CS3 MESH4 EOSC

Connecting European Data

Cubbit

A distributed use case

CS3MESH4EOSC has received funding from the European Union's Horizon 2020 Research and Innovation programme under **Grant Agreement No. 863353**.

Who am I

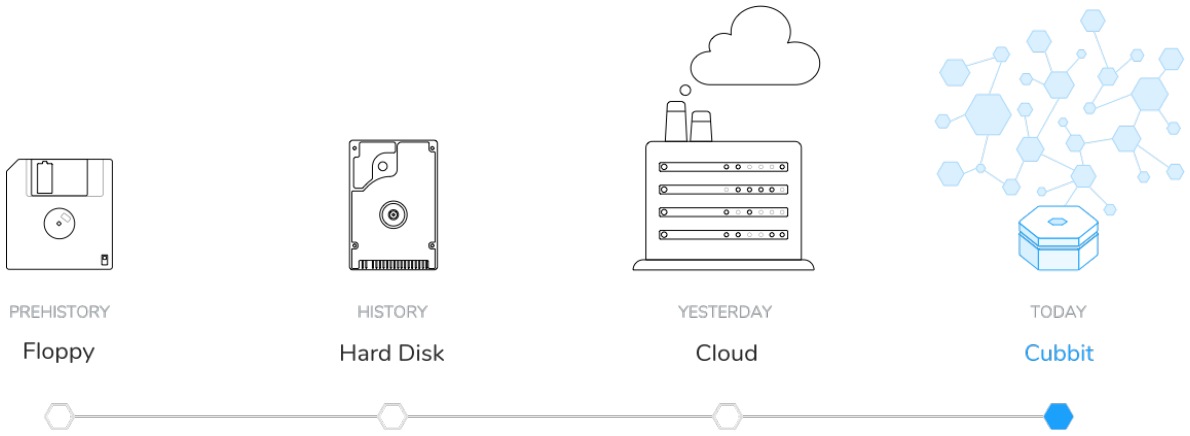
- Alessandro Petraro
- Master Degree in
  Software Engineering @ University of Bologna
- Software Engineer &
  Full Stack Developer @ Cubbit

Summary

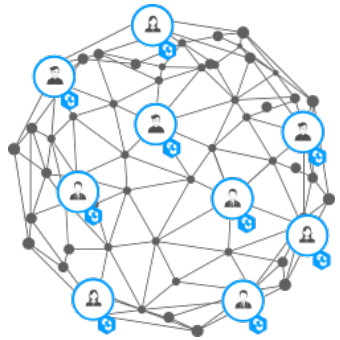
- Cubbit at a glance
- CS3 integration overview
  - What has been done
  - Next steps & challenges

Cubbit at a glance

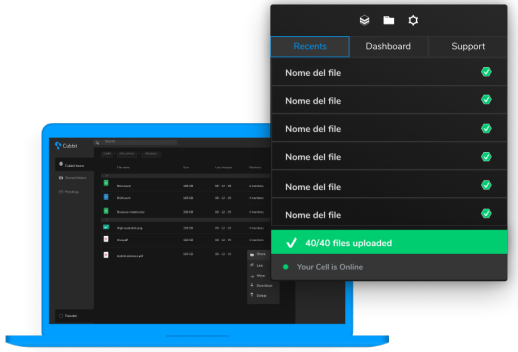
Cubbit Company Profile



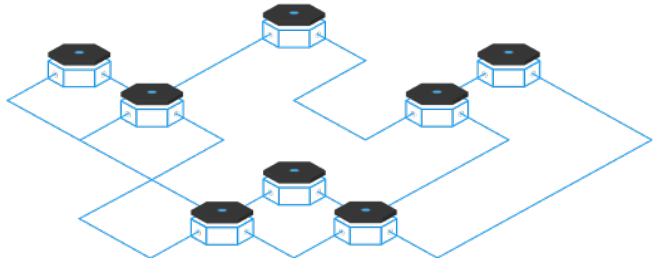
- World's first distributed cloud provider
- We recycle the internet we waste into the most accessible, green and privacy keeper web services



SMART COORDINATOR

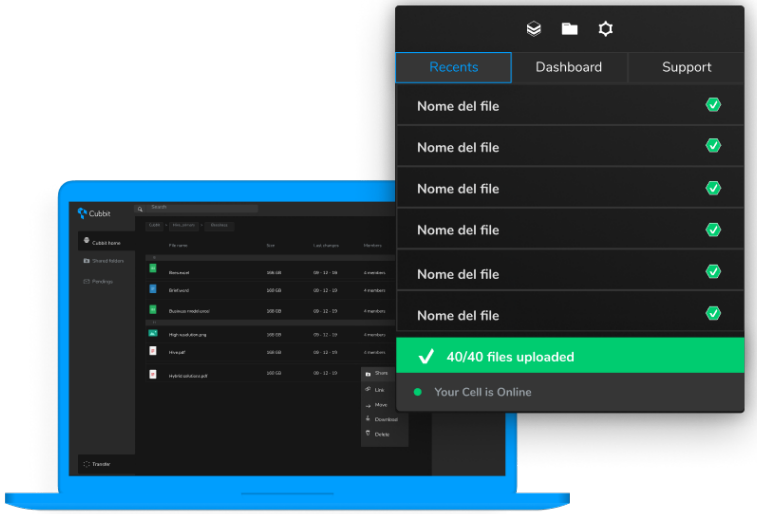


CLIENT



SWARM

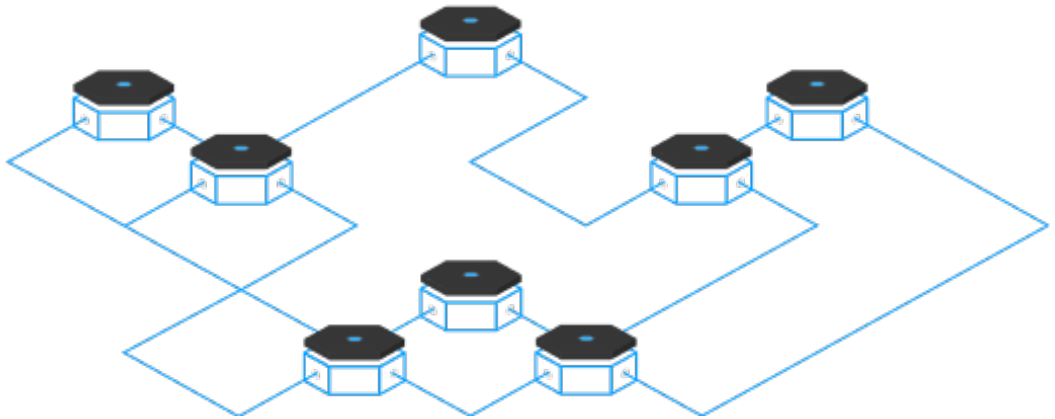
Cubbit client



- Cubbit client is a software (available both for desktop and web) with a “Dropbox-like” interface, designed to interact with the Cubbit distributed cloud storage.

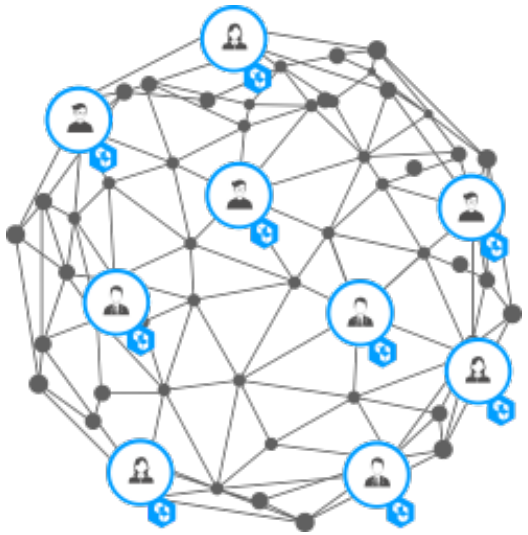
- It allows users to:
  - Claim and manage their devices
  - Backup, Sync and Share their files with friends and colleagues
  - Access their data anytime, from everywhere

The swarm



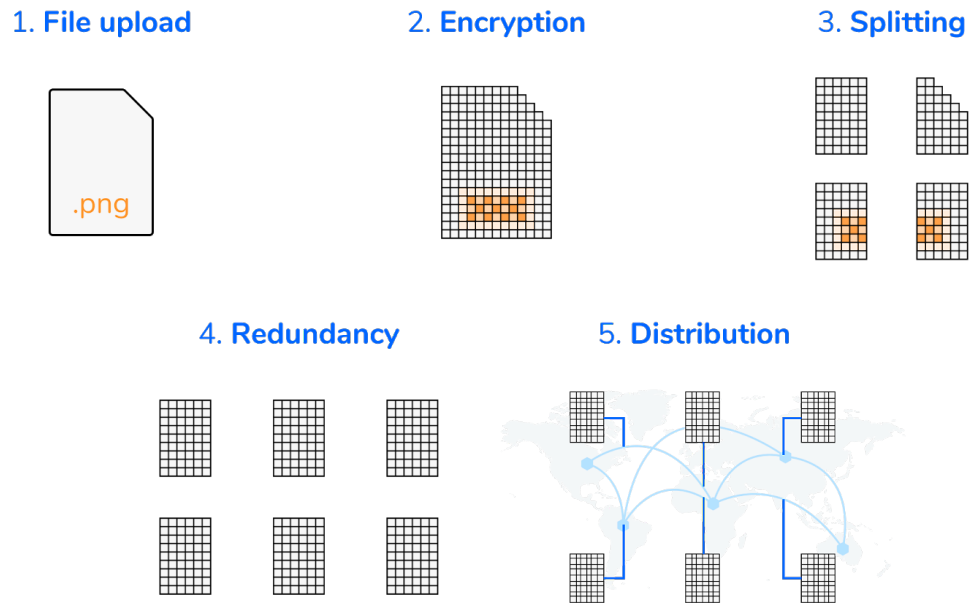
- Peer-to-peer: cells communicate with each other through p2p data-channels boosting up network performances.
- Distributed Redundancy: based on Reed Solomon error correcting codes. Ensures high availability while maintaining a low storage overhead.
- Recovery: the network is provided with a smart self-healing algorithm which recovers data automatically if needed

A central coordinator



- A smart super-node: a special node of the network that handles metadata and optimises the overall performances
- Optimisations: it employs machine learning algorithms to minimise latency while better distributing payloads across the swarm.
- Monitoring: it monitors the network to resolve congestions and trigger recoveries.

The path of a file



- Enhanced security: each file is encrypted with a randomly generated key. This key is never stored on any super-node accessible from Cubbit.
- Zero knowledge: our technology ensures that only the final user can access his/her own files.
- Transfer: the client splits each encrypted file in 36 encrypted chunks and spread them across the swarm

CS3 Integration

CS3 Apis Integration

.gitmodules ×

.gitmodules

You, a few seconds ago | 2 authors (You and others)

```
1  [submodule "third_party/cs3apis"]
2      path = third_party/cs3apis
3      url = https://github.com/cs3org/cs3apis.git
4
```

Proto build from script

```
$ yarn workspace @cubbit/ocm make:ocm
```

Proto definitions

https://www.npmjs.com/package/protobufjs

Revad Dockerfile

```
1   FROM cs3org/revad:latest
2
3   COPY ./config/cubbit.toml /etc/revad/revad.toml
4   COPY ./config/users.json /etc/revad/users.json
5   COPY ./config/ocm-providers.json /etc/revad/ocm-providers.json
6
7   COPY ./start.sh ./
8
9   EXPOSE 9999
10  EXPOSE 10000
11
12  CMD ["bash", "start.sh"]
```

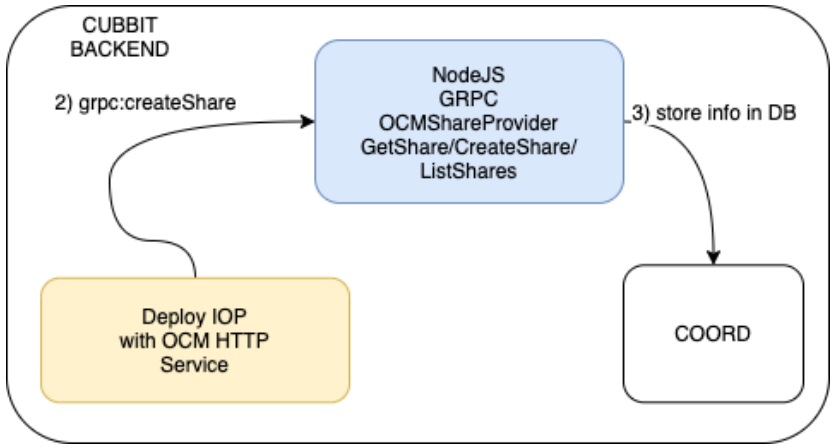
- We wrapped revad to make it more customisable
- `start.sh` is responsible for adding custom parameters and secrets

Example `start.sh`

```
#!/bin/bash

if [ -z $REVAD_JWT_SECRET ] ||
   [ -z $OCM_SERVICE_HOST ] || [ -z $OCM_SERVICE_PORT ] ; then
    echo "OCM service env variable have not been defined yet. Exiting..."
    exit 1
fi

echo "Configuring host: $OCM_SERVICE_HOST and $OCM_SERVICE_PORT for revad"

sed -i "s/{{REVAD_JWT_SECRET}}/$REVAD_JWT_SECRET/g" /etc/revad/revad.toml
sed -i "s/{{OCM_SERVICE_HOST}}/$OCM_SERVICE_HOST/g" /etc/revad/revad.toml
sed -i "s/{{OCM_SERVICE_PORT}}/$OCM_SERVICE_PORT/g" /etc/revad/revad.toml

echo "Configuring smtp: $SMTP_SERVICE_HOST and $SMTP_SERVICE_PORT for revad"
sed -i "s/{{SMTP_USER}}/$SMTP_USER/g" /etc/revad/revad.toml
sed -i "s/{{SMTP_PASSWORD}}/$SMTP_PASSWORD/g" /etc/revad/revad.toml
sed -i "s/{{SMTP_SERVICE_HOST}}/$SMTP_SERVICE_HOST/g" /etc/revad/revad.toml
sed -i "s/{{SMTP_SERVICE_PORT}}/$SMTP_SERVICE_PORT/g" /etc/revad/revad.toml

/go/bin/revad -c /etc/revad/revad.toml -p /var/run/revad.pid

```

- Waiting for K8S environment variables
- Replace secrets and environment variables
- Start reva

High level architecture



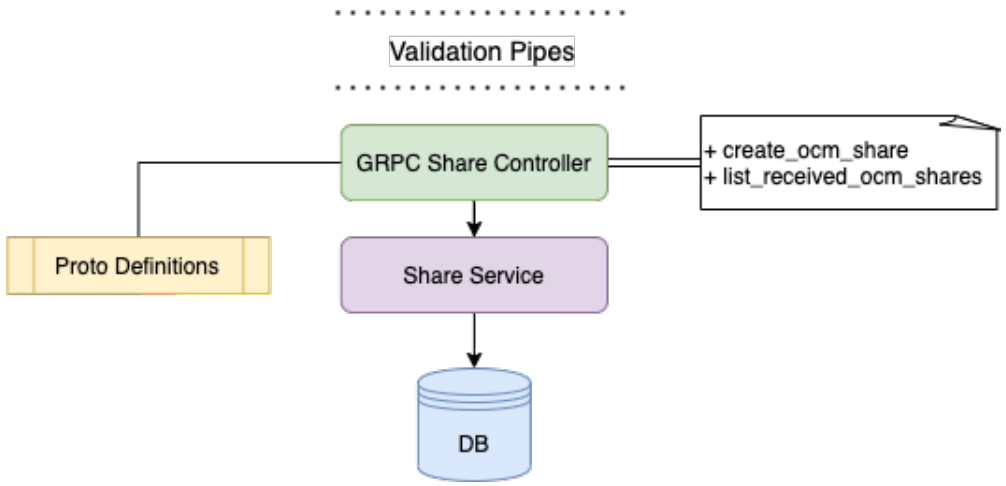
- Each call received from the IOP is forwarded to our GRPC controller
- Shares are then validated and saved into the database
- Coordinator micro service allows OCM service to reach the Cubbit network

Nest JS

- A progressive Node.js framework for building efficient, reliable and scalable server-side applications
- Three main building blocks
  - Module
  - Controller
  - Service

https://nestjs.com/

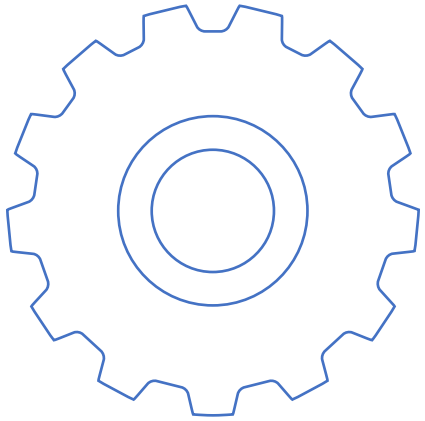
OCM Service Architecture



- Validation pipes parse the input
- Controller is responsible for building the response
- Service is responsible for the business logic

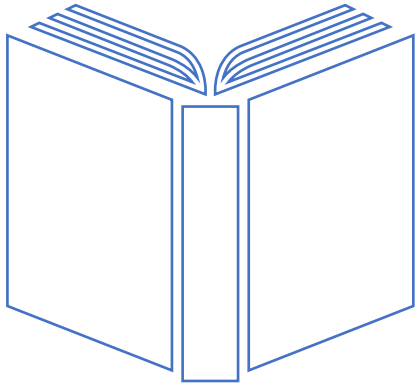
Next steps & challenges

Cubbit apis design



- **/files (POST):** upload a file to the Cubbit network (encryption and redundancy as an option)
- **/files/{file_id} (GET)**: download a file with the specified file_id from the Cubbit network (optional key if encrypted)

Next Steps: Cubbit translator



- Create a SDK that can be used to call the Cubbit CORE API
- Cubbit CORE responsible for:
  - Upload / Download files
  - Encryption and redundancy

CS3 MESH4 EOSC

Connecting European Data

Thank you!

Discover more on...

cs3mesh4eosc.eu

company/cs3mesh4eosc

@cs3mesh4eosc

CS3MESH4EOSC has received funding from the European Union's Horizon 2020 Research and Innovation programme under **Grant Agreement No. 863353**.