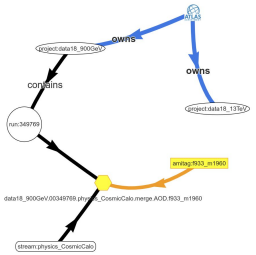
An Old/New

Universal & Native Client



A Unique (and Native) client to all implementations

(= extension of the current client to new implementations)

```
$ el -details dataset -p all -d AOD -p r9264_p3083 -e '284154 43859839'
5858E98E-97B4-7846-BF7E-CA8A2D941164 data15_13TeV.00284154.physics_Main.merge.AOD.r9264_p3083
C4CFE0EA-F37E-E511-85FD-44A8420A7771 data15_13TeV.00284154.physics_Main.merge.AOD.r9264_p3083

$ el -details dataset -p all -d AOD -p r9264_p3083 -e '284154 43859839' -api phoenix
D82968A1-CF91-4320-B2DD-E0F739CBC7E6 data15_13TeV.284154.physics_Main.merge.AOD.r9264_p3083

$ el -details dataset -p all -d AOD -p r9264_p3083 -e '284154 43859839' -api graph
*** available soon ***
```

- Universal EL Client
- Phoenix Client
- Graph Client
- Advantages
- Problems
- Plan
- Native Client
- Usage Possibilities

Of course, the arguments have been chosen so that the example works fine :-)

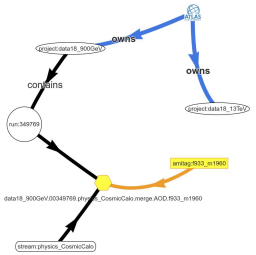
In reality, arguments have been only partially implemented and databases are not (yet) equivalent.

But the framework is deployed and works with already installed clients (try that example). The difficult part is done.

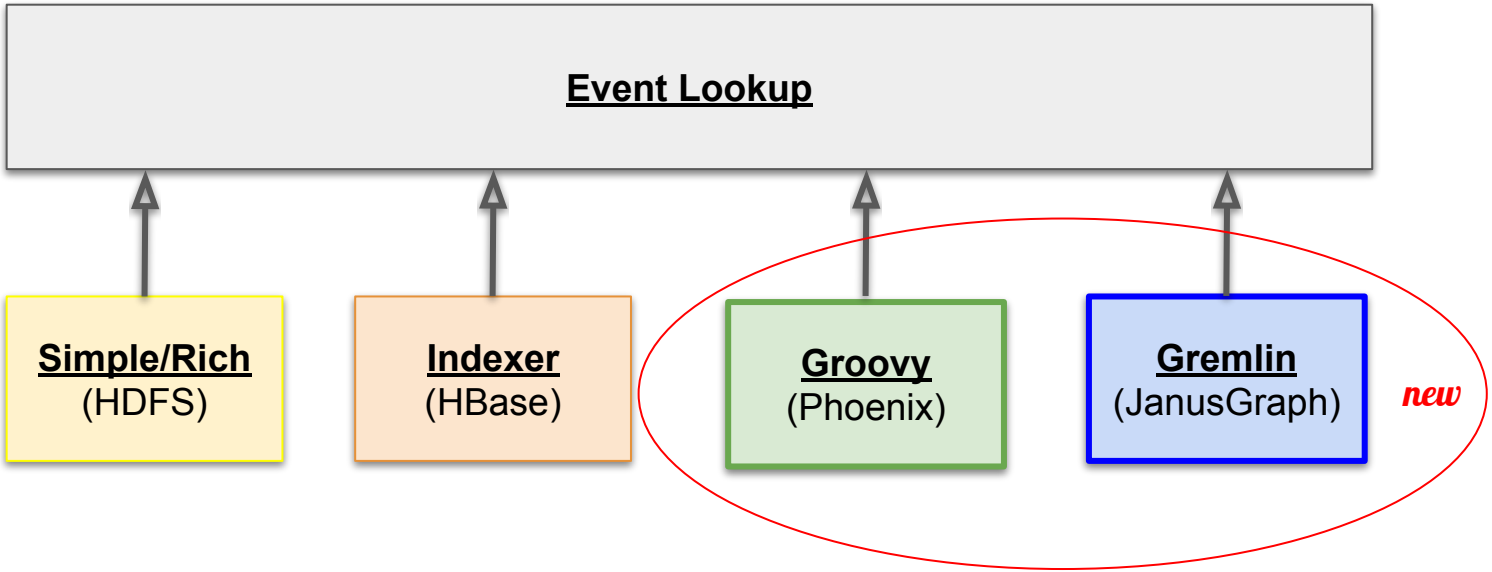
Julius Hrivnac, IJCLab

EI, 10/January/2022

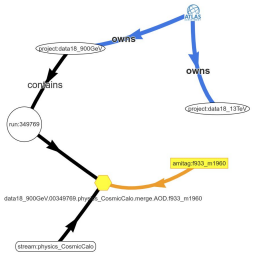
Universal EL Client



- EL client now understands **-api phoenix** and **-api graph**
  - It executes the search in the Phoenix or Janus implementation
  - EL client is unchanged, everything is implemented on the server



Phoenix Client



- Implemented as a **Groovy** script called from Java implementation
- Doesn't (yet) handle all el arguments
  - Will be implemented soon
- Groovy script is compiled as any Java code
  - But can be also run directly as a script => easy development/debugging/reuse

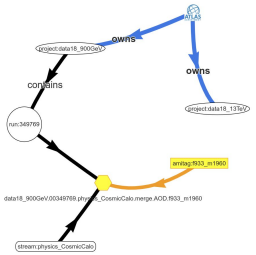
el.groovy

```
import groovy.sql.Sql

def el(runno, eventno) {
  sql = Sql.newInstance("jdbc:phoenix:ithdp2101.cern.ch:2181",
                        "org.apache.phoenix.jdbc.PhoenixDriver")
  result = ""
  sql.eachRow("select * from AEIDEV.DATASETS_0 where runno = " + runno) {dataset ->
    sql.eachRow("select * from AEIDEV.EVENTS_0 where eventno = " + eventno + " and dspid = " + dataset.dspid + " and dstypeid = " + dataset.dstypeid) {event ->
      result += [dataset.project, dataset.runno, dataset.streamname, dataset.prodstep, dataset.datatype, dataset.version].join(".") + "\n"
      }
    }
  return result
  }

result = el("$runno", "$eventno")
```

```
String processingCode = new StringResource(EL_GROOVY).toString();
Binding sharedData = new Binding();
GroovyShell shell = new GroovyShell(sharedData);
sharedData.setProperty("runno", runno);
sharedData.setProperty("eventno", eventno);
shell.evaluate(processingCode);
String result = sharedData.getProperty("result");
```

Gremlin
$G = (V, E)$

Graph Client

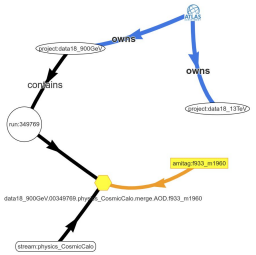


- Implemented as a **Gremlin** script called from Java implementation
  - Gremlin = Groovy with Graph functionality (functional syntax used for navigational semantics)

el.gremlin

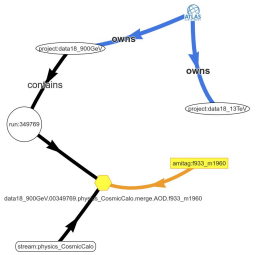
```
def el(g, runno, eventno) {
  return g.V().has('lbl', 'event').
               has('eventno', eventno).
               in('has').
               has('runno', runno).
               dedup()
 }
```

Advantages



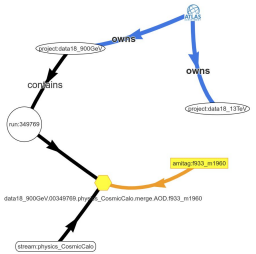
- Smooth transition
  - All users keep the same client app (already @CVMFS), just have to change **`-api`** argument
  - API will be made more conform to Unix standards later
- Easy to compare and verify old/new implementations
- Profit from existing infrastructure:
  - Interactive Web Service
  - Standalone client (standalone Java executable, doesn't need anything else)
  - REST Web Service (curl/wget) conform to CLI
  - Standard SSO protection
  - Possibility to authenticate using AFS kerberos ticket or Grid certificate (implemented, but currently not deployed)
  - Journaling, failure-to-mail bridge,...
- Actual executing code (for Phoenix and Janus) well encapsulated
  - As Groovy/Gremlin scripts, which can be also run independently on the framework
    - => easy development, debugging
    - => possibility to easy inclusion in other frameworks
- Futureproof: other implementations and components can be included

Problems



- All solved (so far)
- Configuration:
  - Running at the same time on lxhadoop and analytics
    - So far using some ugly hacks and the help of Emil
    - lxhadoop will disappear anyway
- Running in the same application Hadoop, HBase, Phoenix and Janus (at least clients):
  - Conflicting configuration, many third-party packages coming in different/conflicting versions

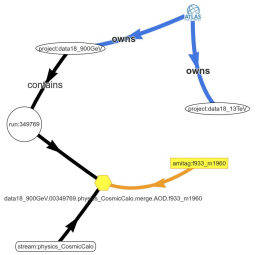
Plan



- Soon:
  - Implement all el arguments
    - It would be useful to have modular functions to decode Phoenix binary fields
  - Optimize
  - Make conform to Unix standards
- Later:
  - Implement other commands (ei, catalog, ti, …)
- After successful migration:
  - Eliminate obsolete code (i.e. HDFS/HBase implementation - most of the current **Core** code), cleanup

GraalVM™

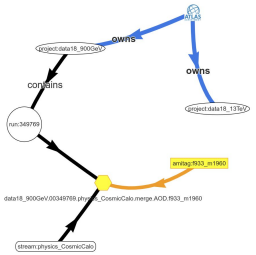
Native Client



- Client has been compiled into Linux native executable
  - Using *GraalVM* framework
  - Can be done for Mac, MS too
- Advantages:
  - Faster startup (important in small application)
  - No need for JVM on client machine
    - Except when you use Log4J
  - Directly callable from C/C++ code (when compiled as a sharedlib)
- Any client can be processed in the same way
  - We can also compile `el.groovy`, `el.gremlin` into native executables (or sharedlibs)

```
$ java -jar ../lib/EIHadoopEL.exe.jar -e ‘284154 43859839’
5858E98E-97B4-7846-BF7E-CA8A2D941164 StreamAOD
BA414650-E54C-214D-8DC7-89CD6815E628 StreamDAOD_TOPQ1
C4CFE0EA-F37E-E511-85FD-44A8420A7771 StreamRAW
# build native client (needs GraalVM and GCC)
$ ../src/sh/ni.sh
# call it
$ ../bin/EIHadoopEL.exe  -e '284154 43859839'
5858E98E-97B4-7846-BF7E-CA8A2D941164 StreamAOD
BA414650-E54C-214D-8DC7-89CD6815E628 StreamDAOD_TOPQ1
C4CFE0EA-F37E-E511-85FD-44A8420A7771 StreamRAW
```

Usage Possibilities



client	as script	in JVM compiled to JAR	as native compiled to exe/so
Universal CLI			
Groovy	the core functionality used by all others		
Graphical WS			
REST WS			

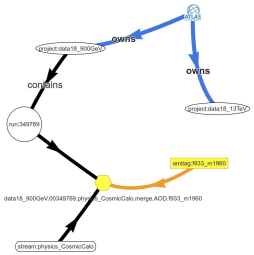
simpler

more services

faster

more generic

Links



- **Home page:**
  - https://atlas-event-index.cern.ch/doc
  - https://cern.ch/hrivnac/Activities/Packages/Lomikel
  - https://cern.ch/hrivnac/Activities/Packages/Atlascope
- **Code:**
  - https://gitlab.cern.ch/atlas-event-index/Atlas-Event-Index-Core
  - https://github.com/hrivnac/Lomikel
  - https://github.com/hrivnac/Atlascope
- **Using:**
  - https://groovy-lang.org
  - https://tinkerpop.apache.org/gremlin.html
  - https://www.graalvm.org

The only problem is the configuration, all the rest is almost trivial.