Table of Contents

LHCb Core Software Programme of Work

This page documents the current work plan of the LHCb Core Software team. It is based on discussions that took place at the programme of work meeting on 11th and 12th January 2012. Tasks are currently listed here but should eventually be moved to Savannah to track their status.Tasks marked as urgent should be completed by the end of March 2012.

Software Development Infrastructure

Development tools

The recommended IDE is Eclipse. Specific Eclipse plugins for LHCb have been developed. Note however that Eclipse is rather heavyweight, so it should not be the only supported option. Support for Emacs, and in particular the LHCb Emacs templates, must continue. The following tasks have been identified

- Increase user awareness of Eclipse
    - Identify early adopters and testers, who can provide feedback on directions for development
    - Organise a specific tutorial
    - Provide examples for debugging
    - Improve documentation
- Eclipse integration with SVN
- Continue maintenance of Emacs templates
- Integrate Emacs with SVN, to use it as external editor instead of pico
- Create a Savannah portal for documentation tasks

Software Build, Release, Deployment

This is a major area of activity, involving the reorganisation of procedures and infrastructure for software build, release, distribution, removal. It is coupled also to the introduction of CMake, and needs futher discussion and design work.

Nightlies

The LHCb nightlies infrastructure has diverged from the PH-SFT version. PH-SFT group are doing exploratory work on a new version, for which we should provide input and feedback. We should aim to move to this new version as soon as it becomes available, but this will not be for several months. In the meantime we should implement urgent changes in the existing system.

Deployment

The software release procedure should be made independent of AFS: we should use for the releases software that is built and tested in the nightlies, creating the tarballs in the nightlies for installation in the release areas. The primary shared release area should be /cvmfs, but a shared deployment in AFS is also needed for sites that do not have cvmfs. The structure of the AFS and cvmfs deployment areas should be the same. A decision is needed on whether to pick up the external software directly from the LCG shared areas, or to make a copy. A distribution of the nightly builds in cvmfs would also be helpful, as would the possibility for users to trigger their own builds to allow a faster turnaround of continuous integration. The installation tar balls should reside in a single web accessible repository that can act also as long term archive.

dirac-lhcb-add-software should be kept in step with supported platforms

Dependencies

Dependencies between projects and on external software need to be documented, e.g. to to facilitate the removal of obsolete software from the deployment areas. PH-SFT has a database containing information about LCGCMT releases. Maybe we need a proper packaging system for the LCGCMT externals, and install only tarballs of needed packages in a single install area (would solve the problem the the length of LD_LIBRARY_PATH). CMake/CPack may help with this, or we could implement a dependencies database.

Software documentation

The $LHCBDOC software documentation infrastructure should be consolidated. There is a plan to move this to Drupal.

Urgent tasks

- Produce design document for new release and distrubution infrastructure (Ben)
- Task 25664 (On pause until the end of Feb): Build tar balls in the nightlies, repository for tarballs (Ben)
- Task 25589 (In Progress - CVMFS can now be used by touching ${HOME}/.devLHCBuseCVMFS): Use /cvmfs as default shared area in CERN login (Ben)
- Task 25580: Update dirac-lhcb-add-software to remove slc4 platforms and add gcc46 when available (FS)

Tracking/Management Tools

Tag Collector

The tag collector is now an excellent tool for release management and documentation which is beyond its original intended functionality, it should be redesigned in view of its current usage (task 25633). Support of branching is needed, and integration with the nightlies system.

SVN

There are communities looking at git as an alternative to SVN, and a Gaudi git repository is maintained, synchronised with SVN. There is however no compelling argument for moving LHCb software to git, we should concentrate on consolidating our use of SVN. Low level tools for branching are needed, and we need to identify someone for the maintenance of SVN, e.g. to provide pre-commit hooks

Bug and task tracking

PH-SFT is making a feasibility study or migration from Savannah to Jira. We have no compelling need to move to a new, better supported tool, wait and see.

Urgent tasks

- task 25633 Brainstorm (with PAC release managers) and collect requirements for tag collector (MCat)
- task 25633 Prepare new design for tag collector and related tools (MCat+MCl)
- Provide low level tools to support SVN branches (MCl)
- Identify maintainer for SVN repository (MCat)

Monitoring of the infrastructure

The status and logs of the LHCb software infrastructure applications (nightlies, tag collector) and CRON jobs should be monitored, ideally with existing tools. SLS could be good for this but we should first identify what we want to monitor and then choose the technology. Guidelines will then have to be provided to developers of services in order to instrument theit code for monitoring.

Urgent tasks

- Identify what we want to monitor (MCl)

Platforms and compilers

Externals

python 2.7 (LCGCMT_63)

Migration to 2.7 desirable since this will be last version in 2.x series, with improved support for Python 3. However, it was noted that when we do move, we should move also LHCbDirac, to avoid incompatible versions between the applications and Dirac. There are currently no plans to move Dirac to 2.7. So this must be prepared carefully. Noted also that, in order to bootstrap the system, LbScripts must work with the version of Python that comes with the system.

Root 6

Root is (finally) migrating away from CINT/Reflex. A prototype of the new system will be available in the 5.34 release in May 2012, and we **must** provide feedback on that during the summer, using specific tests that will be provided by the Root team. The plugin system should be added explicitly to the list of things to be supported by the new system. The first usable release will be in November 2012, we should plan to migrate during 2013.

gccxml

gccxml is a frozen product and we must look for alternatives. The long term solution (not before mid 2013) is to migrate to the genreflex replacement to be provided with Root 6. In the shorter term we can substitute it with a gcc plugin, currently under development. This is an interesting little project that can proceed at low priority, may become stop-gap solution if necessary.

Urgent tasks

- Prepare a python 2.7 deployment plan, including Dirac (JCl)

Platforms

SLC6

We must prepare the migration to SLC6, though it's not urgent, since the grid will most likely not move during 2012 data taking. For the time being it is sufficient to prepare the CMTCONFIG and then set up a nightly slot. But eventually we will have to prepare for the migration in a similar way to what was done for slc4->slc5, in particular Compat patches for running slc5 binaries on slc6 systems, and the possibility to compile slc5 binaries on slc6 (needed when preparing jobs on slc6 to run on an slc5 grid). It is not yet clear whether we will need 32-bit support on slc6, depends on whether 64-bit problems with Geant4 are resolved.

MacOSX

Will be supported by PH-SFT only after SLC6 support is in production. It is not clear what we gain, so this is postponed.

gcc 4.6.2

Will be supported for slc5 in Gaudi v23r0 release, and should be made the default if validation is successful - remembering that default at CERN must be runable on the Grid.

clang/LLVM

We should start using as an alternative compiler as soon as it becomes available in the nightlies (needs SLC6). We should give this higher priority than MacOSX

icc

We should pursue icc11 only if requested by the HLT for 2012 (unlikely). icc12 will come after SLC6. We do not require it for SLC5

Urgent tasks

- Validate gcc 4.6.2 on SLC5 and make default platform if successful (MCat)

Build Tools

CMake

We plan to move to CMake (and abandon CMT) during 2013. We should get ready fo this in 2012, and in particular be sure that the configuration contains enough information to build the run time environment.

Urgent tasks

- Complete the Gaudi CMake prototype in time for CHEP (MCl)

Run time

SetupProject

SetupProject should be redesigned as part of the move to CMake (2013). This can be prepared by making a review of use cases. In the mean time some small effort should be spent to improve the performance of the current implementation. If the new CMT version helps, we should use it.

CMT

There is a new version of CMT that is claimed to be faster. We should migrate to it if ths is true (and transparent) but not spend too much effort on it since we will abandon CMT in 2013.

- 20120118: Ben has verified that new CMT is only faster for build, not for environment, and has backward compatibility issues due to strict checking of version in requirements. We will not migrate to new CMT version.

Urgent tasks

- Add profiling hooks to SetupProject (MCl) DONE. Confirms that CMT is the main bottleneck (in fact, profiling hooks have been added to the LHCbScripts base class, so availble for all scripts)
- Add possibility to SetupProject to define any ROOT version even if not released with Gaudi
- Deploy new version of CMT if it improves performance (Ben) CANCELLED: new CMT is only faster for build, not for environment.

Persistency

I/O optimisation

There are two types of optimisation that should be investigated:

1. Optimisation of Root I/O parameters on existing event model ("Dynamic Root I/O" on Markus' slides)
2. Event model of what is written out to / read back from DST/MDST (includes, but not limited to, "event model primitives" on Markus' slides)

Metrics for the optimisation should be memory use, CPU use, latency with protocol access. At least two types of access pattern should be considered:

1. Sequential access to either the full event (reconstruction, stripping) or part of it (selection of fired lines from stripped (m)DST)
2. Sparse access to the full event (analysis of stripped (m)DST events that fired lines of interest)

One line of investigation involves greater use of packed containers and optimisation of the packed classes. Usage of such classes would be simplified if the DataOnDemand service configuration could be automated. Is there a role for Gaudi Converters?
OC 20120120 Probably not, without major backward compatibility issues, because with converters name of persistent class must be the same as (unpacked) transient class)

Another line of investigation would be to continue the work started last summer, to look at the granularity of the saved RawEvent and/or of packed classes, to avoid reading and unpacking more than necessary during bulk sequential processing of data. An interesting idea concerning MC truth would be to store in the event just sufficient information to regenerate the needed MC truth (e.g. HepMCEvent) on demand.

Urgent tasks

- Set up a working group to measure Root I/O performance of existing data (MF)
    - MF has defined two possible summer student projects in this area, see Savannah task 25527
    - Define metrics and use cases for which to tune (MCat+CJ)
- Investigate packing of Particles on mDST (OC).
- Review of (m)DST content (CJ with stripping and tracking coordinators)
    - Can any track states be dropped, can stripping reports be limited to fired lines, which MC truth is needed?

Conditions Database

Distribution

Various methods for distribution of the conditions database are under investigation: pull SQLite distribution with semi-automatic snapshots (pulled by Dirac via InstallProject), Oracle credentials in Dirac CS, Frontier interface to Oracle, SQLite distribution via CVMFS servers. The latter can work even if CVMFS client is not installed (http access). It was agreed to start with the first CVMFS option and study the second CVMFS option for later. The Frontier option will be considered as a fallback in case of problems with CVMFS (e.g. due to load), or if shorter latency is needed for the Online snapshots.

Data Quality flags

There is consensus that it is sufficient to skip events at the level of runs, and to postpone indefinitely the file level skipping, that has more complex implications for the framework and for the luminosity calculation

Tag compatibility database

Consensus that it is necessary to provide a database that describes compatibilities between tags, since current system is becoming unmanageable. Starting point should be a document to define what we mean by compatibility, that can serve as the starting point for designing the implementation

Upgrade

The requirements of upgrade simulations (new set of geometry and conditions) have not yet been defined, wait for input from upgrade software people. May need to rethink current approach to versions and tags.

COOL

We have no immediate need to use the new features of COOL but should plan to move to the new framework and migrate the data during the 2013 shutdown. We are not interested in CORAL server

Urgent tasks

- Deploy "pull" SQLite distribution with semi-automatic snapshots (Illya+Joel)
- Task 19007: Commission Run level skipping of events based on DQ information (MCl+Jaap)
- Provide document to define `tag compatibility` (MCat, Illya). A prototype is needed for CHEP2012

Software Optimization

A prerequisite for organised software optimisation is to define a set of rules and tools for producing reproducible results. We should investigate new compilers, but maybe not immediately:

- icc11 is available but icc12 is nearly there, so wait for it (icc11 could be interesting for HLT in 2012 but decided to concentrate on studying gcc46 and -O3 option)
- llvm not yet available in LCG

Further CPU optimsation should be profile driven, e.g. using Sascha's VTune Auditors, shoud be tried on HLT now. An interesting suggestion for the longer term is to provide abstractions that allow to transparently vectorise code such as loops.

Urgent tasks

- Ben will lead software optimisation activity, with help from MCat and PAC. Initial activities:
    - Set up working group to share experience, document tools, define strategies (with PAC)
    - Organise high level profiling of applications (Time per event, Memory, Event Size etc.) (with PAC)
    - Define and document rules for reproducible results (Files to download, job options etc.) (Ben)
- Deploy Sasha s profiling tool * Make public and document usage
- Concentrate on what could be deployed in HLT 2012 (Ben+Sascha)
    - Benchmark with gcc 4.6.2
    - Benchmark with -O3 + architecture specific compiler flags, how deep in stack do we go with -O3?
    - Profile with Sasha's tool

Parallelization and multi-core support

Summary slide from Pere's talk:

The existing Gaudi Parallel (both schemas) solution should be put into production (Already possible locally, on Grid we need a queue that supports multicore scheduling).

- Sound and effective solution for the next few years
- Full output validation is still missing
- Validation tools should be developed and be re-used later

LHCb should be one of the main players providing specific requirements, participating to the [common] project development and taking advantage of the new framework

- Clear benefits need to be demonstrated
- Would imply some re-engineering of parts of the experiment applications

Participation in a R&D program to evaluate existing technologies and development of partial prototypes of critical parts

Additional comments concerning concurrency are that we cannot enforce that TES objects are not modified (e.g. several algorithms add ExtraInfo to Track and may require its presence). On the other hand, we should look into parallelizing operations on containers (e.g. fit several tracks in parallel)

Urgent tasks

- Commission GaudiParallel for use in production (NR et al.)
    - Validation tools
    - Handling of counters
    - Interface to Dirac
- Participation in concurrency project (Coordinated by PM)
    - Define relevant demonstrators
    - Involve applications experts to identify constraints of real applications

Gaudi and LHCb frameworks

Release of Gaudi v23, with the refactoring of GaudiSvc, should be a top priority. It was decided however to not move DetDesc etc. to Gaudi since the main client (Daya Bay) is no longer interested in it. The introduction of backward incompatible changes is not considered a top priority but should be introduced gradually, switching off the backward compatibility flags one by one, in a dedicated nightly slot.

The work on reducing the usage of environment variables should proceed, it will take some time to get it right. Goal is that any file needed at run time should be installed, this will remove the need to distribute sources and result in a major speedup for InstallProject.The migration should not address all possible use cases, just the major ones, then fix what is broken.

The review of configurables should take place this year, with a view to a new implementation next year.

The external configuration service needs more thought and discussion. The ProdJob configurable could be worked on after 2012Q1.

Extension of QMTests using CDash should be part of the CMake migration plan.

Urgent tasks

- Release Gaudi v23r0 (MCl) with:
    - Refactoring of GaudiSvc
    - Migration to Gaudi of RootCnv

    - ♦ Support for gcc 4.6
    - ♦ LCGCMT 62
- Extension of DataOnDemandSvc to support mDST use cases (MCl)

Interface to Dirac

Philippe's slides contain many ideas. Short term priorities are listed as urgent tasks below

Concerning the interface between Dirac and the applications, thought needs to be given to the error handling, with more granular StatusCodes from Gaudi. Also, we need to find a mechanism for handling legacy applications and options in LHCbDirac. The ProdJob Configurable idea seems a good one.

Concerning the book-keeping and CondDB tags, the bkk should add the correct SimCond tag to generated options for MC datasets, but keep it commented so that default behaviour is not changed, and to avoid compatiblity issues when several tags are possible. Eventually (once CondDB tags compatibility database is available) something more clever could be done.

It was agreed not to implement any centralised book-keeping for user datasets. The dataset being in the book-keeping is considered a healthy defintion of public versus private data. The Ganga book-keeping of LFNs is considered adequate for user data.

Urgent tasks

- Put XmlSummary in production to replace CheckLogFile (MU)
    - ♦ Iterate with core software if necessary
- Document API between Applications and Dirac (FS+MCl)
    - ♦ Starting point implemention of ProdJob configurable
- Deploy persistency of XmlSummary (PAC)
    - ♦ What should go in bkk?
    - ♦ What should go to FSR? (i.e. has same lifetime as data)
    - ♦ What stays only in XmlSummary (available for how long?)
- Bkk to add CondDB tags to job options (ZM)

-- MarcoCattaneo - 27-Jan-2012

This topic: LHCb > ProgrammeOfWork
Topic revision: r16 - 2012-02-09 - MarcoClemencic