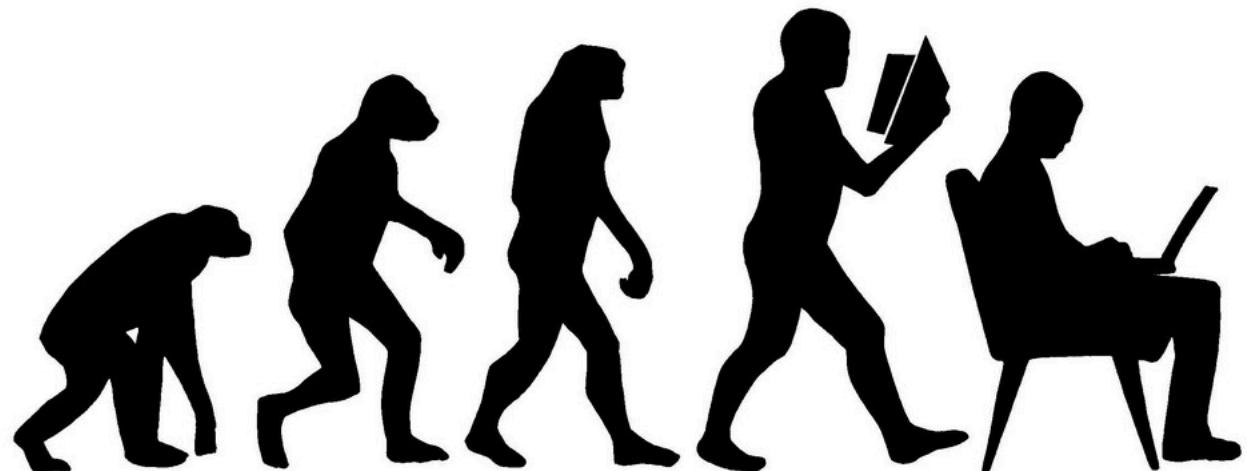


Python パッケージの影響を 歴史から理解してみよう！

Kir Chou @ PyCon JP 2020



自己紹介

- Kir Chou
- PyConの発表経験
 - PyCon JP 2019 発表
 - PyCon TW 2017, 2018, 2019 発表
- その他
 - ソフトウェアエンジニア @ アマゾンジャパン



[@k1rch0u](https://twitter.com/k1rch0u)



[note35](https://github.com/note35)



kir.chou

今日の献立

- 動機
- Python パッケージの歴史
- 歴史から学ぶこと
- コミュニティが Python パッケージを解決する前の挑戦
- コミュニティが Python パッケージを解決した後の挑戦
- まとめ
- Python パッケージの歴史に関する資料

動機



ある日、ビルダーツールチームが発表しました...
これから、オープンソース化されたツールに移行しましょう！



良いじゃないか？でも、なんで？

PEP 566 -- Metadata for Python Software Packages 2.1

PEP:	566
Title:	Metadata for Python Software Packages 2.1
Author:	Dustin Ingram <di at python.org>
BDFL-Delegate:	Daniel Holth
Discussions-To:	distutils-sig <distutils-sig at python.org>
Status:	Final
Type:	Standards Track
Created:	1-Dec-2017



@di_codes

Python パッケージの歴史

パッケージとは software distribution

1

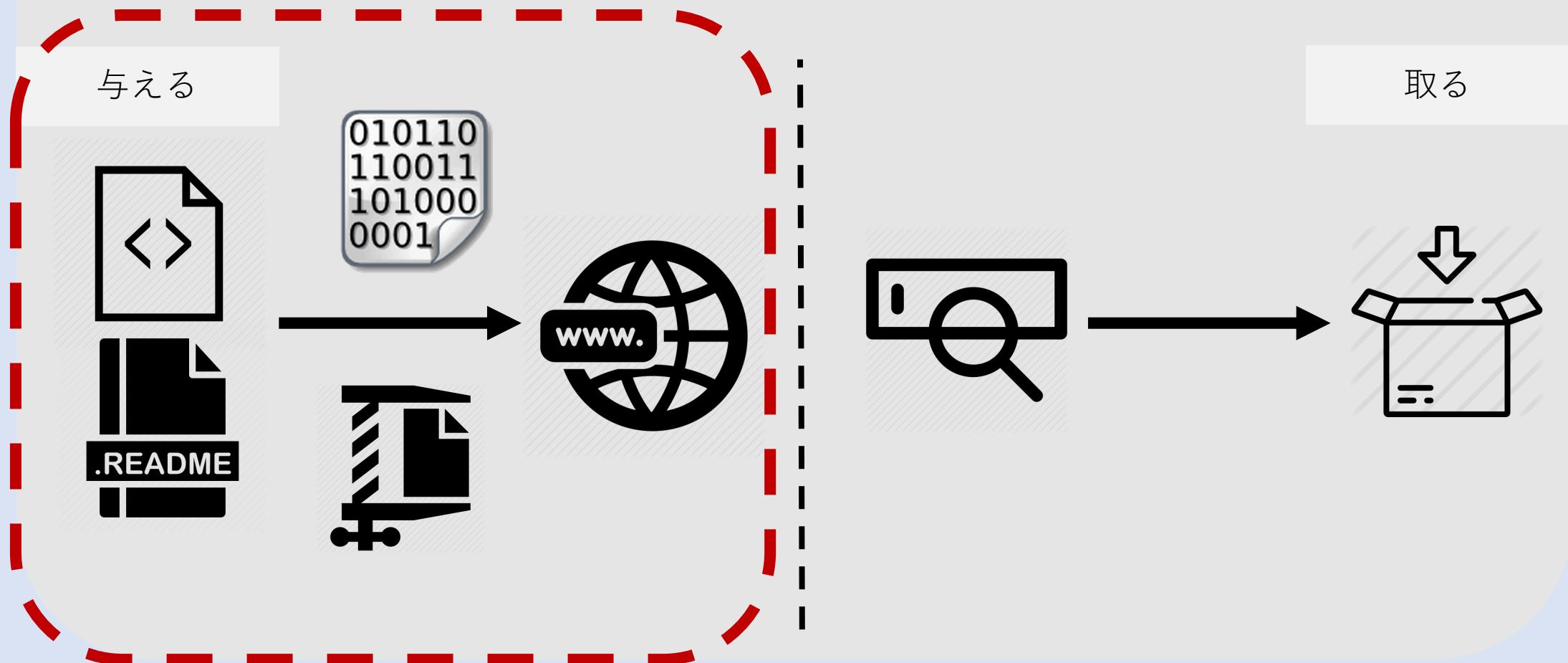
Python 1.0

1991

ソースコードを他の人
に（から）与える（取る）ことは？

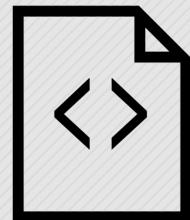


ソースコードを他の人に与えることは？

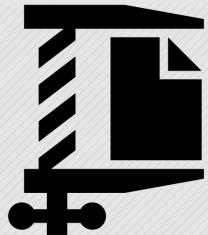


ソースコードを他の人から取ることは？

与える



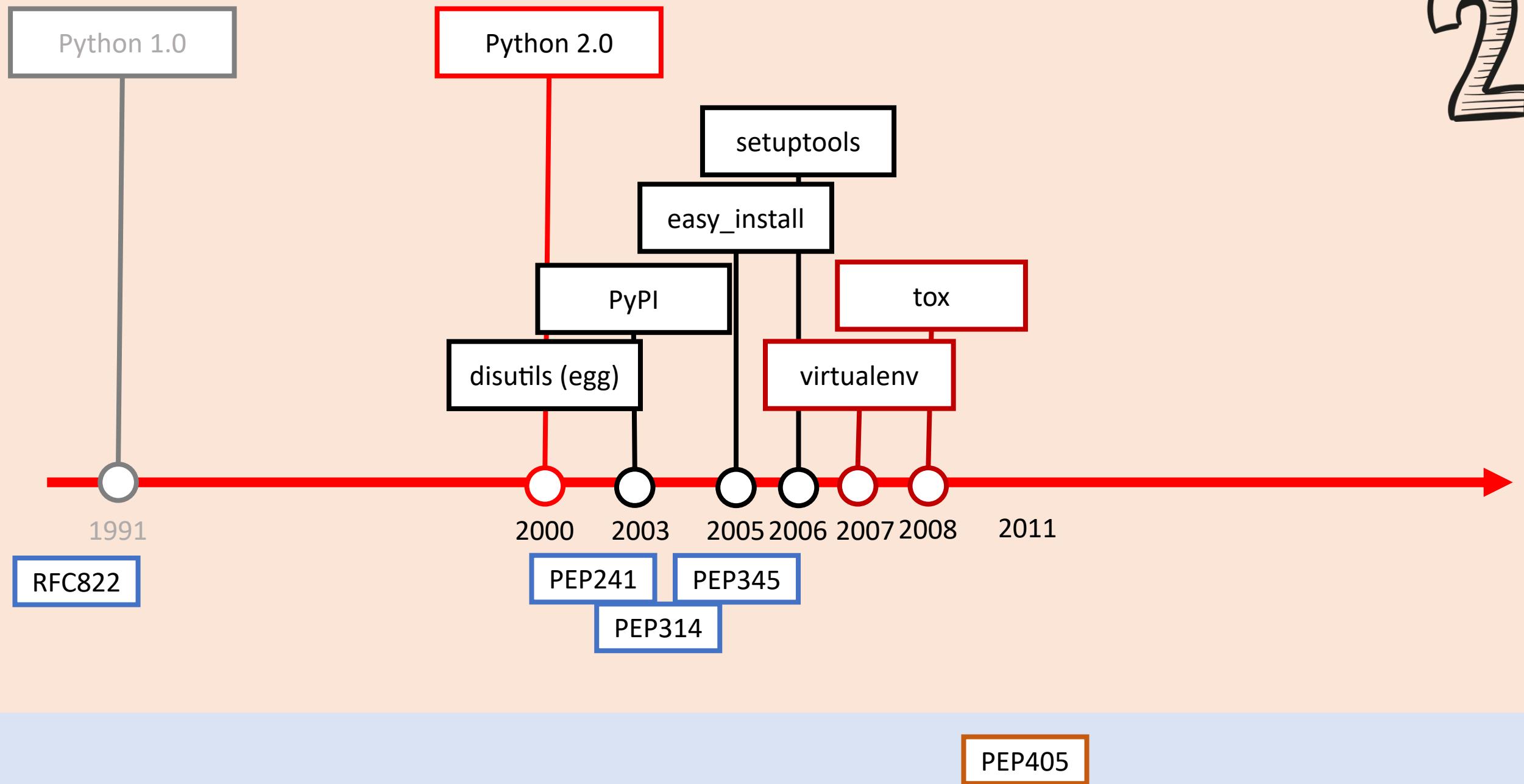
010110
110011
101000
0001



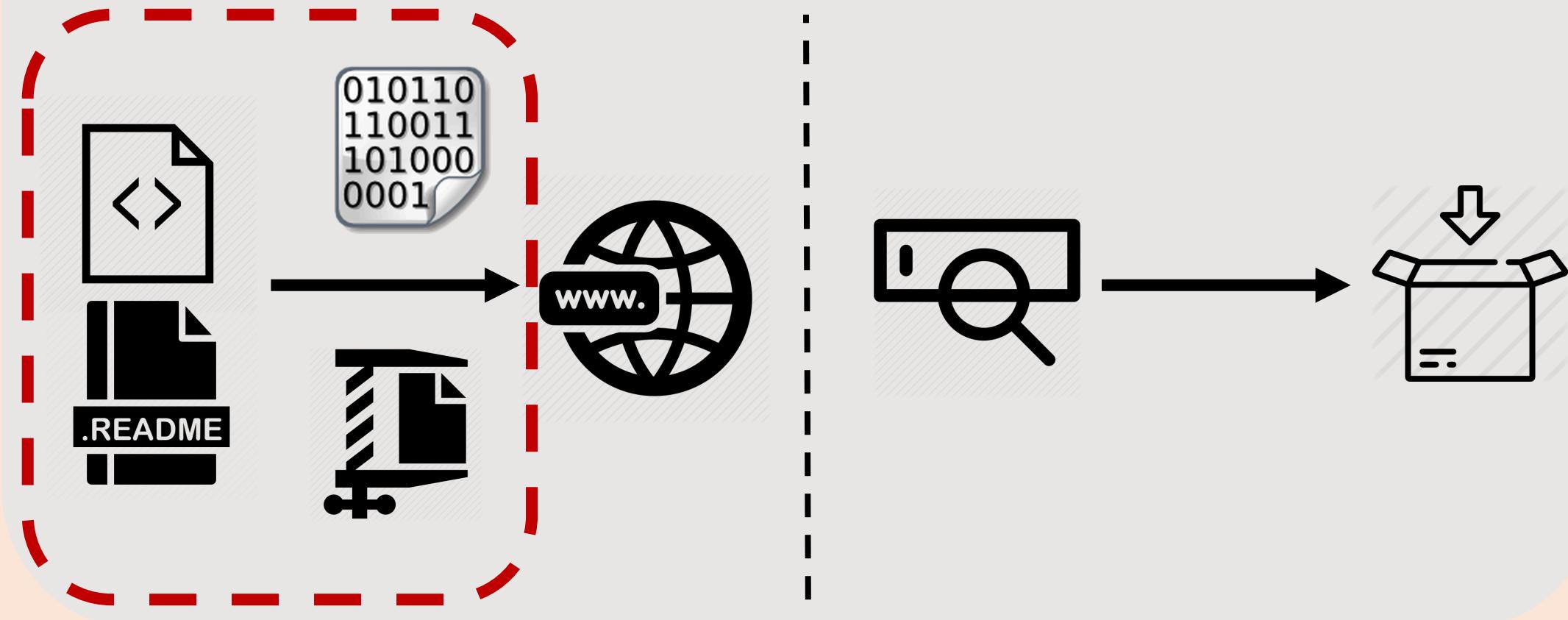
取る







メタデータ: PEP241 (2001), PEP314 (2003), PEP345 (2005)
ビルドツール: distutils, setuptools



```
from setuptools import setup, find_packages

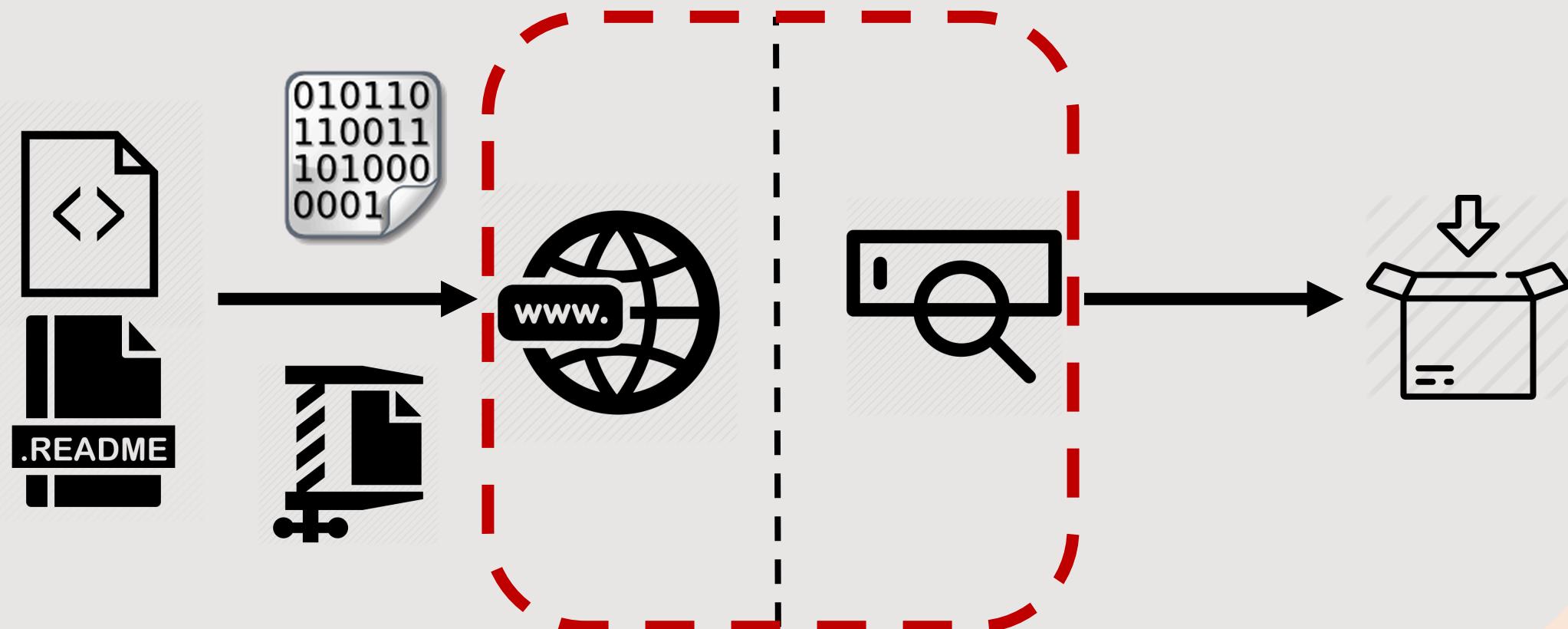
setup(
    name='helloworld',
    version='0.1',
    description="Hello World with setuptools",
    classifiers=[
        'Development Status :: 4 - Beta',
    ],
)
```

setuptools: setup.py

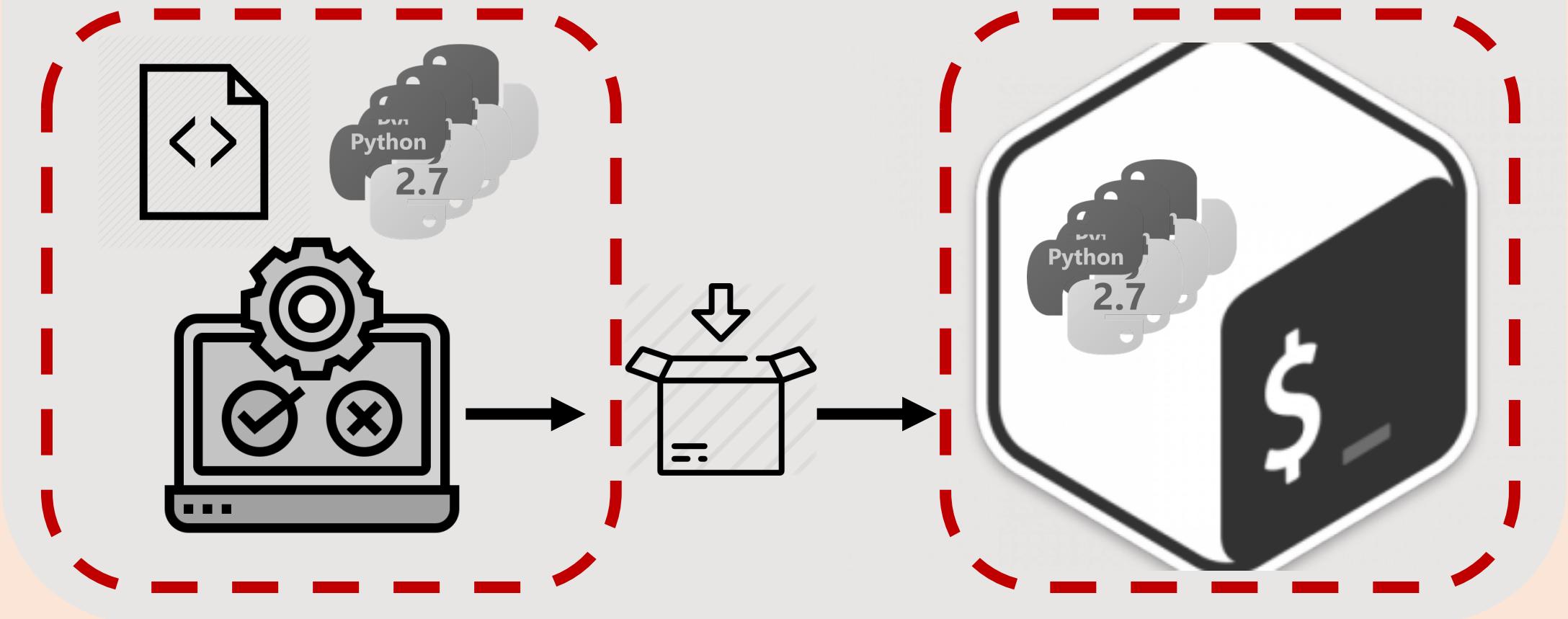
EGGのメタデータ
xxx.egg-info/PKG-INFO

```
Metadata-Version: 1.1
Name: helloworld
Version: 0.1
Summary: Hello World with setuptools
Home-page: UNKNOWN
Author: UNKNOWN
Author-email: UNKNOWN
License: UNKNOWN
Description: UNKNOWN
Platform: UNKNOWN
Classifier: Development Status :: 4 - Beta
```

PyPI (Cheese Shop): PEP301 (2002/Nov) Package インストーラ: easy_install

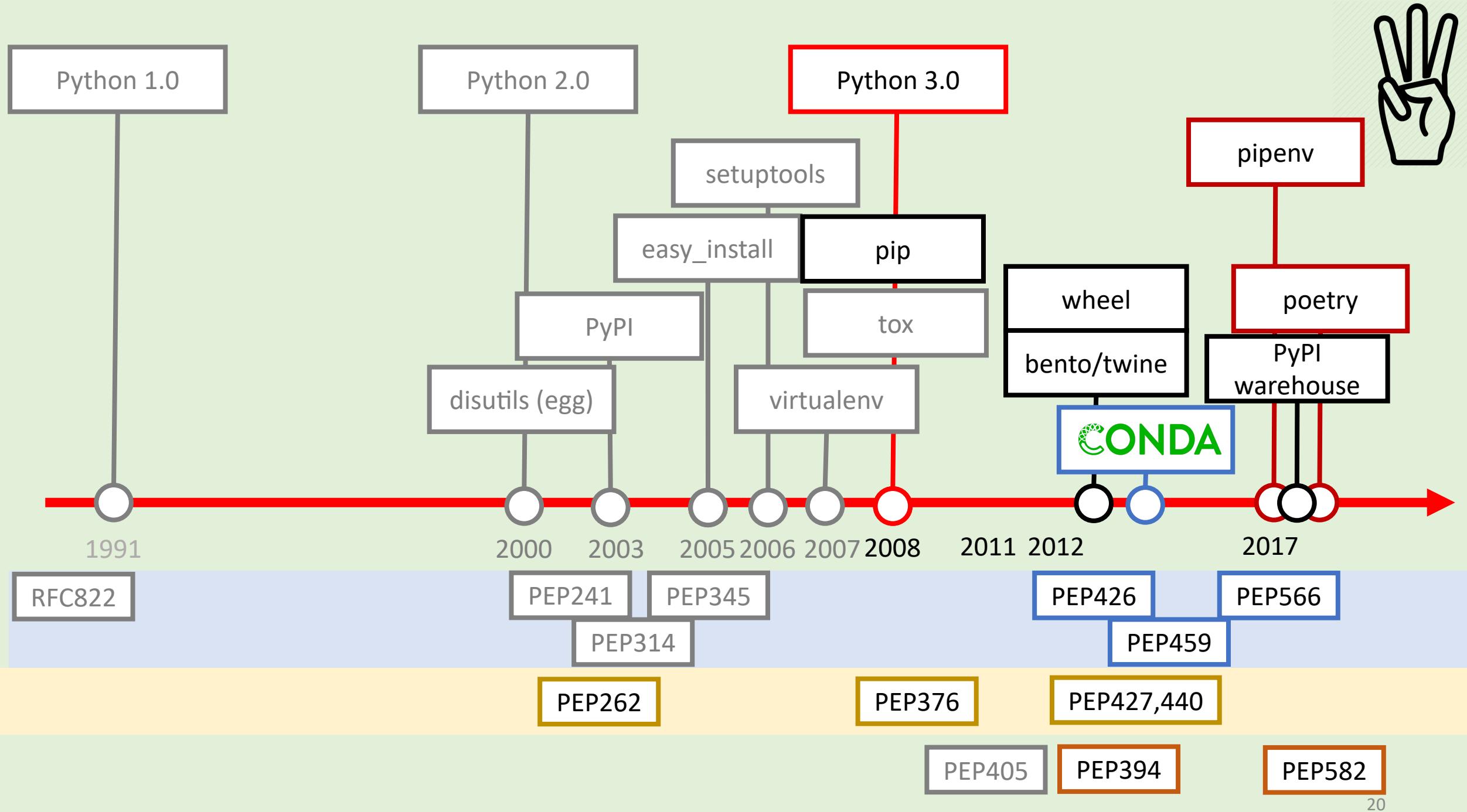


仮想環境（ビルド / テスト / リリース）：
tox, virtualenv (2007), venv@PEP405 (2011)



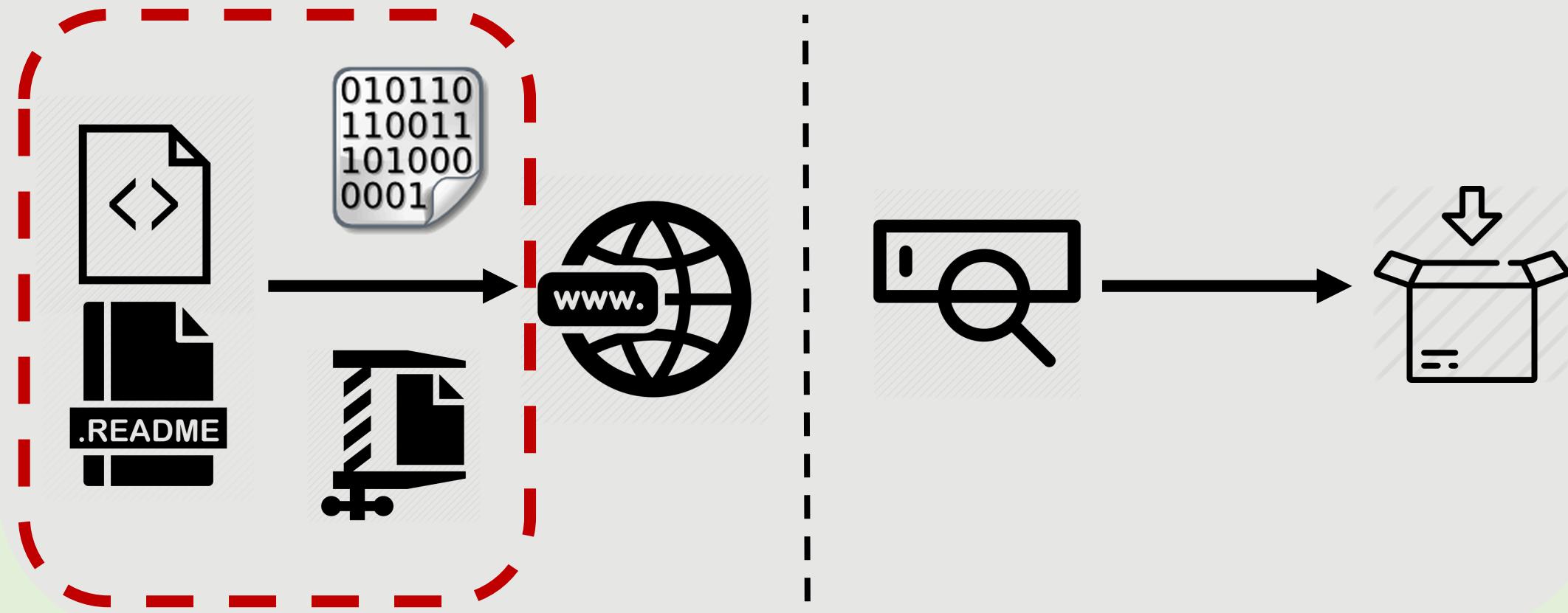


10 YEARS LATER





(延期) メタデータ2.0: PEP426 (2012), PEP459 (2013)
メタデータ 2.1: PEP566 (2017/Dec)





PEP566: pyproject.toml

```
[tool.poetry]
name = "Snake-DeepLearning"
version = "0.1.0"
description = ""
authors = ["Kir Chou"]

[tool.poetry.dependencies]
python = "^3.7"
numpy = "^1.18.3"
keras = "^2.3.1"
tensorflow = "^2.1.0"
tensorflow-estimator = "2.1"
# See README.md, pygame should be installed manually
# pygame = "^1.9.6"

[tool.poetry.dev-dependencies]

[tool.poetry.scripts]
game = "snake_deeplearning.game:main"
generate_data = "snake_deeplearning.training_data_generator:main_generate_data"
generate_model = "snake_deeplearning.training_data_generator:main_generate_model"
test_model = "snake_deeplearning.training_data_generator:main_apply_model"
test_genetic = "snake_deeplearning.training_data_generator:main_apply_genetic_algorithm"

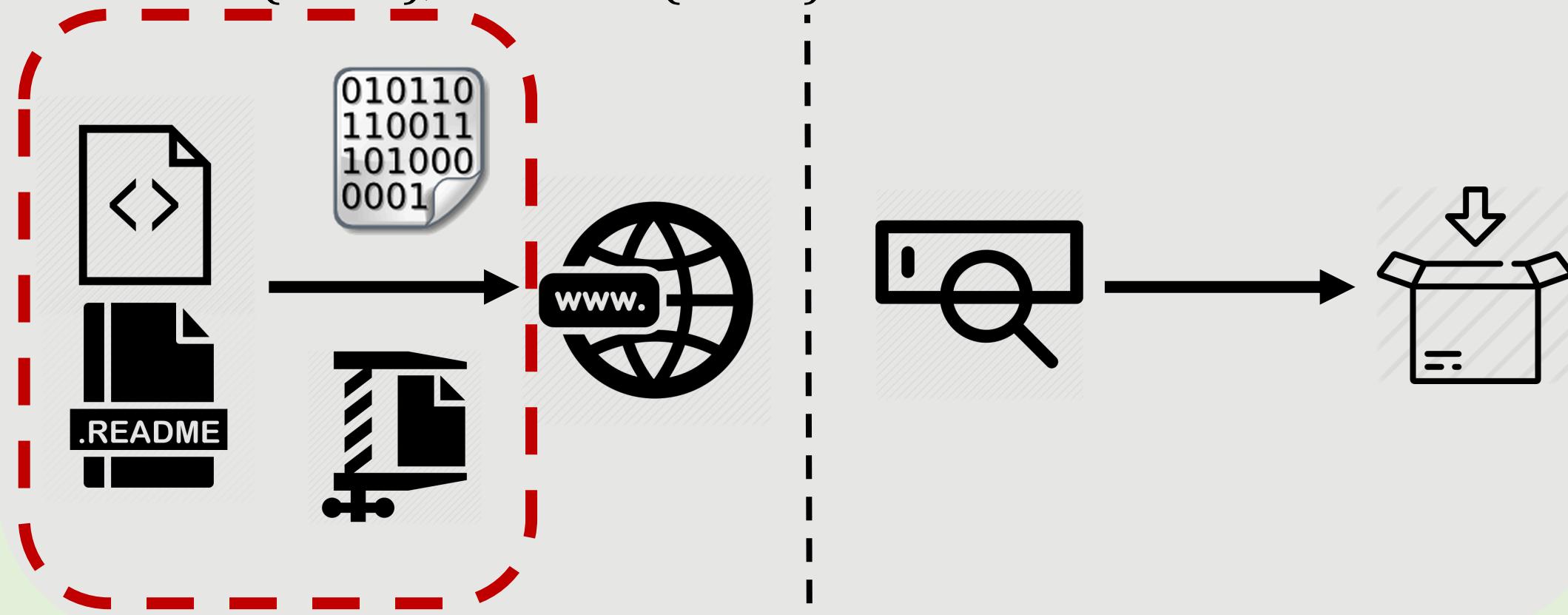
[build-system]
requires = ["poetry>=0.12"]
build-backend = "poetry.masonry.api"
```



バージョン識別と依存関係

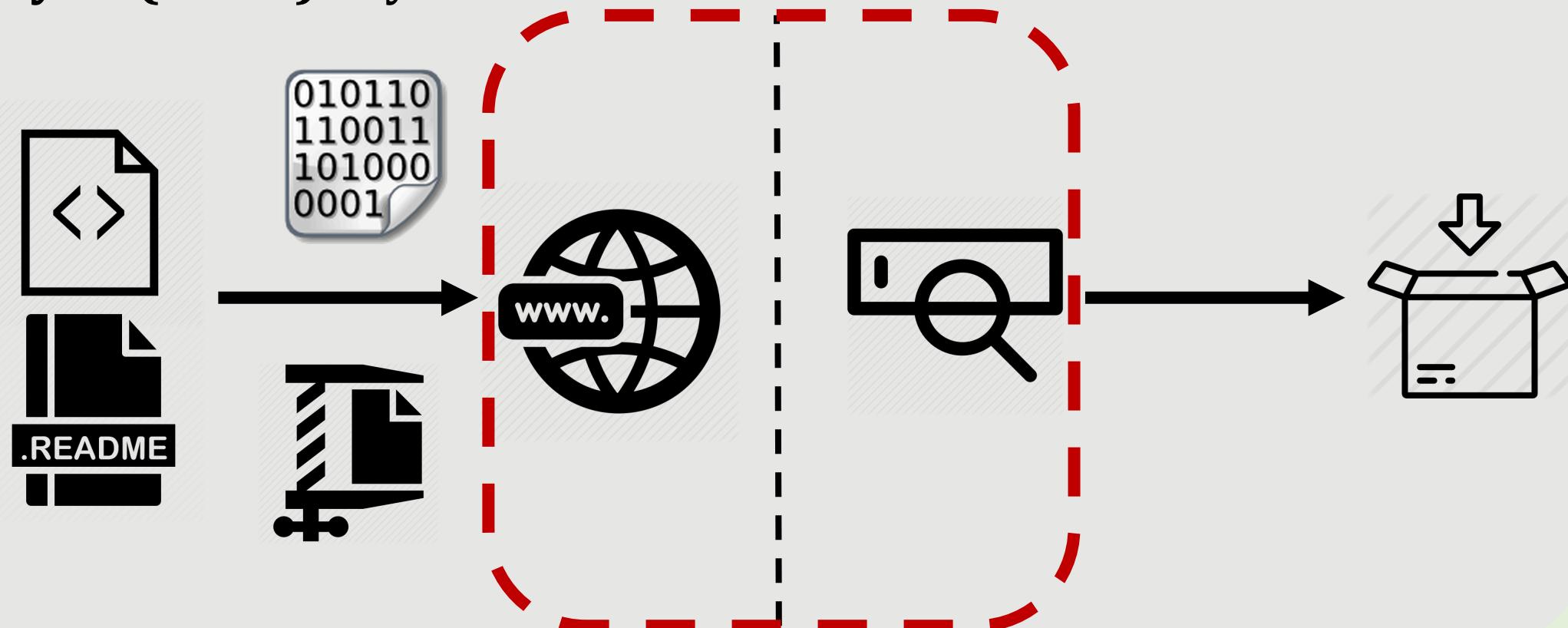
PEP386 (2009/Superseded), pip-tools/pip-sync (2012)

PEP440 (2013), PEP508 (2015)



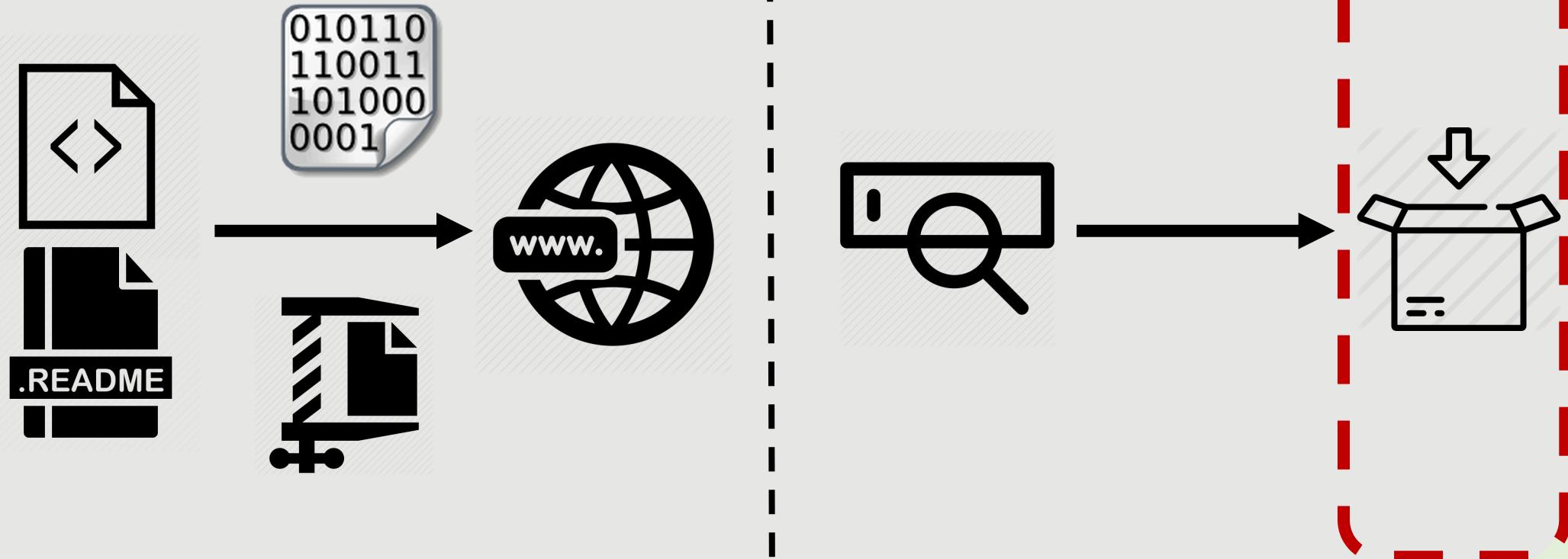


パッケージインストーラー (2008): pip
公開ツール(2013): bento / twine / flit
PyPI (2018): PyPI warehouse





データベース : PEP262 (2001/Deferred), PEP376 (2009)
Wheel: PEP425 (2012) / PEP427 (2012)





PEP376 データベースインターフェース

```
>>> import pkg_resources
>>> [k for k in dir(pkg_resources) if k.startswith('get')]
['get_build_platform', 'get_cache_path', 'get_default_cache', 'get_distribution',
 'get_entry_info', 'get_entry_map', 'get_importer', 'get_platform', 'get_provider',
 'get_supported_platform']
>>> pkg_resources.get_distribution('helloworld')
helloworld 0.1 (/Users/kirchou/test_lib/py38/lib/python3.8/site-packages/hello
world-0.1-py3.8.egg)
>>> pkg_resources.get_platform()
'macosx-10.14-x86_64'
```

xxx.dist-info/WHEEL

```
Wheel-Version: 1.0
Generator: bdist_wheel (0.34.2)
Root-Is-Purelib: true
Tag: py3-none-any
```



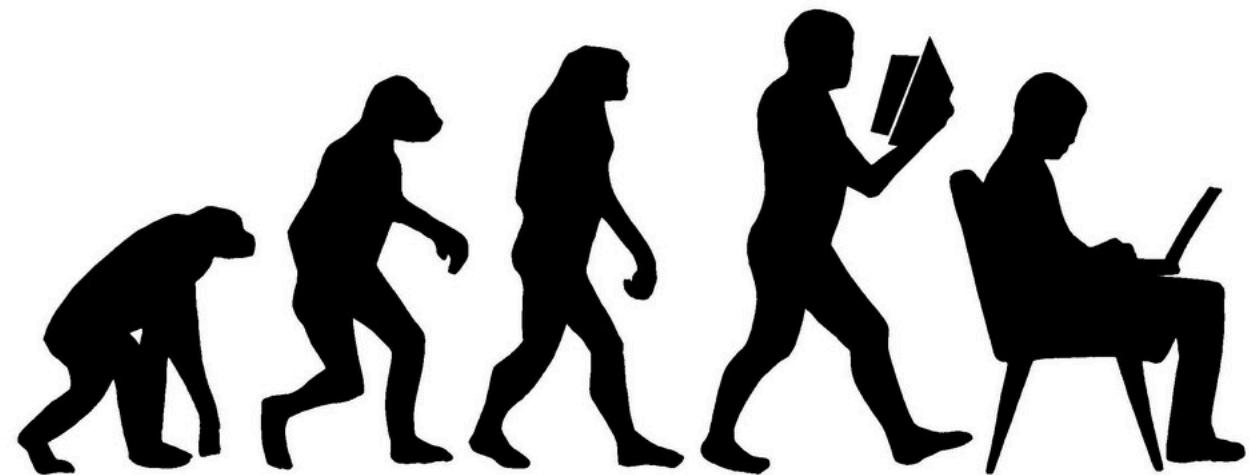


ワン・フォー・オール: Conda / Pipenv / Poetry

PEP517 (2015), PEP518 (2016)

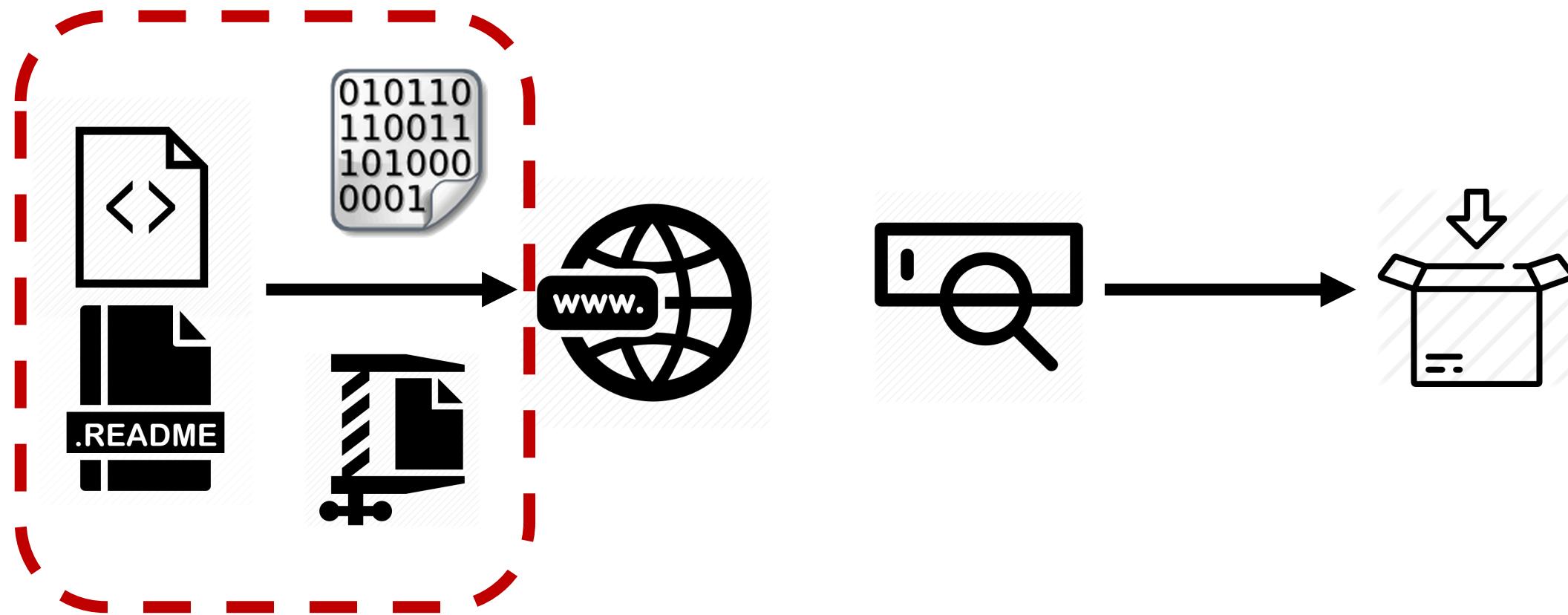


歴史から学ぶこと



1. メタデータスキーマの定義

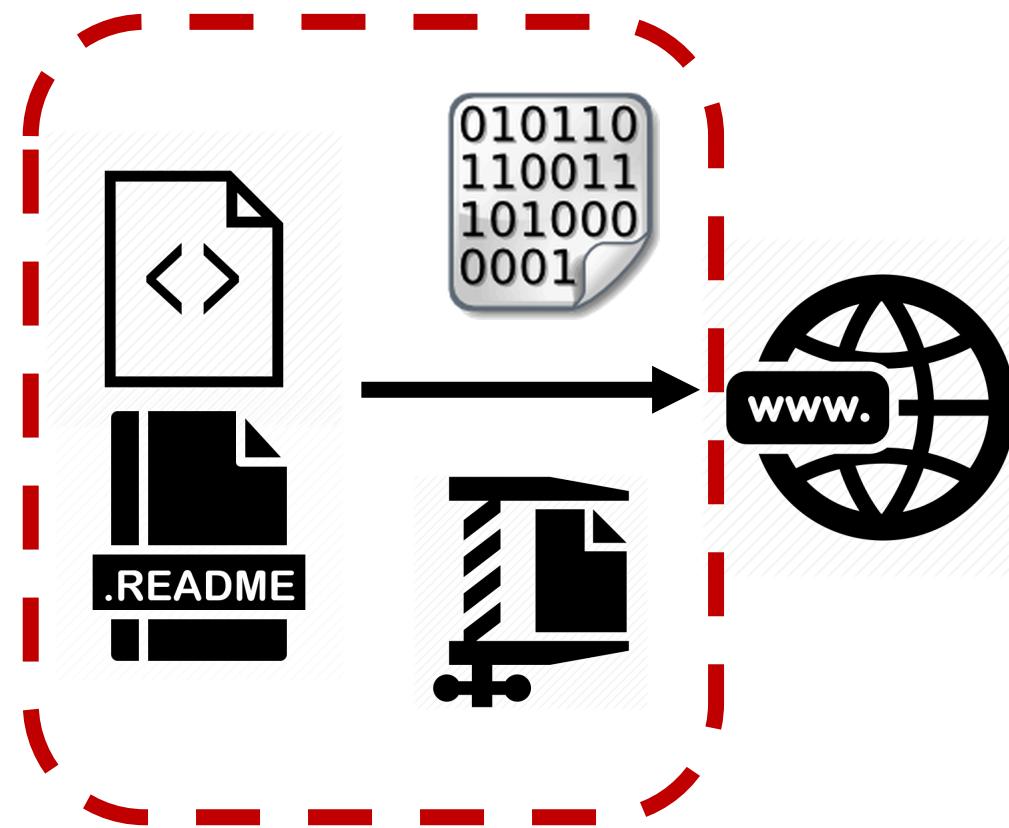
Python パッケージの進化



Python パッケージの進化

1. メタデータスキーマの定義

2. ビルドツール



1. メタデータスキーマの定義

Python パッケージの進化

2. ビルドツール



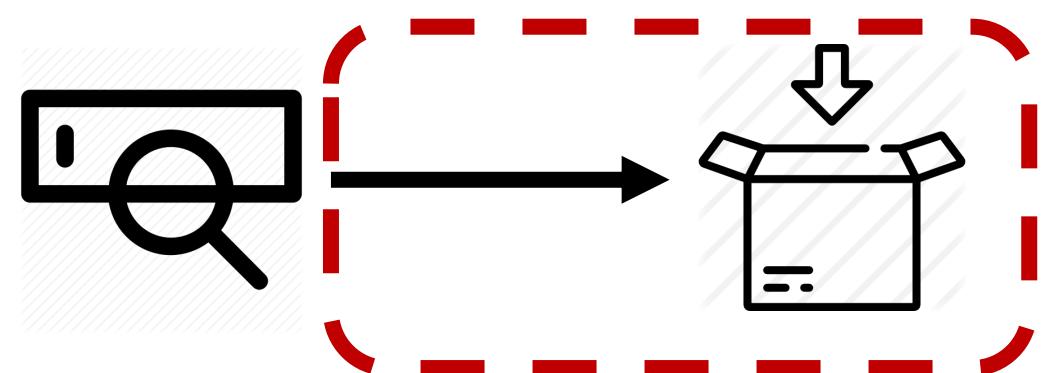
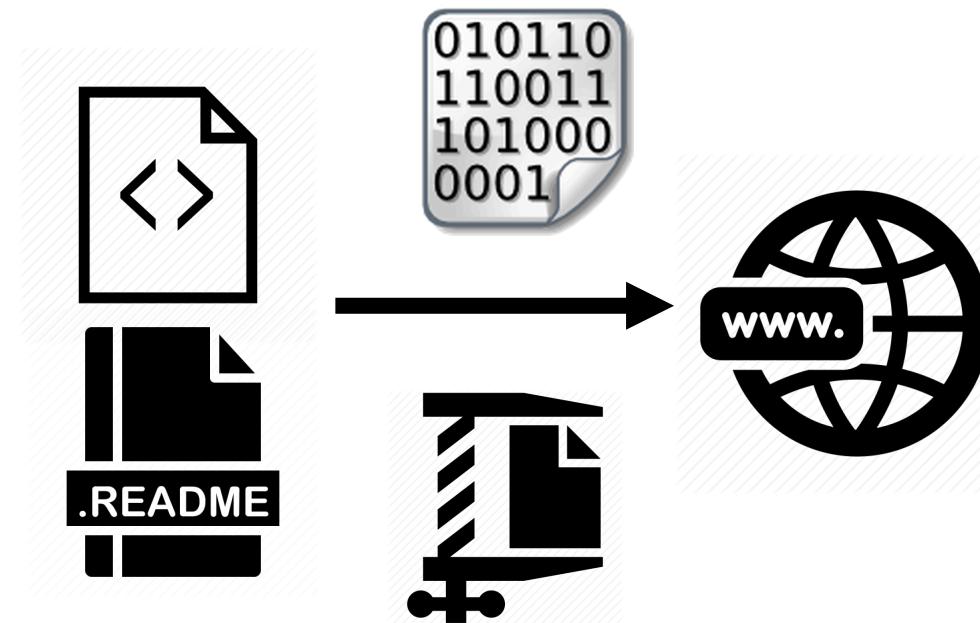
3. 公開ツール・PyPI

Python パッケージの進化

1. メタデータスキーマの定義

2. ビルドツール

4. インストーラ
デプロイツール



3. 公開ツール・PyPI

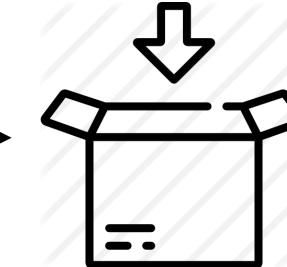
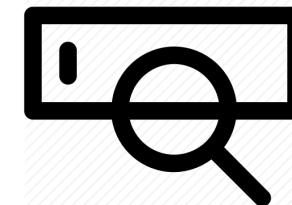
Python パッケージの進化

1. メタデータスキーマの定義

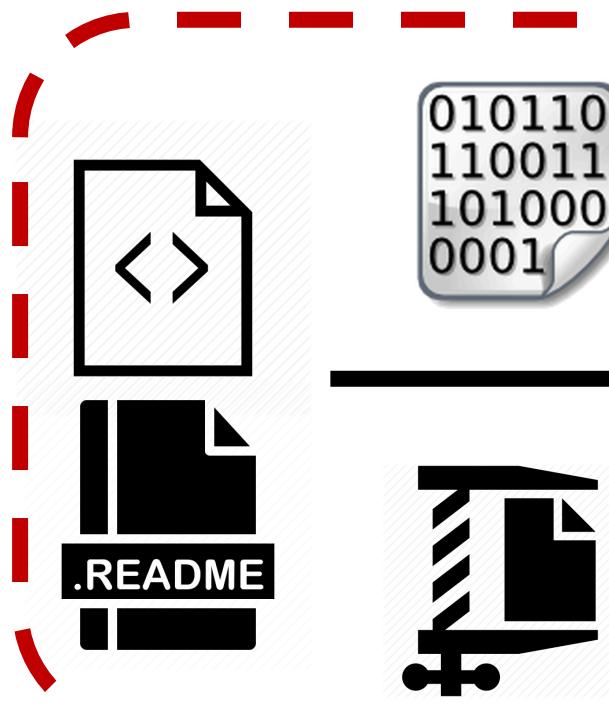
2. ビルドツール

4. インストーラ
デプロイツール

5. 複数 OS の問題



3. 公開ツール・PyPI



Python パッケージの進化

1. メタデータスキーマの定義

2. ビルドツール

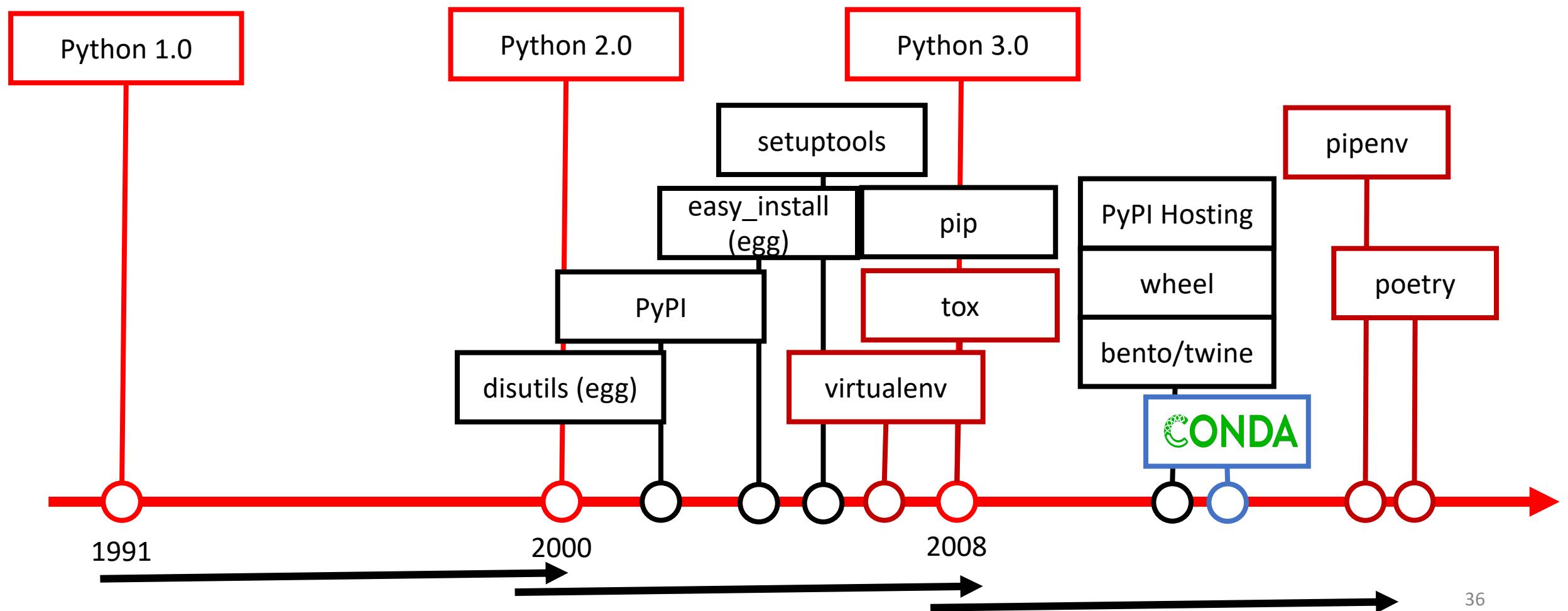
4. インストーラ
デプロイツール

5. 複数 OS とバージョンの問題



3. 公開ツール・PyPI

パターンは世代ごとに繰り返されます



コミュニティが解決する前の挑戦

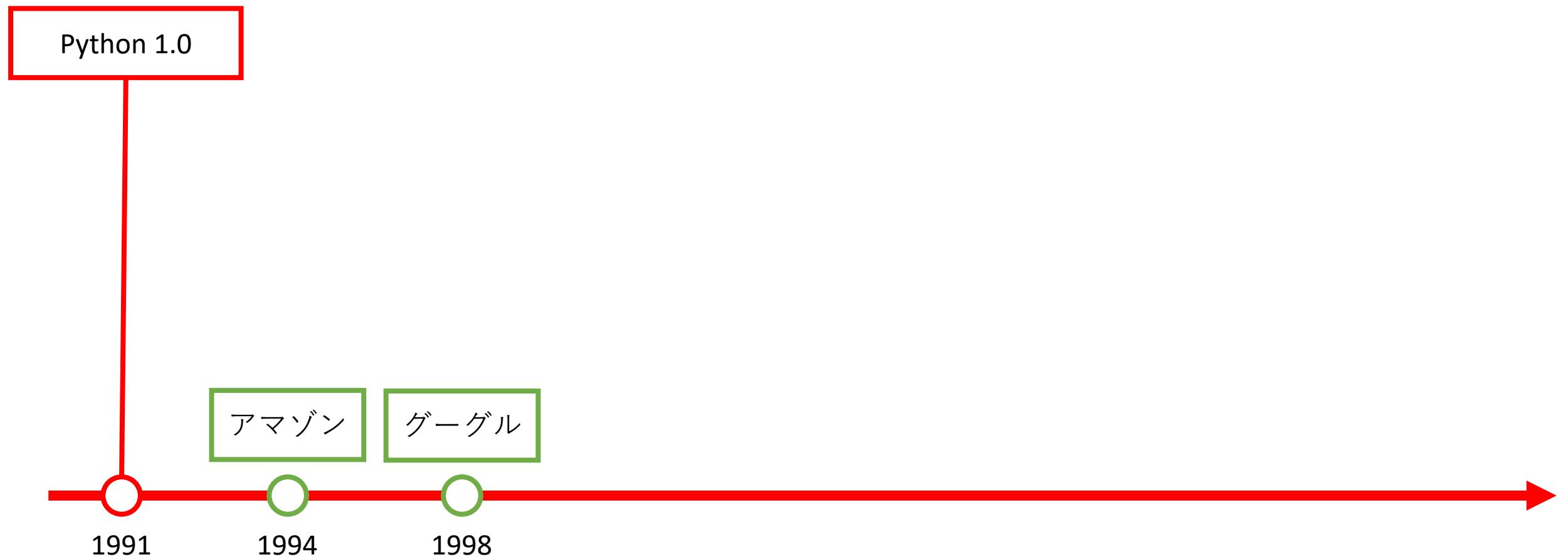
1994 アマゾン



1998 グーグル



Python パッケージに標準はない



ソースコードを他の人
に（から）与える（取る）ことは？



歴史の中で同じ問題を解決する

- ビルドツール
 - 依存関係リゾルバー
 - 複数Pythonバージョンのサポート
- 公開ツール / パッケージレジストリ
- インストーラー / デプロイツール
 - 仮想環境のサポート
 - 複数OSのサポート



1. 内部パッケージメタデータ

2. 内部構築ツール

4. 内部デプロイツール



3. 内部パッケージレジストリ

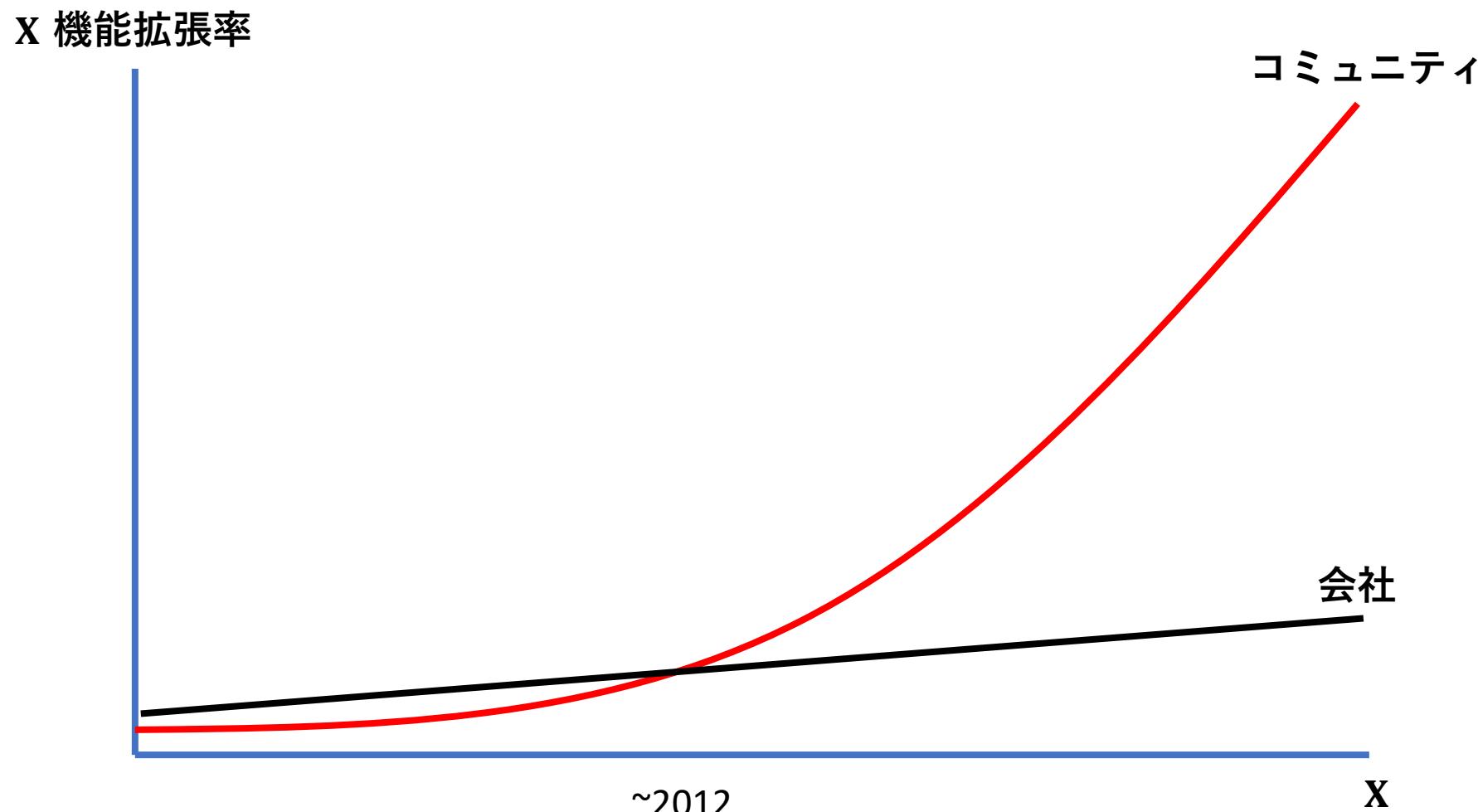


3pパッケージは
すべて内部ツールによって管理されます！



imgflip.com

コミュニティソリューションは
より速く確実に拡張されます



パッケージングの問題は ほとんどが一般的です

- ビルドツール
 - Python パッケージングは？
 - Python パッケージング依存関係はどうでしょうか？
 - Python パッケージをビルドする方法は？
- 公開ツール / パッケージレジストリ
 - Python パッケージを公開する方法は？
 - 他のPythonの依存関係をダウンロードする方法は？
- インストーラー / デプロイツール
 - Python 環境を分離する方法は？（仮想環境）
 - Python パッケージはどのOSをサポートしていますか？

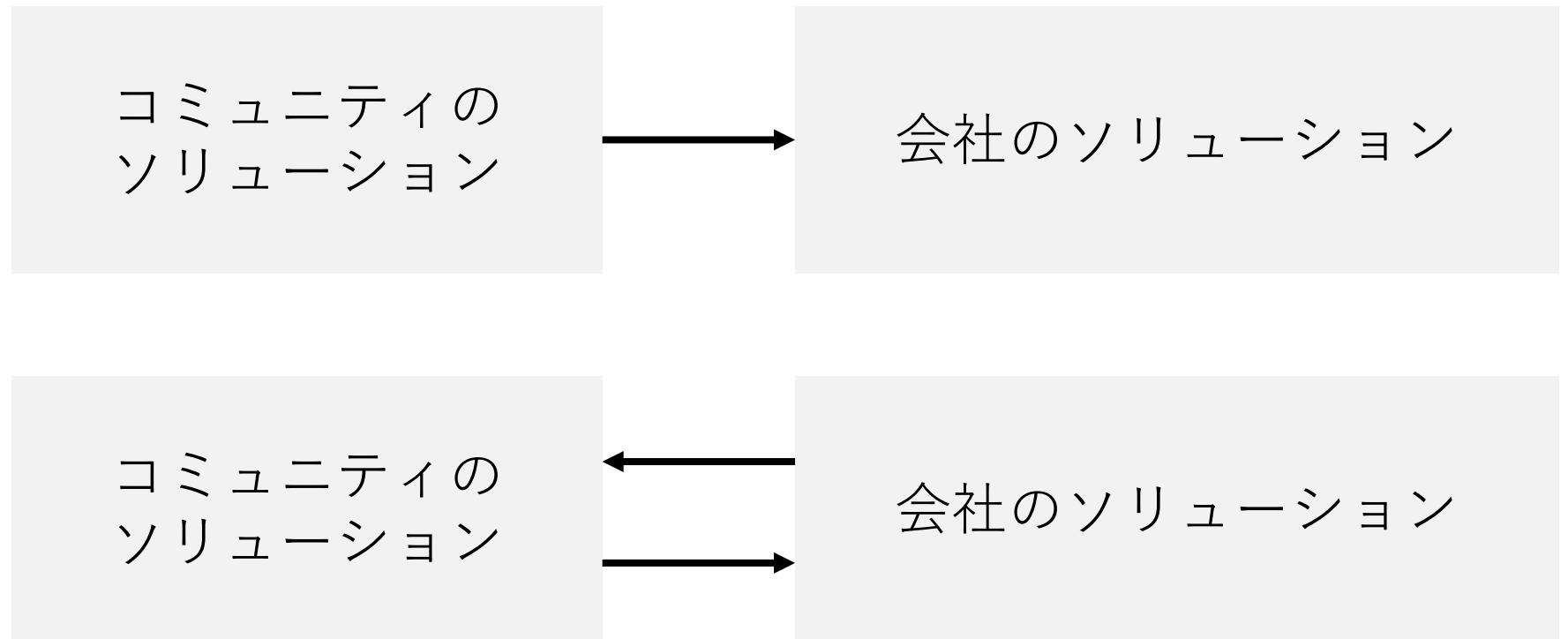




これがコミュニティソリューションに
移行する理由です

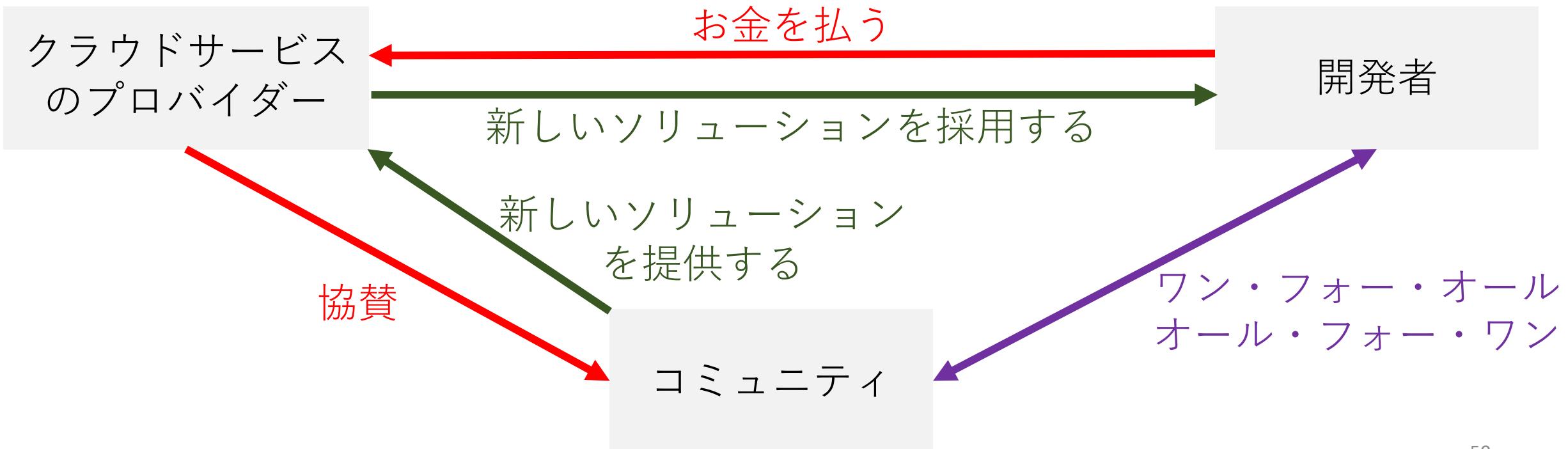
コミュニティが解決した後の挑戦

1. 企業とコミュニティの理想的な関係





クラウドサービス のビジネス



2. メタデータ標準の数 \propto ツールの数



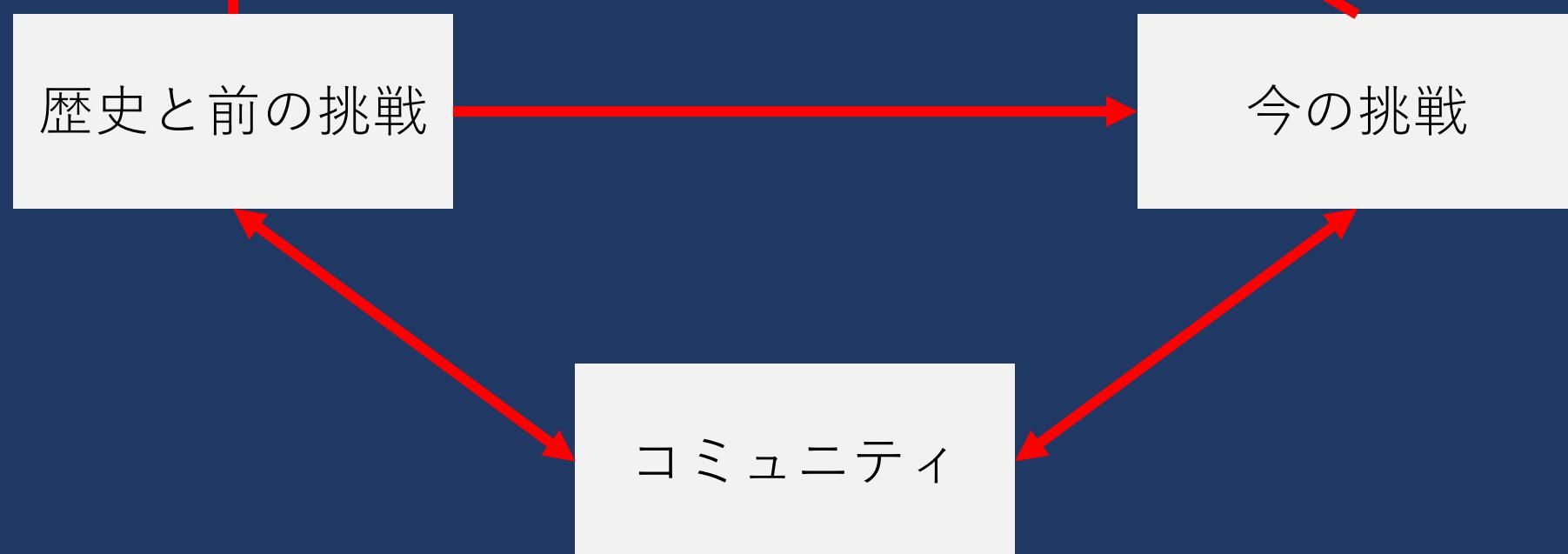




PyConに参加しましょう！
コミュニティに参加しましょう！

まとめ

古きをたずねて新しきを知る



背景資料

歴史

- Dustin Ingram @ SciPy 2018
Inside the Cheeseshop: How Python Packaging Works
- History of packaging written by Martijn Faassen
- Kenneth Reitz @ PyCon 2018
Pipenv: The Future of Python Dependency Management
- Clinton Roy @ Kiwi PyCon X (2019)
The Packaging Lifecycle with Poetry
- History of wheel written by Daniel Holth

パッケージ

- Dave Forzac @ PyOhio 2015
[Python Packaging from Init to Deploy](#)
- Elana Hashman @ PyCon 2019
[The Black Magic of Python Wheel](#)
- [Official Document: Packaging binary extensions](#) (2013)

デプロイ（仮想環境）

- Carl Meyer @ PyCon 2011
[Reverse-engineering Ian Bicking's brain: inside pip and virtualenv](#)
- Bernat Gabor @ EuroPython 2019
[Status quo of virtual environments](#)



ご静聴ありがとうございました！