Improvement of AMGA Python Client Library for the Belle II Experiment

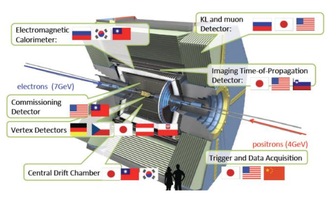
Jae-Hyuck Kwak, Geunchul Park, Taesang Huh, and Soonwook Hwang
jhkwak@kisti.re.kr
KISTI (Korea Institute of Science and Technology Information), Republic of Korea

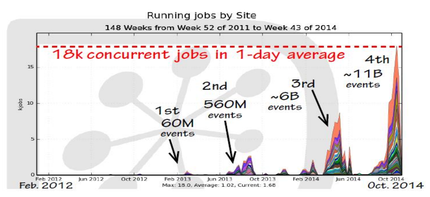
Motivation

- gBasf2 uses AMGA python client API to access AMGA
- AMGA python client API has a lack of functionalities, compared to C++ client API
  - No support for client-side metadata federation
- Developer from the Belle II distributed computing system asks for improvement of AMGA python client API
  - Client-side metadata federation support, GSI (python SSL wrapper for DIRAC) support, API refinement

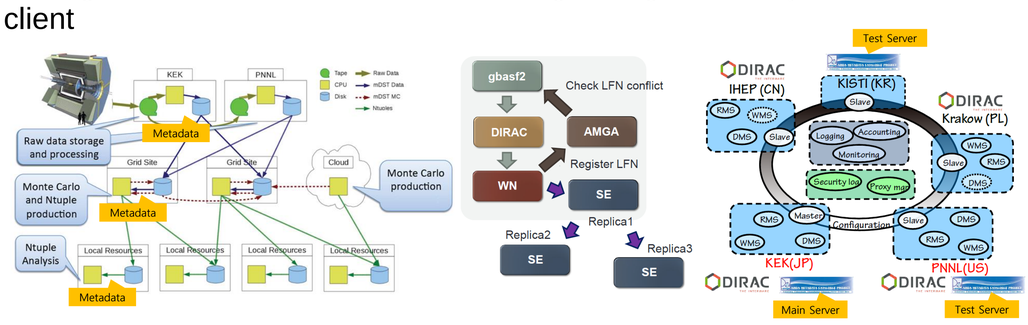
Overview of the Belle II Experiment

- An upgrade of the B factory experiment Belle at the KEK laboratory in Tsukuba, Japan, investigating CP-violation, which explains why the universe today consists only of matter and no anti-matter
  - SuperKEKB accelerator commissioning starts in early 2016
  - Phase 2 run (w/o the vertex detector) starts in 2017
  - Phase 3 run (w/ the full detector) starts in 2018
- Expected to produce ~100PB (one set of raw data) and another ~100PB (MC/analysis data)
  - 1.8GB/s @ Storage
  - 10s of millions of files distributed across multiple grid sites



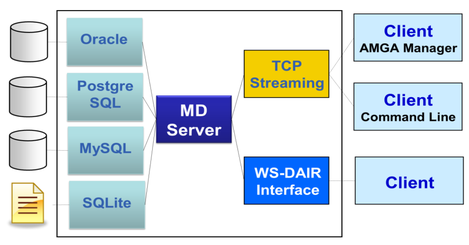


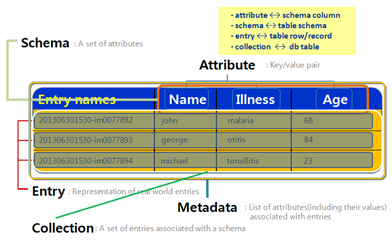
- Belle II Computing Model uses DIRAC for distributed workload management, AMGA for metadata catalog, and gBasf2 for job submission client



AMGA Overview

- Standalone Grid metadata catalog for supporting metadata description, discovery and archive of large-scale scientific data
  - Directory-like hierarchical structure
  - Various authentication methods (ID/password, VOMS certificate)
  - ACL-based authorization
  - Heterogeneous DB back-ends (Modular back-end: PostgreSQL, MySQL, Oracle, SQLite)
  - Standardized access methods (Modular front-end: TCP Streaming, SOAP WS-DAIR, SSL)
  - Multi-process/Multi-thread DB connection
  - Pre-existing DB import
  - Metadata replication and federation (experimental)
  - General-purpose AMGA Manager GUI
  - Various programming API (C++, Java, Python)





Summary and Future Plans

- Improvement of AMGA python client library, based on requirements from developer of Belle II distributed computing system
  - Added support for client-side metadata federation and GSI, along with API refinement
- Investigation of new action items based on the result of the recent MC campaign
  - Will continue to maintain AMGA python client library based on requirements

Support for Client-side Metadata Federation

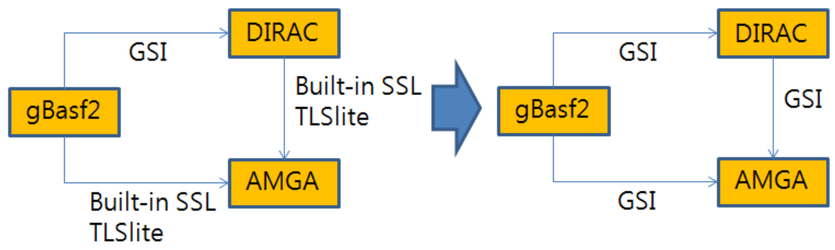
- Limitation of AMGA client-side metadata federation
  - Supported by C++ client API only
  - Creates entry of root priviledge only, not of user priviledge
  - Supports password-based authentication only
- Implemented client-side metadata federation on python client API
  - Introduces new module for federation (*mdfed.py*)
  - Modifies AMGA python client API (*mdclient.py*) to use new module, transparently to users
  - Preserves ownership of the entry created by non-root user
  - Supports certificate-based authentication

```
Query> whoami
>> belle2
Query> fed_umount /dir2
Query> fed_mount COMIC /dir2
Query> addentry /dir2/5 status 'OK'
Query> stat /dir2/5
>> /dir2/5
>> entry
>> rw-
>> r--
>> root
>>
>>
>>
Query>
```

```
Query> whoami
>> belle2
Query> fed_umount /dir2
Query> fed_mount COMIC /dir2
Query> addentry /dir2/5 status 'OK'
Query> stat /dir2/5
>> /dir2/5
>> entry
>> rw-
>> r--
>> belle2
>>
>>
>>
Query>
```

Support for GSI (python SSL wrapper for DIRAC)

- Use of different SSL libraries between DIRAC and AMGA
  - AMGA supports built-in SSL and TLSlite, while DIRAC supports GSI
- Added GSI support to AMGA python client API
  - Introduces *USE_GSI* flag to turn it on
  - Could streamline SSL communication scheme for the Belle II distributed computing system
  - Could observe increased stability of SSL connection



Refinement of python client API

- Separate API calls between execution and fetch of AMGA command
  - Prone to misuse AMGA python client API from gBasf2
- Refined AMGA python client API to hide separation between execution and fetch to user
  - Synchronizes execution and fetch of AMGA command by introducing *fetch* flag to the related python client API
  - Modified API: listEntries, selectAttr, getAttr
  - New API: selectQuery
  - Simplifies use of AMGA python client API

listEntries:

```
client.listEntries('/gilda/movies')
while not client.eot():
   file, type=client.getEntry()
   print "->", file, type[0]
```

```
entries = client.listEntries ('/gilda/movies', 1)
print entries
['/gilda/movies/1', '/gilda/movies/2']
```

selectQuery:

```
values = client.selectQuery('/gilda/movies', ['FILE','title','LFN'], 'FILE>₩'1₩'')
print values
[['2', 'Batman', 'file://batman']]
```

getattr:

```
client.getattr('/gilda/movies', ['title', 'LFN'])
while not client.eot():
  file, values=client.getEntry()
  print "->",file, values
```

```
attries = client.getattr('/gilda/movies', ['title', 'LFN'], 1)
print attries
{'1': ['Spiderman', 'file://spiderman'], '2':['Batman', 'file://batman']}
```

selectAttr:

```
client.selectAttr(['/gilda/movies:title', '/gilda/movies:LFN'], '')
while not client.eot():
  values=client.getSelectAttrEntry()
  print  "selcted ->", values
```

```
values = client.selectAttr(['/gilda/movies:title', '/gilda/movies:LFN'], '', 1)
print values
[['Spiderman', 'file://spiderman'], ['Batman', 'file://batman']]
```