MOPV036

Porting Control System Software from Python 2 to 3

A. F. Joubert, M.T. Ockards, S. Wai, SARAO, Cape Town, South Africa

ICALEPCS 2021

MeerKAT radio telescope

- Developed by South Africa as a precursor to the Square Kilometre Array
- Array of 64 interlinked antennas, fully operational since 2018
- 48 antennas concentrated in the core area, approximately 1 km in diameter
- 8 km maximum baseline (longest distance between any two antennas)
- 13.5 m diameter main reflector, and a 3.8 m diameter sub-reflector per antenna
- 2.2 Tbit/s combined data from all antennas
- 170 km of fibre connect to the central processing building in the Karoo, with the maximum length being 12 km.
- 120k sensors managed by the control and monitoring system

www.sarao.ac.za

The South African Radio Astronomy Observatory (SARAO) is a National Facility managed by the National Research Foundation and incorporates all national radio astronomy telescopes and programmes. SARAO is responsible for implementing the Square Kilometre Array (SKA) in South Africa.

science & innovation
Department:
Science and Innovation
REPUBLIC OF SOUTH AFRICA

NRF
National Research Foundation

SARAO
South African Radio Astronomy Observatory

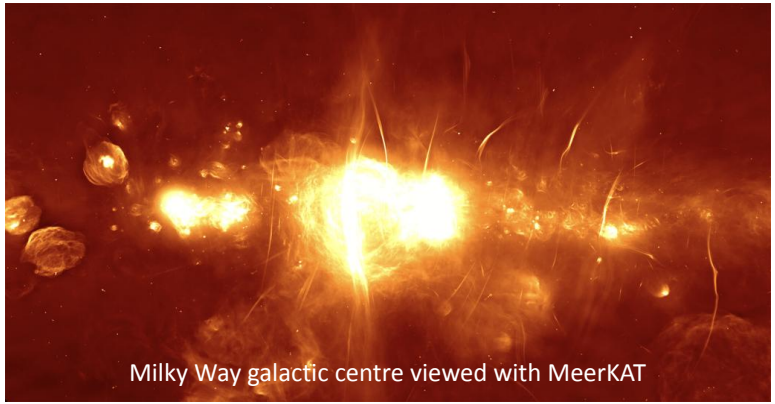
Our issue

MeerKAT control and monitoring system software needs to be ported, 20+ packages

Python 2 end-of-life on 1 January 2020

Possible strategies

- **Separate branches for Python 2 and Python 3**
  - Expensive to maintain
- **Single source Python 2, autogenerate Python 3**
  - Need “fixers” for 2to3 conversion
  - Python 3 code isn’t seen by developers
  - Need to verify autogenerated code
- **Single source Python 3, auto-generate Python 2**
  - Need to convert all to Python 3 first
  - Need “fixers” for 3to2 conversion
  - Need to verify autogenerated code
- **Single source Python 2 and 3 compatible (wrapper)**
  - Code will have some special cases
  - No autogeneration, so developers see the code that is used

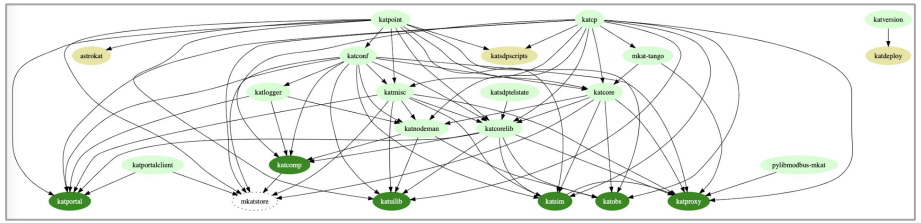

Milky Way galactic centre viewed with MeerKAT

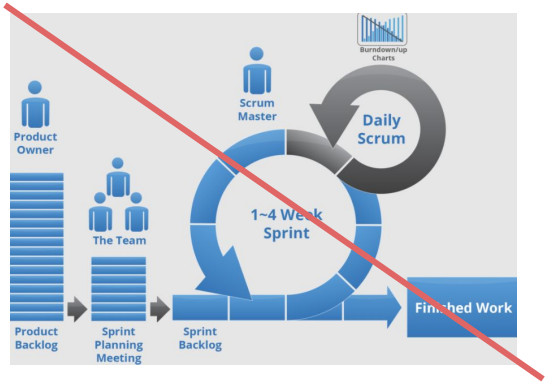
Selected

Wrapper:
python-future

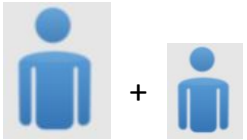
Approach

Work top-down according to dependency tree





Tried scrum, but too much specialised knowledge required, and estimation impossible



Switched to single developer, focussed on porting, plus one reviewer

Evaluate
- Review the repository to provide context for the developer

Clean up
- Delete unused code and scripts to simplify the porting effort

Update CI
- Add python3 tests to Continuous Integration
- Allow failure

Futurize (python2)
- Futurize stage 1 (modernize python2 code only, no compatibility with python3)
- Auto code format (chance to clean-up, with Black code formatter)
- Futurize stage 2: use the python-future wrapper to add (broken) python3 support

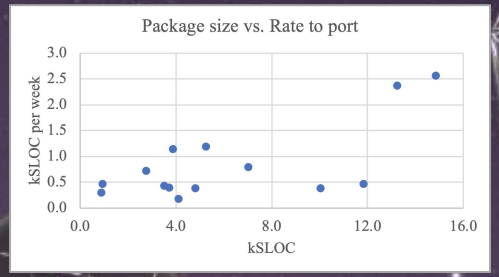
Futurize (python3)
- Update dependencies, fix conversion issues
- Get tests passing under python3

Update CI
- Fail if python3 tests fail
- Publish python3 package

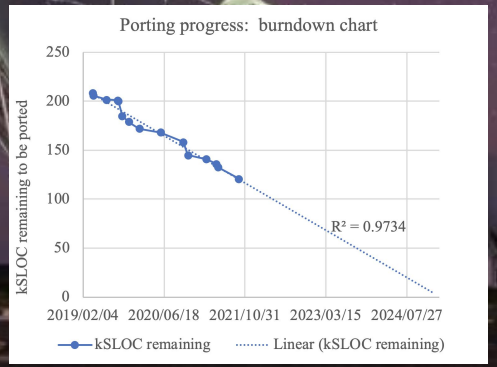
Verify
- Downstream packages still pass python2 tests
- System integration tests still pass

Results

Estimation is very difficult.
No correlation with lines of code in a package



Progress has been made, but still require a significant effort.



Conclusion

Porting a codebase is a significant undertaking, and needs to be carefully managed. Prioritisation and dedicated resources are important.
Success is not guaranteed.