Keep your META where your DATA is. In the filesystem?

Peter Bubestinger-Steindl

`(p.bubestinger@ArkThis.com)`

November 2023

Speaker notes

No notes on this slide.

What’s this all about?

Metadata- and File-Wrangling.
Professional and Private.

Speaker notes

No notes on this slide.

My “AHA” Effect…

- Simply making use of “name=value” tags;
- **Moving metadata to the same level as a file or foldername;**
- Keeping the META where the DATA is.
- Allow using the filesystem directly as database.
- All FOSS and standard and awesome, of course.

Speaker notes

The idea for this first came to my mind, on April 21 in 2022, while I was preparing 2 presentations for the National Archives of Singapore (by INA):

- Large scale archival data storage.
- Digital asset management systems.
- Including workflows and reality checks.

There was my first “AHA” Moment: This may be a key to building blocks towards data-“enjoyment” (rather than Management): Simply by dissolving the distance between the database and the described objects.

By simply converting both of them into filesystem “Data Objects”. Then go along with that new box of old, but now “the new default” options. Enjoy.

Why bother at all? I assume most of us here are already engaging in “wrestling” an increasing amount of “Digital Objects” in digital workflows - and systems all interconnected - and hopefully well-configured and working smoothly.

If you’re nodding now, you may feel happy to hear that there may be some change coming up: Object Storages.

What’s with the “Holodeck”?

Yes, I’m trying to be funny.
Yes, I’m very serious about this idea.

Speaker notes

A “Holodeck” is a fictional device from Star Trek which can generate any computer-generated images/objects, that allow to be interacted with “as if they were real”.

Short: A virtual holographic “anything”-simulator.

I’d already be happy if I could tag my personal photo collection - without having to worry about a lock-in commitment to any application that annotates my pictures. Or media files. Or documents.

Damn. We all need Digital Asset Management systems. Like glasses, when you get older: If you’re older than 25, you probably want/need one.

A cloud storage with “search by metadata/tags” is doing that already, btw.

By taking the step to using “Data Objects” (meta+data) as regular paradigm for handling our computing needs, we might seriously get closer to something as generic (and stable) as Star Trek technology 😇.

Speaker notes

No notes on this slide.

My Perception:

Current Project Desires:

“Make things all digital and awesome! And easy. In no-time.”

My Perception:

Speaker notes

From my point of view as an engineer, the wishlist/requirements for metadata-handling projects often are like:

“we’d like a holodeck. By August. Can you do that?”

Disclaimer

What I'm proposing is not a new technology.

I'm suggesting a different usage of what already exists.

And I apologize: I may be an ad for FOSS.

Speaker notes

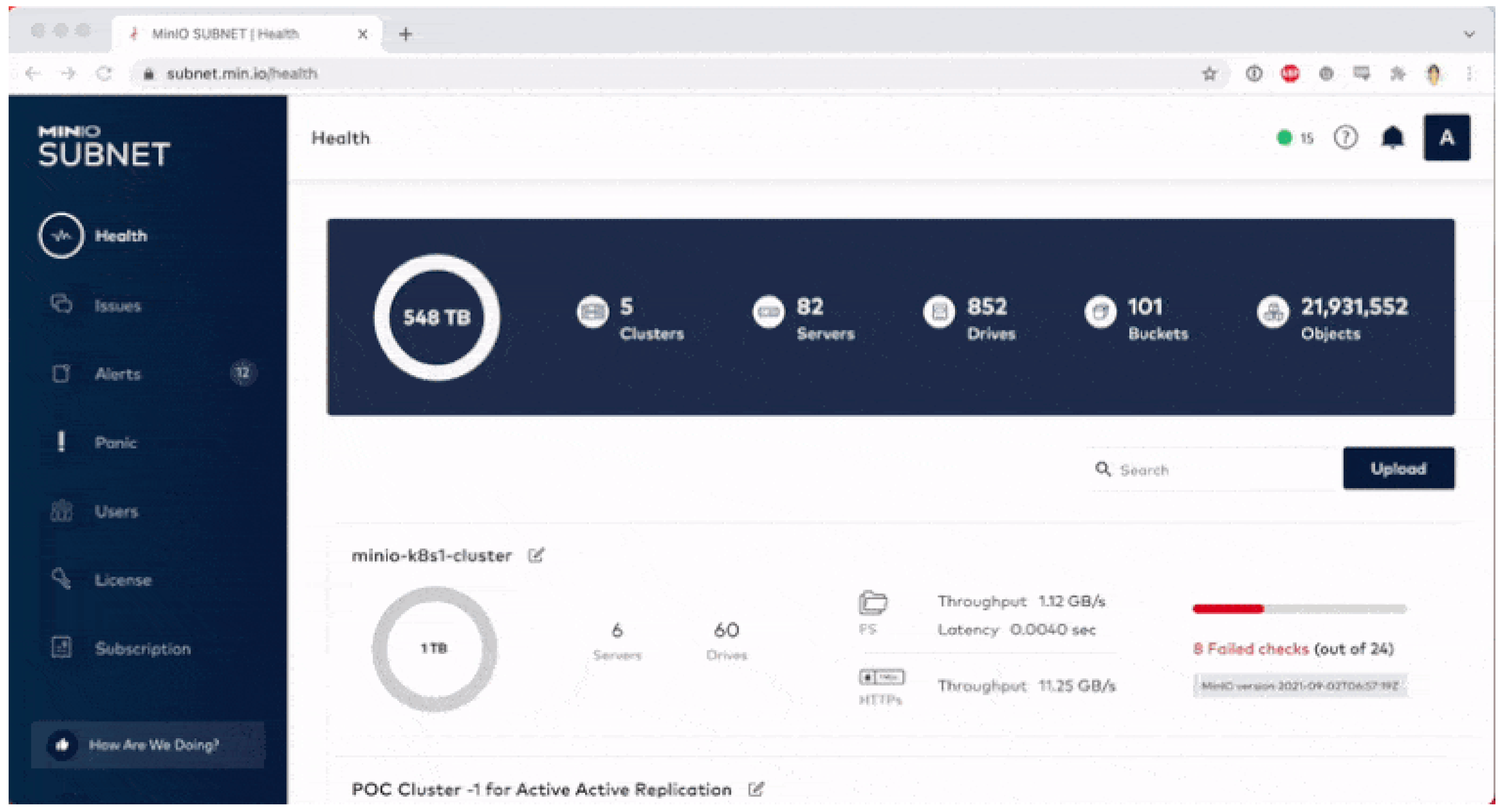
I'm a passionate engineer and inventor, not a salesman.

I hope I only "advertise" (or simply promote) things that are "in the spirit of FOSS" - and/or that I have personal, good experience with.

Damn! I do sound like an evangelist here ;) Don't listen to me. Please make up your own mind.

I make my living with paid FOSS services, and believe in commercial OpenSource - in order to be high-quality and sustainable. The perceived "quality" difference between proprietary and FOSS is merely due to their "slightly different" resources available.

Existing Components



- “Object Storages” already support

Speaker notes

Interested in discussing and exploring the possibilities and implications of having a FOSS-standard filesystem (local & network) which would hold our “assets” as digital objects: where data payload and/or metadata are linked and structured by design?

In this session, I’d like to invite all of you who’re dealing with search/find/tagging/structuring any kind of “digitial (file) collection” to engage in a possible “Making of Brainstorming” for a new computing paradigm which may solve many metadata/DAM issues we all know and eh… love?

Where to begin?

Speaker notes

No notes on this slide.

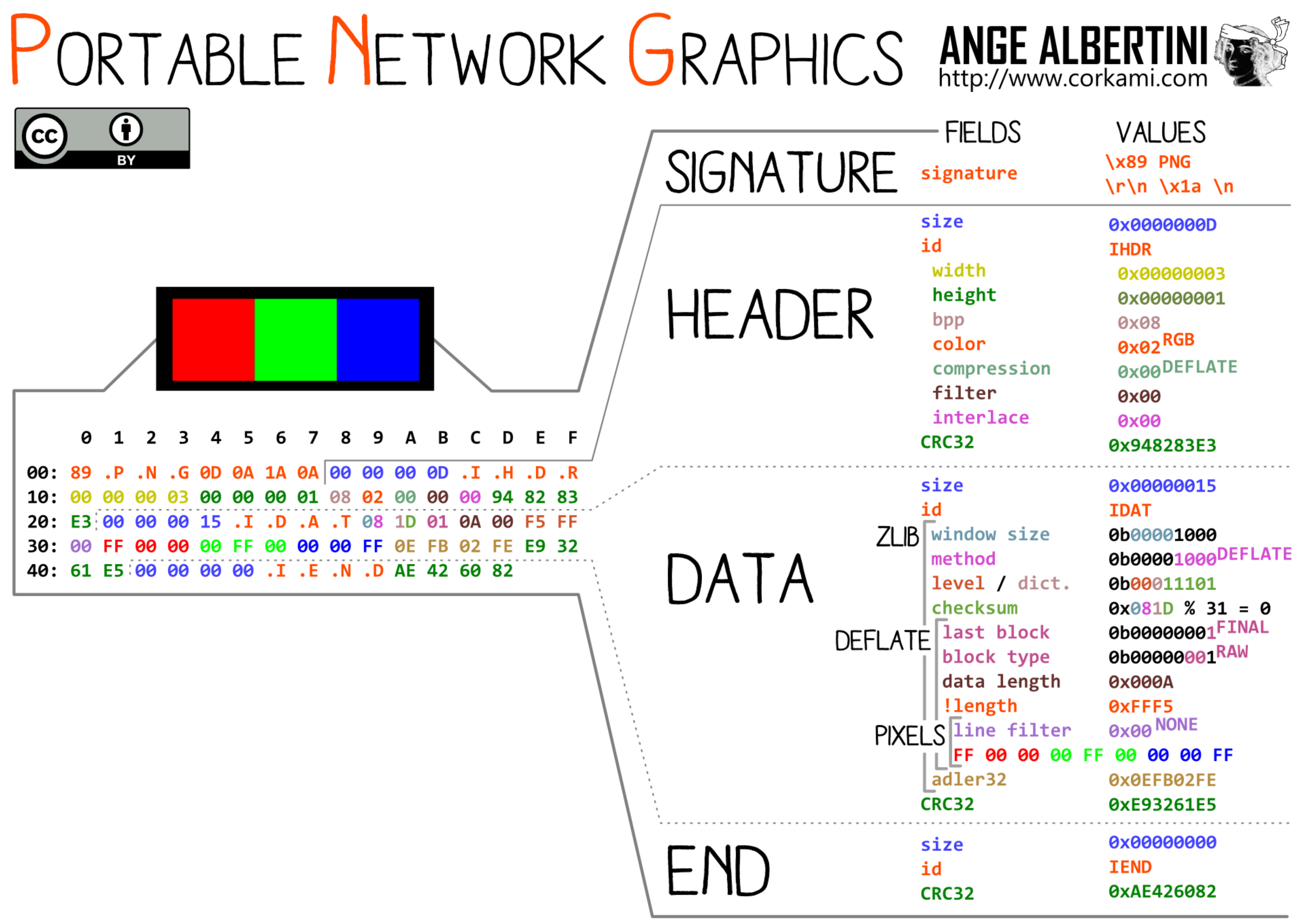
Terms

- Files/Folders?
- Metadata?
- Data “payload”?
- Data Objects?
- Filesystem?
- All clear?

Speaker notes

No notes on this slide.

Status quo: Assigning “a plain title”?



```
     0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F
00: 89 .P .N .G 0D 0A 1A 0A 00 00 00 0D .I .H .D .R
10: 00 00 00 03 00 00 00 01 08 02 00 00 00 94 82 83
20: E3 00 00 00 15 .I .D .A .T 08 1D 01 0A 00 F5 FF
30: 00 FF 00 00 00 FF 00 00 00 FF 0E FB 02 FE E9 32
40: 61 E5 00 00 00 00 .I .E .N .D AE 42 60 82
```

		FIELDS	VALUES
SIGNATURE		signature	\x89 PNG \r\n \x1a \n
HEADER		size	0x0000000D
		id	IHDR
		width	0x00000003
		height	0x00000001
		bpp	0x08
		color	0x02 RGB
		compression	0x00 DEFLATE
		filter	0x00
		interlace	0x00
		CRC32	0x948283E3
DATA		size	0x00000015
		id	IDAT
	ZLIB	window size	0b00001000
		method	0b00001000 DEFLATE
		level / dict.	0b00011101
		checksum	0x081D % 31 = 0
	DEFLATE	last block	0b00000001 FINAL
		block type	0b00000001 RAW
		data length	0x000A
		!length	0xFFF5
	PIXELS	line filter	0x00 NONE
		FF 00 00 00 FF 00 00 00 FF	
		adler32	0x0EFB02FE
		CRC32	0xE93261E5
END		size	0x00000000
		id	IEND
		CRC32	0xAE426082

See: “Keywords and Text Strings” (PNG Specification, W3.org)
And: “What software can I use to read png metadata?”

Speaker notes

This is just a random, yet not unpopular example: Ever tried to assign a “title” (plain text) to a PNG file?

It is possible, and the specification allows it - yet, only a few applications support metadata for PNGs. Hardly any application displays is, and even less support editing it. Even though it’s such a popular file format.

Every file format currently has to be treated differently. And some file formats simply do not have means for storing any metadata - or the information you want/need.

If we already were in an Object Storage environment, this would be trivial: For users as well as developers.

Speaker notes

No notes on this slide.

The Unix Philosophy

“Everything is a file.”

See: https://en.wikipedia.org/wiki/Unix_philosophy

If we translate files to Objects, it becomes…?

The AHA-Holodeck Philosophy

“Everything is an Object.”

- Anything can have *its* metadata with *its* payload content.
- As simple a Lego and to-be taken for granted like filenames.
- The filesystem serves as (semantic-graph-) database.

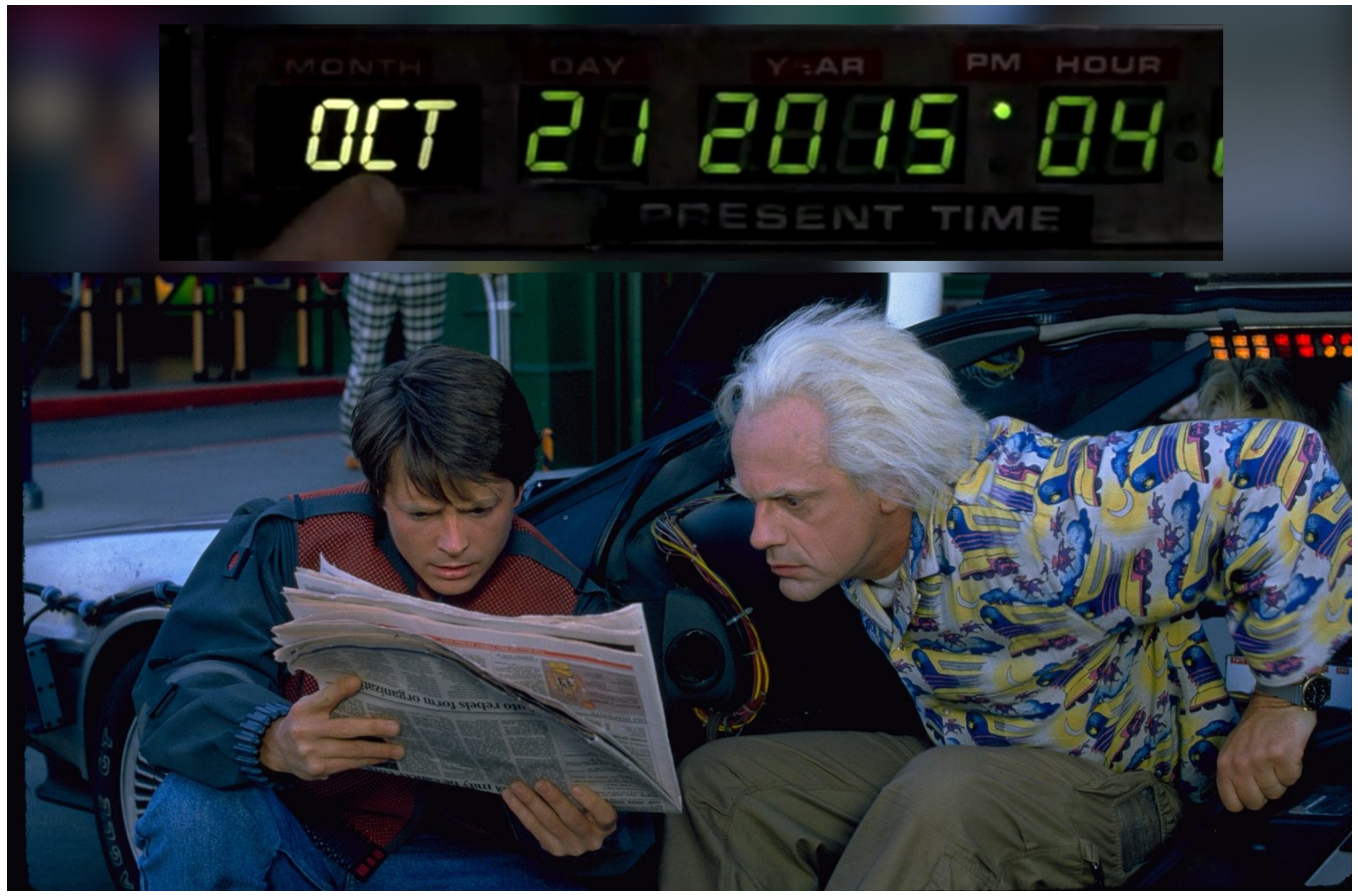
What if you could simply…?

- store (and use) **any metadata** reliably/persistent *with* its payload content?
- drag-n-drop, copy/paste, convert, view, edit **any metadata**: In any file manager or tool?
- import/export catalog (database) entries, like copy/pasting files?

Speaker notes

Because there is no technical difference between a metadata-catalogue entry and a data file anymore? 😮

What if...?



- that was as easy
  as using

Speaker notes

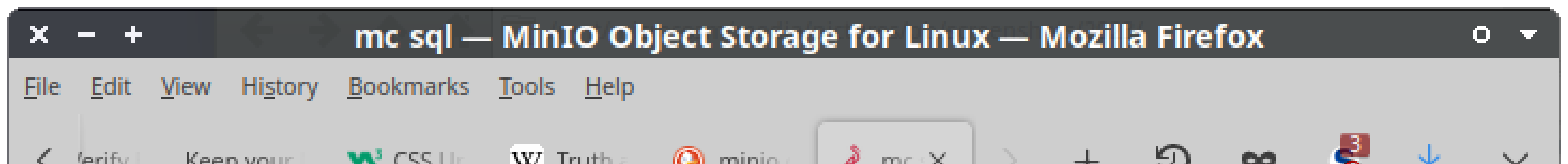
No notes on this slide.

Speaker notes

Except for the last point, this is already implemented in existing Object Storage systems.

What if...?

- you don’t have
  to worry/think



Metadata-only Objects = Catalog entries

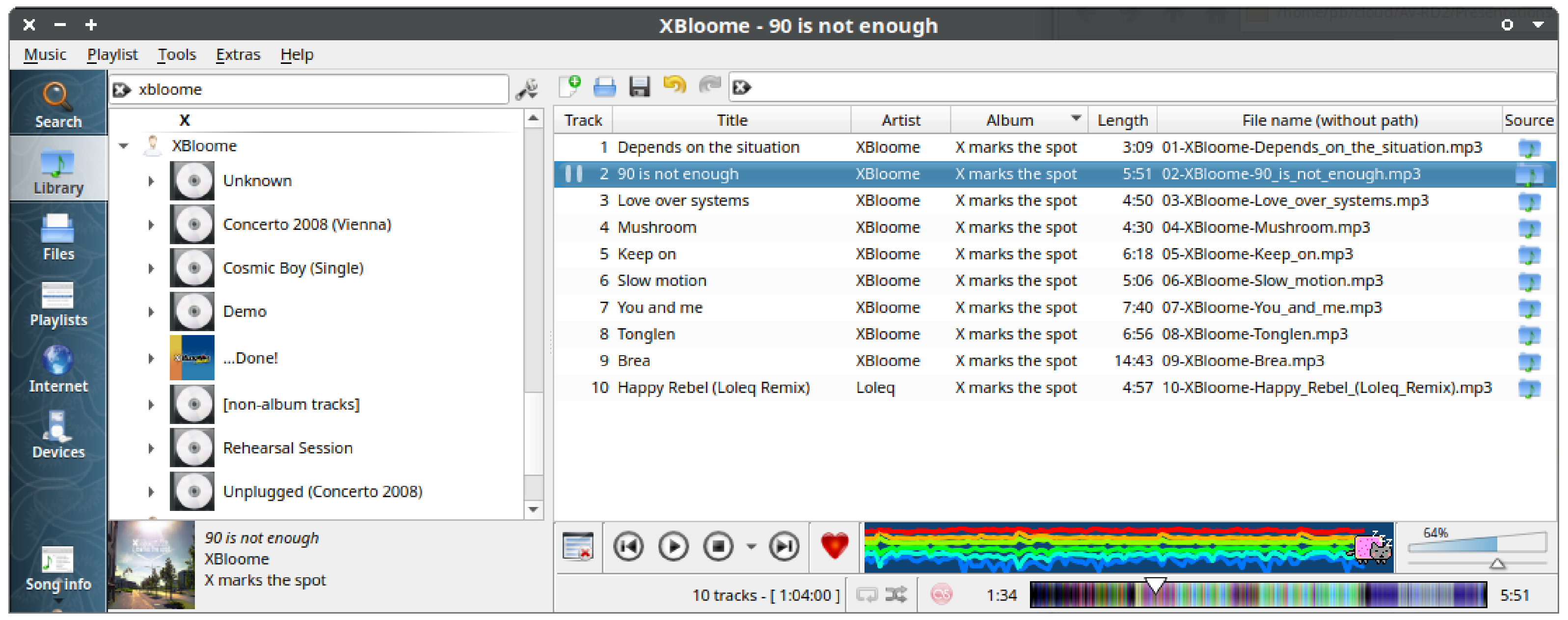
- Right-click “New
  Object”:

Speaker notes

No notes on this slide.

Interoperability of Features

- Any music/photo/collection application becomes (even (more)) compatible.
- Metadata is transformed on access/demand.
- Metadata exchange and usage between systems is facilitated.



Track	Title	Artist	Album	Length	File name (without path)	Source
1	Depends on the situation	XBloome	X marks the spot	3:09	01-XBloome-Depends_on_the_situation.mp3	
2	90 is not enough	XBloome	X marks the spot	5:51	02-XBloome-90_is_not_enough.mp3	
3	Love over systems	XBloome	X marks the spot	4:50	03-XBloome-Love_over_systems.mp3	
4	Mushroom	XBloome	X marks the spot	4:30	04-XBloome-Mushroom.mp3	
5	Keep on	XBloome	X marks the spot	6:18	05-XBloome-Keep_on.mp3	
6	Slow motion	XBloome	X marks the spot	5:06	06-XBloome-Slow_motion.mp3	
7	You and me	XBloome	X marks the spot	7:40	07-XBloome-You_and_me.mp3	
8	Tonglen	XBloome	X marks the spot	6:56	08-XBloome-Tonglen.mp3	
9	Brea	XBloome	X marks the spot	14:43	09-XBloome-Brea.mp3	
10	Happy Rebel (Loleq Remix)	Loleq	X marks the spot	4:57	10-XBloome-Happy_Rebel_(Loleq_Remix).mp3	

Speaker notes

Music library applications:

- Keep their cache/index (in a DB).
- May store tags beyond the capability of the underlying audio file formats.
- May now store *all tags* with the audio Objects - on the filesystem level.

Storing these custom tags as Object metadata, makes it possible to transfer music "Objects" to

Embedded metadata?

What is the use-case for embedding any metadata?

So the meta stays with the data!

Speaker notes

- Copy embedded metadata to the filesystem level?
    - Which impact does this have on the choice/difference of file formats?

Media container formats?

What is the use-case for (media) container formats?

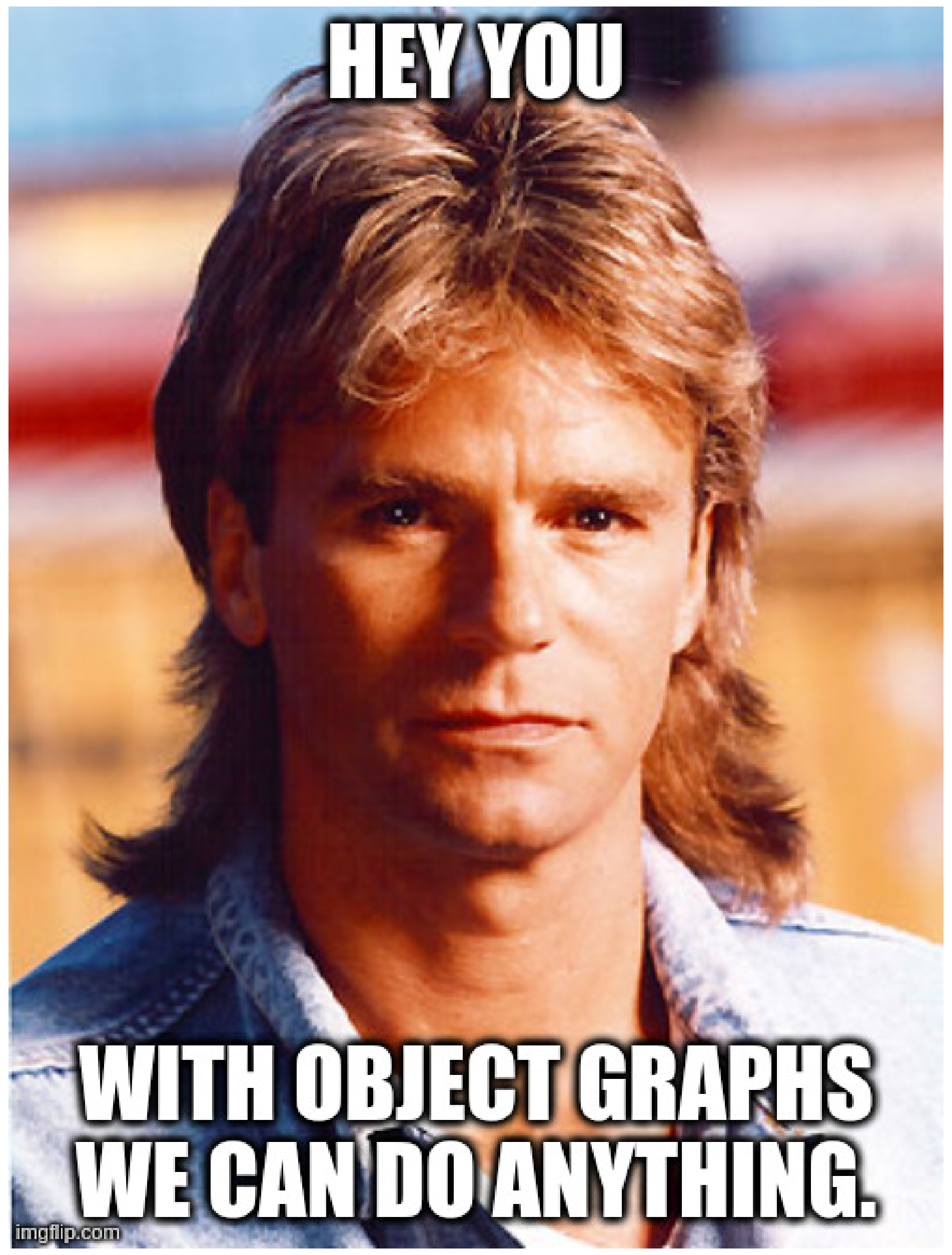
So related data stays together!

Speaker notes

- Resolve container formats into its components as Data Objects?
  (entry Object, video stream(s), audio stream(s), subtitle(s), container-metadata)
    - You may add new streams simply by drag-n-dropping a newly related Object. (Like a new language audio track)

Performance?!

- Size?
- Speed?
- Interoperability?
- MacGyver-able?



Speaker notes

Size: I don’t think that’s a problem.

1. It’s Metadata. With todays “default” storage capacities: trivial. The payload is currently stored already - so that has to work anyways.
2. The Metadata is also currently stored (in databases/files, etc).
3. Even if you keep/accumulate Metadata for a longer period: It’s still Metadata(-sizes).
4. Okay… I propose storing metadata as binary-proof UTF-8 encoded strings/text. Even numbers. For starters. That might increase storage demands by “blowing up” long numbers (16, 32, 64 word-size (bits), for example) into “way larger” strings. Yet, if we can “afford” this “waste” of digital storage space, imagine you could see/read any information as plaintext by default.

Or maybe the metadata layout can indeed by declared like a programming code Object (instead of a SQL-table design). Like “Class” definitions per Data Object Types? Then numbers could be integer/long/etc data types. And strings could even be active methods. ;)

Speed: Speed could indeed be a major factor.

However, I still think that any performance “Einbußen”, compared to now are still worth investigating on how to make things faster.

For now, I propose the same technology that is used for indexing websites (and even locally stored files - in different formats). All this already exists, and is widely in use - even by “smaller” websites. So it can’t be that hard. And hardware is still cheaper than manpower and life- and braintime.

Also, any application that has its own kind of “library” database/config stored somewhere, is already performing these tasks sufficiently. Initial indexing of existing contents takes a while, but then, only changes are monitored - and that doesn’t even bother nowadays “average” computer systems.

Maybe it’d drain a bit more on battery-powered systems. But hey, I’ve heard they’re working on “batterifying” everything by 2020. Or some other horizon.

On the other hand: Considering, that with Data Objects, less individual code-libraries will need to be used (to access metadata, and provide library-functionality, etc). And since this filesystem provides basic functionality by default, there’d be less code necessary to run (and use resources).

How large the impact of this could be, and how far in the future, the coverage and support to speak of “real world tested” is unclear at the moment.

Interoperability:

Would be greatly improved. I’m quite serious about this. Most of my work in the last 20+ years was: Making things interoperable, and improve upon existing technologies and possibilities.

With files becoming Objects - and Metadata “accumulating” along its payload, becoming well-annotated (semi-automatically) and therefore (more) self-sustaining and easier (re-)usable.

Where to begin (implementing this)?

- MinIO?
- NoSQL / MongoDB?
- Search Indexers?
- …More ideas?

Speaker notes

Object Storage (MinIO, S3, …) can already store/query metadata.

What to feed it with?

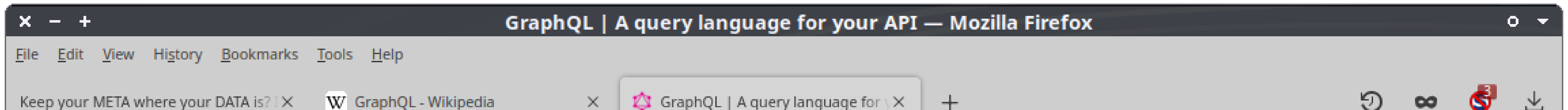
- Real-world collection (web-)access copies.
- Corresponding data (catalog) entries. (XML, JSON, CSV, etc)
- …More ideas?

Speaker notes

No notes on this slide.

The rabbit hole goes deeper.

But that’s a story for another time…

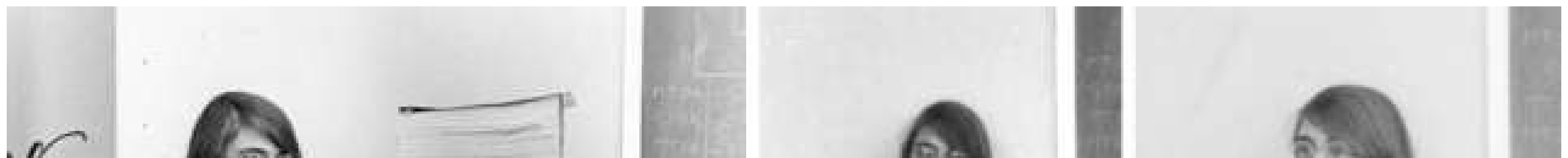

GraphQL | A query language for your API — Mozilla Firefox
File Edit View History Bookmarks Tools Help
Keep your META where your DATA is?
GraphQL - Wikipedia
GraphQL | A query language for

Speaker notes

No notes on this slide.

If such an implementation/prototype is not awesome...

...then it's not what I suggest here 😎
(Or simply not finished yet)



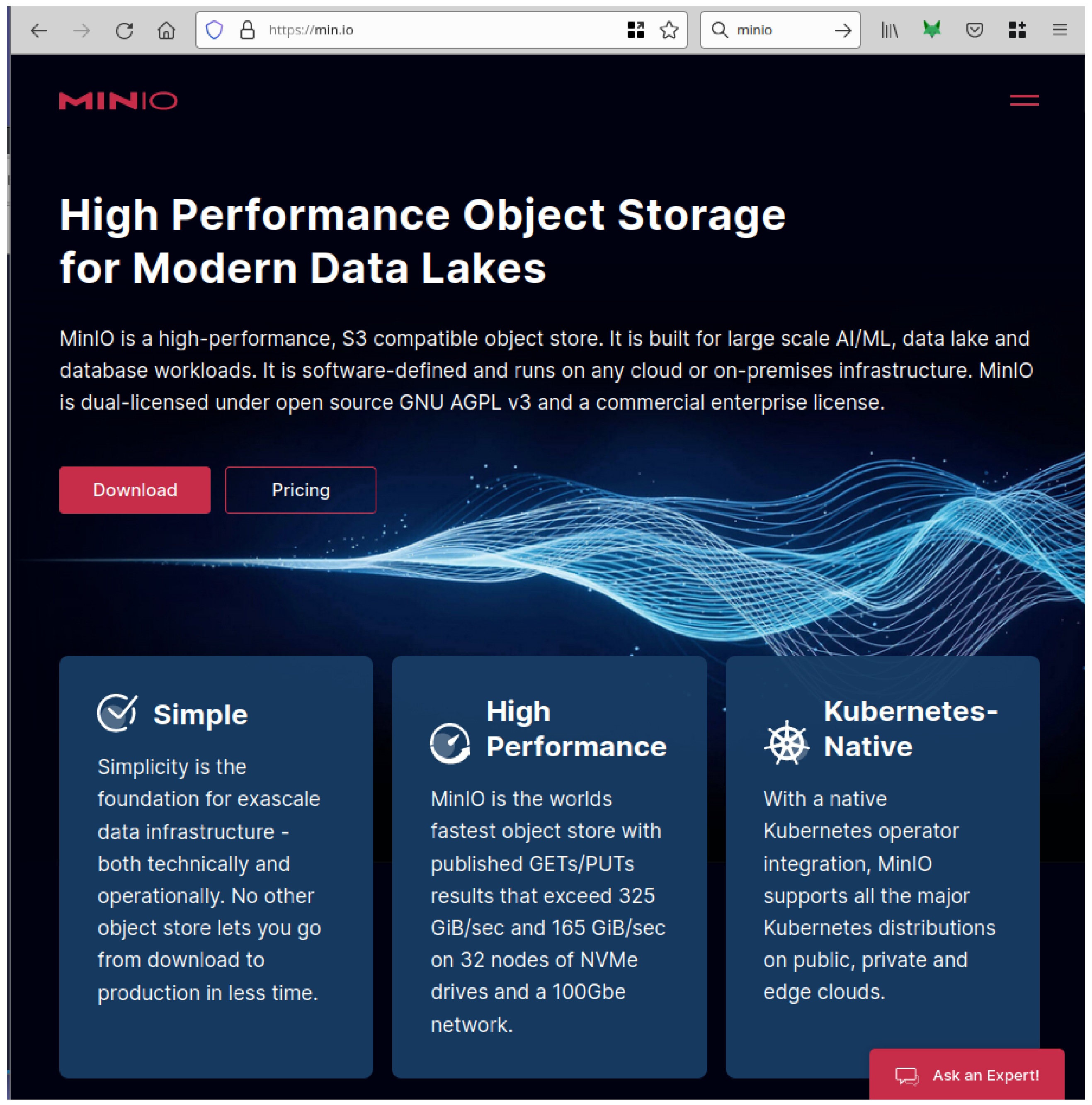
Speaker notes

https://phys.org/news/2016-08-image-margaret-hamilton-apollo-code.html

Oh, btw:

- IMO, **we all will sooner or later use Object storages,** because hierarchical filesystems don't scale well enough anymore
  (with today's use cases and sizes)

- So you'll have the underbelly to support metadata-with-payload Objects anyways 🤩



High Performance Object Storage for Modern Data Lakes

MinIO is a high-performance, S3 compatible object store. It is built for large scale AI/ML, data lake and database workloads. It is software-defined and runs on any cloud or on-premises infrastructure. MinIO is dual-licensed under open source GNU AGPL v3 and a commercial enterprise license.

Simple
Simplicity is the foundation for exascale data infrastructure - both technically and operationally. No other object store lets you go from download to production in less time.

High Performance
MinIO is the worlds fastest object store with published GETs/PUTs results that exceed 325 GiB/sec and 165 GiB/sec on 32 nodes of NVMe drives and a 100Gbe network.

Kubernetes-Native
With a native Kubernetes operator integration, MinIO supports all the major Kubernetes distributions on public, private and edge clouds.

Speaker notes

It's already called "Modern Data Lakes". So we're going fishing wish Objects - by search queries.

Ideas? Input? Questions?

Please! Go ahead. Now and later :)

Peter Bubestinger-Steindl

`Peter@ArkThis.com`

https://github.com/ArkThis/AHA_ObjectWorld/
https://diode.av-rd.com/nextcloud/index.php/s/z2M4JZY8RFt8Nnd

Speaker notes

By proposing serious usage of Metadata on any kind of Digital Object (Collection) - from personal to professional, small to large scale:

We archivists can now pimp the computing world, by applying our decades of real “META-EXPERIENCE FU” knowledge and skills.