Design of the Jet Performance Software for the ATLAS Experiment at LHC

C. Doglioni[1], P. Francavilla[2], P. Loch[3], K. Perez[4], C. Roda[2], R. A. Vitillo[2]

on behalf of the ATLAS collaboration

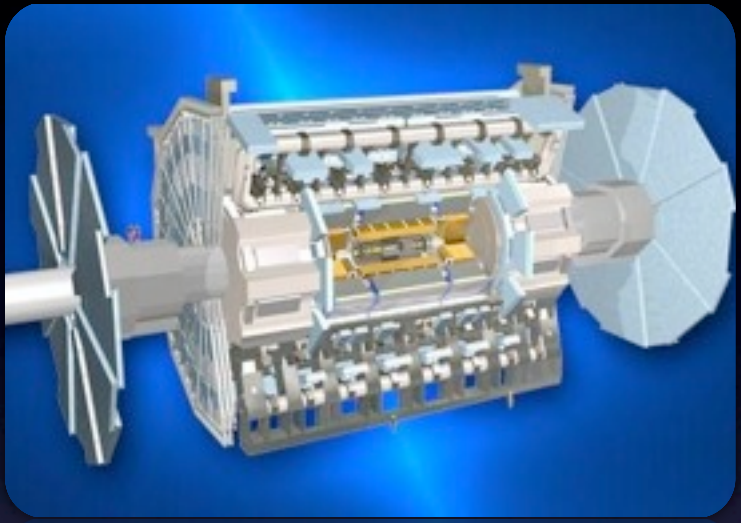
MASCOT 10 Las Palmas de Gran Canaria, Oct 20-22, 2010

1.University of Oxford
2.Università degli Studi di Pisa, INFN Pisa
3.University of Arizona
4.California Institute of Technology

Outline

- The ATLAS detector at LHC
- Software Design
- Athena
- Jet performance evaluation
- Motivation for a new framework
- Jet Performance Framework
- Conclusions

The ATLAS detector at LHC



- ATLAS is one of the six particle physics experiment at the Large Hadron Collider (LHC)
- Proton beams interact in the center of the detector
- The ATLAS detector has been designed to provide the most complete information of the ~1000 particles that emerge from the proton collisions
- ATLAS acts as a huge digital camera composed of ~100 Million of channels which provide a "picture" of about 1 Mbyte defined as event
- Each event is fully analyzed to obtain the information needed for the physics analysis

Software Design

- ATLAS is a large community, ~ 3000 researchers
- The design of a software can make a big difference for the success of an experiment
- Can save man-hours
- **Abstraction**
- **Generalization**

Athena

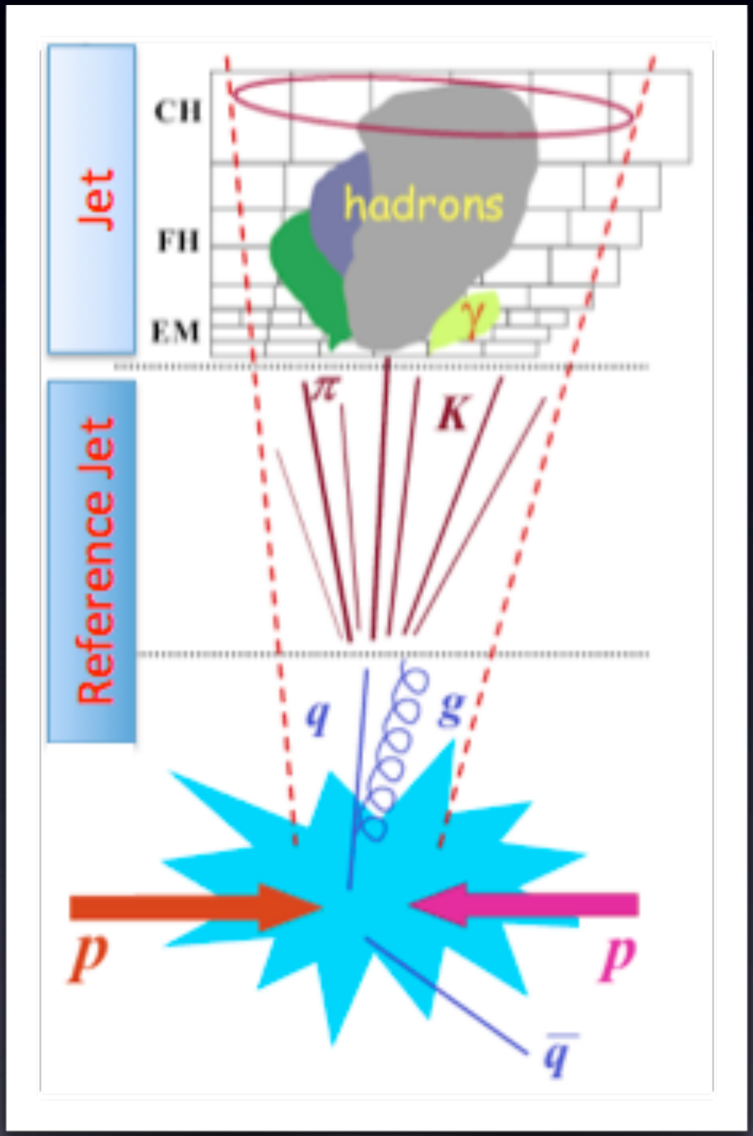
- Provides the most common functionalities for analysis
- Analysis composed of a series of algorithms
- Algorithms access data contained in stores
- Data converters serialize/deserialize from persistent data format to a transient one
- Control and data parameters can be configured via a set of python scripts

Jet performance evaluation (1)

- A jet is a narrow cone of particles produced by the hadronization of a quark or gluon
- The particles in the cone produce the signal that is visible in the detector and from this signal, that is stored in the event picture, the information on jets is extracted

Jet performance evaluation (2)

- **Jet collection**: jets as seen in the detector are ensemble of detector cells
- **Jet reference collection**: reference jets is what we assume has produced the detector signal and are produced with simulations
- Performance evaluation is used to compare e.g. jet reconstruction algorithms or different datasets



An example

- We treat jets, photons, electrons and other particles as particle collections
- The following is part of the workflow for performance evaluation:

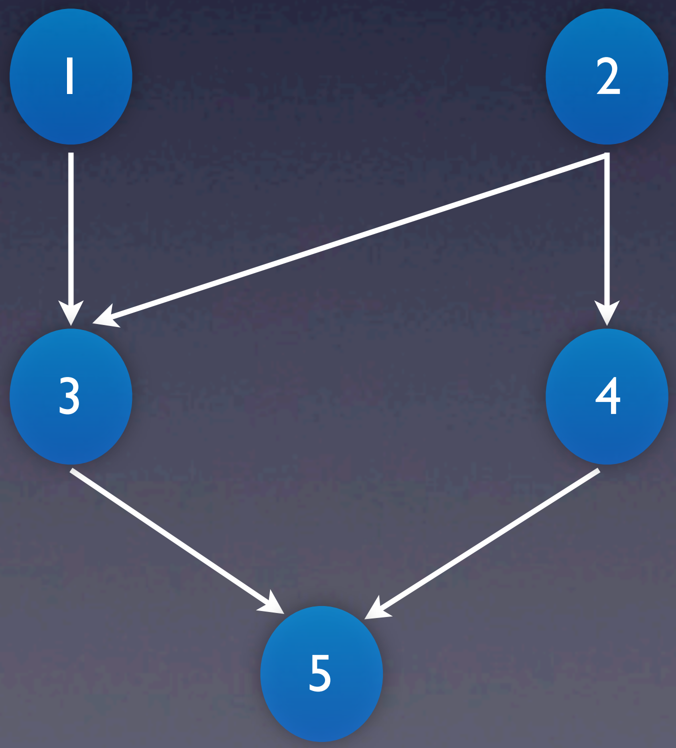
1. For each collision event, we have two jet collections: a reconstructed and a reference one
2. From the reconstructed one we want to remove all jets that overlap with electrons or photons
3. We also want to remove jets, from both collections, cutting on kinematic and geometrical variables
4. Save interesting data and proceed with the next event
5. Produce performance plots

Motivation for a new framework

- Performance applications have many commonalities that in the first analysis packages were not exploited
- Many repetitions
- Lack of scalability
- Lack of configurability
- Lack of portability

The Jet Performance Framework (1)

- A performance evaluation can be seen as a data-flow graph (DFG)
- A DFG represents dependencies between a number of operations



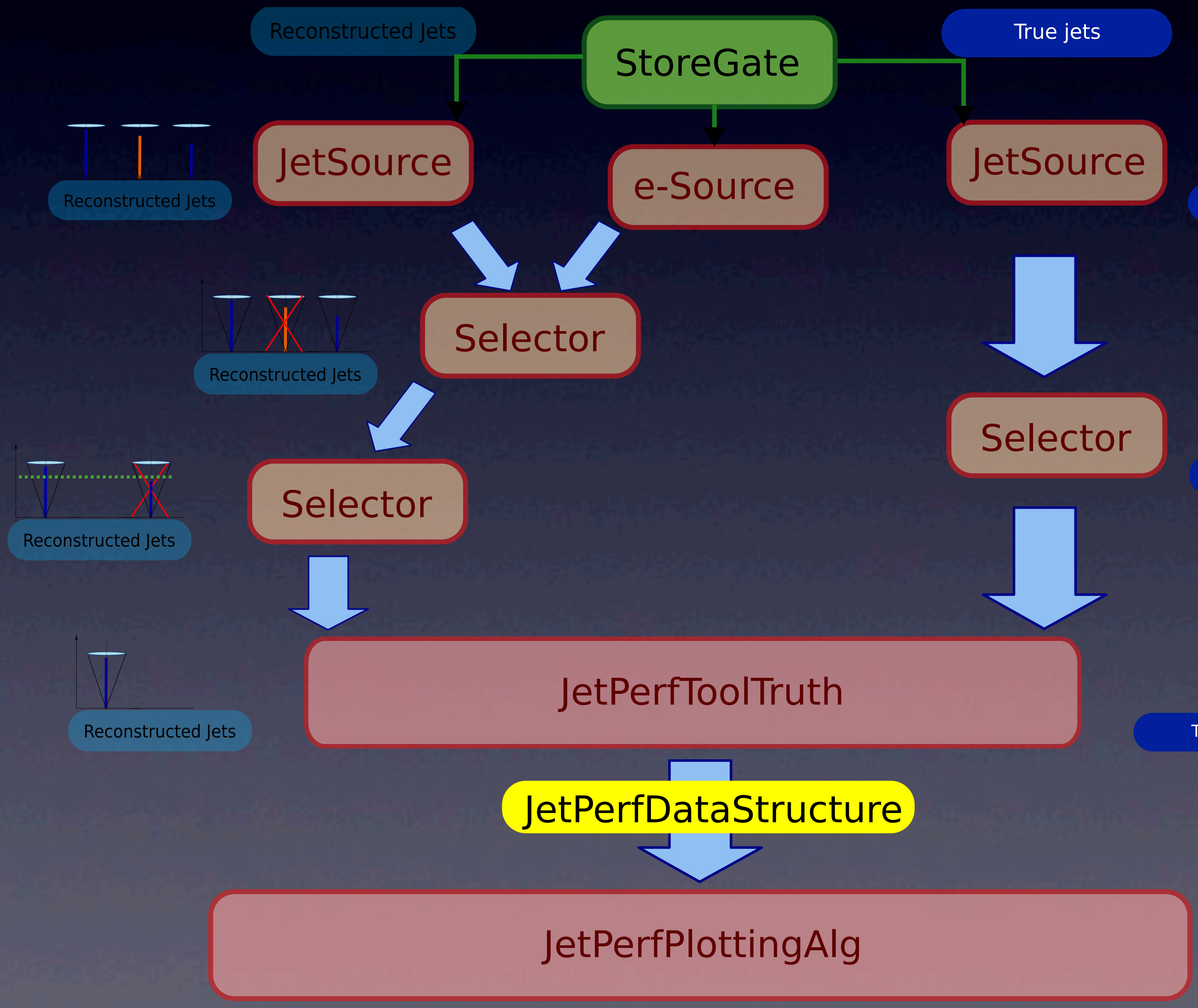
The Jet Performance Framework (2)

- In our case, data represented as particle collections
- An operation can take a set of collections as input, execute an algorithm, and return a new collection
- User can define their own DFGs via configuration files

An example (2)

1. Source nodes permit to access data
2. Overlap removal nodes permit to remove overlapping particles
3. Filtering nodes permit to remove particles that do not match a criteria
4. Persistifier nodes save particles in transient form

An example (3)



Reconstructed Jets

StoreGate

True jets

JetSource

e-Source

JetSource

Reconstructed Jets

True jets

Selector

Reconstructed Jets

Selector

True jets

Selector

Reconstructed Jets

JetPerfToolTruth

Reconstructed Jets

True jets

JetPerfDataStructure

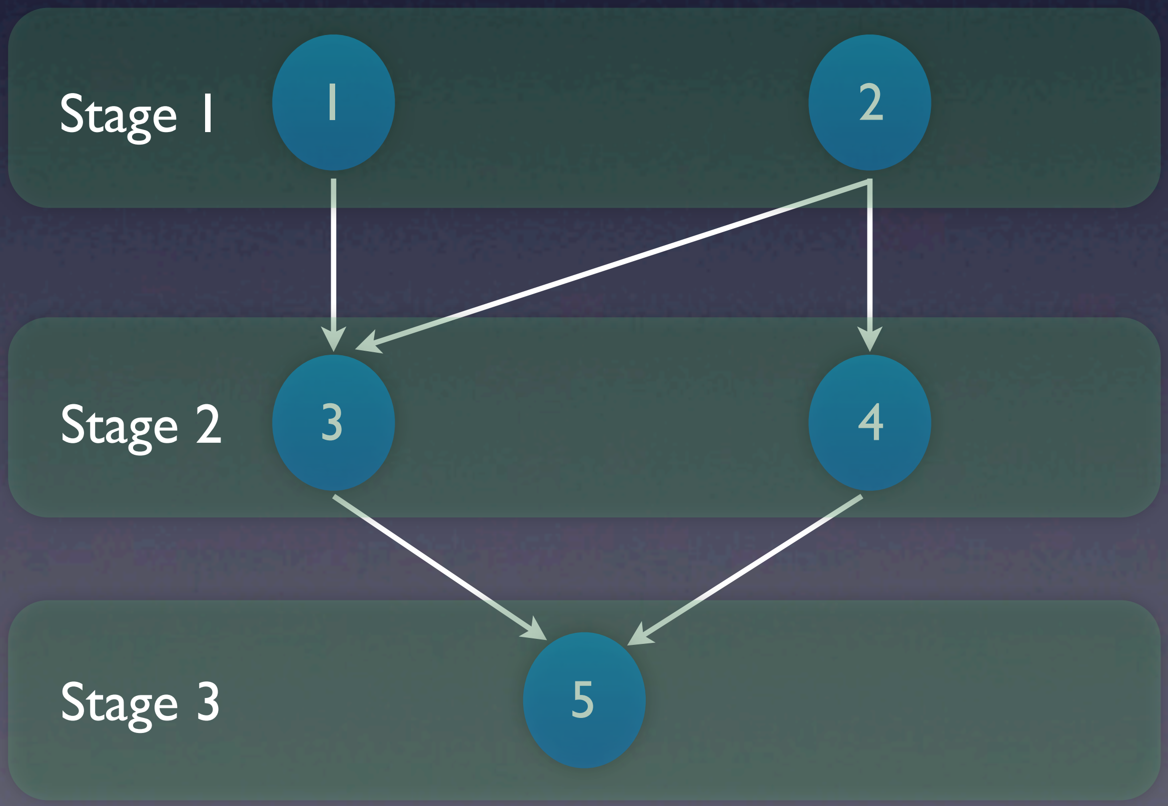
JetPerfPlottingAlg

The Jet Performance Framework (3)

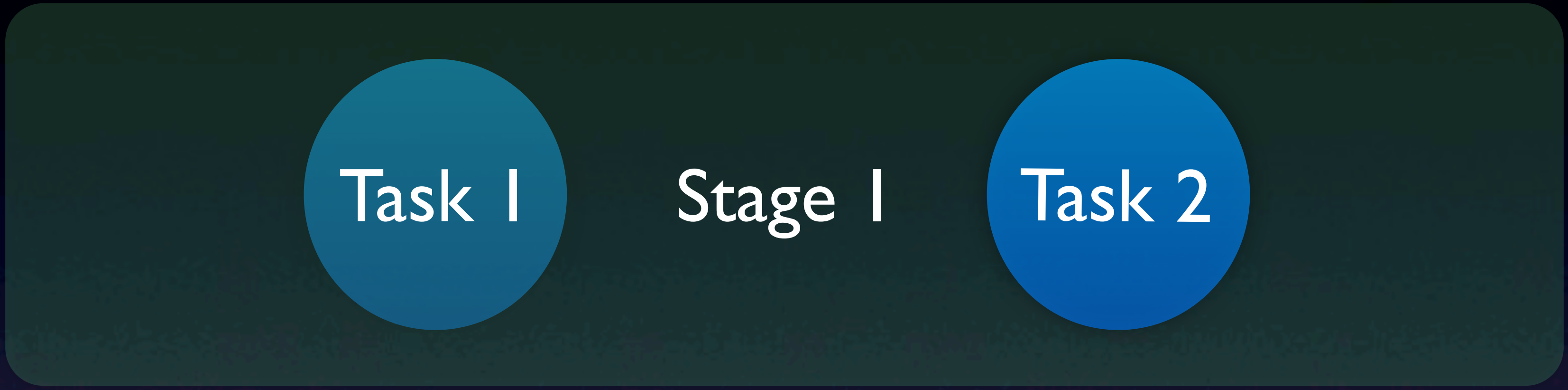
- Each node of the graph is scriptable from the configuration
- The scripting facility permit e.g. to define cut expressions
- A set of basic operations has been provided to retrieve, filter, store and plot data
- A repository permit user to share their custom operations with the community

The Jet Performance Framework (4)

- Possible side benefit: stream parallelism
- In a DFG we can exploit pipeline parallelism between different stages
- Execution time depends "only" on the execution time of the heaviest stage



The Jet Performance Framework (5)



Task 1

Stage 1

Task 2

- In each stage task parallelism can also be exploited
- Needs further study

Conclusions

- Performance evaluation can be written just by assembling basic blocks
- High customization degree without the need to recompile
- A GUI can replace the configuration file
- Fully operational and used for many physics analysis

Questions?