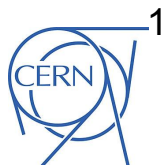




# Exploiting CRIC to Streamline the Configuration Management of GlideinWMS Factories for CMS Support

J. Andreeva<sup>1</sup>, D. Box<sup>2</sup>, J. Dost<sup>3</sup>, A. Di Girolamo<sup>1</sup>, S. Haleem<sup>4</sup>, E. Kizinevič<sup>5</sup>, K. Majewski<sup>2</sup>,  
J. Letts<sup>3</sup>, L. Lobato Pardavila<sup>2</sup>, B. Moreira Coimbra<sup>2</sup>, A. Pérez-Calero Yzquierdo<sup>6,7</sup>,  
M. Mambelli<sup>2</sup>, M. Mascheroni<sup>3</sup>, P. Paparrigopoulos<sup>1</sup>, M. Zvada<sup>8</sup>,  
and on behalf of the CMS Experiment at CERN



1



2



3



4



5



6

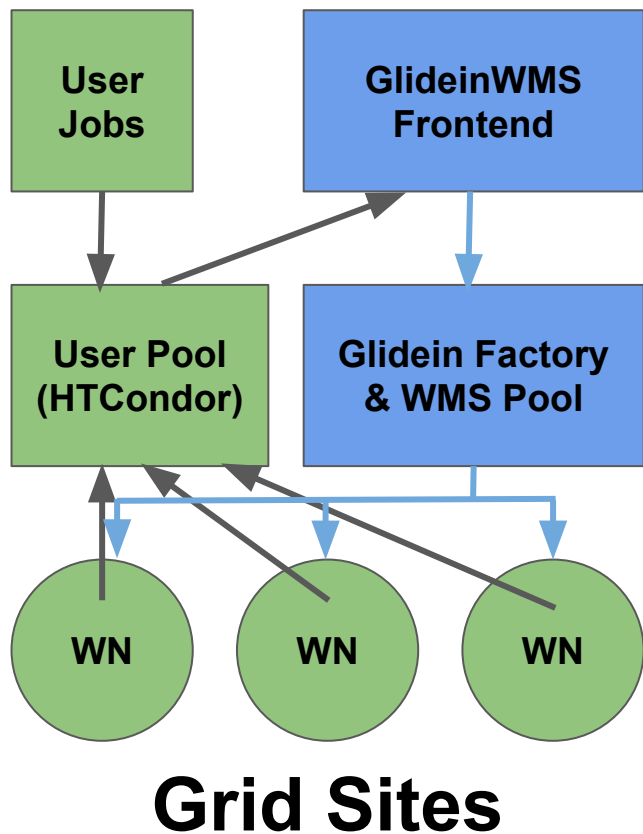


7



8

# The GlideinWMS pool



- GlideinWMS (GWMS) is the workload management system used by CMS to **access Grid resources**
- GWMS **Frontend** looks at the global pool, and **decides pressure** for each Factory entry (~site)
- GWMS **Factory submits pilot jobs** to entries through condor-G
- An “entry” is a set of configuration parameters (for a **CE/cluster**) that defines a submission point in the Factory, i.e. hostname, CE type (HTCondor, CREAM, ARC, e.g.), number of CPUs, Memory, etc.
- A “Compute Element” (CE) is a Grid portal to a computing cluster.

# Factory operations

- A dedicated Operations Team maintains the Factories:
  - The GlideinWMS Factories obtain leases on Grid resources when there is demand for them.
- Ensure the system is optimally delivering useful resources to our users when these are requested:
  - Minimize waste in the process of ensuring the above
- Configurations of all the entries maintained in a set of [xml files](#):
  - Configuration files have been populated over several years with compute elements. Requests through ticketing system.
- Factories might be **shared** with other experiments
  - The maintained factories have ~250 CMS entries out of ~400 in total.
  - CMS served by multiple redundant factories. High availability!

# Example Entry Configuration

```
<entry name="CMSHTPC_T2_IT_Bari_recas_ce04" auth_method="grid_proxy" comment="Added entry 2018-10-11 --Edita" enabled="True"
gatekeeper="ce-04.recas.ba.infn.it:8443/cream-condor-mcoore" gridtype="cream" rsl="WholeNodes = False; HostNumber = 1; CPUNumber = 8" trust_domain="grid" verbosity="std" work_dir=".>
  <config>
    <max_jobs>
      <default_per_frontend glideins="100" held="20" idle="20"/>
      <per_entry glideins="100" held="20" idle="20"/>
      <per_frontends>
      </per_frontends>
    </max_jobs>
    <release max_per_cycle="20" sleep="0.2"/>
    <remove max_per_cycle="5" sleep="0.2"/>
    <restrictions require_glidein_glexec_use="False" require_voms_proxy="False"/>
    <submit cluster_size="10" max_per_cycle="10" sleep="2" slots_layout="fixed">
    </submit>
  </config>
  <allow_frontends>
  </allow_frontends>
  <attrs>
    <attr name="GLEXEC_BIN" const="True" glidein_publish="False" job_publish="False" parameter="True" publish="True" type="string" value="NONE"/>
    <attr name="GLIDEIN_CMSSite" const="True" glidein_publish="True" job_publish="True" parameter="True" publish="True" type="string" value="T2_IT_Bari"/>
    <attr name="GLIDEIN_CPUS" const="True" glidein_publish="False" job_publish="True" parameter="True" publish="True" type="string" value="8"/>
    <attr name="GLIDEIN_Country" const="True" glidein_publish="True" job_publish="True" parameter="True" publish="True" type="string" value="IT"/>
    <attr name="GLIDEIN_MaxMemMbs" const="True" glidein_publish="False" job_publish="True" parameter="True" publish="True" type="int" value="20240"/>
    <attr name="GLIDEIN_Max_Walltime" const="True" glidein_publish="False" job_publish="False" parameter="True" publish="True" type="int" value="171000"/>
    <attr name="GLIDEIN_REQUIRED_OS" const="True" glidein_publish="True" job_publish="False" parameter="True" publish="True" type="string" value="any"/>
    <attr name="GLIDEIN_ResourceName" const="True" glidein_publish="True" job_publish="True" parameter="True" publish="True" type="string" value="INFN-BARI"/>
    <attr name="GLIDEIN_Retire_Time" const="True" glidein_publish="False" job_publish="False" parameter="True" publish="True" type="int" value="108000"/>
    <attr name="GLIDEIN_SEs" const="True" glidein_publish="True" job_publish="True" parameter="True" publish="True" type="string" value="storm-se-01.ba.infn.it"/>
    <attr name="GLIDEIN_Site" const="True" glidein_publish="True" job_publish="True" parameter="True" publish="True" type="string" value="Bari"/>
    <attr name="GLIDEIN_Supported_VO" const="True" glidein_publish="False" job_publish="False" parameter="True" publish="True" type="string" value="CMS"/>
  </attrs>
</entry>
```

# Entry Header

```
<entry name="CMSHTPC_T2_IT_Bari_recas_ce04"  
auth_method="grid_proxy" comment="Added entry  
2018-10-11 --Edita" enabled="True"  
gatekeeper="ce-04.recas.ba.infn.it:8443/cream  
-condor-mcore" gridtype="cream"  
rsl="WholeNodes = False; HostNumber = 1;  
CPUNumber = 8" trust_domain="grid"  
verbosity="std" work_dir=".">
```

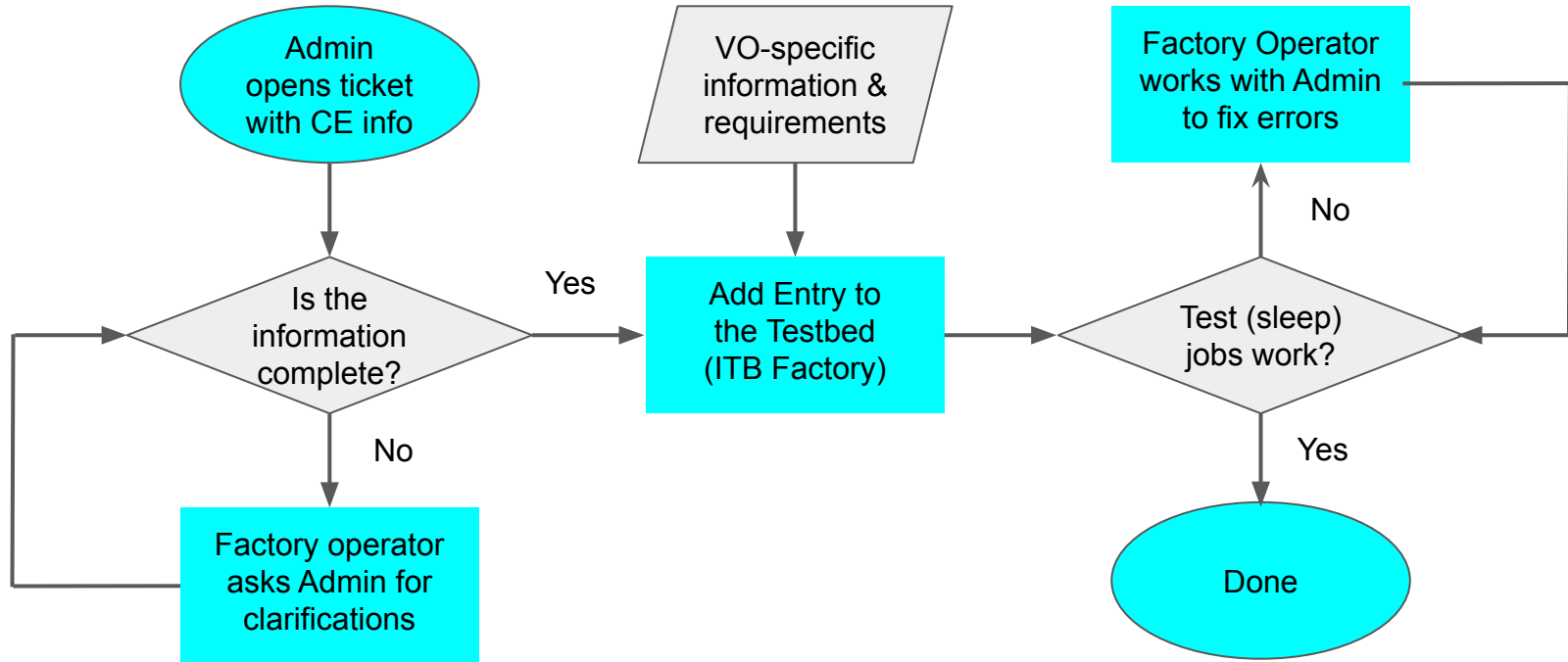
# Configurations ...

```
<config>
  <max_jobs>
    <default_per_frontend glideins="100" held="20" idle="20"/>
    <per_entry glideins="100" held="20" idle="20"/>
    <per_frontends>
    </per_frontends>
  </max_jobs>
  <release max_per_cycle="20" sleep="0.2"/>
  <remove max_per_cycle="5" sleep="0.2"/>
  <restrictions require_glidein_glexec_use="False" require_vo
  <submit cluster_size="10" max_per_cycle="10" sleep="2" slot
  </submit>
</config>
```

# Arritributes, Attributes, Attributes ...

```
<attrs>
  <attr name="GLExec_BIN" const="True" glidein_publish="False"
  <attr name="GLIDEIN_CMSSite" const="True" glidein_publish="True"
  <attr name="GLIDEIN_CPUS" const="True" glidein_publish="False"
  <attr name="GLIDEIN_Country" const="True" glidein_publish="True"
  <attr name="GLIDEIN_MaxMemMBs" const="True" glidein_publish="True"
  <attr name="GLIDEIN_Max_Walltime" const="True" glidein_publish="True"
  <attr name="GLIDEIN_REQUIRED_OS" const="True" glidein_publish="True"
  <attr name="GLIDEIN_ResourceName" const="True" glidein_publish="True"
  <attr name="GLIDEIN_Retire_Time" const="True" glidein_publish="True"
  <attr name="GLIDEIN_SEs" const="True" glidein_publish="True"
  <attr name="GLIDEIN_Site" const="True" glidein_publish="True"
  <attr name="GLIDEIN_Supported_VOs" const="True" glidein_publish="True"
</attrs>
```

# Operations: Adding an Entry



- Everything done manually! Process can take a week per entry.
- Same procedures (and time) to make a change! Can we improve?



# Different types of information

1. **Grid-infrastructure** (OSG, EGI) **owned** information systems
  - e.g.: CE hostname, CE type (Arc, HTCondor, etc), queue name, resource name, supported VOs, etc.
  - Can be found in various information systems (e.g. [OSG Topology](#), [EGI gocdb](#))
2. **Site-specific**, but not published
  - e.g: working directory, maximum jobs, operating system, etc.
3. Internal **Factory settings**
  - e.g: Grid submission rates, entry names, etc.
4. **VO pilot configuration**
  - e.g.: CPU/Memory configuration

**Idea: Automate how we get #1 and #2**

# The Computing Resource Information Cache (CRIC)

In 2015 the OSG announced plans to stop using BDII to publish their computing resources. This triggered a review of information systems in WLCG.

CMS began to evaluate adopting CRIC in 2016 as an information service.

It is a community project, based on a refactorization of AGIS, the ATLAS Grid Information System.

Part of a trend in CMS (and HEP at large) to leverage **community software projects**. Other examples in CMS include GlidienWMS, HTCondor, Rucio, MonIT.

# CRIC



Gather and allow access to information about physical and CMS logical computing resources:

- [Core CRIC](#) with info from gocdb/oim, for example (CE's and their queues)
- [CMS CRIC](#) with experiment-specific info on how we use resources
- CMS is using these CE's through the computing units (Sites)
- Each compute unit has multiple compute resources (GlideinWMS entries)

**API's available to retrieve this information.**

Leverage CRIC to automate Factory configuration generation.

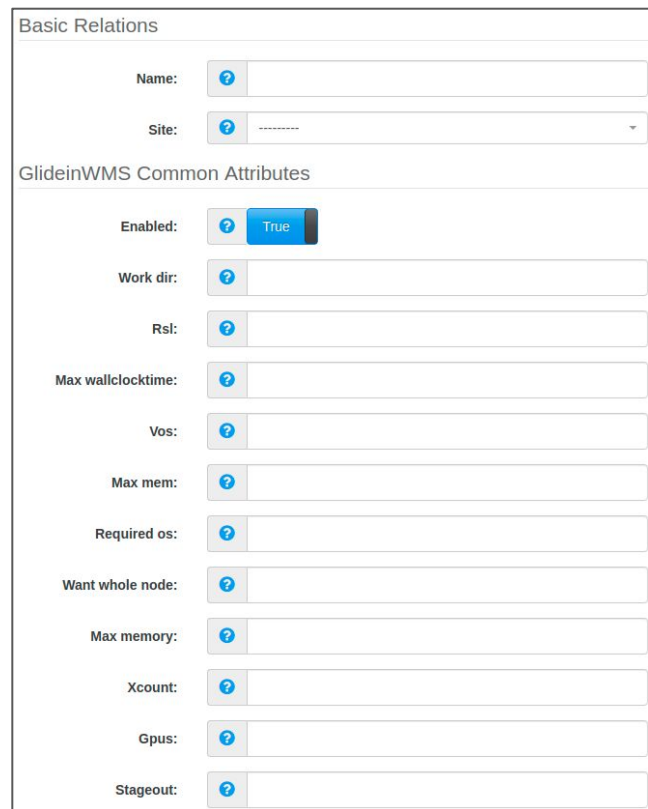
# Generating configs from CRIC



- Since the environment is complex, we want to design a system that gives Factory operations some flexibility to take into account all use cases:
  - Get the relevant info from CRIC,
  - but then generate xml configurations in the Factory itself.
- Gives Factory ops some extra control:
  - In case we need to overwrite some values, or add specific ones
  - Allows use of plug-in modules to take care of non-WLCG use cases
- Developed scripts that saves those info from both CMS and core CRIC:
  - Information saved in a set of yaml files
    - Different files for different types of information (e.g.: Grid-owned info vs. site-specific)
  - Another script merges this information and produces the final xml configuration.

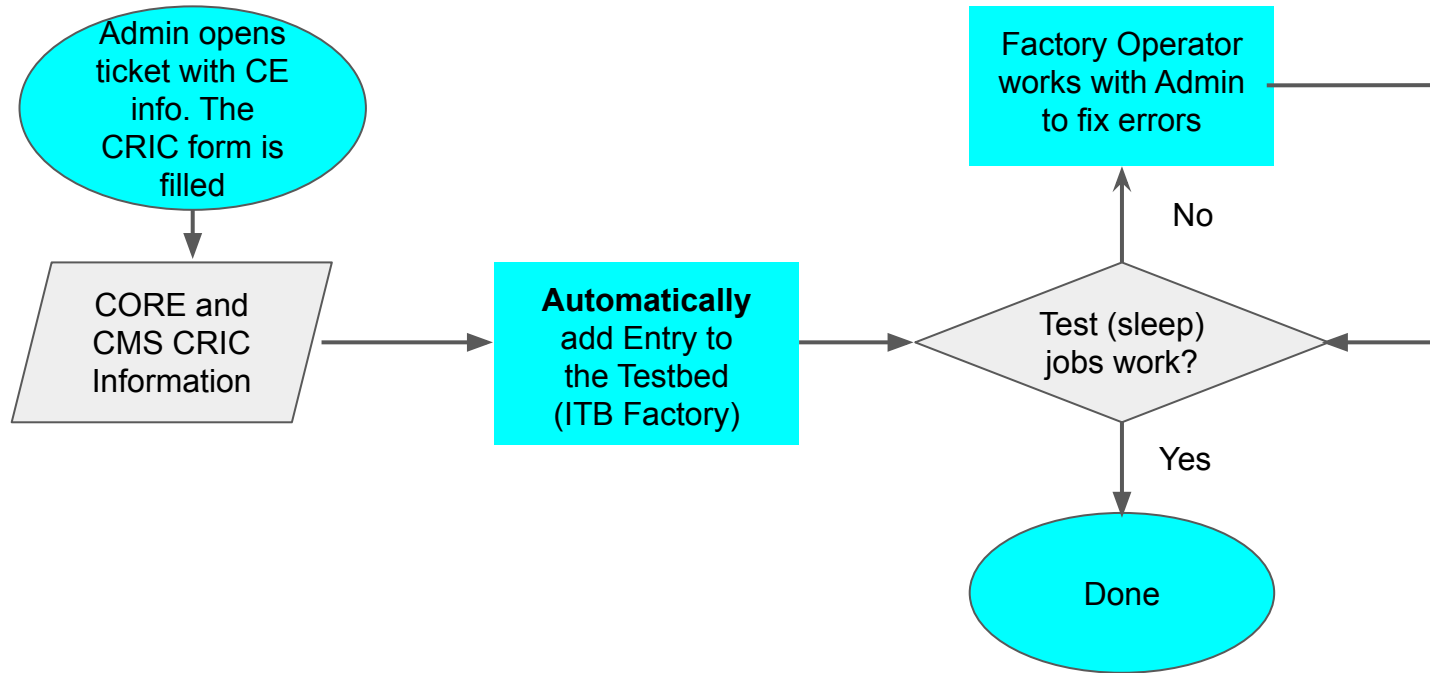
# Advantages of this approach

- Central and public place where all the site configurations are available (through web)
- Streamline Factory operation workflow: less back and forth with sites
- Free up some load on Factory operations
  - Info can be inserted by site admins when not available in information systems
  - Easier to fill in a web form than edit a big XML file
- Starting point for more automation
  - e.g. automatically send test jobs once an entry is created in CRIC

A screenshot of the CRIC web interface showing a configuration form. The form is titled "Basic Relations" and "GlideinWMS Common Attributes". It contains several input fields, each with a question mark icon to its left. The "Enabled" field is a toggle switch set to "True". The other fields are empty text boxes.

Basic Relations	
Name:	<input type="text"/>
Site:	<input type="text" value="-----"/>
GlideinWMS Common Attributes	
Enabled:	<input checked="" type="checkbox"/> True
Work dir:	<input type="text"/>
Rsl:	<input type="text"/>
Max wallclocktime:	<input type="text"/>
Vos:	<input type="text"/>
Max mem:	<input type="text"/>
Required os:	<input type="text"/>
Want whole node:	<input type="text"/>
Max memory:	<input type="text"/>
Xcount:	<input type="text"/>
Gpus:	<input type="text"/>
Stageout:	<input type="text"/>

# Operations: Adding an Entry



- Less back and forth with sites
- Eventually can automate submission of test jobs once entry is in CRIC.

# Current situation

- A form to add an entry is available in CRIC.
  - Many thanks to the CRIC development team who listened to our feedback!
- Prototype has been extensively tested.
- Successfully used it to add 20 entries in production factories:
  - Tested on sites with a complicated topology and many different entries like Nebraska and UCSD.

# Next Step: Addressing more use cases

Extending the CMS-centric solution to a more complex environment

- Factory team serves **many** other **VOs**: ICEDCube, LIGO, nanoHUB, Glow...
  - Different Frontends with different credentials sending pilot requests
- They have **different types of sites**: OSG, WLCG, EGI
- And **different technologies (CE's)**: HTCondorCE, CREAM, NorduGrid, Arc ...
- And all the different **permutations**:
  - e.g. OSG only sites that serves LIGO, OSG/WLCG site serving multiple experiments, etc.
  - For CMS we focus on WLCG sites (EGI, OSG, and others)

Factory operators need a **common interface** to configure entries regardless of who owns the resources. We are in close contact with the CRIC team and will evaluate possible solutions.



# Conclusions

- Factory operations historically done by managing a set of big xml files.
- Worked on a new solution that allows CMS Factory operators to leverage the CRIC interface to insert and manage entries:
  - With all the advantages of a web-based solution, compared to manipulating xml files.
- Solution already validated in a production environment.
- Evaluating how to address a wider variety of use cases in addition to CMS VO and WLCG Sites.

# Abstract

GlideinWMS is a workload management and provisioning system that allows sharing computing resources distributed over independent sites. Based on the requests made by GlideinWMS Frontends, a dynamically sized pool of resources is created by GlideinWMS pilot Factories via pilot job submission to resource sites' computing elements. More than 400 computing elements (CE) are currently serving more than 10 virtual organizations through GlideinWMS, with CMS being the biggest user with 230 CEs. The complex configurations of the parameters defining resource requests, as submitted to those CEs, have been historically managed by manually editing a set of different xml files. New possibilities arise with CMS adopting the Computing Resource Information Catalogue (CRIC), an information system that collects, aggregates, stores, and exposes, among other things, computing resource data coming from various data providers. The talk will describe the challenges faced when CMS started to use CRIC to automatically generate the GlideinWMS Factory configurations. The architecture of the prototype, and the ancillary tools developed to ease this transition, will be discussed. Finally, future plans and milestones will be outlined.

# Acknowledgements

This work was partially supported by the U.S. Department of Energy, the National Science Foundation, and by Spain's Ministry of Economy and Competitiveness grant FPA2016-80994.

CMS thanks our partners in the GlideinWMS, HTCondor, and CRIC development teams, the OSG, and our colleagues at CERN, all of whom make the shared computing infrastructure a success.