iRODS as back-end in a data transfer and exchange application

Claudio Cacciari

SURF

29 June, 2026

Neptune

A tool to orchestrate data transfer between

- **different data providers,**
- **different connection protocols,**
- **different authentication mechanisms**
- **different users**

The challenge is to keep it as much interoperable as possible and at the same time simple and user friendly

What is it?

It provides an API which models a generic Data Infrastructure

Think about a hub for cargo ships.

Many ships, different in model and size, can dock.

The transfer of the containers is taken care, regardless of their content

This is what we call *orchestration*

No, I mean, what is it really about?

Giving *researchers* an easy way to transfer data between different systems

Giving *researchers* an easy way to request and share data between different users

Giving *system administrators* an easy way to connect their systems and automate workflows

Giving *developers* an easy way to implement solutions on top of data API

SURF

iRODS

iRODS is one of the supported **data providers** via an irods **fsspec** implementation:

https://gitlab.com/surf-dms/neptune_project/irods_fsspec2

- Tested with iRODS 5.0.2 and python irods client 3.3.0
- The component is used in both the server and the Neptune CLI, "surfie".
- In the CLI it supports upload and download of data

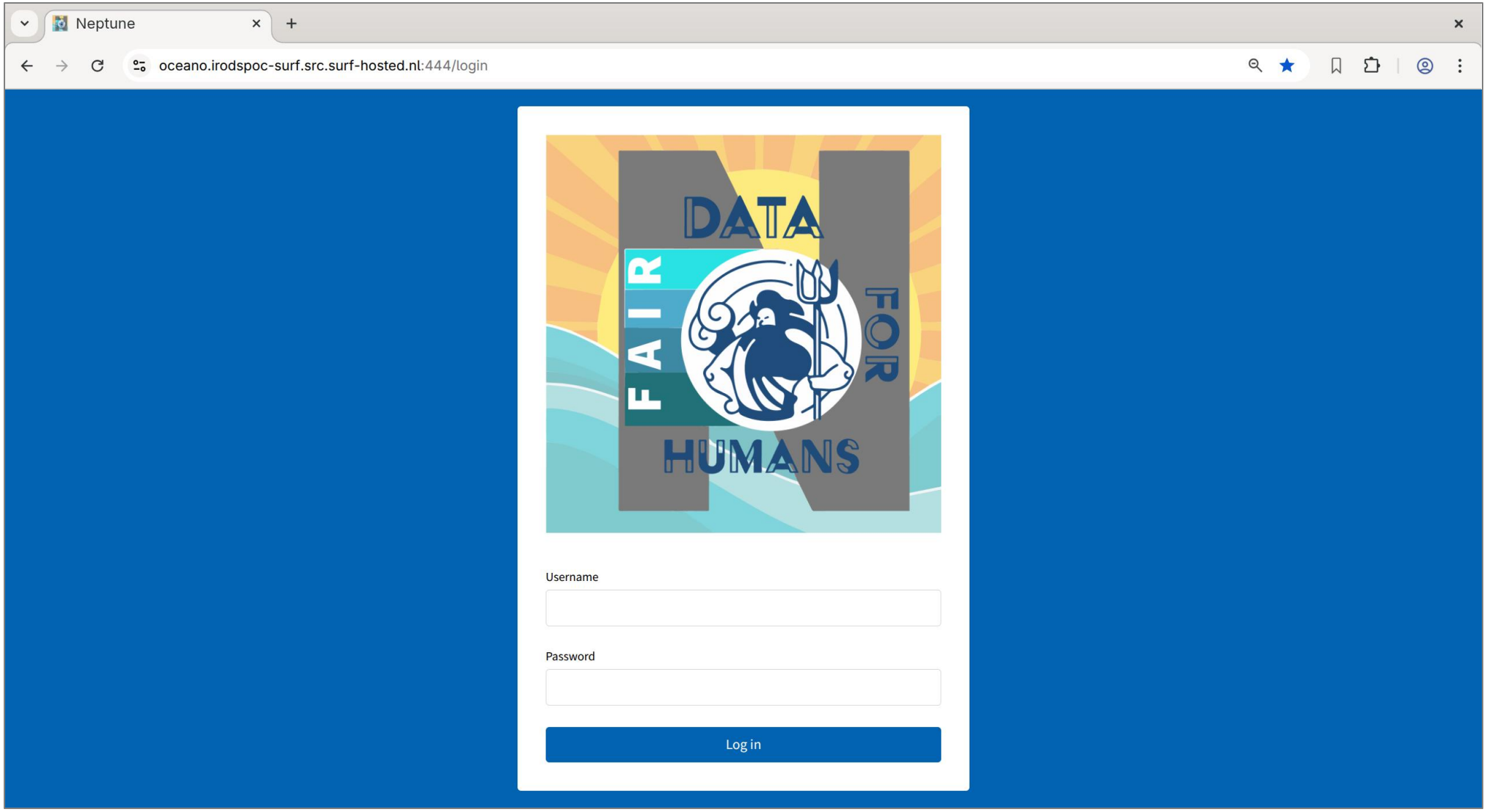
edu.nl/7jagn

IRODS fsspec implementation

The fsspec implementation for iRODS provides the following benefits:

- Automatic registration into fsspec registry
- Ticket-based authentication
- Session management: implicit and explicit authentication
- Session concurrency: bounded semaphore (max 32 per process)
- Retry logic for authentication failures: 3 attempts, exponential backoff (2s, 4s, 8s)

Demo

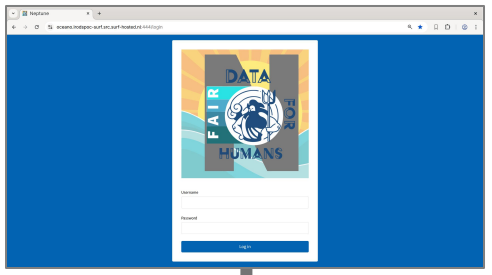

Neptune
oceano.irodspoc-surf.src.surf-hosted.nl:444/login
DATA
FAIR
FOR
HUMANS
Username
Password
Log in

How it works

The agent collects the requests

Streams the data between the two services

Monitors the transfer and reports back



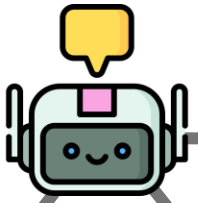
DataConnect (Web UI)

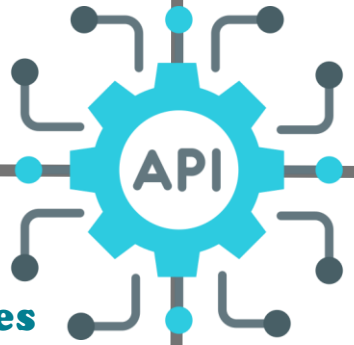
User logs in

Connects to services

Schedules a data transfer

Triton (agent)





Waves

DB

The data transfer request is stored in the DB

Service 1

Service 2

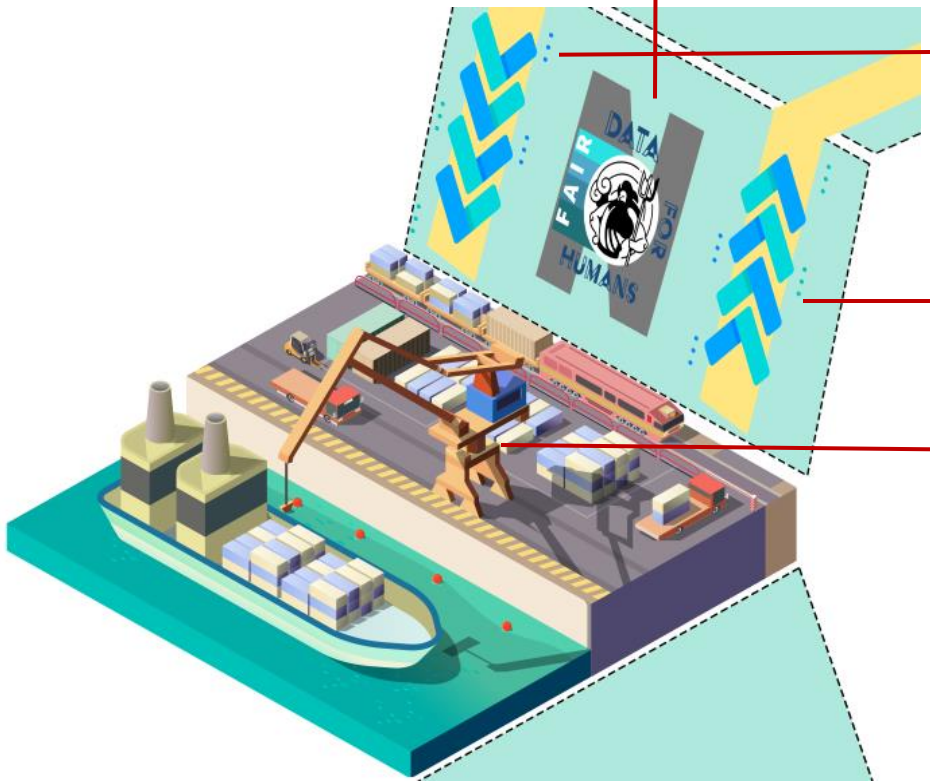
Vault

The credentials are encrypted and stored in the vault

SURF

Agent icons created by Freepik - Flaticon Api icons created by SBTS2018 - Flaticon DB icons created by Design Circle - Flaticon Secret icons created by Freepik - Flaticon

Triton



Authenticates to the server
with username and password as a regular user

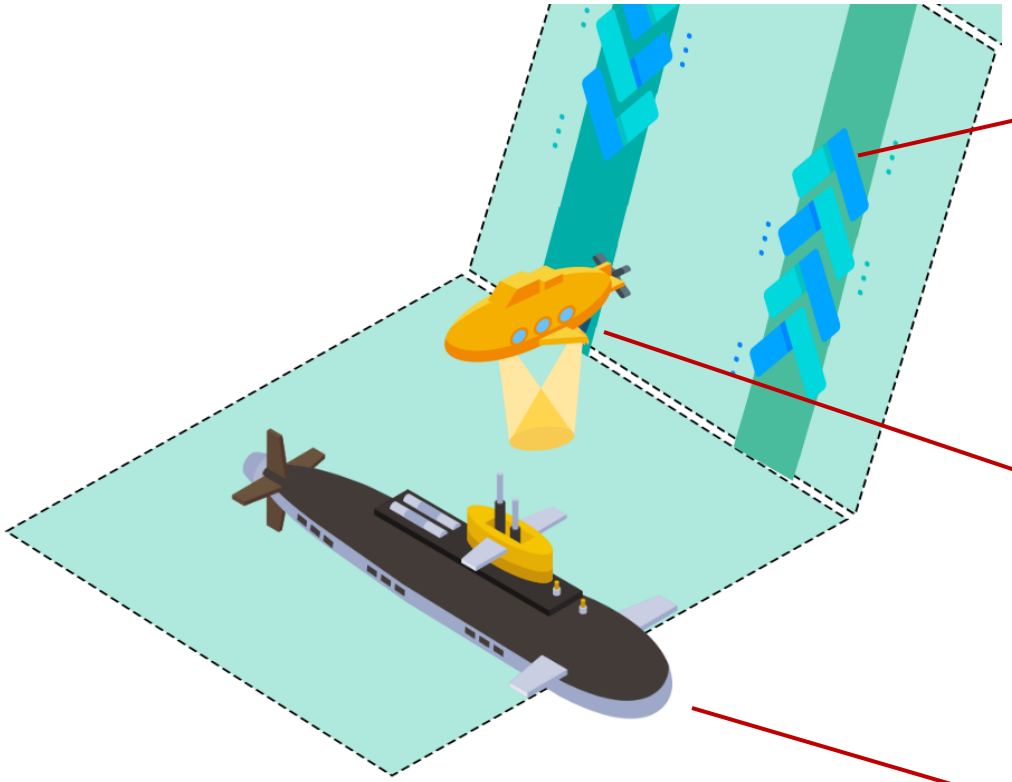
Collect the data transfer requests
Decrypts credentials
Asymmetric key -> decrypt symmetric key -> decrypt credentials

Executes the copy
via fsspec (**iRODS**) or rclone
Retries on failure

Reports progress
Update provenance records

SURF

Search back-end



Search

Asynchronous tasks executed by the agent, Triton.
They have a status: pending, partial, completed

Virtual collections

The result of a search task, they can be static or dynamic

Dedicated search back-end

iRODS python client and genquery

Generic search back-end

fsspec file listing matching folder and file names

The team

- Alice Stuart-Lee
- Claudio Cacciari
- Eduard Klapwijk
- Francisco Morales
- Marcel Koek
- Maryam Soleimani Dodaran
- René Wiermer
- Malou Allertz
- Simone Tertoolen

Questions

Neptune code repository:

https://gitlab.com/surf-dms/neptune_project

edu.nl/3ybpj