A Holodeck by August - Proposal 1

Peter Bubestinger-Steindl
(peter @ ArkThis.com)

2024-08-27

Project Plans

- Solving GLAM daily issues by shifting to a modern (big data) tech-stack and filesystems.
- 3y funded project.
- 3 software developers
- 2 dev-ops/sysadmin
- 1 mastermind with a roadmap for a useful working prototype

Outcome:

- User Interface (UI) tools for catalogue needs on filesystem-level.
- Practically approved suggestions for meta+data structuring with Data Lake Object capabilities.
- Tutorials, presets, profiles, etc - for common use-cases of data collection handling: using filesystem as DB.
- Blueprint for professional GLAM storage (network).

Addresses the following issues

- spreadsheets for collection/data annotation.
- file-format specific, embedded metadata.
- store/query/retrieval of any digital information.
- interoperability, migration, updates, obsolescence.

A Holodeck by August: Resolving annoying daily GLAM issues by switching to a Big Data tech-stack.

(**GLAM**: Galleries, Libraries, Archives and Museums)

Until 1995, a filename (in DOS) may only consist of:

- Uppercase-ASCII characters only (no SPACE)
- maximum 8 characters as file/foldername
- and 3 for the filetype.
- only 1 dot.

Imagine how much all users and developers have been empowered simply by "allowing" long filenames on popular filesystems?

> **The usage of different data-handling paradigms, proposed in this document, may be comparable to allowing 250+ characters in 1995. No more need for a spreadsheet or database for most daily needs: Simply right-click-edit-metadata.**

Now hardly anyone remembers that "long filenames" have not always been around.

Technologies have evolved, which allow completely different ways of data-handling.

There exist technologies developed since 2001, originally intended for proper meta+data handling - instead of relying on file/foldernames - and later "Big Data" processing and scaling demands: These tools and setups provide generic "processing workflows" that scale out by hardware - not manpower.

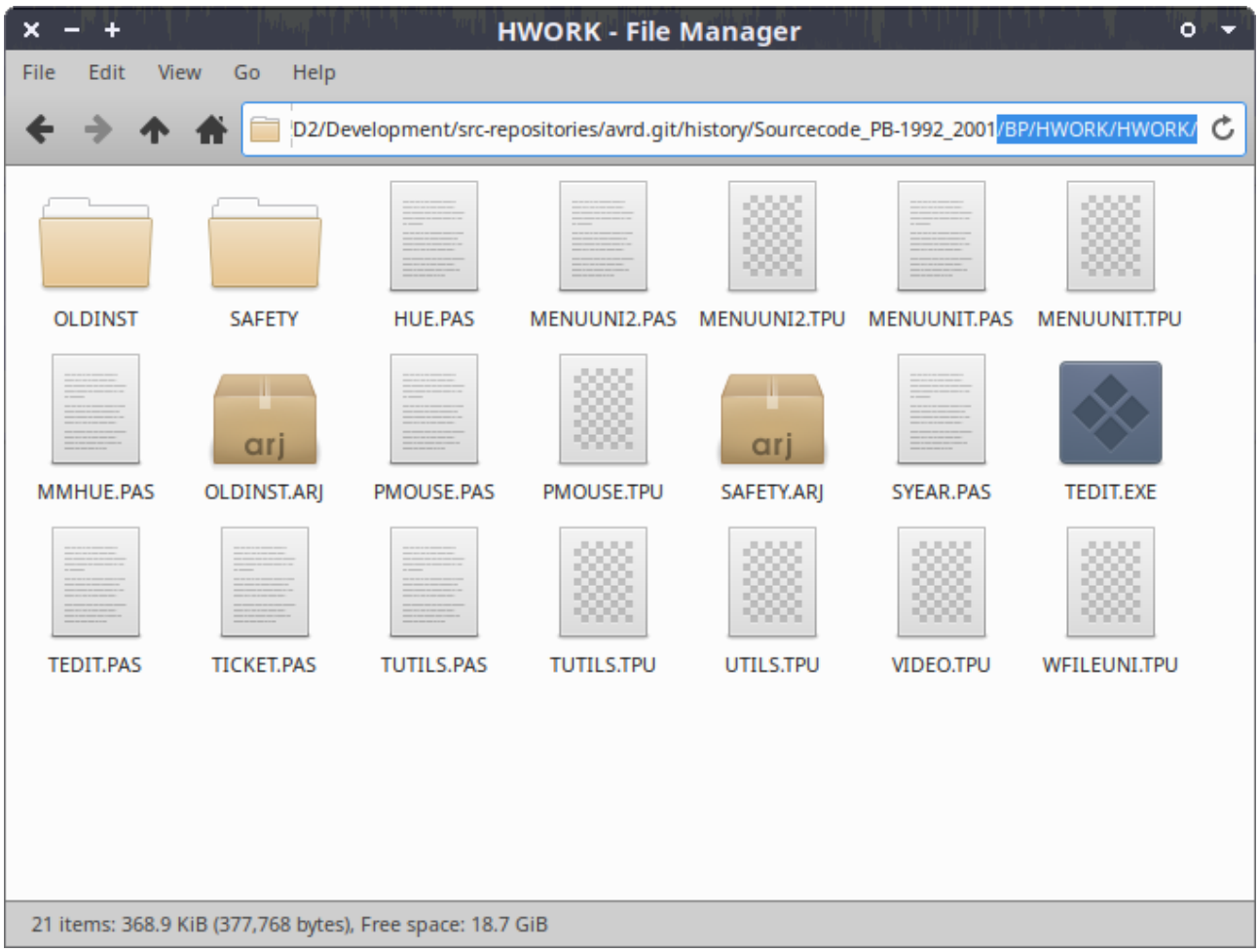

Figure 1: Real world “8.3” example

The current situation for *any* digital data collection is problematic, and stuck in a neverending migration of data and tools: **The reason is that the files-in-folders paradigm is at its end.**

This is where "Big Data" comes to play:

They have realized the "files-in-folders paradigm limits", and provided working options for making a folder structure optional - and storing "Files" as Objects (with Metadata). Each Object has a technical identifier - and an arbitrary number of key/value data with it.

Access (search and retrieval) for usage is done by querying indexers that have cached the filesystem data. Merely for fast(er) access. This allows real-time typing-suggestions, etc.

> Being able to store and access (for use) any "file content" as kind of "Data Object" is like introducing long filenames back then: Giving the average user (and any developer) the option to "simply" provide right-click-edit-metadata capabilities.

And simply ask for "keywords/tags" or provide "filename/foldername" as 2 metadata fields when "saving your work". Imagine handling your files like your browser bookmarks: "save" - then use tags and fulltext search.

This already works out of the box on POSIX systems (since 2003).

Common storage configuration and network

The storage design proposed for this project, allows a blueprint "Linux Package" to install and setup a GLAM-Object Storage Node in less than 30 minutes. An easy and well-known-procedure, like setting up a Linux Webserver after 2000.

This storage may be accessed by Windows/MacOS machines, too.

This will allow to have common hardware/software/config setups for storing data professionally. And interconnect those storages (if intended) to a storage-network of this kind.

Because it is based on Cloud-Stack components, and therefore scales identically. It also allows to access metadata or data in the same way (programatically) - and at comparable performance, even on lower-end or simply older hardware.

The outcome of this proposed project will be to orchestrate existing technologies, mainly borrowed from Big Data and Open Standards environments: **resulting in a basic software environment that allows curating digital collections in a new computing paradigm: reducing interoperability issues and migration efforts (costs)**

This new computing paradigm is basically comparable to what online-service providers are using large-scale productively since around 2005. It goes beyond what Cloud Providers currently offer:

By adding our GLAM experience in handling all kinds of data under all kinds of "epochs" to the current change of digital data-handling usage, we can introduce proper asset-management means based on "filesystem LEGO blocks".

As basic as "put it in the filename". Now without the need to choose a folder, and with the basic option to "add keyword".

That's it.

Cloud features everwhere: even local.

There are certain search/find/annotate/relate features that basic Cloud storage providers offer users already: Tag your images, link to something, search/query/filter - instead of "walking directory paths".

Just hit "save" - and it's going to be "stored somewhere". All you need to provide is something you remember to find it again.

> **The reason why online Cloud services can provide these features is that they are based (internally) on a completely different technology stack than even the most professional "state of the art conventional database+filesystem" applications.**

Since professional "Big Data" Cloud tech-stacks are now available as paid-support FOSS, such a feature set can now easily be deployed locally too. From a single-disk-single-host to an on-premis storage cluster - and if desired, possibly connected to a geo-replicated Memory-Institution Storage Cloud.

And you get all the tag/search/find Data Object features for your local, offline works too.

Software components

The key components this project will focus/base on, are:

- Metadata capabilities of filesystem/OS (key-value store)
- Object storage (as successor of files-in-folders)
- Data Lake(house) handling
- Search/Filter caching & indexing
- External, common (standard) lists for controlled vocabulary/facets.

Leading to:

- A generic set of building blocks and tools to work with arbitrary digital collections.
- A resolution of most interoperability issues (file-formats, metadata-layouts, interfaces, databases, etc) natively by the differnt new paradigm-properties.
- The ability to (re)use these new basic data-handling options anywhere (Standards: POSIX (xattr), Object Store (ANSI T10), etc).

All software components required for a working prototype which already covers many daily use-cases within GLAM (and even personal to professional tasks where one would use a DAM/DB or Spreadsheet application).

What is different about this approach?

There are still many (valid) reasons why everyone of us still puts "more information than they should" into a filename. And coming up with good-and-lasting folder (naming) structures is another quest on its own.

That we all "love filenames" as go-to metadata field is still very common.

This is IMO proof that the common filesystem is still the one "safe" option that *anyone* can easily access, view, edit - and somewhat understand.

> "There is a file with that name in that folder . . . "

Now it gets a bit trickier:

> "And the information about it is somewhere else."
>
> ". . . or burried inside. Usually binary. Can you read hex?"

Using the Filesystem as Database - and the OS as interface

This can be handled differently already:

On a Linux setup from 2020, storing metadata (key/value) information with files-and-folders, using common tools, common libraries: works out of the box (ext4), persistent even when being copied over the network (smb/cifs) onto ZFS - and back.

MacOS supports and uses these features themselves for more than 20 years - so it can be considered stable.

Then moving gradually from conventional "files-in-folders" storages onto Object Storage systems.

Online cloud already does it.

When asking younger users, how they deal with search/using/sorting their files - their answer is often:

> "I do not have any files. I have my data in the cloud."

And there they simply use a search-box and tags or links - fulltext search and filtering by default.

Due to the lack of comparable "basic" features "offline", many turn to rather having most of their data (private and professional) on some (cheap/free) online cloud provider networks.

This project plan includes means for bridging this gap, and having seamless filesystem-level interoperability of related-data-objects. Any format, any size, any number.

Having Meta-with-Data as basic option

Existing common filesystems have the ability to store metadata with files/folders since ~2003. This would cover the following use-cases out-of-the-box by the operating system and basic tools:

- annotate any key-value information regardless of file format.
- thereby replacing the need for embedding metadata.
- created related objects, by adding links by metadata key-values.
- replace most spreadsheet for annotation cases.
- replace most basic database (filemaker, access) cases.

This is not yet-another-asset-management-system with its own database and environment: This is an orchestration of existing tools and existing serious system-standards - using Big Data tech-stack components, documentation and support. Resulting in a working-blueprint prototype that provides a proper environment for handling of Data Objects - in the most interoperable and LEGO-like way.

It is merely like the step from 8.3 to 250+ characters - and Unicode later: Having Meta-with-Data anywhere. Right-Click-Edit-Metadata. Just like that.

And maybe in 10 years noone will remember it has ever been different? :P

Known issues / FAQs

Common questions are:

- Performance?
- Scalability?
- Hard/easy/auto to setup?
- Stable?
- Reliable? Gets lost on copy/zip/convert?
- Compatible with...?