Python vs. R

A Data Engineering perspective

imagination at work

Outline

Introduction

Technological aspects

- Connectivity/Portability
- Traceability and reproducibility
- Tooling/ ready to go solutions
- Results/valorization/decisions making

Organizational aspects

- Projects
- Skills management

Our ecosystem

Basics

We deliver a software solution in hospital network (radiology), used for local reporting/decision making

Technological

Java EE software on top of a SQL database. Two main components :

- a data collection gateway
- a data integration/processing/display engine.

Operational

- Machine(s) set up on customer infrastructure, with admin privilege (remote access).
- The solution is connected to different actors in the network using DICOM/HL7…
- Personal Health Information (PHI) are ***confidential*** and must remain on site
- (very) bad network gateway

Our organization

Actors

- Product team : gives priority among possible new features
- Adv Dev Team : pre-develops/tests new technologies or dangerous features
- Dev Team : develops the features and commits to new releases
- V&V Team : makes sure it works and answers the needs
- Data Engineering/Integration Team : brings medical data expertise, scientific guidance, and analyzes data for current features and new ones.

Constraints

- Technological control : no isolated/orthogonal choices
- Rapid delivery : compatibility or functional coverage between Adv Dev/Dev/Integration
- Integrated solutions : make use of our tools (Atlassian/Git/etc)
- People background : project oriented computing schools, graduate scientists with not much computing skills, business people

Technological aspects

Data Connectivity

We are in heterogenous env, and need to make profit of all possible datasources

DICOM

Dataset : Python (pyDICOM), R (Rigorous/DICOM)

Network : Python (binding C++), R (None?)

Databases

Support level : Python (always official drivers), R (maintained by interested people)

Compliance : Python (PEP-249/specs), R (DBI/lib)

Files

Larger than RAM datasets : Python (Blaze/Dask), R (?tricks?)

over network

Webcrawling/Rest : Python, R

As a Service : Python (no comment), R (plumber?)

Portability/3rd party language

Some working envs are constrained, no right to install a new technical stack.

We are on Windows Server envs (may change).

Java VM : Python (Jython, 2.7old), R (Renjin, lot of bugs?)

.NET : Python (IronPython, 2.7new), R (deprecated)

Python : PyPy (no kidding), rpy2 (very active)

Sometimes others languages do the job better (not reinventing the wheel)

C/C++ : Python (GIL/Cython/brute force), R (Rcpp)

Traceability/Reproducibility

We are medical device, under ISO 13485.

Official repos : Python (PyPI>100000), R (CRAN>10 000)

Versioning possible : Python, R

Dependencies Handling : Python (embedded), R (embedded)

Binary libs : Python (wheel), R (Win/MacOS)

Binary builds : Python (ugly), R (no idea)

Backward compatibilities : Python (2 vs. 3/virtualenv), R (no?)

Testing : Python (UnitTest/Nose/pytest/etc…), R (stubthat/testthat)

Modularity : Python (embedded), R (modulesInR/import)

Ready to go solutions/results rendering

Setting up default env : Python (Anaconda, etc…), R (Anaconda)

Interactive shell : Python (Ipython/Jupyter), R (Rstudio/Rkernel/Jupyter)

Advanced package manager : Python (conda, Canopy), R (conda, ?)

Graphical output : Python (matplotlib), R (ggplot2)

Interactive results analysis : Python (Jupyter), R (Shiny)

Interactive results demonstration : Python (Jupyter), R (Shiny)

Third party dedicated display: Python (all you want), R (limited choice)

Organizational aspects

Projects handling

We are working with a dev lifecycle, with unequally experienced people

Multiple contributors: Python (best practices, seniority diffs), R (modularity)

Cloud stuff: Python (everywhere), R/R (recent appearance)

Continuous integration: Python (e. g. Atlassian), R (?)

Deployment: Python (heroku, venv copy, etc…), R (?)

We are not a charity mission, we sell our solutions or have trade secrets in it.

Intellectual Property : Python (PSF, friendly for critical libs), R ((L)GPL everywhere)

Skills management

We address various topics

Image processing: Python (SciPy/scikit image), R (magick)

Computer Vision : Python (OpenCV/mahotas), R (?)

Machine Learning: Python (scikit learn/theano/etc…), R (stuffs)

Data processing parallel : Python (celery/std lib), R (?/parallel)

We have people from other fields that need to demonstrate their point in autonomy

Quick onboarding : Python (stupid at beginning), R (stiff curve)

Trainings (phys/MOOC) : Python (local skills/cheap), R (more expensive/cheap)

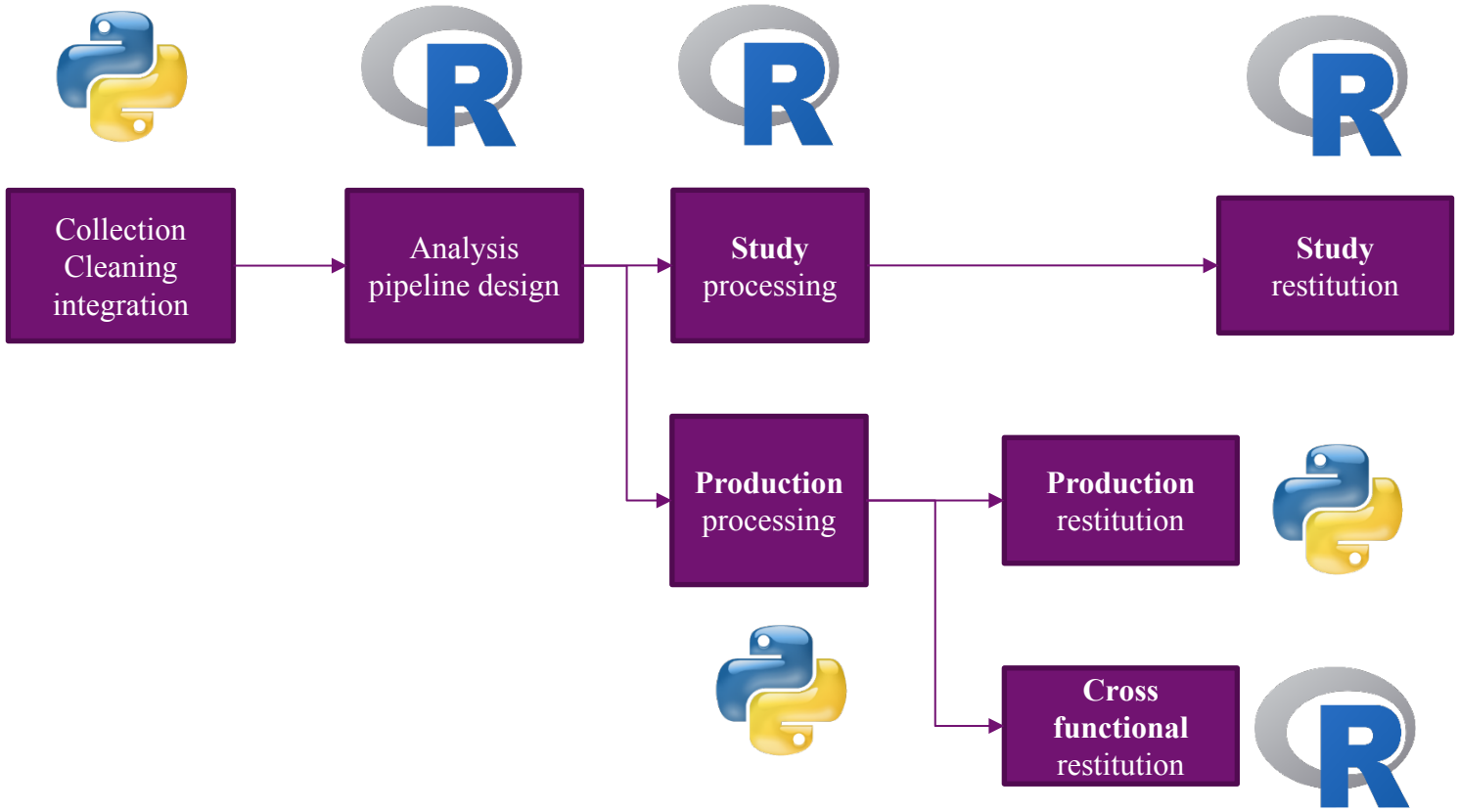
We are growing and we have turn over

Sourcing width : Python (physical sciences), R (statisticians/biologists)

Industrial porosity: Python (common lexicon), R (more business)

Summary

Along a data analysis pipe




imagination at work