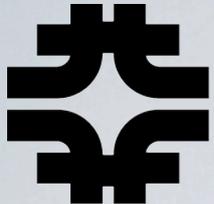


DEVELOPMENT OF A DATA ACQUISITION WEB SERVICE FOR ACNET CONTROL SYSTEM

PARTI student meeting
August 24, 2011

Alexey Moroz, MIPT
Supervisor: Andrey Petrov



ACNET CONTROL SYSTEM

Applications

Graphical (mainly Java-based)

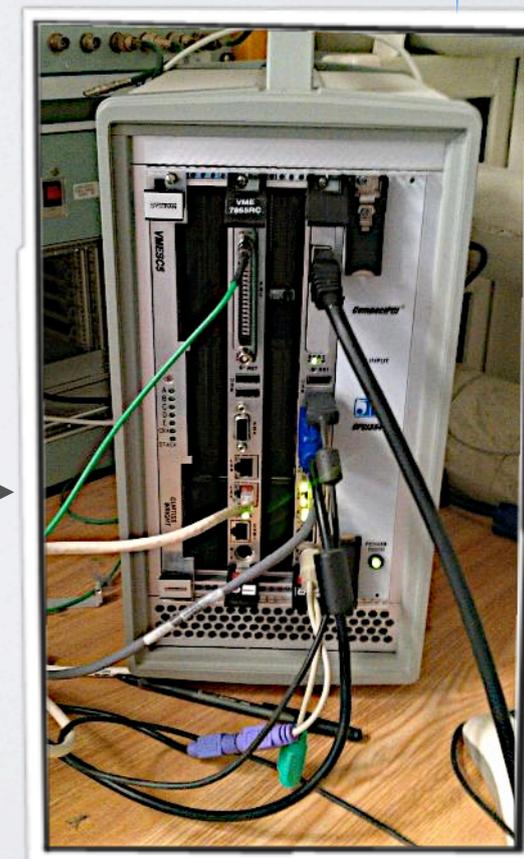
Console (old)

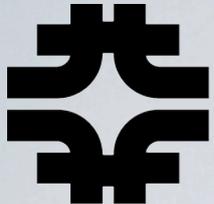
Middle Layer
(central services for data acquisition)

ACNET is a uniform control system at Fermilab

- Experiments use site-wide network of devices, each representing a single measurement point or a control item
- Middle layer servers collect data from frontends and provide it to application level
- Operators and engineers use applications for managing data and system control

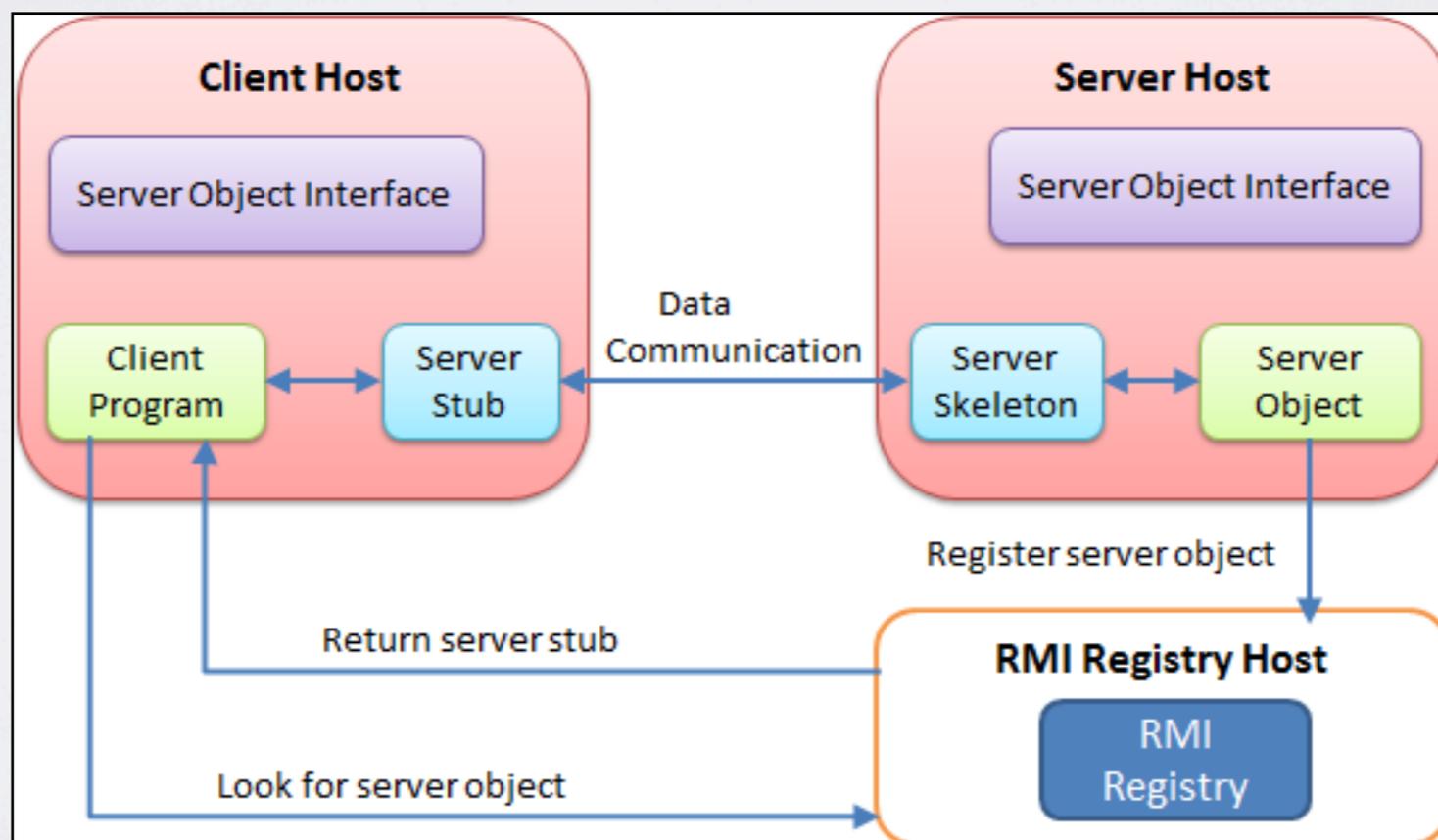
Data acquisition frontends (~200000 devices)





PROBLEMS

- ◆ ACNET was designed many years ago and was not supposed to provide data outside of the controls' network.
- ◆ Current data acquisition system uses RMI (remote method invocation) technology, which is too complex and heavy-weight for the system in its current state.
- ◆ There's a need in more simple way of reading data from the control system.

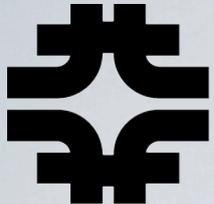


Scheme: Java RMI Overview

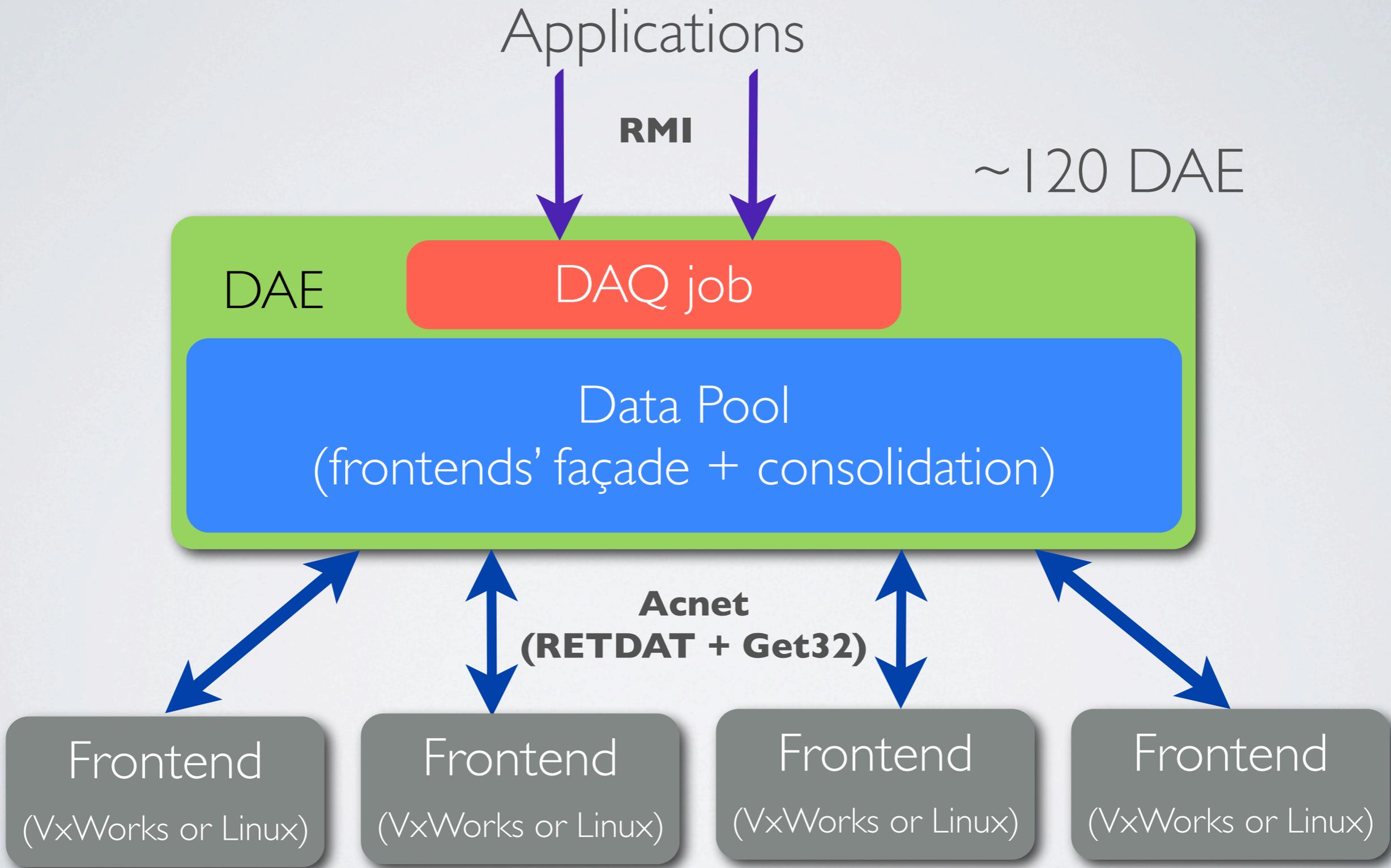


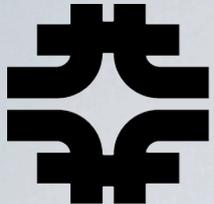
MAIN PROJECT REQUIREMENTS

- ◆ In order to make system more light-weight, universal and flexible decision was made to use an HTTP-based request-response text protocol.
- ◆ The data acquisition from the site-located devices remains RMI-based while outer world will obtain data by text-based protocol.
- ◆ HTTP responses are available in different formats (e. g. plain-text, XML, Javascript) allowing data to be easily accessible on all major software platforms.
- ◆ Hardware is resource-limited, while every request can query hundreds of devices, thus caching system was introduced.
- ◆ Since client could be both human and program, data queries may be encapsulated into easily readable RESTful URL.

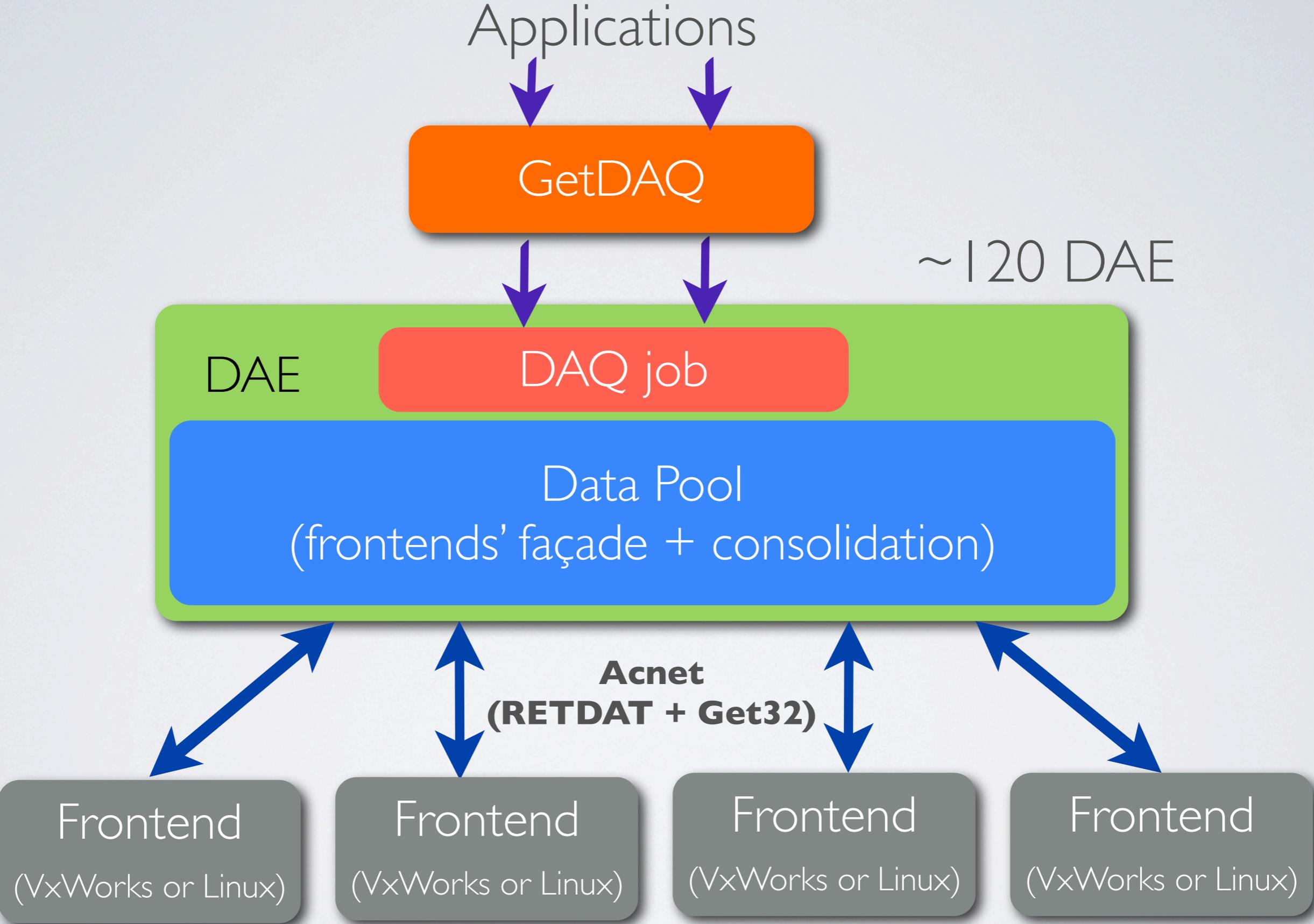


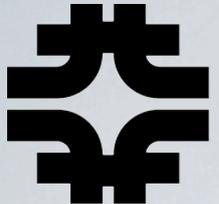
CURRENT CONTROLS INFRASTRUCTURE



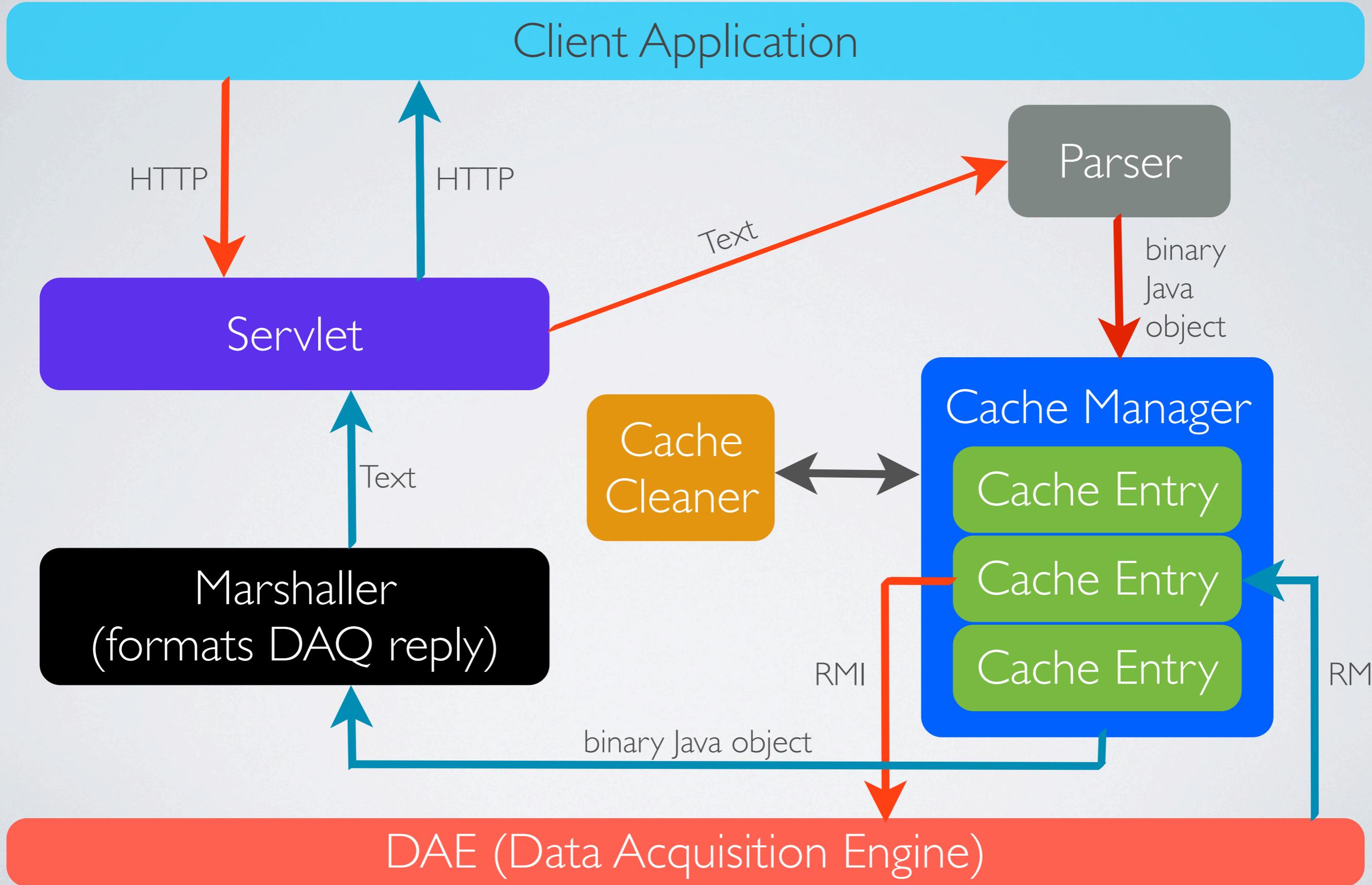


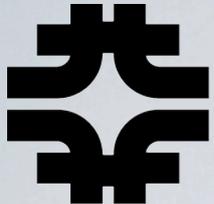
CURRENT CONTROLS INFRASTRUCTURE





GETDAQ APPLICATION ARCHITECTURE





CACHING SYSTEM OVERVIEW

Device

a separate measurement point or control item

Request containing devices not present in cache leads to creation of a new DAQ job, right after request data from newly created job is usually unavailable.

r:pelcdc.raw

g:tlgm04.reading[0:5]

m:outhum

m:outtmp

Cache Cleaner

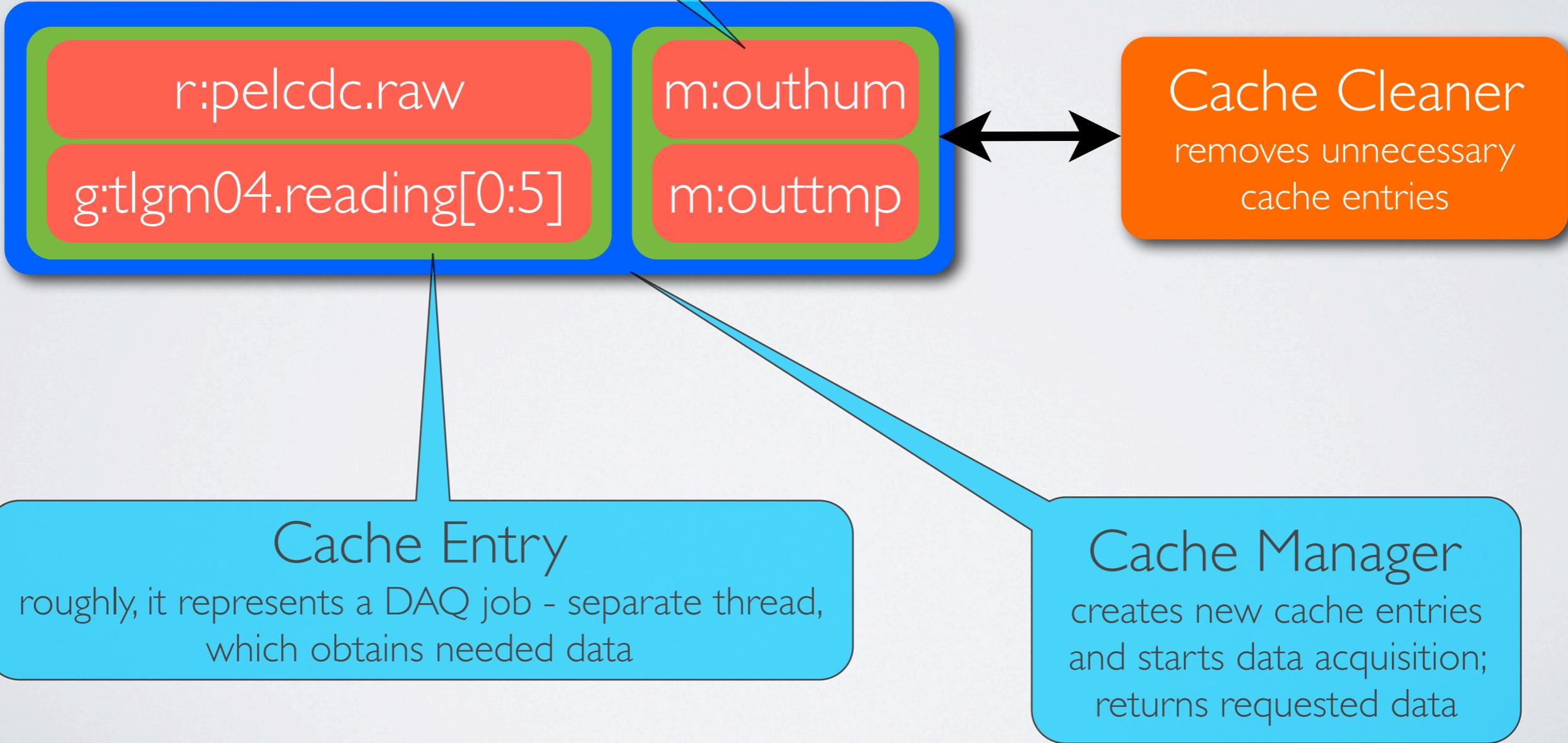
removes unnecessary cache entries

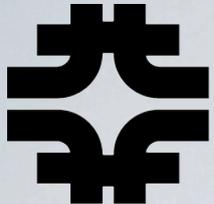
Cache Entry

roughly, it represents a DAQ job - separate thread, which obtains needed data

Cache Manager

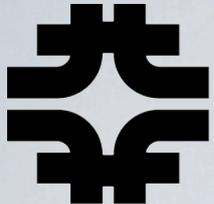
creates new cache entries and starts data acquisition; returns requested data





DETAILS

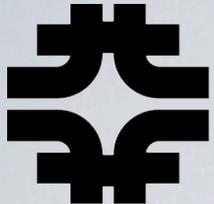
* Parameters could be tuned through application settings



DETAILS

Marshaller
can transform DAQ replies
to 4 different formats:
XML, HTML (human-
readable), plaintext CSV,
JSON (Javascript)

* Parameters could be tuned through application settings

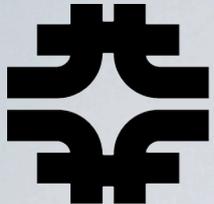


DETAILS

Cache Cleaner removes DAQ jobs which are not requested during 5 minutes*

Marshaller can transform DAQ replies to 4 different formats: XML, HTML (human-readable), plaintext CSV, JSON (Javascript)

* Parameters could be tuned through application settings



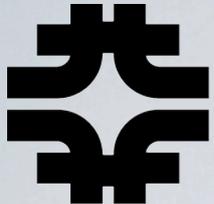
DETAILS

Cache Manager exists only in one instance and makes all connections to DAE in a thread-safe manner, hence simultaneous requests will never result in two same parallel DAQ jobs started. Maximum number of simultaneous DAQ jobs is 64*

Cache Cleaner removes DAQ jobs which are not requested during 5 minutes*

Marshaller can transform DAQ replies to 4 different formats: XML, HTML (human-readable), plaintext CSV, JSON (Javascript)

* Parameters could be tuned through application settings



DETAILS

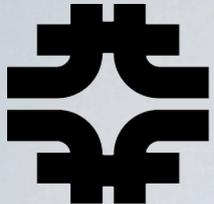
Cache Manager exists only in one instance and makes all connections to DAE in a thread-safe manner, hence simultaneous requests will never result in two same parallel DAQ jobs started. Maximum number of simultaneous DAQ jobs is 64*

Cache Entry holds an internal array of requested devices and updates data only when it becomes available. Every entry corresponds to one DAQ job. Maximum amount of requested devices can be as large as 128*

Cache Cleaner removes DAQ jobs which are not requested during 5 minutes*

Marshaller can transform DAQ replies to 4 different formats: XML, HTML (human-readable), plaintext CSV, JSON (Javascript)

* Parameters could be tuned through application settings



EXAMPLES

Mozilla Firefox

http://www-bd.fnal.gov/getdaq/job/m:outtmp;m:outhum

Time	Request	Value	Unit
1314192021018	m:outtmp	72.86760130286217	DegF
1314192021018	m:outhum	70.86463408736509	%RH

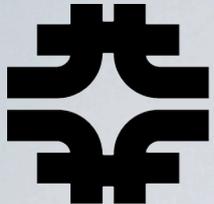
Done

Mozilla Firefox

http://www-bd.fnal.gov/getdaq/job/m:outtmp;m:outhum?type=xml&iso-time=true

```
- <data-set time="2011-08-24T08:38:22.288-0500">
- <reply format_hint="0" ref_id="m:outtmp" time="2011-08-24T08:38:21.210-0500" type="DoubleSample"
  unit="DegF">
  <request canonical="m:outtmp.READING" device="m:outtmp" event="U" field="SCALED"
  property="READING" range="[0]"/>
  <value type="double">74.12341182112692</value>
</reply>
+ <reply format_hint="0" ref_id="m:outhum" time="2011-08-24T08:38:21.210-0500" type="DoubleSample"
  unit="%RH"></reply>
</data-set>
```

Done



EXAMPLES

GetDAQ

Write your request here (newlines will be trimmed):

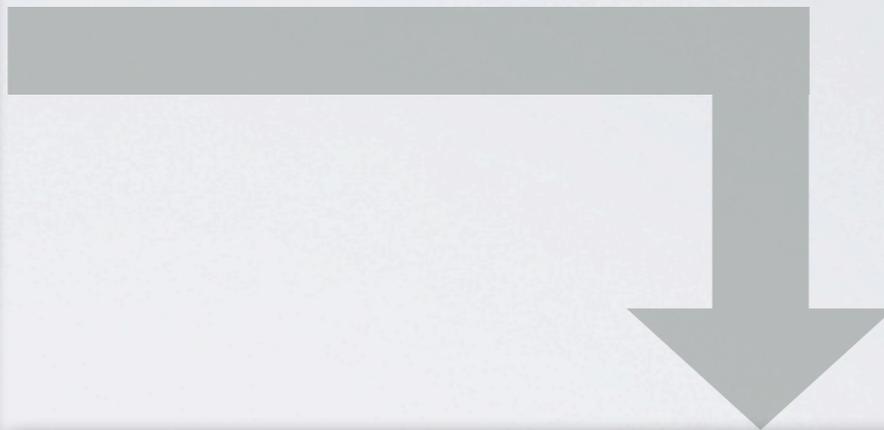
```
m:outtmp  
m:outhum  
r:pelcdc.raw;g:tlgm04.reading[0:5]
```

Brackets-Separated (otherwise semicolon-separated)
 ISO-time
 Quiet
 Include DRF2 request

Type: XML HTML PlainText JSON

Submit

Done



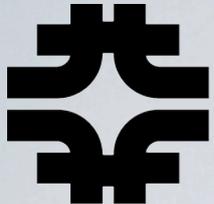
http://www-bd.fnal.gov - Mozilla Firefox

http://www-bd.f...gov/getdaq/job

Data-Set
Time: 2011-08-24T08:49:46.807-0500

Time	Request	Value	Unit
2011-08-24T08:49:46.350-0500	m:outtmp	74.68154982924463	DegF
2011-08-24T08:49:46.350-0500	m:outhum	67.49154262678167	%RH
2011-08-24T08:49:46.639-0500	r:pelcdc.raw	192	
2011-08-24T08:49:46.505-0500	g:tlgm04.reading[0:5]	4.0 3.0 9.0 26957.0 26990.0 28482.0	

Done



EXAMPLES

GetDAQ

Write your request here (newlines will be trimmed):

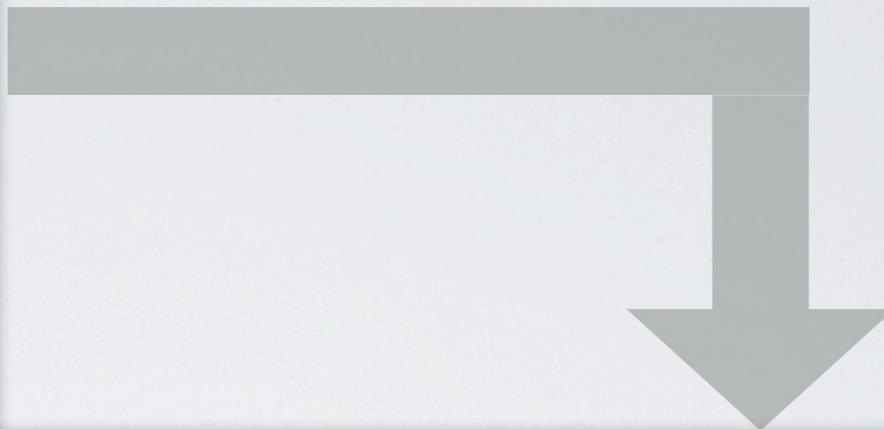
```
m:outtmp  
m:outhum  
r:pelcdc.raw;g:tlgm04.reading[0:5]
```

Brackets-Separated (otherwise semicolon-separated)
 ISO-time
 Quiet
 Include DRF2 request

Type: XML HTML PlainText JSON

Submit

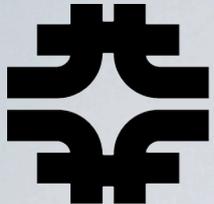
Done



http://www-bd.fnal.gov - Mozilla Firefox

```
unit="DegF"></reply>  
+ <reply ref_id="m:outhum" time="2011-08-24T08:51:12.381-0500" type="DoubleSample"  
  unit="%RH"></reply>  
+ <reply ref_id="r:pelcdc.raw" time="2011-08-24T08:51:12.669-0500"  
  type="IntegerSample"></reply>  
- <reply ref_id="g:tlgm04.reading[0:5]" time="2011-08-24T08:51:13.150-0500"  
  type="DoubleArraySample" unit="">  
  <request canonical="g:tlgm04.READING[0:5]" device="g:tlgm04" event="U"  
    field="SCALED" property="READING" range="[0:5]"/>  
  - <array size="6">  
    <value>4.0</value>  
    <value>3.0</value>  
    <value>9.0</value>  
    <value>26957.0</value>  
    <value>26990.0</value>  
    <value>28482.0</value>  
  </array>  
  </reply>  
</data-set>
```

Done



EXAMPLES

GetDAQ

GetDAQ +

Write your request here (newlines will be trimmed):

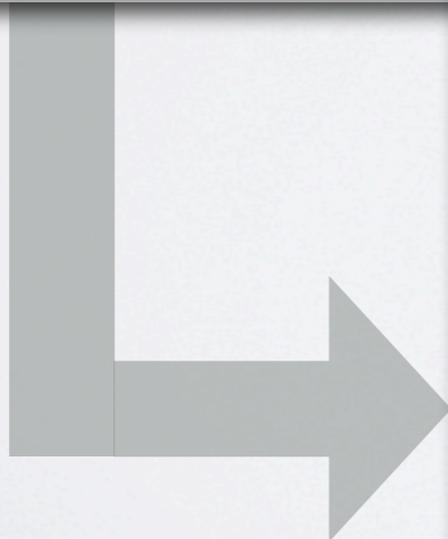
```
m:outtmp;  
m:outhum  
r:pelcdc  
g:tlgm04.reading[0:1]
```

Brackets-Separated (otherwise semicolon-separated)
 ISO-time
 Quiet
 Include DRF2 request

Type: XML HTML PlainText JSON

Submit

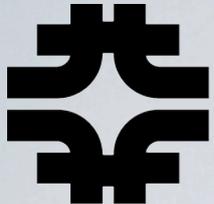
Done



job - /Users/amoroz/Downloads

```
1 {  
2   "data-set": {  
3     "time": "2011-08-24T08:55:04.196-0500",  
4     "reply": [  
5       {  
6         "unit": "DegF",  
7         "time": "2011-08-24T08:55:03.469-0500",  
8         "format_hint": 0,  
9         "request": {  
10          "field": "SCALED",  
11          "range": "[0]",  
12          "event": "U",  
13          "device": "m:outtmp",  
14          "property": "READING",  
15          "canonical": "m:outtmp.READING"  
16        },  
17        "value": {  
18          "content": 74.96061883330344,  
19          "type": "double"  
20        },  
21        "ref_id": "m:outtmp",  
22        "type": "DoubleSample"  
23      },  
24      {  
25        "unit": "%RH",  
26        "time": "2011-08-24T08:55:03.469-0500",  
27        "format_hint": 0,  
28        "request": {  
29          "field": "SCALED",  
30          "range": "[0]",  
31          "event": "U",  
32          "device": "m:outhum",  
33          "property": "READING",  
34          "canonical": "m:outhum.READING"  
35        },  
36        "value": {  
37          "content": 66.0123997009473,  
38          "type": "double"  
39        },  
40        "ref_id": "m:outhum",  
41        "type": "DoubleSample"  
42      }  
43    ]  
44  }  
45 }
```

Saved: August 24, 2011 8:55 AM · Length: 1,841 · Encoding: Unicode (UTF-8)



RESULTS

- ◆ A web service allowing one to read site-located devices from outside world is up and running on existing infrastructure.
- ◆ All code is integrated into current control system as well and is available in CVS.
- ◆ Data could be obtained through an HTTP protocol either by using a RESTful URL or by using a POST request for large sets of data.
- ◆ Data is available in XML, plaintext and JSON formats for major software platforms and in HTML form for human-readable representation.
- ◆ Caching system was introduced for mapping of DAQ jobs and to prevent excessive load on DAE servers.
- ◆ Users may use different parameters to choose the representation format of requested data.



QUESTIONS

Link: <http://www-bd.fnal.gov/getdaq>