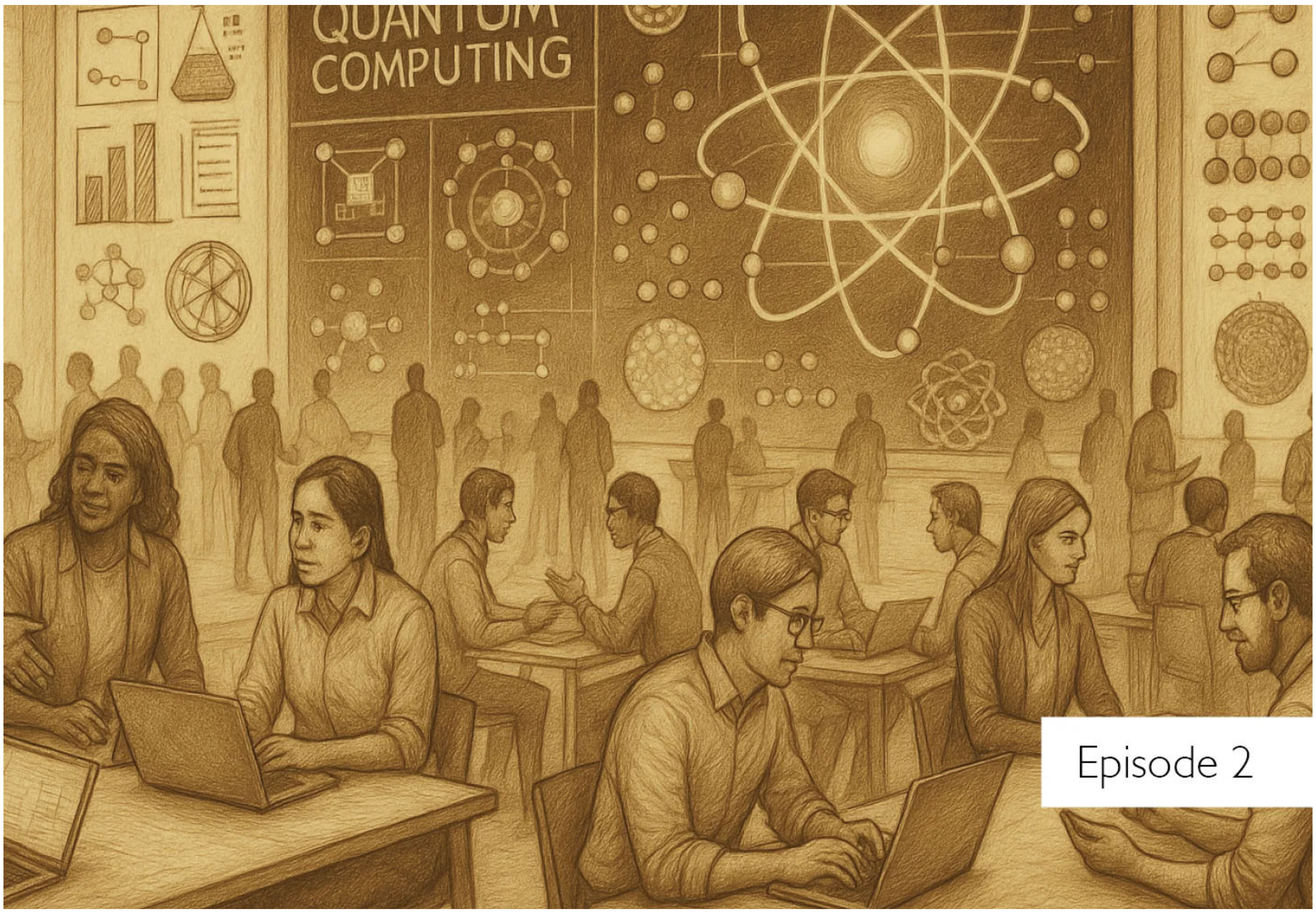
Quantum Computing: Open Source Communities Tackle Unique Challenges, Part 2

2025 08 14 - By Andrew Oram



The first article in this series explained the special challenges faced in quantum computing to teams and companies building open source communities. For that article, I interviewed leaders for the classiq library and OQTOPUS. In this concluding article, I interview one more project and offer some general thoughts.

QuTiP

QuTiP (which stands for “Quantum Toolkit in Python”) offers a story of a project at an earlier stage of organization—even though it has been around since 2012 and serves tens of thousands of users. I interviewed developers Alex Pitchford, Simon Cross, and Neill Lambert for this article.

Essentially, QuTiP seems to have solved two tasks of open source communities: recruiting developers and building a strong user community. They lack a reliable long-term source of funding, though, and a formal structure for bringing users into the discussion. Their relatively ad-hoc organizational structure can be seen in the many references to particular individual contributors in my history of the project.

They are therefore making a challenging transition from a research project to one that is ready for industrial use. They have achieved the technical transition, but not the organizational one they need.

Origins and Growth of QuTiP

QuTiP is a simulation library, claiming to support “a wide variety of Hamiltonians.” (Hamiltonians measure the energy in a physical system.) It began when two postdocs in Japan, Paul Nation and Robert Johansson in a group led by Franco Nori, discovered that the MATLAB library they had depended on was unsupported and lacked maintenance. They joined RIKEN (already mentioned in the OQTOPUS section of this article), which had the same problem and decided to reproduce the functions in Python. (The old MATLAB toolkit is now archived and explicitly recommends QuTiP as its successor.)

The team made their project open source in order to broaden the potential user base, a strategy that has paid off handsomely.

The scope of the project was augmented when Pitchford joined in 2015, bringing some algorithms about quantum control that had a similar origin: his Python rewrite of a MATLAB tool.

Recently, two other developers (Alberto Mercurio and Yi-Te Huang) have reproduced the Python interface in the Julia language, sometimes cloning the Python functions and sometimes creating new capabilities. The addition of Julia to the project makes sense because it has many powerful math-related features. However, it is very different from Python and requires new forms of coordination.

Recruitment Strategy

At the beginning, creating a team was a haphazard process depending a good deal on personal connections. Pitchford and Cross provided a couple examples of this sort of outreach that built their team of developers:

- Pitchford joined the project because his PhD supervisor had worked for a while at RIKEN and had become friends with Johansson. When the original founders finished their postdocs and got jobs, they no longer had time to maintain the project, so Pitchford’s supervisor persuaded him to join the admin team even though he wasn’t familiar with most of the code. (An admin does the work of accepting pull requests and maintaining the code, like a maintainer or committer.)
- Pitchford and Nathan Shammah—another key QuTiP developer—later went to EuroSciPy 2019 in Bilbao to deliver a talk, and discovered that Cross, whom they hadn’t known before, was also speaking on QuTiP. They recruited Cross to join the development team.

Google Summer of Code (GSoC) is an important source of developers now. Students on the verge of graduating desire highly to be chosen for GSoC, which gives them an opportunity to advertise their skills as well as earn a stipend. Many choose to write a contribution to QuTiP, and the QuTiP admins can choose participants who have done a good job.

Some of the GSoC contributors go on to join the admin team. For instance, after working on the QuTiP QIP package, Boxi Li joined the admin team and is now the lead developer for the package.

A Robust Maintenance Strategy

From early on, the people working on QuTiP have demonstrated a professional approach to high-quality, maintainable code. For instance, after Pitchford's initial contribution submission in 2015, Nori invited him to RIKEN for a 6-week internship to learn more about their open-source projects and QuTiP's coding standards, during which time he revised his code to enable it to be merged.

When the QuTiP admins find a developer they're interested in recruiting, either at GSoC or elsewhere, they ask the applicant to make a small contribution to the project. They can judge from this contribution whether the applicant is comfortable with Python and GitHub, can work with their code and make a coherent pull request, etc.

About five years ago, a developer named Jake Lishman, supervised by Eric Giguère and Pitchford, completely rewrote the underlying layer of the library, touching on almost every file. It was not ready yet for production use, but a set of admins worked on the code further and released a fresh version, QuTiP 5, on March 28, 2024. Pitchford and Cross pointed out that as QuTiP spreads beyond the original research community, being employed in industry now along with universities, the quality provided by QuTiP 5 is crucial.

Funding, Governance, and Continuity

Admins on the QuTiP project spend a lot of time mentoring developers and bringing their code up to the necessary quality. Obtaining funding is a burden that they are trying to hand off to a board, but it hasn't found a stable source yet.

Nori has operated as QuTiP project director since its inception and is largely responsible for the financial support of QuTiP throughout its history, through research grants. The Japan Society for the Promotion of Science (JSPS) is a source of short-term postdocs for QuTiP developers, and the Moonshot program at the Japan Science and Technology Agency (JST) has given a lot of support over the past five to six years.

Lambert, another QuTiP developer and RIKEN research scientist, also helped support QuTiP through JST's PRESTO program for several years, and he is described by Pitchford and Cross as the "glue" who keeps the project together. Giguère is employed by the Université de Sherbrooke in Québec and can work on QuTiP fairly intensively.

But most developers are committed to other jobs and fit their QuTiP work into their free time. Luckily, QuTiP is a good source for research papers, or provides the tools needed by researchers to produce the results covered in their papers. QuTiP turns out to be a "good launchpad for careers," as shown by the story of the original developers Johansson and Nation, but it is always time to move on.

Currently, there is no real user involvement in governing QuTiP. The board consists of former developers such as Johansson, and managers of developers. Continuation of the project rests on overburdened admins. Their task currently is to expand the scope of the board and turn questions of funding over to it.

The Elements of Successful Open Source

The three projects in this article demonstrate common ways to maintain open source. All the projects have made the crucial leap beyond the founding organizations to build loyal communities of developers. Each, in its own way, actively and consciously recruits developers. Their governance and funding differ a great deal.

In the case of classiq, a single company maintains quality control and manages the project, and funds it through their “closed core” offering.

The stability of the OQTOPUS project, along with funding, comes from partnerships between a research institution and companies who benefit from the open source project. Its appeal as a source of research papers also draws developers from academia.

The QuTiP project is also a draw for researchers. The project soldiers on through code contributed as part of research work or from sympathetic developers with other jobs. A few developers can afford to put in bursts of activity to pull all the contributions together. Besides using the library, most developers can justify their contributions by publishing research based on it.

The stories told in this article indicate that even in a demanding field with a field of contributors that is inherently limited, open source can thrive. A conscious attention to recruitment, user engagement, continuity, and quality control is required.

About Andrew Oram:

Andy is a writer and editor in the computer field. His editorial projects at O'Reilly Media ranged from a legal guide covering intellectual property to a graphic novel about teenage hackers. Andy also writes often on health IT, on policy issues related to the Internet, and on trends affecting technical innovation and its effects on society. Print publications where his work has appeared include The Economist, Communications of the ACM, Copyright World, the Journal of Information Technology & Politics, Vanguardia Dossier, and Internet Law and Business. Conferences where he has presented talks include O'Reilly's Open Source Convention, FISL (Brazil), FOSDEM (Brussels), DebConf, and LibrePlanet. Andy participates in the Association for Computing Machinery's policy organization, USTPC.