NEW DEVELOPMENTS IN FAIRROOT

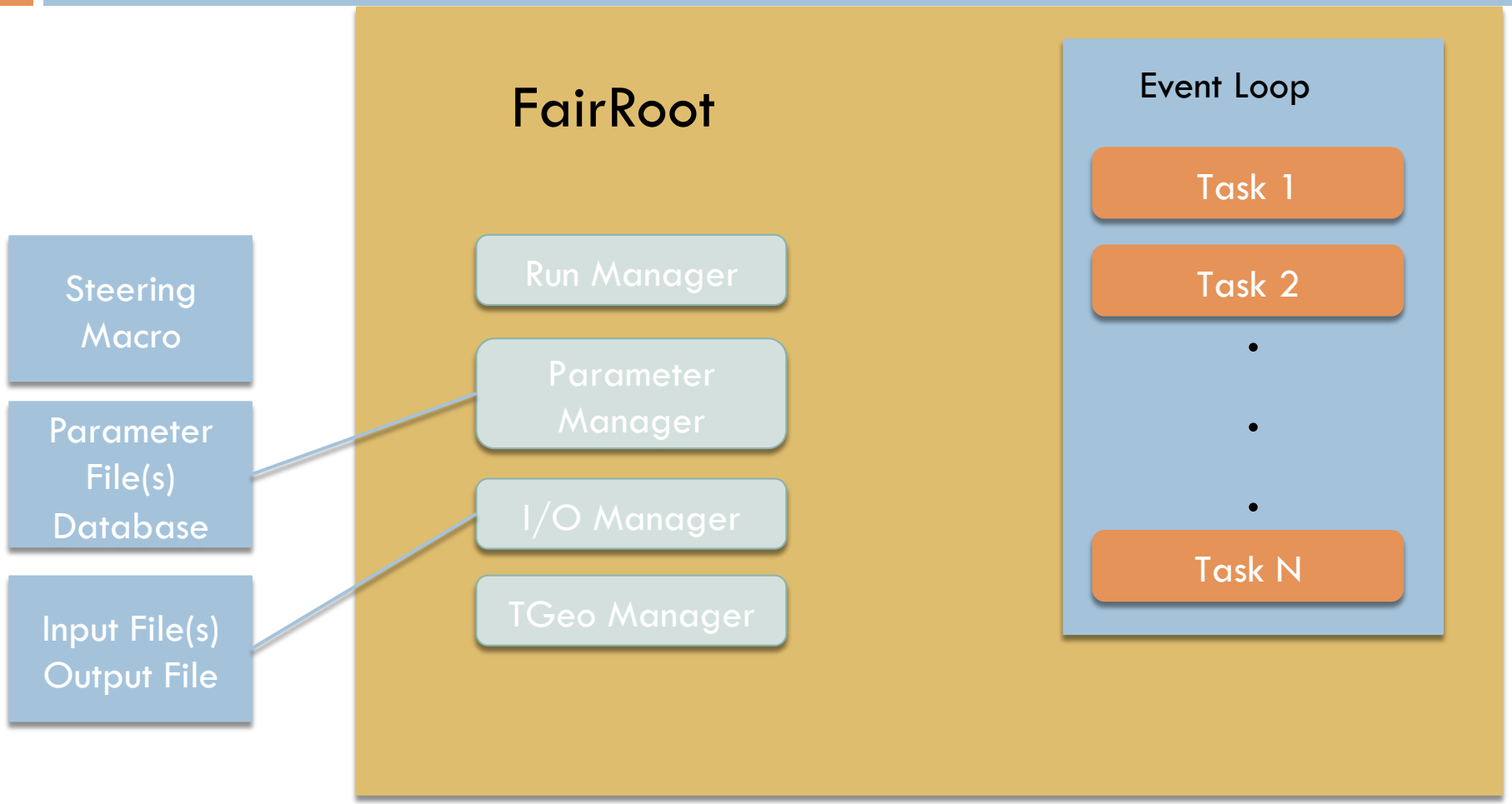
Selling FairRoot in1Slide

- Open Source project (LGPL v3) for Simulation, Reconstruction, and Analysis of HEP experiments
- Hosted at Github
  - https://github.com/FairRootGroup/FairRoot
- Used by ~ 10 experiments
  - Mostly at GSI
  - Some at CERN
  - Some more are evaluating FairRoot
- Core Development team at GSI
  - Each experiment has one developer with a shared position experiment/core team
    - Improves communication between core team and experiments
  - Many developments from the experiments went into FairRoot
    - Tobias Stockmanns, Continuous Readout Simulation with FairRoot on the example ot the PANDA experiment, Track 2, 14.04. 18:00 – 18:15
    - Dmytro Kresan, Online/Offline reconstruction of trigger-less readout in the R3B experiment at FAIR, Track 1, 16.04. 10:15 – 10:30

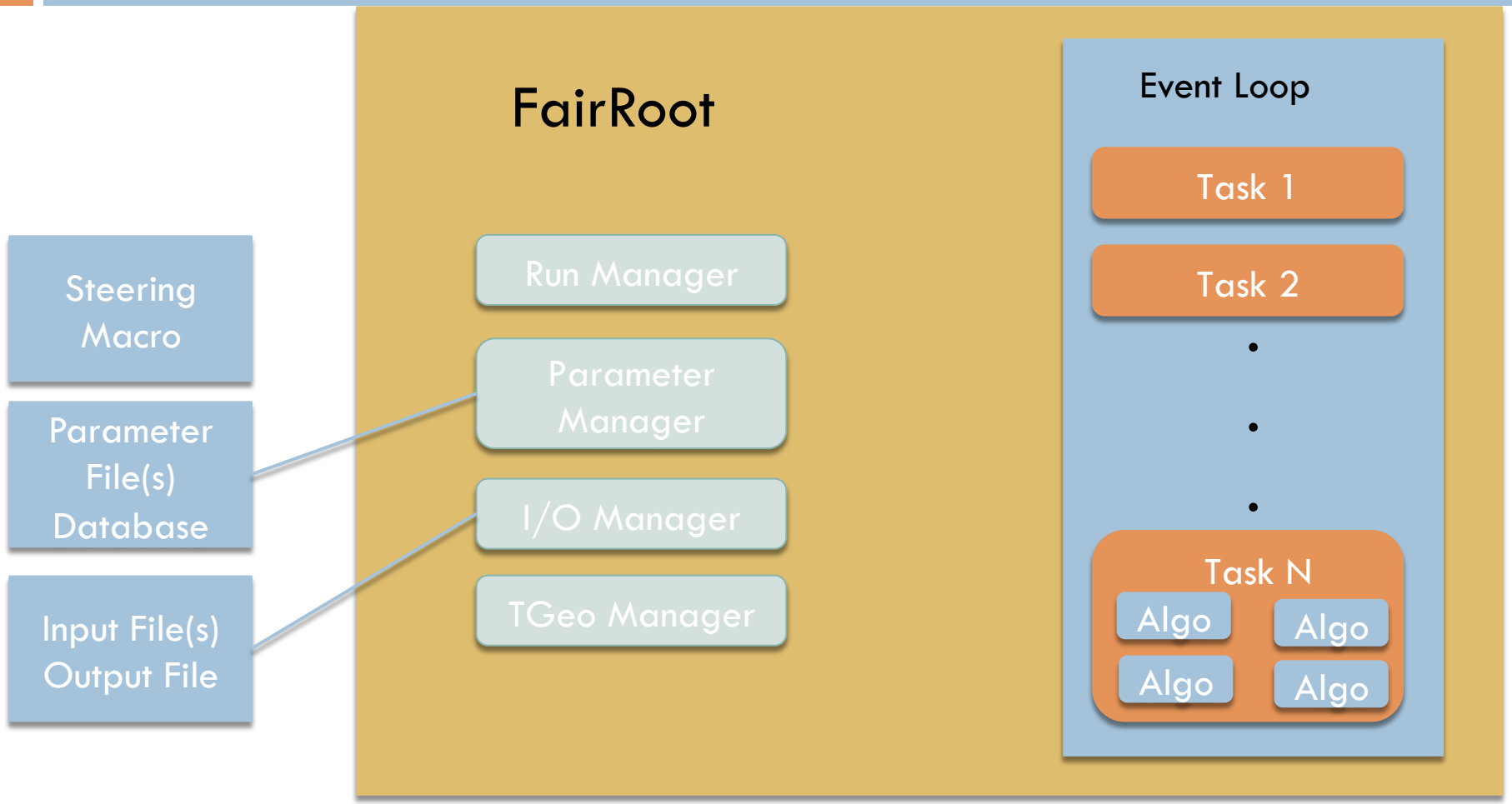
Ok. 2 Slides

- Make users life as easy as possible
  - Provide scripts which install all the needed dependencies
  - Modern build system using CMake
    - Continuous Integration
    - Coverage Analysis
    - Dashboard
  - Define simulation, reconstruction, or analysis workflow in a ROOT macro which is executed in ROOT
- Code runs on Mac OSX and (all) Linux flavors
  - Tested on
    - Mac OSX 10.6 – 10.10; OpenSuse 13.1 and 13.2; Fedora 19, 20 and 21; Debian 6, 7, and 8; SLC 6, and 7; Ubuntu 14.04, and 14.10 (all 64bit)
    - Ubuntu 14.04 (32bit)
    - Probably many more different flavors on user systems

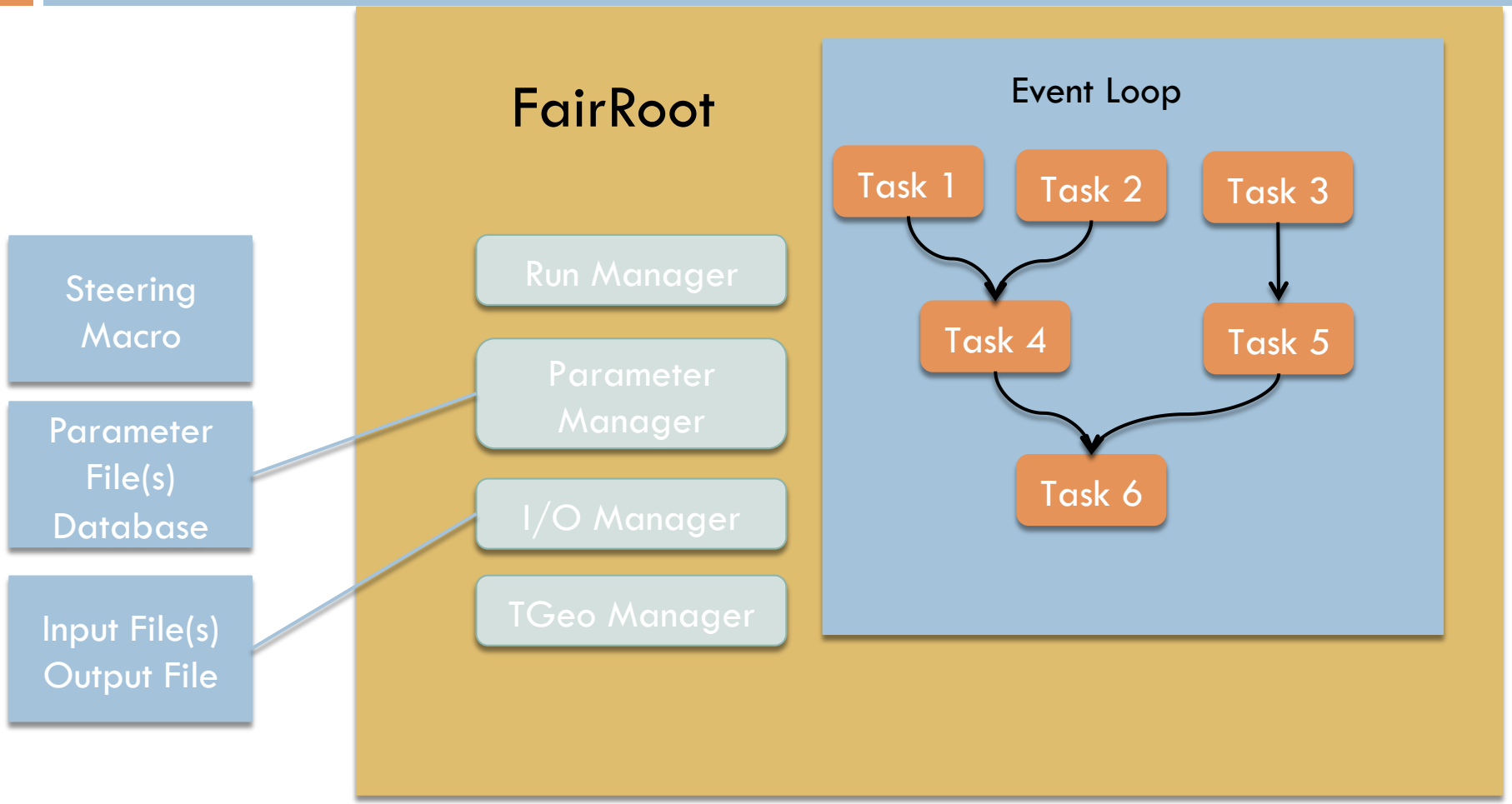
Current Layout



FairRoot: parallelized Tasks



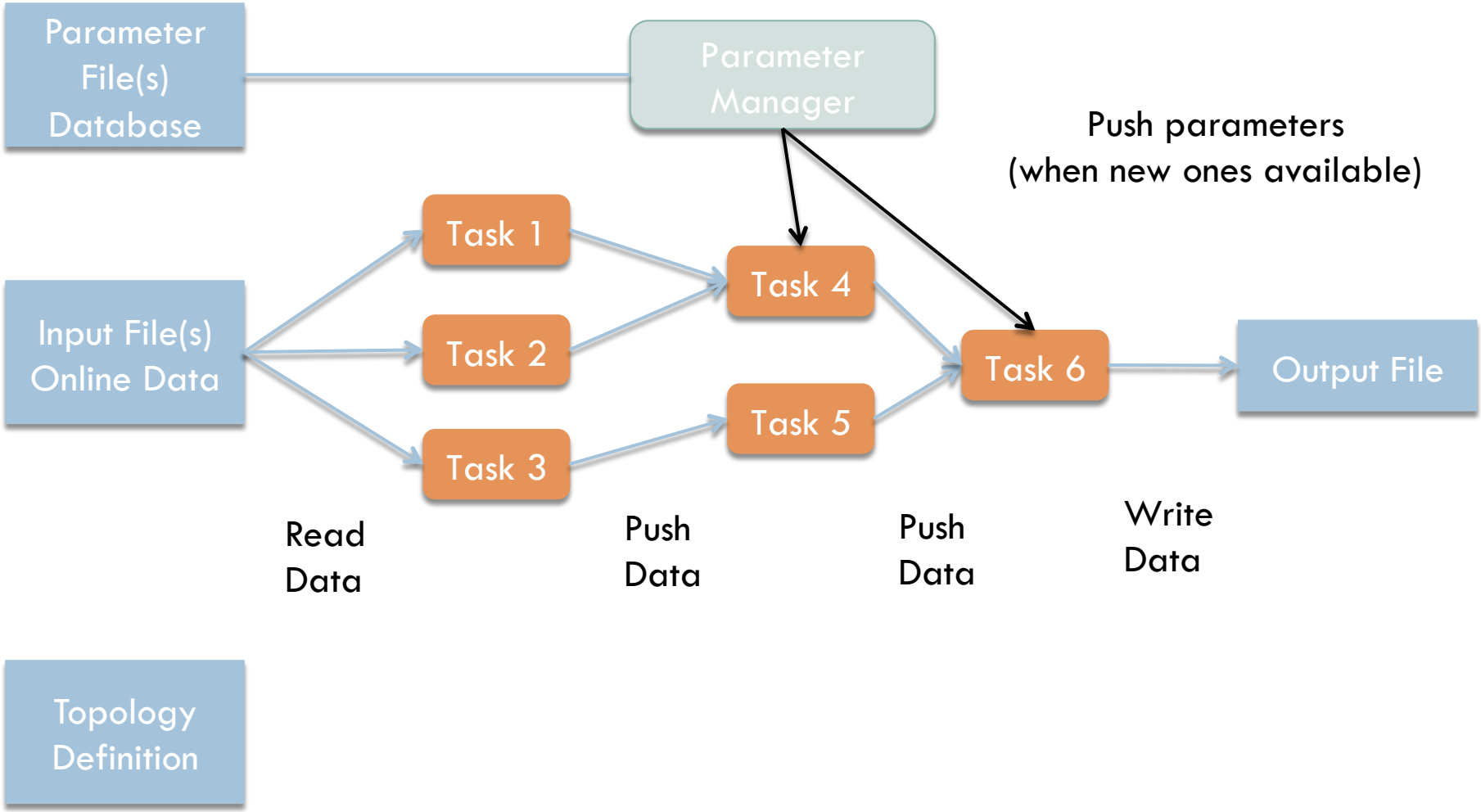
FairRoot: parallel Tasks



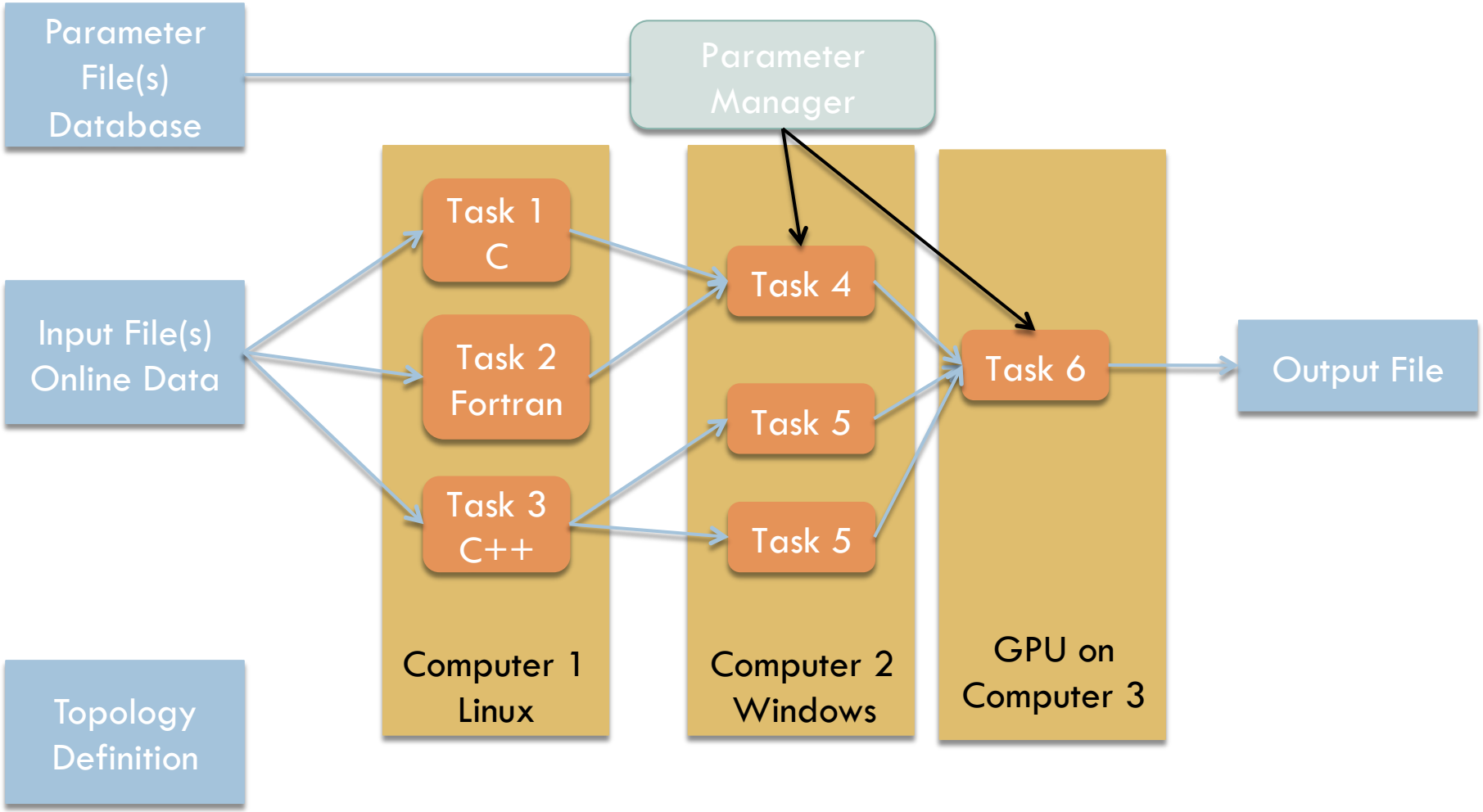
Problems of current layout

- Start Tasks in different independent threads.
  - Huge changes in the framework needed
  - Management overhead?
    - How to handle concurrent data access?
  - Program is still monolithic
    - If one tasks crashes the whole program may crash
  - Program has still one event loop
    - Execution of next event can only start when the execution of the previous event has finished

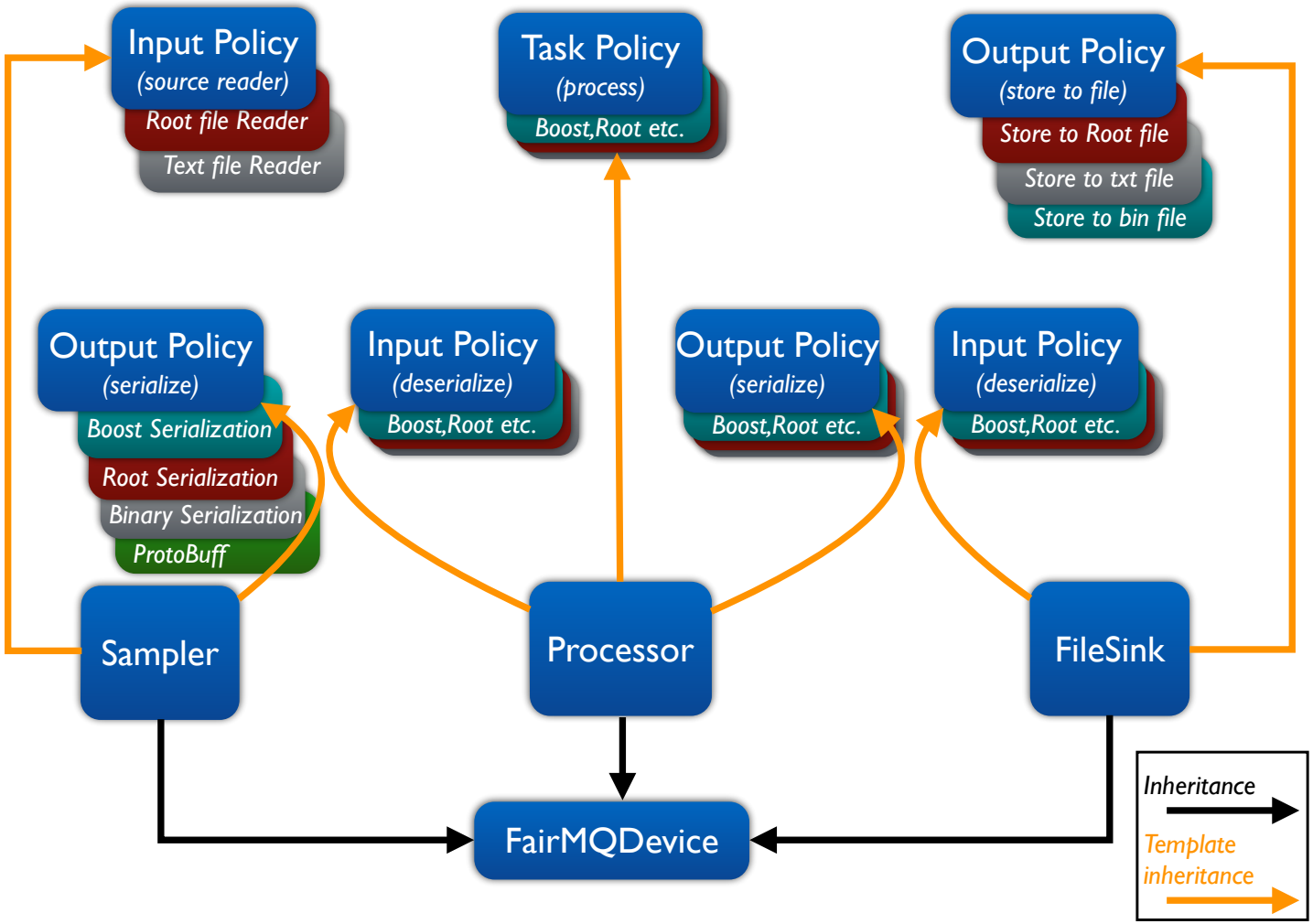
FairMQ


Parameter File(s) Database
Parameter Manager
Push parameters (when new ones available)
Input File(s) Online Data
Task 1
Task 2
Task 3
Task 4
Task 5
Task 6
Output File
Read Data
Push Data
Push Data
Write Data
Topology Definition

FairMQ


Parameter File(s) Database
Parameter Manager
Input File(s) Online Data
Task 1 C
Task 2 Fortran
Task 3 C++
Task 4
Task 5
Task 5
Task 6
Output File
Topology Definition
Computer 1 Linux
Computer 2 Windows
GPU on Computer 3

FairMQ Building Blocks



Advantages of FairMQ

- Do not use an event loop
  - No Run Manager is needed
  - Work asynchronously
- Each task is a single independent process
  - If one task fails only restart this task
- Each Task sends its results to the next task in the row
  - No central I/O Manager is needed
  - Different possible connection scenarios
  - No task has to wait, except for input data
- Allow to setup exactly the needed chain
- Implementations:
  - Matthias Richter, A design study for the upgraded ALICE O2 computing facility, Track 1, 14.04. 16:30 – 16:45
  - Alexey Rybalchenko, Efficient time frame building for online data reconstruction in ALICE experiment, Track 1, 14.04. 16:45 – 17:00

Setup of Topologies

- Setup of small number of processes easily possible with scripts
- Does not scale for large and complex topologies
- Solution: Dynamic Deployment System
  - Mohammad Al-Turany, ALFA: the new ALICE-FAIR software framework, Track 2, 14.04. 15:30 - 15:45

Outlook

- DDS
  - GUI for topology creation
- Monitoring for DDS
  - automatically (re)start devices when needed
    - Bottlenecks
    - Devices crashed
- Help experiments to move from monolithic version to MQ based version of FairRoot
- More FairRoot related presentations
  - Ludovico Bianchi, Online tracking with GPUs at PANDA, Track1, 13.04. 15:30 – 15:45
  - Vikas Singhal, Event Building Process for Time streamed data, Track 2, Poster Session A,