13.2 ANYWHERE NEAR THE SEVEN-YEAR ITCH? ON THE STATE OF WRADLIB UPON THE RELEASE OF V1.0

K. MÜHLBAUER[1] M. HEISTERMANN[2]
[1] University of Bonn, Germany
[2] University of Potsdam, Germany
heisterm@uni-potsdam.de

The free and open source software communities tend to release early and often. Initial versions are numbered like 0.x in order to convey that the software is a work in progress. Version 1.0 is used as a major milestone, indicating that the software is "complete", that it has all major features, and is considered reliable enough for general release.

Seven years ago, wradlib (http://wradlib.org) was born, and soon released as v0.1: an open source Python library for weather radar data processing, analysis and visualisation. Since then, wradlib has grown and matured, and, as of today, serves a large user community from all around the world, rooted in academia, government agencies, and the private sector. wradlib supports the essential steps of weather radar data processing which include: reading a range of weather radar data formats; geo-referencing observations from spherical coordinates to spatial reference systems; detection and correction of errors (e.g. beam blockage, attenuation, clutter); phase processing; quantitative precipitation estimation; quality control; integration with other sensors (gauges, space-borne platforms); compositing; matching retrievals to hydrological application scales; and, last but not least, rich visualisation options along different dimensions. The overall development paradigm was and still is a flat data model, allowing for minimal entry thresholds and maximum flexibility in building applications.

In 2018, we will finally release wradlib v1.0, and thus commit to the maturity, stability and reliability of the services it delivers. Here, we review and discuss the past seven years in terms of key developments, lessons learned, but also with regard to some promises yet to be kept. Some of these developments and issues can be considered as representative for many scientific software packages in the past decade: e.g. how GitHub and its integrated services (management of issues, pull requests, and Continuous Integration) have boosted the standards of collaborative scientific software development; how self-sustained tutorials in combination with jupyter notebooks have revolutionized package documentation; how deployment via conda-forge has finally eliminated the vast majority of issues with dependencies as complex but vital as gdal. And finally (and somewhat interesting for researchers) how the main development workload has shifted dramatically from rapid prototyping of scientific algorithms to maintenance and support of usability and reliability, which is why we can gladly release v1.0.