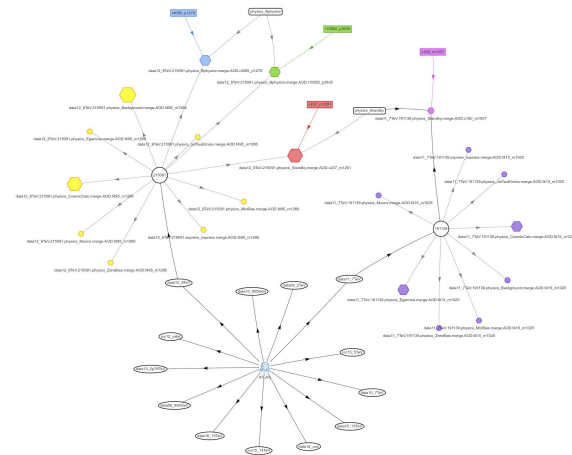




Atlascope



- Architecture
- Status
- Schema & Indexes
- Clients & Examples
- Web Client

ATLAScope 01.02.00+ [24/Apr/2021 at 10:12:16 CEST by atlewind for CERN] Reset

http://localhost:8182 add

Search: ATLAS Execute g.V().has('lbl', 'canonical')

dataset: DAOD_HIGG2D1 Actions:

Graph Image Plot

Customize the interactions with the graph.

☐ Cluster by group type ☐ Cluster by group size ☐ Expand all clusters ☐ Show all edges ☐ Hierarchical

☒ clusterize ☒ zoom cluster ☐ stabilize ☒ get children ☒ get parents ☐ remove old

filter: Apply select: limit(10)

canonical:AOD
10221559 events

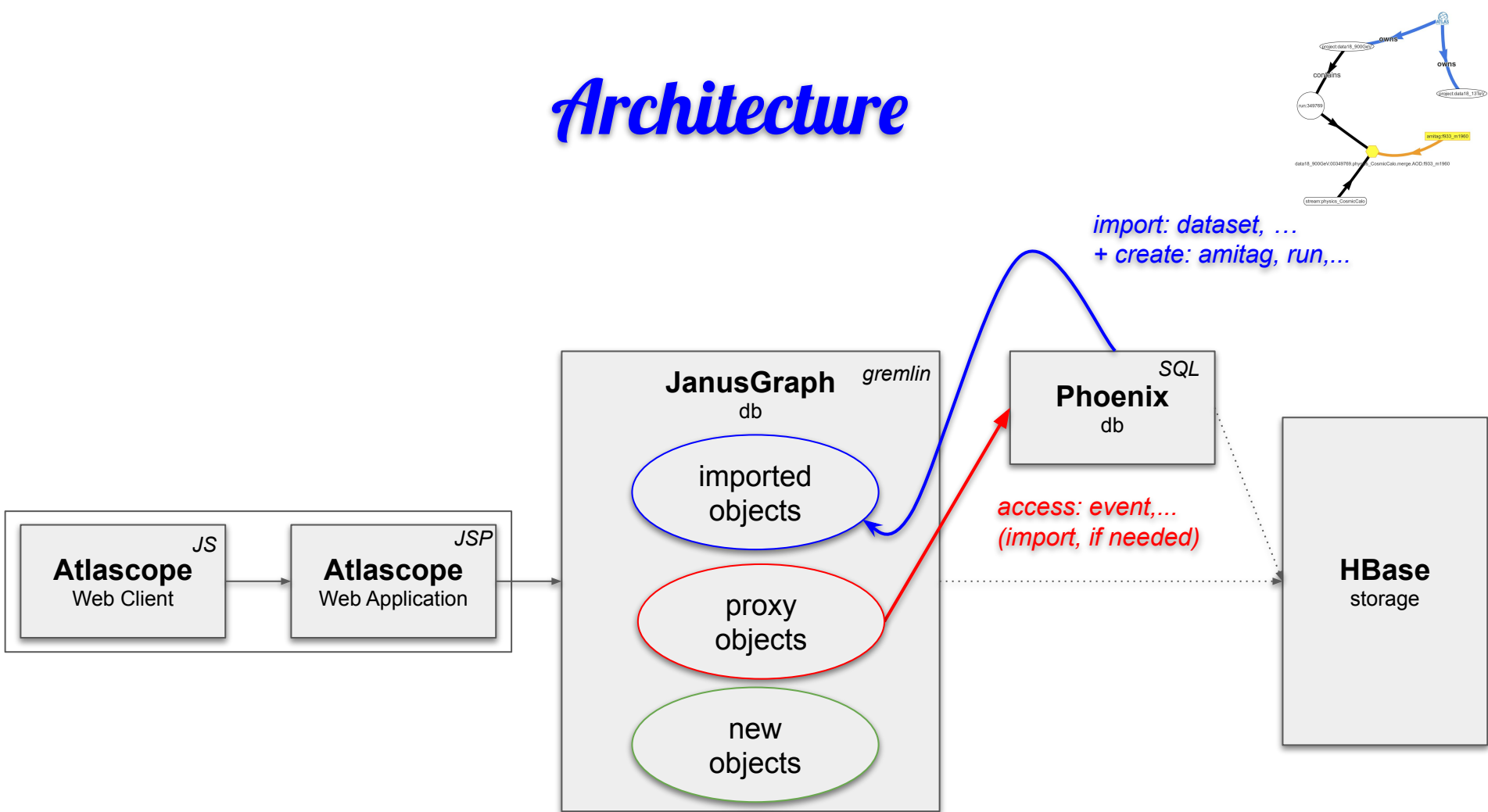
version:f832_m1812
runno:326870
project:data17_13TeV
streamname:physics_Main
prodstep:merge
datatype:AOD
dspsid:34930176
dstypeid:8192
smk:2573
events_rucio:10221559
rucio_at:Thu Nov 16 12:34:18 CET 2017
files:742
events:10221559
updated_at:Tue Mar 30 09:29:07 CEST 2021
is open:false
is deleted:false
status:IMPORTED
has_raw:true
has_trigger:true
prov_seen:2048
tbl:canonical
phenix:true
fullfill:true

close

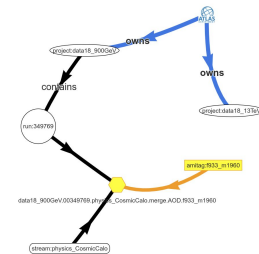
Using Demo data
from
Small IFIC sample
@CERN

Julius Hrivnac, IJCLab
EI WS, 4/Oct/2021

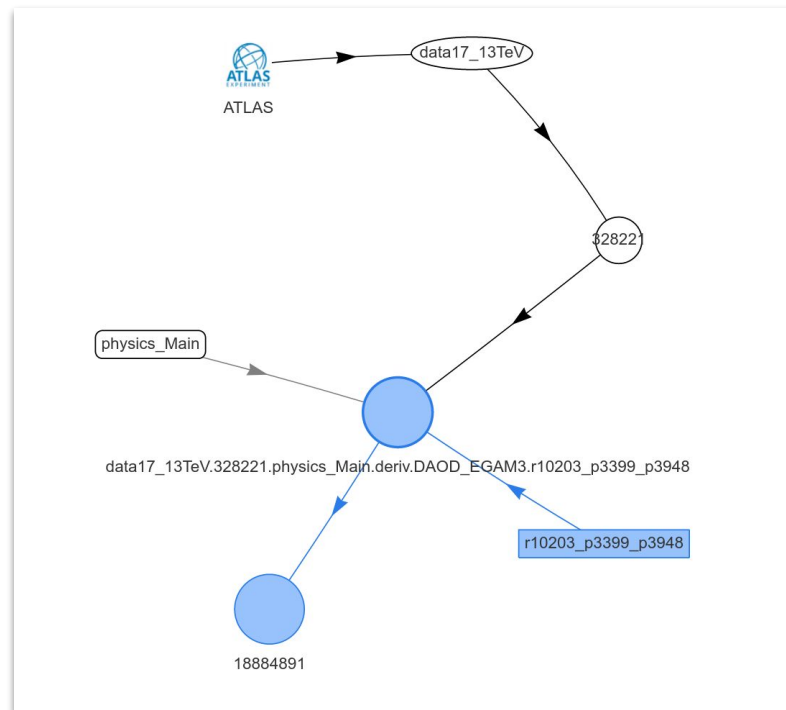
Architecture



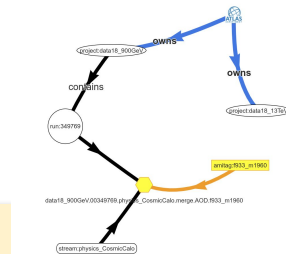
Status



- Using IFIC demo database @CERN (AEIDEV.JSZ3_*)
 - Importing @ 50Hz
 - Mostly due to Phoenix overhead and connection over Socket
 - Direct importing from files gives about 1kHz
 - Will try new Spark Scala package
 - Importing **datasets** and **canonical**
 - While importing, creating full graph structure
 - So enabling searching by navigation
- A lot of Phoenix table fields are in fact relations
 - They are replaced by Edges in Graph



Schema & Indexes



- Graph query is executed in three steps:
 - Initial search
 - Uses underlying database technology (HBase)
 - Can profit from Indexes (HBase + Elastic Search)
 - Navigation (very fast)
 - Accumulation of properties
- Schema & Indexes created to support the Initial search
 - Graph can still contain Vertexes, Edges and properties not covered by Schema

```
// Schema
canonical = mgmt.makeVertexLabel('canonical').make()
dataset   = mgmt.makeVertexLabel('dataset').make()
stream    = mgmt.makeVertexLabel('stream').make()

...
tags      = mgmt.makeEdgeLabel('tags').multiplicity(ONE2MANY).make()
fills     = mgmt.makeEdgeLabel('fills').multiplicity(ONE2MANY).make()

...
events_rucio = mgmt.makePropertyKey('events_rucio').dataType(Long.class).cardinality(Cardinality.SINGLE).make()
rucio_at     = mgmt.makePropertyKey('rucio_at').dataType(Date.class).cardinality(Cardinality.SINGLE).make()
files        = mgmt.makePropertyKey('files').dataType(Integer.class).cardinality(Cardinality.SINGLE).make()

...
mgmt.addProperties(canonical,
    lbl,
    importDate,
    phoenix,
    fullfill,
    runno,
    streamname,
    prodstep,
    datatype,
    version,
    dspid,
    dstypeid,
    smk,

    ...
    mgmt.addConnection(has, stream, canonical)
    mgmt.addConnection(tags, amitag, canonical)
    mgmt.addConnection(fills, run, canonical)
    mgmt.addConnection(gets, project, run)
    mgmt.addConnection(owns, ATLAS, project)
    mgmt.addConnection(contains, canonical, dataset)
    ...

// Indexes
mgmt.buildIndex('byVersionES', Vertex.class).addKey(lbl).
    addKey(version, Mapping.TEXTSTRING.asParameter()).
    buildMixedIndex('search')

mgmt.buildIndex('byRunnoES', Vertex.class).addKey(lbl).
    addKey(runno).
    buildMixedIndex('search')
```

(C) Python Client

```
#pip install gremlinpython
```

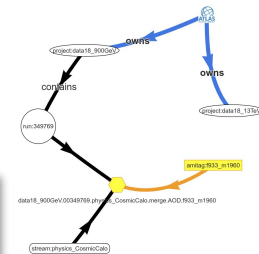
```
from gremlin_python import statics
from gremlin_python.process.anonymous_traversal import traversal
from gremlin_python.process.graph_traversal import __
from gremlin_python.process.strategies import *
from gremlin_python.driver.driver_remote_connection import DriverRemoteConnection
from gremlin_python.process.traversal import T
from gremlin_python.process.traversal import Order
from gremlin_python.process.traversal import Cardinality
from gremlin_python.process.traversal import Column
from gremlin_python.process.traversal import Direction
from gremlin_python.process.traversal import Operator
from gremlin_python.process.traversal import P
from gremlin_python.process.traversal import Pop
from gremlin_python.process.traversal import Scope
from gremlin_python.process.traversal import Barrier
from gremlin_python.process.traversal import Bindings
from gremlin_python.process.traversal import WithOptions
```

```
statics.load_statics(globals())
```

```
g = traversal().withRemote(DriverRemoteConnection('ws://aiatlas073.cern.ch:8182/gremlin','g'))
```

```
x = g.V().has('lbl', 'dataset').has(...).valueMap().next()
```

*Easy integration in
Atlas Framework*

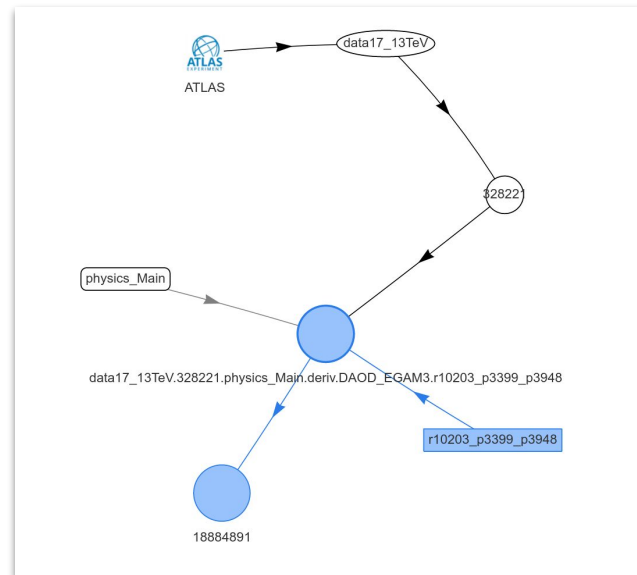
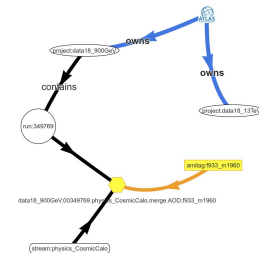


Clients exist in most languages

Easy integration in Any Framework

Event Lookup with Graphs

```
// only talks to JanusGraph => very fast
// (pure Gremlin)
dataset = g.V().has('lbl', 'ATLAS')
               .out().has('name', 'data17_13TeV')
               .out().has('runno', 328374)
               .out().has('prodstep', 'merge')
               .has('datatype', 'AOD')
               .out()
               .next();
// may talks also to Phoenix
// (special method, wrapping Phoenix object as a Graph Vertex)
events = Event.getOrCreate(dataset, g, 22222, true)
```



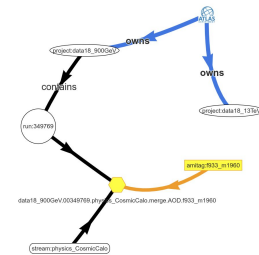
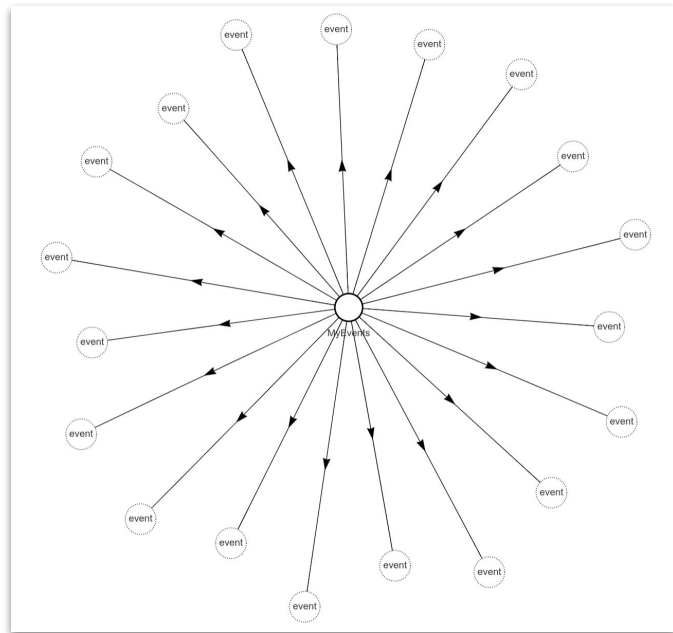
Virtual Collections

- Virtual Collection:
 - Collection Vertex
 - + Edges to contained Elements
 - + Recipy to get contained Elements

```
// Create new collection of events
eventsCollection = g.addV('ecollection')
                    .property('name', 'MyEvents');

// Find all events satisfying certain conditions
// and connect them to the event collection
g.V().has('lbl', 'event')
    .has(...some selection...)
    .collect {
        eventsCollection.addEdge('contains', it)
    };

graph.tx().commit();
```



(‘Proxy’ tunnels requests to the database through the Web Service to by-pass firewalls)

Initial Gremlin request


Atlascope 02.00.00+ [02/Oct/2021 at 23:13:57 CEST by atlevind for CERN]
Reset

?

CERN-Proxy ▾

☐ add

Search

ATLAS ▾

Execute

g.V().has('lbl', 'dataset')

Connect to the **graph server**
and request the initial **graph**

?

Customize the interactions with the **graph**.

Cluster by group type

Cluster by group size

Expand all clusters

Show all edges

☒ clusterize

☒ zoom cluster

☐ stabilize

☒ get children

☐ get parents

☐ remove old

filter:

Apply

select:

☐ hierarchical (☐ up/lr ☐ size/hierarchy)

☒ live

?

Options for interactive Graph manipulation

Figure 1: A schematic diagram of the data integration workflow. The workflow starts with two input datasets: 'dataset_18_300Gen' and 'dataset_18_13D'. 'dataset_18_300Gen' is processed through 'contigs' to produce 'contigs300'. 'dataset_18_13D' is processed through 'ovns' to produce 'ovns13D'. These two intermediate datasets are then merged into a central node labeled 'dataset_18_300Gen.phylo_CoarseGrain.merge_ADA.R53.1n180'. Finally, this merged dataset is processed through 'stream.phylo_CoarseGrain' to produce the final output.

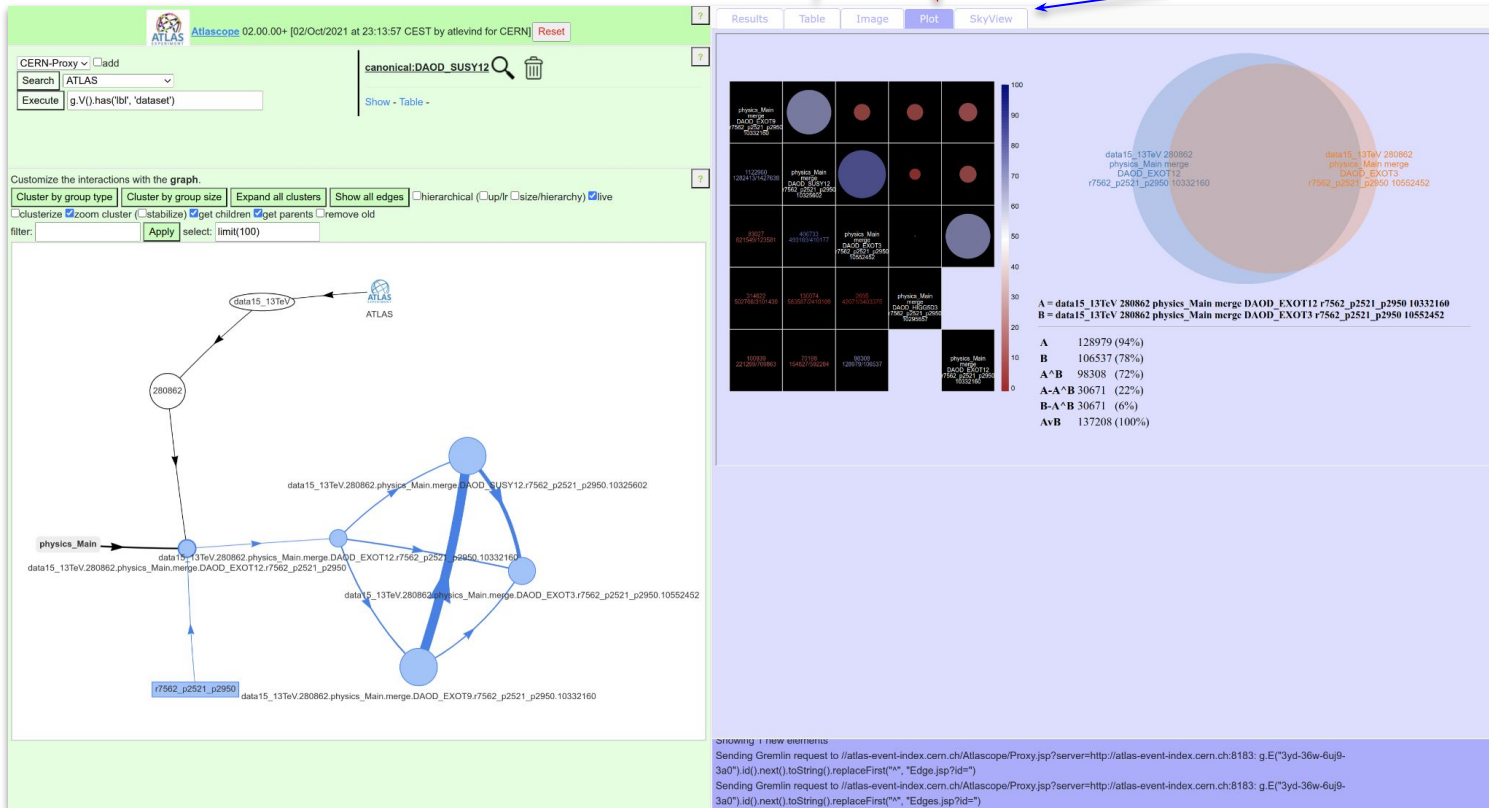
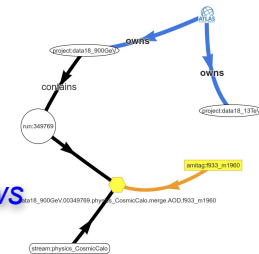
Table data can be plotted in various ways

A Click on an element (Vertex or Edge) will offer a set of available operations (internal or external applications)

Table view customisation
(visible columns, searches,...)

Executed actions
(so they can be re-used from the command line)

Various types of data views

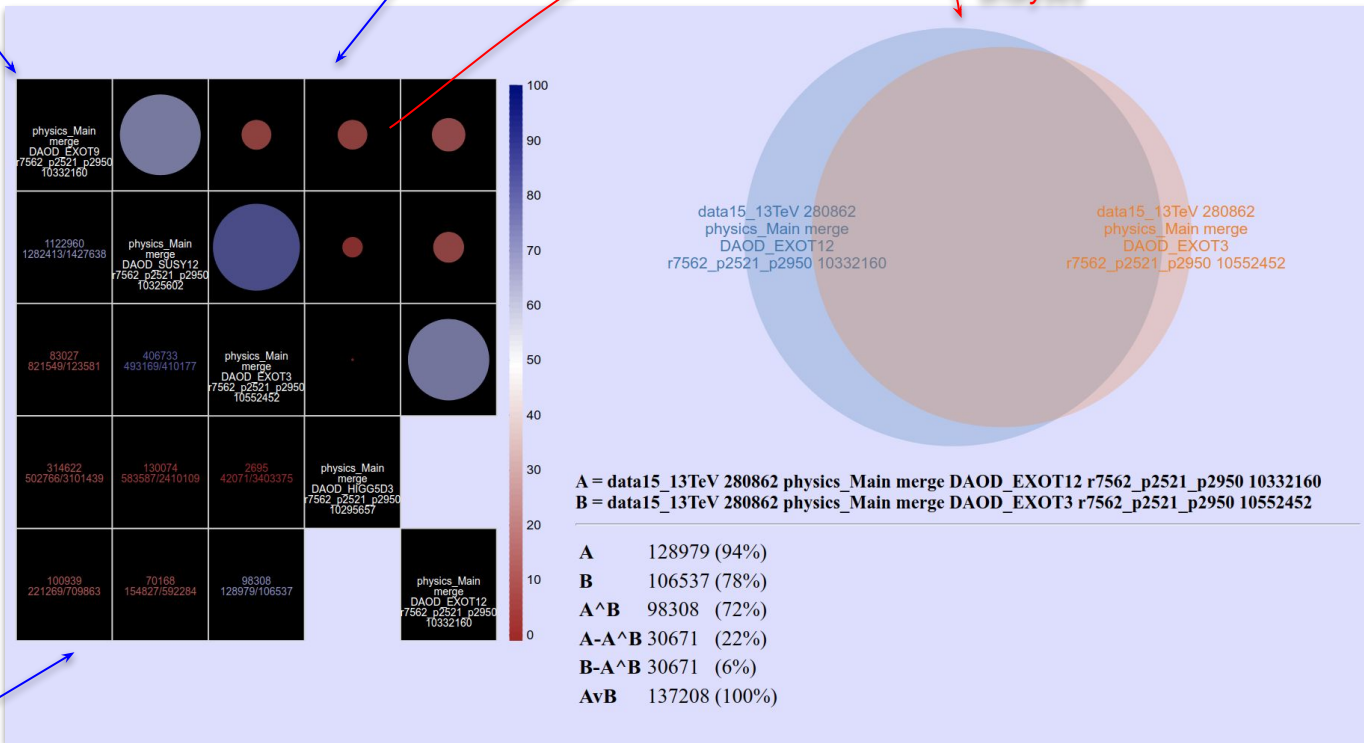


Web Client - Overlaps

Dataset names

Overlaps as circles

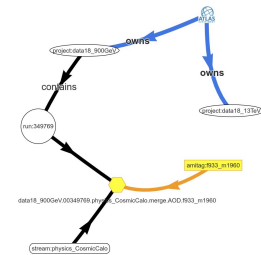
Hovering over an overlap element will give a Venn diagram with detailed analyses



Overlaps as numbers

Web Client - Plots

Table columns can be presented in various plot types



Atlascope 02.00.00+ [03/Oct/2021 at 10:49:34 CEST by atevind for CERN] Reset

CERN-Proxy v1 ☐ Add
Search:
Execute:

dataset:DAOD_EXOT8
Show - Table -

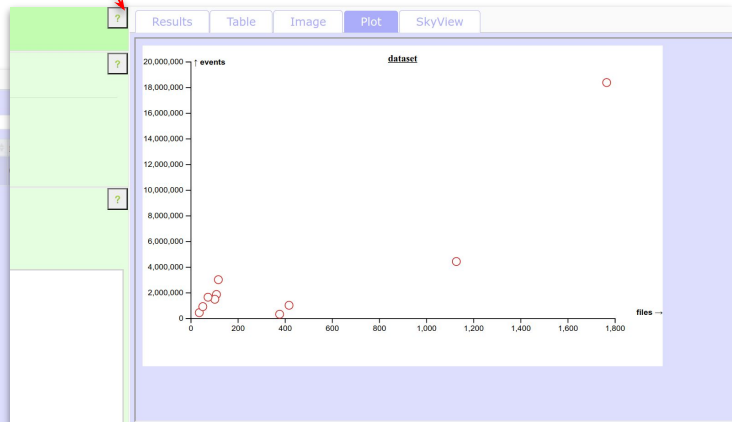
Customize the interactions with the graph.
☐ Cluster by group type ☐ Cluster by group size ☐ Expand all clusters ☐ Show all edges ☐ Hierarchical (Laplace) ☐ Size/hierarchy ☐ Live
☐ Clusterize ☐ Zoom cluster ☐ Stabilize ☐ Get children ☐ Get parents ☐ Remove old
Filter: Apply Select Limit(10)

Results
Table
Image
Plot
SkyView

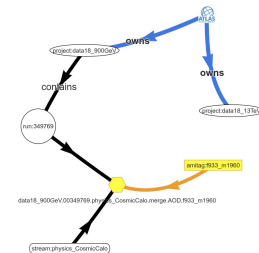
Evolution Plot
Scatter Plot
Sky View

version	runno	project	streamname	prodstep	datatype	dsid	dstypeid
r264_p3083_p4077 300800 data16_13TeV physics_Main deriv DAOD_BPHY21 11950848 10387							
X [v] [z] [s] [k] 2048: undefined							
X [v] [z] [s] [k] version: r264_p3083_p4077							
X [v] [z] [s] [k] runno: 300800							
X [v] [z] [s] [k] project: data16_13TeV							
X [v] [z] [s] [k] streamname: physics_Main							
X [v] [z] [s] [k] prodstep: deriv							
X [v] [z] [s] [k] datatype: DAOD_BPHY21							
X [v] [z] [s] [k] dsid: 11950848							
X [v] [z] [s] [k] dstypeid: 10387							
X [v] [z] [s] [k] emk: 0							
X [v] [z] [s] [k] events_nuclei: 1843958							
X [v] [z] [s] [k] nucio_at Wed Jun 09 20							
X [v] [z] [s] [k] files: 110							
X [v] [z] [s] [k] events: 1843958							
X [v] [z] [s] [k] updated_at Fri Jul 02 09							
X [v] [z] [s] [k] is_open: false							
X [v] [z] [s] [k] is_deleted: false							
X [v] [z] [s] [k] status: AVAILABLE							
X [v] [z] [s] [k] has_raw: true							
X [v] [z] [s] [k] has_trigger: false							
X [v] [z] [s] [k] prov_seen: 8192							
X [v] [z] [s] [k] tid: 20406081							
X [v] [z] [s] [k] lbl: dataset							
X [v] [z] [s] [k] phoenix: true							
X [v] [z] [s] [k] fulfill: true							
X [v] [z] [s] [k] importDate: Mon Sep 27 18							
+ r264_p3083_p4077 299584 data16_13TeV physics_Main deriv DAOD_BPHY21 44456704 10387 0 29969070 W 11							
+ r264_p3083_p4077 302393 data16_13TeV physics_Main deriv DAOD_BPHY21 24009472 10387 0 4412611 Tl 22							
+ r7562_p2521_p2950 280862 data16_13TeV physics_Main merge DAOD_EXOT12 42326576 10274 0 304311 Ss 22							
+ r264_p3083_p4077 300655 data16_13TeV physics_Main deriv DAOD_BPHY21 940800 10387 0 1469531 Tl 21							
+ r264_p3083_p4077 302919 data16_13TeV physics_Main deriv DAOD_BPHY21 26106624 10387 0 425607 Tl 22							

Select graph server and initial graph.
then select an element to see possible actions.
Sending Gremlin request to atlas-event-index.com.ch/AtlascopeProxy/jsp?server=http://atlas-event-index.com.ch:8183 g V().has('tbl', 'dataset').limit(10)
Showing 10 new elements



Web Client



- Each element (Vertex or Edge) has defined graphical presentation, behaviour and available action/operations (internal or external)
 - Customisable by a StyleSheet
 - Applied either to the one element or to all elements of the same type on the screen
- All Views (tables, overlaps,...) are generic and customized for a particular case
 - All (groups of) Vertexes or Edges can be presented as a table (with their properties)
 - Any table fields can be presented as scatter plots, time evolution plots, correlation plots / Venn diagrams, sky views (3d spherical plots),... if they contain required data types
- All visual operations can be executed from a command line (by clients in many languages or via REST)
 - As they are shown on the Feedback pane

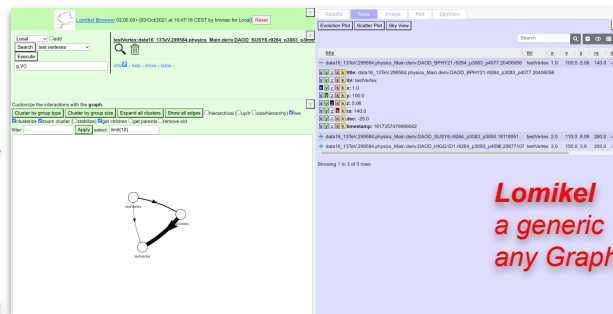
Web Client - Architecture

```

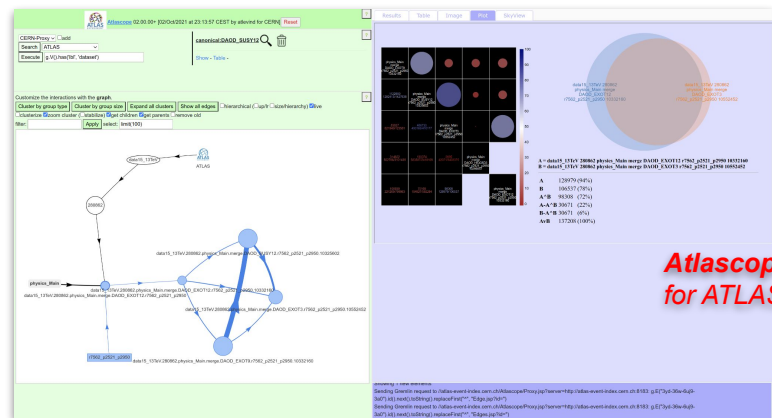
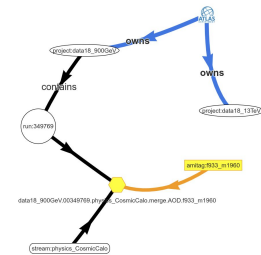
},
stylesheet.nodes.dataset = {
  properties:{gremlin:'valueMap('project', 'datatype', 'runno', 'streamname', 'prod', 'events')',
  graphics:{
    label:{js:'project + ' + runno + ' + streamname + ' + prod',
    title:'datatype',
    subtitle:{js:'events + ' events'}},
    group:'version',
    shape:{js:'datatype == 'ADD' ? 'hexagon' : 'dot'},
    image:' ',
    borderRadius:'0',
    borderWidth:'1',
    borderDashes:[1,0],
    value:'events'
  }
},
actions: [
  {name:'Show', url:{gremlin:'id().next().toString().replaceFirst(' ', '%20')'},
  {name:'Table', url:{gremlin:'id().next().toString().replaceFirst(' ', '%20')'}
},
stylesheet.nodes.canonical = {
  properties:{gremlin:'valueMap('project', 'datatype', 'runno', 'streamname', 'prod', 'events')',
  graphics:{
    label:{js:'project + ' + runno + ' + streamname + ' + prod',
    title:'datatype',
    subtitle:{js:'events + ' events'}},
    group:'version',
    shape:{js:'datatype == 'ADD' ? 'hexagon' : 'dot'},
    image:' ',
    borderRadius:'0',
    borderWidth:'1',
    borderDashes:[1,0],
    value:'events'
  }
},
actions: [
  {name:'Show', url:{gremlin:'id().next().toString().replaceFirst(' ', '%20')'},
  {name:'Table', url:{gremlin:'id().next().toString().replaceFirst(' ', '%20')'}
}

```

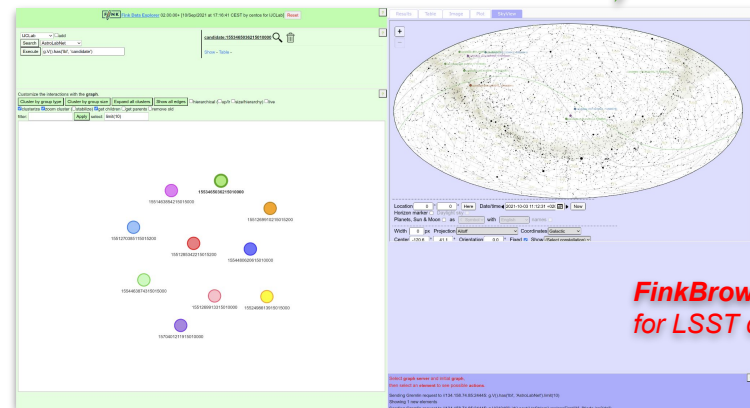
Customising
stylesheet
(json with
embedded JS
and Gremlin)



Lomikel
a generic Tool for exploring
any Graph database

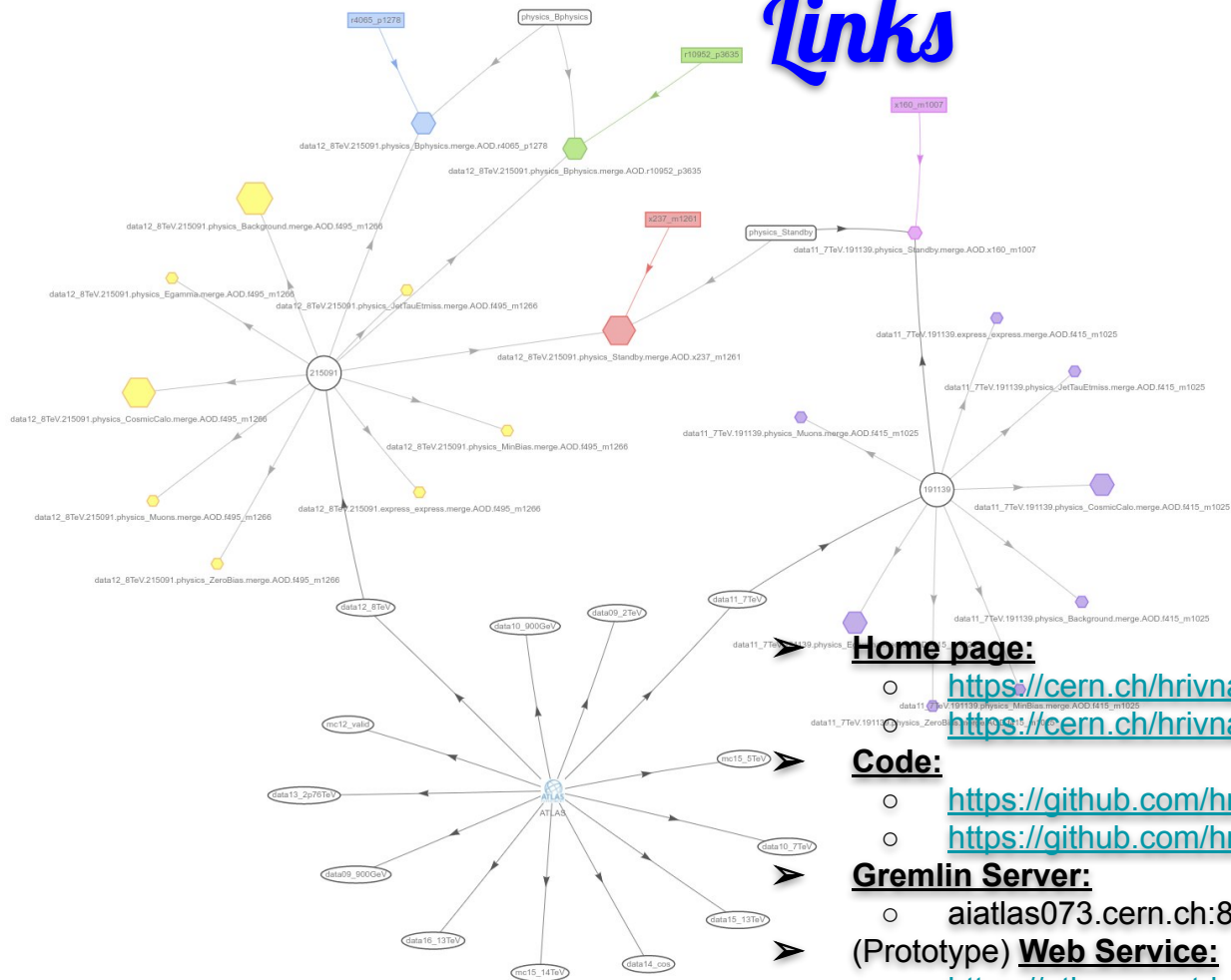


Atlascope
for ATLAS data

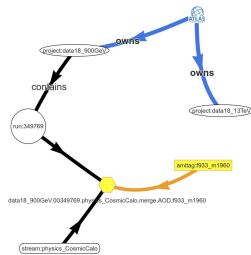


FinkBrowser
for LSST data

Links



Doesn't use
Core Phoenix
framework
yet.



Home page:

- <https://cern.ch/hrivnac/Activities/Packages/Lomikel>
- <https://cern.ch/hrivnac/Activities/Packages/Atlascope>

Code:

- <https://github.com/hrivnac/Lomikel>
- <https://github.com/hrivnac/Atlascope>

Gremlin Server:

- [aiatlas073.cern.ch:8182](https://cern.ch/aiatlas073)

(Prototype) Web Service:

- <https://atlas-event-index.cern.ch/Atlascope/?profile=CERN>