INSTALLATION Laser MEGA JOULE

DE LA RECHERCHE À L'INDUSTRIE

PARC: A COMPUTIONAL SYSTEM IN SUPPORT OF LMJ FACILITY OPERATIONS

Jean-Philippe Airiau / CEA-CESTA

ICALEPCS2017 – WEAPL05

ICALEPCS2017
Barcelona · Spain, October 8-13 · Palau de Congressos de Catalunya

www.cea.fr

CONTEXT

The quest for performance is a recurring theme when managing feedback software for large physics instrument.

The constant increase of computational power coupled with software engineering can break through barriers that seemed unreachable a few years ago.

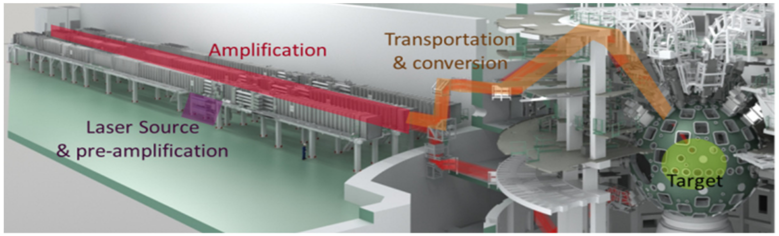
For the Laser MegaJoule (LMJ), the CEA faced a big challenge in order to meet the specifications of an automatic predictive and reporting system.

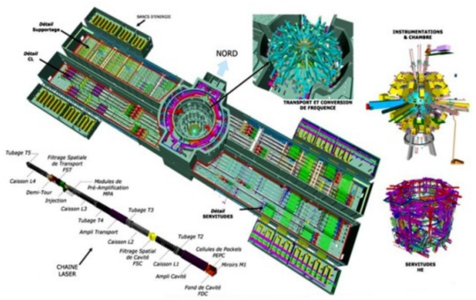
Such a system is able to :

- Manage multiple fields of application : laser, synchronization and alignment,
- Deal with heterogeneous data (type and units) coming from Control Command System,
- Gather and run a lot of heteregeneous micro-software (langage and API),
- Integrate heavy laser simulation software : Miró,
- Run thousands of computation in a few minutes (176 beams),

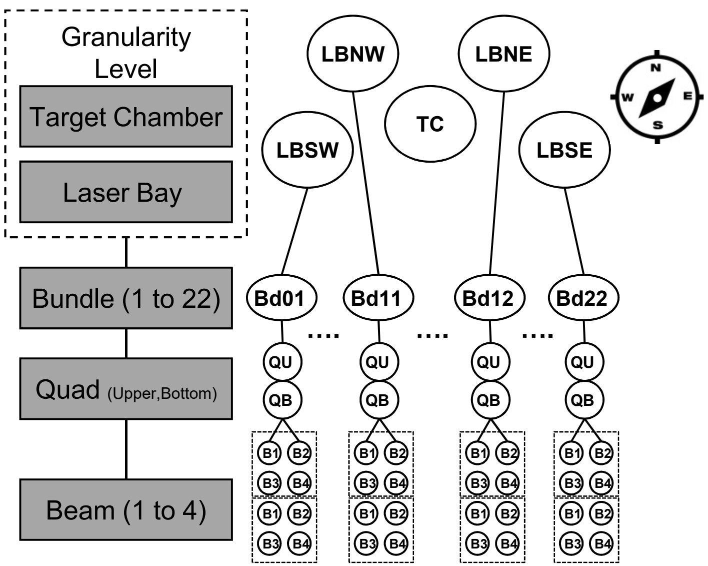
PARC solves this challenge and enhances the overall process.

ENVIRONMENT

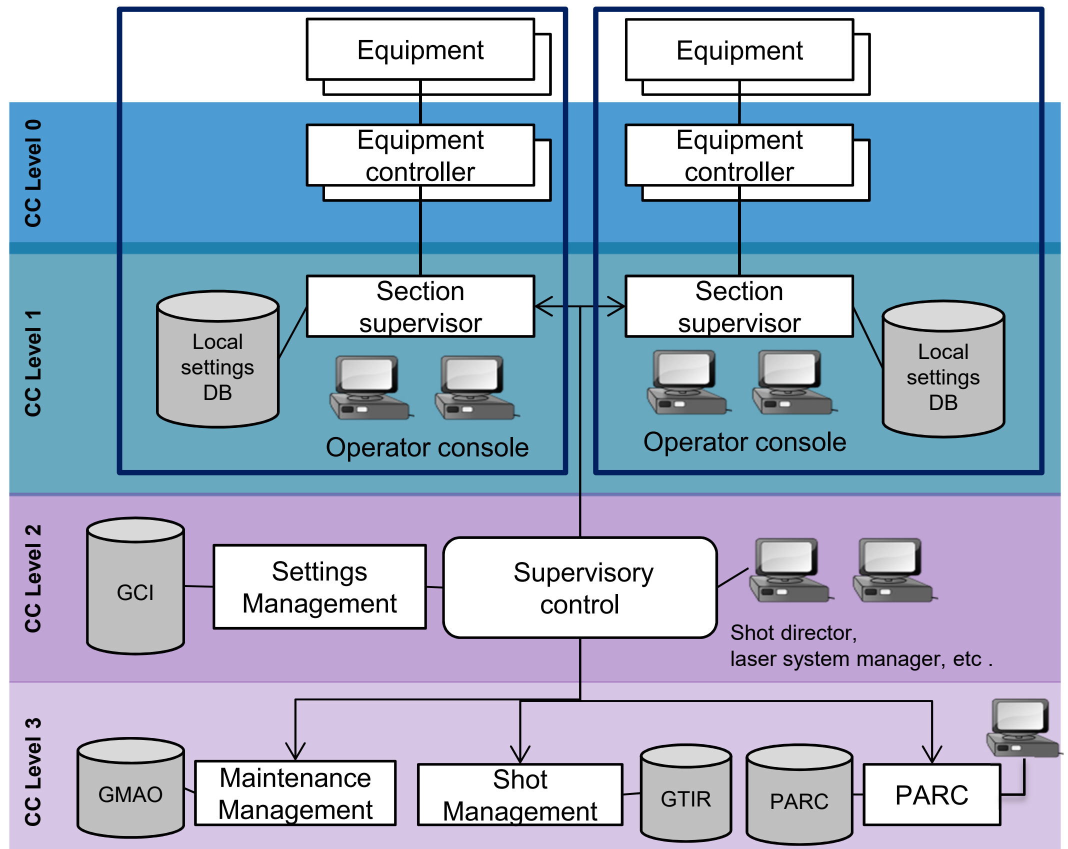




22 bundles / 44 quads/ 176 beams



- ➢ **LMJ facility design**
- ➢ **Granularity level concept**
- ➢ **Control command architecture**



GCI : LMJ settings manager system
GMAO : LMJ logistic manager system
GTIR : LMJ shot results manager system

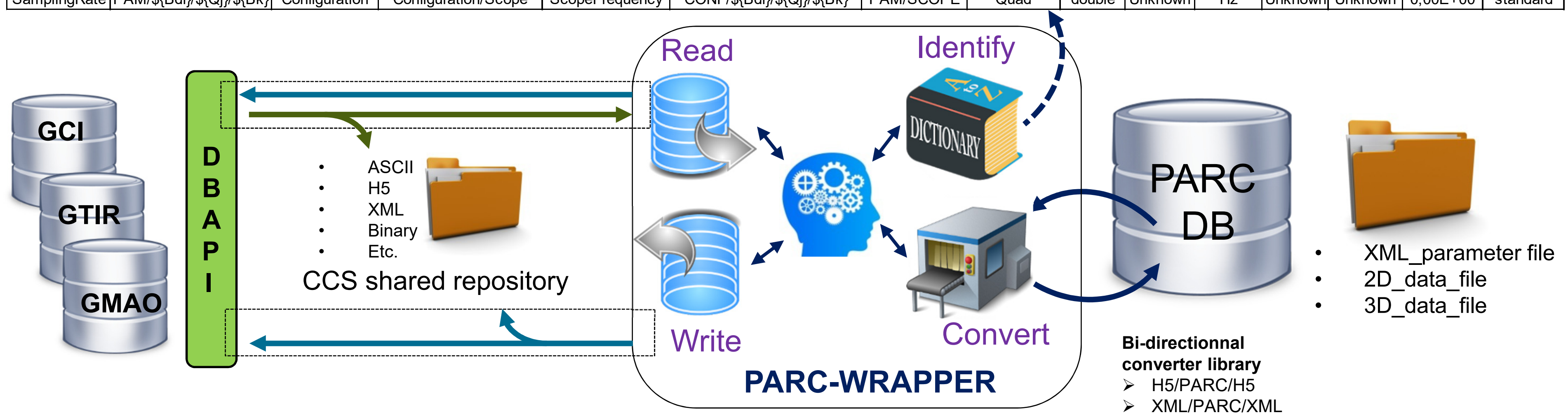
DATA EXCHANGE WITH CONTROL COMMAND SYSTEM (CCS)

- Identify data source producer
- Read from CSS and convert for PARC
- Produce homogeneous data (type, unit system, validity)
- Identify data target consumer
- Convert and write for CCS

<u>Example of wrapping :</u>
PARC needs the scope frequency for Pre-Amplifier Module (PAM) mounted on the upper quad of Bundle 02

Instanciate template id in function of level granularity and experiment perimeter

CCS				PARC										
Name	**FCN**	**Primary id**	**Secondary id**	**Name**	**PhysicalPath**	**MemoryPath**	**Granularity**	**Type**	**Format**	**Val_unit**	**X_unit**	**Y_unit**	**Default**	**Model**
SamplingRate	PAM/${Bdi}/${Qj}/${Bk}	Configuration	Configuration/Scope	ScopeFrequency	CONF/${Bdi}/${Qj}/${Bk}	PAM/SCOPE	Quad	double	Unknown	Hz	Unknown	Unknown	0,00E+00	standard



Bi-directionnal converter library
- H5/PARC/H5
- XML/PARC/XML
- Binary/PARC/Binary
- ASCII/PARC/ASCII
- Miró/PARC/Miró
- CCS units / ISU / CCS units

DATA EXCHANGE WITH CONTROL COMMAND SYSTEM (CCS)

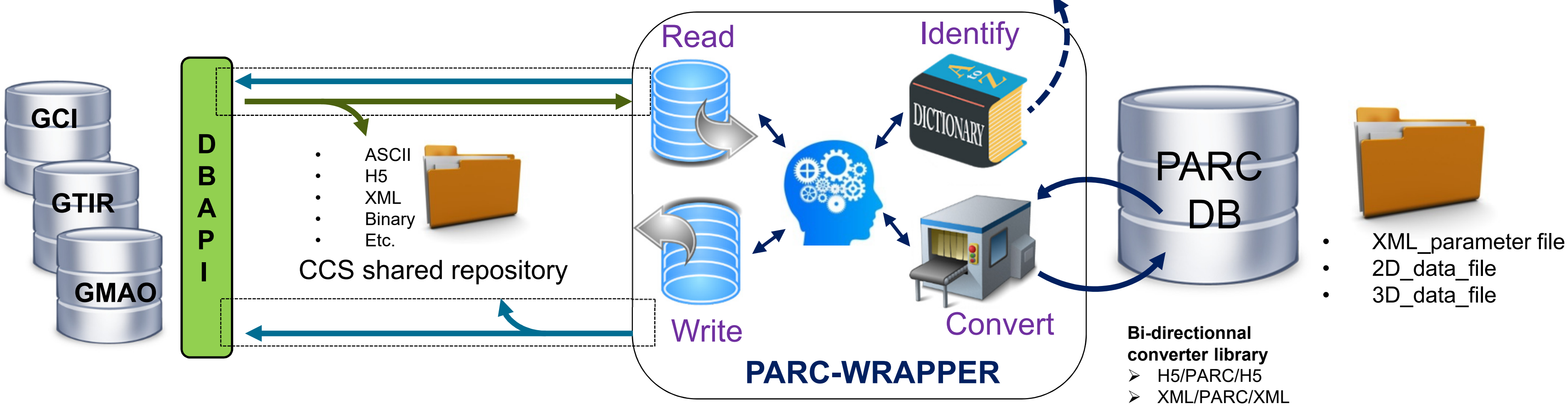
- ➢ Identify data source producer
- ➢ Read from CSS and convert for PARC
- ➢ Produce homogeneous data (type, unit system, validity)
- ➢ Identify data target consumer
- ➢ Convert and write for CCS

Instanciate template id in function of level granularity and exp perimeter

Identify in dictionary:

- ➢ PARC gets corresponding CCS template ids
- ➢ PARC instanciates template Ids for Bundle 02 and Upper Quad
- ➢ PARC builds the request :
  - FCN = PAM/Bd02/QU/B1
  - Primary id = Configuration
  - Secondary id = Configuration/Scope

CCS				PARC											
Name	**FCN**	**Primary id**	**Secondary id**	**Name**	**PhysicalPath**	**MemoryPath**	**Granularity**	**Type**	**Format**	**Val_unit**	**X_unit**	**Y_unit**	**Default**	**Model**
SamplingRate	PAM/${Bdi}/${Qj}/${Bk}	Configuration	Configuration/Scope	ScopeFrequency	CONF/${Bdi}/${Qj}/${Bk}	PAM/SCOPE	Quad	double	Unknown	Hz	Unknown	Unknown	0,00E+00	standard



Bi-directionnal converter library

- ➢ H5/PARC/H5
- ➢ XML/PARC/XML
- ➢ Binary/PARC/Binary
- ➢ ASCII/PARC/ASCII
- ➢ Miró/PARC/Miró
- ➢ CCS units / ISU / CCS units

DATA EXCHANGE WITH CONTROL COMMAND SYSTEM (CCS)

- Identify data source producer
- Read from CSS and convert for PARC
- Produce homogeneous data (type, unit system, valic
- Identify data target consumer
- Convert and write for CCS

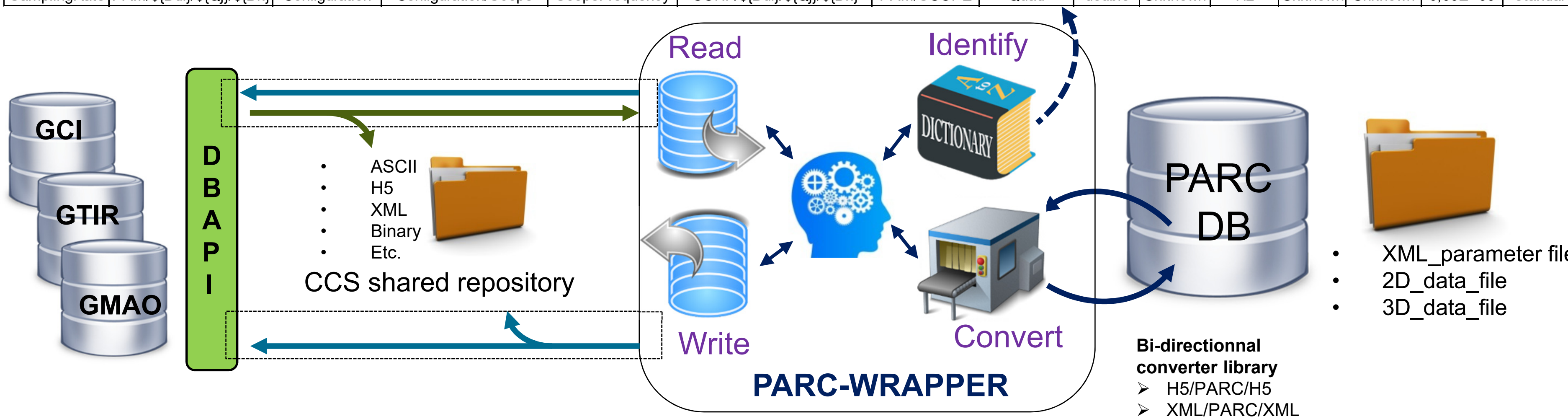
Instanciate template id in function of level granularity ar
perimeter

Read in CCS DB:

- PARC sends the request and wait for the answer
- CCS DB API copy the file in the shared repository and returns the path
- PARC reads the H5 file and access the variable :
  - Configuration/Scope/SamplingRate

The variable is an integer with the value 2000000000, unit = sample per second, validity = true

CCS				PARC										
Name	FCN	Primary id	Secondary id	Name	PhysicalPath	MemoryPath	Granularity	Type	Format	Val_unit	X_unit	Y_unit	Default	Model
SamplingRate	PAM/${Bdi}/${Qj}/${Bk}	Configuration	Configuration/Scope	ScopeFrequency	CONF/${Bdi}/${Qj}/${Bk}	PAM/SCOPE	Quad	double	Unknown	Hz	Unknown	Unknown	0,00E+00	standard



Bi-directionnal converter library

- H5/PARC/H5
- XML/PARC/XML
- Binary/PARC/Binary
- ASCII/PARC/ASCII
- Miró/PARC/Miró
- CCS units / ISU / CCS units

DATA EXCHANGE WITH CONTROL COMMAND SYSTEM (CCS)

- Identify data source producer
- Read from CSS and convert for PARC
- Produce homogeneous data (type, unit system, vali
- Identify data target consumer
- Convert and write for CCS

Instanciate template id in function of level granularity ar perimeter

Convert to PARC homogeneous format :

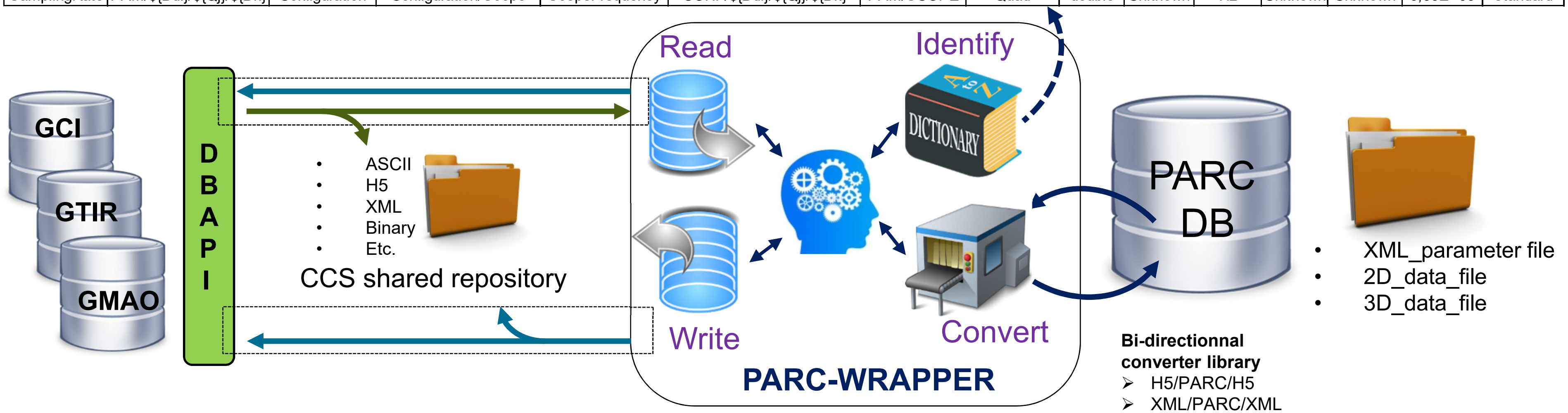
- PARC converts the integer in double, the value from 2000000000 to 2,00^{e}09, the unit from ‘sample per second’ to ‘Hz’
- PARC gets corresponding PARC ids
  - Path = CONF/Bd02/QU
  - Memory Path = PAM/Scope
- PARC create a XML parameter file including the following parameter:

Name = ScopeFrequency , Type= double, Val_unit = Hz, Value =2,00^{e}09 , status=0, …

- PARC saves the XML parameter file at path :

\\BD-PARC\CONF\Bd02\QU\Conf_Param_201709250923.xml

CCS				PARC										
Name	FCN	Primary id	Secondary id	Name	PhysicalPath	MemoryPath	Granularity	Type	Format	Val_unit	X_unit	Y_unit	Default	Model
SamplingRate	PAM/${Bdi}/${Qj}/${Bk}	Configuration	Configuration/Scope	ScopeFrequency	CONF/${Bdi}/${Qj}/${Bk}	PAM/SCOPE	Quad	double	Unknown	Hz	Unknown	Unknown	0,00E+00	standard



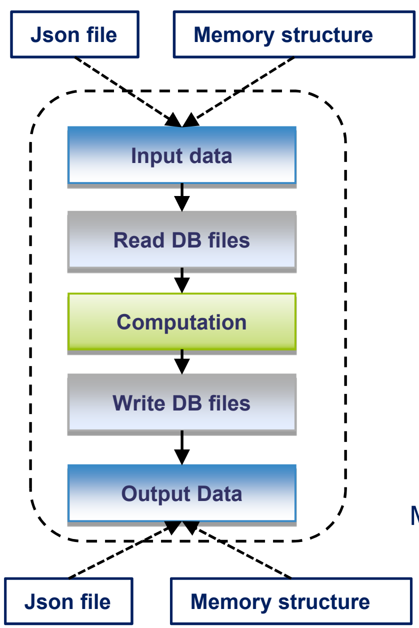
Bi-directionnal converter library

- H5/PARC/H5
- XML/PARC/XML
- Binary/PARC/Binary
- ASCII/PARC/ASCII
- Miró/PARC/Miró
- CCS units / ISU / CCS units

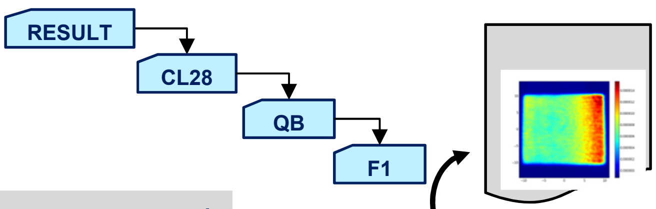
COMPUTATION MODULE

- Short source file : python, IDL or C++.
- Performs specific feature (e.g. : get mean fluence).
- Generic design ensure reusability.
- Same mapping between memory structure / storage file format.
- Description file i(name, langage, input, output, sub-module, etc.)

Computation module code



Storage file format



parameter.xml

-CL28QBF1_MDI_FLUENCE_Fluence.flu

```
<MEMOIRE Niveau="MDI">
<MEMOIRE Niveau="FLUENCE">
     <Param NomParam="Moyenne" Type="double" Unite_val="J.m^-2" Valeur="1.3500000E+01"/>
     <Param NomParam="Fluence" Type="URL" " Format="FLU"  Unite_val="J.m^-2" Unite_x="m"
      Unite_y="m" Valeur="CL28QBF1_MDI_FLUENCE_Fluence.flu"/>
</MEMOIRE>
```

Read (toolbox)

Memory.input.MDI.FLUENCE.fluence.value = [2D array]

Computation

Memory.output.MDI.FLUENCE.Mean.value = 1.3500000E+01

Write

parameter.xml

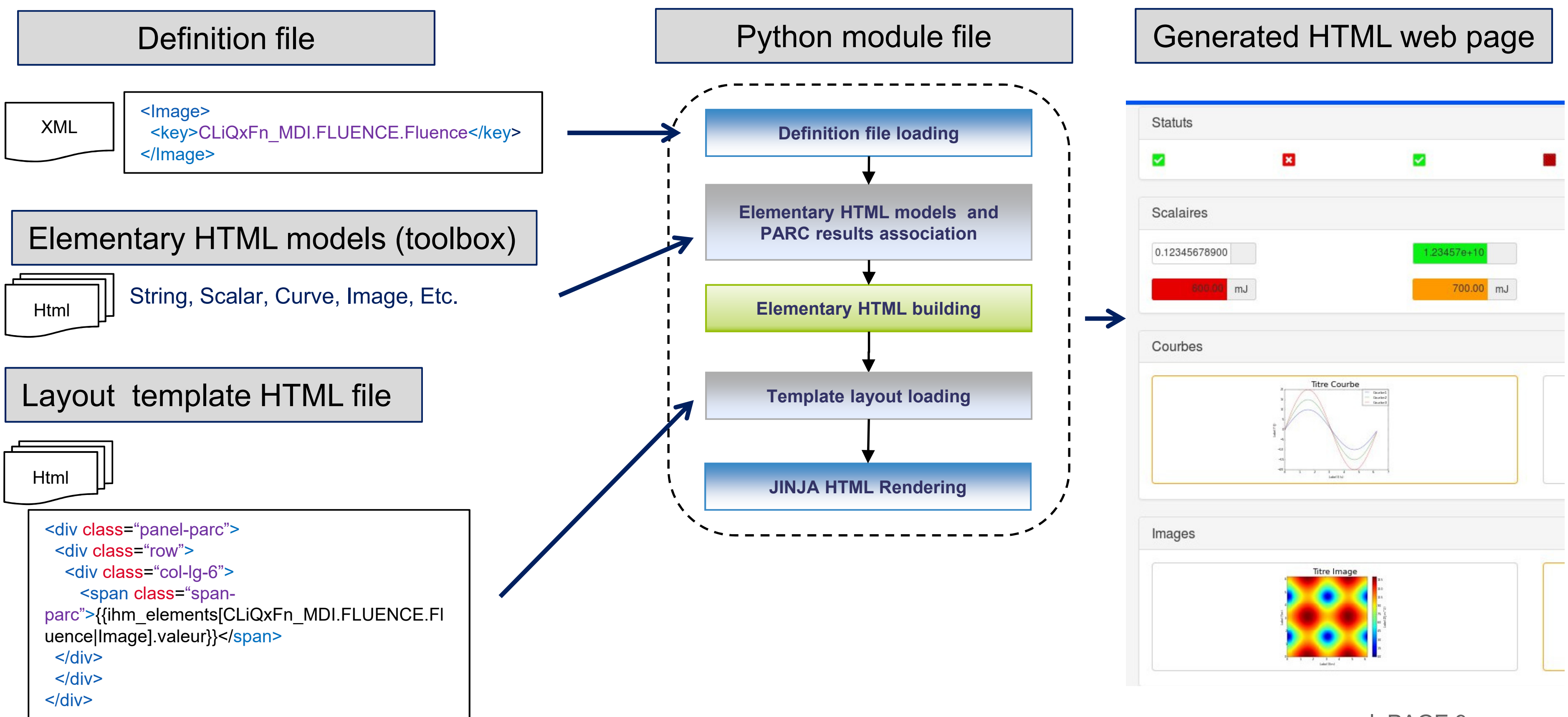
XML description file

```
<?xml version="1.0" encoding="utf-8"?>
<!-- Renseignement des méta-données associées au module : nom du module, langage du
code et description des fonctionnalités du module -->
<Module xmlns="XMLSchema_Module.xsd"
        Nom="Nom_Module"
        Version="2014_11_18_1100"
        Langage="Python"
        Description="Description_Module">
  <!-- Renseignement des entrées du module -->
  <Entrees>
    <!-- A dupliquer : une balise par entrée -->
    <Entree Nom="Nom_Entree1"
            Type="double"
            Option="1"
            Description="Description_Entree1"/>
  </Entrees>
  <!-- Renseignement des sorties du module -->
  <Sorties>
    <!-- A dupliquer : une balise par sortie -->
    <Sortie Nom="Nom_Sortie1"
            Type="double"
            Description="Description_Sortie1"/>
  </Sorties>
  <!-- Renseignement des modules appelés dans le code du module -->
  <ModulesAppeles>
    <!-- A dupliquer : une balise par module appelé -->
    <ModuleAppele Nom="Nom_Module_Appele" Version="Version_Module_Appele"/>
  </ModulesAppeles>
  <!– Lien vers le code industrialisé du module (renseigné automatiquement par PARC
lors de l’intégration du module -->
  <Code Chemin ="Code_Module_Courant.py"/>
</Module>
```

GUI MODULE

A GUI module describes the content of a HTML web page (parametrization or reporting).
It is composed of :

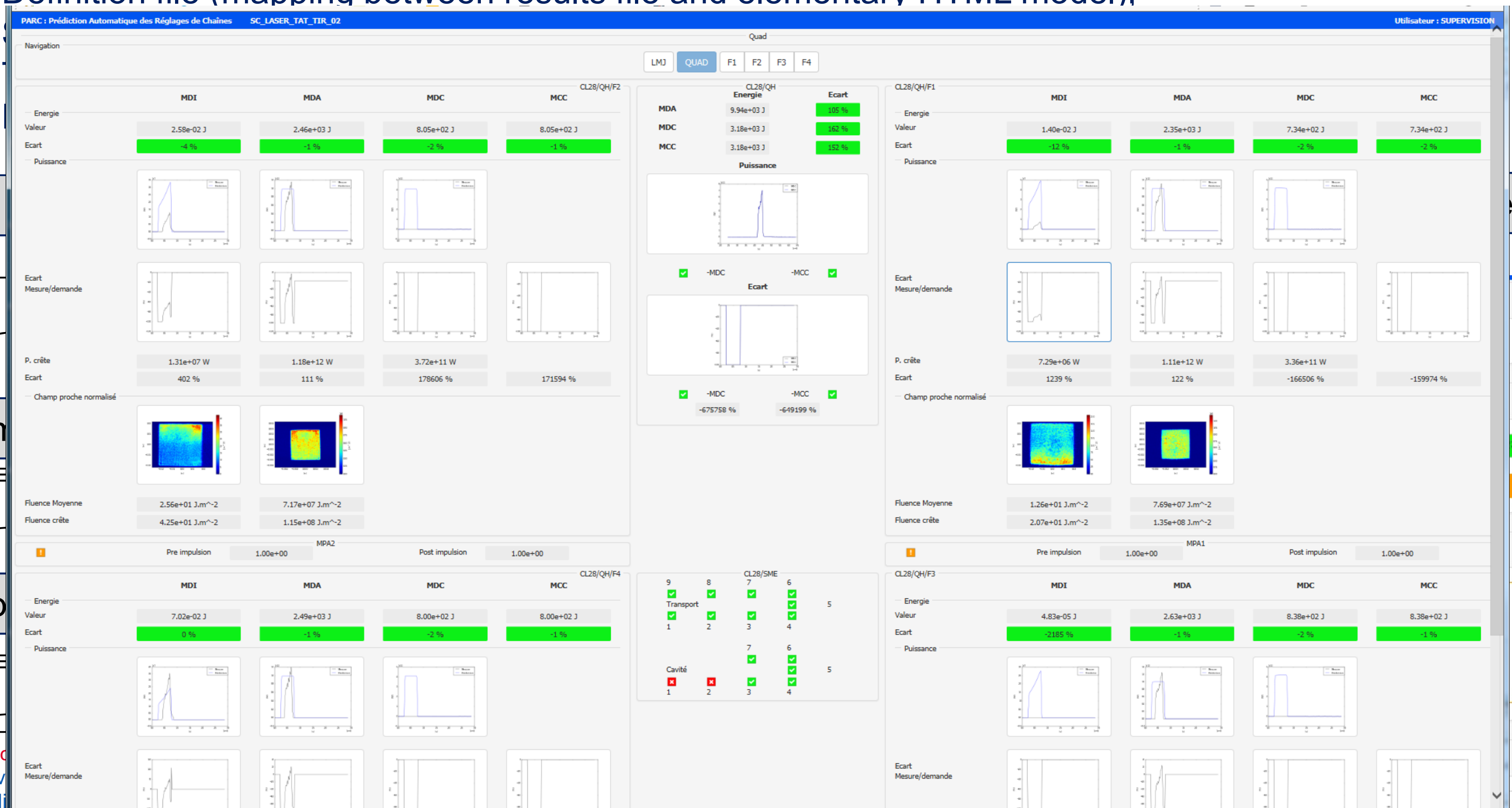
- Definition file (mapping between results file and elementary HTML model),
- Several HTML files (layout and browsing options (JS) of the generated page),
- Template python module file (code generation),
- Description file (name, langage, input, output, sub-module, etc.).



GUI MODULE

A GUI module describes the content of a HTML web page (parametrization or reporting).
It is composed of :

- Definition file (mapping between results file and elementary HTML model),



XML

Elem

Html

Layo

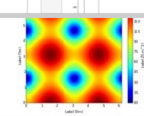
Html

b page

```
<div c
  <div
    <di
      <span class="span-
parc">{{ihm_elements[CLiQxFn_MDI.FLUENCE.Fl
uence|Image].valeur}}</span>
  </div>
  </div>
</div>
```

3457e+10

700.00 mJ

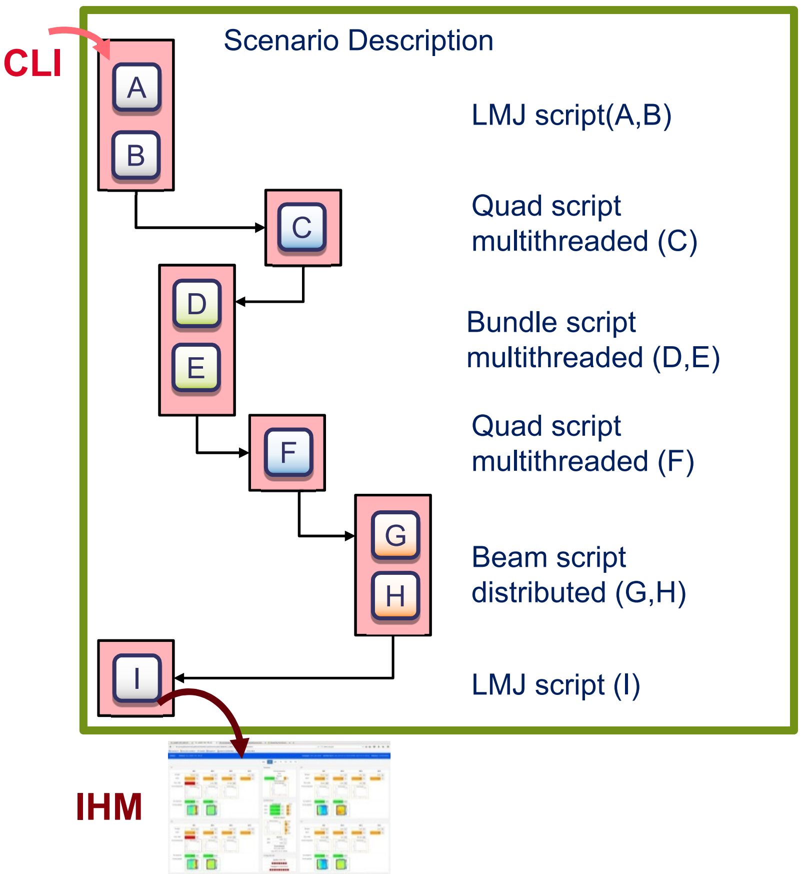


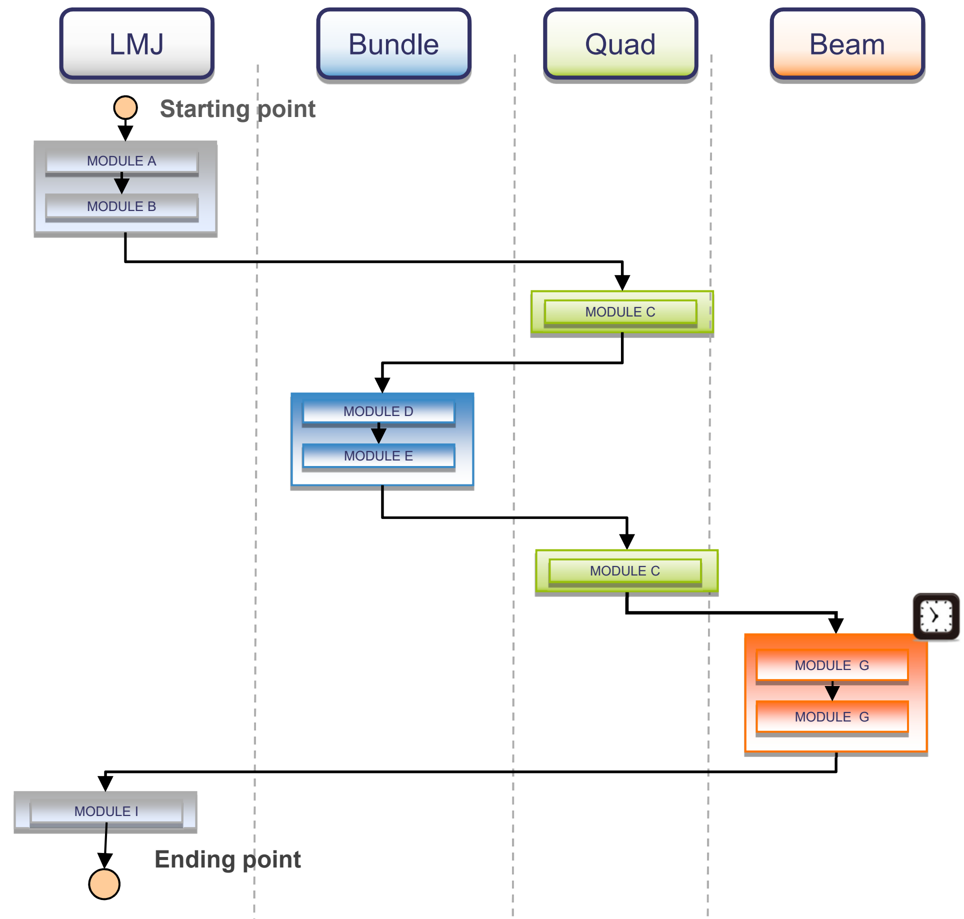
SCENARIO

Scenario is described as a simple xml file.

- List of modules and their granularity of execution.

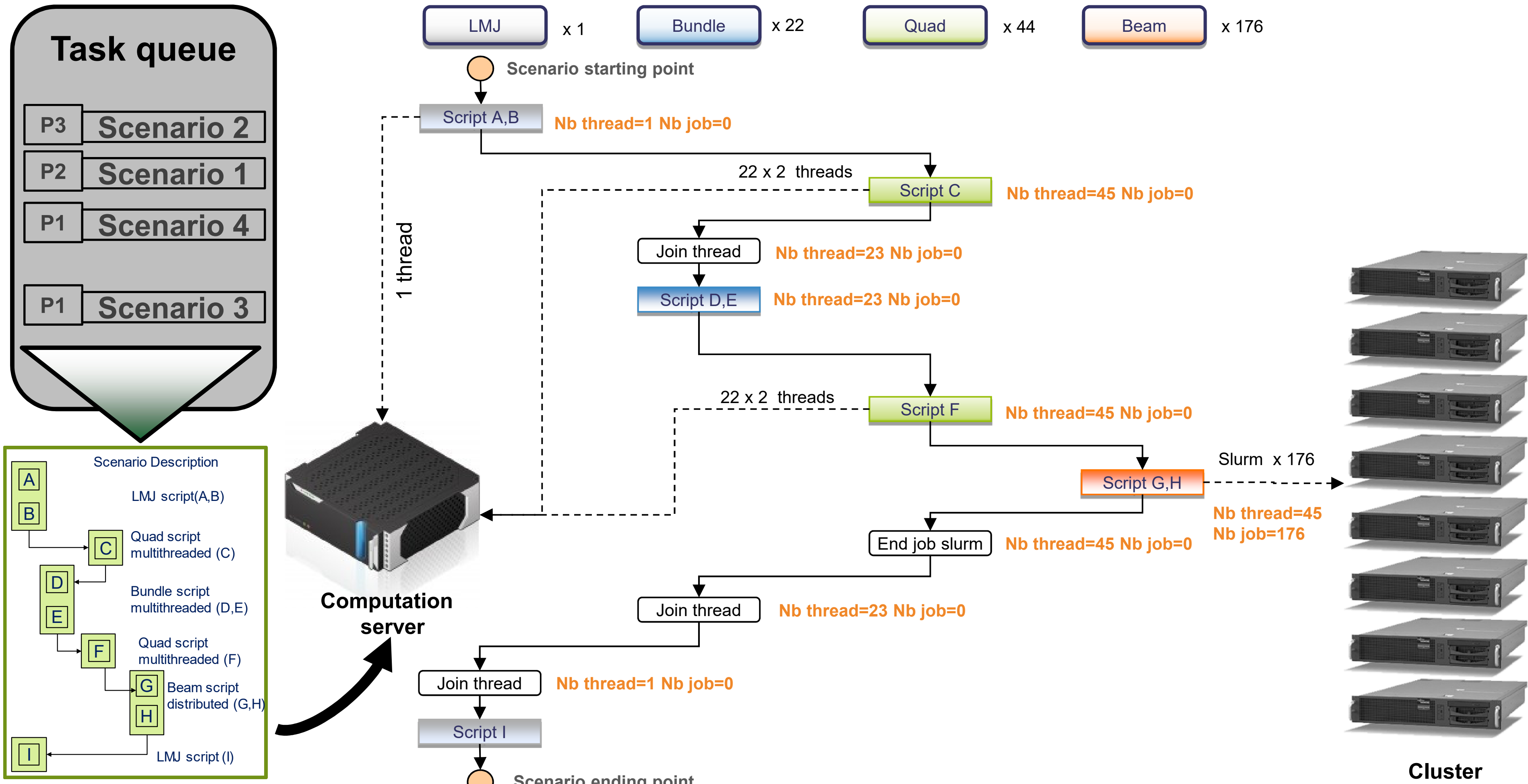
Modules of the same granularity are grouped in a script.
The Common Interface Langage (CLI) is generated by a pseudo-compiler.



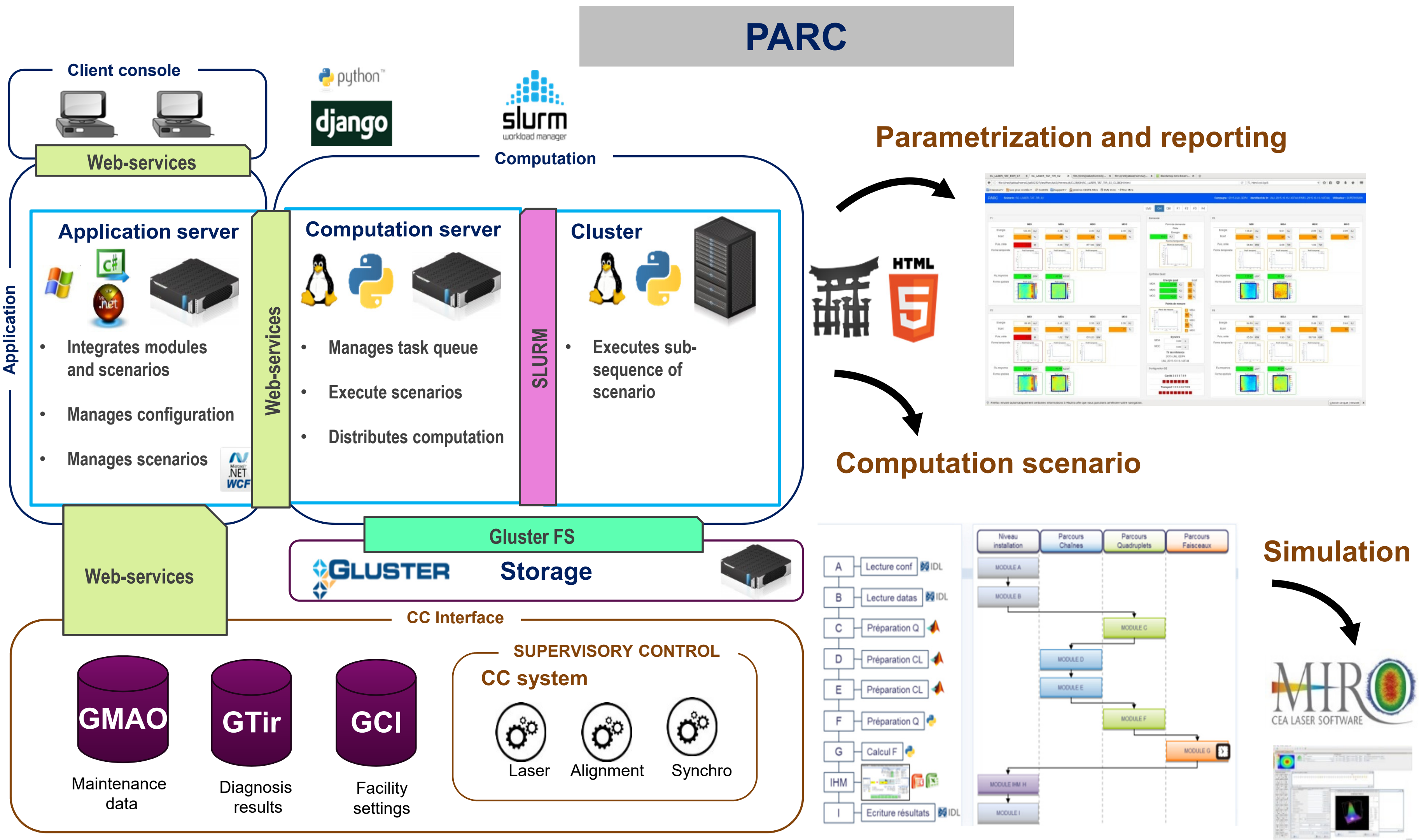


MULTITHREADING / DISTRIBUTION

- Computation server manages tasks depending on priority (Celery).
- Computation server runs jobs on cluster if necessary (SLURM).

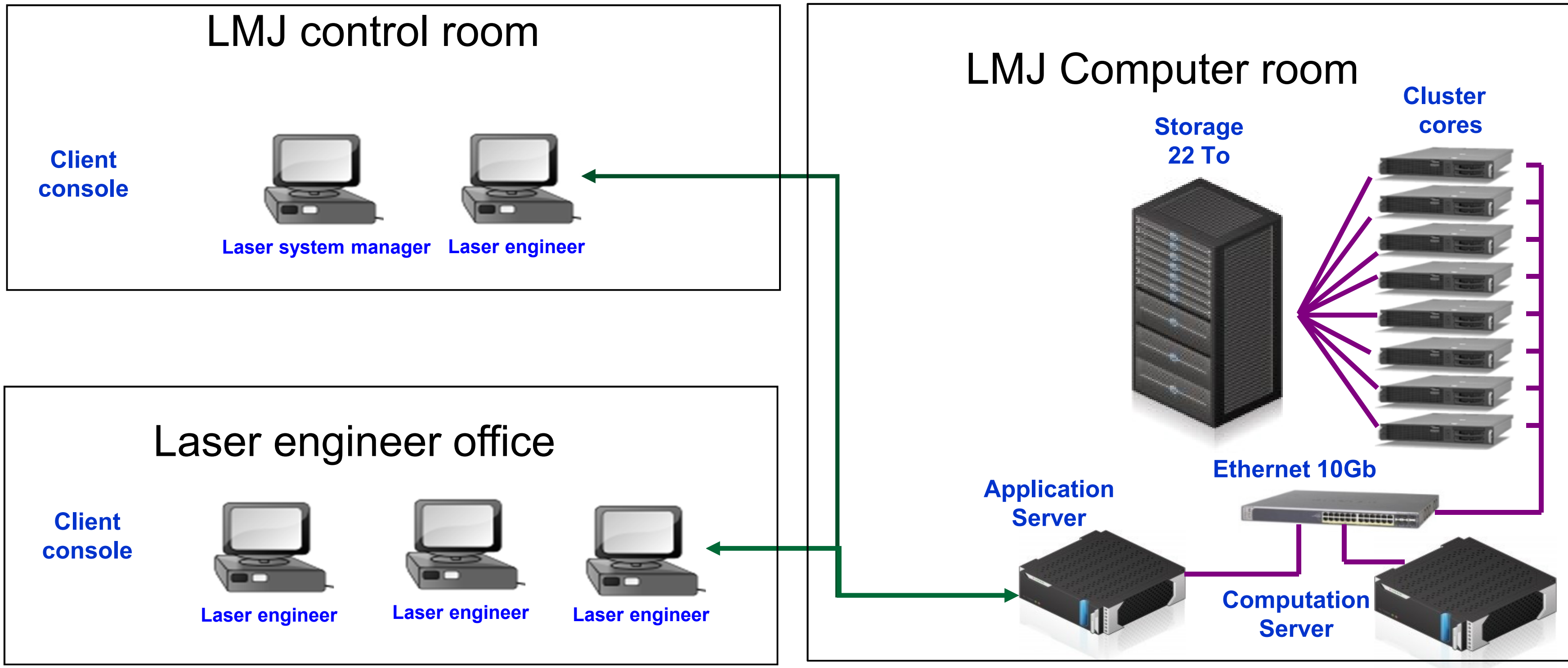


SOFTWARE CONFIGURATION

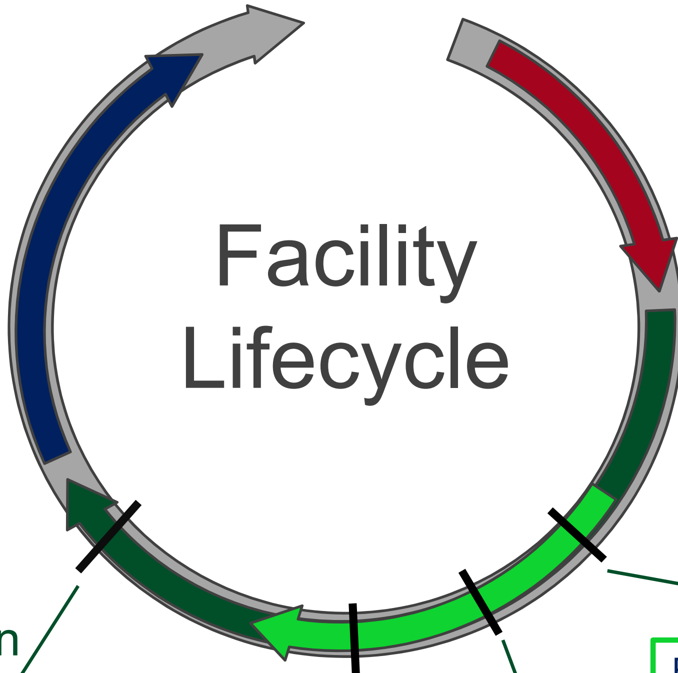


HARDWARE CONFIGURATION

- Different using profile depending on the role and the localisation
- Cluster metrics : 4 cores / beam (parallel computing) , 32 cores blade / bundle
- Distributed storage system (GlusterFS): 1To / bundle



PARC IN LMJ FACILITY LIFECYCLE



Facility maintenance

PARC bundle configuration update
- Uses setting database (GCI)
- Uses diagnosis results (GTIR)
- Uses logistical database (GMAO)
- Updates PARC configuration

New Bundle integration

PARC bundle configuration update
- Uses setting database (GCI)
- Uses Diagnosis results (GTIR)
- Updates PARC configuration

Experiment's campaign

PARC model calibration
- Uses shot results history
- Updates PARC configuration

PARC Prediction / validation
- Uses shot requirement database (GTIR)
- Uses setting database (GCI)
- Generates prediction and validation report
- Generates setting files (FdS)

PARC Power shot result processing
- Uses diagnosis results (GTIR)
- Generates result report
- Generates shot report

Campaign shot

PARC Rod shot result processing
- Uses diagnosis results (GTIR)
- Generates result report

CONCLUSION

PARC fulfills its role of adaptative computational platform in support of LMJ operations.

About 10 scenarii are currently used on the facility

- Essential to generate the input setting file for the sequence
- 2500 predictions and 3000 report in 2 years (10 users)

27 scenarii have been identified (3 developpers and contractors)

- 4 dedicated to shot sequence (new diagnosis)
- 23 dedicated to machine sequence (laser, alignment, synchronisation instrument calibration)

Many results are stored in PARC database, we are actually working on a new project to gather and index all the data needed to monitor the « health » of the facility. This system will provide a variety of tools to inspect these data (timeline, uncertainty, etc.) and elaborate cross analysis (data mining).

Commissariat à l’énergie atomique et aux énergies alternatives
CESTA | 33114 Le Barp
T +33 (0)5 57 04 00 00 | F +33 (0)5 57 04 00 00

Etablissement public à caractère industriel et commercial | RCS Paris B 775 685 019