What stalls amateur radio development?

An open letter exposing the current state of the amateur radio world, and roadblocks on the way to making it technologically up-to-date,

by Wojciech Kaczmarski, SP5WWP
w.kaczmarski@teletra.pl

Warsaw, June 2024

Background

I've been a licensed amateur radio operator since 2016. In 2019, I started the M17 Project[1], a highly successful and widely acclaimed international endeavor. A few years later, in 2021, I was honored with the ARRL Technical Innovation Award for my contribution to the technical development of our hobby[2]. M17 has become a large, overarching ecosystem, offering both software and hardware, open-source solutions[3]. Through this time, I have worked with many people from all around the world. Most of them were exceptionally brilliant, but since M17 is based on volunteers' work, our team members rotated constantly.

The current status of Amateur Radio

It's not a secret that most of the amateur radio community depends on large companies (Icom, Yaesu, Kenwood) and solutions they provide. The *status quo* is all about keeping hardware and software proprietary, minimizing users' chances to modify it. While there are new models of radios being advertised all the time, they do not offer anything *new*. This keeps the amateur radio world in a state of artificially sustained technological stagnation, short-sightedly throttling down the progress in order to maximize someone's profit. It's been already shown that community-driven projects can lead to technological advancements - the advent of MMDVM[4], M17, OpenRTX[5], WPSD[6], OpenWebRX[7], and many others. The purpose of this open letter is to show that cutting-edge, open-source solutions can only be successfully proliferated by a group of skilled amateur radio developers.

Stock, commercial firmware mostly lacks functionalities amateur radio operators seek. The reason behind that is simple - large corporations often don't have a clue what amateur radio operators want. A solution to this seems very simple - let skilled users write their own replacement firmware, then release it under an appropriate open license, for the rest to use. Sounds easy, but as always, there's a catch. The firmware flashing process is not always easy (binary files are almost always obfuscated) and requires a fair amount of reverse engineering effort.

---

[1] M17 Project's homepage - https://m17project.org
[2] https://www.arrl.org/news/view/the-2021-arrl-technical-innovation-award-honors-wojciech-kaczmarski-sp5wwp
[3] M17 Project's repositories - https://github.com/M17-Project
[4] Multimode Digital Voice Modem - https://mmdvm.com
[5] OpenRTX project's homepage - https://openrtx.org
[6] W0CHP Pi-Star Dashboard - https://w0chp.radio/wpsd/
[7] Homepage - https://www.openwebrx.de/

The most popular digital voice mode is DMR[8] (looking at the users count and radio infrastructure size). It had to be adapted for ham radio use cases, as it was originally designed for professional use. DMR is based on an open standard released by ETSI. The standard by itself does not mention what voice coder should be used, but the *de facto* protocol seen “in the wild”, enforced by manufacturers, uses AMBE.

Linking radio access networks (RAN) using IPv6 and geostationary satellites and utilizing remote radio units (RRU) with powerful I/Q modems is without a doubt a technological advancement. Wouldn’t it be amazing to see Daniel Estevez’s (EA4GPZ) 32APSK modem[9] “move out” from the shacks of a few and be used in real hardware and by hams worldwide? Or at least in the QO-100’s footprint? This is just the tip of an iceberg of ideas to implement in the real world. We can’t just passively wait and expect major manufacturers to pick up on it. It is something we can do on our own.

The pitfalls of volunteer-based work

Volunteering is a wonderful work model - you get excellent, qualified workforce for free. It allowed many amazing projects to appear - MMDVM, OpenRTX, WPSD, M17, to name a few in the amateur radio community. There is a big problem behind it though - volunteers can rarely be bound with any obligations or time constraints. This also means no one can have any expectations against volunteers. They can be distracted, their reliability and commitment can span from anything between extremely enthusiastic to hardly interested. It is understandable that people prioritize tasks in their lives - family and daily job is by far more important for most of us than hobby-related projects[10].

This reveals the first issue of volunteer-based work: the difficulty of scheduling work when there aren't reliable resources available[11]. Moreover, one’s good will is not enough to maintain focus on the less fun and more administrative sides of the project. Lifespan of a project can be short regardless of the level of technological advancement offered, due to lack of workforce.

Second issue relates to long term commitment required for sophisticated projects.Many complex functions require more than one person to be involved[12]. This implies project management, reporting, planning and documentation, tasks seldom attractive for volunteers. People come and go, leaving unfinished tasks behind. The turnover rate varies mostly between days and, more rarely, months.

A volunteer-powered project works well when each task can be handled by a single person and when there are not too many interconnections with other functional blocks or submodules. This, of course,

---

[8] https://www.repeaterbook.com/repeaters/niche/index.php?mode=DMR
[9] https://destevez.net/2021/05/32apsk-narrowband-modem-for-qo-100
[10] https://networkingnerd.net/2024/04/26/on-open-source-and-volunteering
[11] IARU “Shaping the Future” programme - https://storage.iaru-r1.org/index.php/s/DAtTorPyFaNFdXK/download?path=%2FInput%20documents%20incl%20amendments&files=ZL23_C3_54%20Shaping%20the%20Future%20programme.docx&downloadStartSecret=5bs5djy8c13
[12] https://stackoverflow.blog/2021/01/07/open-source-has-a-funding-problem

only applies with the assumption that there are no deadlines or other expected time constraints. The project basically has to “live its own life”, at its own pace, dictated by its contributors.

Third issue is maintainer burnout[13], widespread in the open-source community. Volunteering contributors come and go, but maintainers bear the long-term responsibility for the project's health and sustainability. This burden grows quickly with the project's popularity, leading to a form of burnout that leaves maintainers with emotional exhaustion and a decreased sense of accomplishment. One of the major contributors to maintainer burnout is loneliness[14].

As mentioned by Artem Sapegin[15]:
“Open source became a synonym of free labor, not just free code, and it’s not only harmful for the whole community, but mostly for the maintainers of open source projects.”

The seemingly *free labor* suddenly turns expensive in terms of management and commitment sustenance.

In conclusion, this reasoning shows the difficulty of foreseeing a project's future in terms of development, funds management and allocation. That also makes any granting request process extremely hard or even unfeasible, leaving the following questions open:

- Why are there so few developers paid for their work?
- Is there an entity out there having enough funds and willing to change this?

“They all want, but do not commit”

Project’s followers usually have brilliant ideas and provide valuable feedback, but when it gets to implementation, suddenly everyone turns impotent. There are also those affected with severe non-committal disorder, who still keep repeating that they "would love to help", but for some reason nothing ever fits their specializations. Frequently, empty promises appear. Sooner or later, ideas are written off as “not having enough developers to implement” - there’s simply no one to perform the required task - an inherent developer shortage. Most projects just don’t have enough staff (are under-resourced)[16]. This causes frustration and breaks down existing developers’ morale - they are overwhelmed by the number of pending tasks to do.

Another example is the shortage of educational and explanatory materials created by the community, despite the fact that the community has enough knowledge to create it.

---

[13] https://opensauced.pizza/blog/the-lonely-journey-of-open-source-maintainers
[14] https://nutjs.dev/blog/i-give-up
[15] https://dev.to/sapegin/why-i-quit-open-source-1n2e
[16] IARU “Shaping the Future” volunteers and contributors - https://storage.iaru-r1.org/index.php/s/DAtTorPyFaNFdXK/download?path=%2FInput%20documents%20incl%20amendments&files=ZL23_C3_59%20Shaping%20the%20Future%20volunteers%20and%20contributors.docx&downloadStartSecret=9qjobuk5o3v

For this exact reason, most subprojects are run by a single person, or mostly by a single person (*vide*: M17 specification document[17], WPSD[18], the Remote Radio Unit[19], OpenRTX[20]). This burden causes significant emotional stress, easily deteriorating the lone developer's psyche. The effect is further amplified by the pressure coming from the user base, with its never-ending requests and expectations[21].

A possible solution

To be a real threat to the aforementioned *status quo* and bring amateur radio back to its open-source tinkering roots, it is not enough to rely on volunteers, as this model is too inefficient for large, high-impact projects. There is a significant, consistent effort required to provide the critical mass needed to bring products to market.

Justification

There are some good examples of companies that are the owners of open-source products[22]. They all have paid staff to drive the vision and schedule for products, while maintaining good relationships with volunteers contributing to the projects. There must be a clear vision of the for-profit company to help align the volunteers who wish to remain incidental contributors.

Monetary profit is a human motivator that can be used to push the state-of-art forward faster than it would otherwise move. There seems to be a wide assumption that technology will continue to improve, however, that is not a certainty. Hiring an engineer who believes in the goals and vision of the organization relieves the pressure for the individual to have to work elsewhere to earn a living. The engineer's priorities will naturally align with the priorities of the organization. This will help the organization to consistently drive to keep striving.

To sum it up, financial support:

- allows contributors to stay focused on tasks (strong monetary incentive)
- allows the team to make necessary purchases - equipment, licenses, etc.
- makes management easier, as the workforce allocation can be adjusted
- fixes timeframes for tasks
- makes work efficient by making human resources engaged over extended periods of time.

---

[17] https://github.com/M17-Project/M17_spec
[18] https://repo.w0chp.net/WPSD-Dev/WPSD-WebCode/commits/branch/master
[19] https://github.com/M17-Project/rru-rf-hw
[20] https://github.com/OpenRTX/OpenRTX/commits/master
[21] https://github.com/OpenRTX/OpenRTX/issues
[22] https://en.wikipedia.org/wiki/Business_models_for_open-source_software

Potential pitfalls

There should be no apologies for making a profit as this allows the organization to increase R&D efforts and add individuals to the engineering workforce. It can be a difficult transition from a non-profit volunteer based organization, as misunderstandings and hurt feelings can occur. However, by ensuring the vision is communicated clearly to the entire organization, both paid and volunteers, many of the issues are avoided.