Good enough is not good enough

A series about why the open web is collected so badly, why that has survived thirty years of everyone knowing better, what it is now costing all of us, and what is needed to go from “good enough” to “doing things right”.

Roberto Di Cosmo · from August 2026
I direct Software Heritage.

What this is

The series runs on LinkedIn. This page is where it lives afterwards: every post in order, and under each one the sources for every claim it makes — dates, references, links, commit identifiers, intended to make it possible for you to check the facts directly.

It is also here for a more practical reason. What follows is partly about things quietly disappearing from the web, and about what it costs to depend on a single platform. Leaving the only copy inside somebody else's feed would have made the point rather too well.

The posts

01	The wasteful sentence	4 Aug
02	The alternative was written down	11 Aug
03	How good enough wins	18 Aug
04	The mechanism: an externality, not a villain	27 Aug
05	The bill arrives	1 Sep
06	What we did to ourselves	3 Sep
07	One logical collection, many copies	8 Sep
08	What collection actually requires	10 Sep

09 The walls stop the legible traffic 15 Sep
10 We built one too, and it stopped our own users 18 Sep
· next: When the source is no longer there

The series

POST 01

The wasteful sentence

4 August 2026

“This way of collecting data is very wasteful.”

David Senecal, principal product architect for fraud and abuse, Akamai
Business Insider · 19 September 2024

> This way of collecting data is very wasteful.

That is how an Akamai specialist described the way many AI crawlers gather material from the web.[1] The report around the quotation gives the concrete mechanism: some botnets revisit an entire site every day merely to discover what changed, transferring the same material again and again. This is not new: search engines have been doing it for decades, in the very same wasteful way, but the scale of "scraping" is now such that what we have known as the "open web" is now closing down, an API at a time[2], a "bot filter" at a time[3], effectively reducing the global value for everybody, while increasing the waste of resources.

The striking part is not that the method is wasteful. Everybody running a sizeable website knows that by now. The striking part is that we speak as though this were a law of nature.

It is not. We knew how to avoid most of this waste in the 1990s, and it turns out I actively tried to push forward a concrete solution, when the problem was still manageable. We'll get back to this.

For nearly three decades, I have repeatedly encountered the same pattern: a system that works well enough for every actor taken separately, while imposing a growing cost on everybody taken together. Each local decision is reasonable. Their aggregate result is damaging. Since nobody owns the aggregate problem, the better solution remains nobody's job.

It is time to call out this state of affairs, and act upon it... again: I personally tried many times, and now I can build on that experience. The first step is to clearly understand what is happening and why: which actor does what and why, who gains, who loses. In the coming weeks I will reconstruct one instance of the pattern from dated documents. It begins with web crawlers and ends with the institutions we build—or fail to build—for shared digital infrastructure.

Let me point at the destination from the outset: I direct Software Heritage, a non-profit organisation that collects public source code once so that others do not all have to collect it again. This is my latest effort to contribute to systematically address this long standing problem.

It is a problem that starts with two simple words: good enough.

POST 02

The alternative was written down

11 August 2026

One of my first encounters with the "good enough" curse came in the mid-nineties, during the ferocious battle to control the rapidly expanding cyberspace.

WebCrawler, AltaVista, Alexa, Yahoo… Google, the eventual winner, had not even been born yet.

What stunned me was that all of them did essentially the same thing: download the Web again and again to discover what had changed.

Sounds familiar? It should: it is exactly what AI crawlers do now.

The Web was much smaller then, but bandwidth, compute and storage were scarcer too, so the waste was already real.

And there was an obvious alternative.

In January 1998, with Pablo E. Martínez López — aka Fidel — I wrote Distributed Robots: a Technology for Fast Web Indexing. It entered the IETF record as an individual Internet-Draft on 4 March 1998,[4] carrying the wonderfully explicit line: "EXPIRES SEPT 1998."

The idea was elementary, familiar to any systems programmer: interrupt-driven beats busy-wait.

A web server knows, cheaply, when its files change. Let it say so, and let crawlers fetch only what is new — instead of asking every server, over and over, "Anything changed?"

In May 1998 an IETF Area Director sent us a generous, detailed critique. By then I had moved on: that March I had put Piège dans le Cyberespace online — CyberSnare in English — which went viral, started thirty years of work on free and open source software, and became a book with Dominique Nora.[5]

It took me seven months to answer point by point. I first asked what he thought of the revised structure.

The answer was an automatic out-of-office message. The thread ended there.

We produced the revised draft anyway, dated 20 April 1999, incorporating the review.[6] Fidel sent it to the RFC Editor. No answer came.

But the problem had not disappeared. In 2001 the IETF's own WEBI group independently produced Requirements for a Resource Update Protocol.

That expired too.[7]

Years later, looking at my web server logs, I found that in 2009 Googlebot had fetched one unchanged page twenty-four times.[8]

Students of mine implemented the Remote Update Protocol twice, in Java and OCaml.[9] The technical idea was not the hard part.

And this is the point: a better technical solution does not deploy itself.

Someone must maintain it, persuade others to adopt it, integrate it into existing systems, and keep pushing after the prototype works. Advocacy on the side of a full-time job is not an institution.

The alternative was written down almost thirty years ago, then rediscovered independently inside the IETF.

It was not defeated by a better idea.
It was not shown to be technically impossible.
It simply never acquired enough organised support to become infrastructure.

Meanwhile, the "good enough" solution kept scaling.

Why?

POST 03

How good enough wins

18 August 2026

So why did it survive?

Not because anyone defended it. Because it was made liveable, one patch at a time.

Let's be precise about what "good enough" means: it does not mean bad. A good enough system does not do the right thing, but it does get the thing done. It is a hack, a wooden leg that gets you across the room… And because it gets you across the room, you stop looking for a better leg.

Downloading the whole web to find out what changed was never sensible. It was patched until it was bearable.

In 1994 Martijn Koster proposed robots.txt, so a site could say: not this, not here. It worked well enough that nobody standardised it until 2022.[10]

In 2006 came sitemaps, so a site could say: here is what I have, and here is when it last changed.

Read the specification closely. The freshness field is "a hint and not a command", and crawlers may ignore it.[11] So a site can say exactly what changed, and you are free not to believe it and download everything anyway. Which is what happened.

Then the crawlers got cleverer about where to spend. Google's own documentation says it plainly: "URLs that are more popular on the

Internet tend to be crawled more often."[12] And if your server struggles, the crawler backs off.

Add it up and you have a system that works. Big sites get visited often, small sites rarely, a struggling server gets a break. Nobody is delighted, but nobody is ruined.

That is what good enough looks like from the inside: not a catastrophe, but a tolerable arrangement everyone has adapted to.

And then the second thing happens, which is worse: it becomes normal.

You are almost certainly reading this on a QWERTY keyboard. Nobody chose it this morning. Everybody knows it, every keyboard has it, and whoever switches first pays the whole cost alone. So it stays: not because anyone re-decided it, but because it is installed.

That is where web crawling ended up. Installed. It always worked like this.

In December 2007 I gave a talk at Inria's fortieth anniversary in Lille, written up as an article I revised into 2011. I wrote that the current system was working « assez bien ».[13] French had no phrase for what I meant, so I borrowed one, in a footnote: « Good enough, comme on dit chez nos amis anglosaxons. »

I was not complaining. I was describing. It was accurate.

Lock-in is harmless when the stakes are low. Nobody is much hurt by a keyboard layout.

Repeated crawling is not. It spends bandwidth, electricity and machine time on a planetary scale to fetch what has not changed.

Thirty years on from robots.txt, the patches are not holding, and everyone can see it.

So the question is not why nobody noticed the waste. Everybody noticed.

The question is why a cost that large stays invisible to the people who could act on it, until it is too late to act easily.

That is not a technical question. Economists have a name for it, and it is the whole point that we need to delve into.

POST 04

The mechanism: an externality, not a villain

27 August 2026

AI bots are redownloading the web at scale, just to see what is new. Everyone can see the waste. Almost nobody can see the full bill.

That is a large part of the problem.

It is tempting to point the finger at somebody who is greedy, careless or malicious. Sometimes that may even be justified. But it does not explain why the same pattern keeps appearing among competent people acting in good faith. My 1998 IETF draft (RUP) showed how to cut the polling; sitemaps let a site declare what changed. Yet twenty years later the repeated downloading continues.[14]

If we want to change the outcome, we first need to understand the mechanism.

Economists have a name for part of it: an externality.[15]

Suppose there are N sites and M organisations that want an up-to-date copy or index. In the naïve arrangement, every collector must discover and revisit every site: N × M relationships, policies, rate limits and repeated transfers.

The cost of each crawl may look small to the crawler, and merely irritating to each site. But the aggregate cost can be enormous, while appearing on no single balance sheet.[16]

We could do much better. Mutualise the collection: one shared infrastructure talks to the N sites. The M organisations obtain the data from it. We move from N × M interactions to N + M.

With a thousand sites and ten collectors, that is 1,010 relationships instead of 10,000.

More importantly, no page is fetched ten times over just to learn it has not changed.

That is more than my 1998 draft did: it cut the polling, not the connections.

This does not mean one machine, one copy, or one organisation controlling everything. The data can be replicated, held by independent institutions under different jurisdictions.

"Collect once" is not "store once".

Why don't we do that? The technique is not missing, and people are not villains. It is what economists call a market failure.[17] Whoever operates the shared path pays for it, while the benefit goes to everyone: exactly the kind of good that markets under-provide, a public good nobody funds. A sound protocol does not create the institution whose job is to run it.

There is another trap. Universities, companies and research infrastructures respond by building their own partial catalogues, caches or archives, reproducing the problem one level higher. Every new silo becomes another place that others must discover, query and reconcile.

So the answer is not to punish people for using an open resource, nor to put a tollbooth in front of it. It is to make the efficient shared path exist, and easier to use than rebuilding the same machinery again and again.

That requires engineering, of course.

But it also requires resources, governance and a mandate: a market failure is not fixed by moral injunctions, but by making someone responsible for the shared path.

And this is where "good enough" stops being merely a technical problem.

POST 05

The bill arrives

1 September 2026

Almost thirty years ago the externality was known, and seen as easy to treat as an elegant systems problem. We made it bearable a patch at a time. Some search engines did eventually get half of the fix: since 2021 a protocol called IndexNow lets a site announce a change once, to everyone[18]. It moves announcements, not bytes, and the crawlers that arrived in 2024 use none of it: the bill started to explode, the pain grew, and other patches arrived.

That summer Software Heritage (the archive I direct) was hit by waves of automated collection associated with AI training.[19] Some public services became unresponsive. Engineers who should have been improving the archive spent their time identifying traffic, blocking addresses and restoring service.

We were not an isolated case, far from it. Small and medium-sized web operators described the same sequence: repeated retrieval of unchanged material, rapidly growing bandwidth and compute bills, degraded service for human users, then emergency blocking. Large platforms tightened unauthenticated access. Smaller ones put computational challenges in front of their websites or shut expensive paths altogether.

These reactions are rational. A service under pressure has to defend itself with the means available that week.

Their aggregate effect is damning. Defensive restrictions do not only stop abusive collectors. They also make legitimate research, preservation, accessibility tools, independent search and ordinary automation harder. The open web becomes less open one emergency measure at a time.[20]

This is how an externality turns into institutional damage. The first-order waste is duplicated traffic. The second-order cost is that every site is pushed to make access more brittle for everybody.

Let's not be naive: some actors act abusively, regardless of efficiency costs. They may disregard declared restrictions, conceal identity or support uses the producer rejects.

A shared collector does not make those disputes disappear. It does make them tractable.

A producer can interact with a small number of identifiable collection services instead of an adversarial cloud of indistinguishable bots. Terms, provenance and update information can be recorded once. Disagreements can be attached to a stable process rather than fought through IP-address whack-a-mole.

A global shared infrastructure has its challenges: will it slow down innovation? Who owns it? Who governs it? But the current "far west" alternative to shared infrastructure is not freedom from governance. It is thousands of hurried, uncoordinated and incompatible governance decisions embedded in firewalls.

This is not the first time that I am confronted first-hand with this pattern: in academia we faced a similar challenge when internet and the web changed scientific publishing, and I had a seat in the front row.

POST 06

What we did to ourselves

3 September 2026

The problem is not confined to commercial crawlers. The research community reproduced the same structure for decades, dressed in the same costume: a law of nature.

In 2005–2006 I wrote about scientific publishing and listed five things a researcher needs from publication[21]: a date that proves who was first; integrity, no change after it appears; circulation without barriers; very-long-term preservation; one reference that never breaks. Judging quality is not on the list; it belongs to editorial boards. The five are infrastructure, and arXiv with its overlay journals showed how to deliver them. I saw the forced transfer of copyright as the blocker, not the infrastructure. When in October 2015 a French bill arrived carrying an article titled "free access to the scientific publications of public research"[22], I filed an amendment. One line: forbid the exclusive transfer of rights. Simple, clear, and it would have worked.

Instead, the law that passed in October 2016[23] grants authors the right to deposit the manuscript their publisher had accepted (never the published version), never for commercial republication, and only after an embargo: six months in the sciences, twelve in the humanities. A law granting researchers permission to circulate their own work, as though the right were the publisher's to concede. Notice which solution won: not the simple one that removes the problem, the complicated one that preserves it.

The simple fix had a vested interest against it; the complicated one had only the unorganised research community against it, which is why it won.

Meanwhile the infrastructure, which I had considered solved in 2005 — just put everything on arXiv — became a problem of its own. Universities and research institutions, footing the bill, bet on Open Access repositories. They built them. Thousands of them.[24] Each local decision was rationally defensible: reporting obligations, local curation, independence from any single service.

The global result was dismal: every aggregator must now find, poll and reconcile thousands of endpoints, where one article lives in many copies with different metadata, different timestamps and, by the law's own design, different texts. By the 2006 list, integrity and unique reference both fail. Local autonomy, implemented as disorganised global duplication.

The lesson, hard earned: keep the local functions where they belong, mutualise aggressively what is universal — durable identification, preservation, deduplication, discovery — and replicate that layer so it depends on no one operator.

It was never a technical failure; repository teams built what their institutions commissioned. The absence was above their level, and above mine: nobody had a mandate strong enough to provide the common layer.

POST 07

One logical collection, many copies

8 September 2026

Post 06 ended with a rule: keep local functions where they belong, mutualise what is universal.

What does that look like in practice?

A concrete example. In May, the Open Invention Network, a sponsor of Software Heritage, announced that the archivable source code behind its OIN 2.0 Linux System — more than 5,100 open-source packages, around 900 million lines of code — is preserved in Software Heritage. [25]

OIN did not have to build and operate another universal crawler and archive. It could build on a collection layer that already existed.

The important point is not the name of the collector. It is the architecture.

Collect once into one uniform model. Deduplicate globally. Give every captured artefact an intrinsic identifier derived from the artefact itself, not assigned by a registry. Then let many services build on the result.

But one logical collection does not mean one physical copy.

A serious archive must have complete replicas held by independent institutions, in different locations and jurisdictions.

One logical collection, plural custody.

Europe can anchor such an infrastructure, but an archive of the world's source code cannot credibly remain a purely European asset. A universal archive needs genuinely international custody and governance.

Intrinsic identifiers are easy to oversell, so let me be precise about what they buy.

Precision: we can establish whether we are talking about the same file, directory, revision or release.

Independent verification: for a file, anybody holding the bytes can recompute the identifier, without consulting a registry.

Portability: the reference survives movement between servers and organisations.

What they do not buy is just as important.

An identifier does not preserve the object it names. It does not guarantee neutrality or good governance. And it does not settle law, consent or authorship.

A hash establishes identity, not legitimacy.

Since April 2025, SWHID is ISO/IEC 18670[26]. That matters because a procurement rule, policy or compliance process can now refer to an international standard, not to one organisation's convention.

Software Heritage is one implementation of this architecture.

The point of the architecture is precisely that you should not have to trust Software Heritage to verify what it says.

If the case for the shared layer depended on trusting us, we would have designed it wrong.

POST 08

What collection actually requires

10 September 2026

“Collect once” may sound like a slogan. It is not. There is quite a bit of engineering behind those two words.

Take Software Heritage.

We do not download every repository again and again just to discover whether something changed.

When a forge provides the necessary information, we first retrieve a compact listing of its projects and their recent activity[27]. We compare that with what we already know. If nothing changed, there is nothing to do.[28]

And when a repository did change, Git gives us another important advantage: we negotiate what is missing and transfer the new objects[29], instead of downloading the whole repository again.

This is still polling, and I would prefer notification. But it is already very different from blindly asking millions of repositories, over and over, “anything new?”

Of course, reality is messier.

Not every forge tells us reliably what changed. Not everything is Git. Projects move, disappear, become private or rewrite their history. Supporting all of this is a permanent engineering job.

And notification, by itself, only solves half of the problem.

This is something I had missed in 1998. My proposal let a server notify crawlers when something changed, but every crawler still had to register with every server.

Today IndexNow goes further: a site can announce a change once, and participating search engines share that notification[30].

That is better.

But every search engine still goes back and fetches the page for itself.

Announcing once is not collecting once.

A shared collector removes that second multiplication too: detect the change once, retrieve it once, verify it once, preserve it once — then let everybody reuse the result.

There is another kind of duplication we should avoid as well.

Every organisation using collected material needs to know where it came from, when it was seen, and what terms and metadata accompanied it. Today that evidence is repeatedly reconstructed, imperfectly, by everybody downstream.

A shared archive cannot decide every question of copyright, consent or jurisdiction. Nor should it try.

But it can preserve the evidence: where and when something was found, what public information accompanied it, and what corrections or claims came later.

So the old problem has two parts.

Avoid repeatedly discovering and downloading the same thing.

And avoid repeatedly reconstructing what we already knew about it.

Both are infrastructure problems. And infrastructure has an owner, a budget and a horizon — or it has none of them.

▸ Notes on the sources

POST 09

The walls stop the legible traffic

15 September 2026

The open paths are being walled off. The awkward part is that the clients easiest to recognise as recurrent visitors are often the ones behaving best.

A preservation service, research crawler or webhook normally does what we ask responsible automated clients to do: it comes from predictable infrastructure, identifies itself, and accepts a rate limit.

That also makes it very easy to block.

A large abusive collector can do the opposite: spread requests over many addresses, rotate identities, and run full browser engines[31] so that each request looks much more like ordinary human traffic.

So a wall based on repeated or easily identifiable access has a perverse effect: the honest client presents a stable target; the abusive one can disappear into the crowd.

That does not make the wall irrational. When a site is being overwhelmed, the operator's first duty is to keep it running.

SourceHut faced exactly this in 2025. After days of disruption from LLM crawlers, it put a proof-of-work challenge in front of the web interface, while leaving API access and Git operations untouched.[32]

That is the right distinction.

Open access does not mean that every interface must accept unlimited anonymous traffic. Authentication, published quotas and dedicated archival feeds are perfectly compatible with openness. What matters is that accountable machine access has an efficient path, instead of forcing every serious client to pretend to be a browser.

And a legitimate large-scale user should not need to crawl the world in the first place. If a documented, versioned, provenance-carrying corpus is available through an efficient bulk interface, using it should be easier than launching yet another crawler. Those terms should be published and the same for everyone — an efficient path in, not a private arrangement with the largest users.

This is where a shared layer changes more than bandwidth. Its access rules can be published, argued about and improved. And somebody has to be accountable for them. When collection stays fragmented, the same decisions are made thousands of times, invisibly, in firewall rules and emergency patches.

We have solved this kind of coordination problem before.

Early versions of the European copyright directive would have put software-development platforms in scope of the upload-filtering debate. The final text explicitly excludes “open source software development and sharing platforms[33]”. Many people worked for that exclusion; I spent part of 2018 and 2019 on it too[34].

Collective action can change the default.

Disclosure: GitHub and Microsoft support Software Heritage. That support does not make our interests identical.

I would like to say that Software Heritage consistently followed the principle I have just stated.

We did not.

▸ Notes on the sources

POST 10

We built one too, and it stopped our own users

18 September 2026

Software Heritage put up a wall too. It stopped some of the people trying to give us code.

The reasons were serious. In April 2024 a single address made more than seven million content downloads in two days.[35] By September 2025 our legacy forge was answering public requests with 403 — “a quick and dirty action to stop the crawlers”, says the ticket[36] — because everything behind that server was going down with it.

So in January 2026 we deployed Anubis, a proof-of-work challenge. If a request does not look sufficiently like an ordinary browser, the browser must solve a small computational challenge before getting the page. It went live on 30 January.[37]

Three days later, a bug was filed on my own repository.[38] The Software Heritage browser extension — the one-click button that means “archive this repository now” — had stopped working.

This was a client used by people who were giving us source code, for free, on purpose.

Someone guessed the cause in under a minute. I confirmed it that morning: “I confirm this is a side effect of Anubis.”

It was patched the same afternoon. The commit message contains the thesis of this series, almost accidentally:

> (This is a trivially circumventable "are you a browser?" check, but one can't answer an Anubis challenge when doing cross-site requests to the API anyway)

There it is.

A crawler prepared to behave like a browser can get through. The honest purpose-built client, which cannot answer a browser challenge it was never designed to see, cannot.

A later rule change caused another API regression, and we ended up adding an explicit allow rule for our own extension.

This is not a story about careless engineers. Under pressure, the team chose a defence that could be deployed quickly and restored the service. I would have approved it.

The point is structural.

A proof-of-work challenge does not know whether a client is useful, honest or abusive. It tests a technical capability. Attackers can adapt to that capability; small specialised clients often cannot.

We knew the distinction. I have been writing about it since 1998. Yet when our own service was under pressure and the resources were not there, the affordable action was still the wall.

That is what “locally rational” means. It is not a polite name for a bad decision. It is why competent people, acting for good reasons, keep reproducing an arrangement that is bad for everyone.

A shared collector is not magically exempt from the externality. Unless it has the resources to provide efficient bulk and incremental access, it will eventually defend itself like every other site.

▸ Notes on the sources

References

Every source for every claim, numbered in the order the posts cite them. Each one carries a way back to the place it was cited from.

[1] POST 01 Darius Rafieyan, *“Like digital locusts, OpenAI and Anthropic AI bots cause havoc and raise costs for websites”*, *Business Insider*, 19 September 2024. Archived copy: annex.softwareheritage.org. The speaker is **David Senecal**, principal product architect for fraud and abuse at Akamai. **The sentence in full:** “This way of collecting data is very wasteful,” he said, “but until the mindset on data sharing changes and a more evolved and mature way to share data exists, scraping will remain the status quo.” The sentence that follows in the post — botnets crawling a whole site daily — is the **reporter’s** prose summarising him, not a quotation, and is given as reported speech for that reason. ↩ back to the text

[2] POST 01 GitHub tightened its limits on unauthenticated access on 8 May 2025, citing “an increase in scraping activity targeting our API”. Smaller operators went further: SourceHut placed a proof-of-work challenge in front of its web interface after a week-long crawler incident in March 2025, and GNOME, KDE, Fedora and Codeberg have each done some version of the same. ↩ back to the text

[3] POST 01 Longpre et al., *Consent in Crisis: The Rapid Decline of the AI Data Commons*, Data Provenance Initiative, July 2024 (arXiv:2407.14933) — an audit of ~14,000 web domains behind three widely used AI training corpora, finding a sharp rise in crawl restrictions across a single year. ↩ back to the text

[4] POST 02 R. Di Cosmo and P. E. Martínez López, *“Distributed Robots: a Technology for Fast Web Indexing”*, written January 1998. Record: datatracker.ietf.org. Full text, with everything that followed it: dicosmo.org/RUP/. The document prints “EXPIRES SEPT 1998” and gives no day, so none is claimed here. ↩ back to the text

[5] POST 02 **« Piège dans le Cyberespace », and the book that came out of it.** The essay appeared in *Multimédium* (Canada) on 17 March 1998. It is online free and complete, in five languages, at dicosmo.org/Piege/cybersnare/ — French, English as *CyberSnare*, German as *Falle im Cyberspace*, Italian as *Trappola nel Cyberspazio*, Spanish as *Trampa en el Cyberespacio* — with a Chinese version at PiegeCN.html. The book is *Le Hold-up planétaire : la face cachée de Microsoft*, with Dominique Nora, Calmann-Lévy 1998, ISBN 2-7021-2923-4; Kirk McElhearn’s English translation, *Hijacking the World: the dark side of Microsoft*, followed in October 1998. When the publisher stopped reprinting in July 2006 the two authors recovered their rights and put the book out under Creative Commons Attribution-NonCommercial-NoDerivs. dicosmo.org/HoldUp/ carries the French, English and Spanish texts in full — the Spanish, *El Asalto Planetario*, never having had a print edition at all. ↩ back to the text

[6] POST 02 **The review, the resubmission, and the silence.** On 10 May 1998 an IETF Applications Area Director reviewed the draft in the IESG and asked that the objects be defined as MIME types and distinguished from the FIND working group’s CIP and SOIF and from the W3C’s RDF. The point-by-point answer went back on 8 December 1998, accepting the MIME and SOIF encapsulation and asking one question before resubmitting; the reply, ten seconds later, was an automatic out-of-office, and no further correspondence followed.

draft-RUP-01.txt, *The Remote Update Protocol (RUP). Part I: RUP Architecture*, is dated 20 April 1999 and does what the review asked. **It was submitted:** Martínez López sent it to the RFC Editor on 16 April 1999, with the file attached, describing it as “an update for the internet-draft <draft-rfced-exp-cosmo-00.txt>”. **No reply ever came** — the same address had answered the 1998 submission twice within hours — and six days later he wrote asking whether to chase them or wait a little longer. No IETF record of draft-RUP-01 exists. **Nobody behaved badly:** a volunteer reviewer was away for a fortnight, the answer had taken seven months, and after that carrying it forward was nobody’s job. The correspondence is in the author’s own archive. ↩ back to the text

[7] POST 02 Twenty-four Googlebot fetches of one unchanged page across 2009, twelve of them answered *304 Not Modified*. The log is reproduced in full on page 9 of the Inria anniversary article, which is now online with the talk it came from: dicosmo.org/Inria40/. ↩ back to the text

[8] POST 02 **“Requirements for a Resource Update Protocol”**, draft-ietf-webi-rup-reqs, IETF **WEBI** (Web Intermediaries) working group. The first revision is M. Hamilton (JANET Web Cache Service) and I. Cooper (Equinix), 22 February 2001; Dawn Li and Mike Dahlin joined later; the last revision is dated 4 March 2002, and it expired. Its consumers are caching proxies and surrogates, *not* crawlers — a different protocol for a different audience. What is shared is the diagnosis, in its own abstract’s words: such a protocol is needed where “periodic revalidation is unacceptable in terms of performance and/or cache consistency”. There is no evidence the two efforts knew of each other, and none is claimed: the point is that the same conclusion was reached twice, independently, and expired twice. ↩ back to the text

[9] POST 02 Both implementations were student projects at Paris 7 — *Travaux d’Étude et de Recherche* on the subject “indexation rapide du Web”, which Roberto set for two years running. The RUP 1.0 Java servlet, by Yerom-David Bromberg, is at dicosmo.org/RUP/RupJava/. **OCamlRup** — a RUP client, a generic RUP server and an Apache CGI server, with `rupinfo.txt` parsing and `robots.txt` integration — was written in 2001 by Samuel Lasry and Xavier Patourel and is archived at ocamlrup-0.1.tar.gz. Copyright remains the authors’; no licence was ever attached and none is asserted — it is published as an archival record, with attribution. ↩ back to the text

[10] POST 03 **robots.txt.** The Robots Exclusion Protocol was, in the words of the RFC that eventually specified it, “originally defined by Martijn Koster in 1994 for service owners to control how content served by their services may be accessed”. It became an IETF standards-track document only in September 2022: RFC 9309, by Koster with G. Illyes, H. Zeller and L. Sassman. Twenty-eight years of running on a convention. ↩ back to the text

[11] POST 03 **Sitemaps, and the word “hint”.** The Sitemap protocol 0.9 says of `changefreq`: “Please note that the value of this tag is considered a hint and not a command.” It goes on: crawlers “may crawl pages marked ‘hourly’ less frequently” than stated. The protocol lets a site describe its own freshness; it obliges no one to act on the description. That asymmetry is the whole of this post. ↩ back to the text

[12] POST 03 **Crawlers spending where it pays.** Google’s own documentation on managing crawl budget: “URLs that are more popular on the Internet tend to be crawled more often to keep them fresher in our systems.” And on backing off: if a site “slows down … or responds with server errors … the limit goes down and Google crawls less.” Adaptation, not repair —

the polling continues, more politely. ↩ back to the text

[13] POST 03 From a talk given at Inria's fortieth anniversary, Lille, December 2007, written up as an article revised into 2011: « le système actuel fonctionne *assez bien* », with the footnote *"Good enough, comme on dit chez nos amis anglosaxons."* Both are now online at dicosmo.org/Inria40/ — the slides as the audience saw them, and the article, which was never finished; the footnote is on page 10. The article is cited as talk December 2007, article revised into 2011; no span of years is computed from it anywhere in this series. ↩ back to the text

[14] POST 04 **Sitemaps, twenty years on.** sitemap.xml was published in 2006; the specification and the word "hint" are in post 03's sources. The same footnote that gave this series its title describes it as a partial push/pull fix — one that worked for organised publishers and left everyone else on polling. That is the point being made here: the interface arrived, and the party whose job it would be to operate the other end never did. ↩ back to the text

[15] POST 04 The externality passage is from the same article as post 03, a few paragraphs after the footnote. The argument is given there in an economist's terms: fixing the arrangement would require cooperation between search engines and sites, while the broken one wastes *common* resources that no isolated actor perceives as a significant cost. ↩ back to the text

[16] POST 04 The term is A. C. Pigou's, from *The Economics of Welfare* (Macmillan, 1920), where it names a cost borne by people who were not party to the transaction that produced it. The 1920 first edition is out of copyright and readable in full at the link. Nothing in the argument here is new economics; the only claim is that web crawling is an instance of it. ↩ back to the text

[17] POST 04 Francis M. Bator, "The Anatomy of Market Failure", *The Quarterly Journal of Economics* 72(3), August 1958, pp. 351–379 (doi:10.2307/1882231) — the paper that gave the term its careful definition. The publisher's page is behind a paywall for the full text; the citation is given in full so the paper can be found in any library. ↩ back to the text

[18] POST 05 **IndexNow**, published 2021 by Microsoft and Yandex. Its fan-out is the protocol's own commitment: "Search engines adopting the IndexNow protocol agree that submitted URLs will be automatically shared with all other participating search engines" (documentation). It carries URLs, not content: "After you submit a URL using IndexNow, the search engine evaluates whether it should crawl the URL" (FAQ) — so the announcement is shared once and the fetching still happens once per engine. The participants listed by the protocol are Amazon, Bing, Naver, Seznam.cz, Yandex and Yep: all search engines, no AI training crawler, and not Google. That the 2024 crawlers use none of it is read from that roster, not from any statement by their operators. ↩ back to the text

[19] POST 05 Darius Rafieyan, *"Like digital locusts, OpenAI and Anthropic AI bots cause havoc and raise costs for websites"*, *Business Insider*, 19 September 2024. Archived copy: annex.softwareheritage.org. The article carries the account of this incident, in my own words, alongside the measurements published by Edd Coates of the Game UI Database and an extrapolation to AWS list rates. ↩ back to the text

[20] POST 05 Longpre et al., *Consent in Crisis: The Rapid Decline of the AI Data Commons*, Data Provenance Initiative, July 2024 (arXiv:2407.14933). Any percentage quoted from it is

restated with the study's own population — tokens across the C4, RefinedWeb and Dolma corpora — and never from a secondary summary. ↩ back to the text

[21] POST 06 « Publication scientifique : le rôle des États dans l'ère des TIC », dated 17 February 2006, with drafts on file from July 2004 and July 2005 — hence "2005–2006". The five requirements are the article's own, in its own order: *prise de date* ("établir l'antériorité d'une découverte"), *intégrité* ("garantir qu'aucune modification ne soit apportée à un article après sa parution"), *large diffusion sans entraves*, *archivage à très long terme*, and *référence unique* (a web of knowledge "ne peut se permettre de liens cassés"). The article also argues that the overlay journals grown on arXiv split the work: quality and date from the editorial board, diffusion, archiving and unique reference from arXiv. Full text. ↩ back to the text

[22] POST 06 The title is quoted here in translation. The article belongs to the autumn-2015 public consultation on the *projet de loi pour une République numérique*, where it is Article 9, « Libre accès aux publications scientifiques de la recherche publique » — not the numbering of the bill as later tabled. The consultation site is still up and the heading can be read there. ↩ back to the text

[23] POST 06 **LOI n° 2016-1321 du 7 octobre 2016 pour une République numérique**, article 30, which created article L533-4 of the Code de la recherche, in force since 9 October 2016. The right is to deposit « la version finale de son manuscrit acceptée pour publication » — the accepted manuscript, never the publisher's version — after « au maximum de six mois » in science, technology and medicine and twelve months in the humanities and social sciences. The wording that matters here is that the right holds « même après avoir accordé des droits exclusifs à un éditeur » — even after exclusive rights have been transferred. The law routes around exclusive transfer instead of forbidding it. ↩ back to the text

[24] POST 06 The count is deliberately given as "thousands". The Directory of Open Access Repositories is the register to check; its published figure moves, and a rounded number would be wrong within months. ↩ back to the text

[25] POST 07 Open Invention Network, announced 18 May 2026: more than 900 million lines of source code making up OIN 2.0's Linux System, across over 5,100 open source technologies. OIN's own wording is that the source code *that is archivable* is now comprehensively preserved — that hedge is theirs and is kept here. The post claims only that a third party with a hard legal requirement used an existing collector rather than building its own; it makes no claim about how OIN assembled its inventory. ↩ back to the text

[26] POST 07 **ISO/IEC 18670:2025**, *Information technology — SoftWare Hash IDentifier (SWHID) Specification V1.2*, edition 1, ISO/IEC JTC 1. The ISO catalogue gives the publication date as 2025-04 and no day, which is why the post says April 2025. The point being made is about what a named standard makes possible downstream for regulators, courts and procurement offices — not that standardisation improves the identifier. ↩ back to the text

[27] POST 08 `swh-lister` records the freshness field each forge publishes: GitHub's `pushed_at`, GitLab's `last_activity_at`. ↩ back to the text

[28] POST 08 The scheduler's `already_visited_order_by_lag` policy in `swh-scheduler`: an origin becomes a candidate for a new visit only when the forge's reported last-update timestamp is newer than our last successful visit. There is a separate, cruder fallback policy for listers that

publish no freshness information at all. ↩ back to the text

[29] POST 08 `swh-loader-git` computes the refs it lacks and hands them to git's own pack negotiation, so the server sends only objects the archive does not already hold. ↩ back to the text

[30] POST 08 **IndexNow**, a push protocol published in 2021 by Microsoft and Yandex. The fan-out is the protocol's own commitment, in its own words: "Search engines adopting the IndexNow protocol agree that submitted URLs will be automatically shared with all other participating search engines" — indexnow.org/documentation. That is one notification per change rather than one per engine, which is the part the 1998 draft did not have: there, every robot registered separately with every server. The participating engines are listed by the protocol itself at indexnow.org/faq — Amazon, Bing, Naver, Seznam.cz, Yandex and Yep. Google is not among them, and that absence is the whole point: the protocol exists, works and is deployed, and it still cannot make the largest crawler take part. ↩ back to the text

[31] POST 09 Drew DeVault, "Please stop externalizing your costs directly into my face", 17 March 2025, on crawlers arriving from tens of thousands of mostly residential addresses, in unrelated subnets, with user agents that blend into ordinary traffic. ↩ back to the text

[32] POST 09 SourceHut status page for the git.sr.ht incident of 17–24 March 2025, 168 hours 30 minutes, verbatim: "Note that Anubis is only being used for the web frontend. API access and git operations are unaffected." See status.sr.ht. ↩ back to the text

[33] POST 09 Directive (EU) 2019/790 on copyright in the Digital Single Market, Article 2(6). The operative exclusion is in the second subparagraph, which lists "open source software-developing and-sharing platforms" among the providers that are *not* online content-sharing service providers — and it is that definition, not any exemption inside Article 17, which keeps the filtering obligations away from code hosting. The phrase quoted above is the wording of recital 62, which reads "open source software development and sharing platforms"; the article itself hyphenates it. Both are in the Official Journal text of 17 May 2019. ↩ back to the text

[34] POST 09 Roberto Di Cosmo, "Saving software development from the European copyright reform", 17 April 2019, written two days after the directive was approved. Verbatim: "one year and half of a hard and exhausting work of explanations and mobilizations, at national and European level, to which I have actively contributed, has led to a clear exclusion formulated in Article 2(6) for open source developing and sharing platforms". The post cites Article 2(6), whose wording is the hyphenated form; the phrase quoted in this post is recital 62's — see the previous reference. ↩ back to the text

[35] POST 10 Software Heritage sysadmin issue #5304, "Block agressive scraper", 10 April 2024. The tracker is public and readable without an account. ↩ back to the text

[36] POST 10 Sysadmin issue #5712, 19 September 2025 — the legacy forge returning 403 to everything, described in the ticket itself as a quick and dirty action to stop the crawlers. The user-agent table quoted elsewhere in this series is issue #5439, 26 September 2024, from one day of logs. ↩ back to the text

[37] POST 10 The Anubis configuration landed in the infrastructure repository on 22 January 2026 and was enabled on the web frontend on 30 January. ↩ back to the text

[38] POST 10 Issue #22 on the `updateswh` browser extension, 2 February 2026: reported at 09:01 UTC, the cause guessed by someone else at 09:02, confirmed the same morning, fixed the same afternoon. The fix commit's own parenthesis is quoted in the post. It broke again on 23 April 2026 when the rule was tightened, and was fixed again the same day. ↩ back to the text

Corrections

Every correction made after publication is logged here, with its date and what changed. Nothing is silently edited. If you find an error, the fastest way to have it fixed is to tell me.

— nothing logged yet —

---

www.dicosmo.org/good-enough/ — the version of record
Roberto Di Cosmo · dicosmo.org

Text licensed CC BY 4.0. Quoted material remains its authors'.

This page loads nothing from any other host: no fonts, no scripts, no analytics.