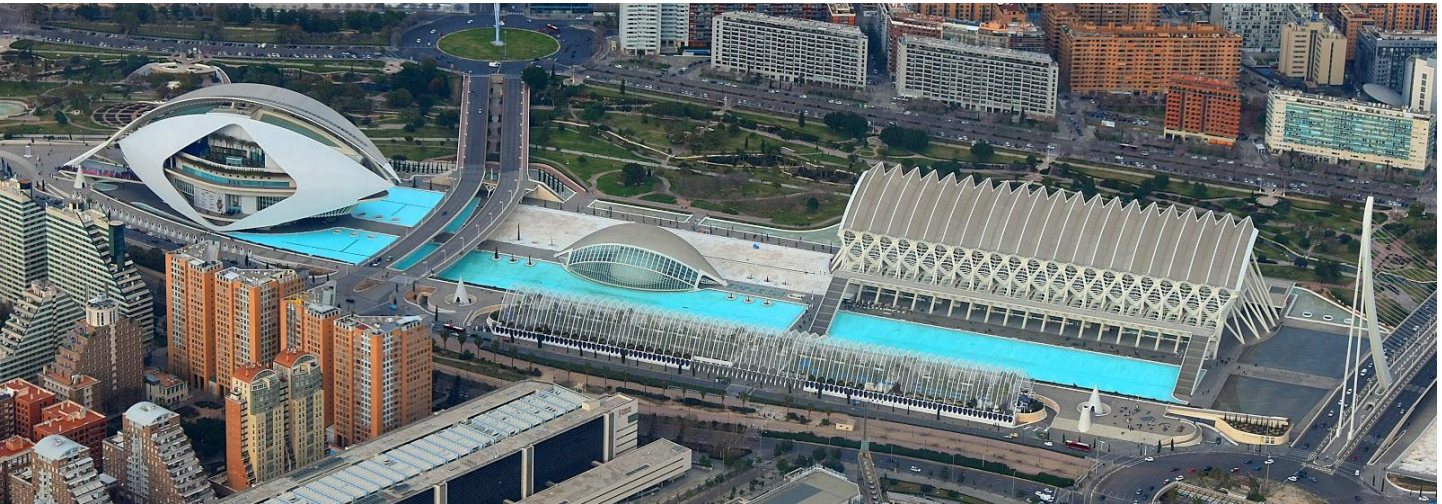
WhiteBoard Project

Core System Evolution

WhiteBoard *in more general definition*



Julius Hrivnac
EI WS, 12-13 Feb 2018, Valencia

EI Architecture

- Event Index = space of Event Collections (called TagFiles) + operations on those collections
- Each operation creates a new TagFile (virtual, referential or full)
- Operations:
  - Selection (searching)
  - Transformation
  - Merge
  - Creation
- Operations can involve any (Java) code executed for selection or creation
- TagFiles may have different schemas
- TagFiles can be stored in different technologies
  - If they support required access methods
- TagFiles can be annotated with additional information (tags,...)
  - This information can be searched on

EI Architecture

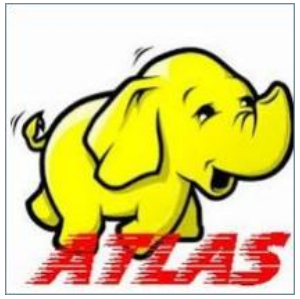
- Key feature: flexibility (important for EWB !)
  - TagFiles (and TagSets) can be freely annotated
  - No fixed schema:
    - Each TagFile can have it's own schema
    - Derived TagFile can have different schema from the original one
      - Each operation creates a derived TagFile
    - TagFiles can be parallel-extended (i.e. with additional attributes)
  - The same events present in different TagFiles (either fully or as a reference) can be extended or annotated in a different way (with different additional information)
  - Considered migration to HBase would add even more flexibility (annotation per event)
- Three layers:
  - Storage: currently Hadoop MapFiles & HBase tables
  - Organisation Framework: TagConvertor (historical name)
  - Presentation: Remote CLI + Interactive Web Service

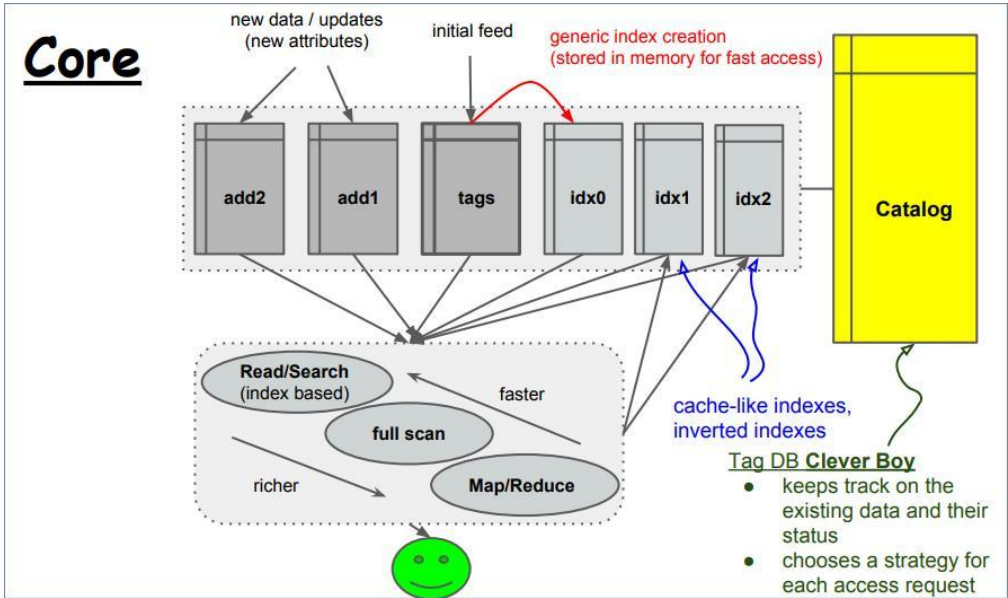
EI Implementation

- All architectured features are implemented
- Subset of features is available to users via several (coherent) interfaces
  - Remote CLI
  - Web Service
  - API
- The level of implementation has been driven by actual user requirements

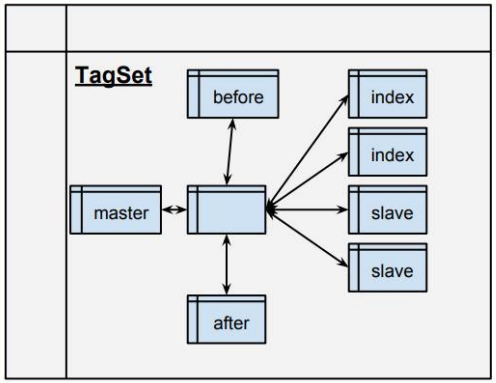
Fully Implementing Original Design

- From the original design (partially implemented)
  - Adding attributes to existing events
    - Currently new attributes can be only added to whole TagFiles
- Four kinds of TagFiles:
  - Full content
  - List of events
  - Virtual (just in Catalog)
  - TagSets = sets of TagFiles





Fully Implementing Original Design



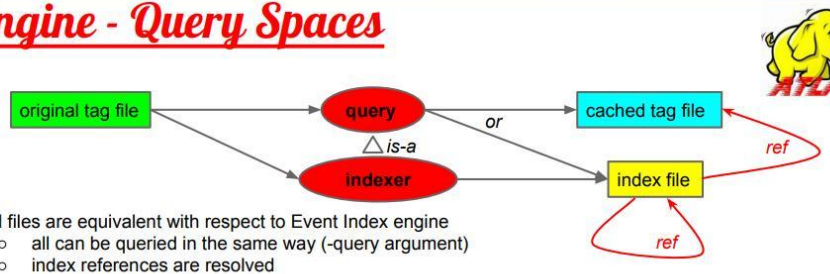
TagFile
description
id
name: "..." (TagSet name for master, extension or index name for others)
path
type: "tags"|"index"|...
format: "seq"|"map"|...
info: "..."
relations
before: id
after: id
master: id (missing for master)
slaves: {id}
indexes: {id}

Other fields (describing structure) will be added later - in coordination with MsgService & Msg format (msg header ?).

Data in HDBS are stored in collections of TagFiles. Collection of TagFiles makes a TagSet. Each TagSet has one master and set of "before/after" filesets (vertical partitions), slave TagFiles (horizontal partitions) and index TagFiles. Catalog should check consistency and presence of all components.

Each TagFile is represented by one entry in HBase table.

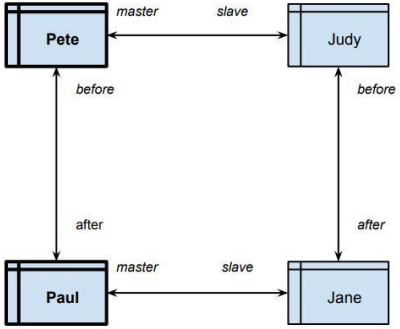
Engine - Query Spaces



- all files are equivalent with respect to Event Index engine
  - all can be queried in the same way (-query argument)
  - index references are resolved
    - including chain of references
  - users can bookmark created TagFiles
- if trigger-based indexes created in systematic way, they would be similar to Jack-like reverse indexes (could evolve to)
- **Query Spaces:**
  - secondary files (cached tag files and indexes) know their 'creation clause'
  - they constitute *dependency tree* between files
  - with *query measure*
  - when new query is requested, it will find the closests larger (by *query measure*) file and execute on it
  - this can speed up queries once database contains lot of data and big cache of results

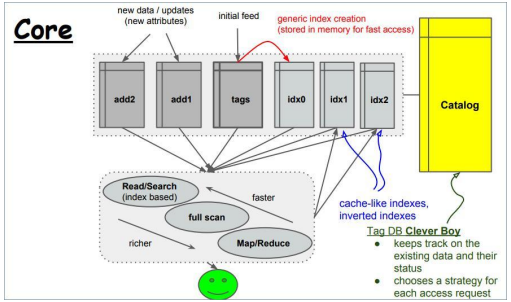
under development

Example TagFiles



Testing FileSets
Pete and Paul contain primary tags, Judy and Jane contain additional information. They are all linked in the Catalog.

Executed by "ant check-full", which calls FullTest.java



Example

We can change implementation, but should not lose this functionality (which maps very well to WB mission).

This command

1. Takes a set of datasets (defined by regexp + explicit name + attribute)
2. Filters them for some triggers + anything else (any Java code)
3. Transforms using user-supplied Java class
4. Extends with another field
5. Writes output into a new TagFile (which can be used in the same way)

The second command annotates new TagFile.

Catalog keeps trace => we can reconstruct genealogy of TagFiles.

```
$ ei
-query 'id:EI15.1.data15_13TeV.*.merge.AOD.f594_*;dataset:data15_13TeV.00279515.physics_Main.merge.AOD.r7562_p2521
        status:good'
-mr 'trigFired("HLT_tau35_medium1_tracktwo_tau25_medium1_tracktwo_L1TAU20IM_2TAU12IM") &&
     !trigFired("HLT_tau35_tight1_tracktwo_tau25_tight1_tracktwo_L1TAU20IM_2TAU12IM") &&
     myFilter()'
-aux 'net.hep.atlas.Database.EIHadoop.Accessor.Aux.MyAuxTransformation;-opt 1'
-extent 'myField=String.valueOf(BunchId*BunchId)'
-outname 'MyExampleTagFile'
```

```
$ catalog -query id:MyExampleTagFile -modify 'myNewTag:abc createdBy:julius'
```

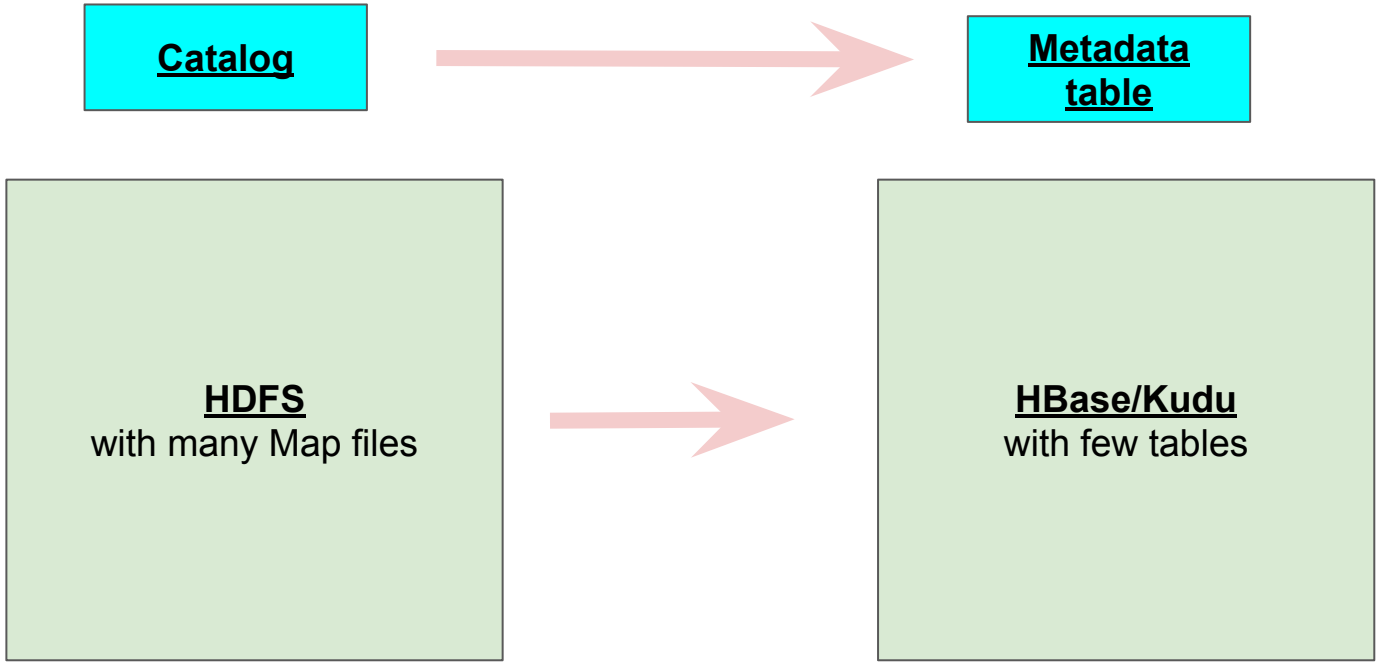
Fully Implementing Original Design

- Other implemented or *prepared* features
  - Creation of derived TagFiles
    - Dataset overlaps
    - Trigger overlaps (on subset of events)
    - Trigger statistics *(coupled with statistics on other variables)*
    - Complex statistics on any variable
    - Searching using any code (in fact, full analyses program can be in principle executed)
  - *Event Lookup with searching / reporting on other variables (e.g.trigger)*
  - Coherent set of client tools (WS, CLI, *portable CLI*)
  - *Integration with AMI*
  - Move all data into HBase
    - Has been evaluated at the beginning, refused for performance reasons
    - All performance reasons fixed
    - Already using HBase for significant part of the data (Event Lookup)

Summary & Notes

- Most requirements on WB were in the original design of EI Core & many of them are already satisfied by the production implementation
  - Level of their implementation depends on actual User Requirements
- We should not lose that functionality by migrating EI to Kudu,...
- The current system delivers **rich functionality, feature flexibility and excellent performance**
- However, if we start today, we would keep the architecture:
  - A space of TagFiles (= Collections of Events)
  - Two-level navigation: Catalog + Event Collections
- but change the storage to HBase, Kudu,...
  - MapFiles too rigid
  - HBase has been evaluated at the beginning and refused for reasons, which are no more valid
- HBase vs Kudu:
  - HBase:
    - +: schema free (we use it)
    - -: typeless (everything is string or bytearray)

Plan ?



How to define TagFiles (Collections) within HBase/Kudu tables ?

(so that an operation would not create new dataset, but update the table)