

Information modelling of computing and storage services/resources

Balázs Kónya, Lund University

Motivation: what is it & why is it needed?

- Information modelling: the process of describing (complex) objects or systems via
 - creating an abstraction layer
 - identification of its key components
 - definition of the component properties
 - specification of the component relations



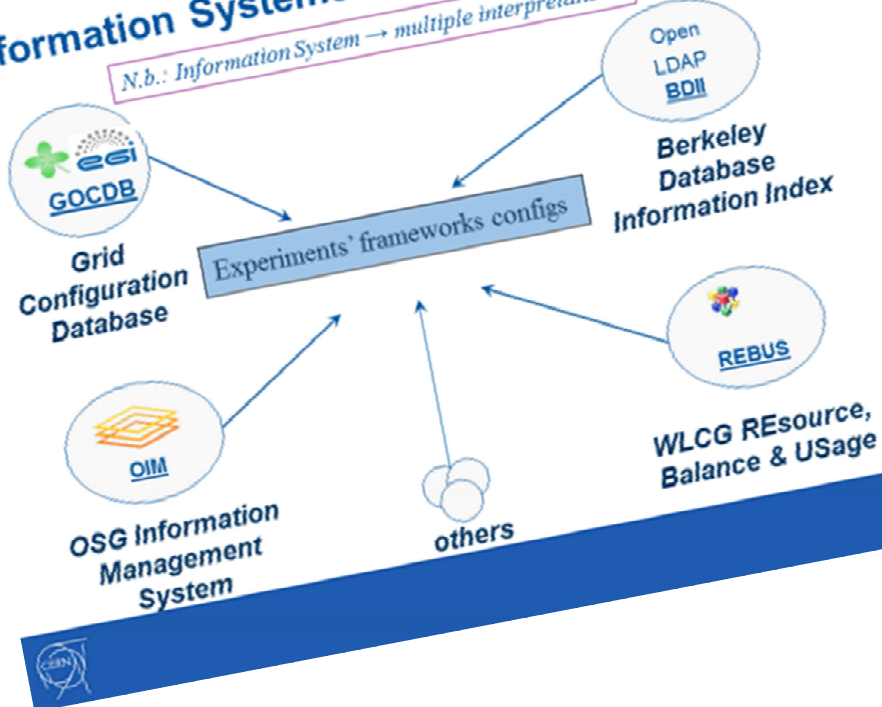
“Now! *That* should clear up a few things around here!”

- Without a common model:
 - there may be a huge difference what we say
 - and what the other understands
 - ... or what we think the other understands
- Critical when information is to be shared among different communities
 - *or computing services/systems*

WLCG landscape: a big world with many players

Information Systems: a big world

N.b.: Information System → multiple interpretations



Experiments frameworks: a big world



some sort of info model would come handy...

Why creating a suitable info model is soo difficult?

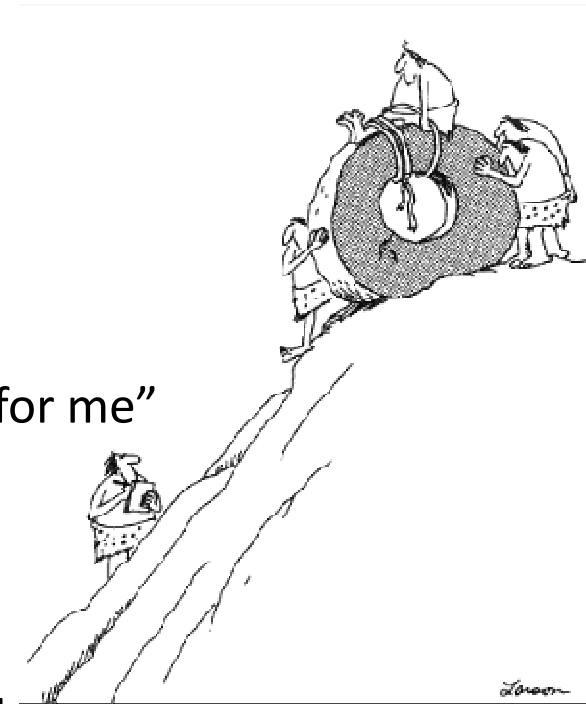
- Time consuming, iterative process
- The task may look trivial at the first time
- Requires commitment and communication from all players
- Cross-domain pre-existing knowledge & readiness for compromise
- Long term investment with large probability of failure
 - the outcome is either too specific or too generic

Therefore, it is often decided

- Let's just quickly put something together that „works for me”

... a clear recepy for disaster:

- defragmentation and continous reinventing the wheel
 - with various „extensions” and „add-ons”



Early experiments in transportation

A previous attempt: GLUE2

A broad community effort started in 2007 and built upon earlier works

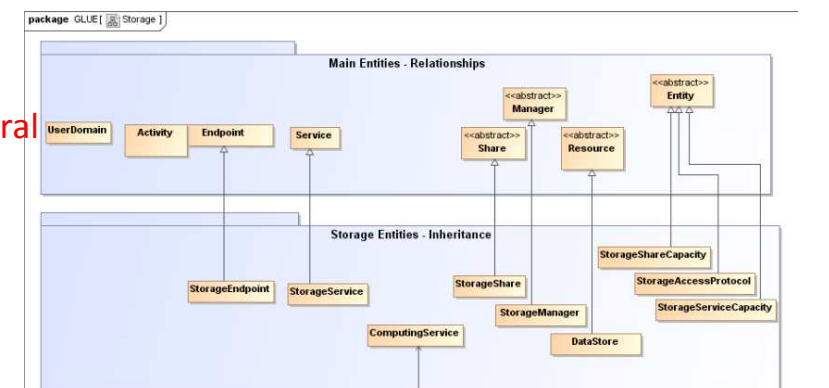
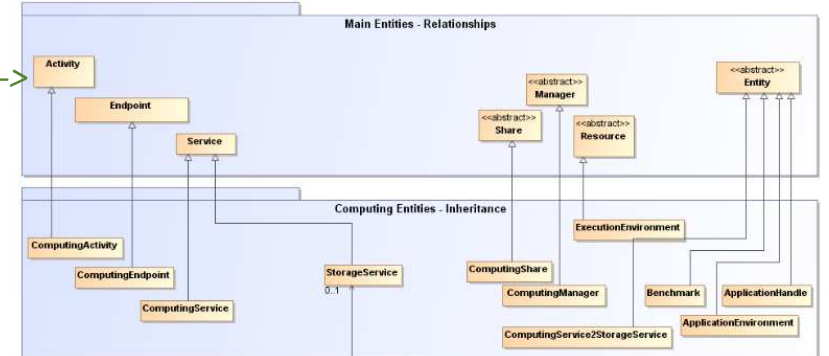
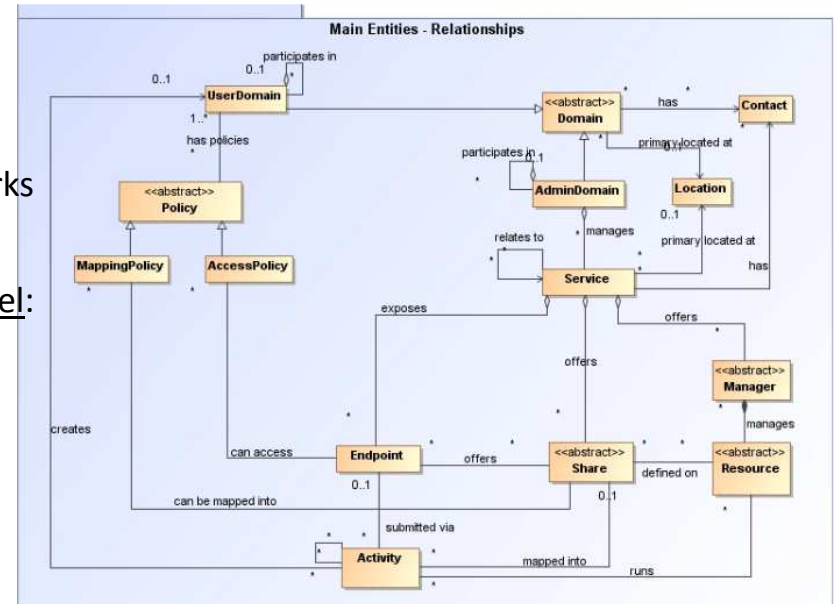
- A still active standardization group within OGF
- GLUE 2 (GFD.147, 2009) specification: a conceptual abstract model:
 - Main entities: *domain, service, endpoint, share, manager, resource, activity, policy*
 - Renderings for *LDAP, XML, SQL and JSON*

Plus:

- Definition of entities & attributes
 - semantics, attr. value syntax, relationship
 - provides enumerations, attribute types
- Main objects are correctly identified
 - Nicely captures the *service -> storageservice, service -> computeservice* inheritance

Minus:

- The community never really figured out how to deploy the model
- Too general and too complex, allows too much flexibility
 - therefore various „profiles“ got created e.g. <http://go.egi.eu/glue2-profile>
- Renderings: not enough iterations
 - Inherited some of the bad BDII choices
- Slow adoption, migration delays (GLUE1.x in US and in several EGI tools)
- Unfortunate extensions hooks
- All in all: GLUE2 somehow accumulated bad reputation and bad user experience



GLUE2 applicability study: space-usage.json (1/4)

Step 0:

- Storage Resource Reporting draft v0.6 by Alessandro & Oliver

```
{
  "capacity_id": "ATLASDATADISK",
  "status": "online/offline",
  "status_message": "The report can not be created because ...",
  "list_of_paths": ["/castor/ads.rl.ac.uk/prod/atlas/stripInput/atlasdatadisk/"],
  "total_space": 50000000000,
  "used_space": 20000000000,
  "num_files": 123456,
  "time_stamp": 1447936989,
  "capacity_id": "ATLASSCRATCHDISK",
  ...
},
{
  "capacity_id": "ATLASLOCALGROUPDISK",
  ...
}
```

GLUE2 applicability study: space-usage.json (2/4)

Step 1&2:

„Since you mentioned standardizing, would you consider using an existing standard for describing services: GLUE? One can describe most of what you want already with GLUE and it has a JSON rendering. Here's a rough idea of how the output would look like:„

- StorageShare & StorageShareCapacity are used
- The objects are linked via Associations
- Missing info added via „OtherInfo“ extension capability
 - Note Path (!)

```
{
  "StorageShare": [
    {
      "Associations": {
        "StorageShareCapacityID": [
          "atlas:big-site.example.org:ATLASDATADISK-usage"
        ]
      }
      "CreationTime": "2016-11-02T12:16:03Z",
      "ID": "atlas:big-site.example.org:ATLASDATADISK",
      "SharingID": "ATLASDATADISK",
      "ServingState": "production", # or closed|draining|queueing
      "Path": "/atlas/data",
      "AccessLatency": "ONLINE",
      "Tag": "ATLASDATADISK",
      "OtherInfo": [
        "NumberOfFiles=123456",
        "Path=/vo/atlas/data",
        "Path=/users/atlas-vo/data",
        "Problem=The report cannot be created because ...",
      ]
    },
    <skipping>
  ],
  "StorageShareCapacity": [
    {
      "Associations": {
        "StorageShareID": "atlas:big-site.example.org:ATLASSCRATCHDISK"
      }
      "CreationTime": "2016-11-02T12:16:03Z",
      "ID": "atlas:big-site.example.org:ATLASSCRATCHDISK:usage",
      "TotalSize": 5000, # NB. values are in GB (not GiB)
      "UsedSize": 2000
    },
    <skipping>
  ]
}
```

GLUE2 applicability study: space-usage.json (3/4)

Step 3:

„no need for such a complex structure and we could merge some of the GLUE2 objects”

- Assume one-to-one relation of Share & ShareCapacity
- Add the useful attributes from StorageShareCapacity into StorageShare
- Get rid of Associations
- That is: reuse GLUE2 attributes but diverge from, sacrifice GLUE2 json rendering structures

```
"StorageShare": [  
  "CreationTime": "2016-11-02T12:16:03Z",  
  "ID": "atlas:big-site.example.org:ATLASDATADISK",  
  "SharingID": "ATLASDATADISK",  
  "ServingState": "production", # or closed|draining|queueing  
  "Path": "/atlas/data",  
  "AccessLatency": "ONLINE",  
  "Tag": "ATLASDATADISK",  
  "OtherInfo": [  
    "NumberOfFiles=123456",  
    "Path=/vo/atlas/data",  
    "Path=/users/atlas-vo/data",  
    "Problem=The report cannot be created because ...",  
  ]  
  "TotalSize": 3000, # these are the two capacity attributes  
  "UsedSize": 1298 # these are the two capacity attributes  
],
```


GLUE2 applicability study: space-usage.json (4/4)

Step 104:

„here comes the cleaned up version of the next iteration example for storage space reporting based on GLUE2 terminology. The storage space is modelled by the (modified) GLUE2 StorageShare entity”

- Composed of VALID GLUE2 attributes
- It was necessary to introduce additional (new) attribute: NumberOfFiles
- Violates GLUE2 json structure
- Redefines Path as multivalued attribute

Conclusion:

It is not feasible to use the current GLUE2 JSON rendering while it can be beneficial to re-use some of the attributes.

```
{  "StorageShare": [
    "PolicyRule": "vo:ATLAS",
    "CreationTime": "2016-11-02T12:16:03Z",
    "ID": "atlas:big-site.example.org:ATLASDATADISK",
    "SharingID": "HUMAN_READABLE_NAME_OF_THE_SHARE",
    "ServingState": "production",
    "Path": ["/atlas/data", "/users/atlas-vo/data", "/another/path"],
    "AccessLatency": "ONLINE",
    "Tag": ["ATLASDATADISK", "SCRATCHDISK"],
    "TotalSize": 3000,
    "UsedSize": 1298,
    "NumberOfFiles": 123456,
    "OtherInfo": "The report was created successfully, all green."
  ]
},
```

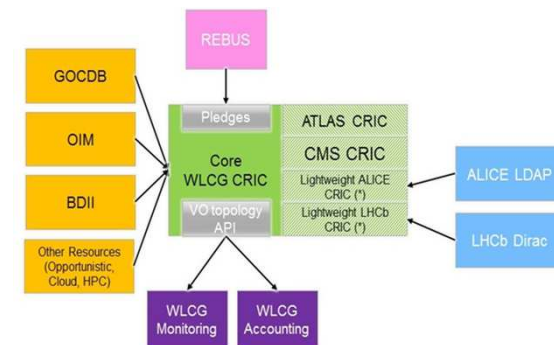
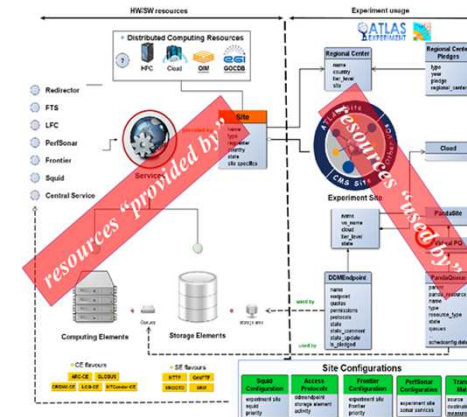
A more suitable approach for WLCG:

The information model:

- ensures clear separation between „provided by” and „consumed by”
- abstraction from physical resource to experiment frameworks
- integrates configuration and status information (resources, services)
- describes the experiment topology in terms of computing infrastructure

Main goals, the driving force behind the info model:

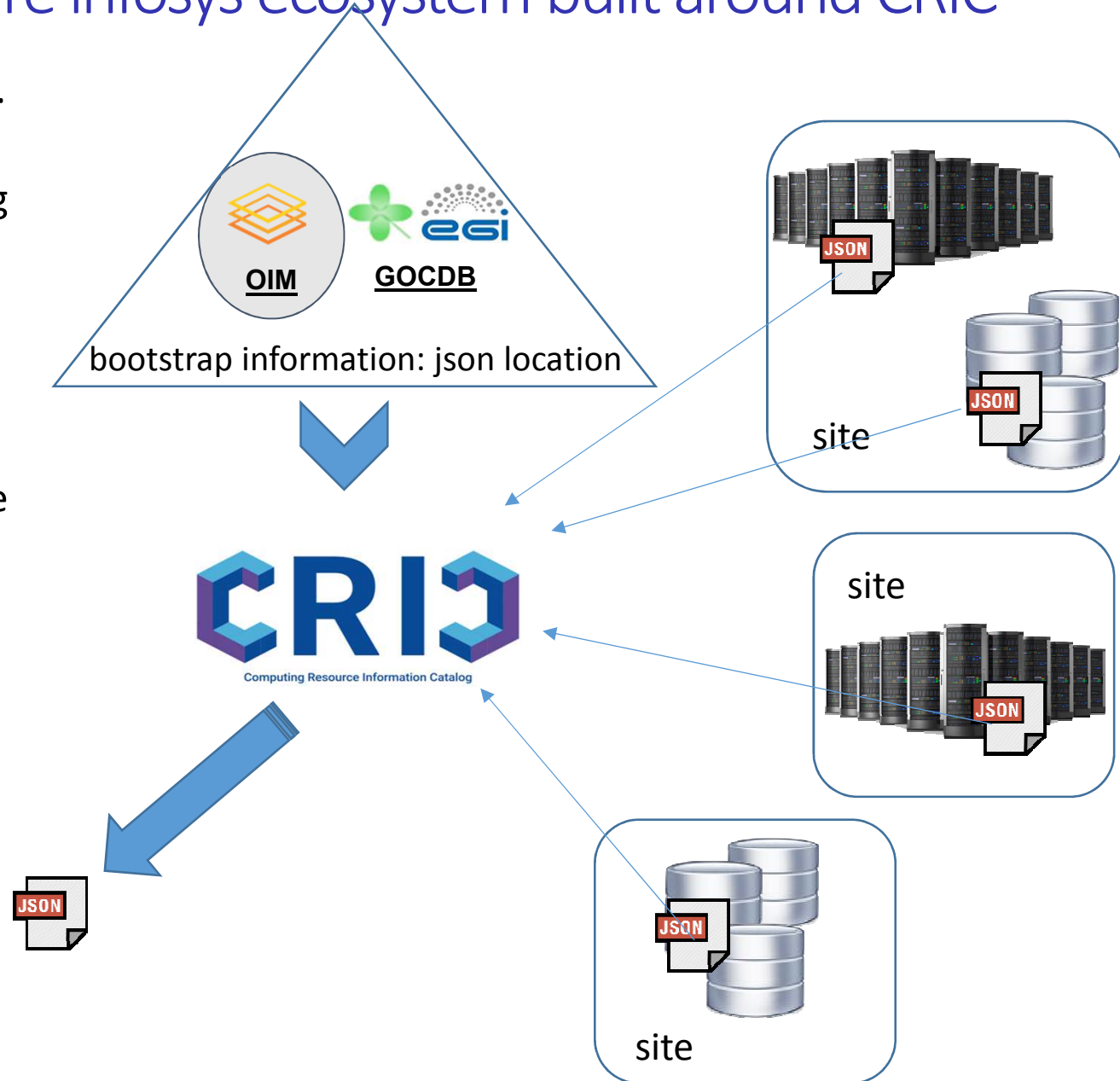
- create the CRIC framework which would allow various groups to define their own topology
- a light CRIC core + simplified CRIC experiments plugin, with basic REST (JSON) exports of e.g. compute or storage units



(*) Maintained by WLCG to store very simple experiment topology information (i.e. experiment names)

A vision of a future infosys ecosystem built around CRIC

- Compute, Storage, etc.. services deployed at sites are described by „simple json” according to info model
- Index services (e.g. GOCDB, OIM) maintain bootstrap info for json location and the „fetch protocol”
- CRIC aggregates service descriptions & maintains all the topology information
- Info consumers obtain exported jsons from CRIC, structured according to use-cases



Composable info models is a must!

Info model: finding the proper balance

- Describe only those system characteristics that are really needed
- Dynamic - static separation
- Keep simple things simple
- BUT avoid oversimplification
- Allow flexibility without losing structure
- ITERATIVE process



Pre-CRIC storage models

... either too little (too simplified)



Service: atlassrm-fzk.gridka.de - SRM

SRM dCache endpoint dedicated to ATLAS.

System

Host name	PROTECTED - Auth required
IP Address	PROTECTED - Auth required
IP v6 Address	PROTECTED - Auth required
Operating System	PROTECTED - Auth required
Architecture	PROTECTED - Auth required
Contact E-Mail	PROTECTED - Auth required

Grid Information

Host DN	PROTECTED - Auth required
URL	
Parent Site	FZK-LCG2
Scope Tags	atlas, EGI, tier1, wlcg

Project Data

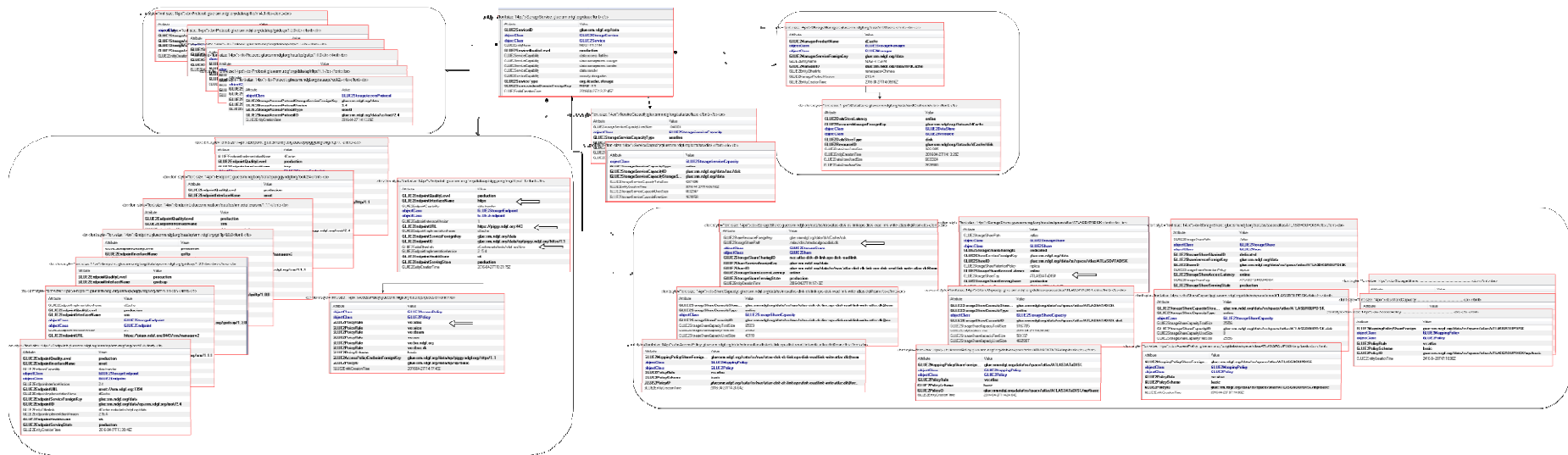
Production Level	✓
Beta	✗
Monitored	✓

Service Groups this Service Belongs To

Service Endpoints (endpoints?)

Name	URL	Interface Name
------	-----	----------------

Pre-CRIC storage models ... or too much!!



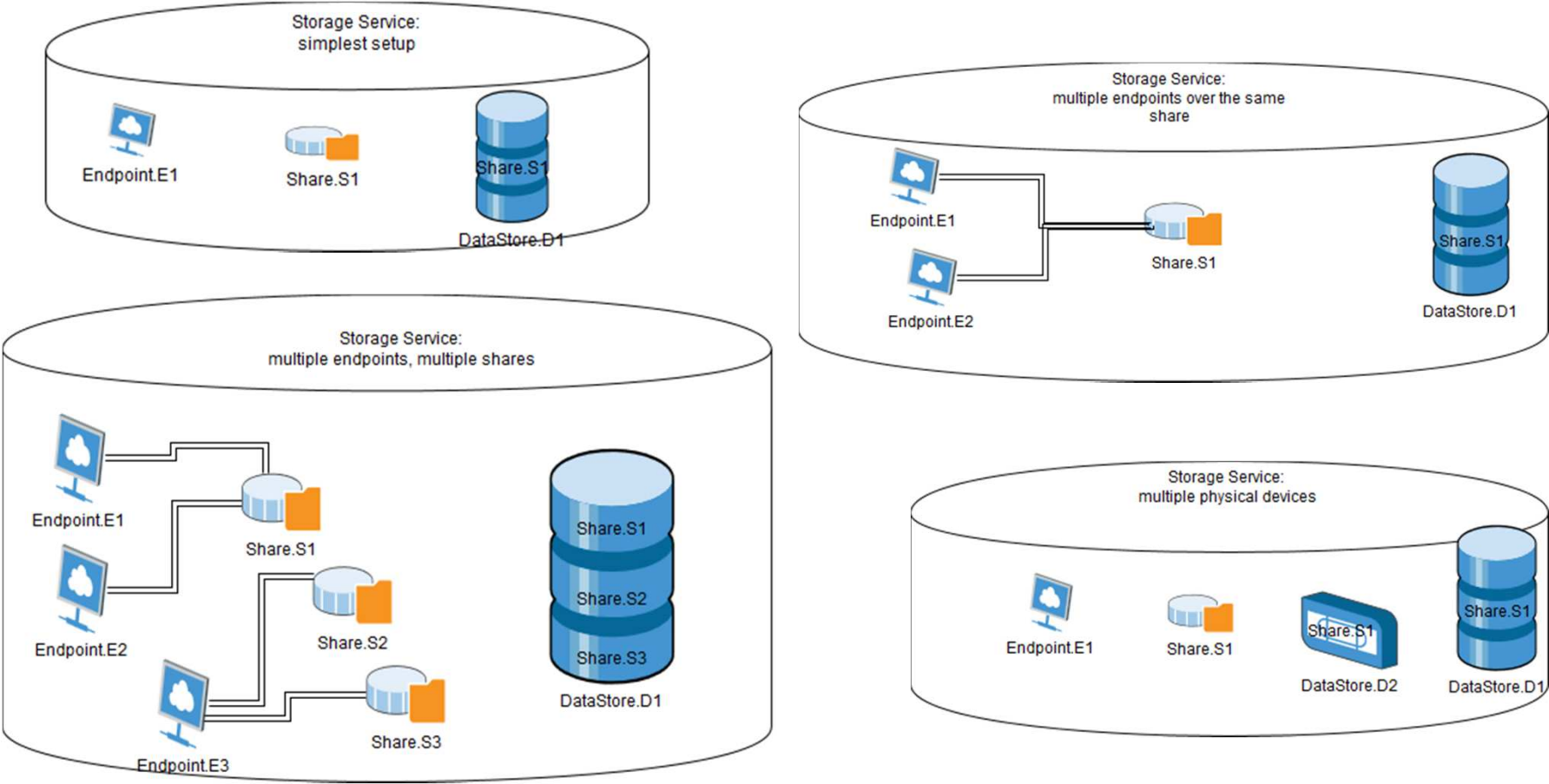
The complete set of GLUE2 records taken from LDAP-BDII describing the NDGF-T1 storage system

<https://twiki.cern.ch/twiki/pub/AtlasComputing/AtlasGridInformationSystemStorageDeclaration/NDGF-T1-GLUE2.svg>

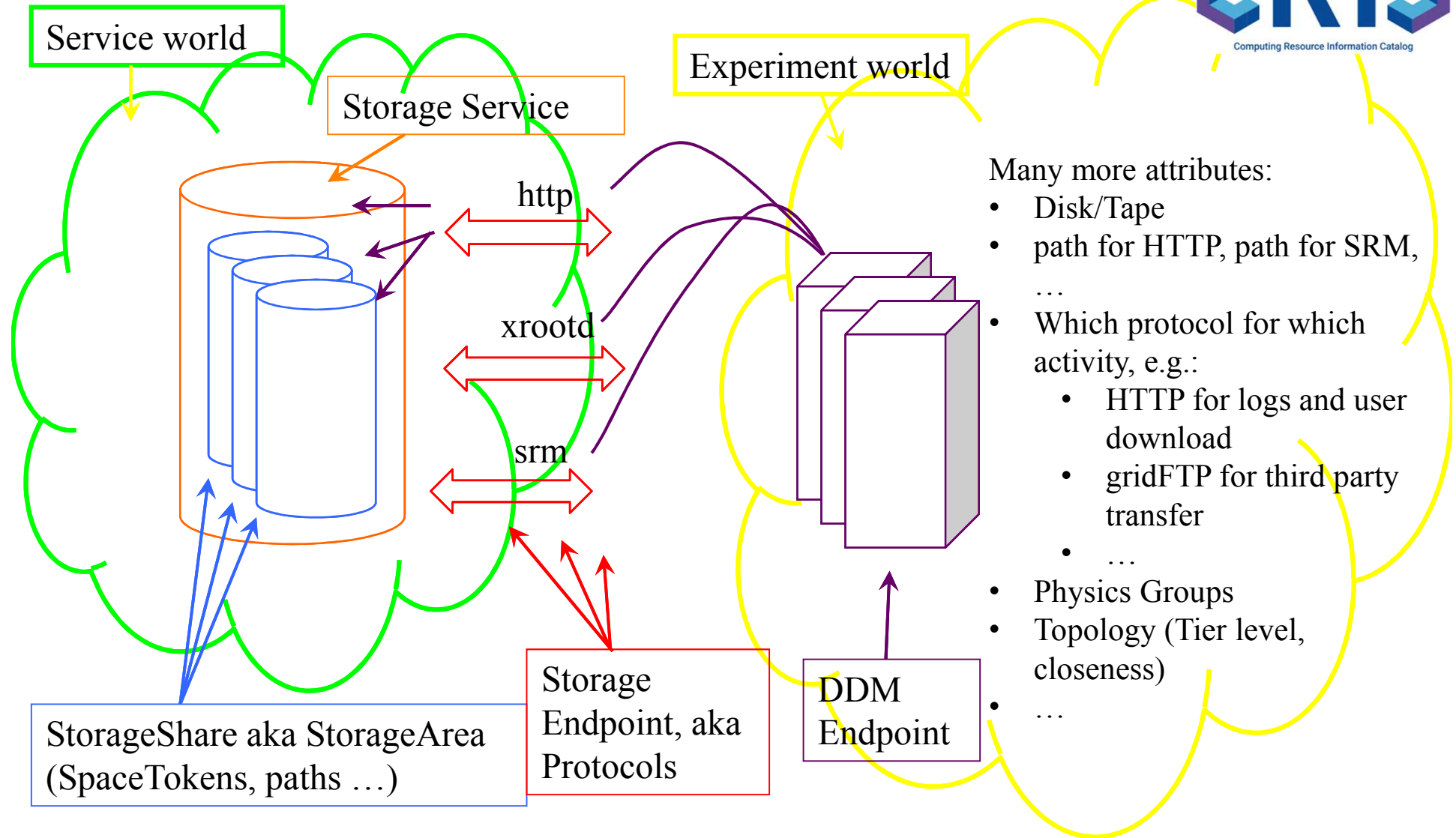
CRIC storage representation details: the set of objects

- **StorageService**: represents the entire storage system that makes its Shares, defined within a DataStore, available via Endpoints. Static information.
 - Attributes: name, id, servicetype, implementation, capabilities, qualitylevel, storagecapacity ...
- **StorageShare**: represents a logical storage area, a part of the storage capacity, allocated on DataStore(s) for a specific user group or use case. May contain dynamic information (e.g. space usage)
 - Attributes: name, id, policyrules, path, assignedendpoints, servingstate, accessmode, maximumlatency, retentionpolicy, defaultlifetime, maximumlifetime, expirationmode, totalsize, usedsize, numberoffiles, message
- **StorageEndpoint**: represents the network interface to the storage system that maybe contacted to manage and access data stored in the StorageShare(s) of DataStore(s). May contain dynamic information (e.g. healthstate or servingstate)
 - Attributes: name, id, endpointurl, assignedshares, interfacetype, interfaceversion, capabilities, qualitylevel, servingstate, healthstate, message
- **DataStore**: describes a homogeneous instance of a physical storage extent (e.g. a tape or a disk server). Static information.
 - Attributes: name, id, datastoretype, latency, totalsize, vendor, message

CRIC representation details: storage object relations



CRIC representation details: Storage Endpoint diagram



and a corresponding storage JSON document (partial):

```
{"storageservice": {  
  "name": "FZK-ATLAS",  
  "id": "global unique",  
  "servicetype": "one of the agreed service types",  
  "implementation": "dcache-2.13.51",  
  "capabilities": [feature1, feature2, feature3],  
  "qualitylevel": "production",  
  "storagecapacity": {},  
  "storageendpoints": [  
    {"name": "atlassrm",  
     "id": "global unique",  
     "endpointurl": "http://srm-fzk.gridka.de:8443/srm/managerv2",  
     "interfacetype": "srm",  
     "interfaceversion": "2.2",  
     "capabilities": ["data.mgt.transfer", "data.mgt.storage"],  
     "qualitylevel": "production",  
     "servicestate": "production",  
     "healthstate": "ok",  
     "assignedshares": ["all"],  
     },  
    {"name": "atlasgsiftp",  
     "id": "global unique",  
     "endpointurl": "gsiftp://f01-060-114-e.gridka.de:2811",  
     ...  
     },  
    {...}  
  ],  
}
```

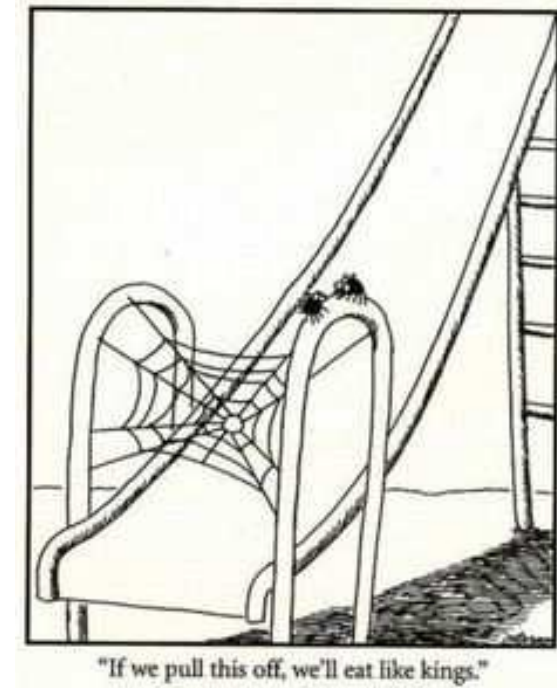
```
"storageshares": [  
  {"name": "ATLASDATADISK",  
   "id": "global unique",  
   "policyrules": ["defaultpermission", "acl1", "acl2"],  
   "servicestate": "open",  
   "accessmode": ["read/0", "delete/2"],  
   "totalsize": 26789588699,  
   "usedsize": 123564878,  
   "numberoffiles": -1  
   "path": ["/../atlasdatadisk/", "/../atlasdata/"],  
   "assignedendpoints": ["all"],  
   ...  
  },  
  {"name": "ATLASSCRATCHDISK",  
   "id": "FZK-ATLAS.S2",  
   "path": ["/storage/data/scratch/"],  
   "assignedendpoints": ["atlassrm"],  
   ...  
  },  
  {"name": "ATLASGROUPDISK",  
   "id": "FZK-ATLAS.S3",  
   "path": ["/storage/data/atlasgroup/"],  
   "assignedendpoints": ["atlasgsiftp"],  
   ...  
  },  
  {...}  
]
```

https://twiki.cern.ch/twiki/pub/AtlasComputing/AtlasGridInformationSystemStorageDeclaration/storage_service.json

Summary & outlook

- WLCG would benefit from a common information model
- The existing models (e.g. GLUE) were found not being suitable
- CRIC, the infosys component of WLCG, offers a better solution
- The work is ongoing as an iterative process
 - In addition to storages, other core CRIC objects such as compute service & site being modelled
 - Mixture of bottom-to-top & top-to-bottom approach
- Based on the early feedbacks, there is a hope this time the info modelling is being done properly (at least for WLCG)

It is by far not a trivial exercise to catch this lunch.....



credits:

- cartoons are taken from „The Far side” by Gary Larson
- CRIC pictures are from CRIC team presentations

backup

simplified JSON for compute resource

```
{ "AdminDomain": [
  {
    "Name": "ATLAS Site 1",
    "Owner": "University of Stockholm",
    "Location.Country": "Sweden",
    "Location.Place": "Stockholm",
    "Location.Latitude": "35",
    "Contact": "email or web address",
    {"ComputingService": [
      {
        "Name": "Monolith",
        "Type": "Traditional batch cluster",
        "Qualitylevel": "Production",
        "Capability": "automatic data staging"
        ...
      }
    ]},
    {"ComputingEndpoint": [
      {
        "URL": "https:url_to_the_ce",
        "InterfaceName": "ARC-CE",
        "InterfaceVersion": "5.0.3"
      }
    ]},
    {"ComputingShare": [
      {
        "Name": "ATLAS Prod",
        "MappingQueue": "longdedic",
        "SchedulingPolicy": "preemptable",
        "MaxWalltime": "16000",
        "MaxMainMemory": "12"
      }
    ]},
    {"ExecutionEnvironment": [
      {
        "Name": "Fat memory node",
        "PhysicalCPUs": "2",
        "LogicalCPUs": "12", // this corresponds to the cores
        "TotalInstances": "5", // number of WNs of this type
        "MainMemorySize": "24GB",
        "CPUVendor": "Intel",
        "ConnectivityIn": "True",
        ...
      }
    ]},
    {"ExecutionEnvironment": [
      {
        "Name": "WN Type2",
        "PhysicalCPUs": "1",
        "LogicalCPUs": "4", // this corresponds to the number of cores
        "TotalInstances": "300", // number of WNs of this type
        "MainMemorySize": "12GB",
        "CPUVendor": "Intel",
        "ConnectivityOut": "False",
        "Scratchdisksize": "800" //not a GLUE2 attribute
        ...
      }
    ]}
  ]
}
```