“A Better Internet Is Possible” – ICT For The Commons

0. Initial Notes

This text is not a comprehensive overview over either how digital information and communication technologies (ICT) can support the commons, nor what projects strive to do so. It is simply an effort to explicate some of the requirements of a ‘commonist internet’ as implied by a “Global Commoning System” (GCS) in terms of its possible technical implementations as well as its ethical considerations, while also making short assessments of software projects possibly of interest to commoners. While it can be understood as an addition to the project of the GCS as well as the discussion about a ‘commonist internet’ in general, it is also critical of some approaches and projects previously advocated for on this blog and by the GCS itself.

1. Goals & Requirements

The core aim of the GCS in its current form is, firstly, the pursuit of the question of how digital ICT can support commoning, in order to, secondly, help commoners working towards realising a utopian society that enables everyone to live a good life. It is problem-focused insofar as it strives to alleviate problems commoners have with currently dominant ICT, but it also tries to find ways of implementing “patterns” of commons-based cooperation in ICT more generally by relying on alternative socio-ecological infrastructures. Currently, projects like Valueflows or the Institutional Grammar seem to be promising projects for supporting the implementation of such patterns on the application level, but this leaves out the equally necessary underlying infrastructure. Naturally, this should be ‘commonist’ as well, but what specific requirements does such software, or software for the GCS in general – if we leave the hardware part out for a moment – have?

1.1. Designing for Usability

Most generally, ICT should be *usable* (as e.g. described in the ISO series 9241 “Ergonomics of human-system interaction”). The encyclopedists of Wikipedia – in what I assume is an undisclosed reference to the paywalled ISO standard 9241-11 – define *usable* as “the capacity of a system to provide a condition for its users to perform [...] tasks safely, effectively, and efficiently while enjoying the experience”. Professionally, adressing these user needs is also known as “UX” (user experience) design. Notably, *usability* here also includes thinking about the utility/usefulness as well as the accessibility of a given system. ICT for the GCS – i.e. software for inter-, as well as transpersonal cooperation – should be assessed based on its usability, with a critical view on its affordances for intersectional usability. The aim of enabling “global” (or pluriversal) transpersonal cooperation – under conditions of imperial hegemony no less – requires selecting *user stories* specific to the currently existing “global” (or pluriversal) hierarchies (and thereby making fundamental architectural decisions in a certain way), with the goal of making the resulting software *usable* for the “globally” marginalized (and them being included in testing/development). In contrast, the requirements and needs of intersectionally privileged users of the software are correspondingly subordinate (although of course by no means irrelevant). Following this design approach, I first list some ethical design principles that should be followed by ICT in 1.2., and then describe some of the specific utility ICT needs to provide by looking at user stories relevant for a ‘commonist internet’ in 1.3..

1.2. Ethical Design Principles

While there are some great ressources collecting different ethical design approaches/paradigms more or less aligned with commonism out there, this text tries to provide a short – and not necessarily complete – summary of what ethical problems should be considered when developing and/or using ‘commonist’ ICT. In that sense, it is not a “manifesto for commonist ICT”, altough you are certainly invited to think that way.

Put positively, ‘commonist’ ICT should be:

- *resistant.* Resistance against repression, e.g. via unjust censorship or surveillance, is paramount in protecting vulnerable groups; especially when working under conditions of war, totalitarianism or under state authority in general.
- *accessible.* Accessibility should mean free access for everyone, but with a design focus on intersectional access.
- *sustainable.* While a minimized ressource usage in combination with a modular and extensible design improves repairability, “sustainable” also means not doing things if they would require an unjustifiably large part of the total ressources available (i.e. moving beyond the planetary/social boundaries, e.g. by requiring new/expensive hardware).

In turn, ‘commonist’ ICT should not be:

- *hierarchical.* A common example of hierarchy in ICT is the client/server paradigm. Alternatives to such centralized (or federated) systems are often described as “distributed” or “peer-to-peer”.
- *exploitative, abusive and/or oppressive*: ICT should be designed specifically against enabling abuse, exploitation and oppression. A common example for doing so is the design of content recommendation algorithms on platforms like Facebook, Instagram or YouTube.
- *monetized.* Money is a technology for enabling barter logic, which runs contrary to the idea of a commons-based society. This may include other units of account like reputation or ressource contributions, if they are used as currency equivalents.

1.3. Technical Design Considerations

For purposes of brevity and to avoid complicated topics like insecure hardware, this part focuses on explicating major parts of the capacity a ‘commonist internet’ needs to provide access to, along with corresponding – albeit short – user stories:

- *networking/communication.*

    - An activist in Taiwan should be able to securely talk and collaborate in real-time with an activist in the PRC.
    - Neighbourhood resistance committees in Sudan should be able to continue their collaboration after war parties shut down or destroy cell towers and/or cable networks (or some billionaire denies them access to his/her satellite communication network).
- *storage* (can be thought of as a 'cloud' in the sense that it is storage 'on other people's computers', but not *the* "cloud" that is the fragile network of semi-centralized data centers – with the associated social and ecological costs – mostly controlled by well-known capitalist corporations).
    - A whistleblower in a US intelligence agency should be able to safely store and share sensitive documents without exposing themselves.
    - A queer writer in Ghana should be able to publish their work without running the risk of unintentionally outing themselves.
- *computing.* While I am reluctant of including this here due to the – in my opinion disproportionately high – potential for misuse[1], I recognize that the scheduled execution of code might be difficult to standardize/implement otherwise (a common usecase for this is a bot moderating a chatroom). The wish for "smart contract"-like functionality also repeatedly comes up in texts by theorists of the commons (cf. Helfrich & Bollier 2019; Meretz 2023).

Specific to a 'commonist internet' along the lines of the GCS is the capacity to mediate needs and need satisfiers (and to facilitate the establishment of 'patterns', rules or 'recipes' conducive to that end), which in practice means user stories like the following:

- MAPA (most affected people and areas) climate justice activists should be able to safely and effectively reach out to LAPA (least affected people and areas) activists to receive support; even if the individuals or groups concerned have no previous knowledge of each other.
- A care network should be able to safely offer help to patients 'sans papiers' and have reliable translation services either by human or by machine readily available.
- Pregnant people should be able to seek/receive healthcare services, despite criminalization or disinformation regarding possible treatments.
- Survivors of sexualized violence wanting to escape the threat of an abuser should be able to seek/receive counseling and refuge.

Accomodating such use cases needs to be mediated with adversarial usage, like that of a capitalist actor exploiting the offer of free storage to externalise costs or the secret service of a nation-state trying to uncover dissidents.

2. Practices & Projects

The technical design considerations in 1.3. have already narrowed the scope of this part of the text down to technologies of a similar function to what is often summarized as "the internet". Therefore, after a critique of this 'internet' in 2.1. and a short assessment of the current projects of interest listed on the GCS

[1] I think that allocating computational ressources should be left to the application layer as much as possible; a successful example for this is the BOINC project.

website in 2.2., a list of projects generally compatible with the requirements described in 1.2 and 1.3 will be provided in 2.3..

2.1. The internet is broken

The internet – at least in its current form and function – is failing commoners. It is not only hierarchical in that it fundamentally relies on the client/server paradigma; it is also not safe to use (at least without hacks on top, like using the Tor network to try to bypass network censorship and surveillance). Last but not least, while its architecture is generally modular and extensible, its resource usage cannot be justified with the features that claim to require them (currently, the "solution" for increased ressource usage is to just allocate the system more ressources). Indeed, one might argue that the requirements explicated above have little overlap with the initial design goals of the protocols that make up the internet in the first place. This also maps to the "Performance Inequality Gap": Ressource-hungry web applications, designed primarily for the rich, white, cis and able-bodied US/EU consumer with the latest iPhone – stereotypically embodying the imperial mode of living –, are effectively making the web inaccessible[2] to the majority of humanity (of which ~33% were also still "offline" in 2023). The supposed "solution" to this ethical crisis is to expand the imperial mode of living, even though that is clearly unsustainable[3].

2.2. Previous projects of interest

The problems described in 2.1. are mirrored by the 'projects of interest' listed on the GCS website:

- *ActivityPub.* Most famously implemented by Mastodon – and now Threads as well –, this protocol is used as an open interoperability framework for applications running on federated servers. It is not designed to solve the problems of the broken internet stack, with the associated privacy – and possibly scalability – implications.
- *Holochain.* A P2P application framework that is monetized by design with a "native, asset-backed, mutual-credit currency" called "HoloFuel" (right now, the VC-backed company behind it sells "HoloToken", a convertible token that they say can later be exchanged for HoloFuel), which they intend to use for allocating ressources inside the P2P network. Helfrich and Bollier seemingly did not think of this as a problem, but a project that wants to become to cloud hosting "what Airbnb was to hotels" does

[2]While as of 2024, over 95% of the world's most visited websites fail to comply with the the Web Content Accessibility Guidelines (WCAG) as well, 'accessibility' is used in a broader sense here.

[3]Sadly, even projects striving towards a better internet continue to rely on technologies like Javascript/Typescript and the internet stack as a whole, with all the associated drawbacks. While "easy" programming languages like Javascript and Python are considered as having a low barrier of entry for new developers, they also come with a disproportionately higher ressource consumption. While having comparatively higher barriers of entry, languages such as C/C++, Pascal, Rust, Fortran, Go and Ada can be hugely more ressource efficient; with Rust, Go and Ada also coming with increased memory safety affordances. A project's choice of a certain programming language, therefore, (or at least the availability of implementations in languages other than Javascript) should be a part of evaluating the respective project in its usefulness for building a 'commonist internet' complying with ressource constraints.

not seem like a great fit for the values of the GCS, despite the nice things built on top of it.

- *Solid.* A protocol for selfhosting data (which can also be combined with ActivityPub) that continues to rely on the client/server paradigm and that does not adress the flaws of the underlying internet stack.
- *Attributable.* With Attributable building on the Matrix protocol and Matrix on the road to a hybrid P2P network (altough this seems to be currently on-hold due to funding issues), this is a somewhat hacky[4] but generally interesting project working along the same lines as Solid. -> Introduction (in German)

Other, similar projects not listed on the GCS website include PubHubs and SemApps, which also build on the technologies listed above; i.e. Matrix, Solid and/or ActivityPub. Because of this, they are generally federated (as opposed to peer-to-peer/distributed) and build on the broken internet stack. There are many (!) other "blockchain"-based projects supposedly trying to build a "better internet" as well, nearly all of which are monetized in some way. An example for this is "The Open Network" (TON), which is used by Telegram; with complementary – i.e. separable – monetization schemes like IPFS/$FIL being the exception rather than the norm (users of IPFS include notable shadow libraries like Anna's Archive and Library Genesis).

2.3. An 'internet for the commons'?

Instead of the projects assessed in 2.2. (possibly with the exception of Attributable), better approaches to an 'internet for the commons' can be found in the following list[5]:

- GNUnet
  - developed by GNUnet e.V.
  - funding via NLnet
  - launched in 2001
  - lots of research about it
- Veilid
  - developed by Veilid Foundation Inc
  - went public in 2023
  - commitment towards non-monetization and accessibility
  - mobile-friendly application framework (Flutter)
- p2panda
  - funding via NLnet
  - nice collaboration project with the Meli Bees Network
  - mobile-friendly application framework planned (Flutter)

Other interesting, but less complete projects:

- Willow/Earthstar

[4]This may be somewhat of an understatement, given the huge overhead it adds to the stack, its total dependency on the development progress of Matrix and the way the people behind it seem to implement it in Javascript.

[5]Please note that a listing here does not indicate total compliance with the ethical design principles described in 1.2. – as such an assessment would require significantly more technical insight and research – but only a general alignment.

    - focused on storage
- Spritely
    - focused on computing

More information about the selection of the projects for this list can be found here.