

PYTHON PACKAGING

Where we are
Where we're headed

WHO?

Pradyun Gedam
@pradyunsg pradyunsg.me

Member of
Python Packaging Authority

Maintainer of
pip, virtualenv, packaging and more

PSF Fellow

College student!

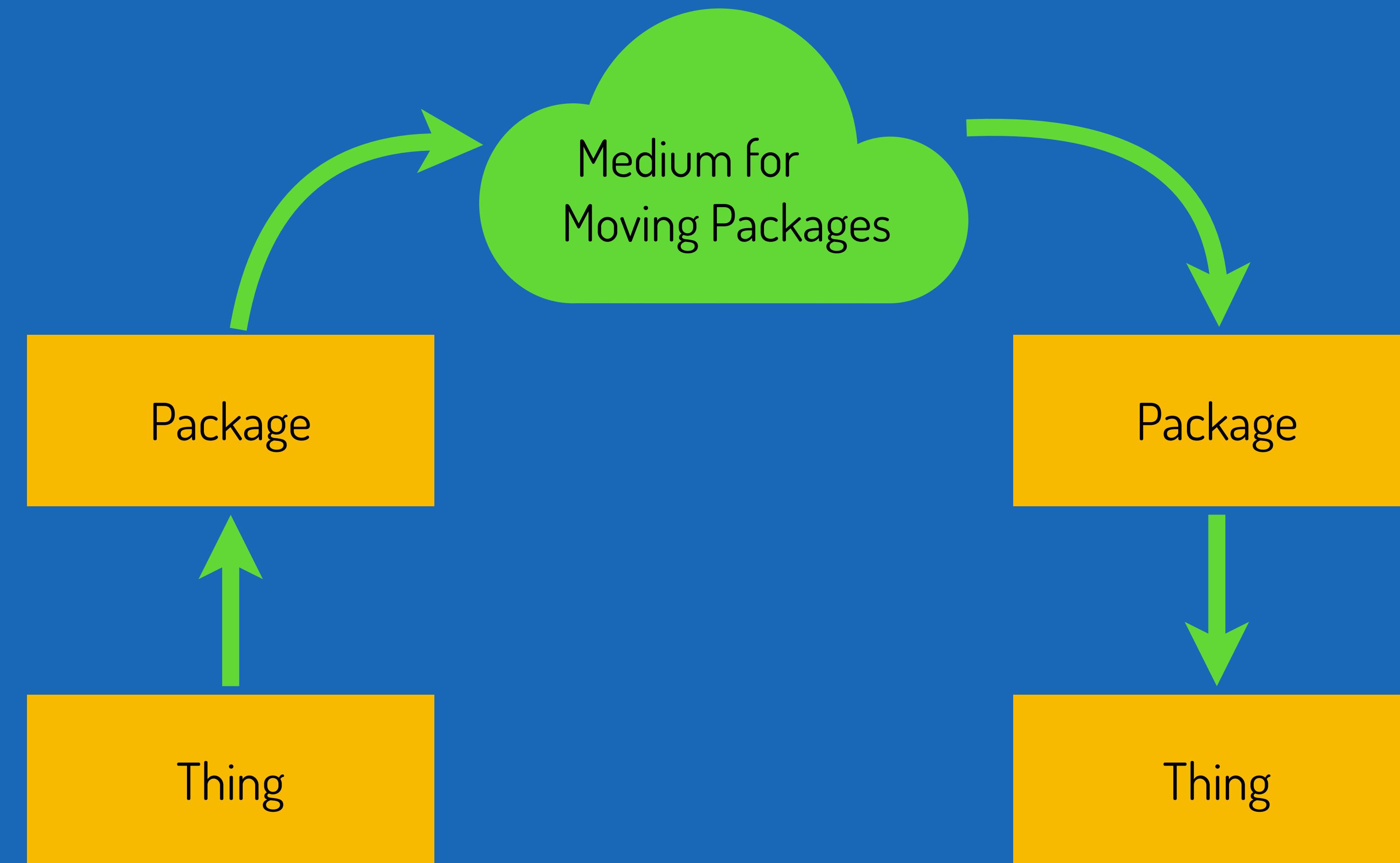
WHERE WE ARE

An Overview of
Tooling in the Python Packaging ecosystem

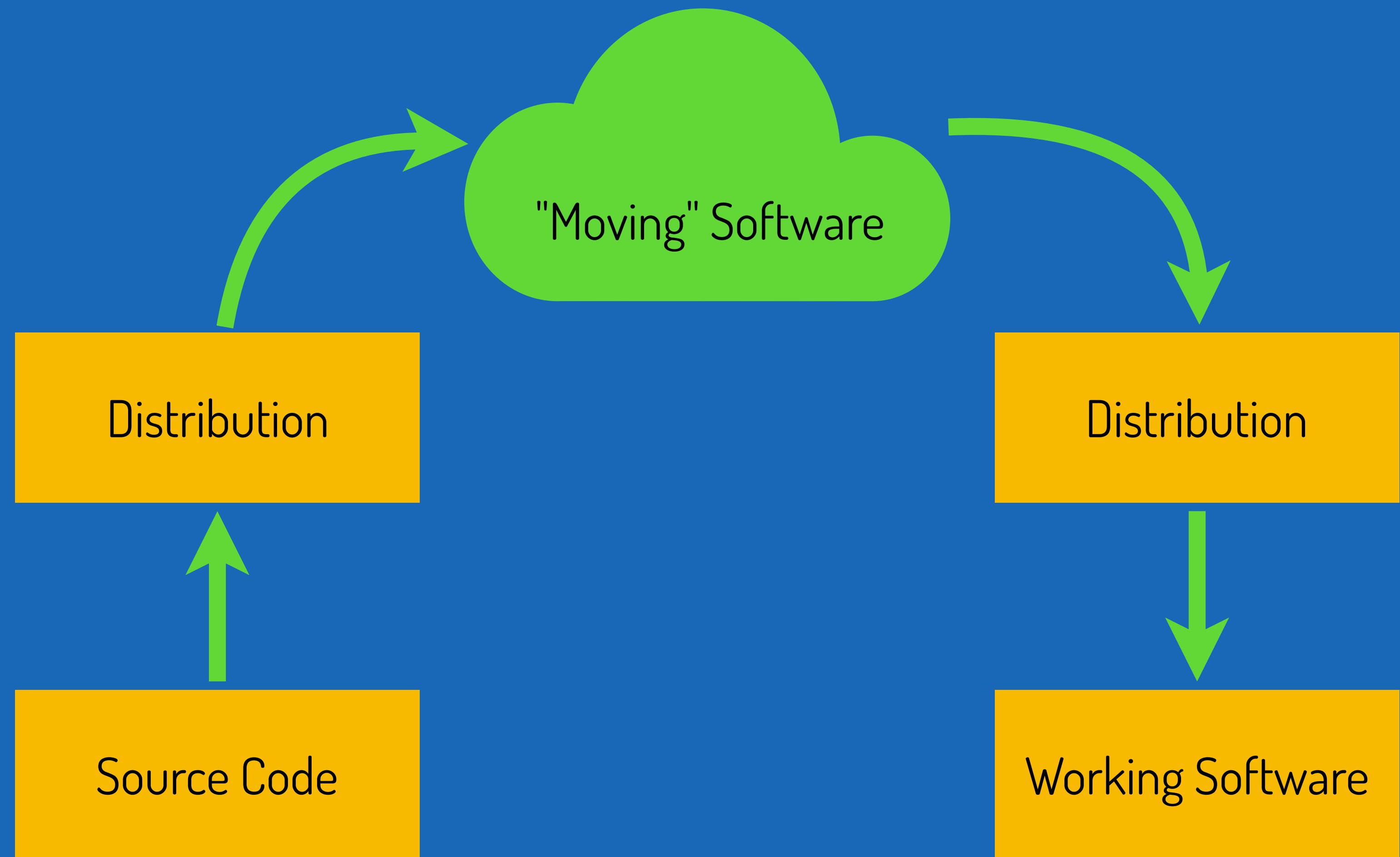
NOT HOW WE GOT HERE

<https://www.youtube.com/watch?v=AQsZsgJ30AE>

WHAT'S PACKAGING?



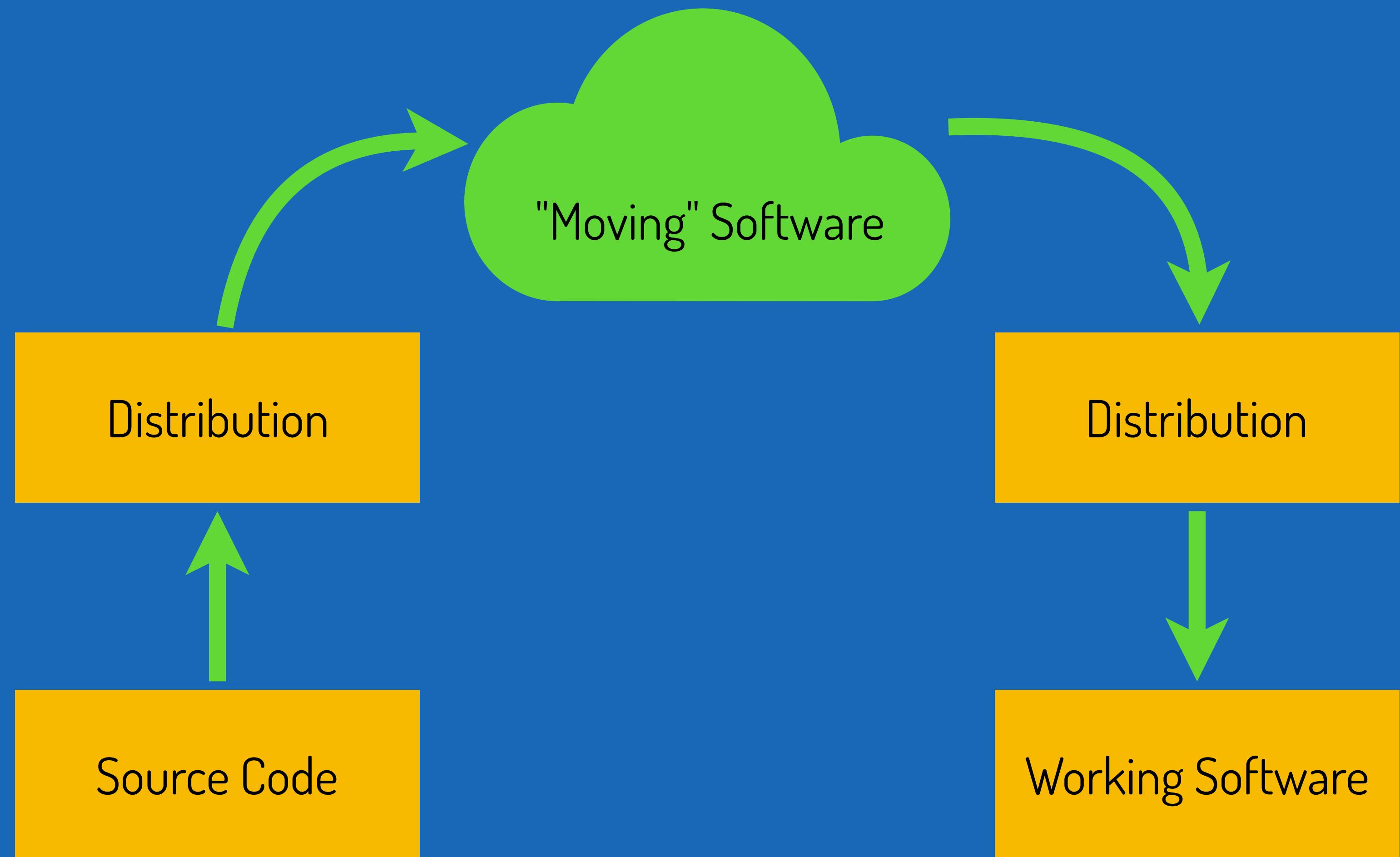
SOFTWARE DISTRIBUTION

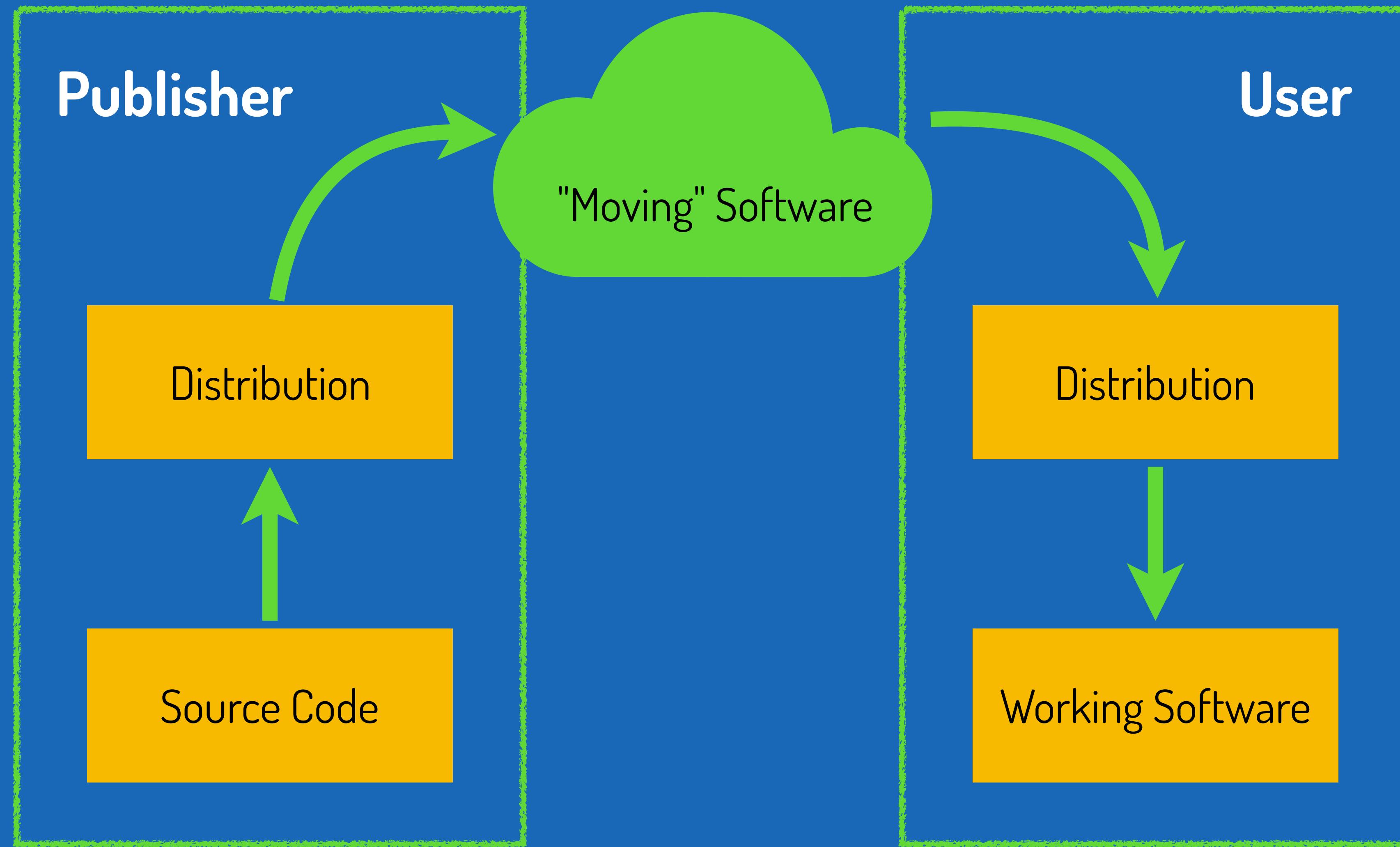


SOFTWARE DISTRIBUTION

AN UNSOLVED PROBLEM

https://youtu.be/lVzjVqr_Bzs?t=430





Publisher

Source Code

Publisher

Distribution

"build"

Source Code



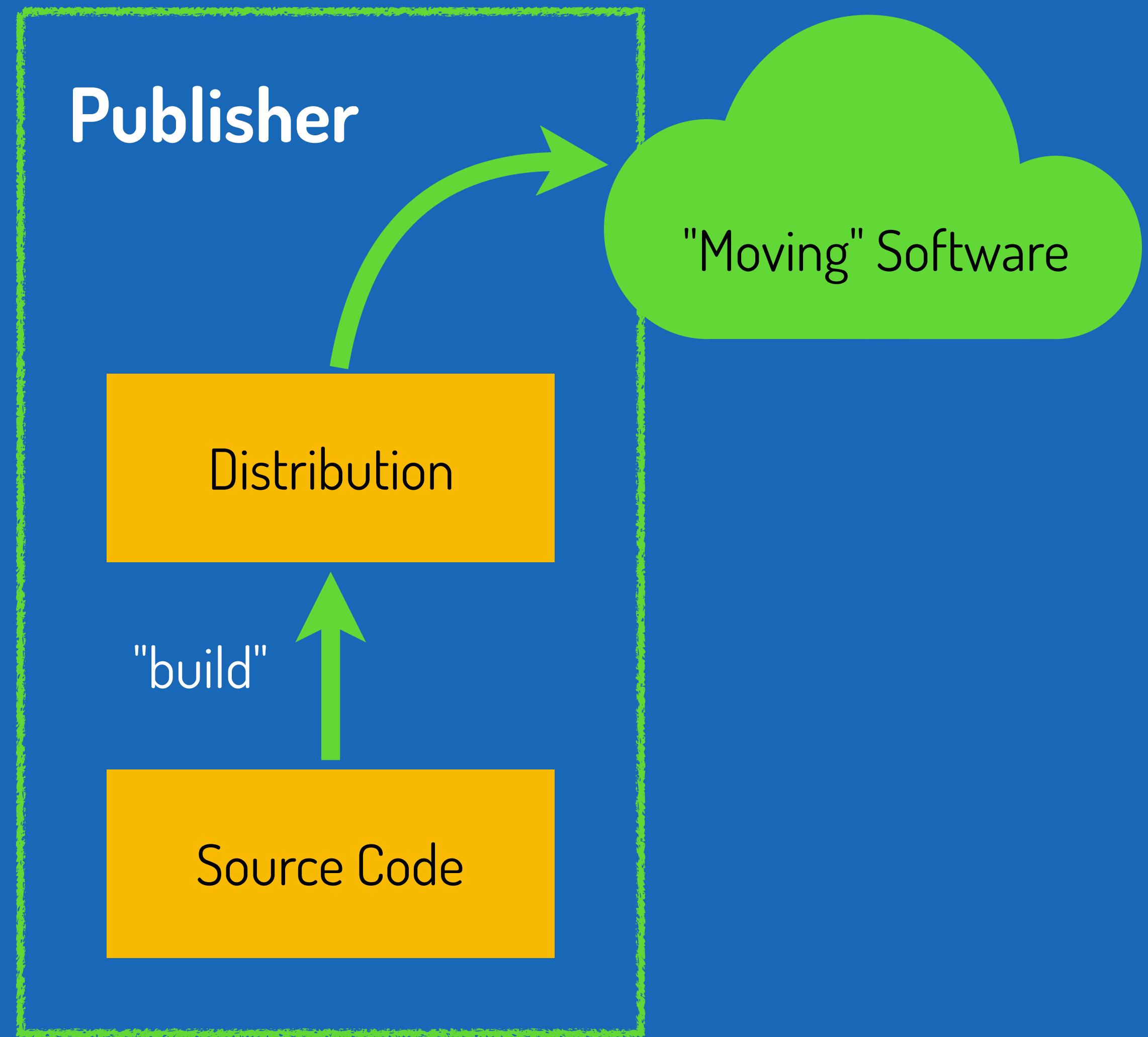
Publisher

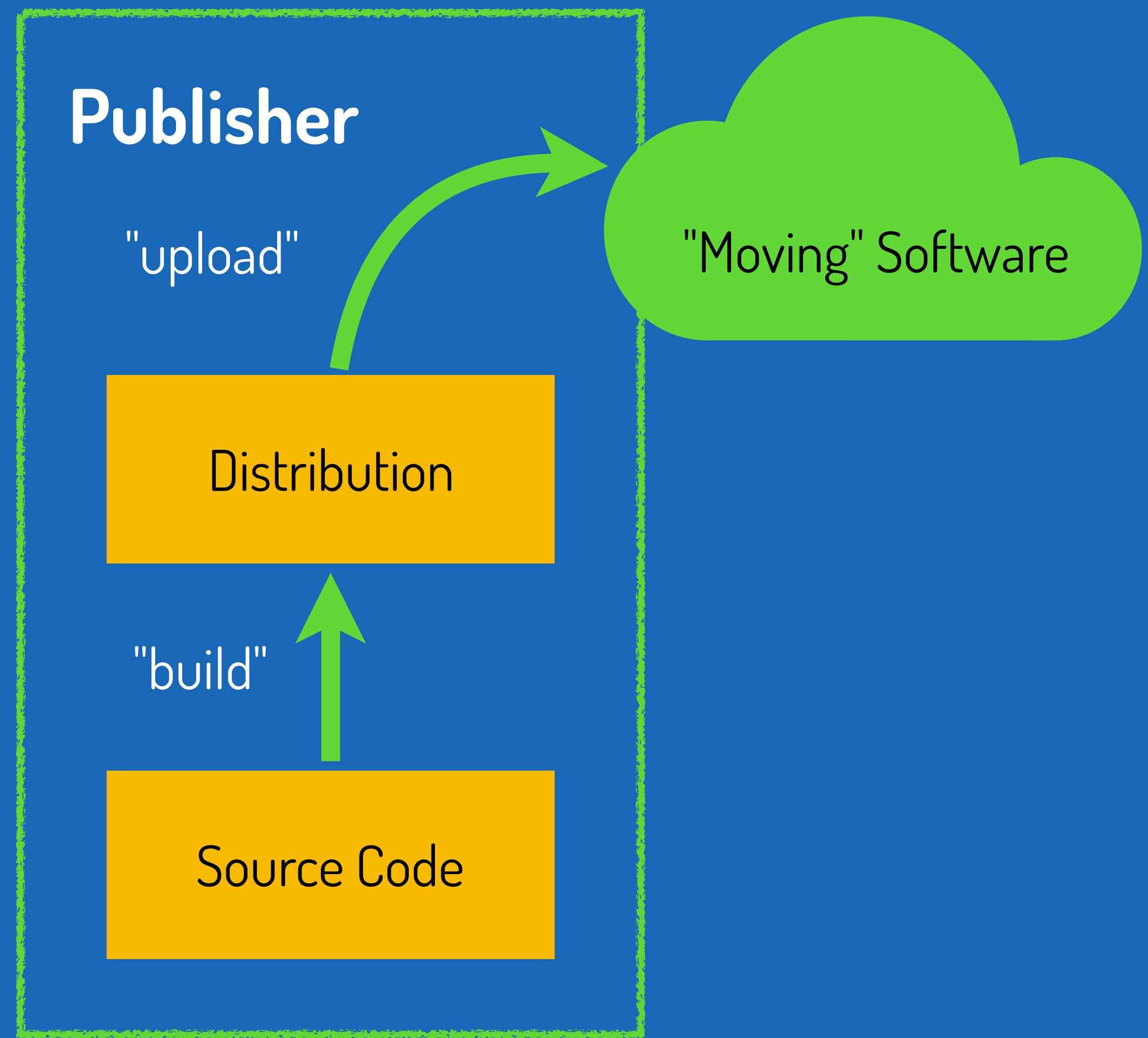
Distribution

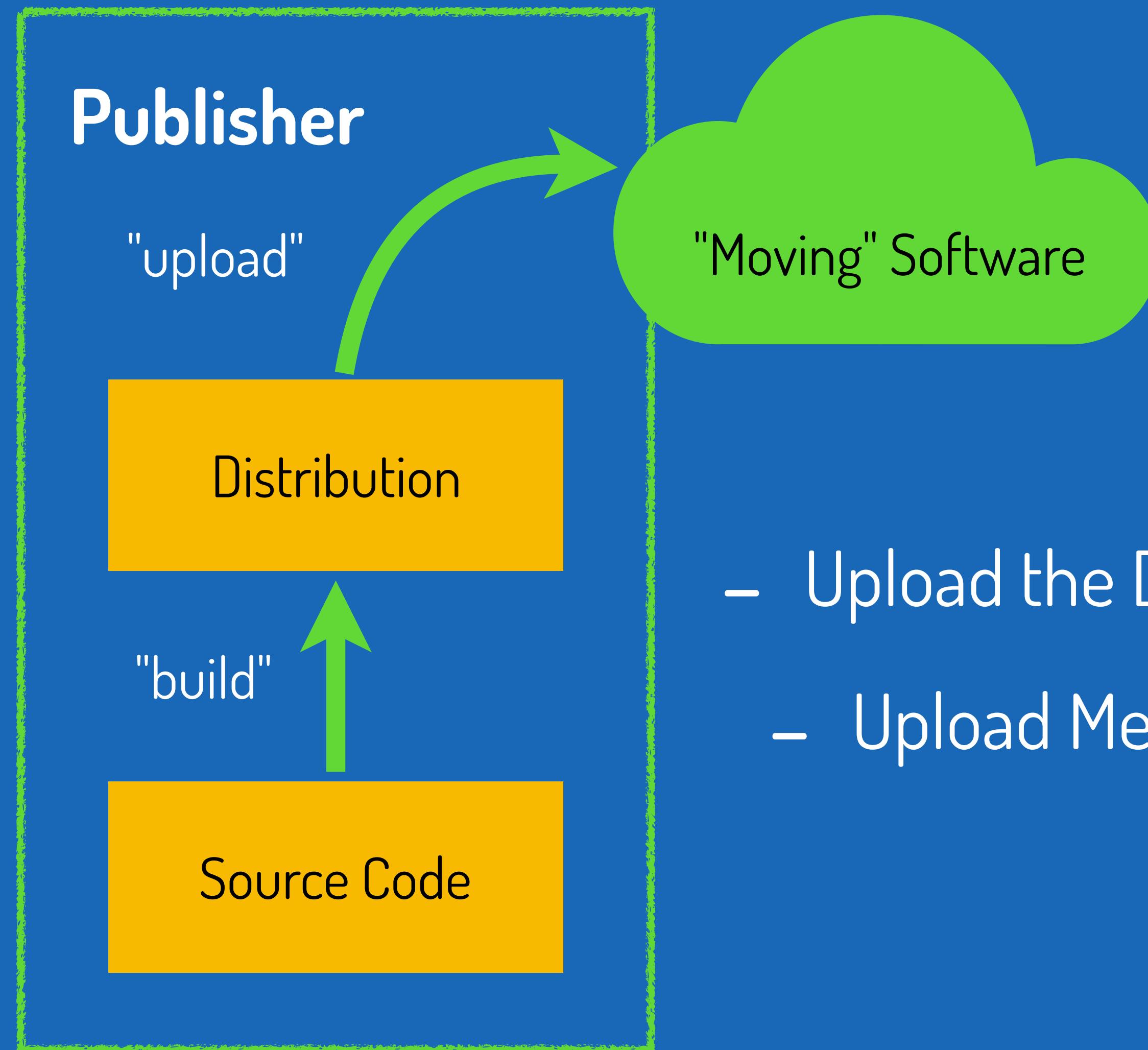
"build"

Source Code

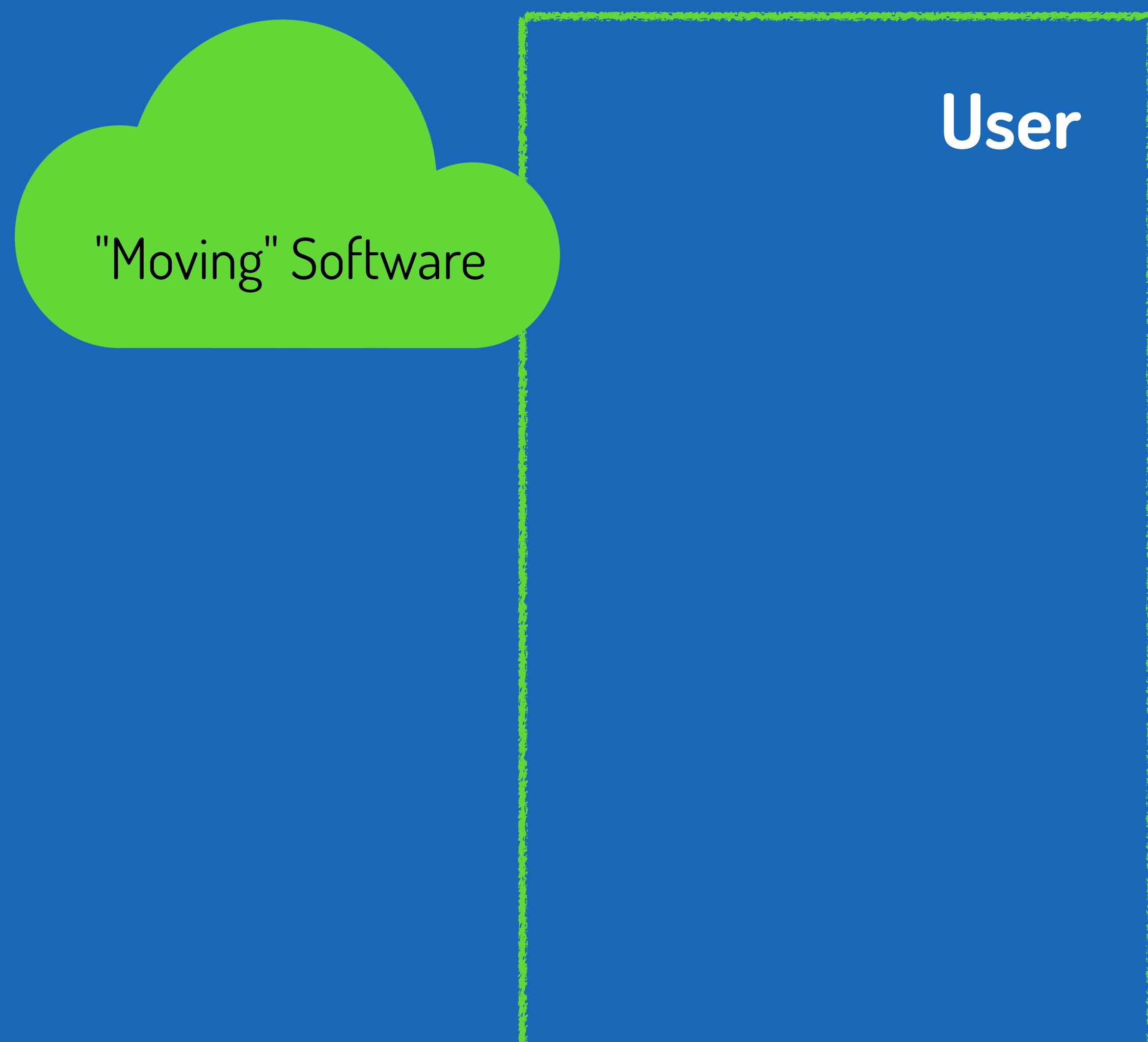
- Build a Distribution from Source Code
 - Build Environment
 - Build Mechanisms





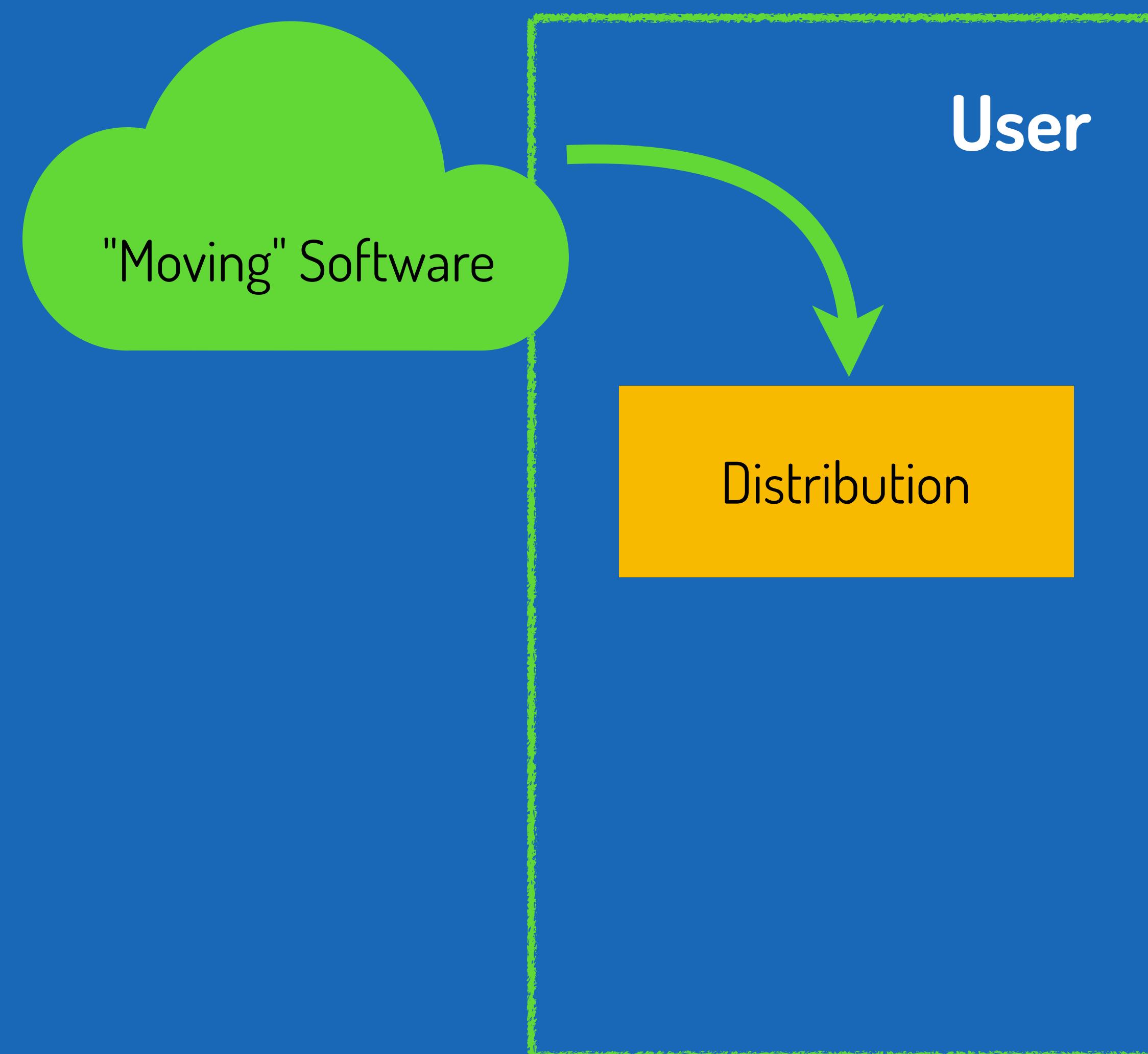


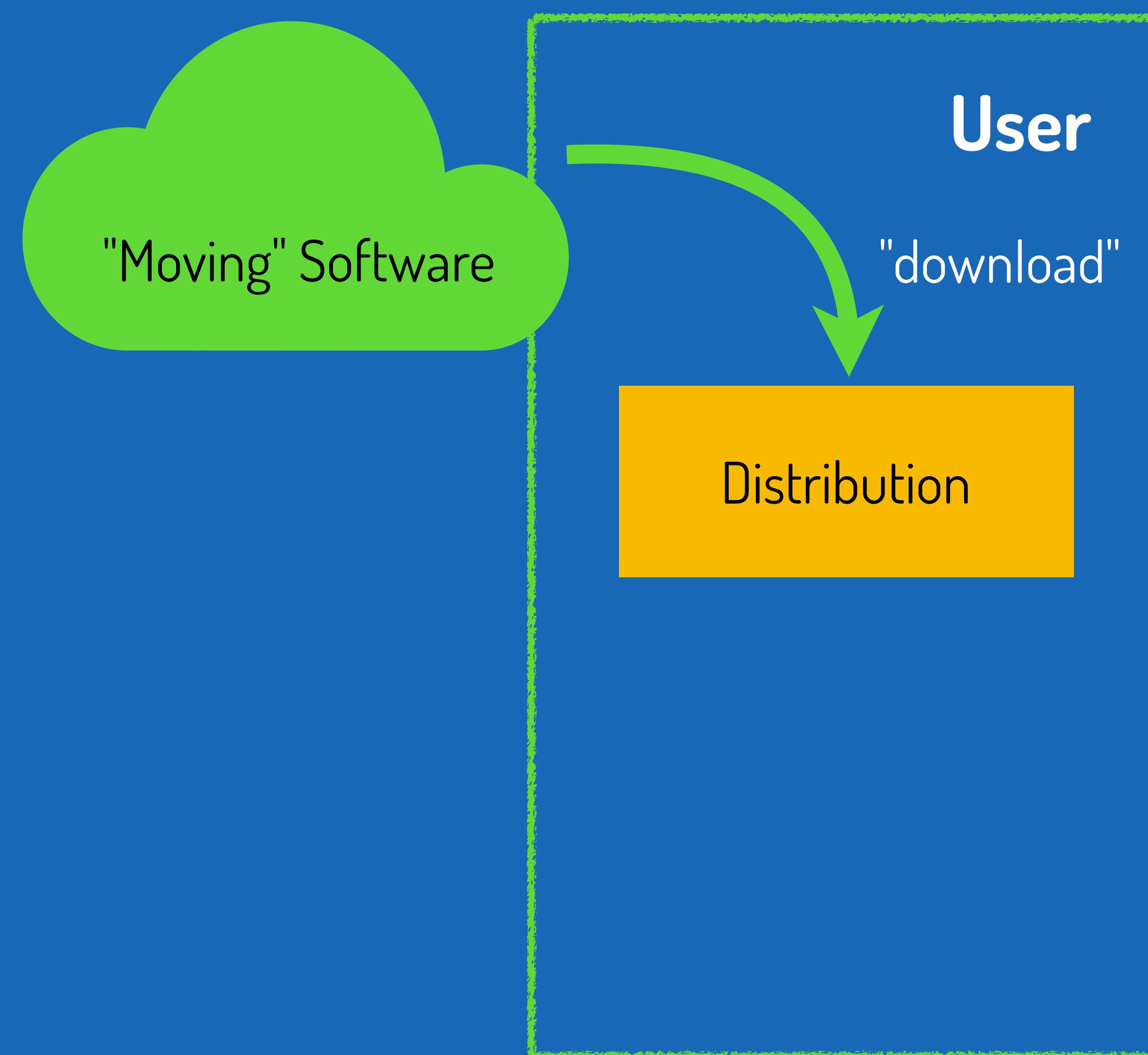
- Upload the Distribution
- Upload Mechanisms

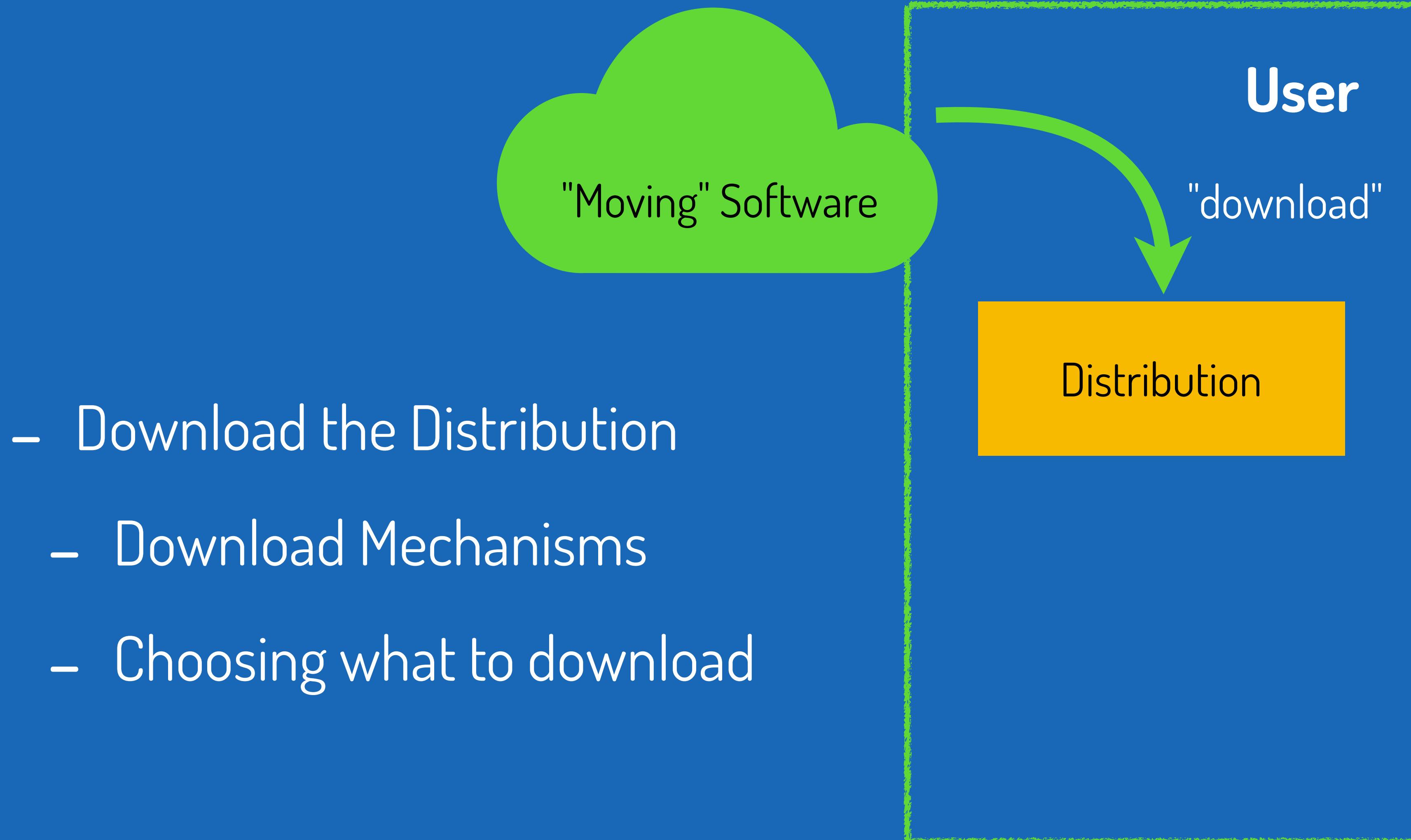


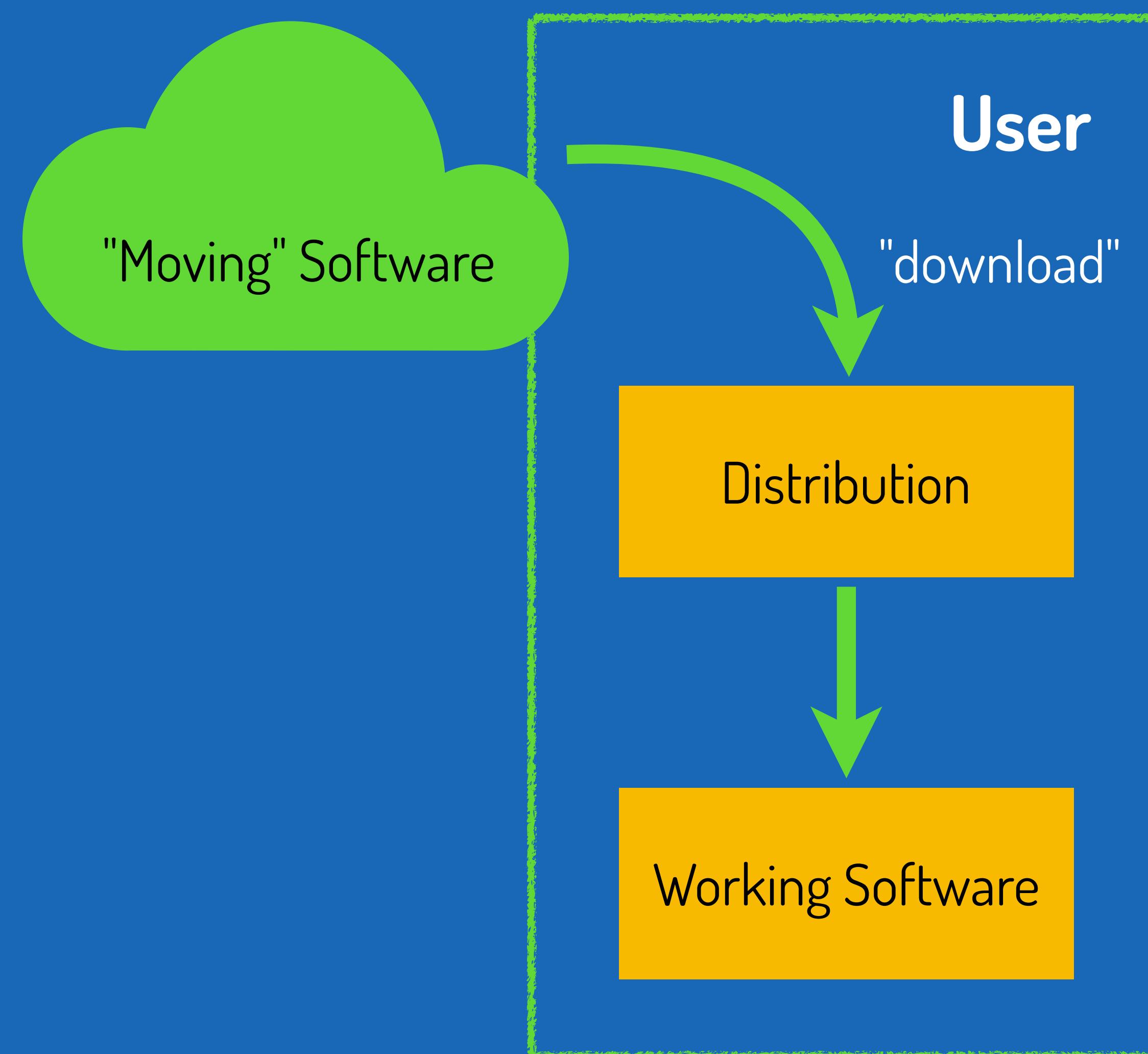
User

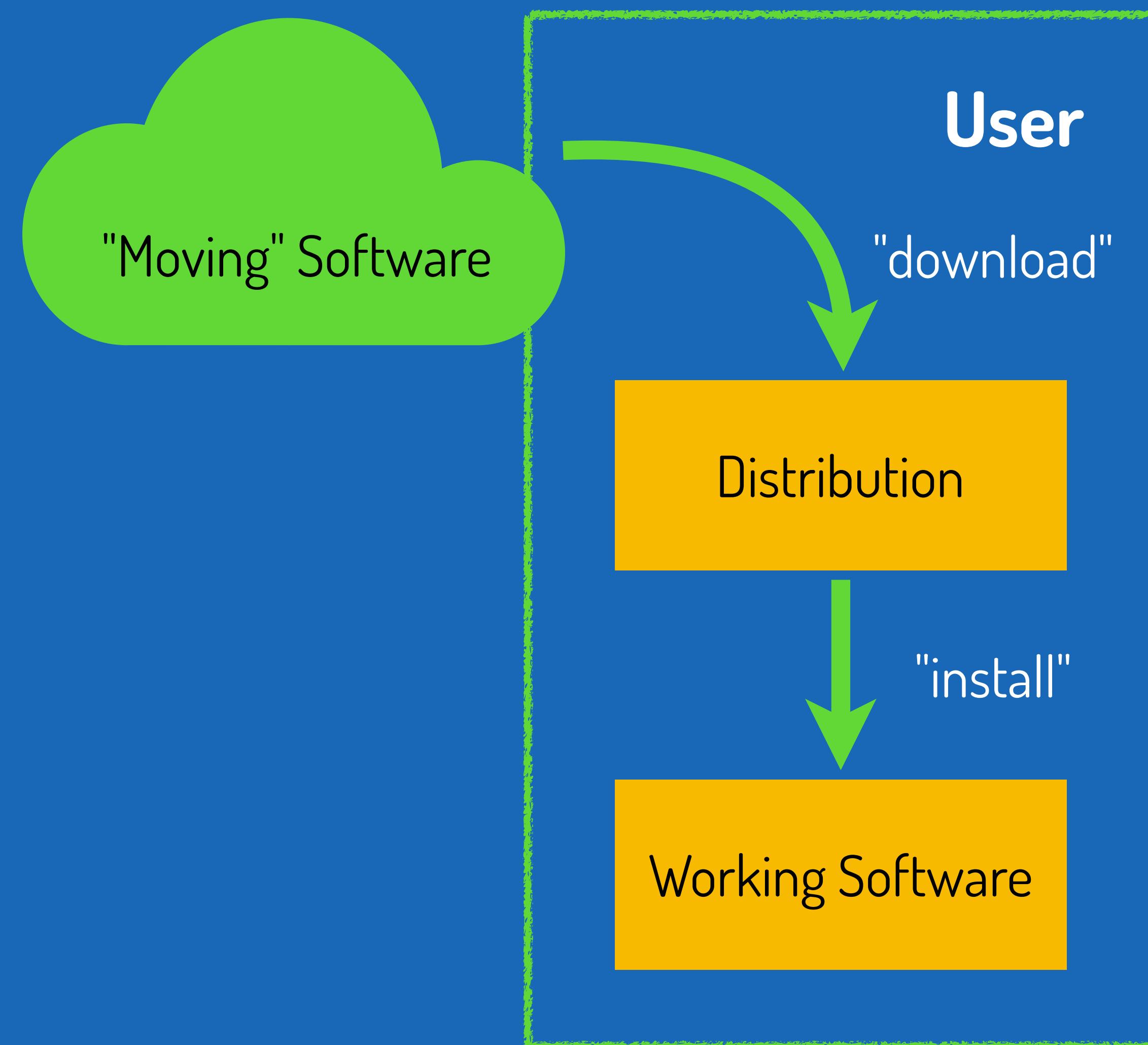
"Moving" Software



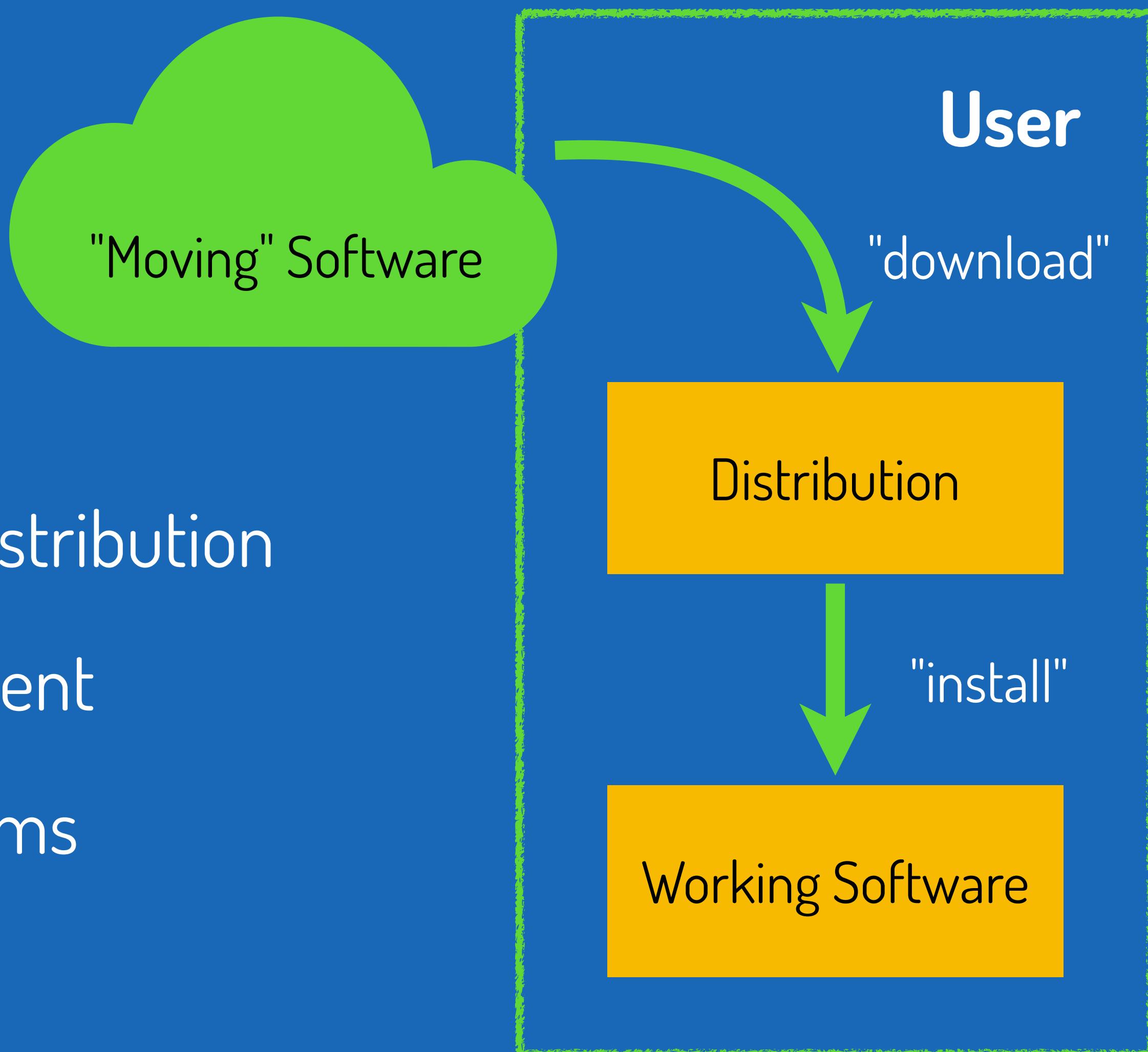


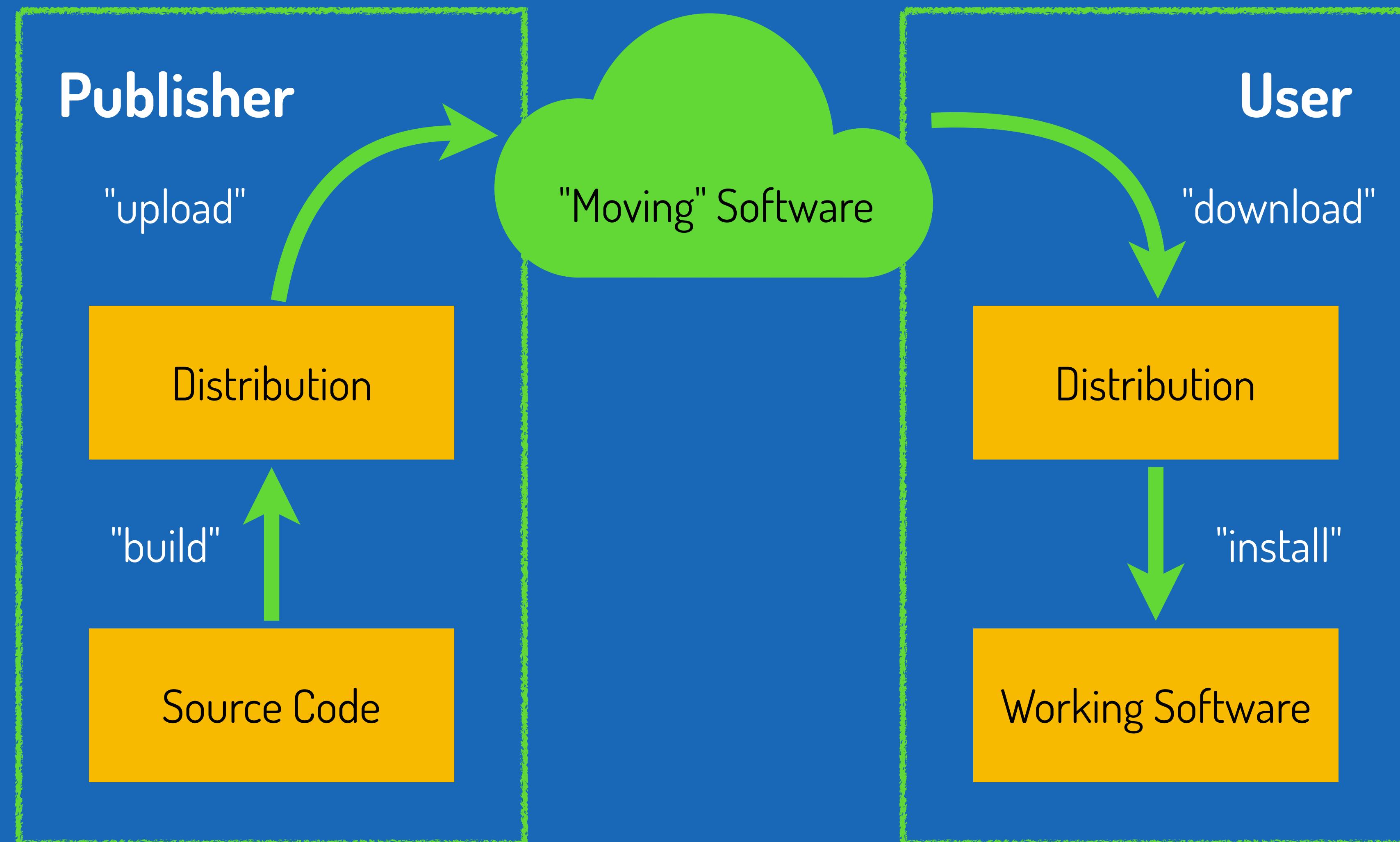


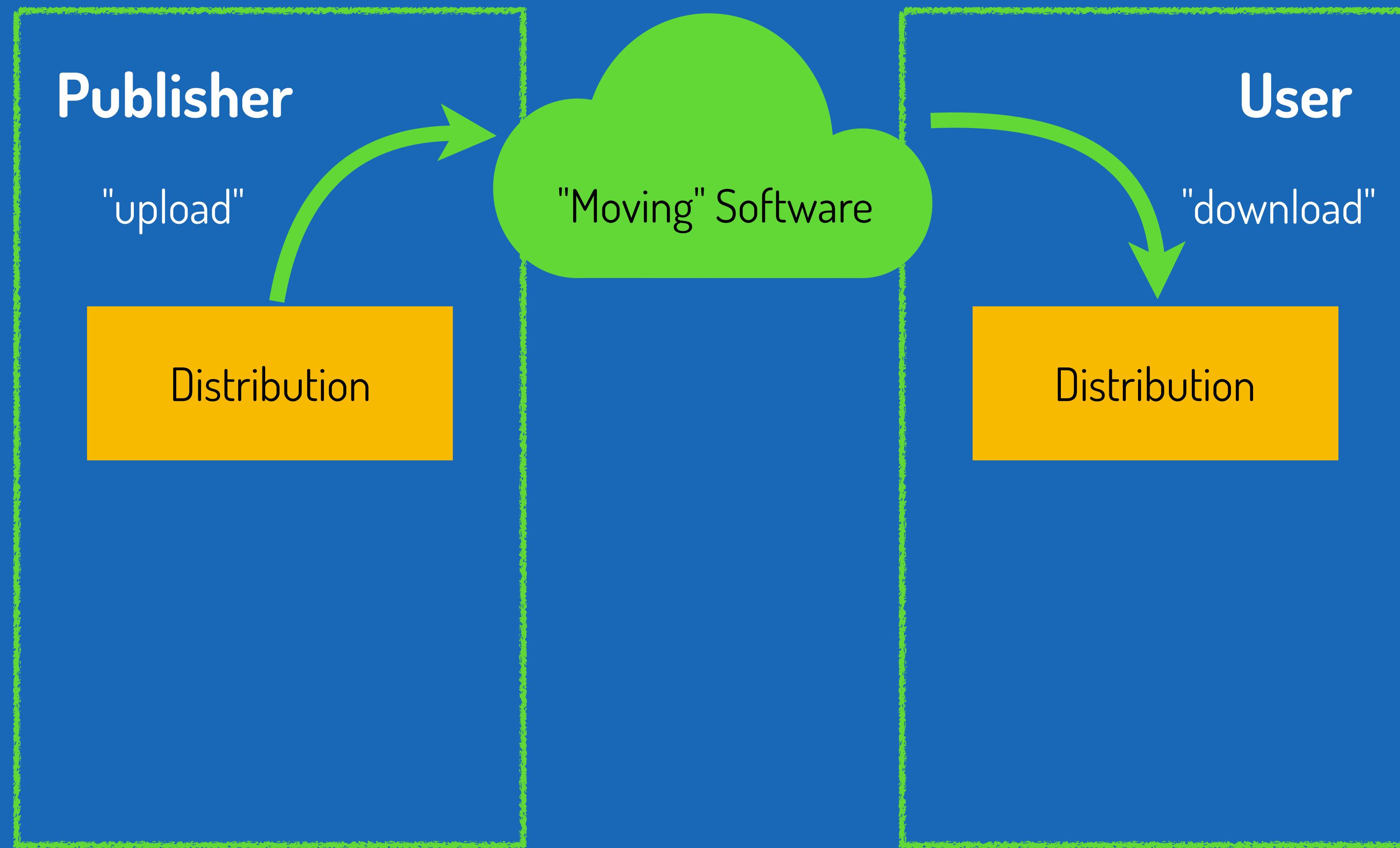


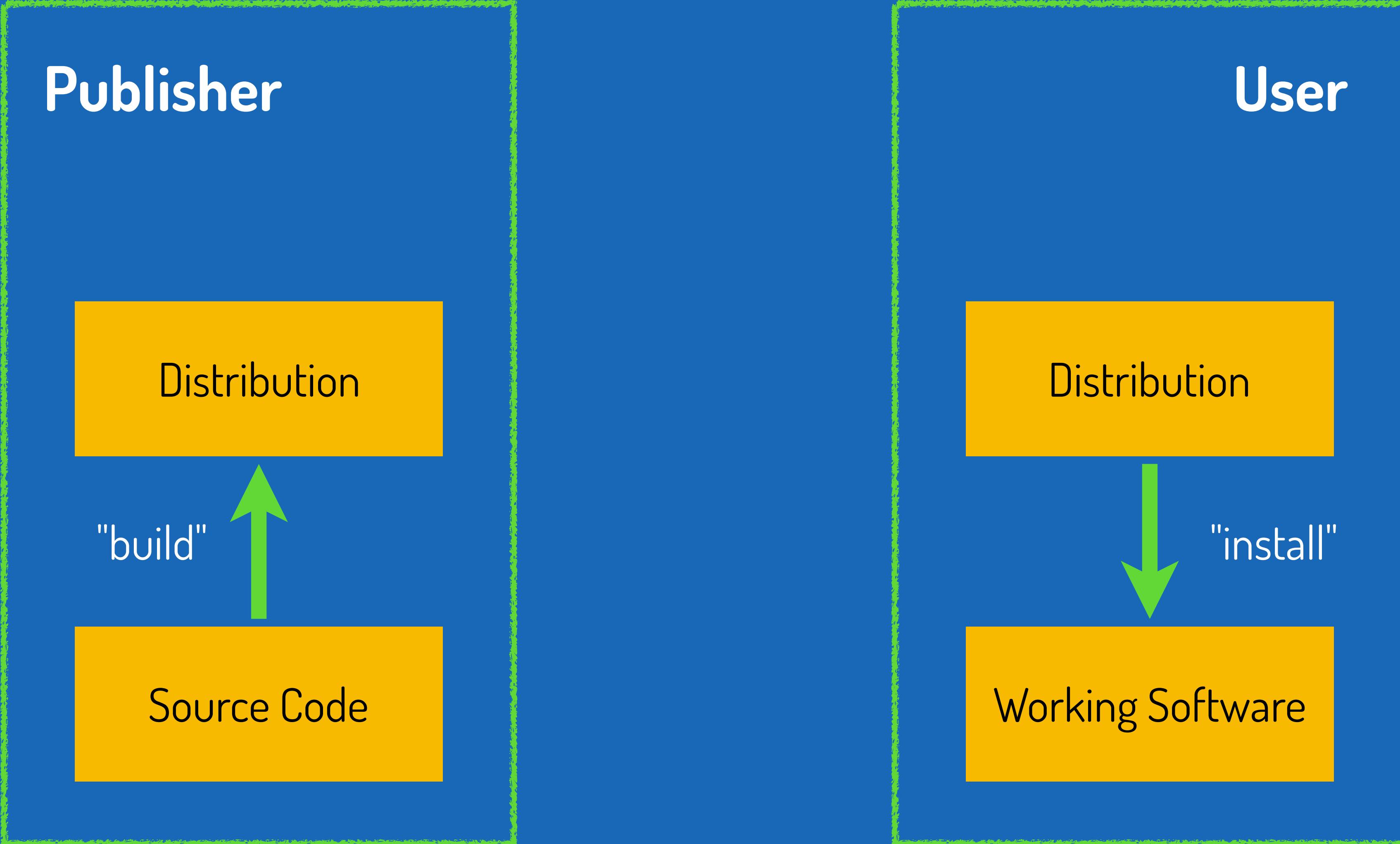


- Install from the Distribution
- Install Environment
- Install Mechanisms









DEALING WITH ENVIRONMENTS

- The Packaging Gradient
https://sedimental.org/the_packaging_gradient.html
<https://www.youtube.com/watch?v=iLVNwfPWAC8>
- Python Packaging Overview
<https://packaging.python.org/overview/>

PyPA TOOLING

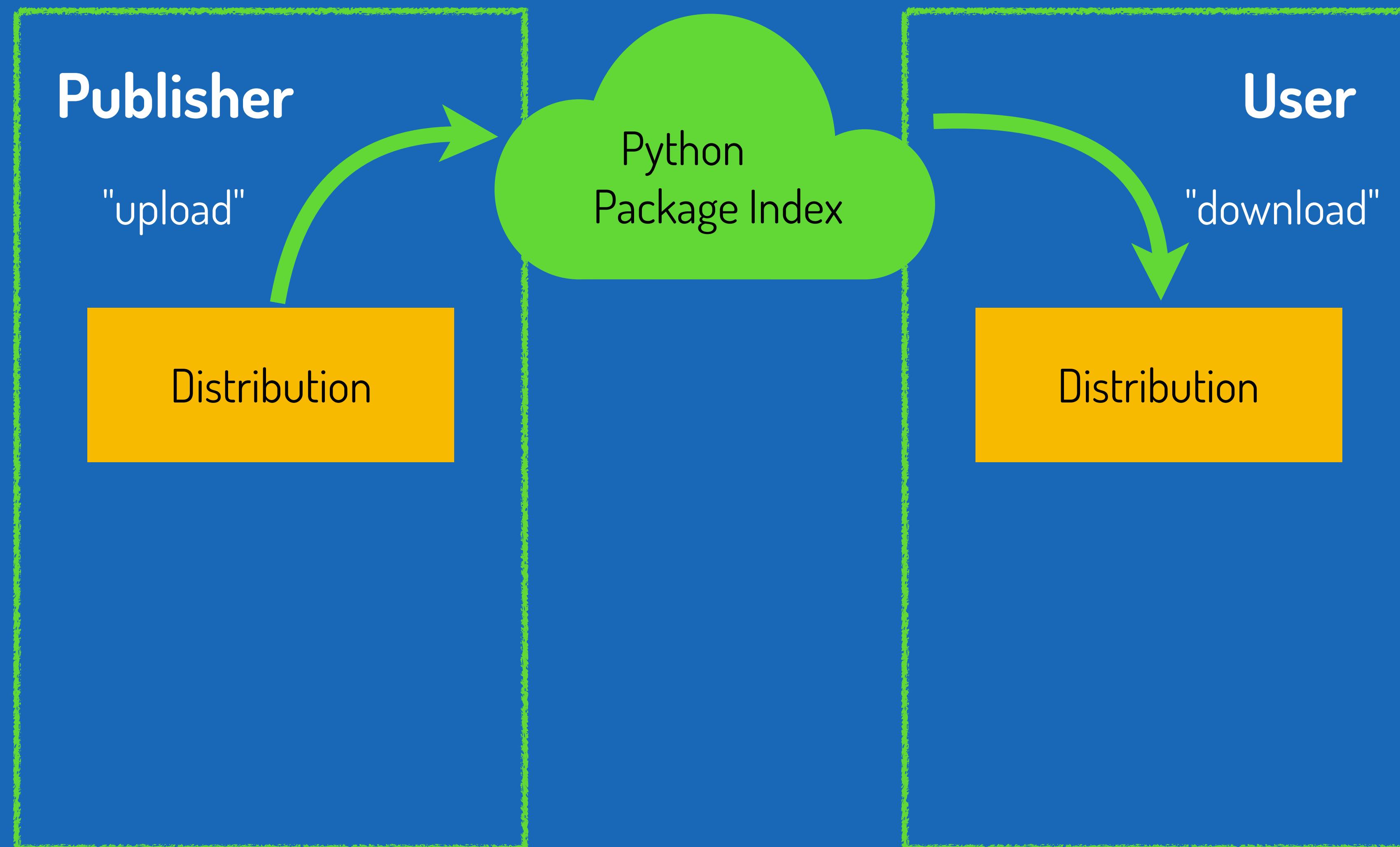
Tools built by volunteers

Built on standards

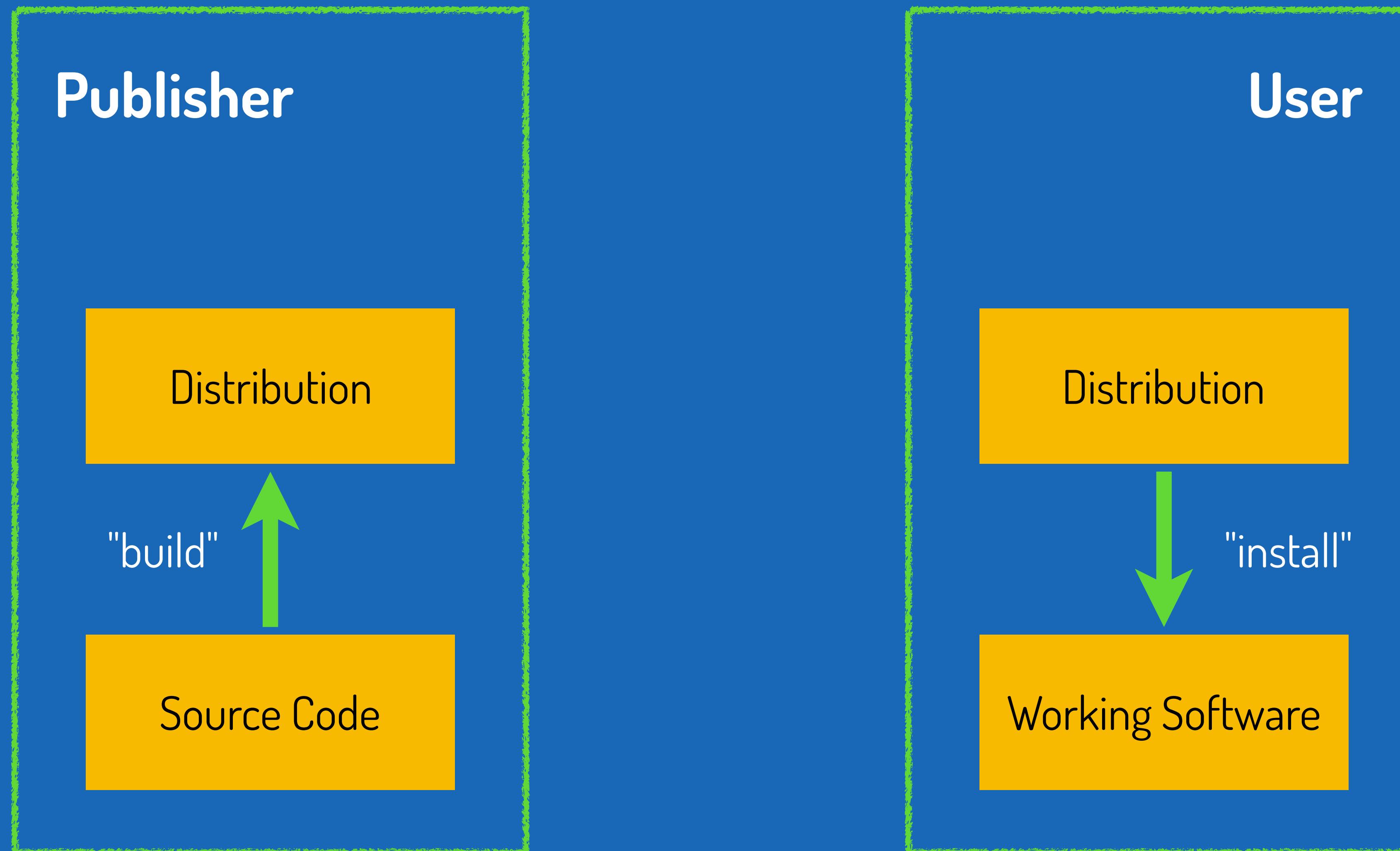
(mostly)

<https://packaging.python.org/specifications/>

Package Index Interfaces



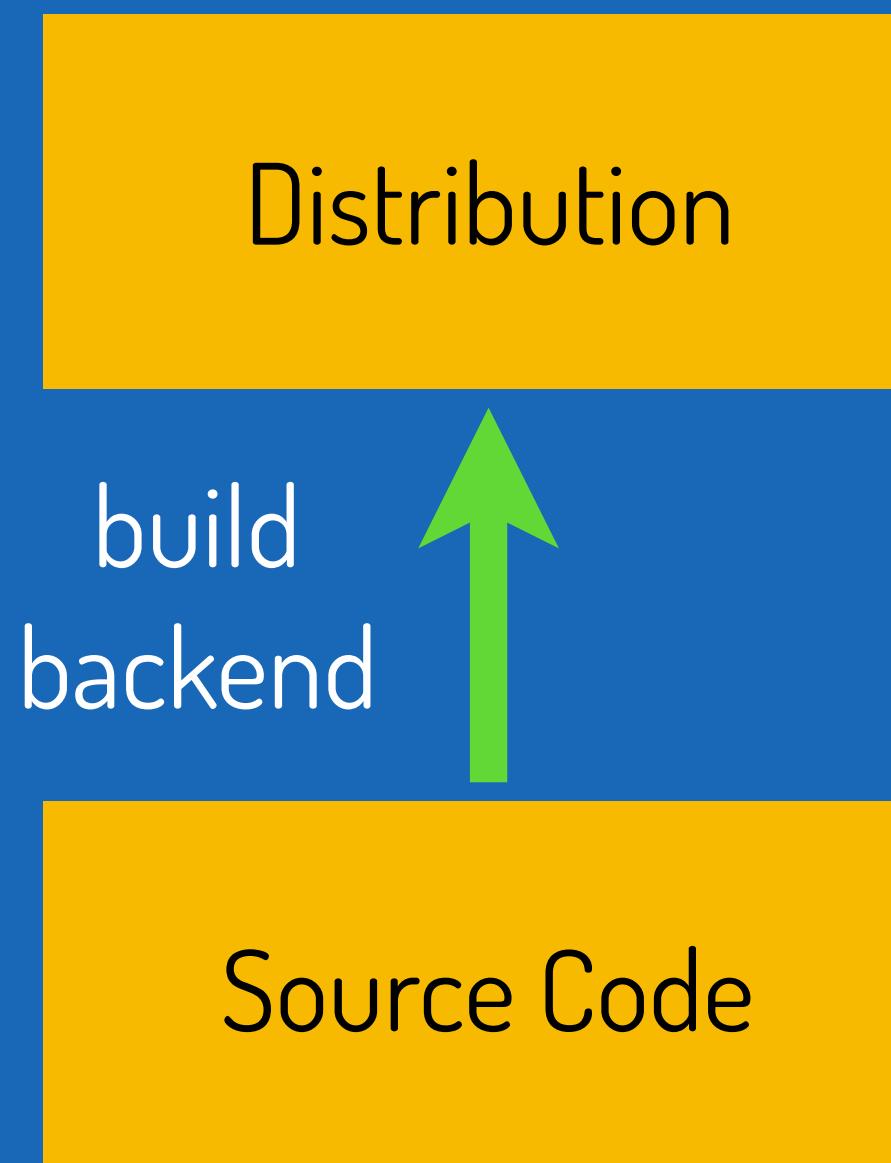
Package Distribution Metadata



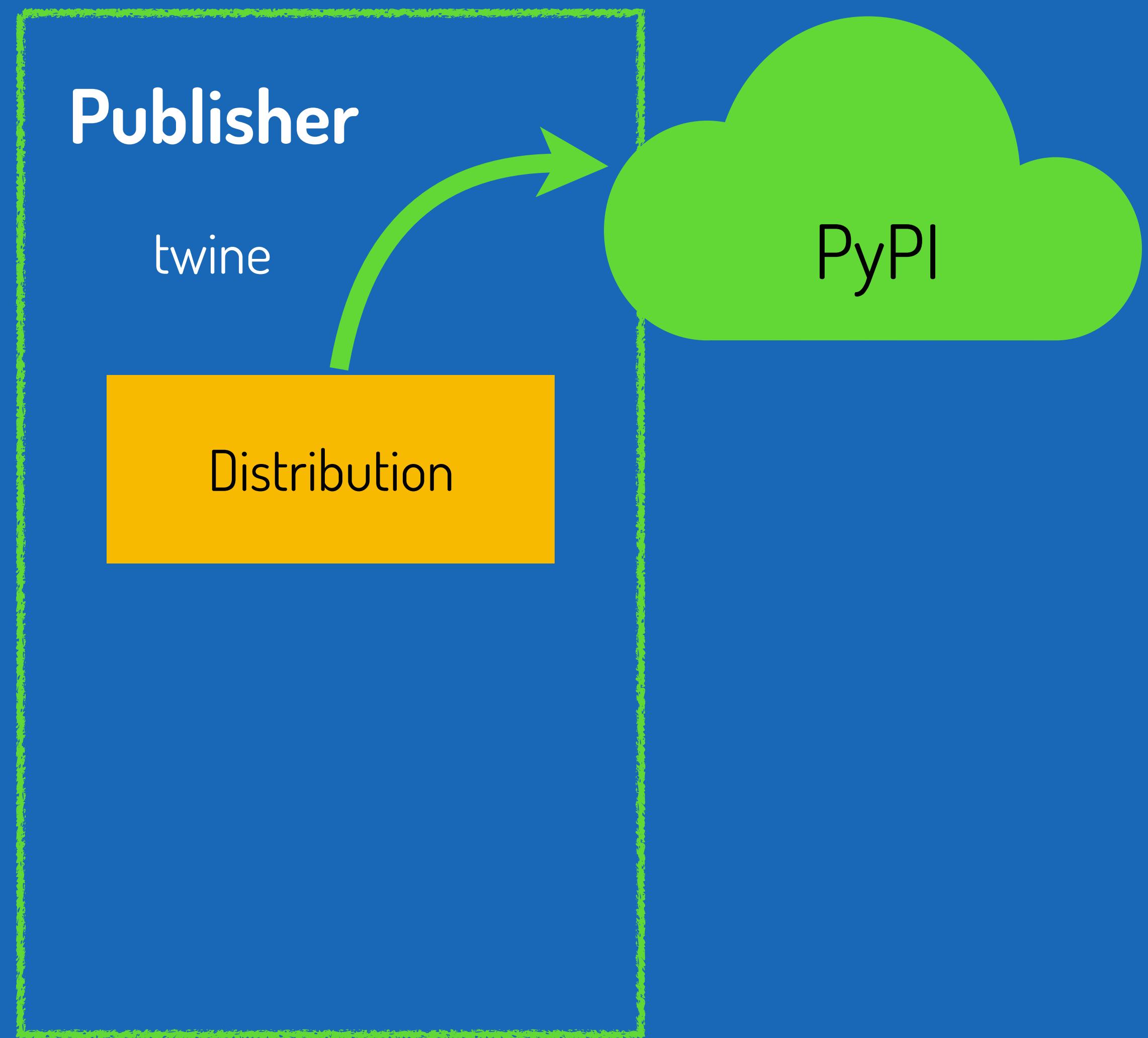
WE WENT TOO FAR ONCE.

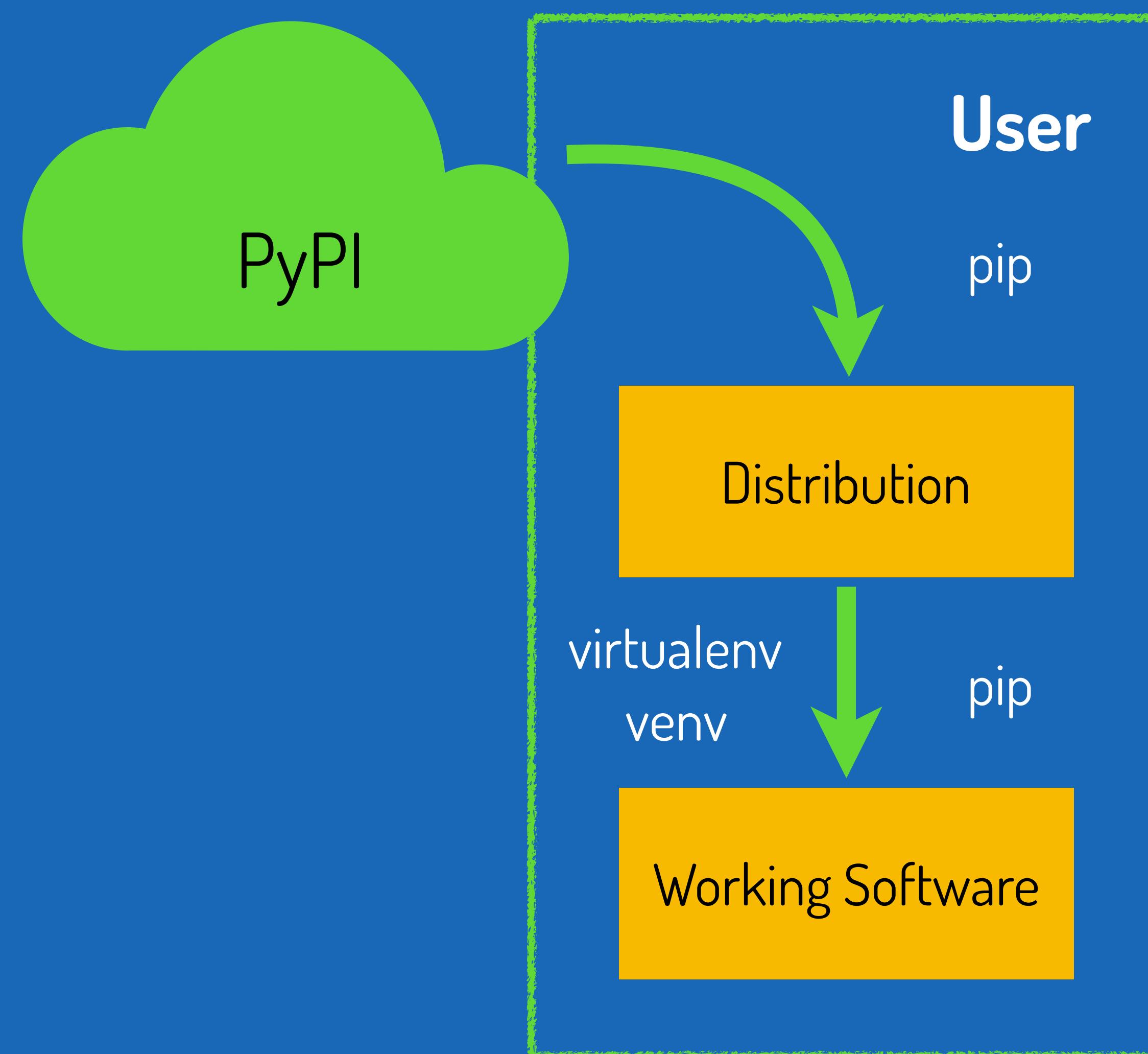
- PEP 426 - Metadata 2.0
 - Was trying to solve all the problems with Python's metadata. Never became a reality.
- Incremental Improvements
 - Our current, softer approach – workable approach with volunteers.

Publisher

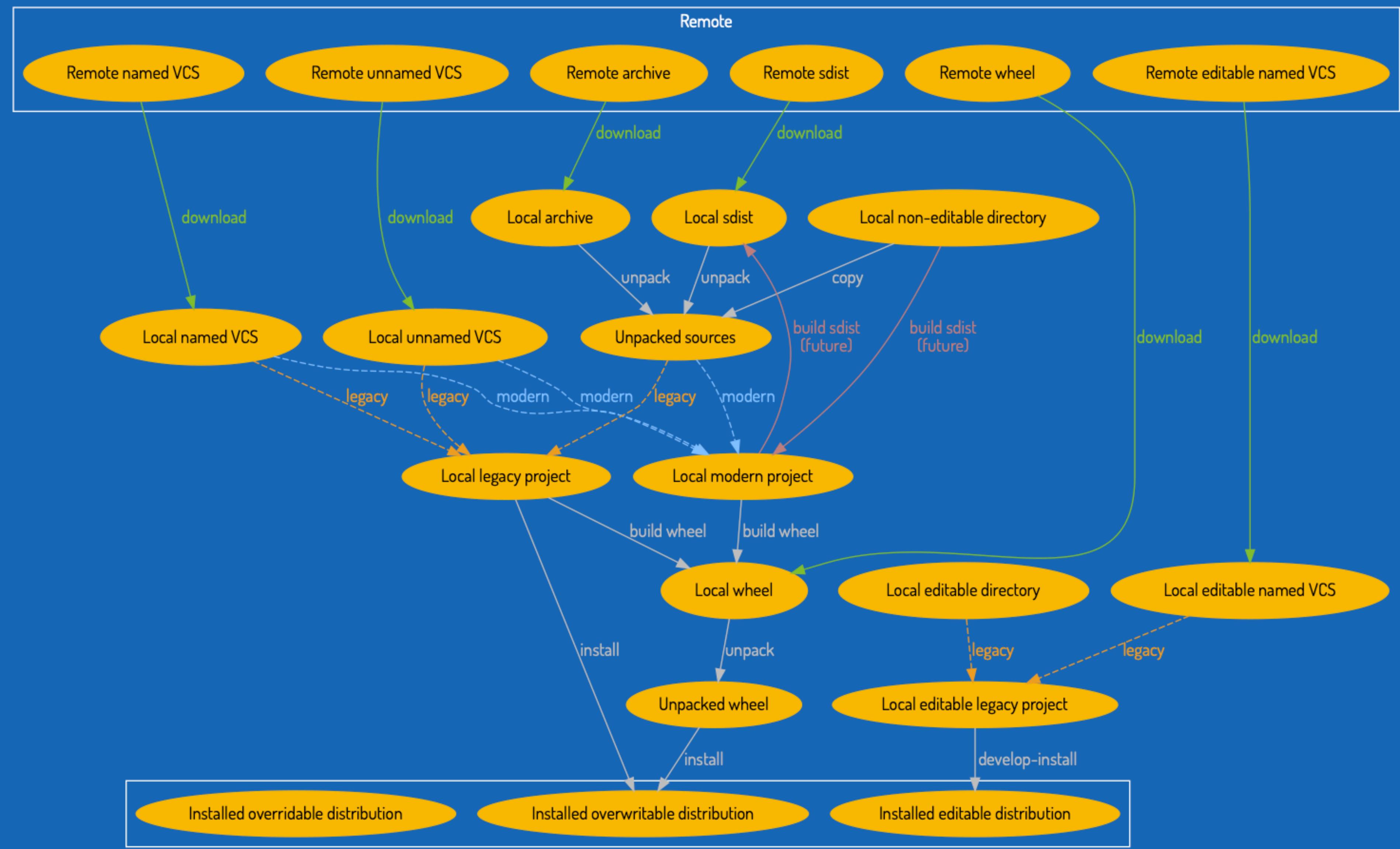


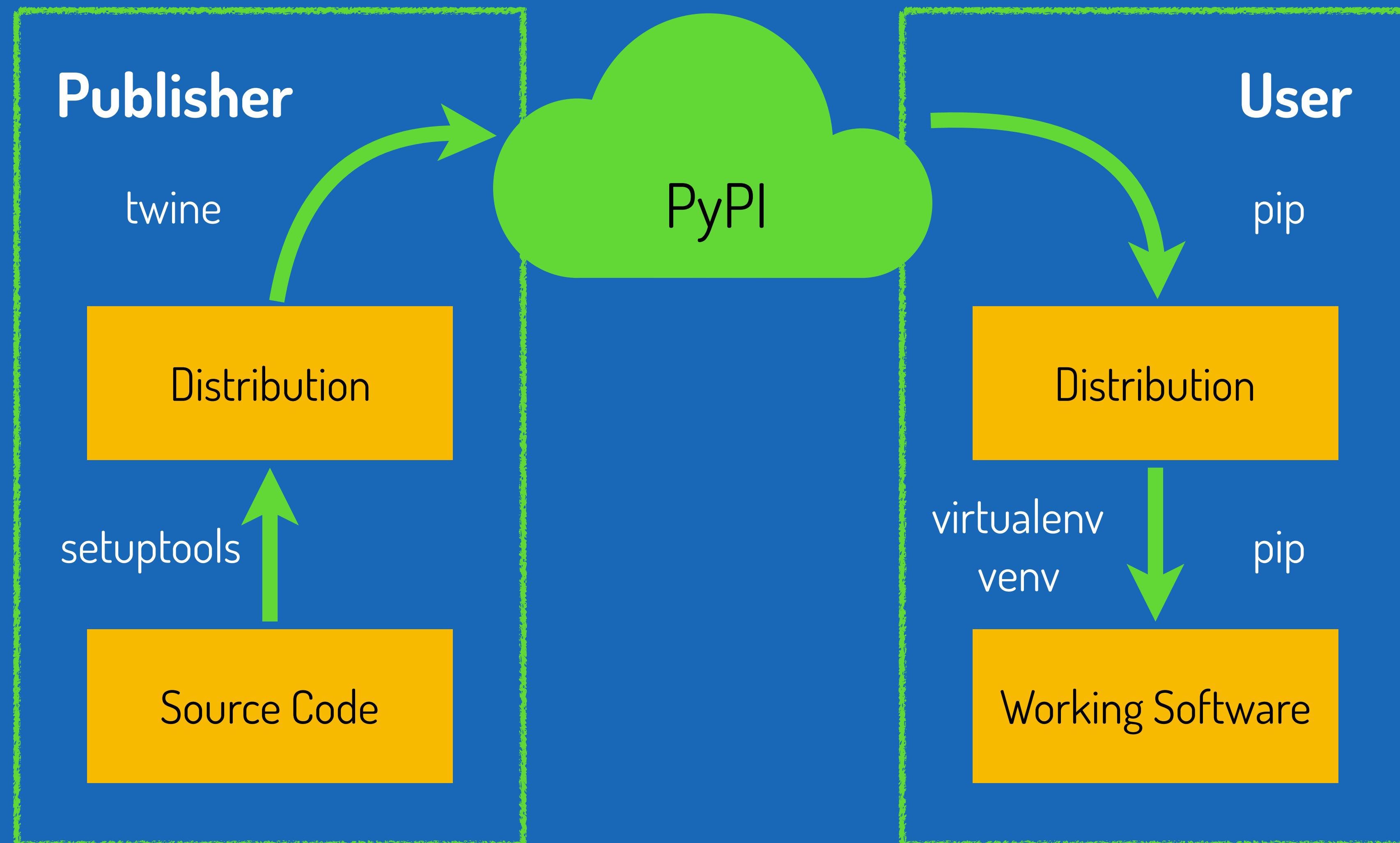
- Usually setuptools
- PEP 517 and PEP 518
 - "modern" source distributions
 - `pyproject.toml`

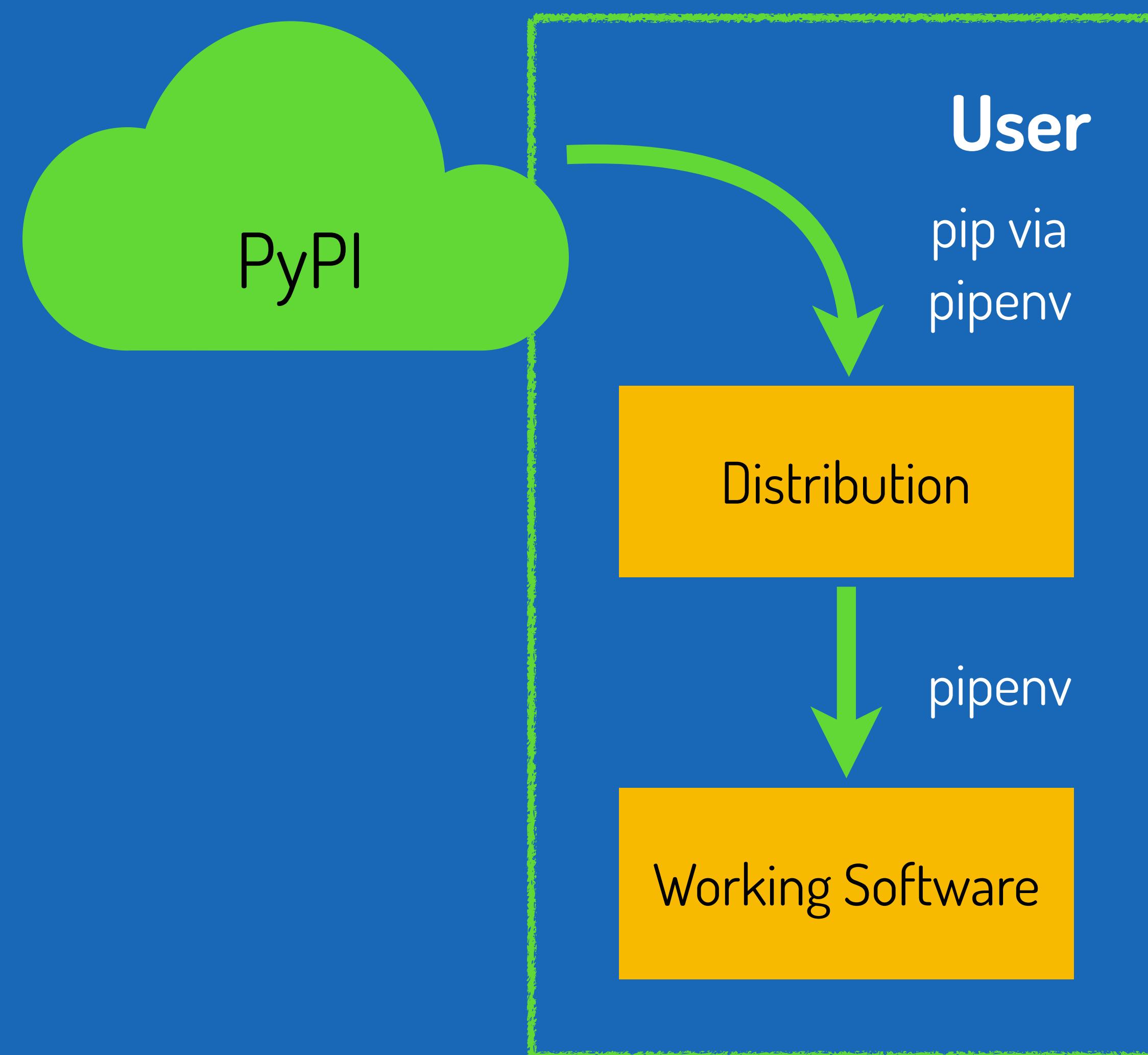




It's not that straightforward.







NON-PYPA TOOLING

Tools built by companies or different volunteers.

Environments are difficult.

Publisher

Distribution

flit

Source Code

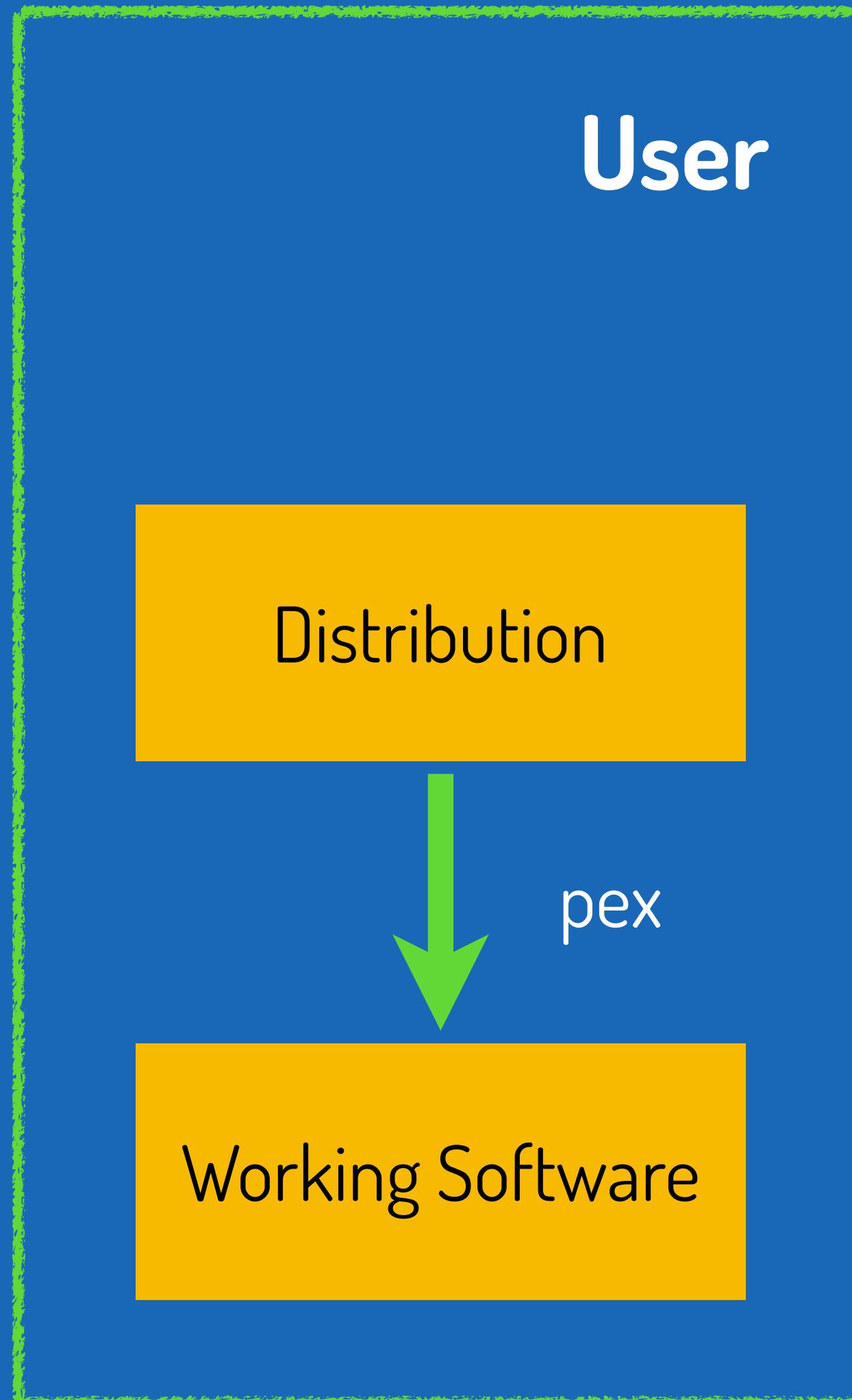
- Simplifies the "build" experience
- Metadata in `pyproject.toml`

User

Distribution

pex

Working Software



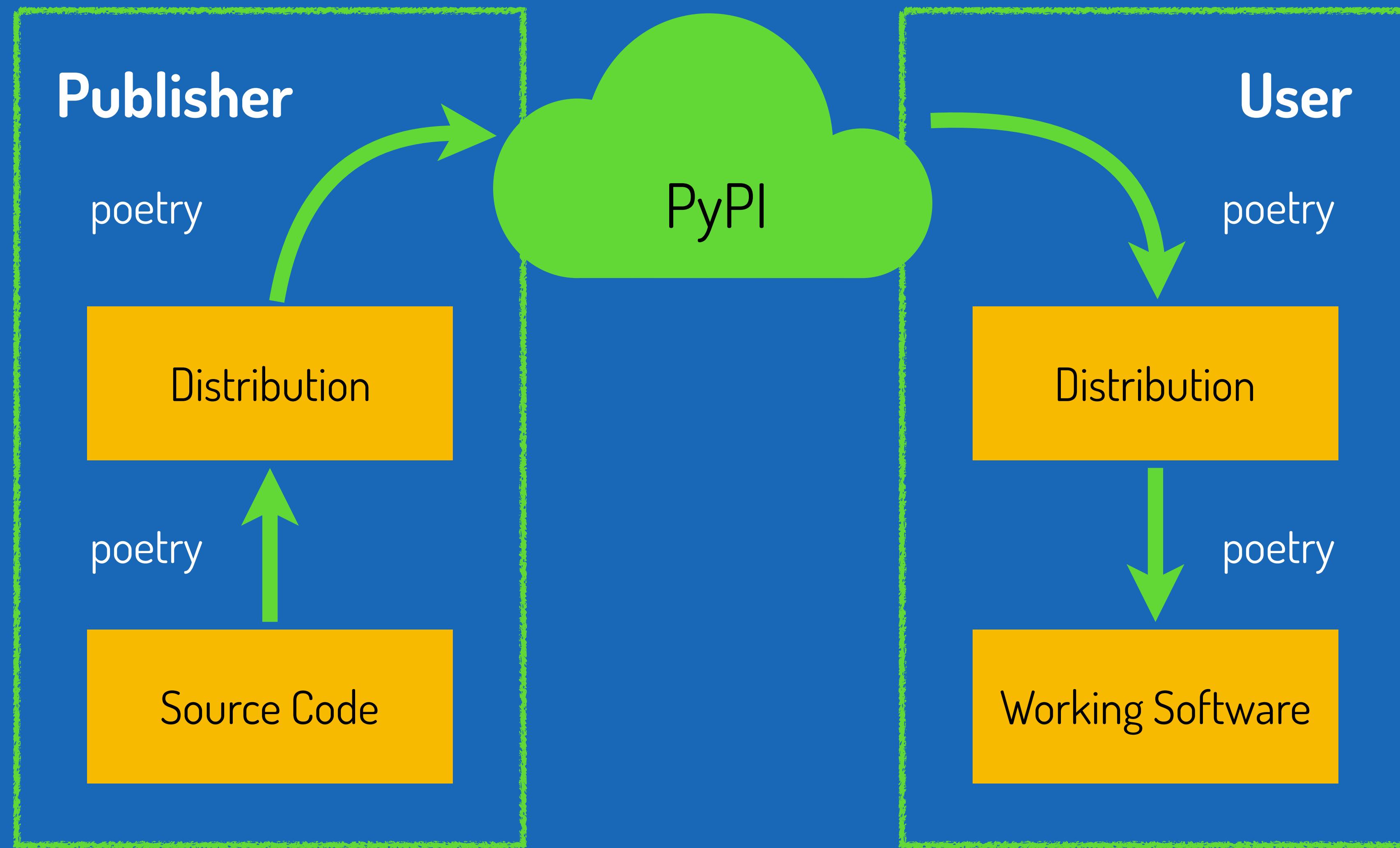
User

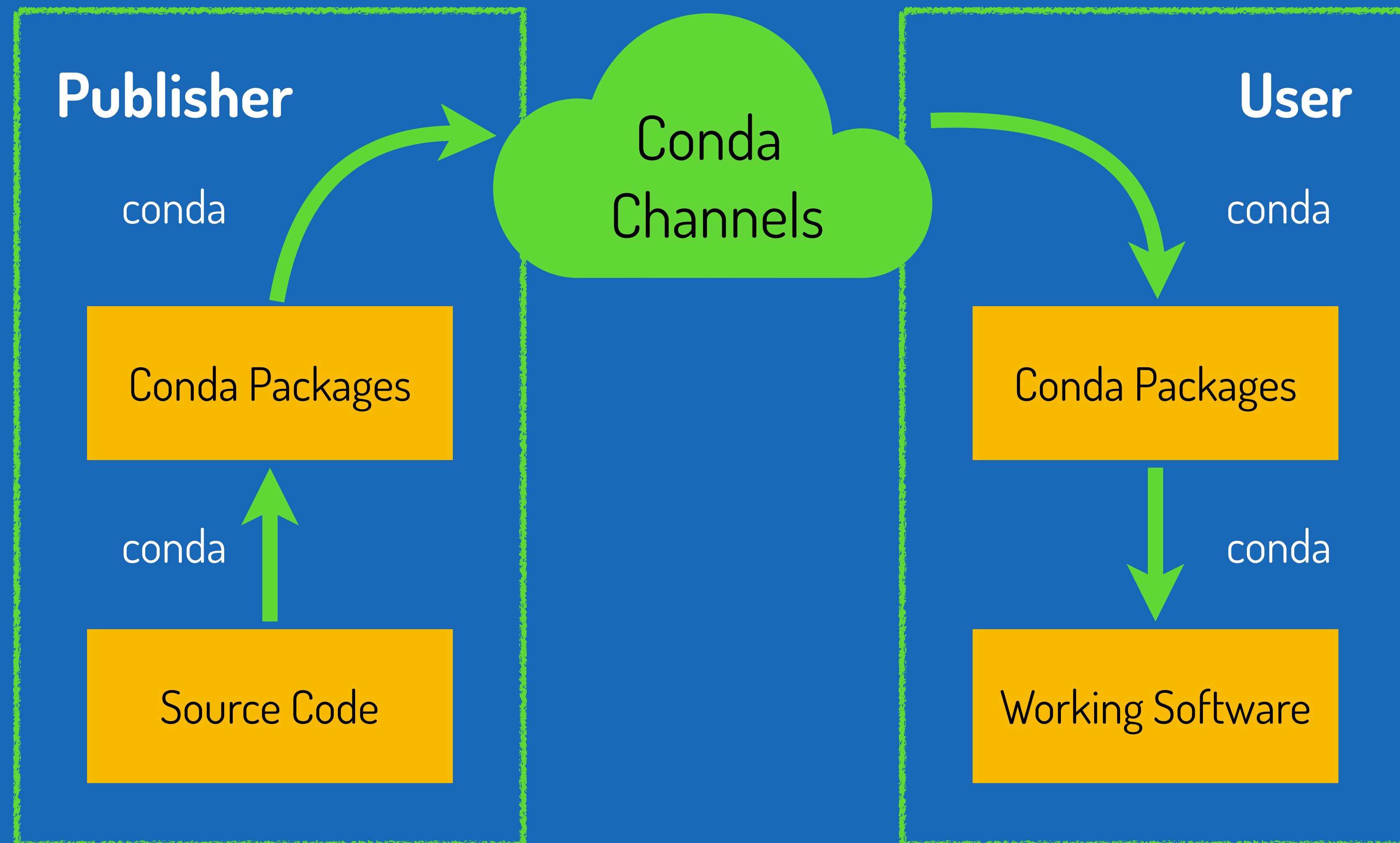
Distribution

pipx

Working Software







"... all of this stuff is really hard and
there's a lot of knowledge scattered around in
people's heads that could really do with being
captured somewhere"

— PAUL MOORE

WHERE WE'RE HEADED

Future Improvements in Python Packaging

I DON'T HAVE
A TIME MACHINE.

CORE FUNCTIONALITY IMPROVEMENTS

ongoing

- securing PyPI – The Update Framework (TUF)
- newer manylinux standards – manylinux2014
- yanking packages from PyPI
- better licensing metadata

CORE FUNCTIONALITY IMPROVEMENTS

soon?

- structured lockfiles in pip
- package preview functionality from PyPI
- security notifications for vulnerable packages
- pip dependency resolver

BETTER USER EXPERIENCE

- better defaults in pip
- feature flags on PyPI
- interoperability testing
- professional UX review

REDUCING TECHNICAL DEBT

- virtualenv rewrite
- making the transition from distutils -> setuptools
- pip build logic refactor
- pip "working schemes" cleanup
 - change the right environments

MOAR STANDARDISATION

- editable installations
- better, more powerful, extras
- less ambiguous licensing
- interoperability with other package managers

HOW DO WE GET THERE?

- Volunteer!
- Tell us how you deal with "interesting" issues.
- Fund one of the targeted funded projects.

Thank you!
\o/