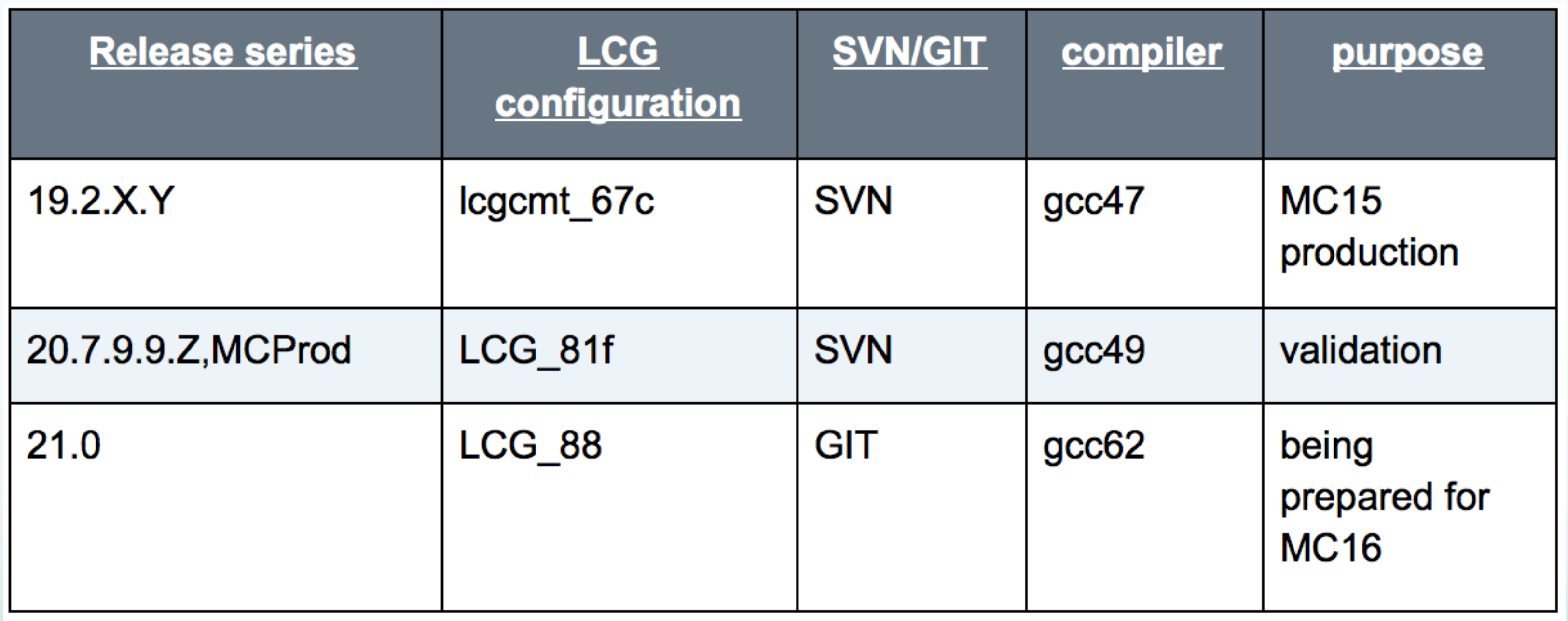
Current model and future requirements

Presented by
Emil.Obreshkov@cern.ch

LCG releases in use (1/2)

- Validation of MC software (Sherpa2.2.5, Herwig7.1.3) in some old releases (SVN & CMT).



Release series	LCG configuration	SVN/GIT	compiler	purpose
19.2.X.Y	lcgcmt_67c	SVN	gcc47	MC15 production
20.7.9.9.Z,MCProd	LCG_81f	SVN	gcc49	validation
21.0	LCG_88	GIT	gcc62	being prepared for MC16

- 67c and 81f LCG releases still used from AFS during our nightly builds, aim is to move away from that at the end of the year, depending on validation success and issues.
- Move MC production to 21.X (using LCG_88 from cvmfs).

LCG releases in use (2/2)

- Most development and production moved to Git, GitLab and CMake, CPack (for packaging rpms).
- Currently active git branches:
  - ...... see next slide (could not fit well here)

Branch	Purpose	Project, Release, LCG version
21.0	Tier0 and MC16e	Athena, 21.0.X, LCG_88
21.0-mc16a	MC16a bugfixes	Athena,21.0.20.Y, LCG_88
21.0-mc16d	MC16d bugfixes	Athena, 21.0.54.Y, LCG_88
21.0-TrigMC	Ad-hoc MC production with custom / updated trigger menu	Athena, 21.5.X, LCG_88
21.1	HLT and P1 monitoring	AthenaP1, 21.1.X, LCG_87
21.1-dev	HLT and P1 Run2 Developments	AthenaP1,NoProd, LCG_87
21.2	Derivations and Analysis	AthDerivations + (Ath)AnalysisBase, 21.2.X, LCG_88, LCG_93
21.3	MC18 Simulation	Athena + AthSimulation, 21.3.X, LCG_88
21.9	Upgrade Phase-2 developments	Athena + AthSimulation, 21.9.X, LCG_88
master	AthenaMT development	Athena, 22.0.X, LCG_93a

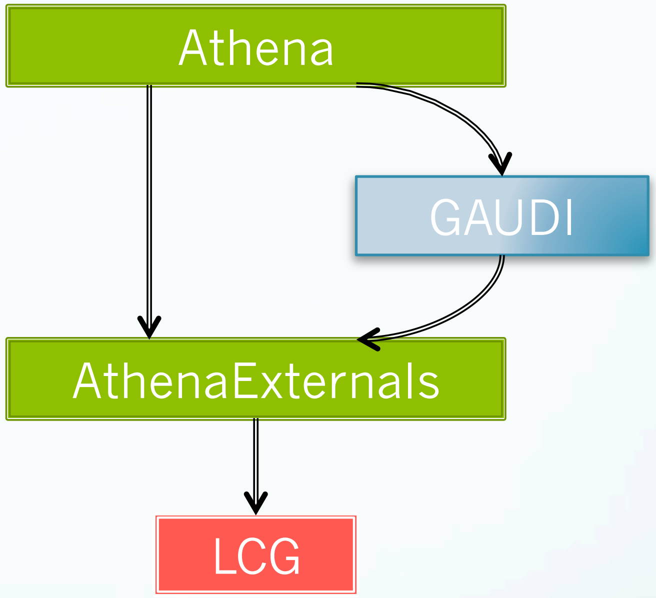
ATLAS sw projects

- Rich ecosystem of different projects to build in different branches:
  - Athena - production software (Event Generation, Digitization, Simulation, Reconstruction, Analysis) run at Tier0.
  - AnalysisBase, AnalysisTop, AthAnalysis*, AthDerivation* - Analysis Software Group (ASG) projects.
  - AthDataQuality* - Data Quality project.
  - AthSimulation* - Simulation project for G4 simulation.
  - AthenaP1 - ATLAS Trigger software project for Point1.

* AthXYZ – these projects are subset of full Athena project.

ATLAS sw projects

- All projects rely on "external project" serving as the basis and containing number of external packages:
  - AnalysisBaseExternals, AthAnalysisExternals, AthDerivationExternals, AthSimulationExternals, AthenaExternals
  - 92 LCG external packages in total used/required by these projects.



Prebuilt or from source

- Some ATLAS projects build/work independently from LCG, set up to look for the externals using Cmake search mechanism:
  - taking externals from the system if present (Boost,ROOT etc.)
  - If not in the system build them as part of the project
    - if ATLAS_BUILD_ROOT (or ATLAS_BUILD_BOOST etc) variable is set to TRUE - the package will be built, responsibility of the project to set that.
  - Done for analysis releases and considered for few other small projects.
- Other projects take externals binaries from LCG prepared releases.

Specific externals

- Most ATLAS specific externals (12) are build from source stored on web accessible location.
- Few ATLAS specific externals prebuild and directly used (Powheg), rpms provided for them for distributions.
- Athena and AthenaP1 project are the major clients of LCG build software using 92 externals provided by LCG release.
- We depend on LCG externals to be provided on /cvmfs/sft.cern.ch when we build our projects
- Our installations and distributions rely on the existence of rpms for LCG externals.
- LCG nightly builds do not provide rpms – had to make specific fixes to be able to have our nightlies against LCG nightly
- Occasionally we encounter problems that come from the packaging (rpms) and are therefore not test-able in the nightlies:
  - Can LCG release installations on cvmfs be done from the rpms ?

Locating/using externals

- For every LCG external we use we have our own Find module under https://gitlab.cern.ch/atlas/atlasexternals/tree/master/Build/AtlasLCG/modules.
- Our specific externals Find<Package>.cmake modules under https://gitlab.cern.ch/atlas/atlasexternals/tree/master/Externals/<Package>

Specific usage (1/5)

- Currently use dev3 nightly slot and many many LCG releases (we build many release for many groups and few years old releases are still being used).
- As it is at the moment - our central release builds procedures do not benefit from LCG Docker builds. Individual users and groups might if they want.
- Setup of validation tests within LCGtest – advertised within our core software group.
  - Not many people who are available to look into the future coming LCG releases.

Specific usage (2/5)

- Our builds pick externals from cvmfs and we rely heavily on cvmfs infrastructure to deploy all our nightly builds and releases.
- All Grid sites have mounted /cvmfs/atlas.cern.ch , /cvmfs/atlas-nightlies.cern.ch, /cvmfs/atlas-condb.cern.ch/ and /cvmfs/sft.cern.ch .
- LCG releases, through the /cvmfs/sft.cern.ch/ repo are also actively utilized by users via lcgenv tool to setup individual software (eg pyanalysis etc):
  - Very pleased by the support by Genser developers for lcgenv - fast and attentive.
- ATLAS ML group using the provided LCG views.
- http://lcgpackages.web.cern.ch/lcgpackages/rpms/ - used to create standalone packages and dependencies for installation in ATLASLocalRootBase (cvmfs) - allows finer grain control and setting only few specific versions.

Specific usage (3/5)

- Few HPC sites run ATLAS software (event generation) and usually cvmfs is not available there – installations there depend on LCG rpms.
  - Importance of HPC is likely to increase in the future.
- Currently there are very promising tests to run ATLAS containers (Docker and Singularity) with ATLAS specific code including LCG externals. That is ongoing on some Grid and HPC sites.
  - https://gitlab.cern.ch/atlas-sit/docker
  - use `yum install` to create our Docker images - rely on RPMs even for this.
  - https://hub.docker.com/u/atlas_ - hosting our images.
- We use EOS to store our release and nightly rpms.

Specific usage (4/5)

- Build systems: 1 Jenkins server for regular full nightly builds and 1 Jenkins server for CI builds and tests (testing individual MR).
- Using build pool of ~60 dedicated machines (highly restricted access). Requirements: at least 8core with 2GB/RAM per core and big enough local disk (~400GB to fit system, caches, builds). IO performance very important for our builds.

Specific usage (5/5)

- We have developed very specific Jenkins configurations adopted to our specific needs (number of git branches, number of projects build per branch, notification schemes, required and optional tests, sweep of changes between branches etc).
  - Plan is to automate further the Jenkins systems and polish it – some parts can be done better in order to have faster builds, and more reliable systems (Git, GitLab, CI specific scripts, EOS) and lessen human interventions.

- We have high hopes that GitLab will introduce CI build pipelines on MR - currently this is available only after accepting the MR. If that is available we might not need Jenkins for CI (and possibly the nightly builds too).

Expectations (1/4)

- Stable and reliable LCG release externals:
  - proper versioning
  - dependencies
  - packaging as rpms (and tested)
- Make sure already built rpms and installations do not change in the future.
  - If there is a change - new versioning should be set, dependencies (re)checked and new rpms provided
  - If old rpms are rebuild/changed that will hit us badly in some places
- <u>Proper RPMs content with dependencies are very important for us!</u>

Expectations (2/4)

- Would be very helpful if:
  - there is a page with reported changes or issues for given LCG release and which ones are fixed.
  - during urgent requests for LCG patch to be pointed out to that information, cross check if we want any of the existing fixes.
  - better usage of Jira for tracking LCG externals related matters (issues, problems, questions etc.)

Expectations (3/4)

- Compilers:
  - gcc62 at the moment used in our production releases.
  - gcc7 in testing.
  - Clang – few people interested in builds with it.
- Platforms
  - SLC6 production.
  - CC7 testing.
  - MAC for ASG releases in testing; could we have LCG releases for MAC OSX ?
    - try to figure out which sw is supported for MAC OSX ?
  - ARM - several groups within ATLAS expressed desire to have ARM builds.

Expectations (4/4)

- As it functions at the moment LIM (Librarians and Integrators Meeting) is enough for ATLAS.
- ATLAS is a huge collaboration with thousands of existing users and many (~hundred) new people joining each year.
- Any direct requests from ATLAS users to SFT/LCG should be checked with ATLAS software experts attending the LIM meetings:
  - To have global overview and define priorities otherwise you might end up satisfying every request which might not be in best interest for ATLAS or other experiments.
  - project-lcg-app-lim can be used for that.

Packaging system

- Regarding new packaging system - at the moment ATLAS is happy with the current packaging system in place (CPack) which is tightly coupled with the build tool we use CMake.
- If SFT/LCG decides to adopt a new packaging system - as long as it provides rpms with proper dependencies we will be fine!

ATLAS LCG effort

- Below statement is true and not known to many people and because of all the previous information presented I want to point it out !
- Since years we have officially 10% FTE dedicated to the externals from the Management – a new person again just about to start joining LIM. The ATLAS people present at LIM are there on their best effort bases trying to cover for the lack of more than 10% FTE support.

Thank you !

- Thanks greatly to the SFT group and the LCG release builders for the continued great support over the years.
  - In general we have been getting good and timely response and we understand that sometimes there are infrastructure glitches and issues but still there are places where things can be improved (see expectations again – slide 15 in particular).

Questions ?