ILDG Services and Middleware

Working Groups

Hands-on Workshop

June 14, 2023

Overview

1. Use cases
2. Distributed Web Services
3. Basic ILDG Services
4. Interacting with ILDG Services

Use Cases

ILDG needs to support 4 different use cases (and user requirements) for sharing and exchanging of gauge configurations:

	data **prodvider**	data **consumer**
community-wide sharing ("public" or "published"?)	✔	✔
collaboration-internal sharing (initial "embargo" restrictions)	✔	✔

Community-wide data sharing

Data provider:

- store precious (meta) data somewhere at no cost (human and storage resources)
- declare it public
- get it used by others
- receive credits/citations

Data consumer:

- we somehow know about existence of interesting and useful data
- get the data at no cost (human and CPU)
- use data freely to do high quality research
- generously acknowledge the source of the data

Data sharing within or between Collaborations

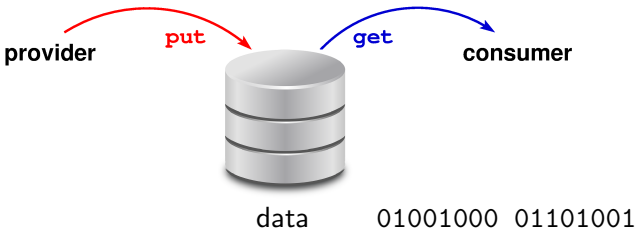
Data provider:

- follow a well-defined and smooth workflow
- public and internal data can be handled in the same way (no extra efforts at end of embargo times)
- public data is also published and citable
- efforts are rewarded by funding agencies

Data consumer:

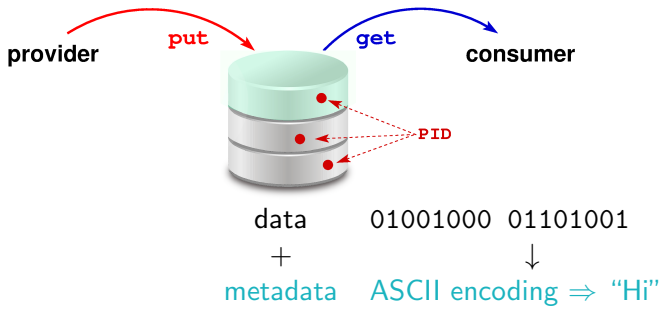
- everything is known about our configs (location, tracking, reliability, ...)
- we have a clear data managment plan
- data stewards take care of all our (meta)data
- usage rules are well defined and known

Naive Data Sharing



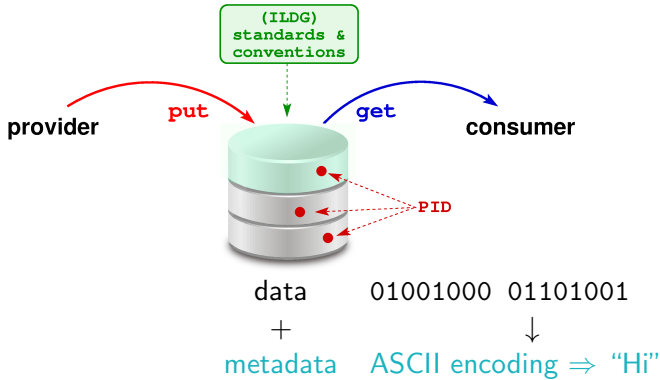
Data (bits) without meta data (= information about a digital object) is useless!

Naive (Meta-) Data Sharing



- Many kinds of metadata (MD): format, content, provenance, access policies, ... **F2, R1**
- (Meta-)data objects must have persistant (globally) unique identifier(s) **F1, A1**

Naive (Meta-) Data Sharing



- Many kinds of metadata (MD): format, content, provenance, access policies, ... **F2, R1**
- (Meta-)data objects must have persistant (globally) unique identifier(s) **F1, A1**
- Standards and (community-specific) conventions **A1, R1.3**

Possible Realizations

- File system:
  - identifier = name and path of files (?)
  - metadata = content of files (separate or combined with data)
    and other attributes (e.g. permissions)

- Database:
  - identifier = primary key
  - metadata + data (!) = "attributes"

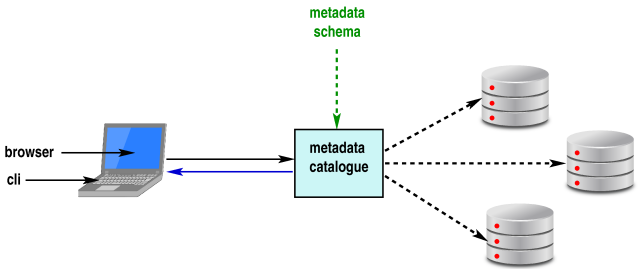
key	metadata	data
⋮	⋮	⋮

Problem: data objects in LQCD are relatively large

- metadata needs to be stored separately from data for efficient searching **F4**
- storage hardware (in practice) needs to be distributed over different sites **$$**
  (institutions, funding agencies, countries, . . .)

➜ **Distributed web services**

Distributed Web Services



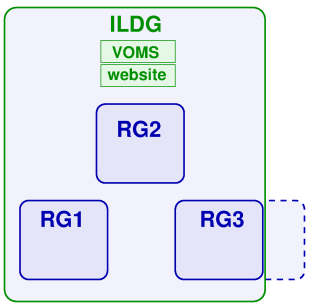
Not (just) web pages!

- grid (or cloud) storage elements (SE)
- central Metadata Catalogue(s) (MDC)

Global Services and Organization of ILDG

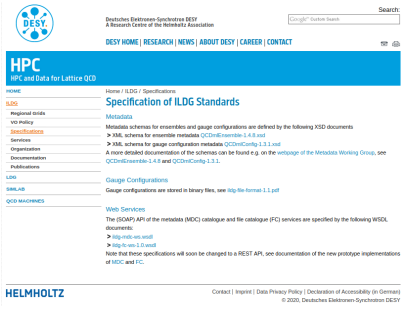
ILDG operates only 2 global services

- VO registration (VOMS)
  registry of ILDG users (groups and roles)
  used for authentication to storage elements
- Web page (temporary)
  specification of standards and conventions
  URLs of services of each regional grid (`Services.xml`)

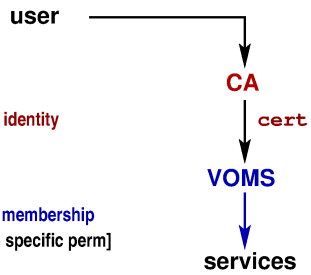


ILDG Web Page

- Under construction!
- VO Policy
- Specifications:
  - metadata schema
  - file formats
  - working groups
  - URLs of services
- (Incomplete) user documentation



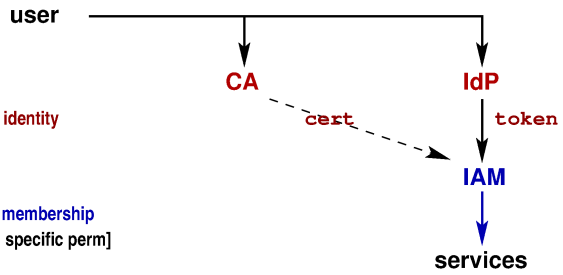
Membership of the Virtual Organization "ILDG"



Currently: (ILDG 1)

- identity = grid certificats from trusted CAs (IGTF)
- membership = VOMS hosted at DESY

Membership of the Virtual Organization "ILDG"



Currently: (ILDG 1)

- identity = grid certificats from trusted CAs (IGTF)
- membership = VOMS hosted at DESY

Future: (ILDG 2)

- identity = token from trusted IdPs (e.g. home institutions in eduGAIN)
- membersiip = IAM (Identity and Access Management) hosted at CNAF/INFN

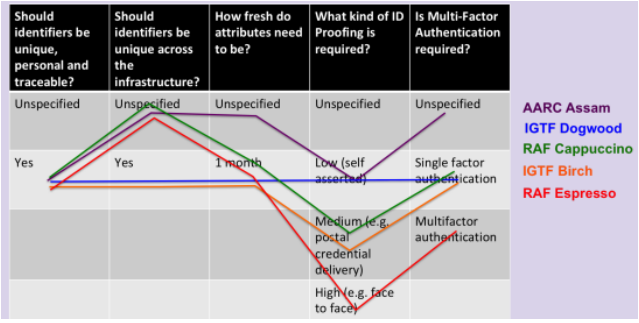
The Need for trusted Identity Proofing

Service providers used by ILDG require a reliable identification of users, e.g. for

- storage (even for read-access only!)
- fast (!) network connections

➜ Federations of CAs/IdPs which can guarantee a well-defined Level of Assurance (LoA)

➜ Users need to respect rules
AUP: SPs ↔ users
VO Policy: users ↔ users



AARC Acceptable Authentication Assurance Policy

Autonomous Regional Grids

Services operated by each Regional Grid

- Metadata Catalog (MDC)
- File Catalog (FC)
- Storage Elements (SE)
- Website with RG-specific information

Regional Grids: CSSM, JLDG, LDG, UKQCD, USQCD

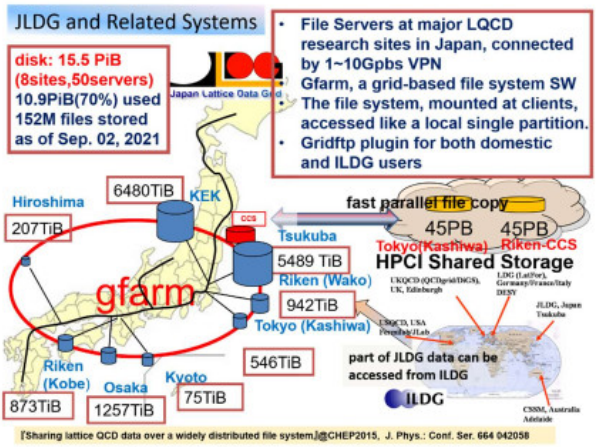
- are implemented with different architectures and technologies
- operate in an autonomous way with individual policies

Examples

- JDLG: single SE, no specific access control
- LDG: multiple SE, fine grained access control

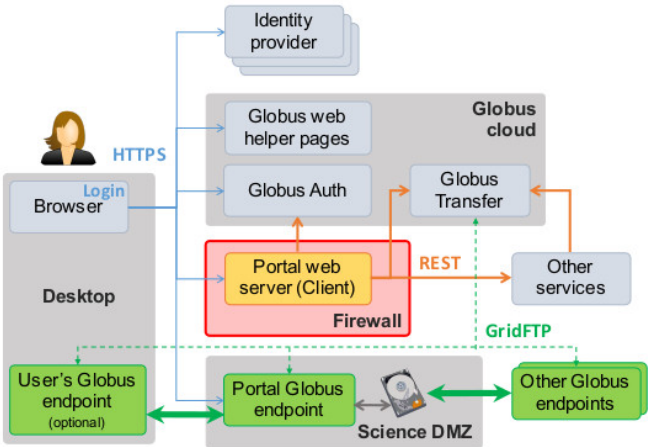
JLDG Architecture

- Single federated storage system (GFARM)
- JLDG-internal write access
- Fast read access (`gridftp`) available for VO members
- Transition to token-based authentication



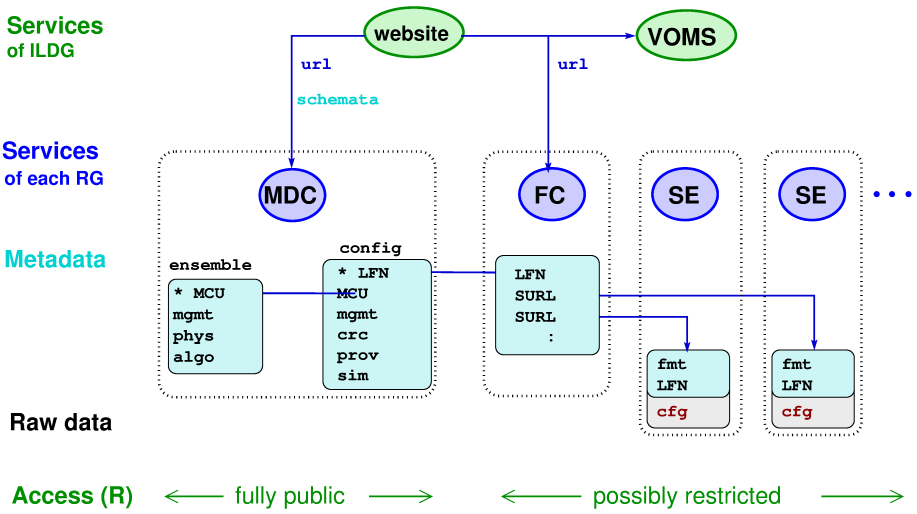
T. Yoshie

USQCD Ideas



K. Chard et al. 2017

Interplay between ILDG Services



Metadata Catalogue (MDC)

Key purpose: ID regisration and metadata search

$$\boxed{\text{ID} \longleftrightarrow \text{metadata}}$$

Database schema

	metadata collection name		
	config	ensemble	publication (?) ...
primary key:	LFN (dataLFN)	MCU (markovChainURI)	DOI (?)
attributes:	QCDml tree	QCDml tree	t.b.d.
	MCU	license (not yet)	list of MCU
			DataCite metadata

Basic operations

- Search: query → list of IDs (supporting powerful Xpath queries) **F4**
- Retrieve: ID → MD (QCDml schema) **A1**
- Validate, insert, update, delete, ...

File Catalogue (FC)

Provides: functional (many-to-one) relation

$$\boxed{\text{FC: SURL} \longrightarrow \text{LFN}}$$

Database schema

primary key:	SURL (Storage URL)
attributes:	LFN
	MCU (or other optional Access Control Attributes)

Basic operations

- ∗ list entries (SURL) by LFN **A1**
- list by other criteria (SURL, Access Control Attributes)
- insert, update, delete, ...

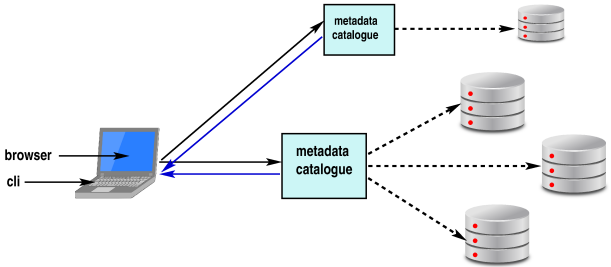
New Implementation of MDC and FC

(by Basavaraja BS @ DESY/NIC)

Technical details

- configurable, e.g. for additional collections (beyond ensembles and configs)
- additional attribute service for access control (ACS)
- REST API (see online documentation MDC, FC, and ACS)
- simple deployment (Docker containers, Kubernetes in preparation)
  e.g. for other regional grids or applications
  - JLDG: 60 ensembles, 40 k configs
  - LDG: 2 instances, 250 ensembles, 250 k configs (not yet consistent with SE)
  - UK: in preparation

Interaction with ILDG Services



- Catalogues of all regional grids are interoperable due to standardized API
- High-level user operations may need several (≤ 10) low-level requests (e.g. HTTP) but still few compared to other web pages (implicitly handled by your browser)
  - www.google.com: O(25) requests
  - www.github.com: O(100) requests (try Ctrl-Shift-I in firefox)
  - your favourite airline: O(200) requests

Consistency of ILDG Data

❒ ILDG is logically a distributed relational database with 2 kinds of entities

- configs: metadata + (binary) data
- ensembles: metadata

and corresponding primary keys

- LFN (dataLFN): `lfn://rg/collaboration/project/name`
- MCU (markovChainURI): `mc://rg/collaboration/project`

☛ Persistence and globally unique identifiers needs to be guaranteed by data providers.

❒ Typical inconsistencies (RDB anomalies) may arise from

- failures of individual services
- incorrect use of low-level tools

and can only be

- detected and fixed by regular scans (with possibly prohibitive cost)
- checked and handled by high-level tools (including roll-back)

Use Cases and "ltools"

Consumer (collaboration internal)

- `lfind`: search in metadata catalog
- `lget`: download data and metadata

↓

Consumer (community wide)

- optionally also use common search engines
- cite DOIs when using published data

Provider (collaboration internal)

- `lpack`: generate markup*) and pack data
- `linit`: register ensemble metadata
- `lput`: upload config data and metadata

↓

Provider (community wide)

- optionally register DOI and generate landing page
- drop access restriction flag
- receive data citation record

*) trivial if information is already collected during production!

Examples of Interactions with ILDG Services

- "Login" to ILDG: `voms-proxy-init`
  - periodically ($\leq$ 2 days) download latest CRLs $\longleftrightarrow$ CA
  - unlock your private key (by pass phrase) or login at your IdP $\longleftrightarrow$ IdP
  - request VO membership info and attributes $\longleftrightarrow$ VOMS | IAM
  - generate VOMS proxy
- "Get" config data (for specific and known LFN)
  - optionally download config (and ensemble) metadata $\longleftrightarrow$ MDC
  - authenticate with proxy to FC and request SURL list $\longleftrightarrow$ FC/AC
  - authenticate to SE and download data $\longleftrightarrow$ SE/AC
- "Put" config data (for existing ensemble)
  - upload config metadata $\longleftrightarrow$ MDC/AC
  - decide SE and SURL (agreed with RG admin)
  - register SURL $\longleftrightarrow$ FC/AC
  - upload binary $\longleftrightarrow$ FC/AC

Hands-on Exercises

Please keep in mind:

- ❒ you are using a prototype system, some components of which have been re-activated or newly developed only during the last months
- ❒ currently we do not (yet/any more) have user-friendly "ltools", but only quick and dirty scripts for low-level operations:
  - `try-mdc`, `try-fc`, `try-acs` (just wrapper scripts for `curl`)
  - `lime-ls`, `lime-cat1`, `lime-pack1`
  - `ildg-cksum`, `ildg-binary`

  ... you all can improve them and contribute to develop proper user tools!

Hands-on Exercises (cont.)

- Required SW packages are installed in the workshop image (docker + apptainer) which you should have running according to the instructions .
  In particular, the container includes
  - `voms-proxy-init`
  - `gfal` commands (Grid File Access Library)
  - `curl` (version working with proxy certificates) and other utilities
- Additional material is on gitlab
  - exercise instructions
  - scripts for low-level testing (`try-*`)
- Ready to get hands on? (and fingers dirty!)
- Please also prepare 1 slide/group for the wrap-up discussion on Friday
  - Which technical aspects did not work or are difficult / inconvenient?
  - Which technical aspects worked well or are convenient?
  - Which aspects of the markup are problematic for your project?
  - Which elements of the metadata schema are missing / incompatible for your projects?