The Short Description

The brief description which will appear in the online program and give attendees a basic sense of your talk. This should be around 100 words or less.

The Long Description

Your placement in the program will be based on reviews of your detailed description. This should be a roughly 500 word detailed outline of your presentation. This outline should concisely describe software of interest to the SciPy community, tools or techniques for more effective computing, or how scientific Python was applied to solve a research problem. A traditional background/motivation, methods, results, and conclusion structure is encouraged but not required. Links to project websites, source code repositories, figures, full papers, and evidence of public speaking ability are encouraged.

Title: Building Communities Around Open-Source Geoscience Projects

Short Description

Scientific communities can use Open Source projects to create standardized, tested, and free tools for their use, but such projects are only as good as the community around them. Here we explore the evolution of MetPy from a repository of graduate school code in 2008 to a National Science Foundation funded project with a strong community of users and growing community of contributors. Essential factors in this community-building include scoping, providing training materials, creating transition paths from the use of legacy tools, and creating a welcoming environment for new contributors. These principles of community building are broadly applicable across any scientific software project and are crucial to adoption and sustainability.

Long Description

The geosciences have a long history of creating and sharing Open Source software tools, but have often had difficulty in creating *sustainable* software with a large community of both users and contributors. MetPy is a package for meteorological calculations, plotting, and file parsing that is envisioned as a set of standard meteorological Python tools with first class documentation and citations. With humble beginnings in 2008 as a repository of code from several students' graduate work, MetPy has become a National Science Foundation funded project with two developers devoting significant amounts of time to the code and associated training materials. Here we will examine steps taken by the developers to build a community around MetPy, and to encourage contribution from community members. The ideas shared are applicable to any Open Source scientific software project.

To begin gaining traction in the meteorological community and replacing the use of legacy tools (specifically GEMPAK in the field of meteorology), MetPy needed users who were capable and excited about using and sharing MetPy. While there was some independent discovery and

adoption of MetPy, most new users have been introduced to the package through a series of workshops the developers were able to conduct at universities to promote the use of Python and MetPy in the classroom and in research. The workshops have been conducted in person, but the training materials are available online at any time and are often referenced by new users as a repository of practical information and examples. Another key factor in increasing adoption was creating a gallery of examples for new users to build upon. For example, users wishing to make a map with radar or satellite data plotted on it need only download the pertinent gallery example and make any modifications they desire, greatly lowering the activation energy required. A more recent addition to the collection of learning resources is the “MetPy Mondays” series, consisting of short (generally less than ten minute) screencasts and blog posts shared with the community every Monday. These posts often are inspired by user support questions and have become very popular within the community.

A community-driven project is only truly driven by the community if it has external contributors in addition to the primary developers. MetPy has worked to foster community contributions by encouraging new users to file issues or submit pull requests during workshops and support interactions, as well as by providing training on how to contribute. MetPy also strives to create a comfortable environment for new contributors, with a code of conduct, simple contribution guidelines, and prompt, friendly code reviews from the developers. We also strive to create a consistent coding style with automated code style checking tools and a set of guidelines for scientific programmers: “The Zen of Scientific Software Maintenance.” MetPy developers curate lists of good “first issues” with most of the fix already done in an issue comment, giving new contributors an easy starting point for their early contributions.

In conclusion, by putting equal effort into software development and community development, MetPy has been able to gain adoption in the atmospheric sciences and create a community of users and developers. These techniques and trials are broadly relevant to any developer or contributor to the open-source scientific software infrastructure.

MetPy: https://github.com/unidata/metpy/
MetPy Gallery: https://unidata.github.io/MetPy/latest/examples/index.html
Unidata Developer’s Blog: https://www.unidata.ucar.edu/blogs/developer/
Unidata YouTube Channel: https://www.youtube.com/user/unidatanews
Zen and the Art of Scientific Software Maintenance:
https://jrleeman.github.io/ScientificSoftwareMaintenance/
John’s SciPy 2016 Talk: https://www.youtube.com/watch?v=ZDVxB2kgOao