Open Tech Strategies

info@opentechstrategies.com
+1 (312) 857-6361

Governance and Collaboration in the Wikimedia Development Ecosystem

22 January 2020

Executive Summary

This report contains some observations and recommendations for improving technical governance and technical collaboration in Wikipedia. It is specifically about the *technical* community: although editorial and content issues in Wikipedia are of course connected to software and platform issues, we have found it useful to separate the two domains as much as possible.

Our original remit was to examine the governance of Wikipedia's Free and Open Source Software (FOSS) projects and to suggest possible improvements, partly through comparison with other FOSS projects. However, our interviews and research quickly led us to see that the most important questions were structural: not *"How are technical decisions made?"*, but rather *"How is the decision-making process guided by the social and organizational structure within which it happens?"*

Thus we have written a somewhat different report than we originally anticipated, but one that we hope sheds light on some underlying issues whose resolution would have a positive effect on Wikipedia.

Here is a summary of our findings:

- **Deployment is the key issue, not development.**

  Technical decision-making in Wikipedia is driven much more by questions of deployment and usage than by questions of software design and development. The governance implications of this are discussed throughout Section 1, and can be summarized as: *Technical decision-making in Wikipedia should start from deployment considerations and then "walk back the cat" to figure out what that means for development.*

  Furthermore, because deployment issues are so dominant, the lessons to be drawn from the realm of free and open source software are limited, as Section 1.1 explains.

- **Make reversible changes and set expectations in advance.**

  There is a strong need for a clearly-defined process — or menu of processes — for deploying changes in a revertible way. This includes post-deployment confirmation that the change is working as expected, which in turn means agreeing on a set of evaluation criteria *before* the rollout, having evaluators actually do the evaluation, and agreeing that if things are not working as planned then there will be swift rollback followed by revision and re-attempt. See Section 1.2 for more.

- **Where possible, decouple political authority from employment status.**

  Where possible and practical, decouple political authority from Wikimedia Foundation (WMF) employment status. In general, whether someone works at the Foundation needn't automatically define their authority in a given domain, although it may affect their level of technical access. Sections 1.3 and 1.5 discuss this in depth.

- **Integrate decision culture into contributor onboarding.**

  Technical contributors are motivated by the prospect of seeing their contributions used in production. Thus, the process of welcoming and onboarding new contributors should provide not only technical knowledge about how Wikipedia works but also cultural knowledge about how decisions are made. Specifically, contributors should be acculturated into designing their contributions to fit the *"reversible changes with clear expectations"* rubric mentioned earlier. Section 2 says more about this.

- **Consider expanding the role of the Community Tech Team.**

  There is potential for the Community Tech Team to serve an even greater cross-pollination and community-amplification role than it serves now. This is discussed in detail in Section 3.

- **Establish practices to counter the centralized↔decentralized mismatch.**

  It is normal for conflicts to arise when a centralized and hierarchical organization, such as the Foundation, interacts on matters of shared concern with a decentralized and loosely polyarchical community, such as the editors and volunteer technical maintainers that that we will broadly call "the Wikipedia community".[1]

  When all participants are encouraged to recognize these conflicts as essentially structural — rather than personal — in nature, it becomes easier to establish practices that can reduce the frequency and severity of such conflicts. This is discussed more in Section 1.4.

We note that the Foundation has already been making changes in the above directions. Indeed, we reached some of our conclusions by looking at various changes the Foundation and the community have made in the last few years and considering their results.

1 Decision-Making and Governance

Wikipedia's technical culture started out need-driven, incremental, and iterative. This was most obviously true of content production, but for a long time it was also true of technical maintenance and development. As the encyclopedia has grown, it is the technical side that has most had to cede its incrementalism. Improvisation works well when everyone knows everyone else: there is a mutual generosity about recovering from mistakes, and mistakes are easier to recover from anyway because one can quickly find and talk with the people whose help is needed for the recovery.

But this approach does not scale either technically or socially, so Wikipedia's growth inevitably raised a question: *What do we replace improvisation with?*

[1]Note that this community includes many WMF employees — "the community" is not the opposite of the Foundation, nor even fully distinct from it.

The current answer looks something like this: enterprise-scale software development performed on an open source code base, plus a functional but imperfectly matched alliance of centralized DevOps (at the WMF) with continued improvisation (from the volunteer technical managers in the per-wiki communities).

We believe this answer can be further improved by moving back toward a slower and more careful incrementalism, with processes in place to protect the stability of the platform and to evaluate the success of changes made. This is not merely about the scope or frequency of technical changes. It is about creating a decision framework that is *designed* to minimize disagreement and conflict, by increasing the degree to which decisions can be subjected to objective evaluation and decreasing the risks involved in implementing those decisions.

1.1 Limited Lessons from Free Software

The most obvious Wikipedia-adjacent example of technical collaboration is the world of FOSS projects.[2] However, the applicability of those projects' governance mechanisms to Wikipedia is somewhat limited.

Very few FOSS projects also host the primary worldwide production instance of their product. Those that do are generally corporate efforts whose Non-Employee Contributors (NECs) have minimal expectation of ownership or of roadmap influence. Wikipedia's focus on its public-facing sites and its sheer scale — in content, in user base, and in community involvement — make it *sui generis* among FOSS projects.[3]

When interviewing for this report, we took care to avoid pushing interviewees either toward or away from the notion that Wikipedia could be managed like most other free software projects.[4] Nevertheless, interviewees steered their comments toward a binary view of technical decision-making, in which the key questions came down to whether the WMF would deploy a given change and how the WMF would negotiate this with the relevant per-wiki communities. Interviewees consistently presented deployment considerations as the main issue, and rarely touched on the kind of development conversations that software projects engage in when they are primarily focused on writing code for others to deploy.

This means that Wikipedia's ability to draw on FOSS projects for governance guidance is somewhat limited. The crucial element of "forkability" does not apply at Wikipedia, because Wikipedia cannot in practice be forked, even though the software that runs it is genuinely free and open source. While we cite lessons from FOSS governance in a few places in this report, we more often recommend importing certain cultural practices from FOSS and transforming them so that they apply to the unique circumstances found in Wikipedia.

[2]Mediawiki as a piece of software is a FOSS project itself, of course.

[3]OpenStreetMap (OSM) may be the closest comparable project, and a focused comparison with OSM project might be a useful avenue for further research. Note that one significant difference between Wikipedia and OSM is that the latter has essentially one primary public-facing instance, and thus correspondingly less potential for localized experimentation and cross-pollination between instances compared to Wikipedia.

[4]All of the interviewees were familiar enough with free and open source software to have a reasonably accurate conception of how such projects are run.

1.2 Predictability: Have a process for rolling out changes

The most important recommendation we can make is simply this:

> *Have clear processes for how changes are described, deployed, and adjusted, and encourage all contributors — Foundation staff and non-Foundation maintainers alike — to make conspicuous use of these processes.*

This will be a set of processes, not just one process. Different guidelines should apply to a change that affects, say, every user on the English Wikipedia than to a change that affects only a specialist interface, or affects only part of a smaller language Wikipedia instance where most of the editors and users know each other.

The following key elements should be part of every such process:

- **Revertibility.** The plan for deployment should include a plan for reversion. Initial success should never be assumed; reversion may be temporary, but it must be possible and planned for.
- **Evaluation.** Every change should be accompanied by the criteria that will be used to evaluate whether it is successful. These criteria should be discussed and agreed on before the change is deployed. For some changes, it may be useful to include explicit failure criteria as well.

  The effort required to conduct the evaluation should be proportional to the significance of the change. (Note that a change's significance may or may not correlate with the change's size or with the effort that went into making it.)

  Evaluations can be qualitative as well as quantitative. Criteria like "we receive mostly positive feedback from users and no strongly negative feedback" are acceptable — provided that feedback channels are in place such that one can reasonably expect to hear from the users affected by the change. Changes that affect a large number of pages or users will tend to be more amenable to quantitative evaluation, including statistical sampling methods.
- **Rollback criteria.** Evaluation criteria make it possible to specify reversion criteria in advance. It is important to do so: knowing them ahead of time helps prevent arguments after a change is deployed with controversial results.

  A reversion may be temporary or permanent, and one may not know at the time which it will be. If people want to press ahead with a re-attempt of the change, then they can make whatever adjustments they need and restart the cycle, re-using evaluation and rollback criteria from the previous attempt as appropriate.

  List only reversion criteria that are specific to the nature of the change. For example, if a change involves deploying a new editing tool, then "we see an increase in bad markup when the tool is used" is a useful reversion criterion, but "the wiki crashes" is not, because the wiki crashing (or any other obviously bad behavior) is *always* a reversion criterion.

- **Schedule.** Lay out in advance the rough schedule for deployment, evaluation period, and possible rollback.

  There is usually no point in planning the schedule beyond one cycle. If a change is temporarily rolled back, one may not know in advance how long the tinkering period will last before a revised version is ready to be deployed. Reversion is an opportunity to revise the evaluation criteria and schedule, along with the technical aspects of the change itself.

Such processes need not be heavyweight — especially for templates and gadgets, they can be quite lightweight. Just make sure that the parties concerned are aware of the upcoming change, the evaluation criteria, the reversion criteria, and the schedule. A rollback is not a sign that something went wrong or that anyone made a mistake; it is a normal occurrence in iterative development and should be treated as a learning opportunity. As one of the interviewees put it: *"The Wiki Way is that we try to make mistakes easy to make and easy to fix."*

If someone shows up after the fact and says "My group wasn't aware of this and we should have part of the discussion", then that is a signal about communications flow rather than about the change itself. Perhaps there needs to be a "Rollout Notifications" register where planned rollouts are posted and that people can subscribe to, similar to how the U.S. Federal Register is a single place where notices of planned government regulatory actions are posted. The key thing is to have all participants bought in to the idea that success criteria are always publicly articulated and agreed on *before* a rollout takes place.

1.2.1 Use Deadlines Only When Needed

Although the scheduling of the next action in any given change process should be explicit, the overall date for the *completion* of the change rarely needs to be predetermined. Situations that require hard deadlines should be rare: instituting a new thing that wasn't there before is unlikely to truly urgent, since whatever that thing is, Wikipedia got along okay without it thus far. If someone is arguing that the new thing be treated as urgent and be given a hard deadline, it is up to them to convincingly articulate the reason to everyone.

1.3 Employment Status and Political Authority

The Foundation is ultimately responsible for Wikipedia. This basic fact is built into the Foundation's legal ownership of servers, domain names, funds, etc, and is reflected in the Foundation's mission. With this responsibility comes a kind of freedom: the WMF is the gatekeeper, but it has wide latitude in choosing what kind of gatekeeper to be.

Many of the recommendations in this document are designed around the idea that, with the right structures in place, the WMF can get more and better technical participation from many contributors, resulting in long-term benefits for the quality and stability of

Wikipedia. However, this expansion of “collaborative surface area” depends on WMF and its partners finding mutually rewarding modes of working together.

One of the most important structural principles Wikipedia can adopt is *to have authority derive primarily from track record and credibility, rather than from employment status.*

This principle does not, in practice, result in revolutions. The people who can devote themselves to Wikipedia all day long are exactly the ones who are most able to accumulate a track record of activity and community credibility the fastest — in other words, being a full-time employee of the Foundation makes it easier for someone to gain technical seniority and influence, just by virtue of their being able to devote themselves to the work. So there will always be a correlation between authority and employment. The principle articulated above is really about the importance of keeping that correlation unofficial.

FOSS development practice provides some prior art here. It is normal in free software projects for someone’s roles in the project to have no formal connection to their employment status; this is especially true in mature projects and in projects that have multiple organizations participating. For example, the person running the release process might be a volunteer with no connection to any of the main corporate sponsors of the project. Even though the sponsors have a strong interest in the next release coming out on time, they just use the same channels of influence available to any other developer to help that release process proceed smoothly.

Similarly, it is a fairly well-established cultural norm in FOSS projects that commit access — i.e., maintainership status — comes through submitting several changes,[5] receiving and handling feedback on them, and successfully shepherding them through to installation in the main code base. Once someone is technically and socially integrated into the project this way, the other maintainers invite that person into the maintainership group.

This “committer” status is traditionally independent of employer. Even if the main corporate sponsor of the project has hired a developer for the specific purpose of working on that code base, the developer still goes through the same review process as anyone else on her way to committership.

In Wikipedia, the analogous positions would have more to do with site maintenance than with software development. And in fact, Wikipedia has already taken steps in this direction, by establishing the Interface Administrator role for example. Our suggestion is to make more such paths available, and that some of those paths be community-initiated rather than WMF-initiated. Think of it as looking for the infrastructure equivalents of what Checkusers and Oversighters are with regards to content.

The advice to “make reversible changes” (Section 1.2) is crucial to this: revertibility and standards for evaluating changes are the technical properties that enable the Foundation to be politically expansive in who it collaborates with and how.

Sometimes a particular Foundation role needs to include technical authority, of course. In those cases it’s worth thinking about exactly how that authority is going to be derived –

[5] “Pull requests” or “patches”, depending on your terminology.

that is, how to ensure that the community as a whole is going to be invested in that role succeeding, so that everyone is comfortable and on board with it. Simply appointing someone to a position won't, in itself, give them any particular credibility with others. However, when someone has already demonstrated credibility in a certain area, appointing them to a role related to that area tends to solidify their credibility, and can even extend it provisionally to adjacent areas.

1.4 When Decentralized and Centralized Meet

The Wikimedia Foundation is a well-defined legal entity with clear boundaries and an internal hierarchy supporting a command structure. "The community", on the other hand, is loosely associated and highly polyarchical. While the community has rules and procedures, they lead to overlapping hierarchies and sometimes even to overlapping claims of authority.

None of that is necessarily bad, and we are not proposing that anything about the fundamental structure of either the community or the WMF should change. It is the difference in structures that makes it procedurally challenging for WMF and any given subset of the community to interact with each other.

When the community as a whole is trying to make a decision, there are fundamental questions such as "Who is the electorate?" and "Who ultimately owns the outcome of this decision?" that may not always have clear answers. For this reason, the WMF, which physically controls the servers and thus feels responsible for outcomes, will tend by default to assign power to itself rather than to a community whose membership (i.e., electorate) and exact boundaries are not always clear.

The best way to temper that natural tendency is to settle on decision criteria for technical matters through public discussion, and then conspicuously adhere to those criteria. Doing this will encourage those non-employee community members who are most invested to participate; those who are capable can eventually take on some technical responsibilities as described in Section 1.3.

From the Foundation side, there is a handy measure available that indicates whether Foundation staff are adhering to this principle: the ratio of public to private technical discussions involving Foundation personnel. The higher that ratio, the more the Foundation is treating the wider community as potential partners in technical matters.

Hiring technical managers who already have FOSS community management experience is one way to improve that ratio quickly. If Foundation employees are talking to each other about technical matters in internal chat channels or via internal email when there is no actual need for confidentiality, it is up to management to notice this and take steps to move the conversations to places where everyone can see them. More broadly, it is management's job to establish a culture of "public by default, private only when necessary" and to make that culture self-perpetuating, so that staff and non-employees mutually support it by consciously pushing each other to have conversations in the open.

There are other advantages as well to hiring FOSS-experienced technical managers. For example, they will be comfortable giving employees time to build community consensus when necessary, and they will understand instinctively that it is as important to help employees develop social and community management skills as it is to help them develop their technical skills.

1.5 Identify the Right Archetype

The Wikimedia Foundation is not an enterprise software company or a VC-funded startup. It is the steward of a multi-stakeholder community. This means it will sometimes have to sacrifice efficiency for long-term stability and for broad buy-in on important decisions. This is normal in multi-stakeholder communities: it is a tradeoff consciously made, to the point where it should show up explicitly in budget projections, staffing decisions, and strategic planning.

A useful comparison from the FOSS world is the Debian Project.[6] Debian produces a distribution of the GNU/Linux operating system, complete with thousands of application packages, development libraries, etc.[7] Although Debian is not primarily an online service, its overall package repository can be thought of as its "production instance": there can be only one repository, and that repository defines the Debian project's output.

No participant in Debian would claim that the project runs efficiently; in fact, complaining about the duration of discussions in Debian is something of a spectator sport among its members. And yet Debian has consistently produced one of the most reliable and widely-used GNU/Linux distributions for more than 20 years. The project is clearly successful, even though — as with any multi-stakeholder community that contains different interest groups — there are always goals some participants have that do not get met.

This does not mean that the Wikimedia Foundation should strive to run Wikipedia like the Debian Project, of course. Our point is merely that the Foundation should not reflexively adopt practices that *appear* applicable at first (such as software development methods used in the for-profit tech sector) but that may not be well-suited to Wikipedia's actual circumstances. The WMF should study its situation carefully, try to adapt methods from other multi-stakeholder groups, and be appropriately cautious when importing methods from endeavors that are very different from Wikipedia.

1.5.1 Educating Funders

The above has an important corollary: Wikipedia's institutional funders need to be made fully aware of what it means for Wikipedia to be a multi-stakeholder project.

Sometimes funders of technical projects attach conditions that, while well-intentioned, are based on past experience with corporate technology development and are thus ill-suited to

[6] *https://www.debian.org/*
[7] A bit over 50,000 packages, as of this writing.

Wikipedia's needs. Most funders are willing to learn when their grantees make an effort to educate them, however. Indeed, funders are just another stakeholder in the multi-stakeholder community. If the process requirements of Section 1.2 and the discussion requirements of Section 1.4 are explained to them, for example, they will usually understand why line items in a proposed budget need to reflect those requirements.

2 Collaboration and Onboarding

Because Wikipedia's collaboration revolves around deployment rather than software development *per se*, bringing contributors on board means teaching them about how the decision to deploy a change is made, in addition to teaching them the technical aspects of the software and infrastructure system. That is, contributors should learn from the start how to design and socialize changes that will be amenable to deployment as per Section 1.2.

One interviewee noted that many of Wikipedia's technology development issues are really about colocated people vs remote people, and that some of the same collaboration practices that would serve the global non-employee contributor community would also serve WMF's in-house development team well. In an entirely colocated team, the line between technical ability to do something and the political authority to do it are blurred. That blur became part of Wikipedia culture for a while early on, when much of the full-time team was colocated, and although it is gradually fading, it is still felt today.

2.1 Unify In-House and Community Onboarding

One way to make the project friendlier to geographically dispersed participants is to *make in-house onboarding be the same process as community onboarding* — or as close to the same as possible, anyway. That is, when a new staff member is brought in to the WMF for a technical position, the onboarding playbook used should be the same one the community uses for anyone new. Of course there will be some WMF-specific internal onboarding as well (access to internal information systems, HR matters, etc). But as far as integration with Wikipedia as a technical project goes, there is no reason to have two different processes.[8]

This implies that the people who help guide the new contributor should, if possible, include both WMF staff and non-employees; this will send the message early on that technical authority and employment status are not the same thing, as described in Section 1.3.

2.2 The New Developer Test

Another useful strategy for maintaining the project's openness to new contributors is to *conduct a "new developer test" on a regular basis* — annually or semi-annually is a good

[8]This is connected to the point (from Section 1.4) that one of the ways Foundation staff can measure the degree to which they are integrated with the non-employee technical community is to look at the ratio of public to private technical discussions.

frequency.

The **new developer test** is to bring in a technical consultant who has no previous experience with Wikipedia,[9] assign them to make a technical contribution, and have them keep a log of the obstacles they encounter on the way to doing so. This log is then made available to the project, so everyone can see where the project needs to improve.

The new developer test is an easy win if done regularly and taken seriously. Among other things, it would help fix the lack of "storytelling" documentation that at least one interviewee noted, because the new developer can say what they most wished they'd had available to read.

2.3 Prioritize Code Review

Code review is one of the areas where compromising on development speed in order to build cohesion in a multi-stakeholder group is a good tradeoff. In Wikipedia's case, "code review" would include not just the code change itself, but review of the full deployment plan, including evaluation criteria and reversion criteria. Prioritizing code review has an amplifying effect for new developers and for one-off or lightweight-but-frequent contributors.

One interviewee suggested that the code review backlog is usually fairly deep — that lots of changes wait a long time to get reviewed. It would be useful to study the size, composition, and provenance of that backlog, given the possibility that potential contributors may be discouraged by the wait and choose not to continue contributing.

As Section 3 discusses, improving code review responsiveness and coverage would be a good area for expanded use of the Community Tech Team. Code review in FOSS projects is traditionally a source of cultural quality as much as of technical quality: it builds social cohesion and crosses organizational boundaries, creating opportunities for peer-to-peer relationships between individuals. These relationships can become the basis for more complex collaborative endeavors later on.

3 Community Tech Team as Glue

The Community Tech Team has the potential to serve an even larger cross-pollination and community-amplification role than it already serves. Community Tech's role is not to work on large core features that taken months to implement — that's for the Core Team — but rather to develop products that address user and editor needs. This means that, unlike the Core Team, Community Tech's work has the potential to scale horizontally, similarly to how wiki content production and editorship have scaled horizontally as Wikipedia has grown.

[9]That is, has no previous experience with Wikipedia as a technical project. If the person is already a content contributor or editor, that's fine and may even be preferable.

This potential for horizontal growth should be fully explored. The first example that comes to mind is that the Community Tech team, if expanded appropriately, might be the right place to coordinate an effort to make templates and gadgets more reusable across different wikis. This would *not* be through Community Tech taking over the writing of templates and gadgets themselves — the energy for that is already coming from the wiki-specific tech communities, almost entirely from NECs in those communities, and the Community Tech team should avoid taking any steps that would usurp or be otherwise suppressive of that energy.

Instead, Community Tech team should focus on amplifying and scaling those efforts. For example, because it naturally has a broader view of the requirements and environments of the different wikis than most individual NECs do, the team is ideally placed to perform code reviews for templates and gadgets. Where reuse possibilities are identified, the team could then work with the original authors to generalize the solution while ensuring that it continues to work smoothly on its original site.

The most effective way to do this is to have a cross-organizational code review team that includes both Community Tech team members, NECs wiki admins from various sites, and any members of the Core Tech team who wish to be involved. Community Tech can provide coordination (such as actively tracking what solution get deployed on what wikis, prioritizing reviews, and providing "honest broker" discussion management services).

Playing such a role would increase the "leaf to trunk" information flow coming in from editors and users, and the Community Tech team is the natural place to aggregate such information. If possible, one or more representatives from Community Tech should regularly join Core Team prioritization meetings as well. This would provide increased assurance to the Core Team that their work is aligned with user needs. Equally importantly it would help all the teams catch problems that might otherwise linger in that middle limbo of solutions that are too large for Community Tech to build but that don't always have a clear route to landing on the Core Team's roadmap.

This recommendation to expand both the scope and the size of the Community Tech team is based on the observation that that team is highly scalable: adding more people — as long as they are the right people – would allow the team to do more of what already works, and to add activities directly adjacent to its current activities, without fundamentally changing the team's nature or management requirements. Just having the Community Tech Team do bigger things probably wouldn't work well. But having it help expand community-initiated work to more environments, and having it help integrate larger development efforts smoothly into the work of the Core Team (or to other teams that typically handle bigger tasks) *would* work, and would be a worthwhile investment.

Another reason to keep the Community Tech team focused on smaller-scale solutions is that then it is less likely to fall victim to the temptation of the sunk-cost fallacy. A classic problem with formalized, heavyweight development methods is that people become reluctant to throw away work: if a manager has just invested six months of a team's time in something, they're going to be reluctant to back out of or change the trajectory of that work even if they really should. For example, new information might indicate that a solution currently under development is likely to be rejected by many of its intended users, or that the solution as designed would place an unexpectedly high M&O burden on wiki admins in their real-world deployment environments.

This does not mean that formalized, heavyweight development methods should never be used; they have advantages too, after all. Our point is just that the Community Tech Team is in an ideal position to serve as a natural check and corrective influence for those inevitable times when the development momentum of a larger feature is mis-aimed or is in danger of becoming so.

4 Methodology

Our research for this report consisted primarily of readings and interviews, and drew on knowledge and experience from all of our work with FOSS projects over the years, especially development and governance guidelines.

The readings were for the most part public documents specific to Wikipedia: Talk pages, Phabricator threads and other discussion forums, development guideline pages, strategy documents, Signpost articles, surveys, blog posts, and various Wikipedia pages about the Wikipedia project itself. We also read some higher-level analyses of FOSS project governance, including Mirko Boehm's "The emergence of governance norms in volunteer driven open source communities",[10] which includes a case study about Wikipedia.

We interviewed five people, the maximum number we could within the scope and budget available. They included both Foundation staff and non-Foundation participants; their names are listed in Section 5 "Acknowledgements".

Each interview lasted between one and two hours. While we examined the background of each interviewee individually and prepared suitable questions, we also encouraged interviewees to lead us where they felt it was most important to go in the conversation, believing that their domain knowledge would naturally steer us to the most important topics. This belief was borne out time and time again in the interviews.

During interviews, we took written notes but did not make audio recordings. We informed each interviewee that our notes were for our internal use and that the raw notes would not be shared outside our company, Open Tech Strategies (OTS) — and thus specifically would not be shared with WMF or with related groups. (WMF had agreed to this condition before we began.) We further told each interviewee that they could request that parts of their interview be kept either fully confidential or unattributed. Interviewees used this very sparingly; the vast majority of what they said was "on the record".

[10] *https://jolts.world/index.php/jolts/article/view/131/249*

4.1 Disclosures

As part of an unrelated contract with another client, we develop an application that extends Mediawiki. That work was simultaneous with the writing of this report, although mostly performed by different staff. It has already resulted in our company having occasional contact with plugin maintainers, and may yet lead to further engagement with upstream core Mediawiki or some of its plugins. After interviewee Peter Forsyth's interview for this report had been completed, we asked him if we could engage him for approximately an hour of his time as a consultant, since he had some familiarity with Semantic Mediawiki that would help us consider whether to use that technology in our application, and he agreed. That engagement did not affect his interview or the content of this report (which he did not have access to), but we mention it here for disclosure's sake.

5 Acknowledgements

We thank the Wikimedia Movement Strategy Core Team and the Product and Technology Working Group for giving us the opportunity to do this fascinating research. In particular, Tisza Gergő assembled a large and well-curated reading list that was immensely valuable throughout our research. Tanveer Hasan focused our inquiry with thoughtfulness and care, which is very important in a social and technical structure as broad and complex as Wikipedia's.

We are especially grateful to our interviewees, who not only gave generously of their time for in-depth conversations but in many cases followed up with supplemental information. They are Derk-Jan Hartman, Amir Aharoni, User:Risker, Peter Forsyth, and Sumana Harihareswara.

6 Acronyms

This chart lists acronyms used in this document, their expanded meanings, and the page on which they first appear.

FOSS	Free and Open Source Software	2
WMF	Wikimedia Foundation	2
NEC	Non-Employee Contributor	4
OSM	OpenStreetMap	4
OTS	Open Tech Strategies	13

Appendix: Elinor Ostrom's 8 Principles for Managing A Commons

While performing research for this report, we came across an interesting set of principles from Elinor Ostrom. Ostrom was an academic economist who shared the 2009 Nobel Memorial Prize in Economic Sciences[11] for her work studying how communities manage resources held in common. The kinds of communal resources she examined (shared agricultural land, irrigation sources, etc) are in many ways analogous to Wikipedia — more analogous than they would be to a pure FOSS project that had only copyable code to manage rather than shared web sites.

We first encountered her "8 Principles for Managing a Commons"[12] after having already drafted a substantial amount of the report, and were struck by the resemblance between some of her principles and some of the conclusions we had come to independently. While not all of the principles apply equally to Wikipedia, we offer her list here because taken as a whole it expresses, in a compact and elegant way, the general idea that commons are best sustained through a shared set of governing agreements that are reinforced and reiterated at every level and adhered to by all.

Elinor Ostrom's 8 Principles for Managing A Commons:

1. Define clear group boundaries.
2. Match rules governing use of common goods to local needs and conditions.
3. Ensure that those affected by the rules can participate in modifying the rules.
4. Make sure the rule-making rights of community members are respected by outside authorities.
5. Develop a system, carried out by community members, for monitoring members' behavior.
6. Use graduated sanctions for rule violators.
7. Provide accessible, low-cost means for dispute resolution.
8. Build responsibility for governing the common resource in nested tiers from the lowest level up to the entire interconnected system.

[11] Formally the "Sveriges Riksbank Prize in Economic Sciences in Memory of Alfred Nobel"; informally and somewhat controversially the "Nobel Prize in Economics". See, of course, *https://en.wikipedia.org/wiki/Nobel_Memorial_Prize_in_Economic_Sciences#Controversies_and_criticisms* for a more detailed discussion of the prize's origin and name.

[12] *https://www.onthecommons.org/magazine/elinor-ostroms-8-principles-managing-commmons* [sic]