

# ***Distributed HTCondor at GRIF***

G.Philippon   A.Sartirana



# ***Plan of the Talk.***

- **HTCondor at GRIF:** motivations, timeline and feedback;
- quick sketch of **how CREAM + HTCondor works** at GRIF\_LLRL and GRIF\_LLRL
  - ❖ needed for better understanding how the distributed setup works;
- **why a distributed HTCondor** pool?
- few **technical details**;
- tests, **current status**, roadmap to production.



# *Introducing GRIF.*

- **Six labs in the Paris region** (5 with Grid resources) grouped into **one federated Grid T2**;
- ~18k computing slots and ~8PB of storage;
- 10/20Gbps NW backbone (upgrade to 100Gb/s);
- ~20VOs supported;
- 5.5 FTEs.



<https://grif.fr>



# ***HTCondor Motivations.***

- **Maui** no longer maintained/developed
  - ❖ potential **security issues**;
  - ❖ potential **scaling issues** as sites grow bigger;
- **multicore jobs support** required by LHC Vos
  - ❖ not straightforward to implement it in torque/maui;
- no hierarchical fairshare;
- a **general tendency** at many grid sites
  - ❖ very **positive feedback**.

**Q4 2014  
statements**



# Timeline & Status.



{ HTCondor + ARC-CE (Puppet setup);  
new cluster/endpoint (node16);  
**prod Q4 2014**. Currently ~60% of resources;



{ HTCondor + CREAM (Quattor setup);  
test 10/2014, **prod 4/2015 (Big Bang)**;  
also used for local HTC batch cluster;



{ **same setup as LLR**;  
new cluster/endpoint;  
**prod 7/2015**. Migration now **completed**;



{ currently **testing** (quattor setup);  
including **mpi** cluster at IPNO.





# ***Feedback on HTCondor.***

- **Easy** to put in place
  - ❖ for std stuff, follow the doc and it works;
- **documentation** can be difficult
  - ❖ **monumental**: actually a ➤ but easily lost in it;
- very **stable** and very **reactive support**...
  - ❖ nearly no issues seen so far (mostly on CE);
  - ❖ doubts/issues quickly addressed and solved;
- **...but few longstanding issues are there**
  - ❖ cgroup pbm at IRFU (no hints from support);
  - ❖ cases of jobs mysteriously not running



# ***Feedback on HTCondor.***

- ➕ very (very, very, ...) **flexible and powerful**;
- 🟡 quite **different logic** (no queues, classAds,...)
  - ❖ may take a while to get used to it;
- ➖ **upgrades** require nodes draining and often come with **non transparent changes**
  - ❖ e.g.: from 8.4 to 8.6: condor\_q out changed, x509\* classAds no defined at submission, shared\_port service on by default.

There are few ➖ and 🟡 but the  
**overall feedback is very positive!**



# Timeline & Status.



{ HTCondor + ARC-CE (Puppet setup);  
new cluster/endpoint (node16);  
prod Q4 2014. Currently ~60% of resources;



{ HTCondor + CREAM (Quattor setup);  
test 10/2014, prod 4/2015 (Big Bang);  
also used for local HTC



{ same setup as LLR;  
new cluster/endpoint;  
prod 7/2015. Migration now completed;

Focus on  
this

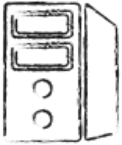


{ currently testing (quattor setup);  
including mpi cluster at IPNO.

➤ Just a sketch of **how CREAM+HTC works** at GRIF.

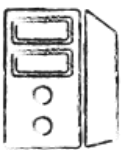
❖ needed to have an idea how the distr. pool works;

**CE**



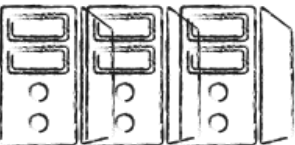
- **cream**
- **condor sched** (submission point + jobs tracking)
- accounting

**CM**



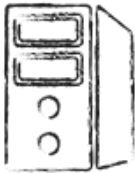
- **condor negotiator** (scheduling, FS implementation)
- resource BDII. To be moved to CE (see later)
- condor defrag (preempting for MC scheduling)

**WNs**



- execution **nodes**

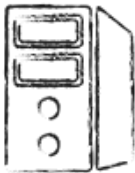
CE



**Blah custom** script `condor_local_submit_attributes.sh` used to **define suitable jobs attributes** (classAds)

VO, DN, FQAN, CREAM QUEUE, UNIX USER

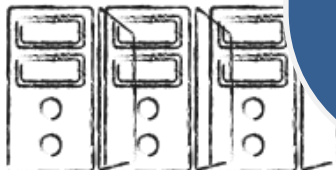
CM



CMS TFC-like  
configurable  
stack of regexp



WNs

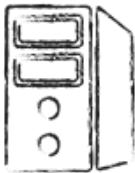


PolicyGroup, CreamQueue, VOName, ProxyDN, Fqan  
accounting\_group/user, WNTag

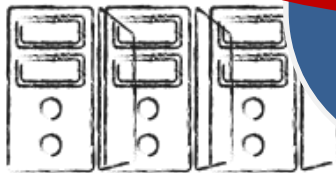
CE



CM



WNs

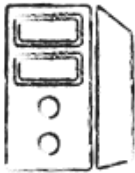


Used with **SYSTEM\_PERIODIC\_REMOVE** to implement, for example, WCT limits .

Used with **SUBMIT\_REQUIREMENTS** to enforce Cream Queues, draining, implement queues policies (e.g.: "MC queues accept only MC jobs")

**PolicyGroup, CreamQueue, VOName, ProxyDN, Fqan**  
accounting\_group/user, WNTag

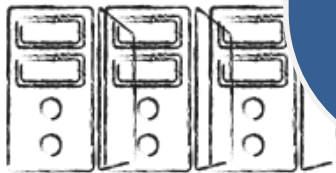
CE



CM



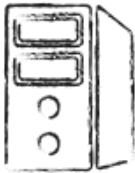
WNs



Used by the **Central Manager** to implement **Hierarchic Fair Share** quotas.

PolicyGroup, CreamQueue, VOName, ProxyDN, Fqan  
**accounting\_group/user**, VNTag

CE



CM



WNs



Used in the **START** expression (at node level) to determine on which nodes a job can run.

```
((WNTag == "ALL") || WNTag == "bigmem") || false)
```

PolicyGroup, CreamQueue, VOName, ProxyDN, Fqan  
accounting\_group/user, **WNTag**



# ***Distributed HTC @ GRIF.***

➤ Historically: **each** GRIF **subsite** has **its own** Grid **cluster**

❖ doing otherwise **with torque/maui** was **not technically straightforward...**

❑ shared homes, non std ports, HN needs WNs list;

❖ ...would have led to a **SPOF...**

❑ no HA setup (at least for CM);

❖ ...and potential **scaling issues**

❑ 2 subsites == ~7k slots;



# ***Distributed HTC @ GRIF.***

➤ This is **not optimal** for various reasons

❖ **inner site complexity exposed** to the VOs

- ❑ VOs have to cope with the splitting of GRIF pledges into sub-resources...
- ❑ ...and balance the usage of sub-sites clusters with different size and FS policies;

❖ **non optimal resources usage**

- ❑ N “small” separated clusters are less effectively used than one big resource;

❖ **no real HA**, just mitigation of CE/CM failures impact

- ❑ one CE/CM down brings down a only one GRIF sub-site (still, the resources of the sub-site are down...).

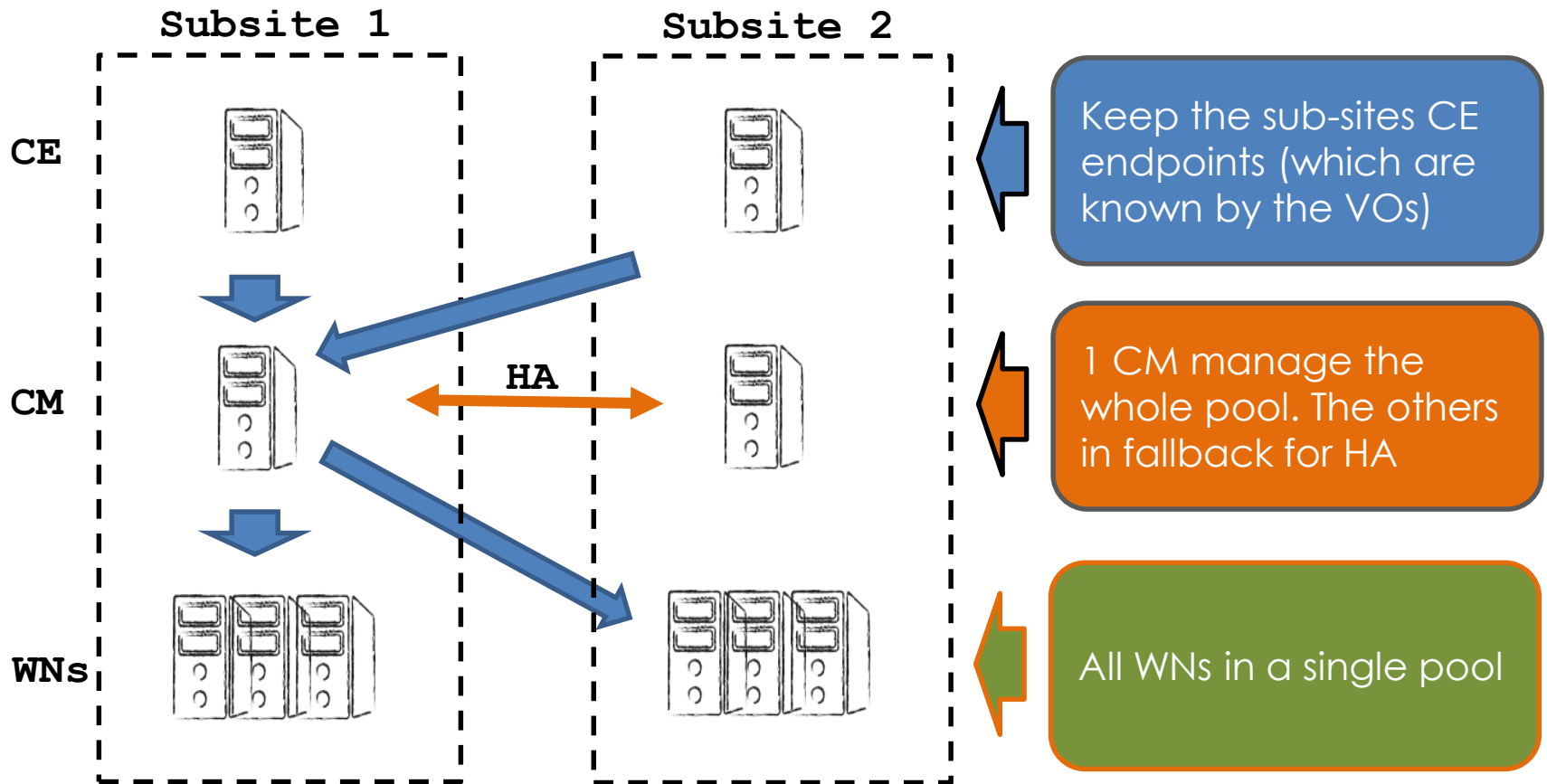


# ***Distributed HTC @ GRIF.***

- now **with HTCondor** we have the possibility to “merge” (some of the) **resources in one pool**
  - ❖ compliant with a **distributed setup**
    - ❑ flexible management of WAN ports;
    - ❑ no need for shared homes;
    - ❑ CM does not need to have the list of WN's;
  - ❖ **HA setup available**
    - ❑ one active Negotiator + N backups;
  - ❖ **no scaling issues**
    - ❑ LHC VO's pilot pools manage  $O(100k)$  jobs in a distributed env.

# ***Distributed HTC @ GRIF.***

➤ What distributed pool looks like...





# ***What it takes...***

- ...to make this work? Not much indeed
  - ❖ the time to look at it...
    - ❑ low prio: 1y WCT ~ 2 weeks CPUT;
  - ❖ ...and some **easy technical setup**;
- take care of cross-firewalls **WAN interaction**
  - ❖ “**shared port**” condor service
    - ❑ pipes all the inter-services communications on one port (default collector port 9618);
    - ❑ good to have it (to avoid ports explosion) even in non distr. setup. **Default since 8.6.**
  - ❖ few other **CREAM ports** (job notification: 9091);



# ***What it takes...***

## ➤ **high availability** setup

### ❖ HAD and REPLICATION services

- ❑ works just fine with the documented config (well... after some iteration with devs);
- ❑ takes care of HA between primary and the secondary negotiator(s);

## ➤ a suitable and hierarchic **Quattor config**

### ❖ each subsite configure its own part: CE and WNs

- ❑ condor makes this easy (e.g. no central WN list);
- ❑ take care of specific stuff: e.g. supported vos;

### ❖ centrally configure the Negotiator and ensure the consistency of the whole.



# ***Devil is in the details.***

## ➤ **Accounting?**

- ❖ **current setup** (each CE accounts its jobs) **ok**

  - ❑ not "subsite contribution". Only CE "popularity";

## ➤ **BDII publication?** Currently on CM

- ❖ glue2 shares depend on queues and VOs

  - ❑ now purely in the CE (and subsite) scope;

- ❖ so publication **moved to CE**

  - ❑ BTW it makes sense removing grid stuff from CM;

- ❖ **redefine glue2 shares ids**

  - ❑ **was:** GLUE2ShareID=<QUEUE>\_<VO>\_<CM>\_ComputingElement

  - ❑ **now:** GLUE2ShareID=<CE>\_<QUEUE>\_<VO>\_<CM>\_ComputingElement



# *Devil is in the details.*

## ➤ nodes/jobs matching

- ❖ varies from subsite to subsite
  - ❑ different (and local) VO supported;
  - ❑ local sw areas;
  - ❑ specific HW (RAM, disk, ...);
  - ❑ local downtimes;
- ❖ technically not pbm (condor kung-fu is strong)
  - ❑ WNTags are there for this;
- ❖ but logistic is complex. **Step-by-step procedure**
  - ❑ **step 0: subsites** logically **separated** (jobs of one CE go in the same subsite's nodes);
  - ❑ and **allow cross-subsite** submission **VO-per-VO** when sure it is ok.



# ***First tests.***

## ➤ Multisite **testbed**

- ❖ **primary** negotiator at **GRIF\_LPNHE**;
- ❖ **secondary** negotiator + **CREAM-CE** at **GRIF\_LL**R;
- ❖ **CREAM-CE** at **GRIF\_LPNHE**;
- ❖ 4 **WNs** at **GRIF\_LL**R;
- ❖ 1 **WN** at **GRIF\_LAL**;

## ➤ **functional tests**

- ❖ job submission, scheduling, execution;
- ❖ WNTags matching;
- ❖ HA switching between primary and secondary;

## ➤ **completed beginning 2017.** All ok.



# ***Running in preprod.***

- Decided to proceed with **"merging"** **GRIF\_LL**R and **GRIF\_L**AL pools
  - ❖ natural choice seen the current status: both sub-sites use HTC in prod with the same conf;
- **"merging"** **test done** using preprod instances
  - ❖ tested that we can **merge pools with jobs running on it**;
  - ❖ **done on ~20/03** and all went fine (that is jobs and services survived);
  - ❖ **running functional SAM tests** since then.



# ***To production.***

- Things have been **running fine in preprod** for >1 month now
  - ❖ used preprod instance to **check/fix last things** about publication/accounting;
- we are now **ready to** do the same with the **production** cluster
  - ❖ **planned** for next Tuesday (9/5);
  - ❖ **first** we will keep the **clusters logically separated...**
  - ❖ ...then we will **allow cross-submission** for some Vos **starting with ops and CMS.**



# *Summing Up.*

➤ **GRIF** running **condor in prod** since **~2 years**

- ❖ ARC-CE/puppet at **GRIF\_RFU**;
- ❖ CREAM-CE/quattor at **GRIF\_LL**R and **GRIF\_LAL**;
- ❖ very **positive feedback**: stable, flexible and powerful.

➤ now possible to aggregate subsites in **distributed HTCondor pools**

- ❖ general **setup quite easy**. Few tricky points (mostly logistics);
- ❖ made functionality tests, **currently running in preprod**;
- ❖ **planned** (for next week) the step **to prod**.



# ***Backup Slides***



# *Quattor config.*

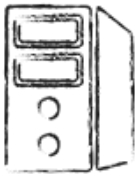
## ➤ Easily configurable via quattor

```
variable CONDOR_CONFIG = {
  SELF['pwd_hash']=...
  SELF['allow'] = '*.in2p3.fr';
  SELF['groups'] = dict( 'group_cms', nlist('quota', '1.0') );
  SELF['params']['MAXWALLTIME']['vo_grif_fr.gridq']=60;
  SELF['group_defaults']['autoregroup'] = true;
  SELF['multicore'] = true;
  SELF['submit_rules'] = dict(
    #SUBMIT_REQUIREMENTS: "name",dict("rule",<RULE>,"reason",<TEXT>)
  );
  SELF['tags_regexp'] = list(<RULES FOR Wntags>);
  SELF['gc_rules']= list(<PERIODIC REMOVES RULES>);
  SELF;
};

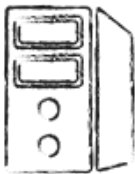
variable WN_ATTRS = dict(
  "DEFAULT", dict("state", "free"),
  "polgrid121.in2p3.fr", dict("tags",list("bigmem")),
  ...
);
```

# *How it works.*

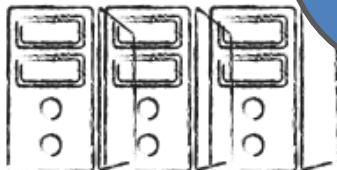
CE



CM



WNs



**Blah custom** script `condor_local_submit_attributes.sh` used to **define suitable jobs attributes** (classAds)

`<unix account>`



`accounting_group_user`

`llcream/cream-condor-<queue>`



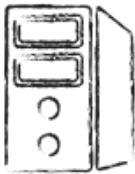
`CreamQueue`

`VO,FQAN,DN`  `MyVOName/ProxySubject/FQAN [*]`

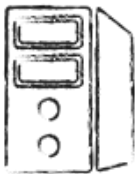
(see next slides to see what these are for)

[\*] there are standard classAds for these. But, since 8.6, they are not defined at submission time.

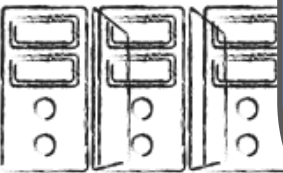
CE



CM



WNs

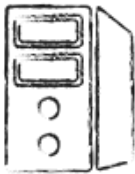


At submission time, the condor\_schedd evaluates the **SUBMIT\_REQUIREMENTS**. Boolean expressions, **if one is false** the submission is **rejected** with a message. E.g.

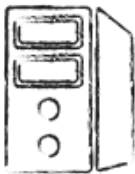
rule	message if false
false	➡ 'This CE is currently draining.'
(CreamQueue == "default")    (CreamQueue == "multicore")	➡ 'This queue does not exists.'
(RequestCpus == 1)    (CreamQueue == "multicore")	➡ 'Multicore jobs should be sent to multicore queue.'
MyVOName != "cms"	➡ 'The CMS queue is draining.'

# How it works.

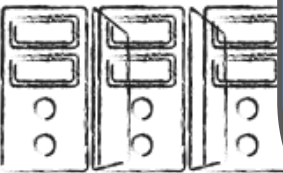
CE



CM



WNs

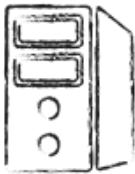


At submission time, the condor\_schedd evaluates the **SUBMIT\_REQUIREMENTS**. Boolean expressions, **if one is false** the submission is **rejected** with a message. E.g.

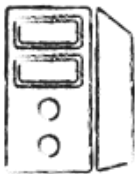
rule	message if false
false	➡ 'This CE is currently draining.'
(CreamQueue == "default")    (CreamQueue == "multicore")	➡ 'This queue does not exists.'
(RequestCpus == 1)    (CreamQueue == "multicore")	➡ 'Multicore jobs should be sent to multicore queue.'
MyVOName != "cms"	➡ 'The CMS queue is draining.'

# How it works.

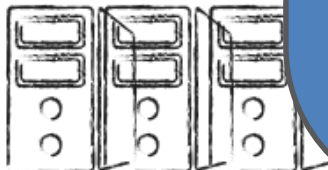
CE



CM



WNs



Once a job is submitted **SYSTEM\_PERIODIC\_REMOVE** is periodically evaluated. **If true it removes the job.** E.g.

**GC-ing held jobs**

**WCT limit on running jobs**

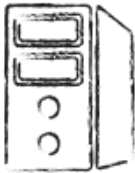
```
(JobStatus == 5 && time() - EnteredCurrentStatus > 3600*48) | |
((JobStatus == 2) && (($ (MAXWALLTIME) > 0) && ((time() -
EnteredCurrentStatus) > (60*$ (MAXWALLTIME)))))
```

**We select the limits** as MAXWALLTIME using the PolicyGroup classAd. e.g.

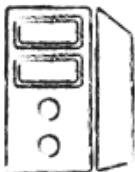
```
MAXWALLTIME = IfThenElse( PolicyGroup ==
"vo_grif_fr.gridq", 60, 4320)
```

# *How it works.*

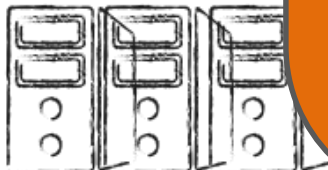
**CE**



**CM**



**WNs**

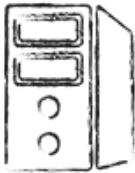


The **Negotiator** implements the **matchmaking** respecting the accounting groups quotas.  
They have **hierarchic quota implementation**

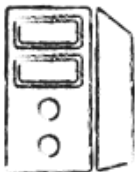
```
GROUP_QUOTA_DYNAMIC_group_mygroup = 1.0
GROUP_QUOTA_DYNAMIC_group_mygroup.sub1 = 0.1
GROUP_QUOTA_DYNAMIC_group_mygroup.sub2 = 0.2
GROUP_QUOTA_DYNAMIC_group_mygroup.sub3 = 0.1
GROUP_QUOTA_DYNAMIC_group_mygroup = 2.0
...
```

# How it works.

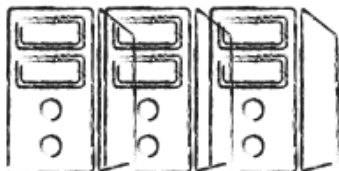
CE



CM



WNs



At **node level** a **START** boolean classAd can be defined to decide whether a **node can run a job**. For example we have it composed of few terms

START\_OFFLINE



false if the node is offline

START\_DRAIN



( x509UserProxyVOName == "ops" )  
if the node is draining

START\_CUSTOM



whatever we want

**START\_TAG**



((WNTag == "ALL") ||  
(WNTag == "bigmem") ||  
false)