The Blue Yonder Python Habitat

Karlsruhe Python Meetup December 2019

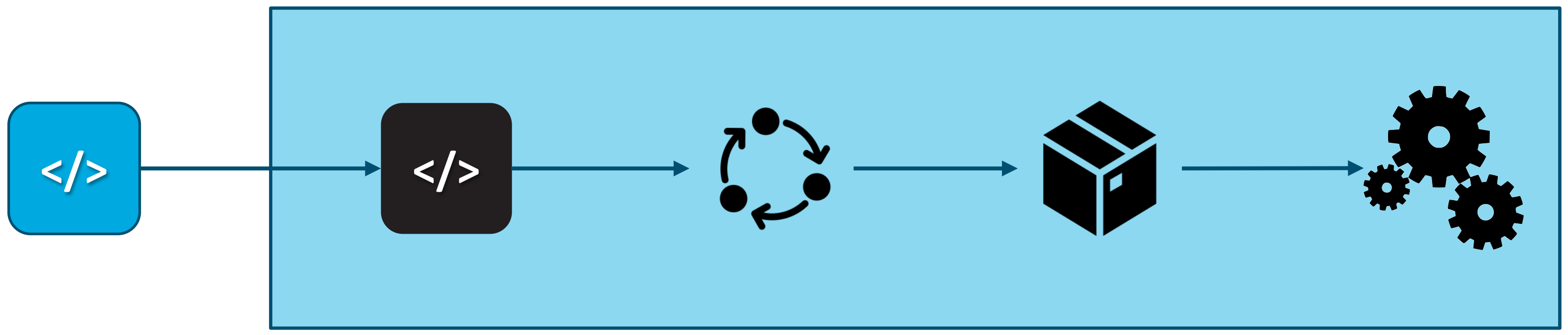
Bjoern Meier

jda.
Plan to deliver™

Do you think this is a good idea?

```
632 # ssh app-user@prod-server
500 app-user@prod-server: /app # git pull
…
501 app-user@prod-server: /app # . venv/bin/activate
(venv) 502 app-user@prod-server: /tmp # pip install -r requirements.txt
Collecting flask
  Downloading https://files.pythonhosted.org/packages/9b/93/628509b8d5dc749656a9641f4caf13540e2cdec85276964ff8f43bbb1d3b/Flask-1.1.1-py2.py3-none-any.whl (94kB)
    |████████████████████████████████| 102kB 2.1MB/s
Collecting requests
  Downloading https://files.pythonhosted.org/packages/51/bd/23c926cd341ea6b7dd0b2a00aba99ae0f828be89d72b2190f27c11d4b7fb/requests-2.22.0-py2.py3-none-any.whl (57kB)
    |████████████████████████████████| 61kB 1.7MB/s
…
```

Building and Running Python Applications in a Commercial Environment



As a company you should solve how to

- Use external software, in this case Open Source Software (OSS)
  - Open Source Compliance
  - Decoupling from external dependencies
  - Handle Risks
- CI/CD
- Distribute internal and external software
- Run applications

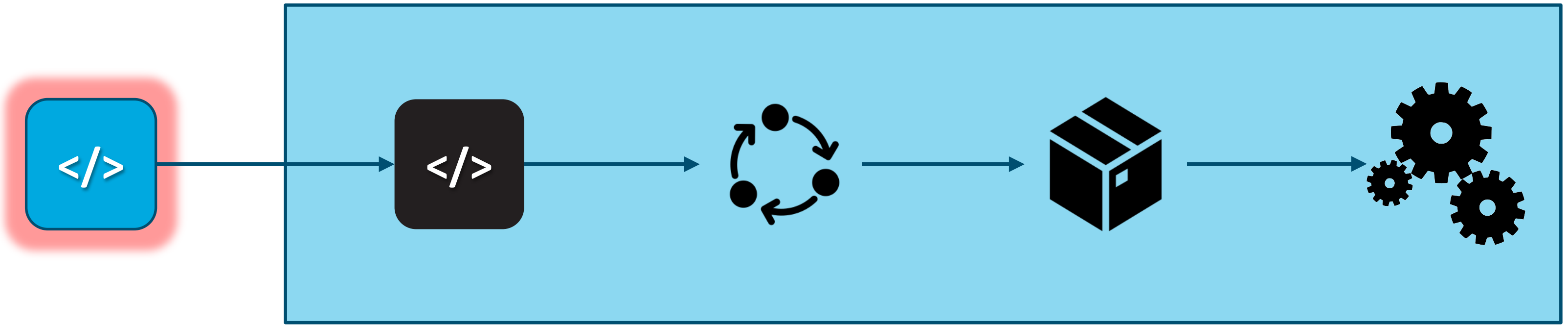
Open Source Software

Accelerator & Risk

Open Source Software as Accelerator

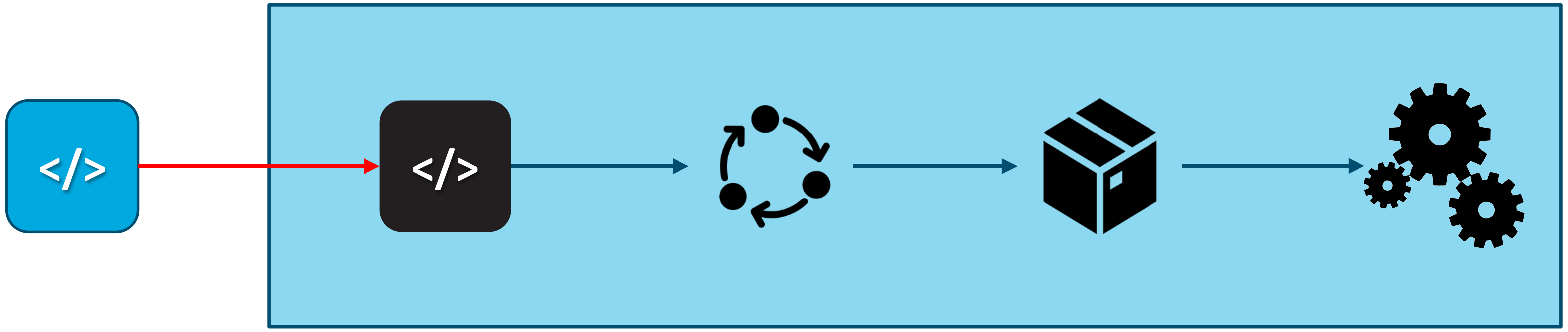
- Accelerator:
  - Libraries/Frameworks for nearly every problem
    - Analytic libraries: NumPy, SciPy, Pandas, Dask, …
    - Web applications: Flask, Django, aiohttp, Sanic, Tornado, …
    - Distributed computing: Dask.distributed, PySpark, TensorFlow, …
    - …
  - Leverage external know-how
  - Standards

Open Source Software as Risk



- OSS Licenses
  - Disclosure of code using/extending OSS code can be required
  - Monetary purposes might be excluded
- Ownership
  - License change (for newer versions)
  - Blocker for changes
  - Dominance of a company in an open source project
- Open reported vulnerabilities

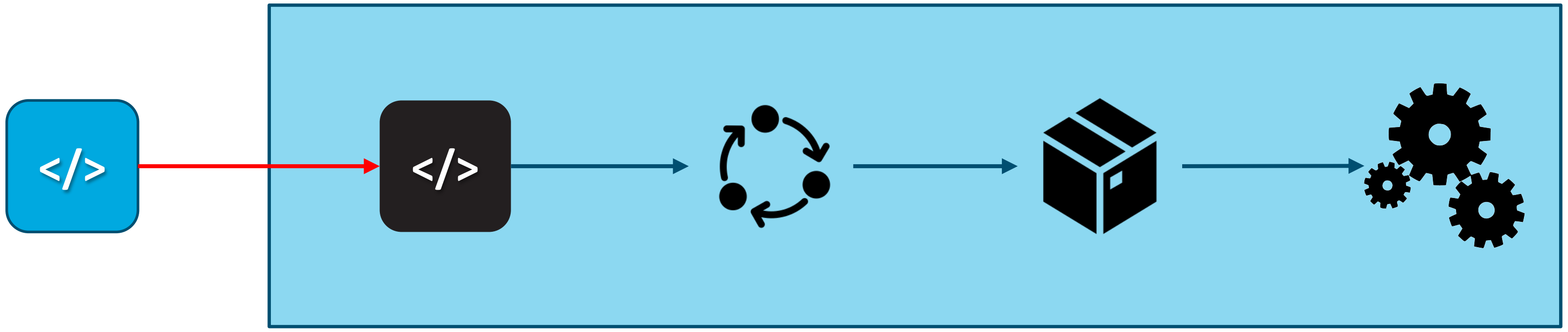
Public Package Repositories as Risk



- Package ownership
- Malicious exploitation (fake packages, e.g. python3-dateutil*)
- SLO/SLA
  - Availability
  - Scalability
  - Support

* https://github.com/dateutil/dateutil/issues/984

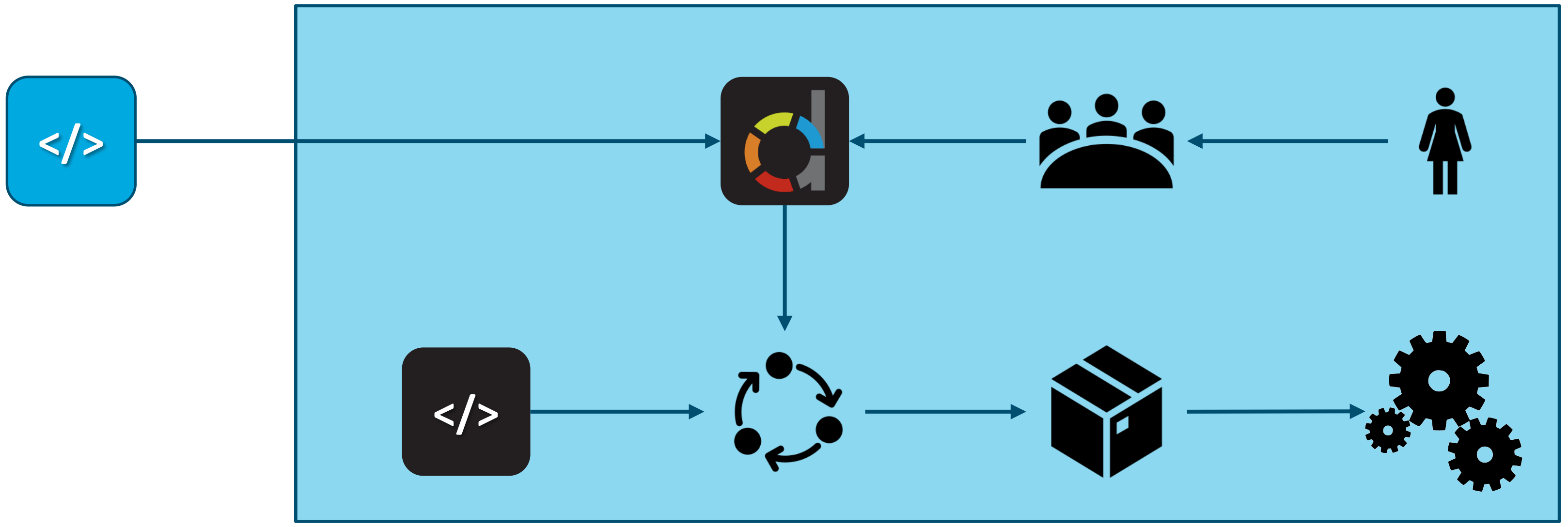
Control the Package Repository



Private Package Repository (e.g. Devpi, Anaconda, bandersnatch, Artifactory, …)

- Whitelisting / blacklisting
- Open Source Review Board controls available packages
  - License
  - Up-to-date
  - Majority
  - …
- Control of service availability and scaling

Infrastructure with Devpi Package Mirror



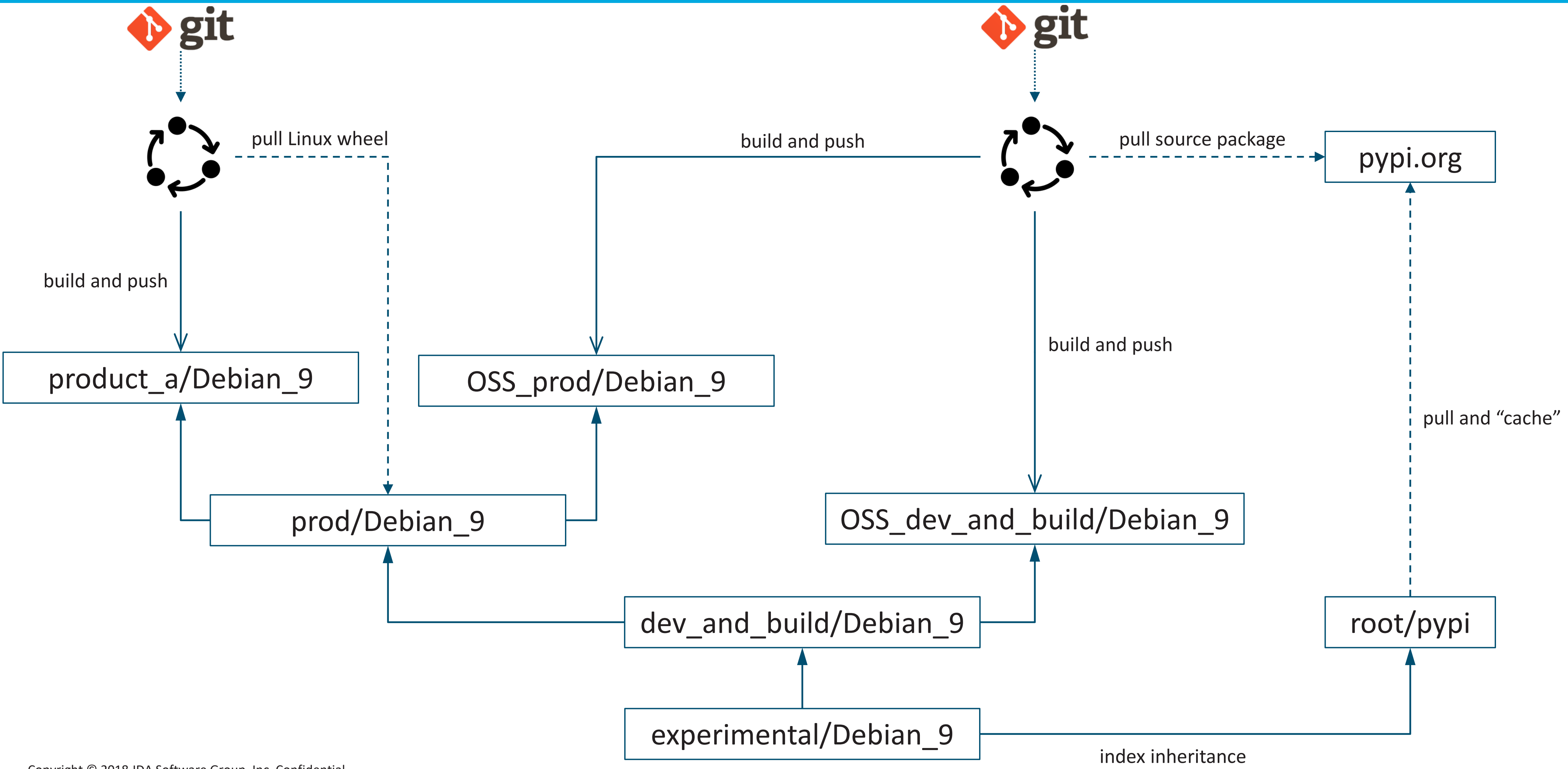
Internal Package Distribution

Internal Package Distribution: Requirements

- Upload/Download packages
- Distribute OSS and internal packages from the same source
- Package compatibility with runtime environment
- Single point of truth
- Scalability

→ We solved it with Devpi as well.

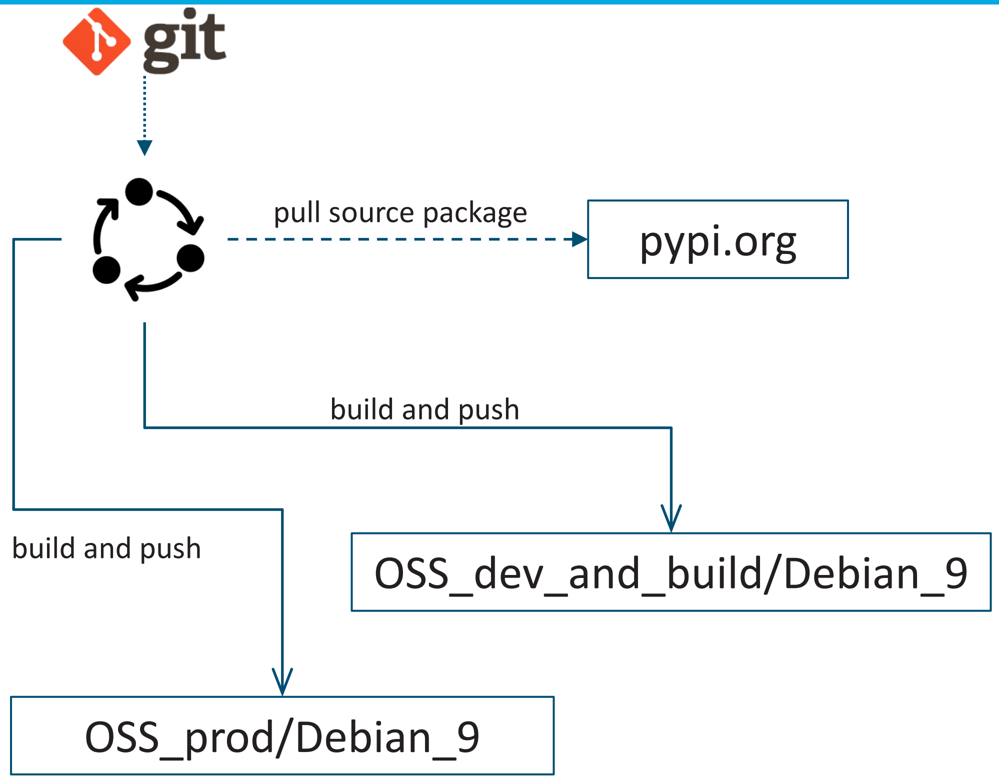
Index Inheritance Tree and Package build



Why Source Builds?

- Ensure OS library dependencies are available and compatible for the used Linux distribution
- Many Linux wheels can ship with linked third-party libraries which can be outdated and vulnerable

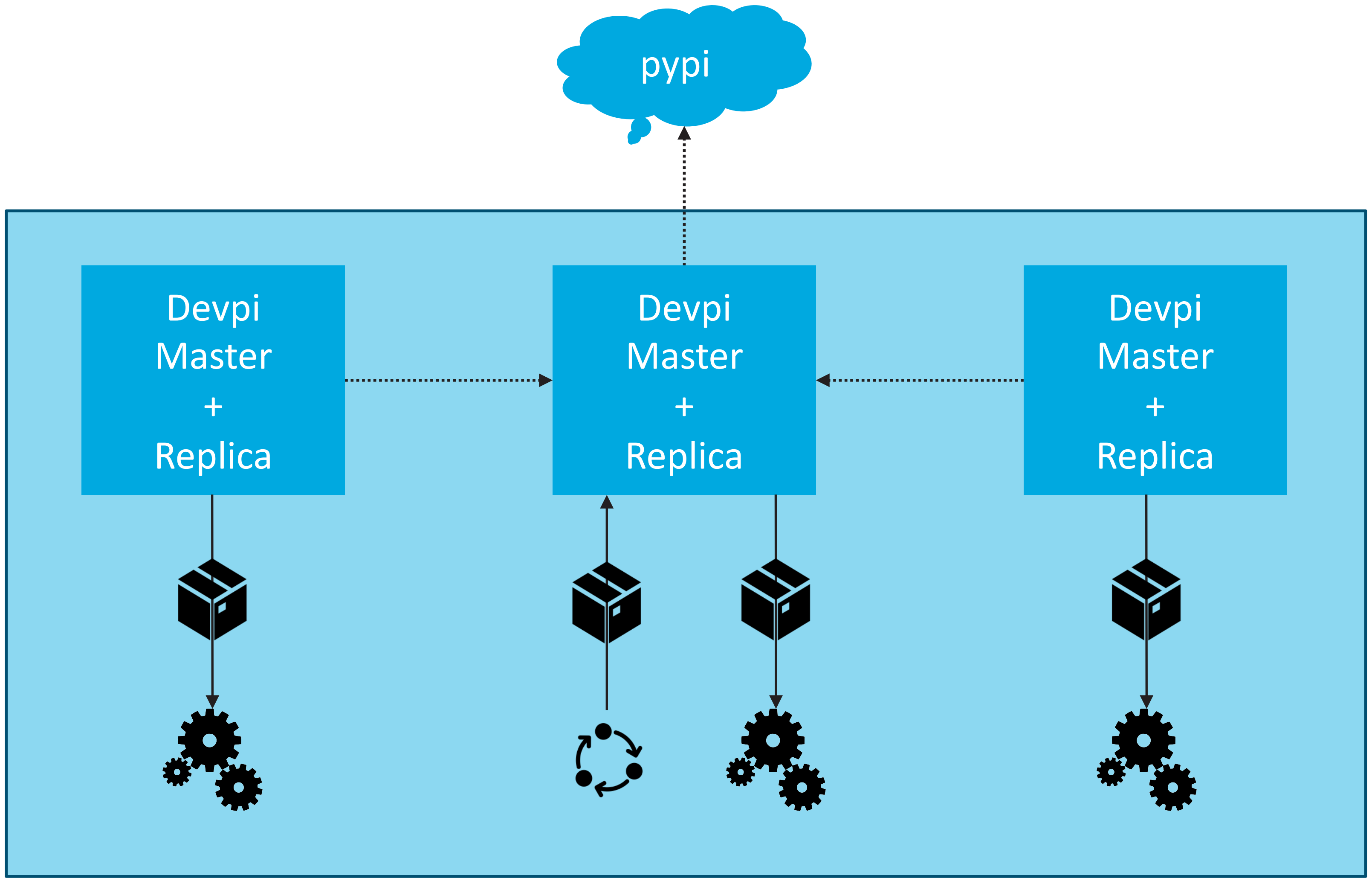
https://www.python.org/dev/peps/pep-0513/



Internal Package Distribution: Requirements

- Upload/Download packages
- Distribute OSS and internal packages from the same source
- Packages compatible with execution infrastructure
- Single point of truth
- Scalability

Scale Devpi


pypi
Devpi Master + Replica
Devpi Master + Replica
Devpi Master + Replica

Runtime Environment

Python Arrived in the Cloud

- Azure App Service update: Free Linux Tier, Python and Java support, and more (7 Mai, 2019)
  - Python (3.7, 3.6, 2.7) support Linux is now generally available.
- Announcing the general availability of Python support in Azure Functions (August 19, 2019)
- You can now develop your [AWS] Lambda function code using Python. (October 08, 2015)
  - Python 3.7 (November 19, 2018)
  - Python 3.8 (November 18, 2019)
- AWS Elastic Beanstalk (?)

https://azure.microsoft.com/de-de/blog/azure-app-service-update-free-linux-tier-python-and-java-support-and-more/
https://azure.microsoft.com/en-us/blog/announcing-the-general-availability-of-python-support-in-azure-functions/
https://docs.aws.amazon.com/lambda/latest/dg/lambda-releases.html

Requirements for “Cloud" Python Application Layers

Microsoft Azure Web Service & Function

- application.py
- requirements.txt

! pypi.org will be used → bypass introduced measurements

AWS Lambda

- get dependencies:
  pip install --target ./package –r requirements.txt
- Function code and dependencies bundled as zip file

AWS Elastic Beanstalk

- application.py
- requirements.txt

Add pip.conf to use internal package repository.

There is more to it than just running it

- Python is easy to execute, and you only need to have
  - the type of service you want to execute,
  - the application code (application.py) and
  - the dependencies (requirements.txt).
- With time or company/project size the importance of the supporting infrastructure will grow significantly:
  - Observability
  - Scalability
  - Cost control
  - …

Thin User Application Layer

		WSGI Service	Dask.Distributed	Generic Python	…
🙂	User Application Layer	• wsgi callable • wsgi app code • requirements.txt • (configuration)	• dask code • requirements.txt • (configuration)	• command • app code • requirements.txt • (configuration)	
😎	Provided Execution Layer	• logging • metrics • alerting • revisions • …	• logging • metrics • alerting • revisions • …	• logging • metrics • alerting • revisions • …	
😎	Provided Infrastructure Layer	• OS management • networking • resources • …	• OS management • networking • resources • …	• OS management • networking • resources • …	

Thin User Application Layer

- Thin application layers
  - expedite adoption and
  - keep off complexity for the user.
- Execution layers for different services can be kept uniform
- Out of the box and uniform infrastructure services
  - logging
  - metrics
  - ...
- Flexibility is traded for simplicity and uniformity

Summary

As a company you need to have:

- Open Source Compliance
- Decoupling from external dependencies (as much as possible)

As an organization you can benefit from:

- Thin User Application Layer
- Observability features provided by default

Give Back

- OSS should not only go into your company
- Give back
  - OSS
  - Time
  - Money

https://github.com/JDASoftwareGroup
https://github.com/blue-yonder