT
LICENSE_FILE= ${WRKSRC}/LICE...

RUN_DEPENDS= zsh:shells/zsh

...
```

CHAOSS

Everyone is solving the same problem

- Software Composition Analysis (SCA)
- License compliance
- SBOM generation
- Vulnerability management
- Dependency update automation
- Supply chain security & provenance
- Ecosystem research
- Ecosystem funding & sustainability
- AI-assisted development tooling
- Others

CHAOSS

Every tool has its own mapping layer for reading package manager metadata.

No shared reference framework.

CHAOSS

Who absorbs the cost?

Package Consumer

Package Maintainer

Package Manager Developer

These are abstract roles. Anyone can wear one to three of these hats.

CHAOSS

Now add scale

- Millions of developers
- Millions of projects
- Mixed ecosystems

CHAOSS

The problem doesn’t stay where it starts

Metadata quality is everyone’s problem, because metadata composes.

CHAOSS

Package managers didn’t evolve together

They weren’t supposed to

Scope	Language-specific vs. OS-specific vs. application-specific
Dependency resolution	Link-time, build-time, deploy-time
Composition model	Source-based vs. binary vs. hybrid
Specification process	Formal specs and RFCs vs. community-driven and evolving
Registry model	Centralized vs. federated vs. distributed
Governance	Foundation-backed vs. volunteer-run vs. vendor-led

“Package managers have existed since 1992. Why would this problem get solved now?”

CHAOSS

External pressure is building again

- SBOMs are becoming legal requirement in multiple jurisdictions.
- SBOMs accuracy is as good as the metadata it’s built from.

EU Cyber Resilience Act · US federal SBOM requirements · and others.

CHAOSS

The package metadata working group

- CHAOSS – a Linux Foundation project
- Neutral space – Not a standards body. Not a vendor.
- Initial meeting 2025-09-03

CHAOSS

So, we formed a working group

CHAOSS Package Metadata Working Group

- Document package managers practices
- Review key metadata concepts across ecosystems
- Bridge knowledge between package managers
- We are about 1 year on it.

23

package managers analyzed

40+

registries observed

CHAOSS

We are building a reference map, not a new standard

- **Document**: how an attribute is implemented in each ecosystem
- **Measure**: ecosystem metrics in the wild
- **Review**: academic research, existing standards, prior art
- **Synthesize**: crosspollinate ecosystem learnings

CHAOSS

First attribute: LICENSE

Queue Criteria: important enough to matter, understood enough to start.

Why License first?

- First thing to check when adopting a dependency
- Central to Open Source distribution since beginning
- Every package manager implemented it, right?

CHAOSS

License Initial Research

23 ecosystems surveyed across package managers and OS distributions.

Field name	license, licenses, licenseInfo, and others...
Validation	Strict enforcement, optional, or none
Structure	SPDX expression, typed expression, free-string, or none
Attribute count	Unique attribute, multiple attributes, overlapping concern attributes

CHAOSS

Analysis Draft: making sense of what we found

Unambiguously specified (8)

Cargo · npm · Composer · NuGet · Homebrew · rpm (Fedora) · Vcpkg · Cabal/Hackage

Ambiguously specified (12)

Conan · Clojars · CPAN · RubyGems · Maven · PyPI · Conda · dpkg · apk · FreeBSD Ports · CocoaPods · Go Modules

Unspecified (3)

Docker · SwiftPM · Carthage

Unambiguously Specified (8)

What is the bar?

A package maintainer can declare:

- License with unique name/id
- License expressions (AND, OR, WITH)
- Custom License support

Examples

Homebrew

- License expressed with ruby symbols
  :MIT
- License expressions supported with structured hash
  { any_of: [:MIT, :Apache_2_0] }
- Custom License support value :cannot_represent value

Cargo (rust)

- SPDX expression support with validation
- Escape hatch to reference a License file in the package.

Ambiguously Specified (12)

What is the bar?

- License metadata related attributes exist
- Lower expression power than SPDX

Examples

CPAN (Perl)

- License field available and mandatory
- CPAN::Meta::Spec formal specification
- Ambiguous identifiers:
  open_source, restricted, unrestricted, unknown
- License field is array of identifiers (OR)
- No AND or WITH available

Unspecified (3)

What is the bar?

No place in metadata for license specification

Examples

Docker

- No license field in image manifests
- No license field in Dockerfiles

Carthage (Swift)

- No package manifest with structured metadata.
- License information is entirely file-inferred.

CHAOSS

Supporting Data collection

How does “recommended but not enforced” actually work?

Sample	Packages with license field	Valid SPDX
Top 1% (most popular)	36,224	**60%**
Top 10%	2,338,707	**47%**

“The long tail is where “recommended” goes to die.”

~6% of critical packages across all ecosystems have no parseable license.

Maven Central, all the same license, none agreeing:

- The Apache Software License, Version 2.0
- APACHE-2
- Apache License, Version 2.0
- Apache-2.0

Data provided by ecosyste.ms

CHAOSS

What good looks like

Mandatory SPDX expression with optional escape hatch

```
MIT
Apache-2.0 OR MIT
GPL-2.0-only AND MIT
LicenseRef-licenses/custom.txt
LicenseRef-licenses/custom.txt OR GPL-3.0-only
```

Partial adoption has its own problems.

The escape hatch is not a workaround. It is part of the SPDX standard.

CHAOSS

Validation

Python	`license-expression`
JavaScript	`spdx-expression-parse`
PHP	`composer/spdx-licenses`
Rust	`spdx`

Where validation happens depends on your ecosystem:

Centralized registry	Validate at submission
Distributed ecosystem	Validate in publisher tooling. Check at consumer.
Universal rule	Never break consumer. Raise quality over time.

CHAOSS

Getting there

› 30 years of packages cannot re-declare their license overnight.
› Five stages designed to never break existing consumers.

1. Parallel support
2. New packages require SPDX
3. Update to packages require SPDX
4. Dormant packages clean up
5. Phase off old format

› Some ecosystems already walked this path.

CHAOSS

Publishing and implementing License recommendations

› Expand feedback collection
› Engage with package managers and registries
› Support communities working toward the SPDX expression requirement

CHAOSS

License is one attribute. The work continues

Currently in progress:

› Source Code URL
› Lock files
› Binary dependencies
› Binary artifact distributions
› Attestations and trusted publishing
› Package Status Metadata
› Funding Links

Already queued:

› SBOM reference name
› Dependency consumption policies
› Contact information

One attribute at a time. Each one the same rigorous process.

CHAOSS

Every tool that depends on package metadata benefits from this work

- Software Composition Analysis (SCA)
- License compliance
- SBOM generation
- Vulnerability management
- Dependency update automation
- Supply chain security & provenance
- Ecosystem research
- Ecosystem funding & sustainability
- AI-assisted development tooling
- Others

A shared reference means less duplicated work. Better tools. More reliable results.

Get involved

CHAOSS Project

chaoss.community

Package Metadata Working Group

chaoss.community/groups/package-metadata

GitHub Repository

github.com/chaoss/wg-package-metadata

Slack

#wg-package-metadata on CHAOSS Slack · chaoss.community/participate

We meet virtually every two weeks. Wednesdays at 16:00 UTC. Everyone is welcome.