Table of Contents

Table of Contents

TABLE OF CONTENTS

Discussions about CMS Specific CRIC plugin

This page is intended to summarize minutes and further iterations with CMS representative concerning the CMS plugin for CRIC.

Google doc with minutes of meetings with CMS

DISCLAIMER: The notes I put here are not exhaustive, and are very likely wrong. You are welcome in amending them: if so, please add a small comment on why. **Nota Bene**: I will also link some notes/proceedings concerning the CMS computing model, which help us figuring out the use cases for the CMS CRIC by understanding the interactions between the CMS middleware and the information system. Again, if you have further documentation, please link it here.

Main data structures

Data structure	Core CRIC	Experiment CRIC	Core - Experiment CRIC objects Relationship	To clarify
Site	Name : Site , Attributes : country, coordinates, type , status	Name: Experiment Site (VOSite), Attributes : tier, state,contact(s) exec/admin/tech, pledges (CPU, disk, tape)	Indirect reference via set of services belonging to experiment site and hosted by Site from Core CRIC	-
Federation	Name: Federation, Attributes: Name, Contact email. Sites in Core CRIC reference to the Federation they belong to	Not clear whether Region in the CMS sense is the same as Federation in CORE. Main attributes for Region (or Federation) in the Experiment CRIC are pledges for (CPU, disk and tape).	In case Federation and Region are the same thing Experiment Region should have a reference to the Federation in Core	Whether Federation and Region are the same thing
Region	-	Not clear whether Region in the CMS sense is the same as Federation in CORE. Main attributes for Region (or Federation) in the Experiment CRIC are pledges for (CPU, disk and tape). In case Federation and Region are not the same thing, Experiment Sites in the Experiment CRIC should reference Region they belong to	In case Federation and Region are not the same thing , there is no peer object for Region in CORE CRIC	Whether Federation and Region are the same thing
Service (CE type)	Combination of Service, Resource and Queue. Service has one to many to Resources. Resource has one to many queues	Name : Resource unit , Represents combination of the CE gatekeeper and the queue. Attributes: name, contact(s), location, type, status, capacity, performance, quota, support-level, stage-out(s) for the time being is a string, factory-config, factory-tag subsite, reference to the experiment site. Resource units of the squid type should reference Resource Unit of the compute site to	Resource Unit should reference the queue in core CRIC. Via such references, relationship between experiment site and site in Core CRIC will be resolved	-
Compute Unit	-	Name : Compute Unit. Attributes: name, contact(s) admin/tech, status, stage-out(s) for the time being is a string to overload stage-outs in the Resource unit of the	-	

		compute site, could be resolved through the references to the Resource Unit of the storage type later. Compute Unit combines Resource Units of various types, namely computing type (list of queues), storage type (where input is located), squids. Therefore, Resource Units of different types should be mapped to the Compute Unit they belong to.Sets of squids can be resolved via resource unit of the compute type. At the level of the Compute Unit they are to overload those which are resolved at the level of the resource unit.	

Meeting on February 2nd 2017

Attendance: Julia, Alessandro, Kate, Alexey, Salvatore (WLCG); Stephan, Giuseppe (CMS computing)

The focus has been on the data structures, starting from what already summarized in WLCGConfigurationEvaluation

Glossary

In order to avoid issues with terminology, if a term is already used for core CRIC description, I will describe the CMS specific term with a leading *Logical*, and label it as VO<Term>.

Region

The Region is a CMS concept: it contains a list of sites, and is used to aggregate pledges assigned to sites by CMS.

Name

The name assigned by CMS.

Contact

The representative of the sites in the region for CMS computing planning and operations.

Pledges

The pledges assigned to a region. See below for further discussion.

List of sites

The list of sites belonging to the region. See below for further discussion.

Relationship between WLCG Federation and CMS Region

In WLCG, pledges are assigned per federation, as per MoU. On the other hand, CMS requires each site to be assigned a pledge, which is extracted from the one assigned to the federation the site belongs to. The site pledges are then summed up in the region.

In the discussion, it was not clear if there is a direct relationship between Federation and Region, i.e. if a region is made up of sites belonging to the same federation, or if it spans between sites in different federations.

```
Example: Consider:
Federation1 : [Site1, Site2, Site3]; Federation2 : [Site 4, Site5]
Possibilities:
a. a region is made up of sites belonging to the same federation:
Region1: [Site1, Site2]; Region2: [Site3]; Region3: [Site4, Site5] ==> Federation is made up of R
b. a region is made up of sites belonging to different federations:
Region1: [Site1, Site2]; Region2: [Site3, Site4]; Region3: [Site5]
```

ACTION: Clarifying the relationship between Federation, Region, and Site.

QUESTION: A federation is made up of *physical* sites, while a CMS *logical* site can *either* group more physical sites, *or* use only a part of the capacity offered by a physical site. How are pledges related? Or does a CMS region group physical sites?

CMS has the responsibility to fill site pledges: an API/UI is therefore needed.

QUESTION: (related to the previous) does CMS need a validation of the pledges constrained by the value assigned to the federation?

Logical Site VOSite

A logical site groups service units, thus offering a hook for resources to be exploited by jobs.

A VOSite is not directly related to a Site: they are referenced through the implementation of services. Site hosts real physical services. VOSite uses a set of those, grouped in service (compute, storage) units.

Name VOName

A uniquely defined string identifying the entity according to the CMS convention: e.g. T0_CH_CERN, T1_IT_CNAF.

CMS Contacts

Three actors are needed, according to the CMS operation policies:

- Administrator: it is in charge of the logical site definition
- Executive contact
- Technical contact

They extend the contact and security responsible for physical sites as in GocDB /OIM.

Location

The coordinates of the logical site. They should be derived from the physical site, but due to the fact that a logical site can aggregate resources from different locations, there should be a possibility to set it from a list of possible entries.

QUESTION: can it be removed from the VOSite? Is it sufficient a reference to the region?

Pledges

See the discussion above.

List of compute units

Each logical site contains at least one compute unit.

List of storage units

Each logical site contains at least one storage unit, a storage unit can be associated to one and only one logical site.

Compute Unit

The compute unit groups a set of compute elements for administrative and organizational purposes. As a reminder, this concept is partially implemented in the CMS computing model.

One of the CMS requirements for the evolution of the IS states that "it should allow to discover all information needed to configure a site entry in the glideinWMS pilot factory". Hence, these values are to be implemented in the CMS specific CRIC plugin.

QUESTION: is this now CU-specific? Any example of this configuration?

Name

A uniquely defined string identifying the entity.

Logical Site

A unique reference to the VOSite the CU is registered to: a compute unit can be associated to one and only one logical site.

Contacts

Two actors are needed, according to the CMS operation policies:

- Administrator: it is in charge of the CU definition, and can be inherited from the VOSite admin.
- Technical contact

Status

Provides information about the status of the compute unit, e.g. development, testing, production, disabled...

List of compute elements

Each compute unit contains at least one compute element. For complex site architecture, the physical CE in the CU could sit at different physical sites.

List of Frontier Squids

Each compute unit contains at least one Frontier Squid cache for accessing conditions. Currently, the squid access is driven by the site-local-config.xml file sitting in the pre-defined path in the site.

This attribute is in the process of being assigned to Compute Units.

List of Storage Units

Each CU points to a list of Storage Units: these are the SU for which there is a direct (administrative/network) link to the CU.

In most cases, the list will contain only one element, but this cannot be valid for sites with complex architecture.

EXAMPLE: Suppose we want to launch a SAM test for a SU. The list of CU is queried, and the test job is sent to one of the CE in the CU. This prevents false positives to occur due to bad network connections which let the test fail.

EXAMPLE: This link helps to optimize, or even by-pass, the stage-out phase (see later).

Stage-out

A temporary location where jobs send their output before being moved to the final destination. This can be configured, e.g. in CRAB.

The configuration of the stage-out is currently defined in the site configuration, using the local-stage-out tag:

- command value: the plugin for moving from the stage out to the final destination: it could be a synchronous move from a local disk to a storage element, or an asynchronous move from two Storage elements (**TBC**).
- option value: the CLI settings.
- catalog URL: (*not clear at all, sorry*) this relates to the way the TFC, or more generally PhEDEx, maps logical filenames to physical filenames. **TO BE CLARIFIED**

QUESTION: Can the stage out be considered as a (temporary) SU? Do we need a parameter to identify this attribute?

References

1. Site local config repository and Example for Pisa Tier2 (`T2_IT_Pisa`)

Bootstrap CMS info into CRIC prototype

New link to CMS VOFeed

CMS site to Core site mapping information comes from two main sources: SiteDB and CMS VO Feed. In the table below you can find the mappings from both sources all merged, together with information on their source and the nature of the relation between them (one to many, many to one or one to one). After inspecting the data some issues have been identified (also visible on the table below).

All the information about sites that exists in Glidein factory also exists in sitedb, so it's safe to ignore Glidein's factory information regarding site names.

CMS Site

Usefull links

Procedure to add T2/T3 CMS sites: https://twiki.cern.ch/twiki/bin/view/CMSPublic/AddCmsSite

CMS Sites with issues

Site Name	Problem	Action Taken
SU-OG-CE	It appears as one of the core sites for T3_US_OSG based on vofeed and glidein entries. Doesn t exist in core sources. In sitedb T3_US_OSG is mapped to T3USOSG	We don t add it to the CORE. All the services / glidein entries of this site are mapped to `SU-ITS `.
SU-OG-CE1	It appears as one of the core sites for T3_US_OSG based on vofeed and glidein entries. Doesn t exist in core sources. In sitedb T3_US_OSG is mapped to T3USOSG.	We don t add it to the CORE. All the services / glidein entries of this site are mapped to `SU-ITS `.
T3USOSG	Mapped as the core site of T3_US_OSG in SiteDB. Doesn t exist in core sources.	We don t add it. We map T3_US_OSG to `SU-ITS `
TAMU_BRAZOS_CE	This appears in glidein entries and vofeed but it s not in CORE. Sitedb correctly maps T3_US_TAMU to TAMU_BRAZOS and has not reference to TAMU_BRAZOS_CE	We don t add it. We link it to TAMU_BRAZOS and we hardcode exception data in order to generate the vofeed.
t3usnersc	This appears in SiteDB as the core site for T3_US_NERSC. Doesn t appear in vofeed. There are glidein entries for it but they have `T3_US_NERSC` as the core site. There are 8 sites that resemble this in core (NERSC-Carver, NERSC-Davinci, NERSC-Franklin, NERSC-ITB, NERSC-Jacquard, NERSC-PDSF, NERSC-PDSFSRM, NERSC-VM-VTB0)	We don t add it. Created new core site NERSC. Merged all the 8 sites into this one and skipped importing them. All the possible names cms has given to this site are now mapped to NERSC.
Umd-ce	Appears in vofeed as a core site for T3_US_UMD. In sitedb T3_US_UMD is correctly mapped to the core site `umd-cms`. Also appears to a glidein entry	We don t add it. Instead we map it to umd-cms. Hardcoding data to generate vofeed.

	as the core site.	
ru-Moscow-SINP-LCG2-t3	Is used in SiteDB to map T3_RU_SINP. Not in core. Site not tested in vofeed. Not in glidein entries.	We don t add it. Instead we map it to ru-Moscow-SINP-T3.
VBU_CMS	Is used in sitedb to map T3_IN_VBU. Not in core. Also used in vofeed but not in glidein entries.	We add it to the CORE because there are no sites to match it in GocDB.
Comet, t3ussdsc	In vofeed it maps to T3_US_SDSC. In sitedb same cmssite is mapped to t3ussdsc (no reference to comet). In glidein entries there is one entry for T3_US_SDSC which has `T3_US_SDSC` as the core site.	We add t3ussdsc to core. We map glidein entries and we hardcode data for vofeed.
UVA-HEP	Used in SiteDB to map T3_US_UVA. Also used in vofeed. Not used in glidein entries. UVA-sunfire is the one found in core.	We don t add it. We map it to UVA-sunfire and we hardcode data for vofeed
Volunteer_CMS	Used in sitedb to map T3_CH_Volunteer. Also used in vofeed and glidein entries.	Adding the site to core.
USCMS-FNAL-LPC	Used in sitedb to map T3_US_FNALLPC. Also used in vofeed not used in glidein entries.	We add it to the core.
T3_US_UTENN	Used in sitedb to map T3_US_UTENN. Not in vofeed, not in glidein entries. No relevant site in core.	We add it to the core.
T3_US_JHU	Used in sitedb to map T3_US_JHU. It is in vofeed (although there are no services mapped except one empty srm (there is no hostname)). Not in glidein entries.	We add it to the core.
T3USHEPCLOUD	Used in sitedb to map T3_US_HEPCloud. Used in vofeed. Not in glidein entries	We add it to the core.
T3_US_Kansas	Used in sitedb to map T3_US_Kansas. Not used in glidein entries, not in vofeed.	We add it to the core.
INFN-IT_OPPORTUNISTIC	Used in sitedb to map T3_IT_Opportunistic. Not in vofeed not in glidein entries.	We add it to the core.
Argonne	Used in sitedb to map T3_US_ANL. Not in vofeed. Not in glideins.	We map it to ANLASC site (already in core).
FNAL_HEPCLOUD	Not used in sitedb not in vofeed but there is one glidein entry (disabled) using it as core and cmssite	
T3_CH_CERN_HelixNebula	Used in sitedb to map T3_CH_CERN_HelixNebula. Is in vofeed. Is in glidein entries.	We add it to core. No site in gocdb/oim.
ru-Moscow-SINP-LCG2	Used in sitedb to map T2_RU_SINP. Used in vofeed and in glidein entries.	We add it to core. No site in gocdb/oim.
T3_US_Princeton_ARM	Used in sitedb to map T3_US_Princeton_ARM. Used in vofeed. Only test entries in glideins.	We add it to core.
T3_US_Princeton_ICSE	Used in sitedb to map T3_US_Princeton_ICSE.	We rename Princeton_ICSE_T3_CMS to T3_US_Princeton_ICSE.
Stampede, t3ustacc	Stampede used in vofeed to test services of T3_US_TACC. In sitedb though	Adding t3ustacc in core. Hardcoding data for vofeed.

	T3_US_TACC is linked to t3ustacc. Stampede is also used instead of t3ustacc in one test glidein entry.	
T3_US_BU	Used in sitedb to map T3_US_BU. Not in vofeed. Not in glidein entries	We add it to the CORE.
cern_prod_wigner, cern_prod_ai, cern_prod_hlt, cern_prod__helix_nebula, public_cern_cms		Adding them all to core.
T2-MY-UMSIFIR	Used in sitedb to map T2_MY_SIFIR. Not in core. Not used in vofeed.	We map everything to MY-UM-SIFIR which is in coe
JINR-Tier2	This one is in OIM (Why?? It should only be in GocDB). GOCDB has the `real` site JINR-LCG2 and it provides all the services (including the ones OIM has for JINR-Tier2)	It s no longer imported into CRIC. Excluded it from the OIM Provider.

Storage

Bootstrapping storage-related information from the CMS-specific sources

Finding out structure of the top-level name spaces at the CMS sites

SiteName	SiteDB Title	protocol used to query	S
T0_CH_CERN		gfal-ls srm://	srm-cms.cern.ch

T1_DE_KIT	KIT	lcg-ls srm://	cmssrm-kit.gridka.de
T1_ES_PIC	PIC	srmls srm://	srm.ciemat.es:8443/

		srmls srm://	srm.ciemat.es:8443/s
T1_IT_CNAF			storm-fe-cms.cr.cnaf
T1_FR_CCIN2P3	CC-IN2P3	lcg-ls srm://	ccsrm.in2p3.fr
T1_RU_JINR_MSS	JINR-T1	lcg-ls srm://	srm-cms-mss.jinr-t1.
T1_RU_JINR_Buffer	JINR-T1	lcg-ls srm://	srm-cms.jinr-t1.ru

T1_RU_JINR_Disk	JINR-T1DISK	lcg-ls srm://	srm-cms.jinr-t1.ru
T1_UK_RAL(T1_UK_RAL_Disk)		srmls srm://	srm-cms-disk.gridpp
T1_US_FNAL		xrdfs root://	cmsxrootd-site.fnal.g

T1_US_FNAL_Disk		xrdfs root://	cmsxrootd-site.fnal.g
		xrdfs root://	cmsdcadisk01.fnal.g
T2_AT_Vienna	Hephy-Vienna	lcg-ls srm://	hephyse.oeaw.ac.at
T2_BE_IIHE	IIHE	lcg-ls srm://	maite.iihe.ac.be
T2_BE_UCL		lcg-ls srm://	ingrid-se02.cism.ucl.

T2_BR_SPRACE		srmls srm://	osg-se.sprace.org.br
T2_BR_UERJ		srmls srm://	se.hepg
T2_CH_CSCS_HPC Swiss-National-Supercomputing-Centre	lcg-ls srm://	storage01.lcg.cscs.ch	/pnfs/lcg.cscs.ch/cms

T2_CH_CSCS	CSCS	lcg-ls srm://	storage01.lcg.cscs.ch
T2_CN_Beijing	Beijing	lcg-ls srm://	srm.ihep.ac.cn
T2_DE_DESY	DESY	lcg-ls srm://	dcache-se-cms.desy.d
T2_DE_RWTH	RWTH	lcg-ls srm://	grid-srm.physik.rwth

T2_EE_Estonia		lcg-ls srm://	ganymede.hep.kbfi.e
T2_ES_CIEMAT	CIEMAT	lcg-ls srm://	srm.ciemat.es
		srmls srm://	srmcms.pic.es
T2_ES_IFCA		srmls srm://	srm01.ifca.es

T2_FI_HIP	Helsinki-Institute-of-Physics	lcg-ls srm://	madhatter.csc.fi/pnfs
T2_FR_CCIN2P3		srmls srm://	ccsrmt2.in2p3.fr
T2_FR_GRIF_LLR	GRIF_LLR	lcg-ls srm://	polgrid4.in2p3.fr

T2_FR_IPHC	IPHC	lcg-ls srm://	sbgse1.in2p3.fr
T2_FR_GRIF_IRFU	GRIF_IRFU	lcg-ls srm://	node12.datagrid.cea.
T2_GR_Ioannina	Ioannina	lcg-ls srm://	grid02.physics.uoi.g
T2_IN_TIFR	TIFR	lcg-ls srm://	se01.indiacms.res.in

T2_IT_Bari		srmls srm://	storm-se-01.ba.infn.i
T2_IT_Pisa		srmls srm://	stormfe1.pi.infn.it

T2_IT_Rome	Rome	lcg-ls srm://	cmsrm-se01.roma1.i
T2_IT_Legnaro	Legnaro	lcg-ls srm://	t2-srm-02.lnl.infn.it
T2_IR_IPM	IPM	lcg-ls srm://	se1.particles.ipm.ac.i
T2_HU_Budapest	Hungary	lcg-ls srm://	grid143.kfki.hu

T2_KR_KISTI	KISTI-GSDC	lcg-ls srm://	cms-t2-se01.sdfarm.k
T2_KR_KNU	KNU	lcg-ls srm://	cluster142.knu.ac.kr
T2_MY_SIFIR			
T2_MY_UPM_BIRUNI			
T2_PK_NCP	NCP-LCG2	lcg-ls srm://	pcncp22.ncp.edu.pk
T2_PL_Swierk	lcg-ls srm://		se.cis.gov.pl

T2_PL_Warsaw			
T2_PT_NCG_Lisbon		srmls srm://	srm01.ncg.ingrid.pt
T2_RU_IHEP	IHEP	lcg-ls srm://	dp0015.m45.ihep.su
T2_RU_JINR	JINR	lcg-ls srm://	lcgsedc01.jinr.ru
T2_RU_INR	INR	lcg-ls srm://	grse001.inr.troitsk.ru

T2_RU_ITEP			
T2_RU_PNPI		srmls srm://	cluster.pnpi.nw.ru
T2_RU_SINP		lcg58.sinp.msu.ru	
T2_TH_CUNSTDA		terbium.lsr.nectec.or.th	
T2_TR_METU	METU	lcg-ls srm://	eymir.grid.metu.edu.
T2_UA_KIPT	KIPT	lcg-ls srm://	cms-se0.kipt.kharkov

T2_UK_London_Brunel	Brunel	lcg-ls srm://	dc2-grid-64.brunel.ac
T2_UK_London_IC	IC	lcg-ls srm://	gfe02.grid.hep.ph.ic.

T2_UK_SGrid_RALPP	Rutherford-PPD	lcg-ls srm://	heplnx204.pp.rl.ac.ul
T2_UK_SouthGrid_RALPPD		srmls srm://	heplnx204.pp.rl.ac.ul
T2_UK_SGrid_Bristol		xrdfs root://	lcgse01.phy.bris.ac.u

T2_US_Wisconsin		xrdfs root://	cmssrm.hep.wisc.edu
T2_US_Vanderbilt			se1.accre.vanderbilt.
T2_US_UCSD		gfal-ls gsiftp://	gftp.t2.ucsd.edu
T2_US_Purdue		gfal-ls gsiftp://	cms-gridftp.rcac.purd
T2_US_Nebraska		gfal-ls gsiftp://	red-gridftp.unl.edu

T2_US_MIT		gfal-ls gsiftp://	se01.cmsaf.mit.edu
T2_US_Florida		gfal-ls gsiftp://	cmsio.rc.ufl.edu
T2_US_Caltech			cit-se.ultralight.org
T3_BG_UNI_SOFIA		gfal-ls srm://	dcache.grid.uni-sofia
T3_BY_NCPHEP		gfal-ls gsiftp://	grid02.hep.by
T3_CH_CERN_CAF			eoscmsftp.cern.ch
T3_CH_CERN_HelixNebula			eoscms.cern.ch
T3_CH_CERN_OpenData			
T3_CH_PSI	PSI	lcg-ls srm://	t3se01.psi.ch
T3_CH_Volunteer			

T3_CN_PKU			
T3_FR_IPNL	IN2P3 -IPNL	lcg-ls srm://	lyogrid06.in2p3.fr
T3_GR_IASA_HG	IASA	lcg-ls srm://	se01.marie.hellasgrid
T3_HR_IRB			grid09.phy.pku.edu.c
T3_HU_Debrecen	lcg-ls srm://	dpm.grid.atomki.hu	/dpm/grid.atomki.hu/
T3_IN_TIFRCloud		srmls srm://	se01.indiacms.res.in
T3_IN_VBU			storage.vb-ehep.in
T3_IR_IPM		srmls srm://	se1.particles.ipm.ac.i
T3_IT_Bologna			
T3_IT_Opportunistic		srmls srm://	stormfe1.pi.infn.it
T3_IT_Perugia		srmls srm://	gridse2.pg.infn.it
T3_IT_Trieste		xrdfs root://	eosinfnts.ts.infn.it

T3_KR_KISTI		gfal-ls srm://	cms-se.sdfarm.kr
T3_KR_KNU		gfal-ls srm://	cluster142.knu.ac.kr
T3_KR_UOS		gfal-ls srm://	uosaf0007.sscc.uos.a
T3_MX_Cinvestav		gfal-ls gsiftp://	proton.fis.cinvestav.r

T3_RU_FIAN		se2.grid.lebedev.ru	
T3_RU_MEPhI			cms-phedex.lxfarm.m
T3_RU_SINP		srmls srm://	lcg87.sinp.msu.ru
T3_TH_CHULA			cms-se.sc.chula.ac.th
T3_TW_NCU	NCU	lcg-ls srm://	grid71.phy.ncu.edu.t
T3_TW_NTU_HEP			ntugrid4.phys.ntu.edu
			ntugrid6.phys.ntu.edu
T3_UK_GridPP_Cloud		gfal-ls srm://	gfe02.grid.hep.ph.ic.a

T3_UK_London_QMUL			gfe02.grid.hep.ph.ic.a
T3_UK_London_RHUL			gfe02.grid.hep.ph.ic.a
T3_UK_SGrid_Oxford			heplnx204.pp.rl.ac.uk
T3_UK_ScotGrid_GLA	UKI-SCOTGRID-GLASGOW	lcg-ls srm://	svr018.gla.scotgrid.a
T3_US_ANL			
T3_US_Baylor			kodiak-se.baylor.edu
T3_US_Brown			srm.hep.brown.edu
T3_US_Colorado			hepse01.colorado.edu
T3_US_Cornell			sg-se.cac.cornell.edu
T3_US_FIT			uscms1-se.fltech-grid
T3_US_FIU		gfal-ls srm://	srm.hep.fiu.edu
T3_US_FNALLPC		gfal-ls srm://	cmseos.fnal.gov
T3_US_FSU		gfal-ls srm://	se.hep.fsu.edu
T3_US_HEPCloud			hepcloud-poc.storage
T3_US_MIT		gfal-ls srm://	t3serv006.mit.edu
T3_US_Minnesota		gfal-ls / srmls srm://	gc1-se.spa.umn.edu
T3_US_NERSC			cmsdcadisk01.fnal.go
T3_US_NEU			
T3_US_NotreDame		gfal-ls gsiftp://	deepthought.crc.nd.e

T3_US_OSG			cmsdcadisk01.fnal.g
T3_US_OSU		gfal-ls srm://	cms-0.mps.ohio-state
T3_US_Princeton_ARM		xrdfs root://	xrootd.unl.edu
T3_US_PuertoRico		gfal-ls srm://	cms-se.hep.uprm.edu
		gfal-ls gsiftp://	cms-grid0.hep.uprm.
T3_US_Rice		gfal-ls gsiftp://	bonner07.rice.edu
T3_US_Rutgers		gfal-ls srm://	ruhex-osgce.rutgers.
T3_US_SDSC			cmsdcadisk01.fnal.g
T3_US_TACC			cmsdcadisk01.fnal.g
T3_US_TAMU		gfal-ls srm://	srm.brazos.tamu.edu
T3_US_TTU		gfal-ls srm://	sigmorgh.hpcc.ttu.ed
T3_US_UB		gfal-ls srm://	u2-grid.ccr.buffalo.e
T3_US_UCD		gfal-ls srm://	
T3_US_UCR		gfal-ls srm://	charm.ucr.edu
T3_US_UCSB		gfal-ls gsiftp://	cms25.physics.ucsb.
T3_US_UIowa		gfal-ls srm://	grow-grid.its.uiowa.
T3_US_UMD		gfal-ls srm://	hepcms-0.umd.edu

T3_US_UMiss		gsiftp://	umiss005.hep.olemis
T3_US_Wisconsin			

Local Stageouts

Local stageouts come from SITECONF repos in gitlab.cern.ch/SITECONF.

Sources of information:

1) https://indico.cern.ch/event/5490/contributions/1207408/attachments/979295/1391796/PhEDEx_tuto.pdf

2) The comment on top of https://github.com/dmwm/PHEDEX/blob/master/Custom/Template/storage.xml describes the LFN to PFN convertion.

3) The testing procedure of stage outs: https://twiki.cern.ch/twiki/bin/view/CMSPublic/SAMMonteCarlo

Compute resources

References

A general description of the CMS Global Pool can be found in GlideinWMS-Factory-CERN.pdf.

The original JIRA ticket with introductory and "CMS operational" material is CRIC-2.

The different components of the GlideinWMS based CMS job submission infrastructure are:

- the GlideinWMS factory: see in particular the configuration section for more details on the factory configuration XML files available on GitHub
- the VO Frontend: see in particular the configuration section for more details on the factory configuration XML file;
- More details about VO frontend and job matchmaking are available here, here, and here

OSG has documented some recipes and examples for:

- submitting jobs to HTCondor-CE
- Job Router recipes

If you really want to know more about HTCondor, you can look at:

- the condor_submit command
- The job submission
- Grid computing with HTCondor, in particular
  - The Grid universe
  - The Job Router

Queues

It has been decided that to bootstrap data into CRIC there is going to be one compute resource created, per cms site.

Regarding queues, data will be coming from Glidein factory configurations. A new model will be created in CRIC (Glidein_Entry) where all the info from these configuration files will be stored.

These Glidein_Entry objects will be linked to Core Queues.

The factories contain static information compiled in the XML files. Part of the submission parameters affecting the chosen queue, memory request, etc, can be included in the submit_attrs tag, however a glideinWMS factory entry definition (which corresponds to a gatekeeper and a set of submission parameters) also usually includes a RSL string, written according to the particular CE technology syntax.

The mapping between gridtype attribute in GlideinWMS entry and CRIC CE flavour is as follows:

GlideinWMS gridtype Attribute	CRIC CE Flavour
nordugruid	ARC-CE
cream	CREAM-CE
gt2 (default), gtX, X>2	GLOBUS-CE (default for OIM collector)
condor	HTCONDOR-CE

Some factory entry attributes are linked to one another: the combination of gridtype, gatekeeper, and rsl allows to identify the requirements for a job to be handled by a CE and land to the associated batch system. Another useful tag in the XML entry description is infosys_ref. This is what we found up to now for all CE flavours, but HTCondor. The format of the gatekeeper+queue identifier (in short CEId) is of the form: CEId = gatekeeper:port/queue with queue = gatekeeper_type-batch_type-batch_queue where gatekeeper_type is:

CRIC CE Flavour	gatekeeper_type	Default port
CREAM-CE	cream	8443
ARC-CE	nordugrid	2811
GLOBUS-CE	jobmanager	2119
HTCONDOR-CE	condor	9619

This value is available on BDII as part od the GlueCEUniqueID attribute.

gridtype Attribute	gatekeeper Attribute	rsl Attribute	infosys_ref Tag
cream	Like CEId: gatekeeper:port/cream-batch_type-batch_queue this is a requirement by CREAM CE in order to identify the batch system and the corresponding batch queue where possibly assign the job.	Specifies further requirements for the job to run on that queue (e.g. memory limits)	If present, it contains the GlueCEU on BDII, where the gatekeeper and identified by the CEId: gatekeeper:port/cream-batch_type-b
nordugrid	gatekeeper	It contains a parameter in the form (queue=batch_queue) If not present, by comparison with the GlueCEUniqueID attribute, it is equivalent to (queue=default). It also specifies further requirements for the	It contains the GlueCEUniqueID at where the gatekeeper and queue are CEId: gatekeeper:port/nordugrid-batch_ty

		job to run on that queue (e.g. CPU count, memory limits)	
gt2	gatekeeper:port/jobmanager-batch_type this is a requirement by Globus CE in order to identify the batch system running the job.	If present, it contains a parameter in the form (queue=batch_queue). If not present, by comparison with the GlueCEUniqueID attribute, it is equivalent to (queue=default). It can specify further requirements for the job to run on that queue: as of now, only (jobtype=single) appears there.	It contains the GlueCEUniqueID att where the gatekeeper and queue are CEId: gatekeeper:port/jobmanager-batch_

Questions:

1) It seems that there could be multiple CMS_Queues associated with one core_queue (e.g CMS_T2_US_Nebraska_Red_gw1, CMS_T2_US_Nebraska_Red_gw1_whole, CMS_T2_US_Nebraska_Red_gw1_whole_cms all seem to point on the same core_queue even thought values change) is this the case and if yes how would the users want to access and edit this information through the UI?

2) Could this many to one relation between cms_queues and core_queues have impact on accounting and monitoring?

On the following table there are all the attributes that could be part of a glidein factory entry (a cms_queue in cric schema). We will have to decided on the description of every attribute (users will be able to see this on the front end) and maybe treat some attributes differently (e.g. only populate them from preselected values etc).

Attribute name	Text Description	Comments
vos_using_se_basepath		
gsi_delegation_keybits		
glidein_req_mupj_glexec		
glidein_max_walltime	Maximum walltime allowed.	
tcp_keepalive_interval		
perentrymaxglideins		
glidein_se_voname_lowercase		
glidein_required_os	Required OS.	
glidein_slotslayout		
removesleep		
glidein_proxyurl		
glidein_cpus	Number of cores requested.	Default is 8 and if the site has exceptions they apply them.
entryvmid		
use_ccb		
glexec_job		
glidein_ses		
glidein_se_basepath		

glidein_trustdomain		Possible values are: grid, Wigner, CERNCAF, T0
requireglideinglexecuse		
gridtype	Type of the CE the queue is using.	Possible values are: cream, condor, nordugrid, gt2, ec2
trustdomain		Possible values are: grid, Wigner, CERNCAF, T0
glidein_gatekeeper		First one is the gatekeeper just by itself (hostname). Second is the gatekeeper with the port.
perentrymaxheld		
vos_using_se_other_subdir		This is always empty
glidein_require_glexec_use		
glexec_bin	Security implementation.	Possible values are: glite, NONE, OSG
gcb_order		This is always None
allowedvos		This is always empty
whitelistmode		This is always off
perfrontendmaxglideins		
glidein_job_min_time		
glidein_workdir		
marians_var		Possible values are: auto
glidein_supportedauthenticationmethod		Possible values are: grid_proxy, key_pair
perentrymaxidle		
defaultperfrontendmaxheld		
glidein_resource_slots		
startupdir		
perfrontendmaxidle		
releasesleep		
glidein_glexec_vos		Possible values are: CMS
gatekeeper		
enable_ipv6	Flag for ipv6 enabled queues.	
glidein_maxmemmbs	Requested maximum memory.	
verbosity		Possible values are: std
glidein_globusrsl		
glidein_is_long		
glidein_retire_time		
glidein_site		
site_req_explicit_auth		
submitcluster		
glidein_require_voms		
maxreleaserate		
submitslotslayout		Possible values are: partitionable, fixed
defaultperfrontendmaxidle		
glidein_is_preemptable		
glidein_singularity_use		
maxremoverate		
factorytype		Possible values are: production
glidein_nickname		

glidein_resourcename		
glidein_supported_vos		
perfrontendmaxheld		
glidein_maxmemmbs_estimate		
entryvmtype		
glidein_gridtype	The same as gridtype	Possible values are: cream ,condor, nordugrid, gt2, ec2
authmethod	Authentication method.	Possible values are: grid_proxy, key_pair
submitsleep		
all_debug		
defaultperfrontendmaxglideins		
proxyurl	Proxy URL.	Possible values are: OSG
globusrsl		
glidein_verbosity		Possible values are: std
maxsubmitrate	Maximum submition rate.	
requirevomsproxy		
glidein_retire_time_spread		
glidein_country	Country of the site that provides the queue.	
vos_using_se_voname_lowercase		This is empty in all entries.
glidein_cmssite	Name of the site providing the queue.	

Mapping Glidein entries to core queues (GOCDB/OIM/OSG/BDII)

We need to relate queues, coming from the `gatekeeper` field of the configuration xml, to core queues.

The first step to do that is to relate these gatekeepers with CE s already in cric (and then connect them with the corresponding core queue).

In order to achieve this we have taken all the gatekeepers from the xml and stripped them down to the base url (removing queue names and ports) and tried to match them to CE s endpoints in CRIC.

After that we will be able to link these entries to core queues.

Inconsistent CEs found

Services that exist in OIM but don't have the flavour set:

Endpoint	Site	Flavour
its-condor-ce.syr.edu	SU-ITS	HTCONDOR-CE
its-condor-ce.syr.edu	SU-ITS	HTCONDOR-CE
its-condor-ce.syr.edu	SU-ITS	HTCONDOR-CE
its-condor-ce1.syr.edu	SU-ITS	HTCONDOR-CE
its-condor-ce1.syr.edu	SU-ITS	HTCONDOR-CE
kodiak-ce.baylor.edu	Baylor-Kodiak	HTCONDOR-CE
kodiak-ce.baylor.edu	Baylor-Kodiak	HTCONDOR-CE
kodiak-ce.baylor.edu	Baylor-Kodiak	HTCONDOR-CE
cet01.cern.ch	CERN-PROD	HTCONDOR-CE
iut2-gk.mwt2.org	MWT2	HTCONDOR-CE
iut2-gk.mwt2.org	MWT2	HTCONDOR-CE
uct2-gk.mwt2.org	MWT2	HTCONDOR-CE

uct2-gk.mwt2.org	MWT2	HTCONDOR-CE
mwt2-gk.campuscluster.illinois.edu	MWT2	HTCONDOR-CE
mwt2-gk.campuscluster.illinois.edu	MWT2	HTCONDOR-CE
osg-ce.clemson.edu	Clemson-Palmetto	HTCONDOR-CE
osg-ce.clemson.edu	Clemson-Palmetto	HTCONDOR-CE
ce515.cern.ch	CERN_HelixNebula_CMS	HTCONDOR-CE
ce516.cern.ch	CERN_HelixNebula_CMS	HTCONDOR-CE
cale.uniandes.edu.co	UNIANDES	CREAM-CE
osgce2.hepgrid.uerj.br	UERJ	HTCONDOR-CE
ce3.accre.vanderbilt.edu	Vanderbilt	HTCONDOR-CE
ce3.accre.vanderbilt.edu	Vanderbilt	HTCONDOR-CE
ce3.accre.vanderbilt.edu	Vanderbilt	HTCONDOR-CE
ce4.accre.vanderbilt.edu	Vanderbilt	HTCONDOR-CE
ce4.accre.vanderbilt.edu	Vanderbilt	HTCONDOR-CE
ce4.accre.vanderbilt.edu	Vanderbilt	HTCONDOR-CE
cms25.physics.ucsb.edu	ucsb-cms	HTCONDOR-CE
sandhills-ce1.unl.edu	Sandhills	HTCONDOR-CE
osg-gw-4.t2.ucsd.edu	UCSDT2	HTCONDOR-CE
deepthought.crc.nd.edu	NWICG_NDCMS	HTCONDOR-CE

CEs used by cms but are not in core sources:

Endpoint	Site	Flavour
cet01.cern.ch	CERN-PROD	HTCONDOR-CE
ce515.cern.ch	CERN_HelixNebula_CMS	HTCONDOR-CE
cale.uniandes.edu.co	UNIANDES	CREAM-CE
ce516.cern.ch	CERN_HelixNebula_CMS	HTCONDOR-CE
xeon.hep.caltech.edu	CIT_CMS_T2	HTCONDOR-CE
condorce02.cern.ch	CERN-PROD	HTCONDOR-CE
cream-ce-2.ba.infn.it	INFN-BARI	CREAM-CE
cccreamceli01-t2.in2p3.fr	IN2P3 -CC-T2	CREAM-CE
cccreamceli03-t2.in2p3.fr	IN2P3 -CC-T2	CREAM-CE
cccreamceli04-t2.in2p3.fr	IN2P3 -CC-T2	CREAM-CE
byggvir.princeton.edu	T3_US_Princeton_ARM	HTCONDOR-CE
condorce02.cern.ch	CERN-PROD	HTCONDOR-CE
login5.stampede.tacc.utexas.edu	t3ustacc	GLOBUS

Squids

For bootstrapping squids into CRIC we will be using the site configuration files from the gitlab repos under https://gitlab.cern.ch/SITECONF.

The process is:

1) Get all the files from the `JobConfig` folder of the root of each repo.

2) Parse only the xml files inside JobConfig and search for the `proxy` tag.

3) Match the proxy tag with a squid (`core_service`) using the url.

4) Associate each squid with the compute resource matching the site that the repo has as name.
(e.g. <proxy url="http://cms02.lcg.cscs.ch:3128"/> from
https://gitlab.cern.ch/SITECONF/T2_CH_CSCS_HPC/blob/master/JobConfig/site-local-config.xml will be

associated with the compute resource bootstrapped for the site T2_CH_CSCS_HPC)

Notes:

- The Squid will not be imported if a compute resource cannot be linked to it.

Results:
After trying to bootstrap squids associations to compute resources we ended up with the following results:
194 total squids were identified from gitlab siteconf files

166 were matched to a CR

51 were matched to a core service (squids declared in GocDB).

The detailed results can be found here.

Questions:

1) It looks like there are squids that don't match a core_service squid. Why is that? Should we import those?

2) Some squids even have an ip address as their endpoint (as delcared in siteconf files). Is this normal?

3) We have observed that some squids appear on multiple cms sites configuration files. For example http://ccsquidli04.in2p3.fr:3128 appears in T1_FR_CCIN2P3, T2_FR_CCIN2P3 and T3_FR_IPNL which is expected. The problem is that when we tried to use nslookup to identify which site the squid realy belongs to (physicaly) none of those 3 matched. It looks like the squid is part of a 4th site. Is this normal? How can we identfy the true source of the squid (in order to import those as new core_services)?

Legacy API

The CRIC CMS plugins should provide, along with a new JSON REST API for the objects defined therein, a "legacy" REST API, i.e. fully compatible with the one already available from SiteDB. The output formats of the SiteDB API are JSON and XML, and both should be kept.

Sites

Endpoint: https://cmsweb.cern.ch/sitedb/data/prod/sites

At a first glance, it is a list of labels (in SiteDB webUI jargon, "site title") of the physical and CMS sites registered in SiteDB, along with their "usage in the grid", tier level, and location. The API documentation describes all fields, including some of them which I can consider deprecated. You can find the table with the results in the table above, along with some explanations and request for clarifications. Below some general questions for some of the fields

- Usage: I see 5 values used repeatedly: LCG (for European sites?), OSG (for US sites?), other, Unknown, null. There is also ARC, which is used only once, see later. Does this field designates which grids are supposed to use the site? Can we correlate this field to the Grid IS (GocDB, OIM) where they are registered? Or is this a CMS specific attribute, related to the way resources are managed? In fact, I have found some inconsistencies, e.g.:
    - ♦ the title "Oviedo" is associated (see later) to physical site "UOGRID" which registered in GocDB, but the corresponding usage is "Unknown"
    - ♦ the title "Cern Tier-0" is associated to physical site "CERN-PROD" which is registered in GocDB, but the corresponding usage is "null". Is this because the Tier0 resources have a

    - dedicated WMAgent instance to launch jobs there?
    - at the same, the physical site "CERN-PROD" is also linked to the title "Cern Tier-2" (pointing to T2_CH_CERN): also here, the usage is "null". Does this point out that those resources are to be used exclusively for AlCa /DPG studies?
    - finally, the physical site "CERN-PROD" is also linked to the title "T3_CH_CERN_CAF" (pointing to T3_CH_CERN_CAF): here the usage is "other". Is this because the resources there are accessible via CRAB for analysis, though with some restrictions to special users/groups?
    - the title "FNALLPC" corresponds to the physical site USCMS-FNAL-LPC, which is not registered in OIM; as previously discussed, we are making Tier3 topology information consistent on best effort, though this site appears used by OSG.
    - the ARC usage (I assume for NordUGrid) is set only for "Helsinki Institute of Physics" title: the corresponding physical site FI_HIP_T2 is registered in GocDB. Can this value be changed into LCG, or should we keep it for some NordUGrid specific features?
    - **Clarification**: this attribute could be useful for enabling some features in job submission, but it is not used in GlideinWMS entries. If not used on WMAgent and/or CRAB either, we can drop it.
- URL: it is the web site the institution managing the logical site: should it be kept, or can we use the web pages by GocDB /OIM? * **Clarification**: this attribute is used only in webUI, and is displayed when not None, so it can be dropped.
- Logo URL: it is the URL of the logo of the institution: is it needed? If not, can we point to, say, the CMS logo? * **Clarification**: this attribute is used only in webUI, and is displayed when not None, so it can be dropped.
- Development release: the doc says it is unknown: can it be dropped completely, or can we set a "fake" value? * **Clarification**: this attribute is not used, so it can be dropped.
- Manual Install: the doc says it is unknown: can it be dropped completely, or can we set a "fake" value? * **Clarification**: this attribute is not used, so it can be dropped.

As a general guideline, we should use the opportunity for house cleaning.

A check of the CMSWeb access log shows that this endpoint is queried by Squid monitoring service, so **it should be kept as legacy** in CRIC.

Implications for CRIC models

As far as models are concerned, we should store the resourceID attribute from OIM ResourceGroup (i.e. Site), in the core site model as oim_resourceid integer field, and retrieve it from OIM XML API in the corresponding data provider. This will allow to check from where the Site was retrieved (GocDB if gocdb_pk is available, OIM if oim_resourceid is available, CRIC if none of them).

In the API result, do not retrieve URL, Logo URL, Development release, Manual install. Also, check the Usage attribute.

Validation

Here you will see differences between the JSON output by SiteDB and the one by CRIC.

Site names

Endpoint: https://cmsweb.cern.ch/sitedb/data/prod/site-names

This API returns the association between the "site title" and the physical (with "prefix" LCG) and CMS site (with "prefix" CMS). As a side note, this has been used to associate the CMS Site to a Core site in CRIC. The API documentation describes all fields. You can find the table with the results in the Twiki [2], along with some explanations and request for clarifications. Below some general questions for some of the fields:

- type: it is the kind of resource we are looking at. AFAICT, the meaning is:
    - lcg: Physical grid Site
    - cms: CMS Site
    - phedex: Phedex node
    - psn: the SiteDB webUI translates this into "Processing", but I cannot say more. **Clarification**: PSN = Processing Site Name, it's another concept that we want to clean up/revise.
- site_name: it is the site "title" in SiteDB, see above
- alias: it is the name of the resource

As a general guideline, this is for pure backward compatibility and something we like to phase out with revised site schema.

A check of the CMSWeb access log shows that this endpoint is queried by DMWM, CRAB3 TaskWorker, CMS Site Support, Dashboard, Squid Monitoring services, so **it should be kept as legacy** in CRIC.

Implications for CRIC

As far as models are concerned, we should store the site_name attribute (in SiteDB webUI jargon, "site title", i.e. the label of the physical and CMS sites registered in SiteDB) in the CMS site model as sitedb_title character field, and retrieve it from SiteDB API in the corresponding data provider.

Given the objects currently modeled in CRIC, only grid (lcg type) and CMS (cms type) sites will be retrieved. PSN concept should be revised.

Validation

Here you will see differences between the JSON output by SiteDB and the one by CRIC.

Site resources

Endpoint: https://cmsweb.cern.ch/sitedb/data/prod/site-resources

According to the documentation, this should list all CE and SE for a site.

- I can only see SE listed in the output. Why are CEs non present? Why only CE and SE? E.g. you could have added also Squids (even if they are listed in the SITECONF).
- The site name in the output is actually the site title, so it can point both the physical and to the CMS site. IIUC, the CMS Site is the "pointee" of the API response: in other terms, the API retrieves the list of CE and SE **used** by the CMS Site having a given "site title", hence it exposes the link between core resources and CMS topology. Is this correct?
- What does "is_primary" stand for?

Clarification: CE and SE information is no longer kept in SiteDB, so do not need to appear in "Site Resources". Squids are no longer of interest to us in the information system as we moved to launch pad.

A check of the CMSWeb access log shows that this endpoint is queried by CMS Site Support and CMS Build (service for building CMSSW), so **it should be kept as legacy** in CRIC.

Implications for CRIC

Given the objects currently modeled in CRIC, only CE can be retrieved.

Site associations

Endpoint: https://cmsweb.cern.ch/sitedb/data/prod/site-associations

This API returns the associations between a "parent" CMS site and its "children". These CMS sites are identified via the "site title", so, in order to get the real site names you should see the alias with type "cms" associated to a given "title" in the "site names" API. Can you please elaborate more on this association? E.g. T1_IT_CNAF "children" are T2_IT_Bari, T2_IT_Legnaro, T2_IT_Pisa, T2_IT_Rome, T3_IT_Bologna, T3_IT_Firenze, T3_IT_MIB. Is a "child" site (usually a T2/T3) a CMS site that can possibly use the resources in the "parent" site, say access dataset files available in SEs used by the "parent"?

Clarification: The parent/child association is used by WMagent to decide on additional processing sites for data at/to be stored at a parent site. We have to investigate to what extend this is still being used or if there are other usages, e.g. if this feature could be used also in CRAB, or are the user jobs completely managed in the GlobalPool.

A check of the CMSWeb access log shows that this endpoint is queried by a single user, so **it should not be kept as legacy** in CRIC.

Implications for CRIC models

If this association is actively used, we have to add a "self-pointing" ForeignKey in the CMSSite model, and put some constraints on the tier level.

Meetings with CMS Glideins experts

Instructions to create a VM running CRIC:

1. Check your permissions (http://configtraining.web.cern.ch/configtraining/introduction/permissions.html) in order to create a vm using puppet. Also make sure you are subscribed to `ai-admins` and `LxAdm-Authorized-Users` egroups.
2. SSH into a `aiadm.cern.ch` machine with your cern username and run the following commands (replacing `{hostname}` with the desired hostname) to create the vm in the 'cric' cluster:

   ```
   eval $(ai-rc 'cric')
   ai-bs --cc7 --foreman-environment production --foreman-hostgroup cric/cms --landb-responsi
   ```

   In ~30 mins to 1 hour, the VM should be ready (you can check its status on the openstack portal after selecting the right project in the top menu bar).
3. SSH into your new VM

   ```
   ssh root@cms-cric-dev-XX.cern.ch
   ```

   and launch a shell as user 'criclocal'

   ```
   sudo -u criclocal sh
   ```
4. Run the following commands to load and configure the virtual environment for CRIC and to run database migrations

   ```
   source /data/cric/workspace/cms/bin/activate; export PYTHONPATH=/etc/cric/web/:${PYTHONPAT
   python -m cric.web.cms.manage makemigrations --all; python -m cric.web.cms.manage migrate
   ```
5. To populate the database, change directory to /data/cric/git and set DJANGO_SETTINGS_MODULE

   ```
   cd /data/cric/git; export DJANGO_SETTINGS_MODULE='cric.web.cms.settings';
   ```

   Use the following commands to fetch data from the collectors,

   ```
   ## fetch sites & services from GOCDB/OIM source
   python -u lib/cric/apps/cric_crons/crons/manager.py --cron OIMGOCDBInfoLoaderCron -c start
   ## init regional centers
   python -u lib/cric/apps/cric_crons/crons/manager.py --cron GStatLoaderCron -c start -o col

   ## collect pledges
   python -u lib/cric/apps/cric_crons/crons/manager.py --cron GStatLoaderCron -c start -o col
   python -u lib/cric/apps/cric_crons/crons/manager.py --cron GStatLoaderCron -c start -o col
   python -u lib/cric/apps/cric_crons/crons/manager.py --cron GStatLoaderCron -c start -o col
   python -u lib/cric/apps/cric_crons/crons/manager.py --cron GStatLoaderCron -c start -o col

   ### collect Downtimes from GOCDB for last 90 days for ALL the sites (skip_unknown_sites=Fa
   ### and push data via REST API
   python -u lib/cric/apps/cric_crons/crons/manager.py --cron GOCDBLoaderCron -c start create

   ## collect (HTCondor) CE & (local) Queues from OSG source
   python -u lib/cric/apps/cric_crons/crons/manager.py --cron OSGLoaderCron -c start createjs

   ## collect Site details, (HTCondor) CE and (local) Queues from BDII
   python -u lib/cric/apps/cric_crons/crons/manager.py --cron BDIILoaderCron -c start createj
   # collect SiteDB information:
   python -u lib/cric/apps/cric_crons/crons/manager.py --cron SITEDBInfoLoaderCron -c start c

   # collect Glidein entries from github
   python -u lib/cric/apps/cric_crons/crons/manager.py --cron GithubLoaderCron -c start creat

   # collect squids information from gitlab-hosted site configuration files
   # Note: This collector uses the GitLab api which requires a token to run.
   ```

```
# in case of issues check token value or add your own at settings_local.py under the name
# Instruction to get the token: https://docs.gitlab.com/ce/user/profile/personal_access_to
python -u lib/cric/apps/cric_crons/crons/manager.py --cron GITLABLoaderCron -c start creat

# collect Storage informations from vofeed
python -u lib/cric/apps/cric_crons/crons/manager.py --cron VofeedLoaderCron -c start creat

python -u lib/cric/apps/cric_crons/crons/manager.py --cron SITEDBAuthLoaderCron -c start c
```

6. Register your application for Shibboleth by filling this form:

- insert `{hostname}` into the field "Application Name"
- insert `https://{hostname}.cern.ch/Shibboleth.sso/ADFS` into the field "Application Uri"
- insert `https://{hostname}.cern.ch/` into the field "Application Homepage"
- fill the application description
- click the button "Send Registration Request"

For instructions to update the VM or learning more details about the steps described above, please read this document on GitLab. Old instructions for setting up a VM, using the qa environment instead of production environment, can be found here.

Model iterations with Stephan

http://lammel.web.cern.ch/lammel/cric/cu_example.html

http://lammel.web.cern.ch/lammel/cric/su_example.html

http://lammel.web.cern.ch/lammel/cric/facility_example.html

http://lammel.web.cern.ch/lammel/cric/site_example.html

* Link to initial CRIC CMS page*

Archive

-- SalvatoreDiGuida - 2017-02-02