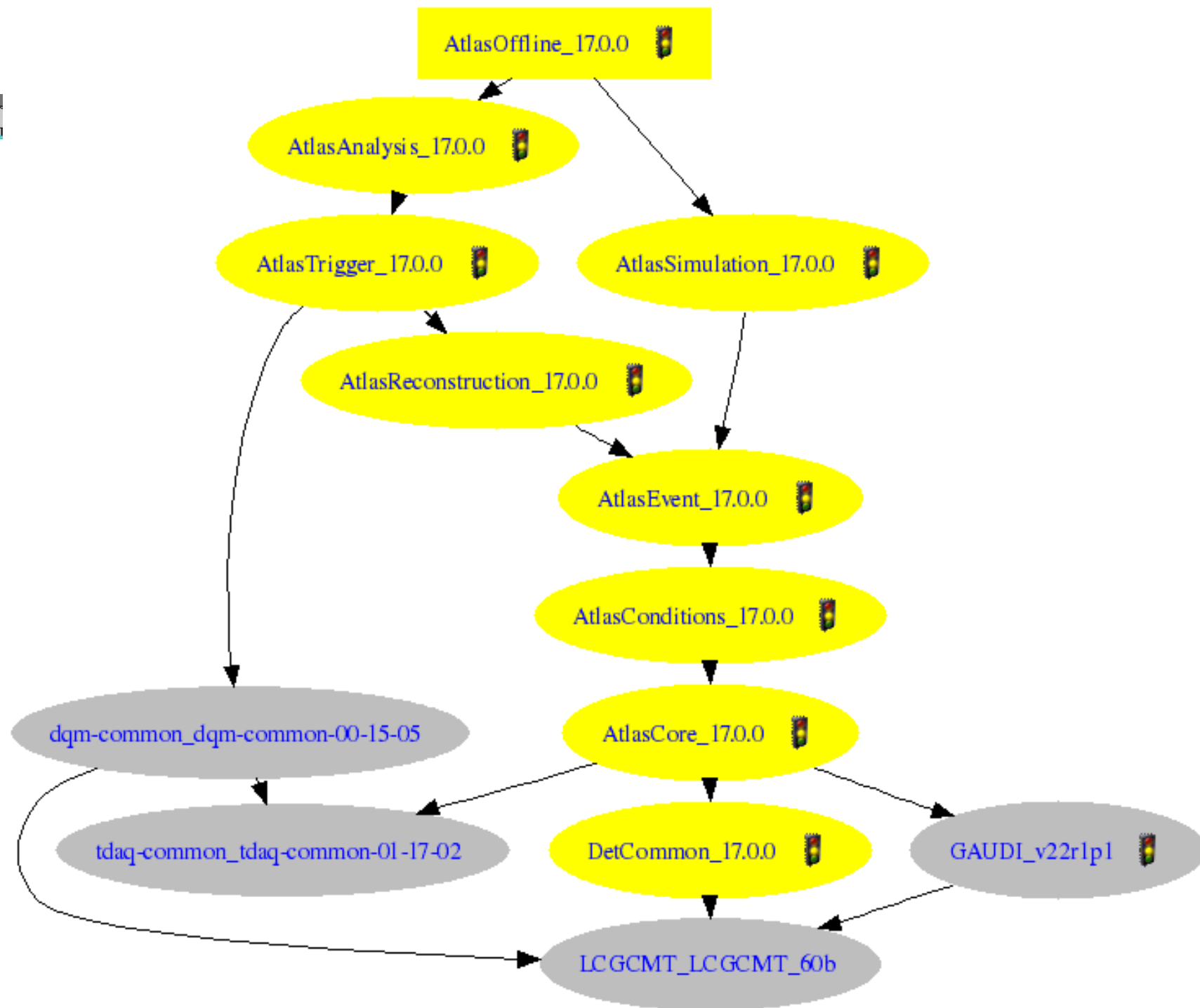


# **One AtlasRelease project?**

**David Rousseau (LAL-Orsay)**



**SIT meeting 6th April 2011**



# Original motivation



- ❑ Original idea was to build separately each projects so that the project on the top have stable based project.  
In practice:
  - This has never worked (attempted once)
  - Study I did 16 months ago only show only small reduction in rate of submission for tags for AtlasCore/AtlasConditions/AtlasEvent before closing a major release
  - We are not making any more significant/disruptive changes to core or EDM
  - We have even included Gaudi in dev build, in order to simplify bug fixes/small enhancement integration
  - We have nowadays other ways to offer stability : separate release branch (e.g. 16.1.X for HLT), caches, MIG
- ❑ So the original motivation is not valid anymore

# Project responsibility



- ❑ Well identified release coordinators for each projects, defining responsibilities
  - Rather clear for AtlasTrigger AtlasSimulation
  - But quite often a package is put in a project, based on the project hierarchy rather than the responsibility. E.g. :
    - Trigger tests in AtlasOffline
    - Monitoring code in AtlasAnalysis
  - In fact an \*application\* : (digi/simu/reco) is very likely to span several projects (also run time dependency violates project dependency)
- ❑ No flexibility : e.g. the simulation project encompasses AtlfastII (not in AtlasSimu), but not generators (in AtlasSimu)
- ❑ Project often duplicates svn container responsibility, sometimes across projects
- ❑ There must be ways to define responsibility across svn containers without defining separate build
  - E.g. a revamp of TagApproval page in TC, revamp of NICOS web pages

# ATN test availability



- ❑ NICOS build project one by one.
  - Start running ATN tests on project N as soon as available (in parallel start building project N+1)
    - =>ATN tests of low level project available early
    - ATN test available on the web when last test of this project is finished
  
- ❑ If we had only one project:
  - Tests would be available late (i.e. lunch time for regular nightly).
  - Possible solutions:
    - Release build could be faster (single makefile, incremental build,CVMFS...)
    - Tests available on the web as soon as they finished
    - Tests could start as soon as the packages they need is available. But run time dependency is not expressed in any way. Would have to have a way to derive it automatically (tricky)

# PATH lengths



- ❑ Reason to suspect that long PATH and LD\_LIBRARY\_PATH has an impact on wall clock performance, through afs latency
  - E.g. strace ls causes 350 "stat"
  - All the paths are scanned systematically when looking for an executable or library
- ❑ Ad hoc solution experimented by Sébastien: "slimfat" is scanning all the PATH and LD\_LIBRARY\_PATH and populates soft links in the first InstallArea
  - Technically works
  - Quite some work still to do performance test and make it usable by default in production
  - Volunteer welcome...

# Another project hassle



- Pick a random package, pick a random developer:
  - Ask him in which project is this package? My guess is that he would give the right answer in <20% of the cases
  - Now ask an expert ? I'm pretty sure hardly any would reach 50%
  - =>we all lose a lot of time searching for packages either in TC, either on afs

# Proposal



- ❑ Explore “slimfat”, maybe even bring it into production (at least explore performances implications of the reduction of PATH lengths)
- ❑ Explore lumping \*all\* atlas projects into one AtlasRelease:
  - Keep Gaudi separate
  - Of course keep 4 and 5 digits cache mechanism
  - Can be done fairly easily in a MIG
  - Explore all possible issues (as in previous slides, but probably others)