A data transfer system for MAGIC based on gLite FTS and multiVAC

Roger Firpo Curcoll, Pau Tallada,

Nadia Tonello, Christian Neissner, Manuel Delfino, Javier Rico, Ignasi Reichardt, MAGIC collaboration

PIC
port d'informació
científica

MAGIC
Major Atmospheric
Gamma Imaging
Cerenkov Telescopes

summary

- Introduction: The MAGIC data center (storage)
- Current data transfer + storage + access
- A new data transfer system
  - gLite FTS
  - multiVAC
- Conclusions

MAGIC telescopes

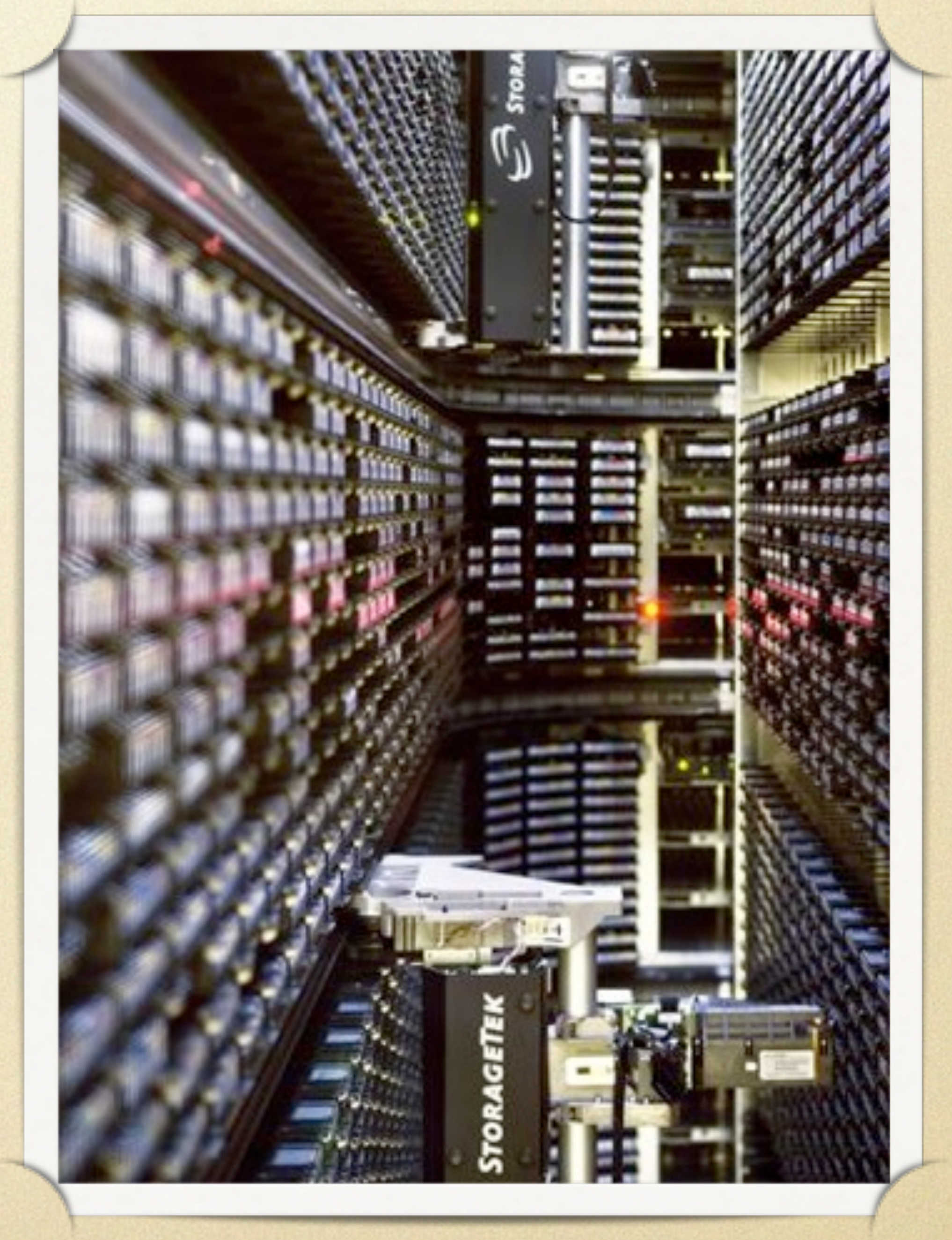
- Cherenkov telescopes ~30GeV to 10TeV γ-ray
- Observatory in La Palma: Canary Islands
- Observing since 2004, second telescope on 2009

data production @LP

- Data volume as of 2011
  - raw data: ~125 TB per year
  - OnSite Analysis: + ~16 TB per year
- ~30 different kinds of data (data + logs) to be transferred to the data center @PIC

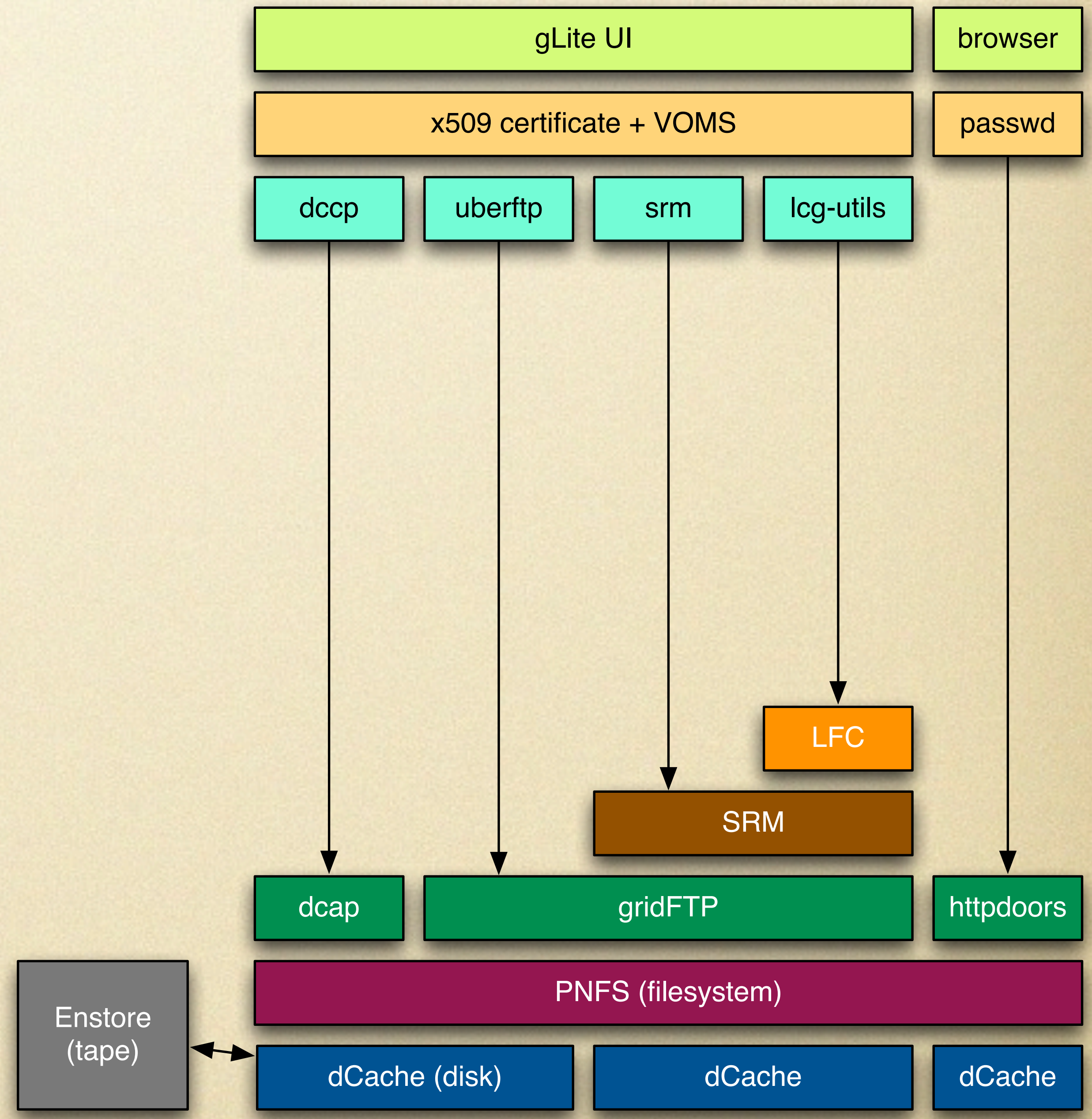
data center @PIC

- Hosted in PIC, Barcelona since 2007
- Storage (150 + 350 TB)
- Data access
- Data transfer from LP
- Computing: official + users' analysis



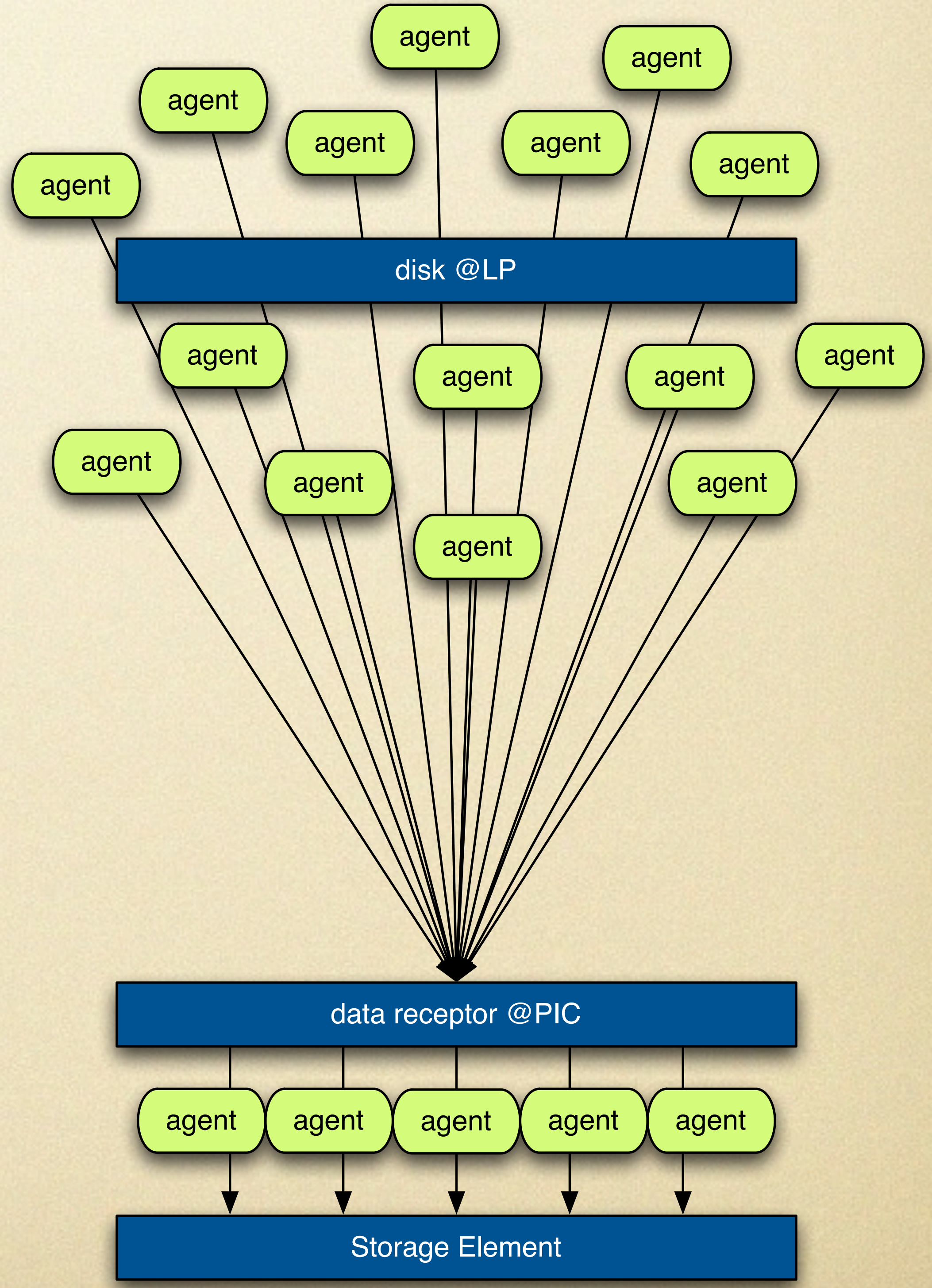
storage & data access

- Multiple protocol and authentication options
- Many disk pools, but single filesystem
- Transparent access to tape library
- POSIX*



current data transfer

- 1 agent per data type
- Temporal NFS disk @PIC to collect data, later moved to Storage
- Bottlenecks, multiple error points, poor monitoring & admin
- Raw data by air mail



need for a change

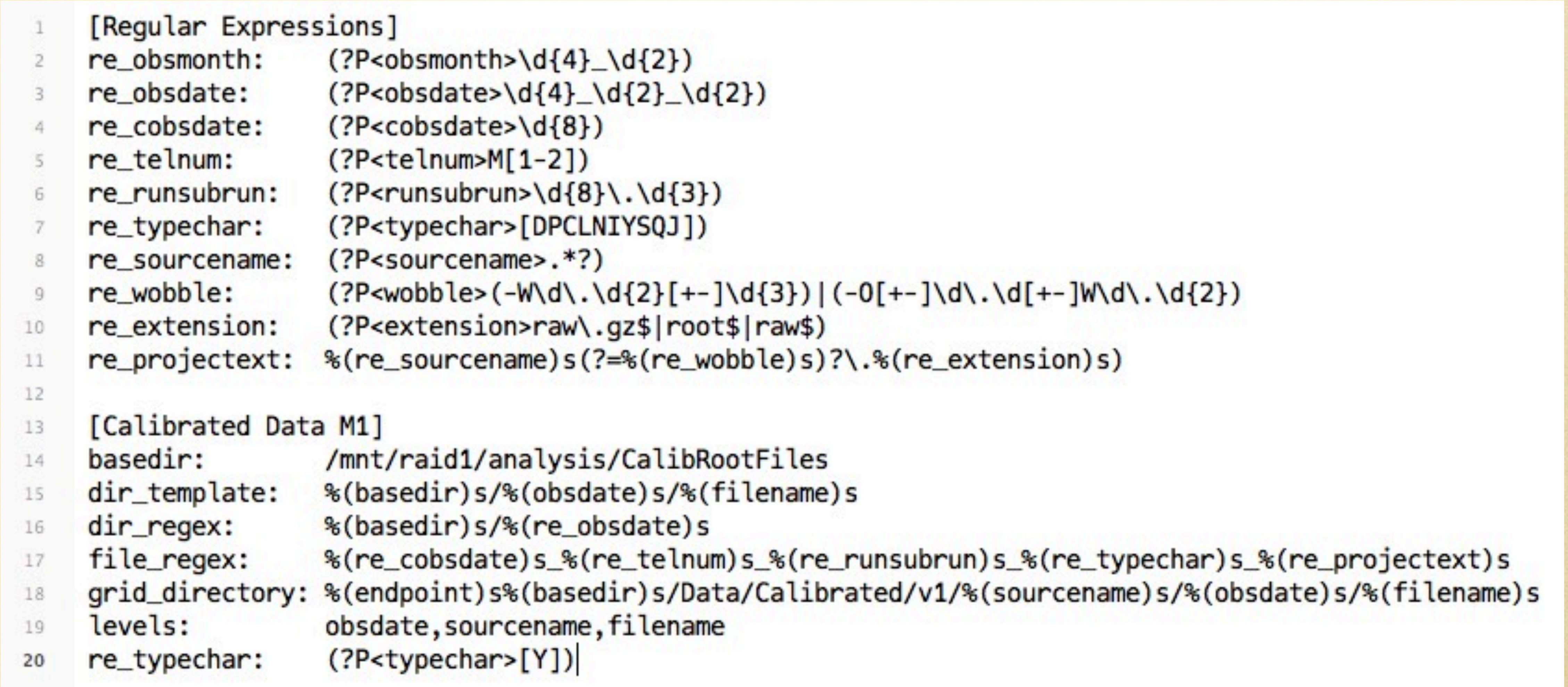
- This system is a legacy from the ‘early days’
- Many changes since 2004:
  - Data center completely renewed
  - 2nd telescope in operation: #agents x2!
  - 5 years of experience
- It’s time to review it!

a new system

- Must deal with 4 key points:
  - **Which** data must be transferred to PIC and **where** this data is and will be copied
  - **How much** data is there
  - **When** this data is ready/safe to transfer
  - **How** to transfer the data to PIC (and mark it as transfered afterwards)

which and where?

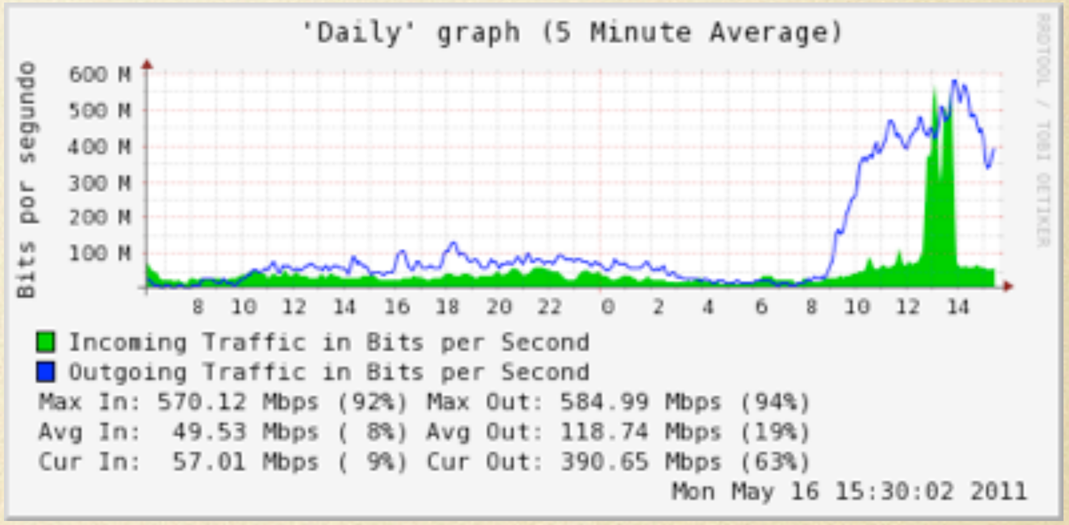
- Common configuration file: simple and intuitive



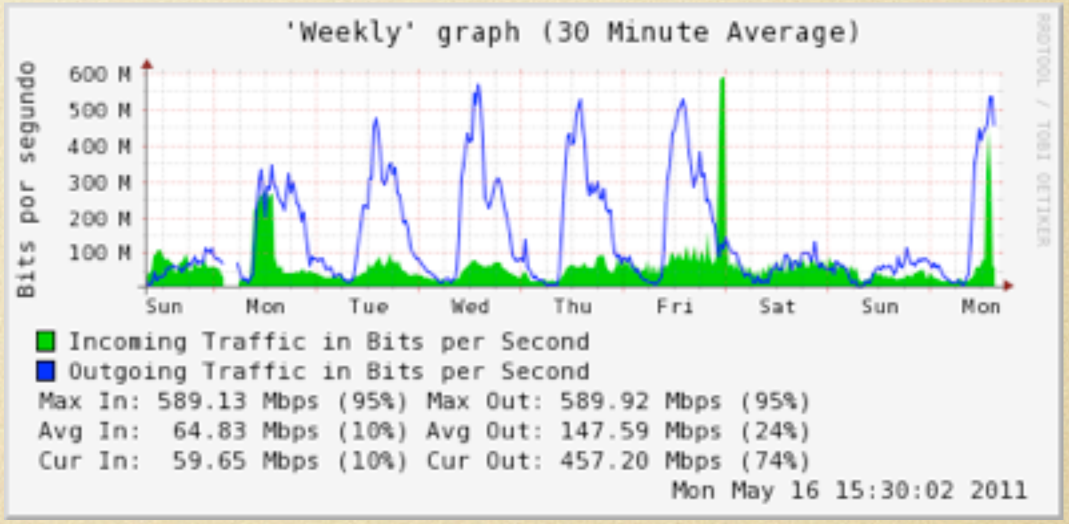
```
[Regular Expressions]
re_obsmonth:    (?P<obsmonth>\d{4}_\d{2})
re_obsdate:     (?P<obsdate>\d{4}_\d{2}_\d{2})
re_cobsdate:    (?P<cobsdate>\d{8})
re_telnum:      (?P<telnum>M[1-2])
re_runsubrun:   (?P<runsubrun>\d{8}\.\d{3})
re_typechar:    (?P<typechar>[DPCLNIYSQJ])
re_sourcename:  (?P<sourcename>.*?)
re_wobble:      (?P<wobble>(-W\d\.\d{2}[+-]\d{3})|(-O[+-]\d\.\d[+-]W\d\.\d{2})
re_extension:   (?P<extension>raw\.gz$|root$|raw$)
re_projectext:  %(re_sourcename)s(?=%(re_wobble)s)?\.%(re_extension)s)

[Calibrated Data M1]
basedir:        /mnt/raid1/analysis/CalibRootFiles
dir_template:   %(basedir)s/%(obsdate)s/%(filename)s
dir_regex:      %(basedir)s/%(re_obsdate)s
file_regex:     %(re_cobsdate)s_%(re_telnum)s_%(re_runsubrun)s_%(re_typechar)s_%(re_projectext)s
grid_directory: %(endpoint)s%(basedir)s/Data/Calibrated/v1/%(sourcename)s/%(obsdate)s/%(filename)s
levels:         obsdate,sourcename,filename
re_typechar:    (?P<typechar>[Y])
```

how much?

- Volume: ~140 TB/year
- Mean: ~50 Mbps for all year data (80%uptime)
- Peaks: ~300 Mbps for winter nights (in 24h)
- Line: 600 Mbps, ~10 Gbps before 2012



```
Max In: 570.12 Mbps (92%) Max Out: 584.99 Mbps (94%)
Avg In:  49.53 Mbps ( 8%) Avg Out: 118.74 Mbps (19%)
Cur In:  57.01 Mbps ( 9%) Cur Out: 390.65 Mbps (63%)
                              Mon May 16 15:30:02 2011
```



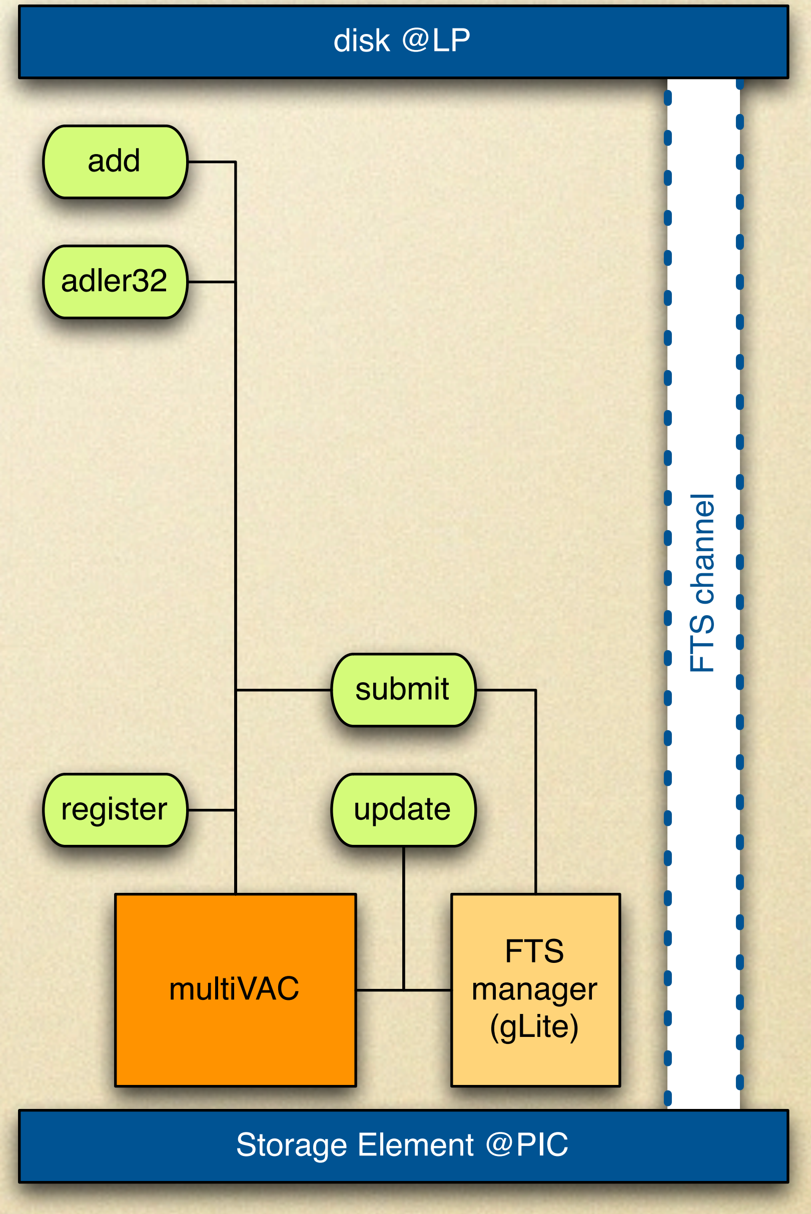
```
Max In: 589.13 Mbps (95%) Max Out: 589.92 Mbps (95%)
Avg In:  64.83 Mbps (10%) Avg Out: 147.59 Mbps (24%)
Cur In:  59.65 Mbps (10%) Cur Out: 457.20 Mbps (74%)
                              Mon May 16 15:30:02 2011
```

when?

- Data ready conditions depend on data type:
  - Raw & subsystem data: observation is over
  - Analysis: ask to OSA manager, webservice
  - ...

how?

- DataTransfer app
- Common workflow for all data types
- 1 agent per step
- Direct channel
- Proper monitoring and management



DataTransfer app

- Extension of multiVAC classes with methods to deal with data transfer using FTS
- Includes the central db and the agents:
  - watch & add files, compute adler32, submit FTS jobs, update status from FTS, register file
- Deals with all data types with simple cfg file
- Developed at PIC

FTS

- FTS = gLite File Transfer Service (by EGEE)
- Between SRM endpoints: BeStMan server in LP
- Queue of files defined by origin, destination & checksum
- Limited protection against errors (n retries)
- Provides information on individual files

multiVAC

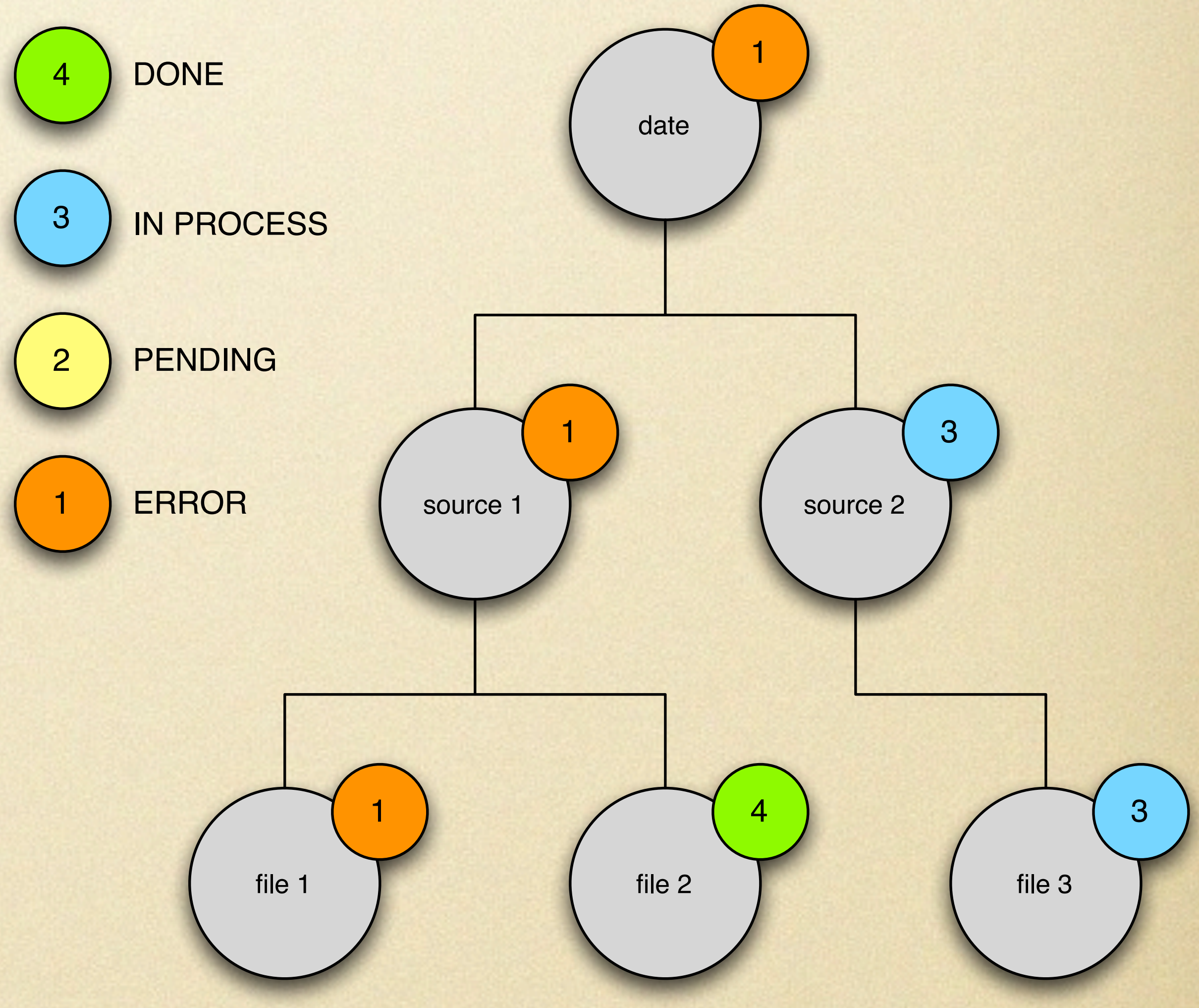
- multiVAC (Versatile Application Core)
- Coded in Python 2.4 (req. by gLite)
- PostgreSQL database
- Db access using sqlalchemy
- Developed at PIC

multiVAC

- Hierarchical collection of elements, with a defined state and arbitrary tags (metadata)
- Workflow: finite-state machine with priorities
- Calculated states based on children & priority
- Versatile: interesting for applications besides data transfer, like computing, monitoring, ...

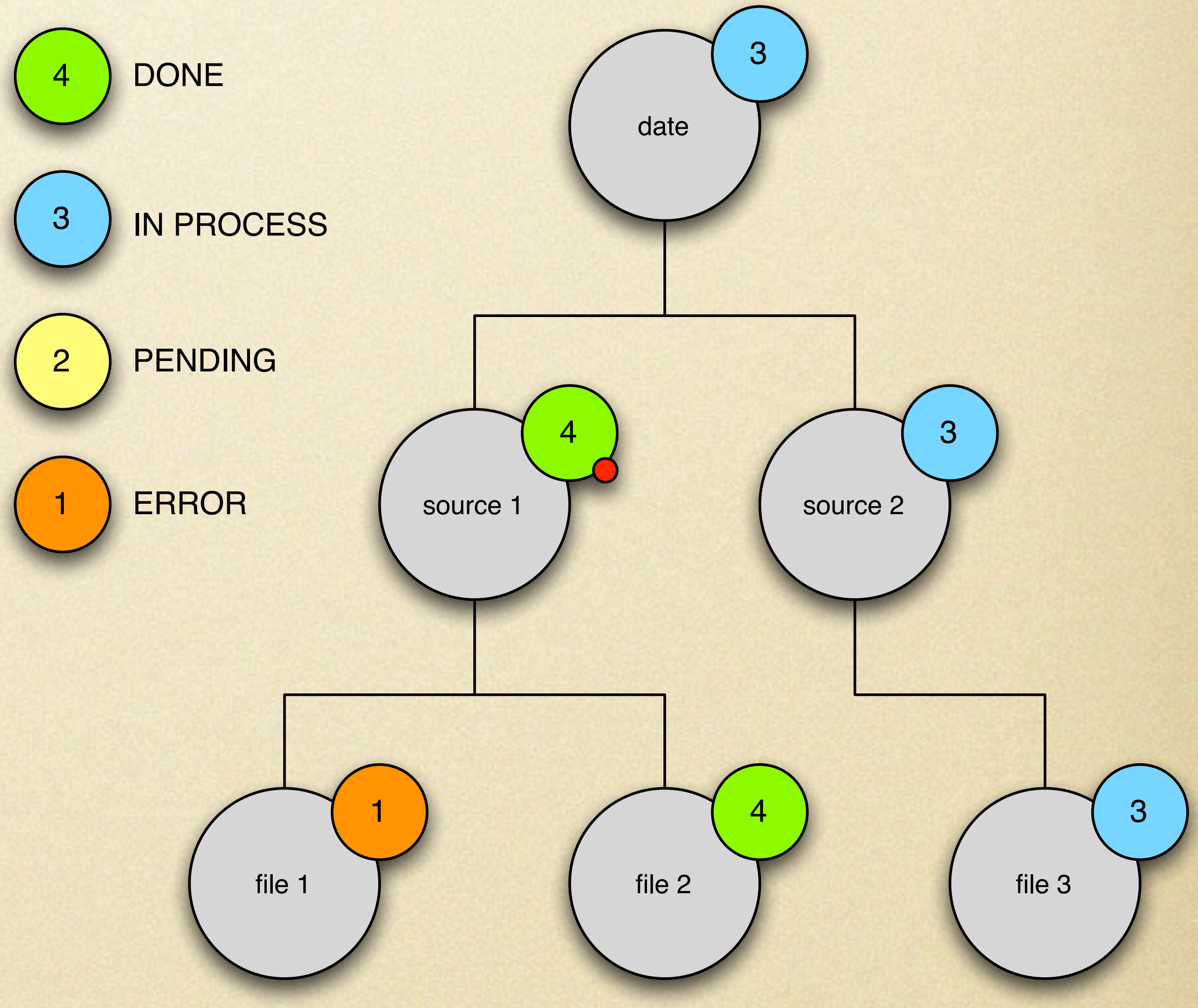
multiVAC element tree

- Hierarchical structure
- Status changes propagate upwards, following priorities
- Easy to track problems
- Can fix element status to avoid propagation



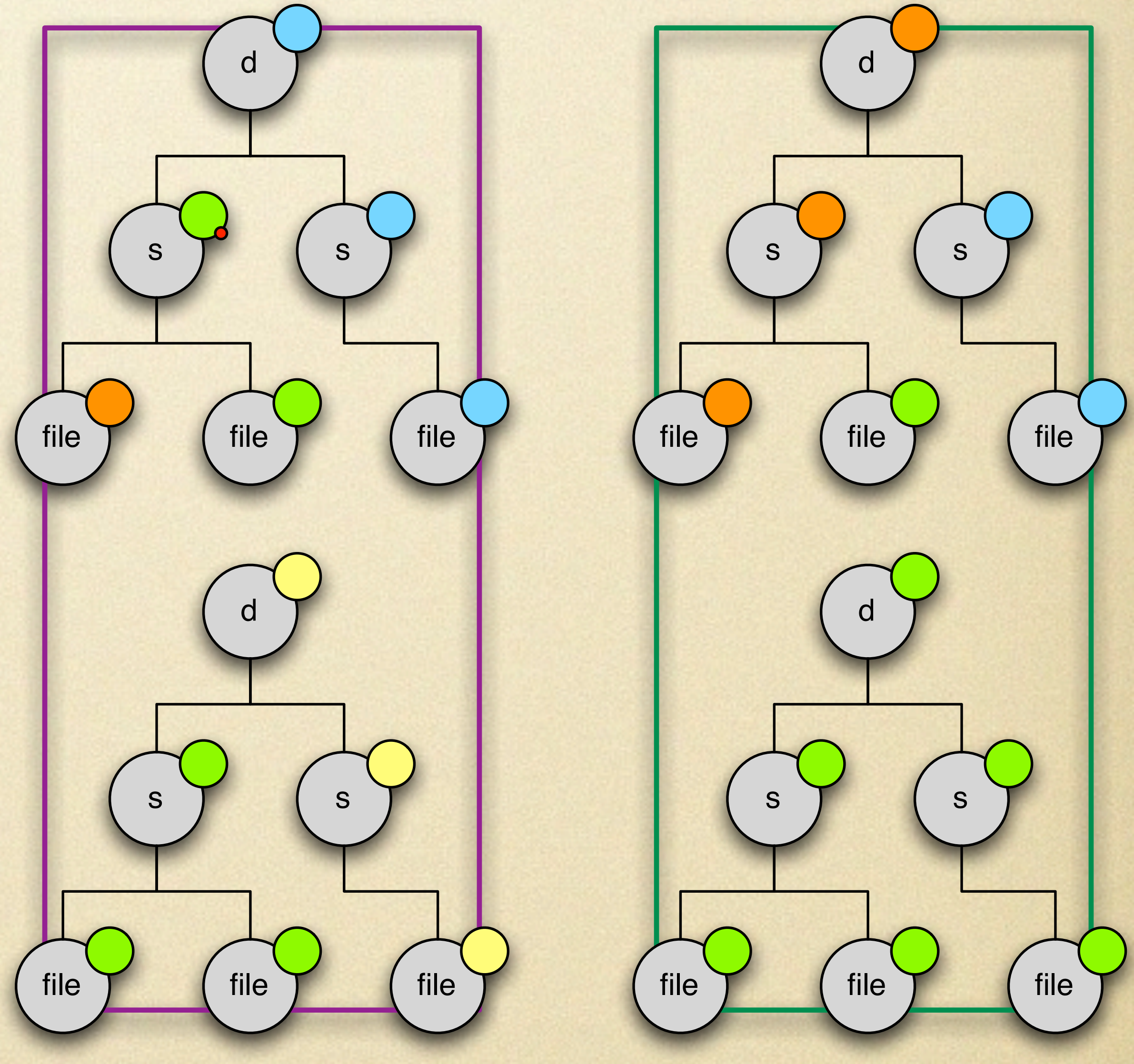
multiVAC element tree

- Hierarchical structure
- Status changes propagate upwards, following priorities
- Easy to track problems
- Can fix element status to avoid propagation

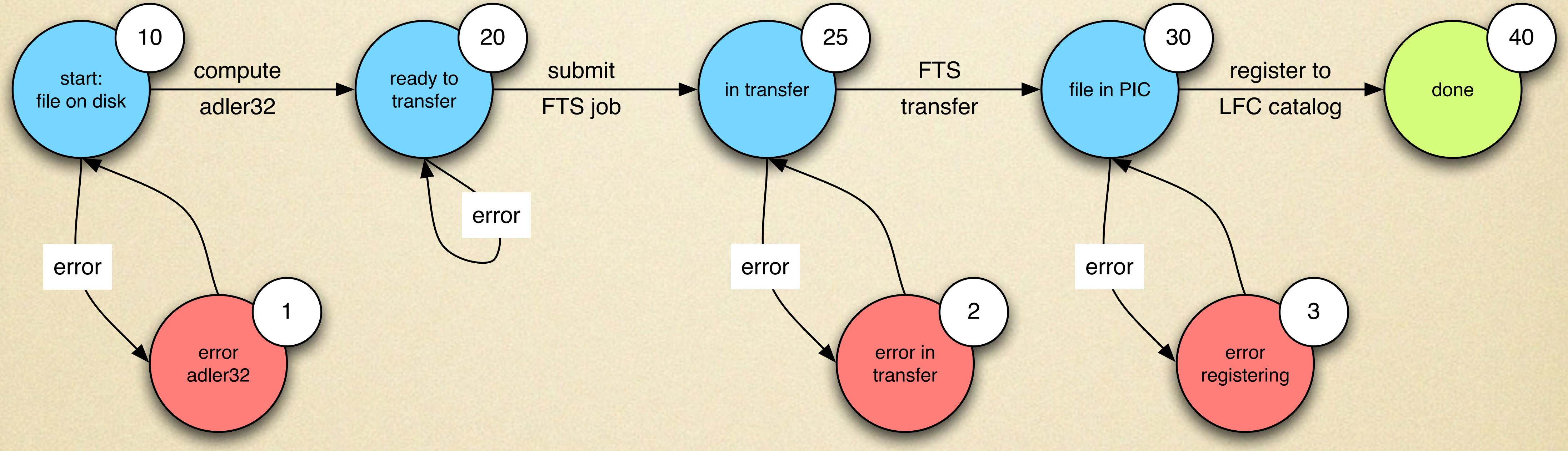


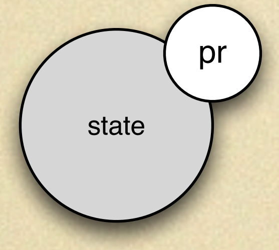
bring it together

- DataTransfer app
- 1 data type = N trees
- agents act on trees looking for specific status, elements, ...
- states defined w.r.t. application workflow

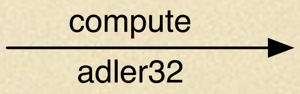


workflow





State with its assigned priority



Process (agent)

DataTransfer status

- Currently testing, in production very soon
- Missing:
  - interaction with OSA: this week
  - automation: next week
  - optimization: always!
- Test results show good performance: OK!

outreach

- Already some projects showed interest in multiVAC and the DataTransfer app in particular
- It may be also interesting for you!

Conclusions

- The DataTransfer application has been developed to deal with the data transfers of a multi-TB/year experiment
- It is based on multiVAC, a PIC development which can be the base of many applications
- It uses gLite FTS as the file transfer method
- Interesting for projects with similar needs