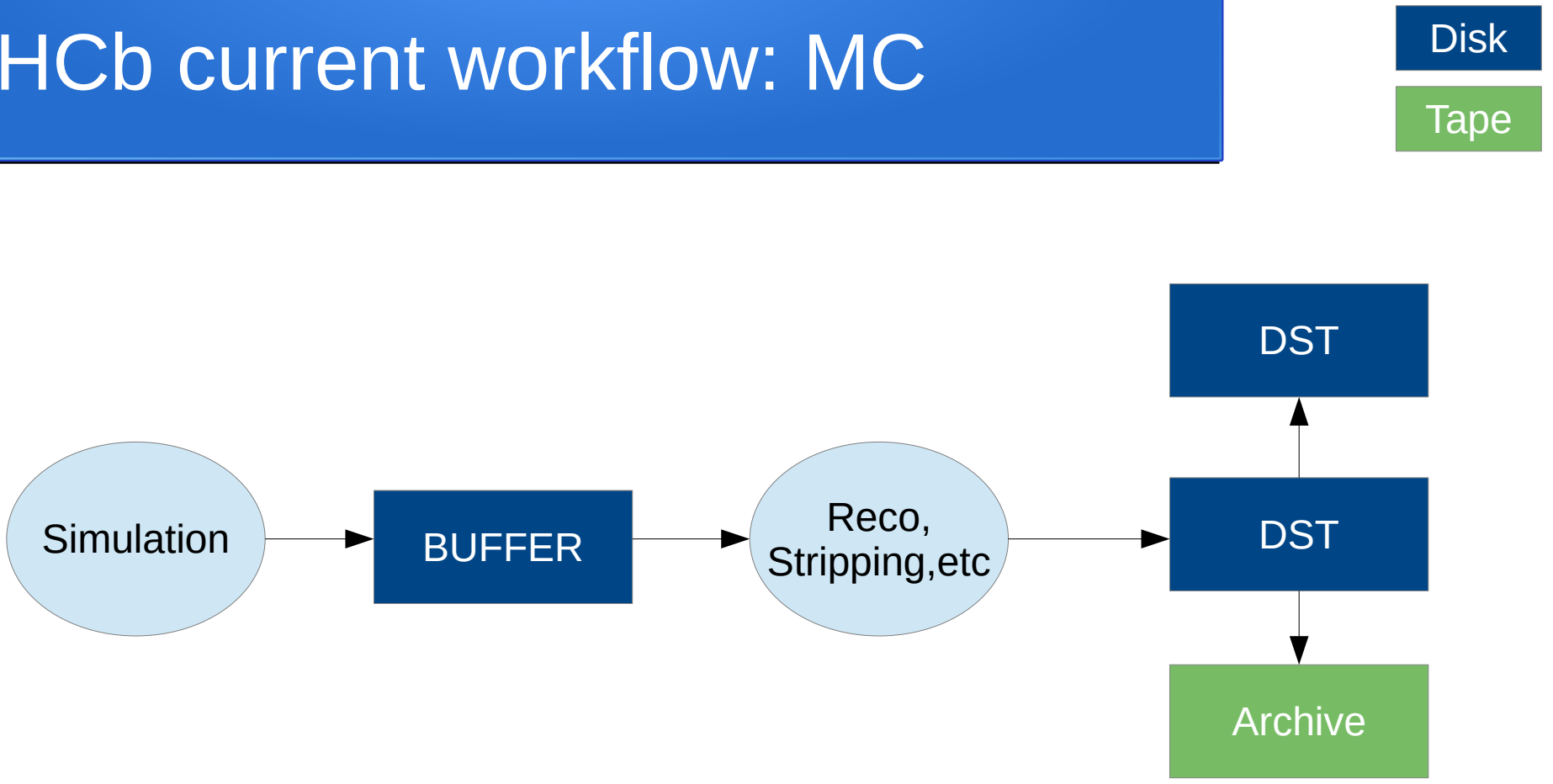
DOMA QoS Workshop

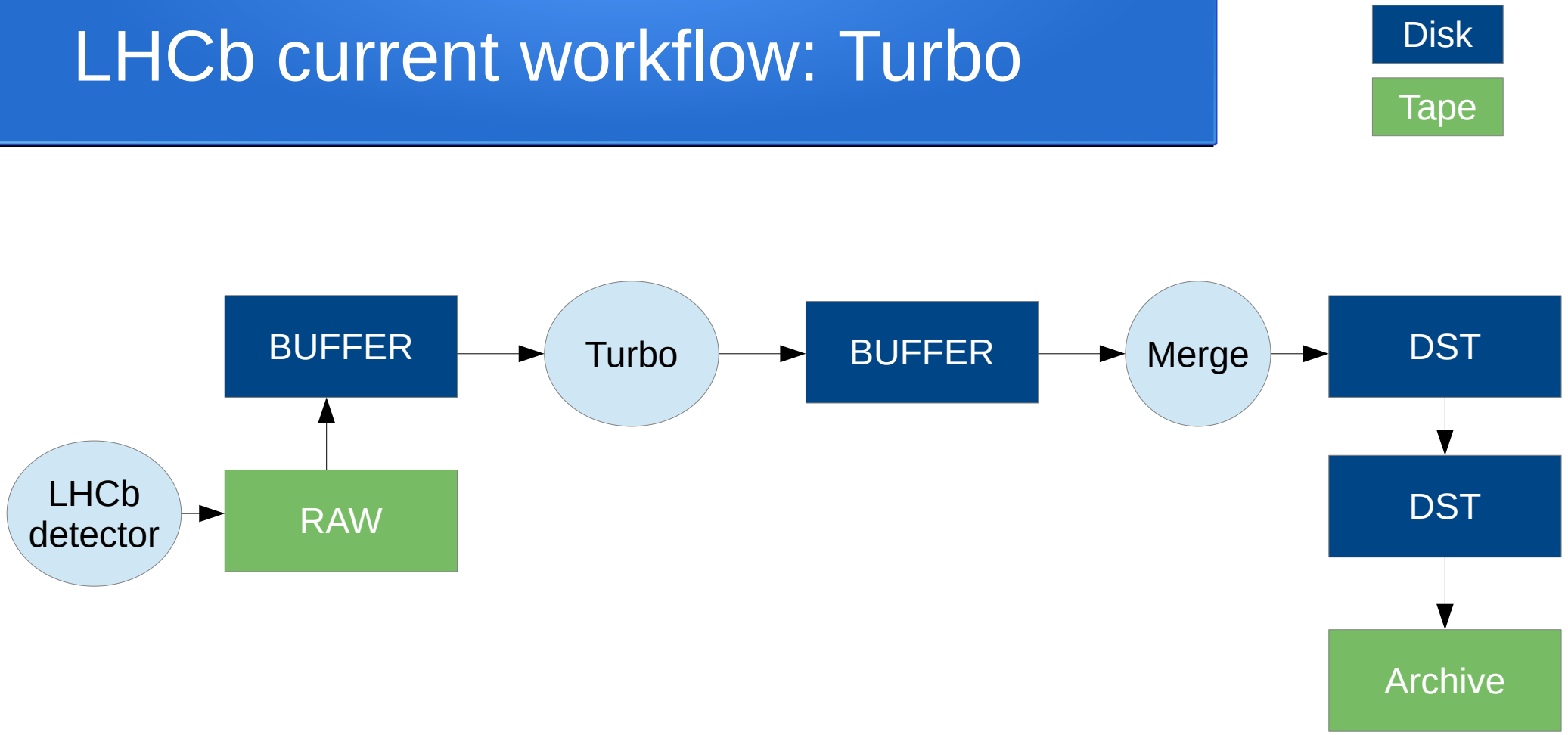
LHCb View
07/02/2020
Christophe Haen

LHCb workflows

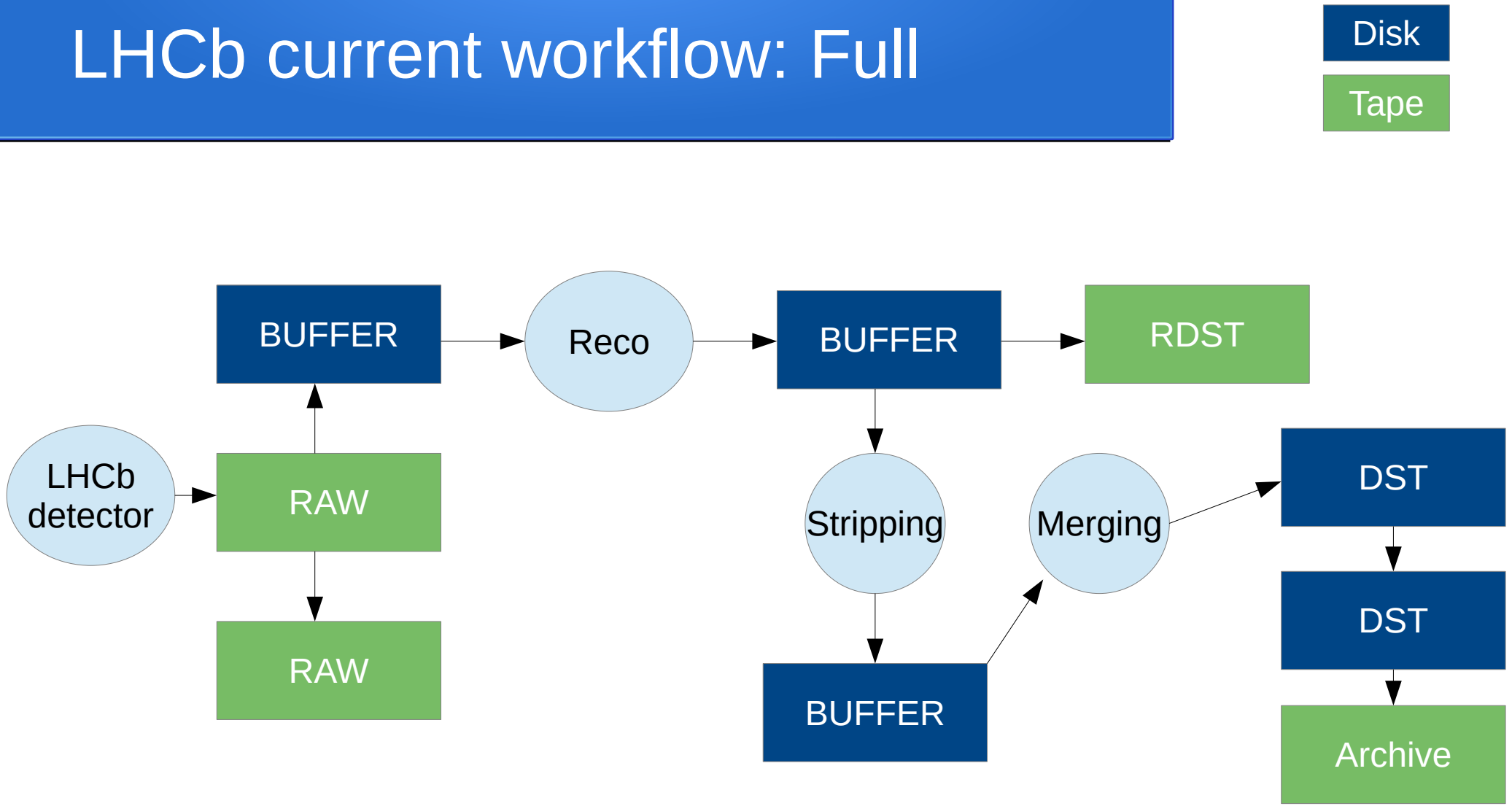
LHCb current workflow: MC



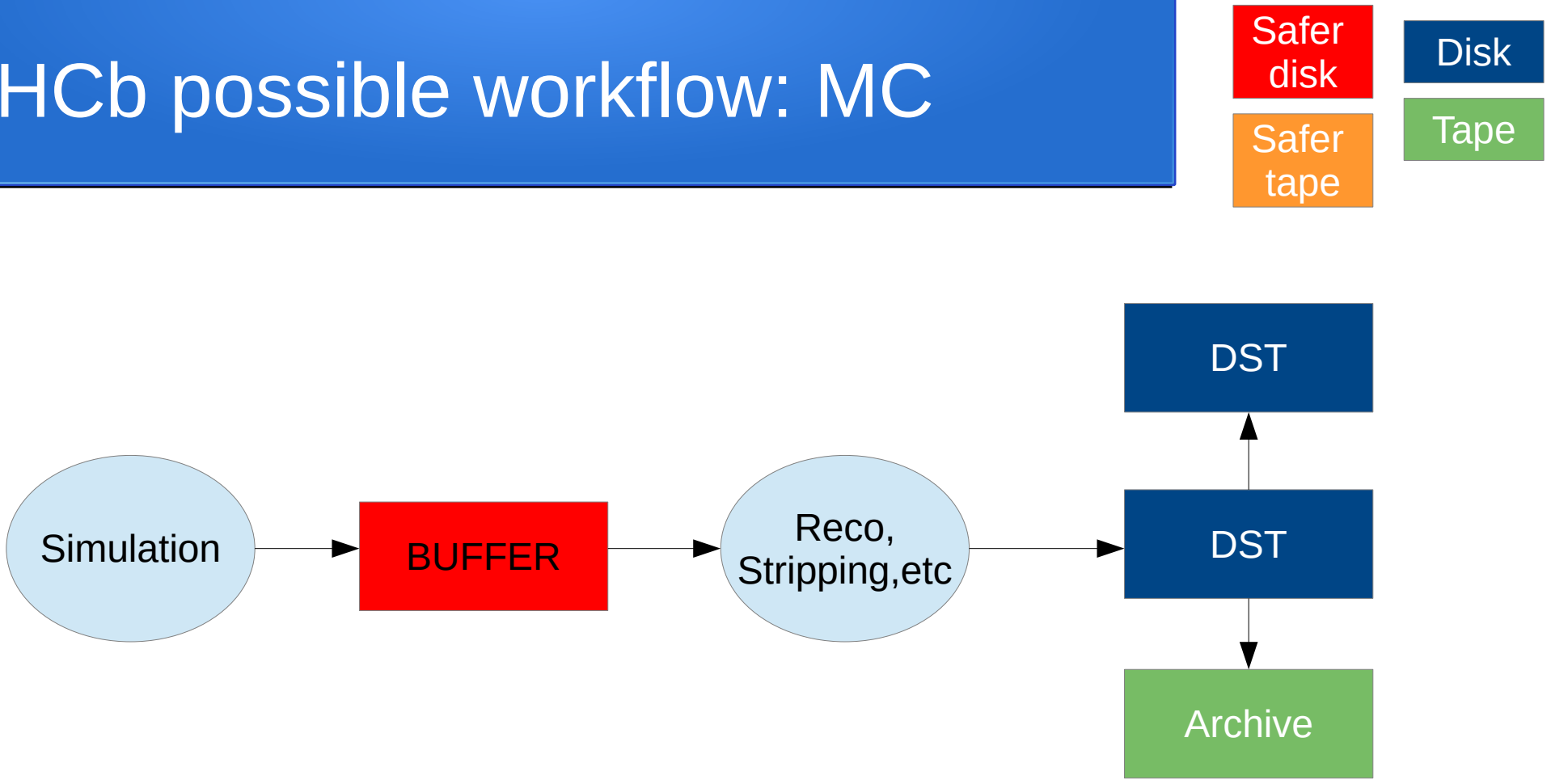
LHCb current workflow: Turbo



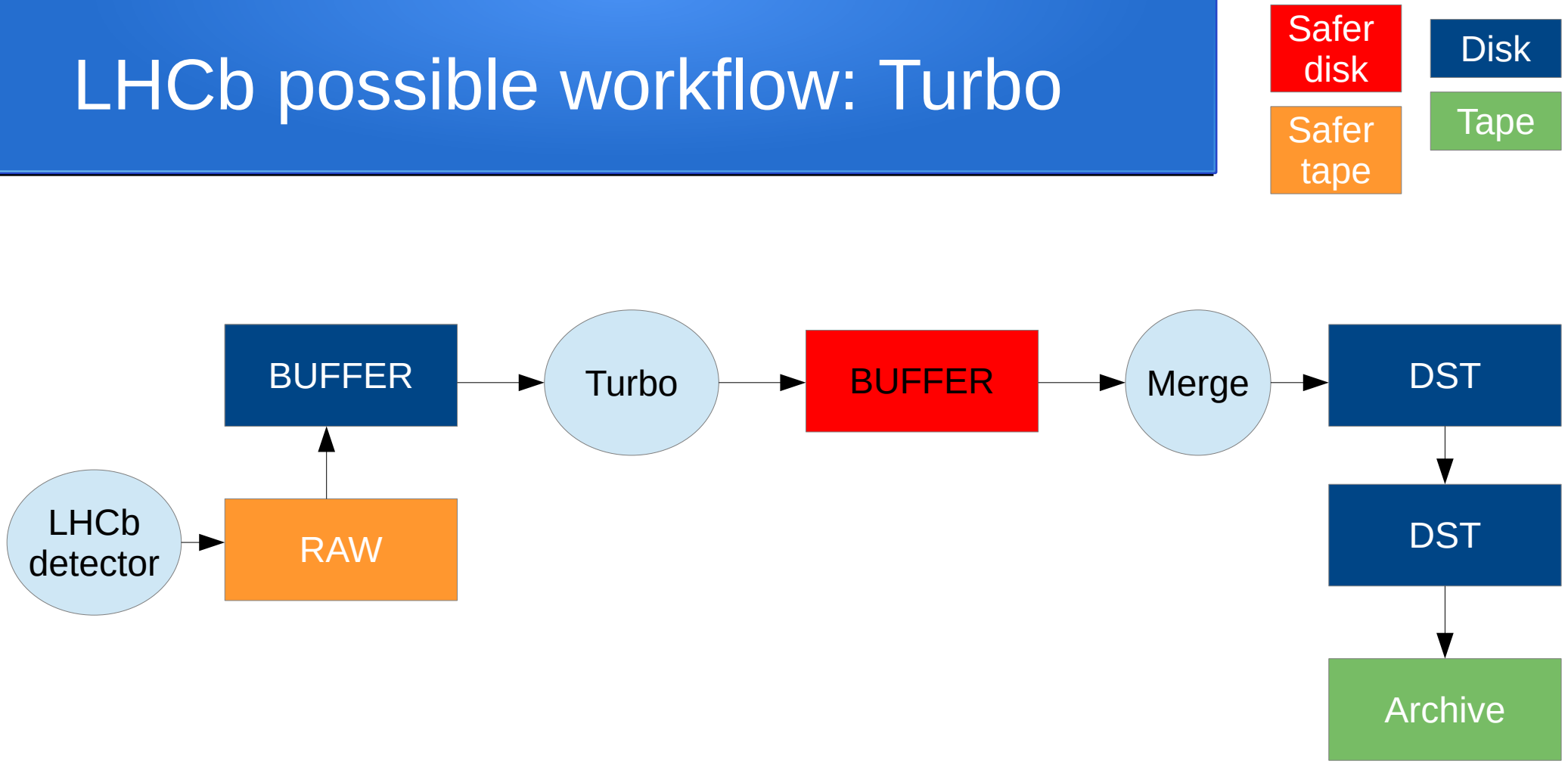
LHCb current workflow: Full



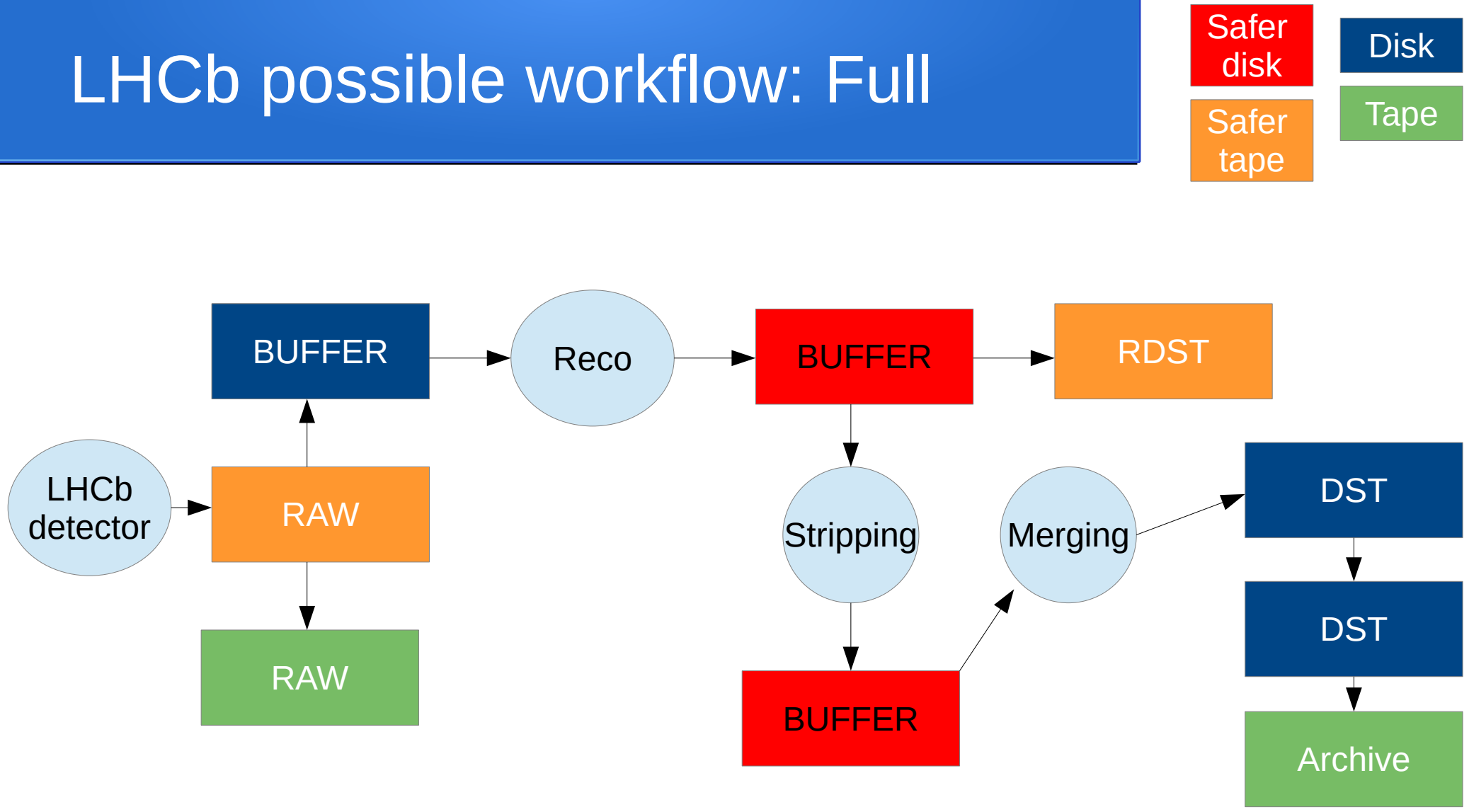
LHCb possible workflow: MC



LHCb possible workflow: Turbo



LHCb possible workflow: Full



General comments on the White paper

Kudos !!

- Well done
- Encompasses a lot of the aspects
- Opens the door to a lot of innovations, improvements (and cost savings…)
- Does not contain insane ideas that requires 50 FTE years to have a prototype

Caching

- Mostly out of QoS
- Very difficult topic
  - Tricky to make efficient (how many re-read over which period of time, file size, cache size, etc)
  - Settings are site dependent
  - Caches work at file level, VOs work at dataset level
  - If there is a cache in front of the storage, it has to be automatic, tuned by the site admins
- Probably no use in LHCb:
  - Not enough re-read
  - Applications not IO bound

Media & related projects

- Underlying & grouping media:
  - VO should not really care
  - Acknowledge that performance is not the only way to evaluate hardware (see Pledge comments)
- Related projects:
  - Is there going to be a “global white paper” ?
  - Data carousel: done in LHCb for restripping for 6 years

Storage abstraction

- “transitions are only possible by copying data”
  - Yes !
  - Aggregate Storage QoS classes: smells very much like SRM to me
- “Optimal block size, parallel streams, etc”
  - Can’t be expected from a VO for all sites: gfal2 negotiation ?
  - Great feature though (Feature request from 2016 https://its.cern.ch/jira/browse/DMC-905)
- “Geographically distributed storage”
  - Not clear to me
  - If abstracted by storage, should be invisible to the VO

QoS orchestration and the VO view

- “Manual approach is perhaps the simplest […] but will become unmanageable as the number of Storage QoS classes increases”
  - You certainly do not want that !!
  - Already no agreement on Information System (Glue, Glue2, GoCDB, CRIC, Carrier pigeon …)

QoS adoption strategies

- In general, example matches quite well with LHCb ideas (see previous slides)
  - Example 1 “on stage out” ~ safer disk
  - Example 3 “cloud storage” ~ safer tape
- Tactical use of low-durability systems : “automatic recovery from data loss”
  - Please, no usine a gaz
  - Rather standard file format to declare data loss (then consumed by DIRAC or Rucio)
- Optimising cost: “This saving may be passed on in terms of increased storage capacity”
  - :-)

Static and dynamic QoS

- Multiple QoS classes with single storage system
  - Please: no fancy parameters or weird calls
  - Pragmatic: expose the classes via different hostname or namespace
- “Automatic QoS transitions”
  - Mildly fond of the idea
  - LHCb wants dataset consistency across sites
  - If automatic transition, QoS has to be comparable, and invisible to the VO catalog:
    - From Hot (disk) to Colder (slow disk) is OK
    - From Hot (disk) to Cold (tape) is not OK
- “Similar principle could be implemented at the experiment level”
  - In place already for many years

Role of WLCG, Pledge & exposing cost

- Role of WLCG: “Validation of declared storage QoS classes”
  - No absolute values can realistically be assigned
  - Very site dependent
- Cost model: man power and expertise have enormous impact on the cost, not only hardware
- “Some way of compensating sites that have deployed alternative media”
  - Dangerous: you do not want the sites to be too fancy
  - Makes sense only if alternative media asked by the VO

A few more thoughts...

A few more thoughts

- Data locality is paramount, and LHCb will stick to it
  - Run the job where the data is
  - Most civil behavior on shared resources (network congestion, etc)
  - Still convince it is the most efficient way of running (no need for remote caches, transfers, etc)
- The performance of the transition between QoS classes is not mentioned, while very important
  - a.k.a staging
  - Especially if other VOs run rolling staging campaign like LHCb (e.g. Data Carousel)

A few more thoughts: be pragmatic

- Sites (and their QoS) have to be ~ uniform within their tier level
  - No T1 is special with respect to the others (not even CERN)
  - You do not bound an experiment workflow to a site hardware tender
    - A special hardware type at one site is useless in that respect
  - Same goes for T2s or T2Ds
- For it to work, QoS has to be either invisible, or pragmatic
  - Limited number of classes, manually manageable
  - No fancy url or parameters
  - The same classes throughout a tier level