Comment on Software Bill of Materials Elements and Considerations

NTIA-2021-0001

National Telecommunications and Information Administration

[Docket No. 210527–0117]

I. Introduction: who we are (PSF Mission & Organizational History)
II. Background context about the Python packaging toolchain and ecosystem
III. Additional use cases and flexibility
IV. Infrastructure funding
V. Importance of using open standards
VI. Conclusion (please contact us for further discussion)

I. PSF Mission & Organizational History

This response to the NTIA request for comment is from the Packaging Working Group of the Python Software Foundation, a registered 501(c)3 nonprofit in the United States.

The Python Software Foundation's (PSF) mission is to promote, protect, and advance the Python programming language, and to support and facilitate the growth of a diverse and international community of Python programmers.

Since its establishment in 2001, the PSF has worked to ensure the long-term sustainability of the open-source Python language and expanded the Python community by providing support for thousands of Python projects and programmers worldwide.

Highlights of the PSF's work over the past two decades includes:

- Produced **18** PyCon US Conferences
- Funded more than **39** sprints worldwide
- Provided more than **651** PyCon scholarships and travel grants for individuals from under-represented and under-resourced backgrounds
- Awarded more than **$2,804,281** in grants to Python projects in **91** countries

The PSF also hosts and improves the core infrastructure for the Python packaging ecosystem (see below).

Packaging Working Group

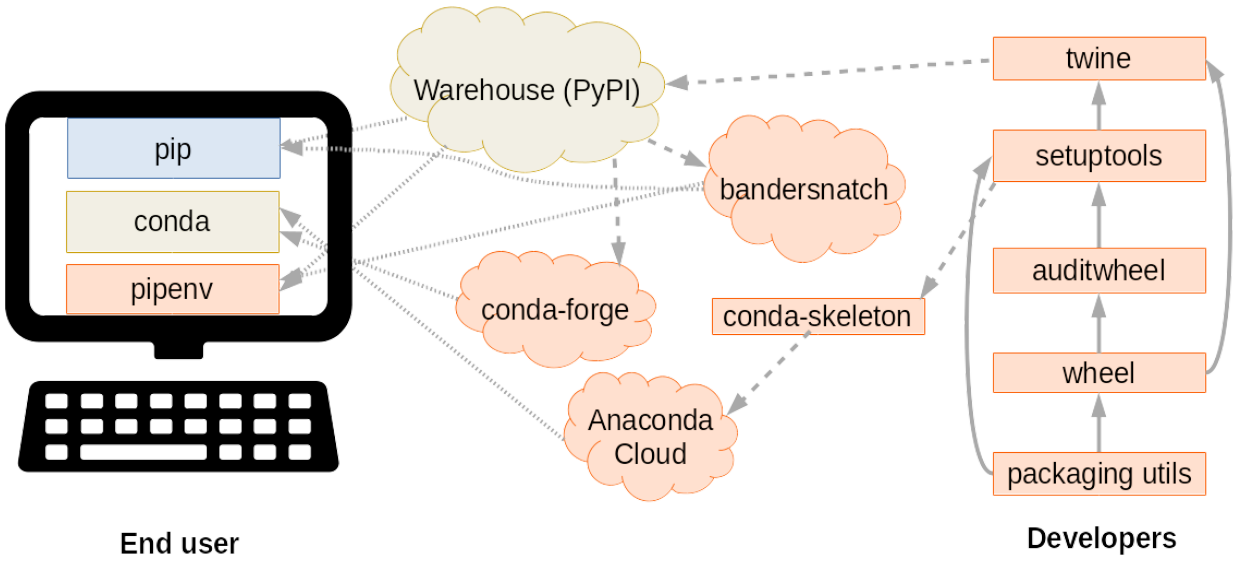
The Packaging Working Group is a volunteer work group of the Python Software Foundation. The purpose of this working group is to support the larger efforts of improving and maintaining the packaging ecosystem in Python through fundraising and disbursement of raised funds. It largely focuses on efforts such as the Python Package Index, pip, packaging.python.org, setuptools, and cross-project efforts.

II. The Python Packaging Ecosystem

The Python packaging ecosystem spans multiple software projects, standards, and needs within the Python community. Major projects include:

- **The Python Package Index**: an online repository of software for the Python programming language. PyPI helps users find and install software developed and shared by the Python community. You can visit it on the web at www.pypi.org .
- **pip**: a command-line tool for resolving Python dependencies and installing them into development, testing, and production environments.
- **virtualenv**: a tool which uses the command-line path environment variable to create isolated Python virtual environments.
- **setuptools**: a tool to easily build and distribute Python distributions, especially ones that have dependencies on other packages.
- **twine**: the primary tool developers use to upload Python packages to the official Python Package Index or other Python package indexes.
- **PyPA/packaging.python.org**: the collection of official projects, specifications, their contributors, and online guides for best practices in the Python packaging ecosystem.

It's particularly worthwhile to note the multiplicity of interoperable tools in the Python packaging toolchain -- mostly volunteer-built and volunteer-maintained. Below is a simplified diagram of how some of them, as well as additional volunteer-maintained or commercially available tools, interact; we include this to illustrate the complexity of adding SBOM metadata production or parsing capabilities to all of them. These tools interoperate based on shared standards that we communally develop using the Python Enhancement Proposal (or PEP) process (https://www.python.org/dev/peps/); developing, ratifying, and implementing a new PEP takes months or years.



It is also worthwhile to note the wide variety of software tools and applications people freely build and share *using* these tools. At this writing, PyPI hosts 310,863 "projects" (different software packages), including 2,669,002 "releases" (files representing different versions of those packages).

These include software created by individuals, companies, governments, charities, universities, and more. Some creators release packages as frequently as multiple times per week, and have resources to add features (such as SBOM metadata) into future releases. However, release frequency varies wildly, and many creators make new releases only every few months. Some creators have not published fresh releases of their packages for years and are no longer available to improve or support them; since users may still depend on those packages, PyPI does not by default remove them, and still makes them available for download and installation.

Organizations inside and outside the US also privately create and distribute Python packages using many of these tools; as long as these organizations follow the shared Python packaging standards (as defined in the PEPs), users can download and install them using the tools in this toolchain.

III. Additional use cases and flexibility

The Request for Comment asked for comments on additional use cases, and on "Flexibility of implementation and potential requirements." In addition to the complexity we discuss in the previous section, we ask you to also take care concerning supplier names in Software Bills of Materials.

At least hundreds of thousands of individuals or groups create and release software written in Python. Many developers of open source software are individuals whose legal identities are not known to their users or to distributors of their packages; they use distinct and consistent pseudonyms instead, and it is important to continue to allow them to do so. We are not alone in saying this. The Digital Identity Attestation working group of the Open Source Security Foundation https://github.com/ossf/wg-digital-identity-attestation (which Google refers to in the "Authentication for Participants in Critical Software" section of their Feb. 3rd, 2021 piece "Know, Prevent, Fix: A framework for shifting the discussion around vulnerabilities in open source" https://security.googleblog.com/2021/02/know-prevent-fix-framework-for-shifting.html) mentions that one necessary use case to support is "Work under a pseudonym to protect oneself, e.g. from discrimination, harassment, or stalking."

IV. Infrastructure funding

The Python Software Foundation is a 501(c)3 nonprofit corporation. Many companies and other organizations, large and small, inside the US and worldwide, depend on the Python packaging infrastructure that the PSF hosts and supports, but do not financially support that infrastructure. Packaging Working Group member Dustin Ingram has recently written about the costs of running the Python Package Index:
https://dustingram.com/articles/2021/04/14/powering-the-python-package-index-in-2021/

This is not just true of Python; many important components of open source infrastructure are run by nonprofits like PSF, or by volunteer groups without any formal organizations or funding. We bring this to your attention as you consider possible legal or regulatory mandates around Software

Bills of Materials. Mandating SBOM compliance, without providing additional funding to help implement changes, may end up causing a burden for nonprofit or informal entities in this space.

V. importance of using open standards

The Request for Comment mentioned SWID, CycloneDX, and SPDX as the standards you are considering. Using open standards is important, and we are glad you are emphasizing standards that are openly defined and whose standards definition processes are open for public comment and input.

As you consider SWID in particular, please bear in mind that it only applies to some kinds of software and would need to be supplemented by other identifier schemes to cover other categories; for example, we believe it would not be viable to assign an SWID for every package on PyPI. Please also look into PURL, which attempts to cover that lacuna: https://github.com/package-url/purl-spec/issues/36 .

VI. Conclusion

Thank you for your request for comment. We are interested in further conversations going into more depth as you develop your minimum elements.

Signed:

Sumana Harihareswara

On behalf of the Packaging Working Group of the Python Software Foundation
https://wiki.python.org/psf/PackagingWG

Contact: email PSF Executive Director Ewa Jodlowska <ewa@python.org> and please cc <packaging-wg@python.org>

Mailing address:
Python Software Foundation
9450 SW Gemini Dr. ECM# 90772
Beaverton, OR 97008
USA