CRIC: A topology system for computing infrastructures and an interface for VO configuration

Panos Paparrigopoulos on behalf of the CRIC team

What is CRIC

Computing Resource Information Catalog

CRIC is a framework providing a centralized (and flexible) way to describe which resources are provided by a computing infrastructure and also how the various organisations that run on the grid use them:

- Clear distinction between resources ***provided by*** (Sites) and resources ***used by*** (Experiments)
- **Experiment independent, but still experiment-oriented**
- Plugin based approach allows customization to address various experiment requirements and implementation of the dedicated experiment instances
- Shared building blocks to optimize development process and to ensure common look and feel. *Think about it in terms of lego bricks*
- Flexibility to address technology evolution and changes in the experiment computing models and applications. *Lego bricks again!*

Who is using CRIC

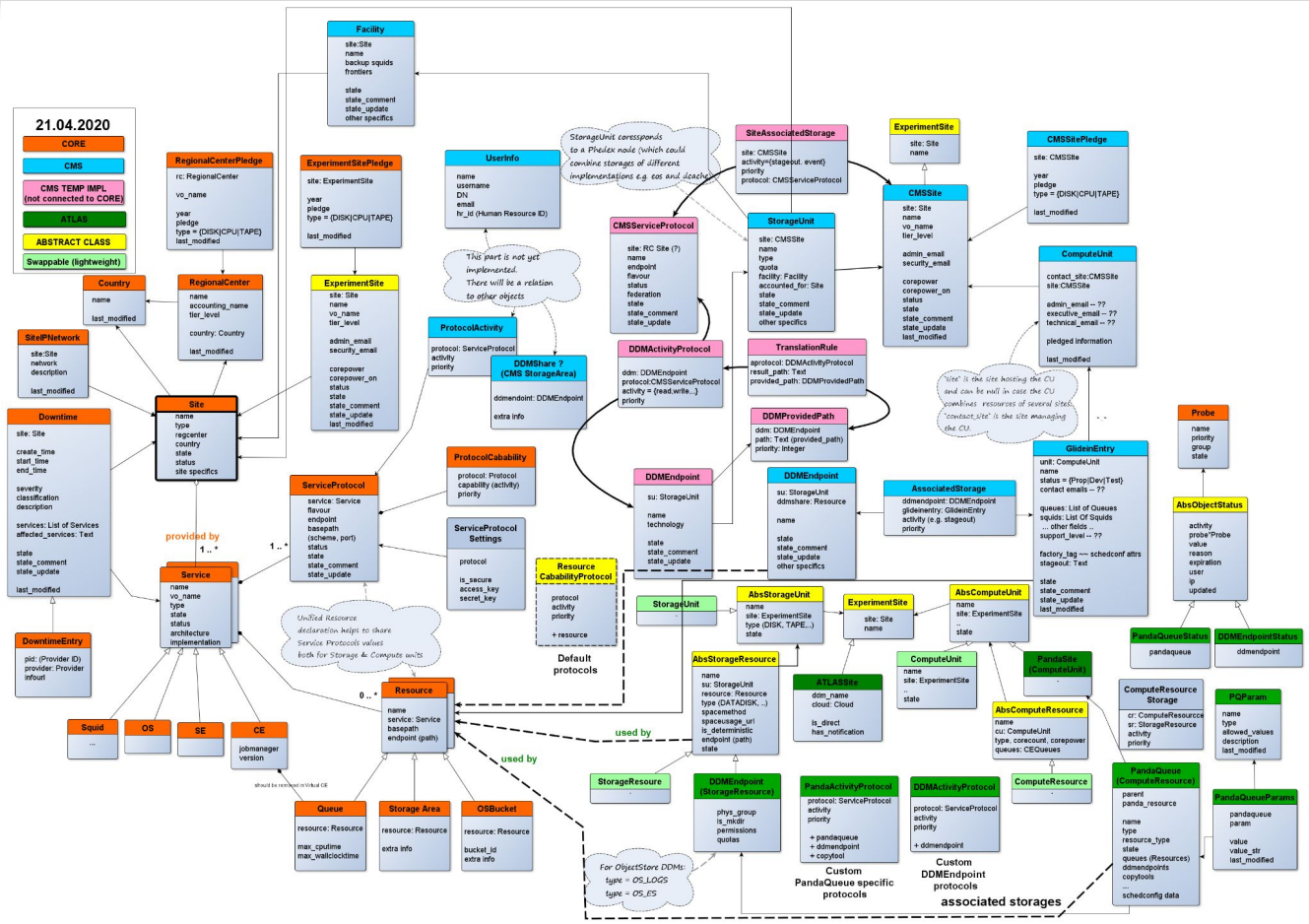
- CMS, ATLAS, Dune and WLCG already have CRIC production instances
- datalake-cric is being developed for LSST
- CMS uses CRIC as the main source for topology and A&A:
  - Rucio, CRAB and CMSWeb are fetching users and topology data from CRIC
  - Plans to configure GlideinEntries through CRIC
- ATLAS uses CRIC as it's main topology and configuration tool:
  - Configuration/Blacklisting of DDMEndpoints and PandaQueues
  - Downtime management, HammerCloud config
- WLCG is using CRIC as the main topology system and the place for central operations (Pledges declaration, accounting data validation, report generation and more)
  - WLCG-CRIC is the central place were all the topology of the different experiments is collected
  - Master source for WLCG monitoring, testing, accounting and network topology

CRIC Architecture and core features

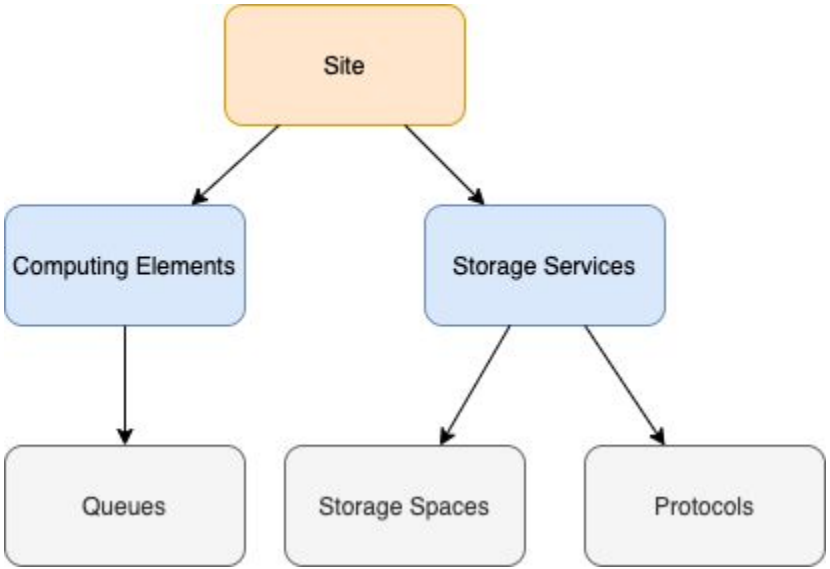
- **Plugin based: VO can configure default behaviour**
- Base implementation for the Resource/Topology description
- Customisable REST API data export (filters, presets, various output formats)
- Shared engine/widgets for WebUI (downtime calendars, table view, tree view, inline editors, etc..)
- Pluggable data sources
- Enhanced Authorization (CERN SSO, SSL, password based;local accounts)
  - More means of authorization can be easily implemented and added
- Enhanced Authentication (instance specific permissions, groups, roles, etc, map permissions to e-groups, fetch info from ext sources)
- Detailed History of Changes (who, when, how interacted with object)

CRIC Features

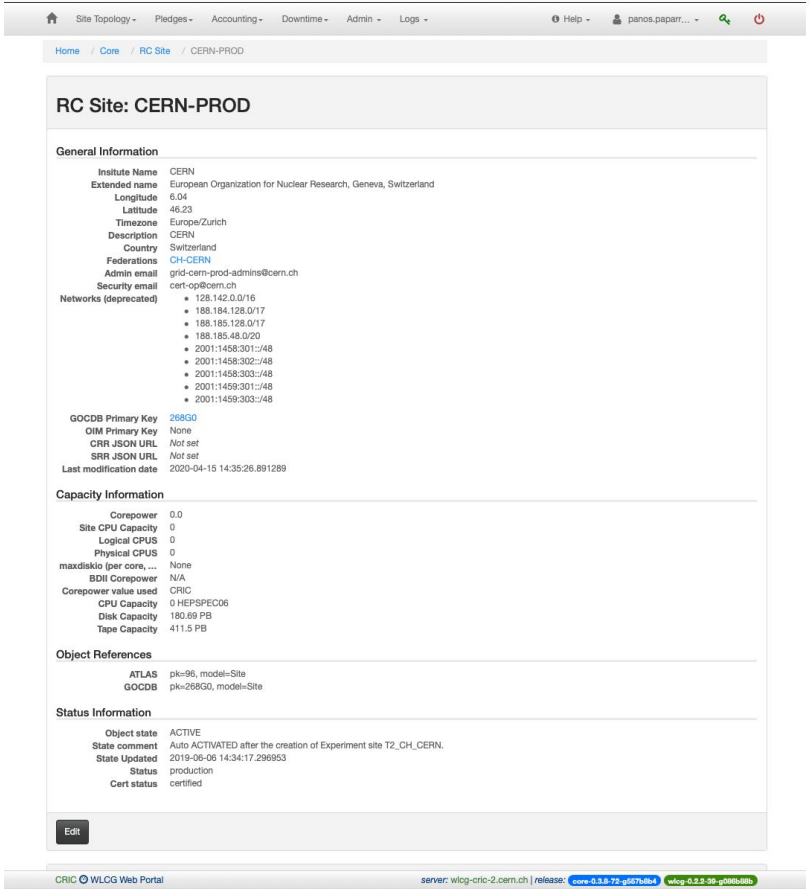
Full Computing Topology



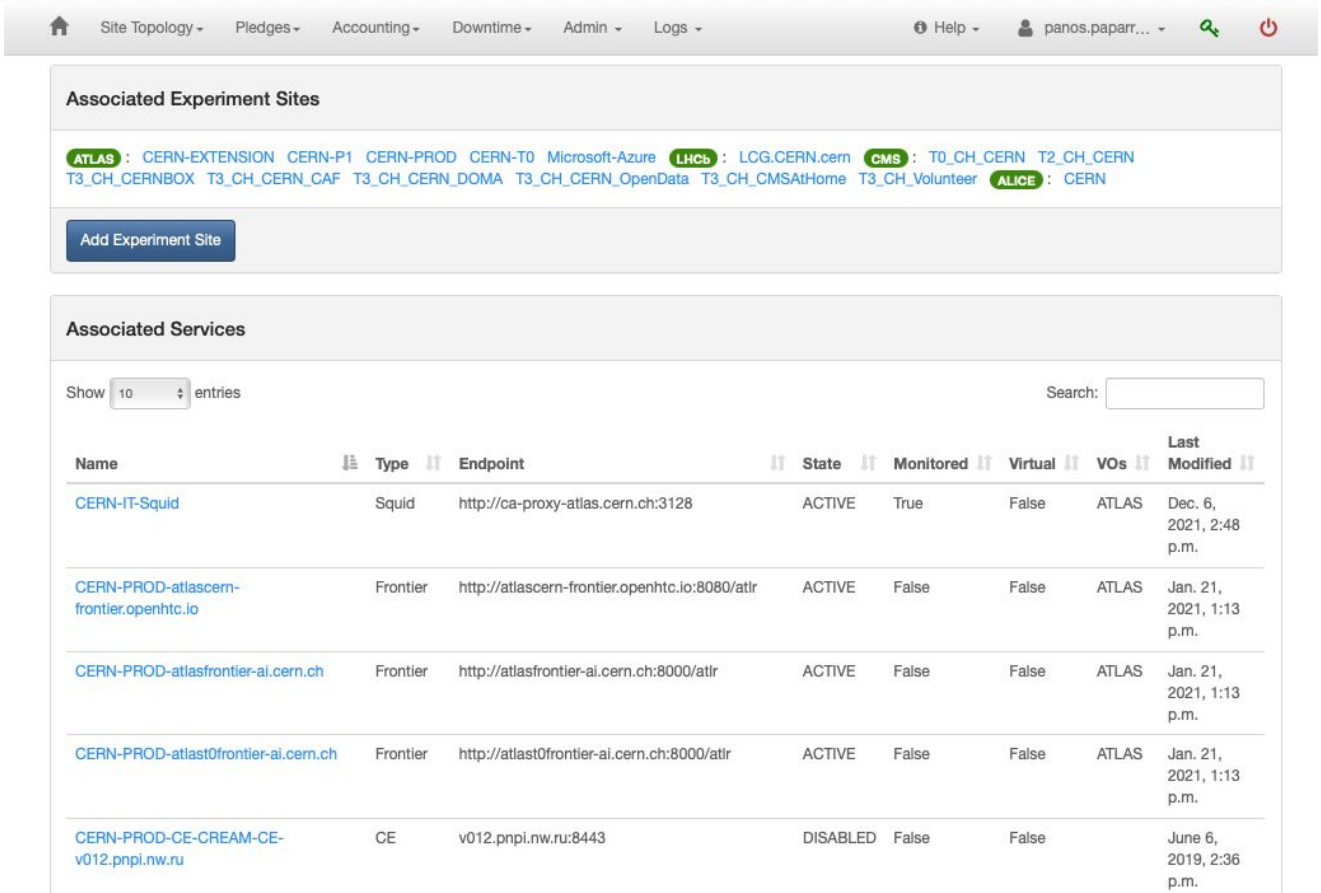
Topology Information for Sites and Services



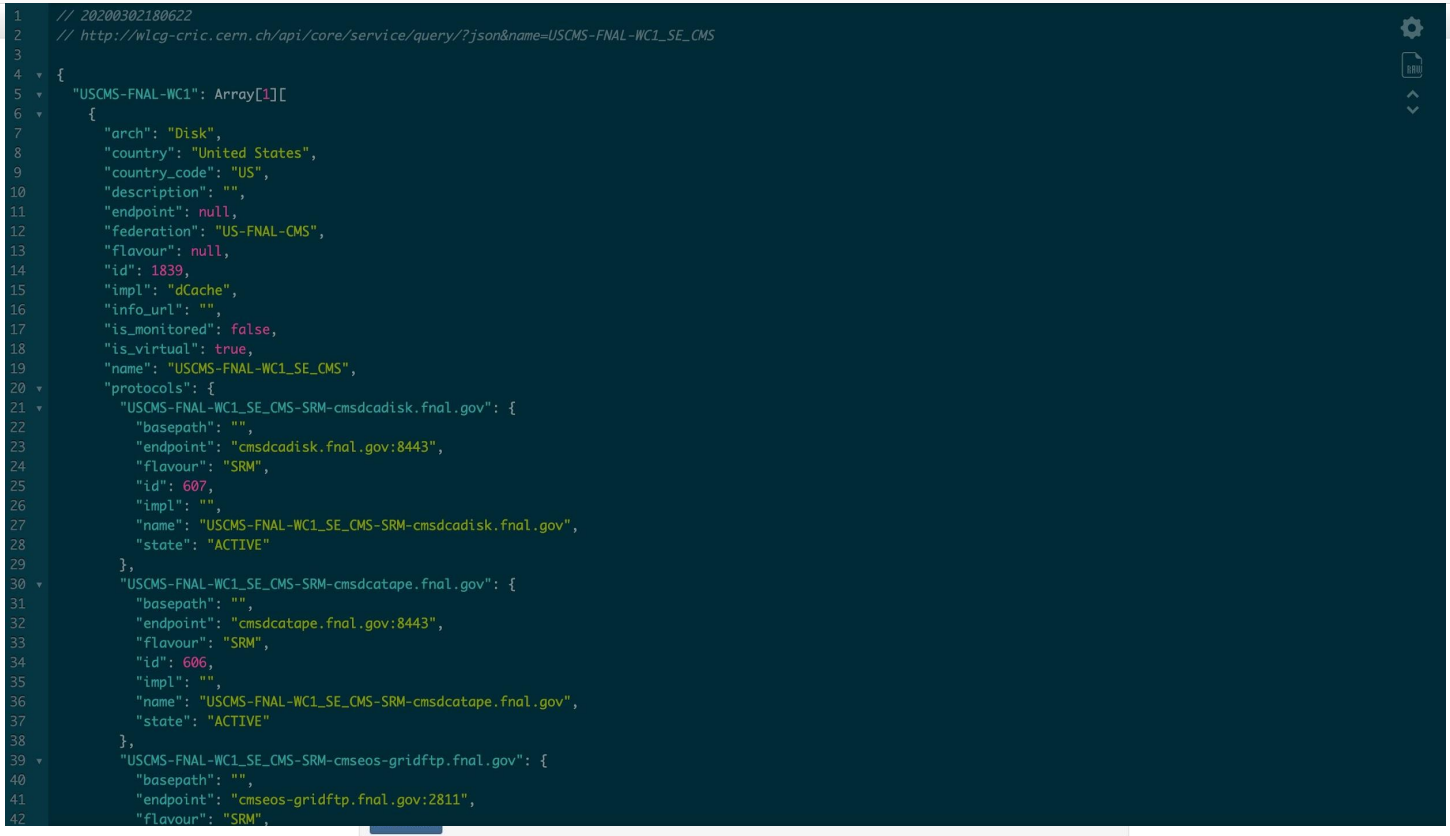
Detailed Pages



Detailed Pages



APIs



```
// 20200302180622
// http://wlcg-cric.cern.ch/api/core/service/query/?json&name=USCMS-FNAL-WC1_SE_CMS

{
  "USCMS-FNAL-WC1": Array[1][
    {
      "arch": "Disk",
      "country": "United States",
      "country_code": "US",
      "description": "",
      "endpoint": null,
      "federation": "US-FNAL-CMS",
      "flavour": null,
      "id": 1839,
      "impl": "dCache",
      "info_url": "",
      "is_monitored": false,
      "is_virtual": true,
      "name": "USCMS-FNAL-WC1_SE_CMS",
      "protocols": {
        "USCMS-FNAL-WC1_SE_CMS-SRM-cmsdcadisk.fnal.gov": {
          "basepath": "",
          "endpoint": "cmsdcadisk.fnal.gov:8443",
          "flavour": "SRM",
          "id": 607,
          "impl": "",
          "name": "USCMS-FNAL-WC1_SE_CMS-SRM-cmsdcadisk.fnal.gov",
          "state": "ACTIVE"
        },
        "USCMS-FNAL-WC1_SE_CMS-SRM-cmsdcatape.fnal.gov": {
          "basepath": "",
          "endpoint": "cmsdcatape.fnal.gov:8443",
          "flavour": "SRM",
          "id": 606,
          "impl": "",
          "name": "USCMS-FNAL-WC1_SE_CMS-SRM-cmsdcatape.fnal.gov",
          "state": "ACTIVE"
        },
        "USCMS-FNAL-WC1_SE_CMS-SRM-cmseos-gridftp.fnal.gov": {
          "basepath": "",
          "endpoint": "cmseos-gridftp.fnal.gov:2811",
          "flavour": "SRM",
```

Detailed Action Logging

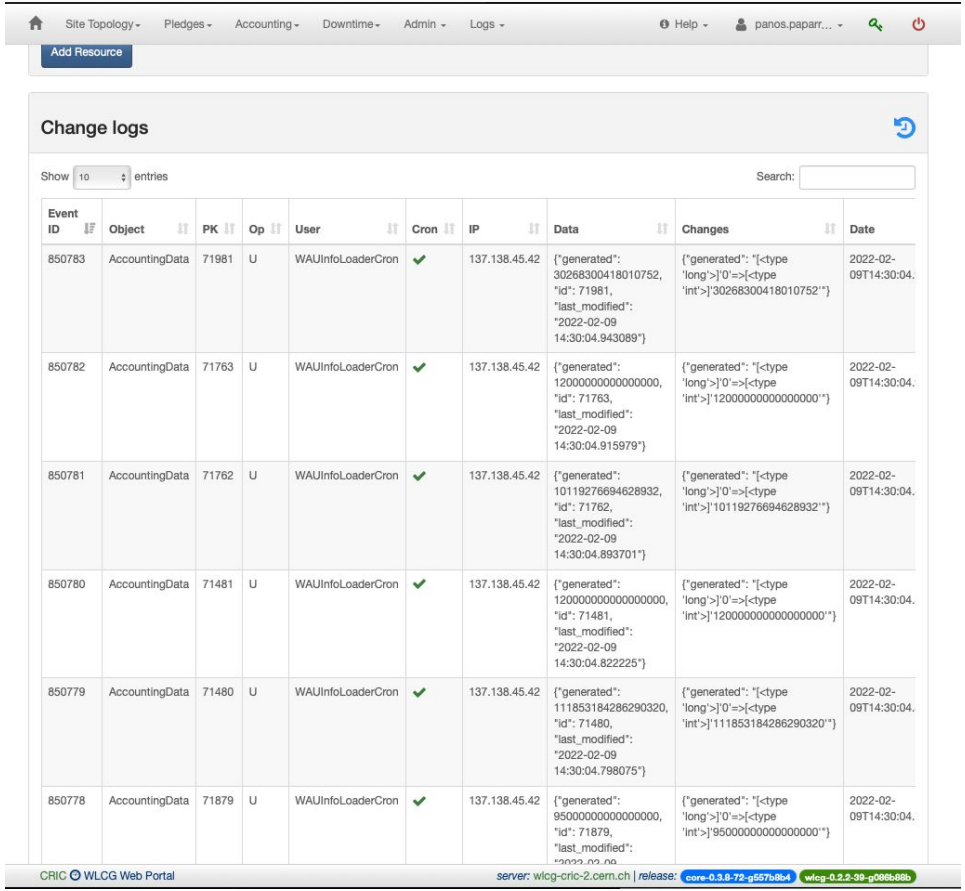
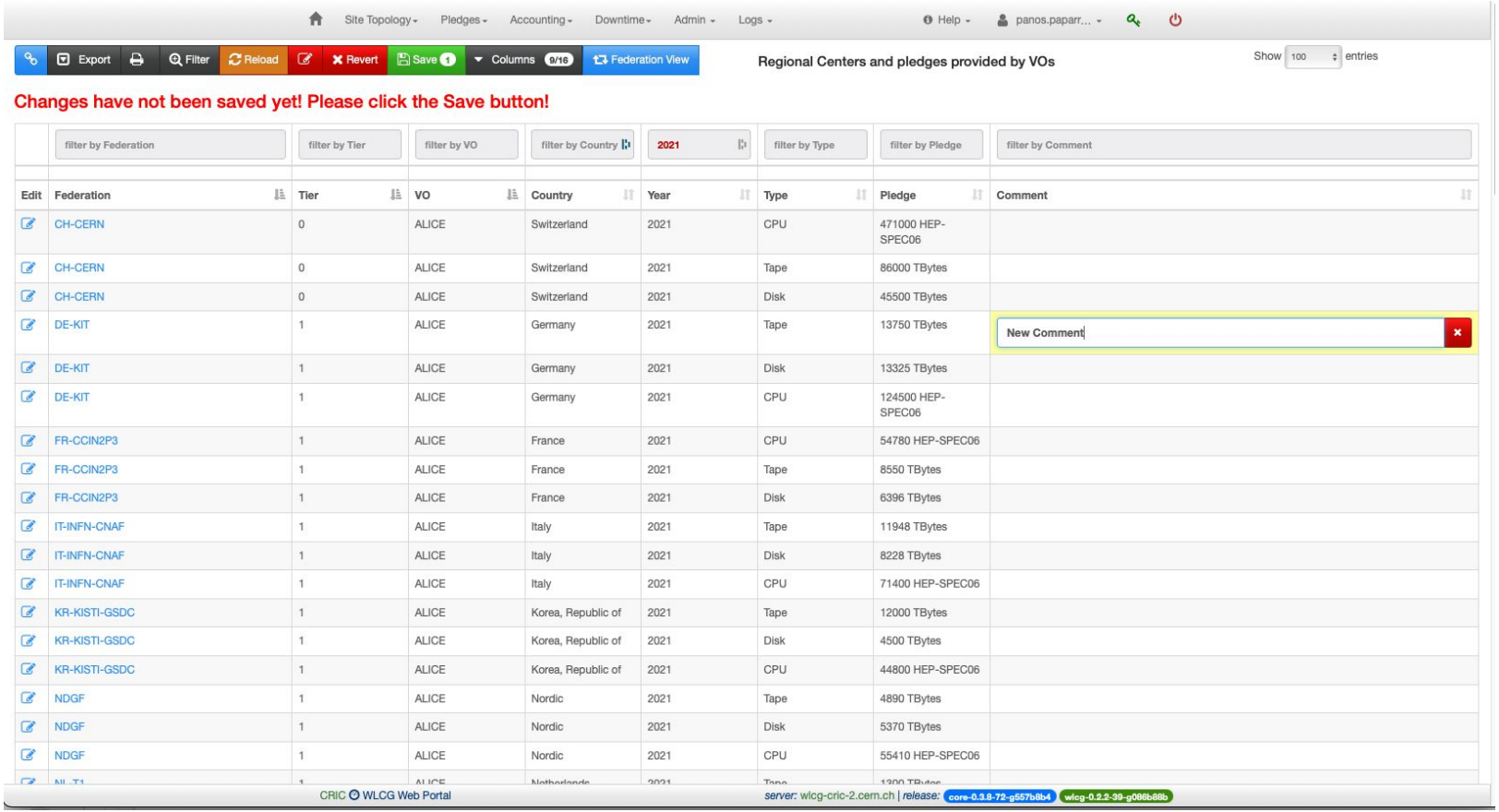
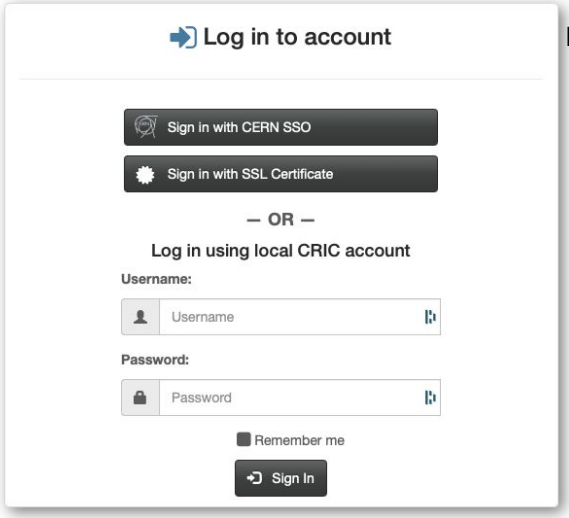


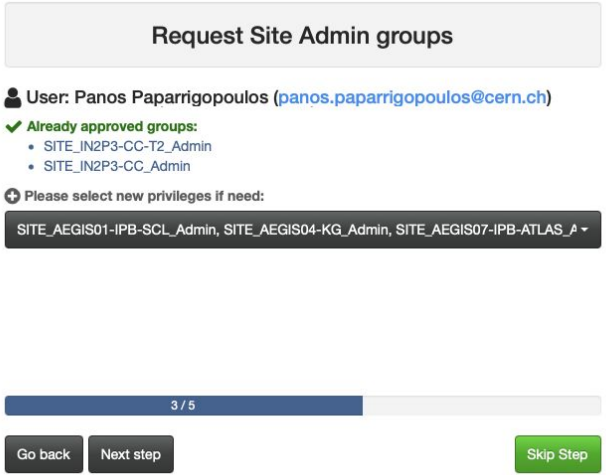
Table views and live table editing



Authorization - Authentication



Multiple authentication methods (plugin based implementation)



Request permissions wizard

Aggregation of info like DNs, usernames, institutes etc

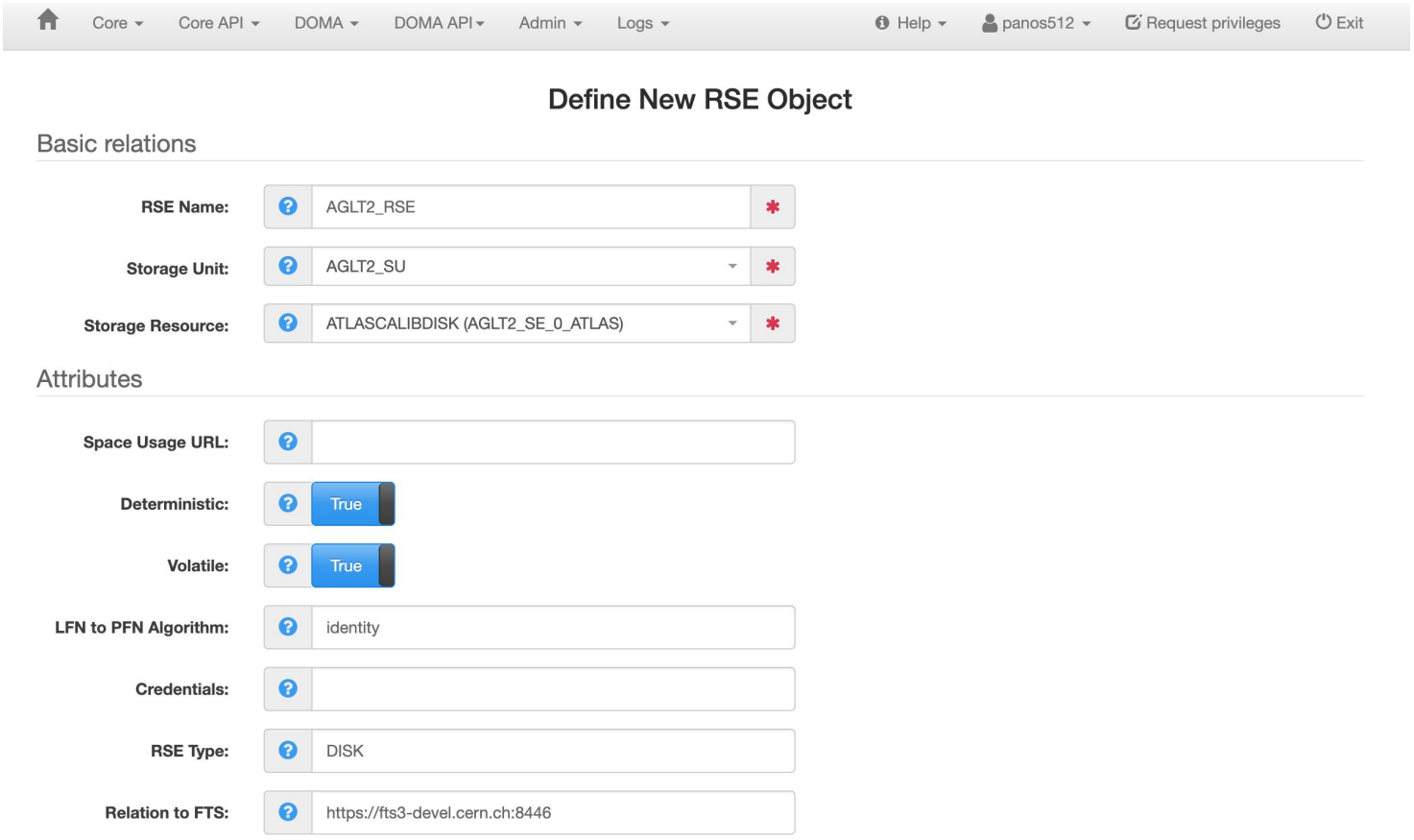
Email	DN	Username	First Name	Last name	Date Joined	Is Active
panos.paparrigopoulos@cern.ch	/DC=ch/DC=cern/OU=Organic Units/OU=Users/CN=ppaparri/CN=779320/CN=Panos Paparrigopoulos	ppaparri	Panos	Paparrigopoulos	2019-05-27T16:46:05	✔

CRIC as a Rucio interface

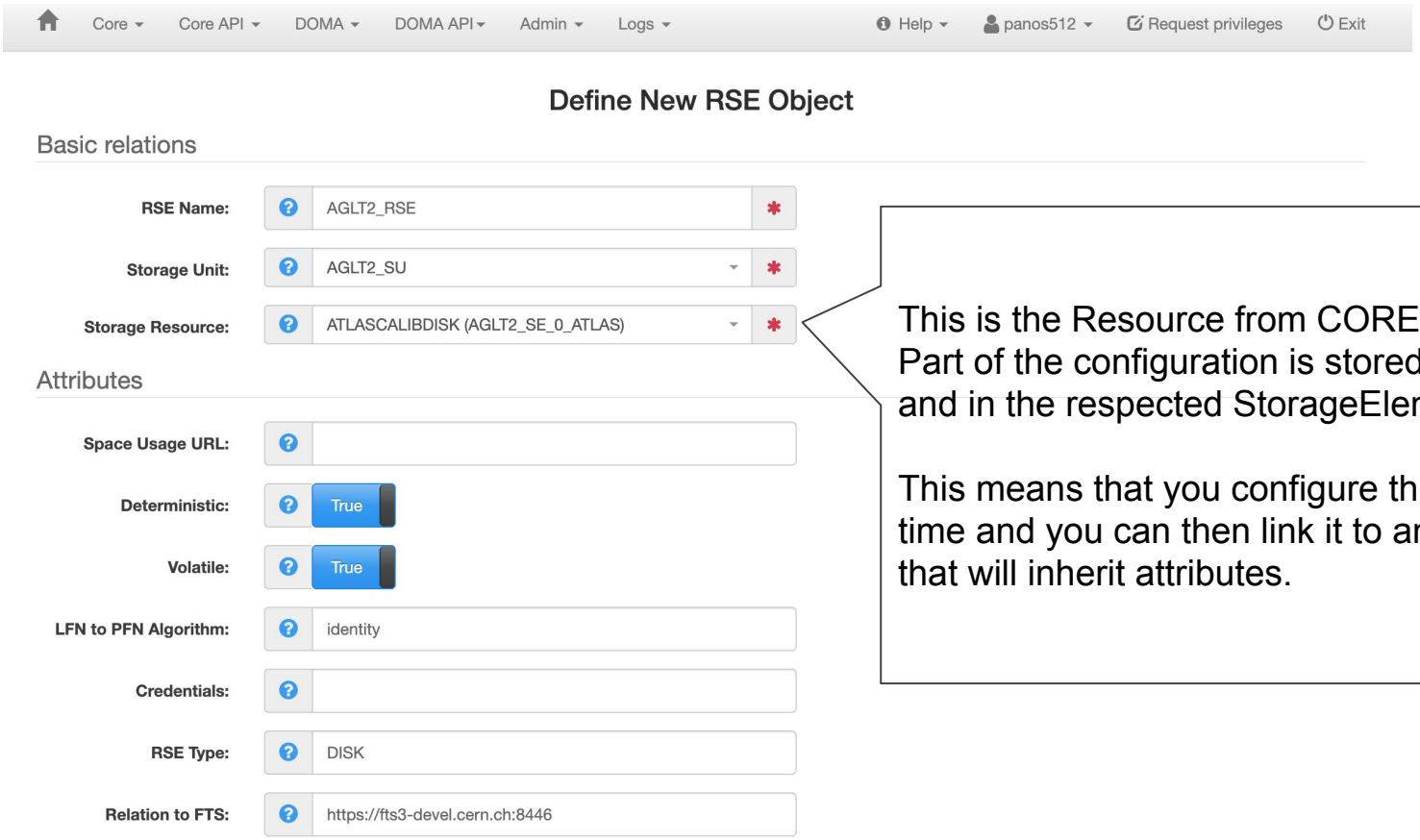
CRIC as an interface for Rucio

- Rucio is a powerful data management framework.
- With great power comes great configuration.
- CRIC already hosts information about the topology of storage services, protocols, shares and their characteristics.
- Interfaces, APIs, users and permissions were already there, offered by CRIC.
- We decided to create a CRIC extension for Rucio configuration (RSEs, Transfer Matrix, Accounts and Identities)
- With the help of a couple of Rucio probes CRIC information can be easily synced into Rucio.
- ESCAPE instance is already using CRIC for it’s configuration.

CRIC as an interface for Rucio - RSEs



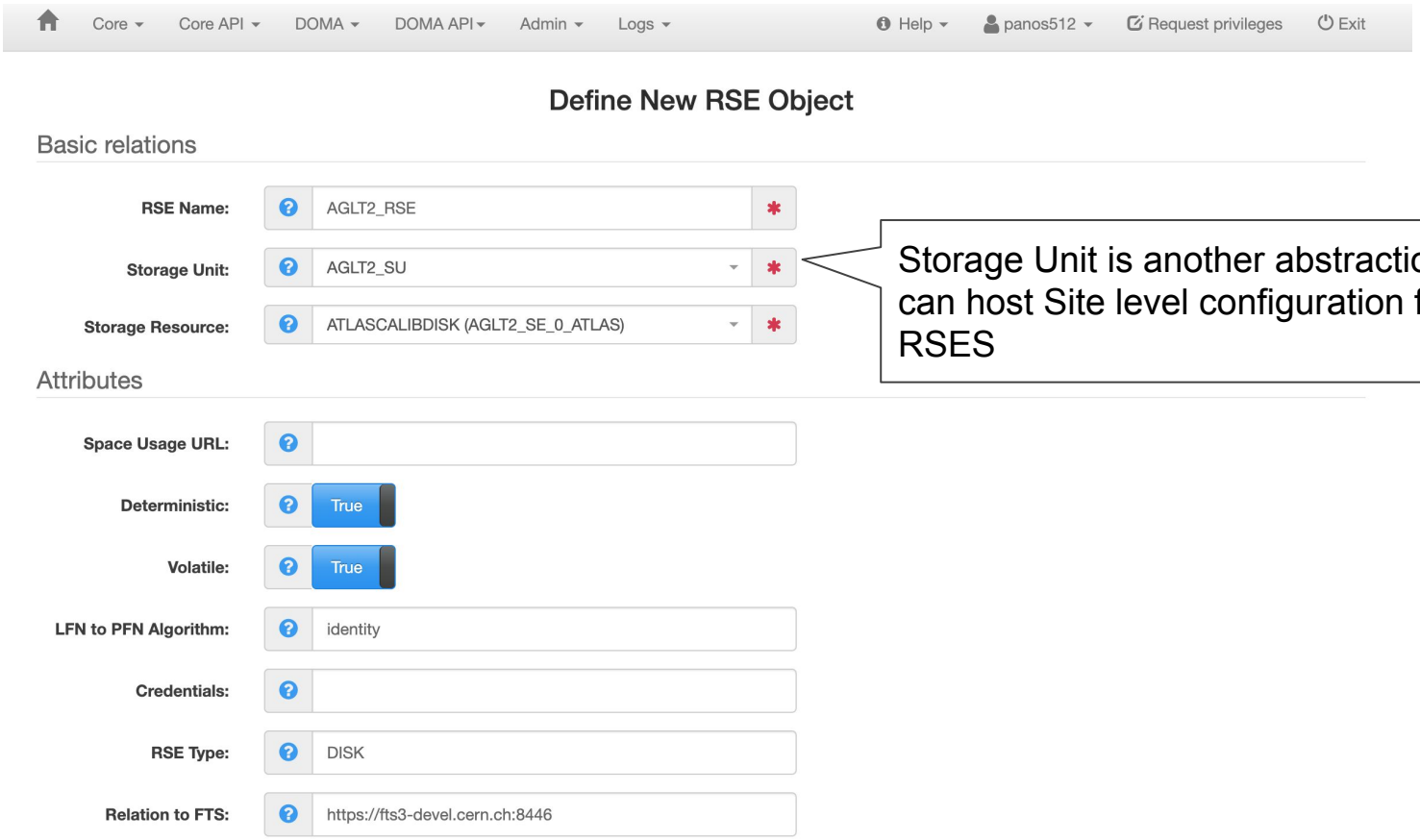
CRIC as an interface for Rucio - RSEs



This is the Resource from CORE.
Part of the configuration is stored there and in the respected StorageElement.

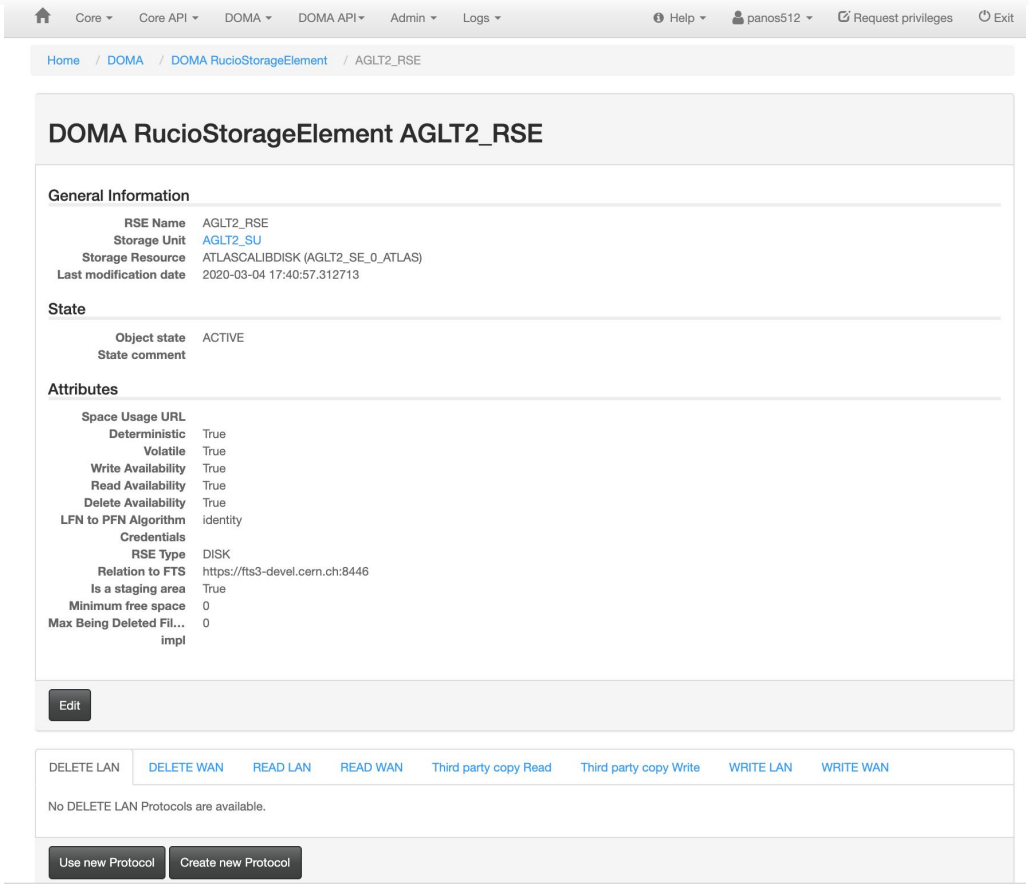
This means that you configure this one time and you can then link it to any RSEs that will inherit attributes.

CRIC as an interface for Rucio - RSEs

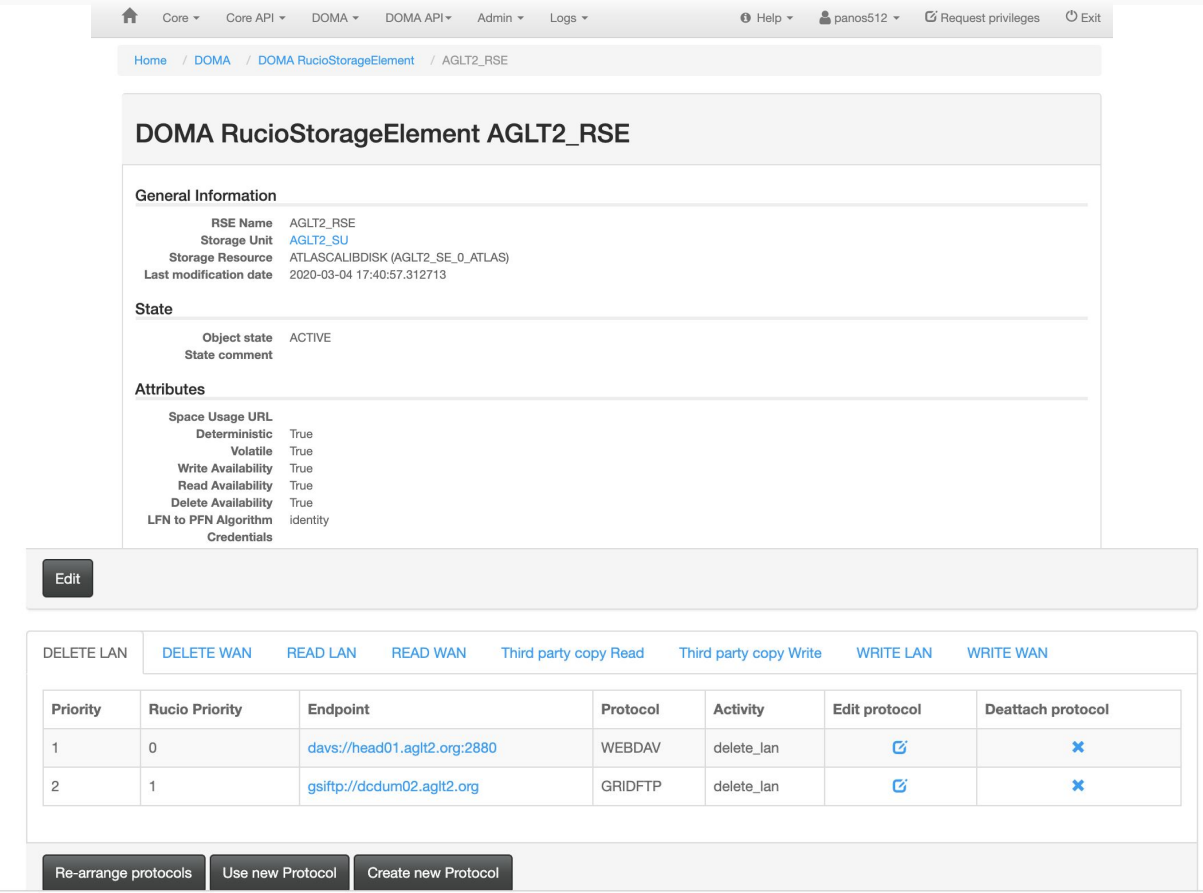


Storage Unit is another abstraction that can host Site level configuration for all the RSES

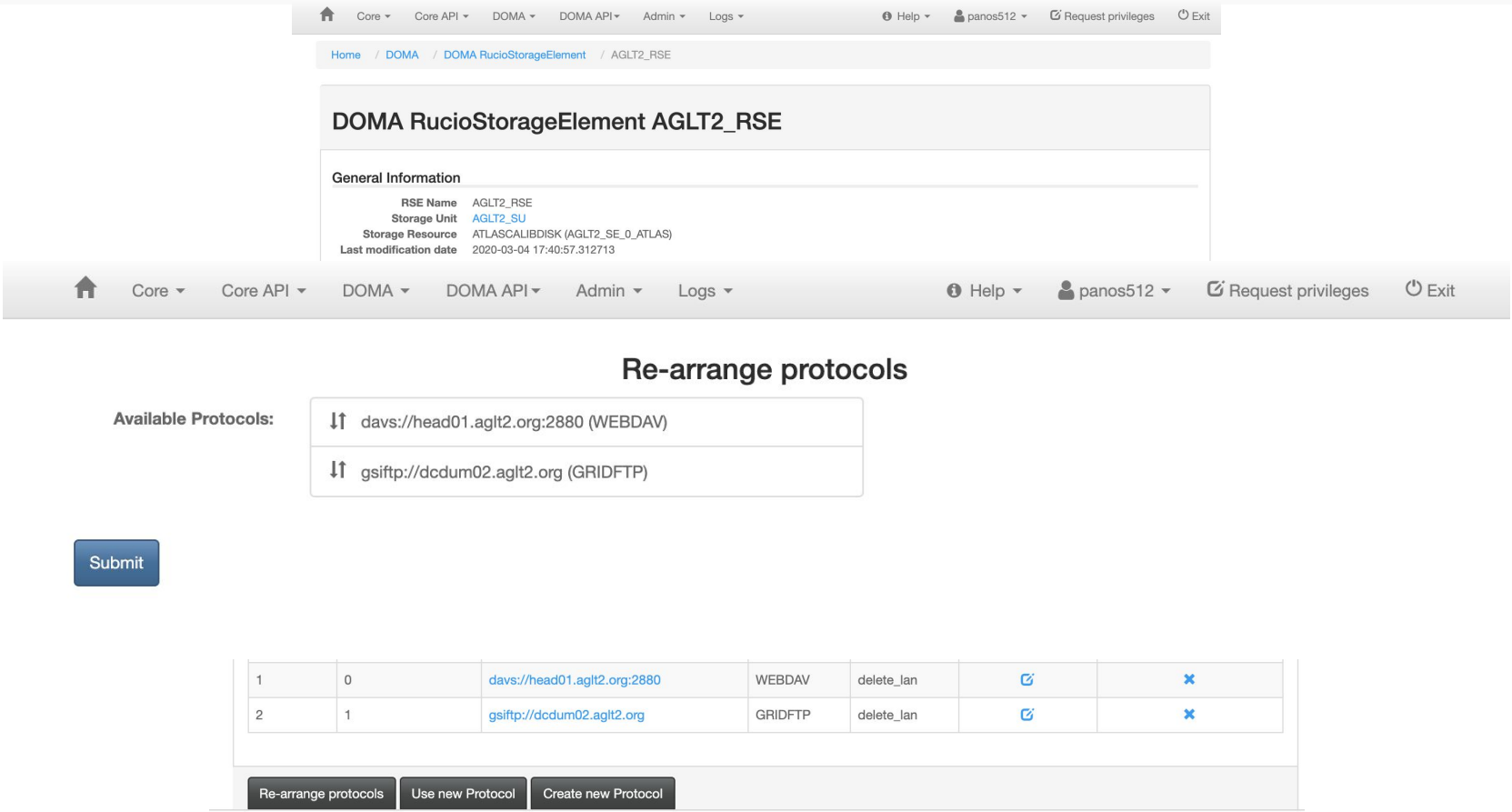
CRIC as an interface for Rucio - RSEs



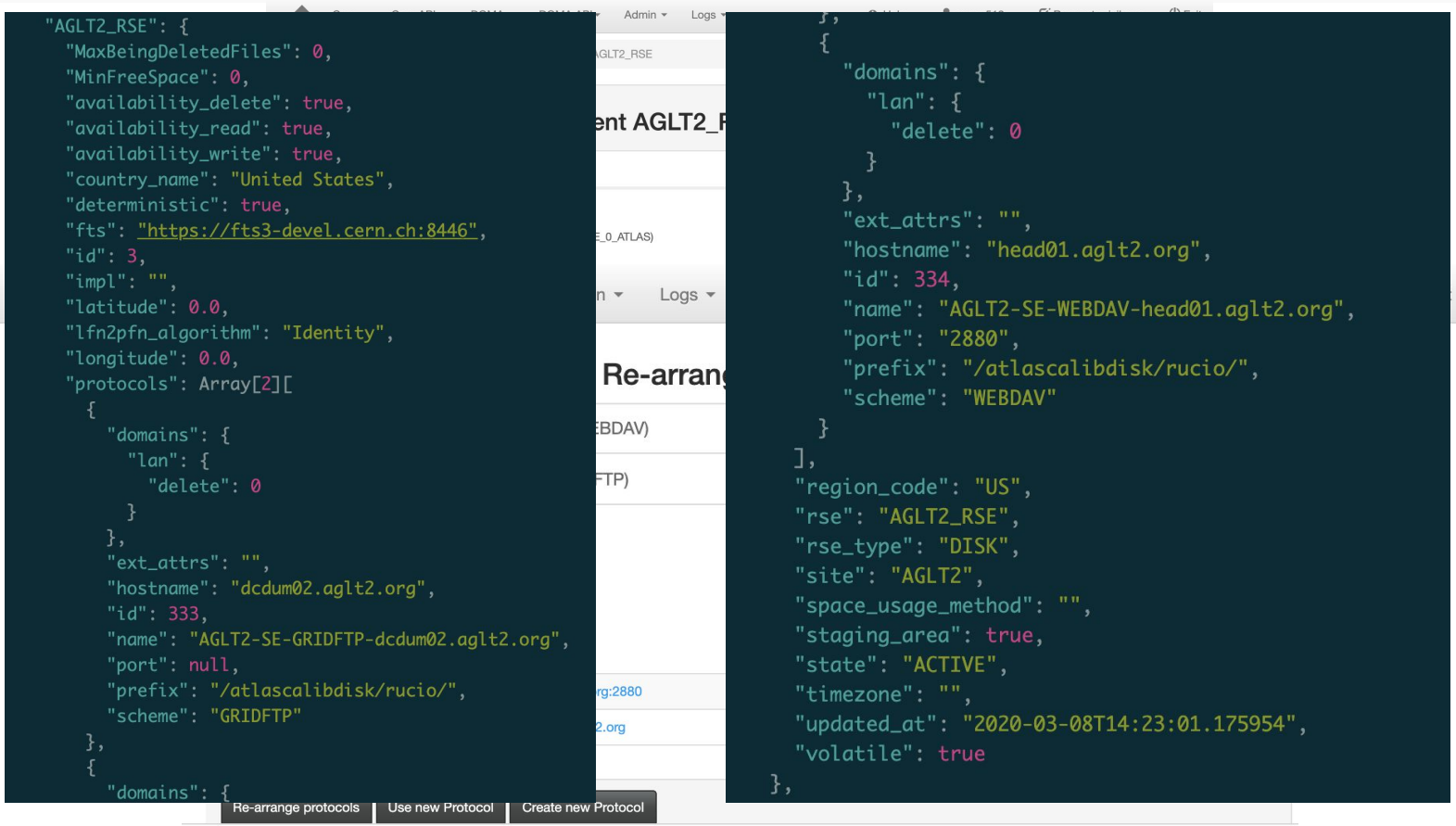
CRIC as an interface for Rucio - RSEs



CRIC as an interface for Rucio - RSEs



CRIC as an interface for Rucio - RSEs

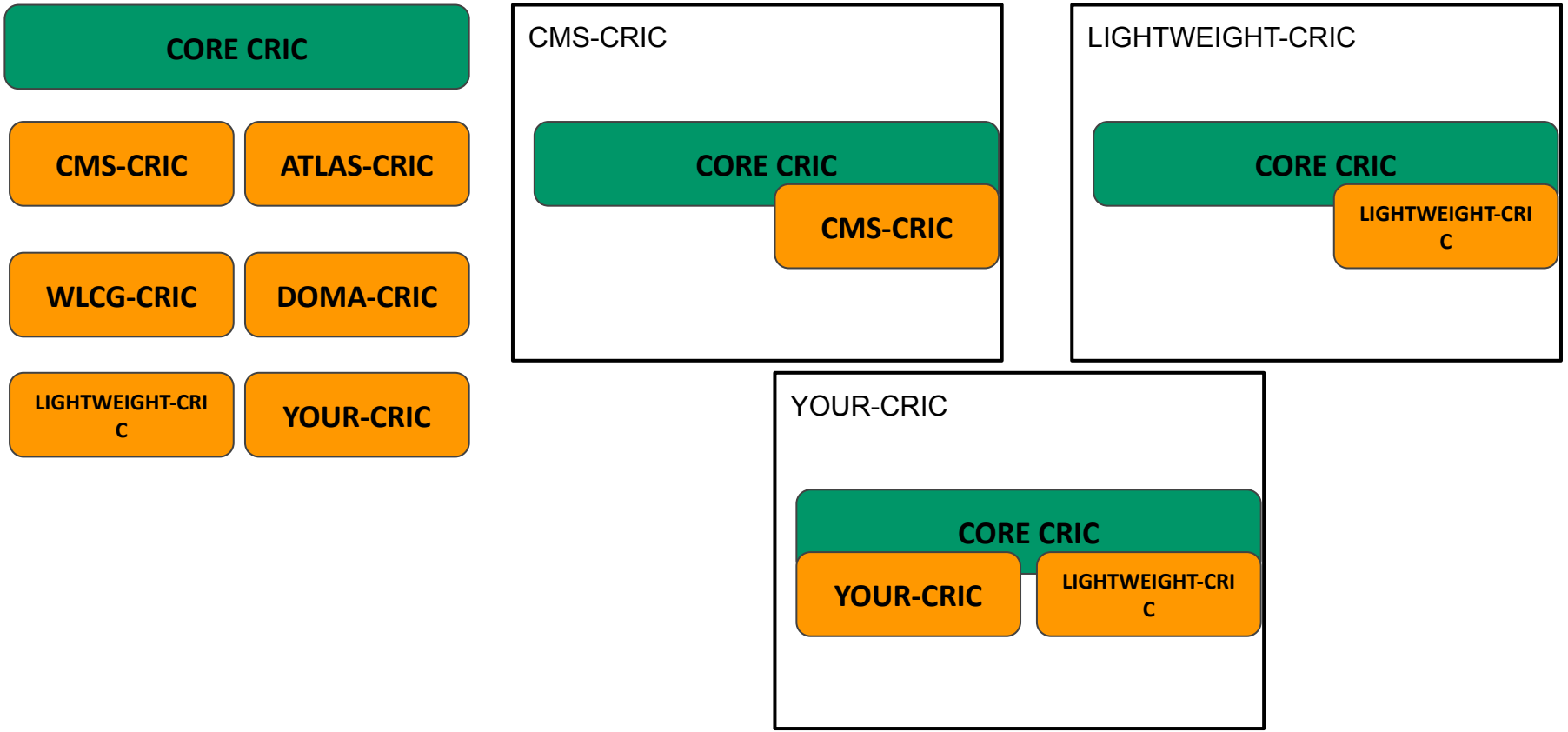


CRIC Deployment

CRIC Deployment

- Puppet module for fast deployment
- You can easily deploy, with minimum configurations, a generic CRIC instance that comes with selected core cric functionality
- You can easily develop your own plugin and customise APIs and interfaces or add extra functionality
  - E.g. adding doma-cric plugin on top gives out of the box Rucio functionalities.
- Detailed documentation is available in our GitLab repo

CRIC Architecture



The DUNE cric experience

- DUNE decided to use CRIC as its main topology system
- In a couple of weeks a new dune-cric plugin was developed and two instances were created: https://dune-cric.cern.ch and https://dune-cric-dev.cern.ch
- DUNE developers were able to easily overwrite the core-cric forms and APIs and:
  - Create pledges for the DUNE experiment
  - Create a custom dune-sites table
  - Create a custom VOFeed file to enable SAM testing for the experiment services
- Overall they reported a very fast and smooth experience

Possible scenarios of using CRIC

Possible scenarios of using CRIC by non-LHC VOs

- VO is sharing WLCG infrastructure with LHC VOs as well as common tools (FTS, Rucio). No need for advanced VO-specific configuration
  - VO resources can be described in the central WLCG CRIC instance. Already partially implemented for VOs which use FTS to improve transfer monitoring applications
- GRID infrastructure like OSG or a particular VO would like to have a dedicated CRIC instance. No need for advanced VO-specific configuration
  - Light-weight generic CRIC instance can be easily configured for this use case
- VO would like to have a dedicated CRIC instance with VO-specific configurations
  - Light-weight generic CRIC instance would need to be extended with VO-specific models

Links

- CRIC instances:
  - https://cms-cric.cern.ch
  - https://atlas-cric.cern.ch
  - https://wlcg-cric.cern.ch
  - https://escape-cric.cern.ch
  - https://dune-cric.cern.ch

- For testing purposes you can log-in at our dev instance: https://wlcg-cric-dev-2.cern.ch and request any permissions.

  Contact cric-devs@cern.ch for any questions or feedback.

Questions?