KLOE code migration from AIX to Linux flavour system

Wojciech Krzemień (NCBJ) &

Elena Perez del Rio(LNF)

Objectives

Establish a general strategy, estimate time and manpower for porting the KLOE code from the AIX platform to a Linux flavor system

Environment

	Current platform	Test Linux system
Operating System	**AIX 5.3.0.0**	**Ubuntu 14.04.1 LTS (trusty)**
Kernel	**64 bit**	**64 bit**
Fortran compiler	**IBM XL Fortran** Enterprise Edition V10.1 for AIX Version: 10.01.0000.0002	**gfortran** Also fort77 translator used **gcc version 4.8.2** (Ubuntu 4.8.2-19ubuntu1)
C/C++ compiler	**IBM XL C/C++** Enterprise Edition V8.0 for AIX Version: 08.00.0000.0011 Also prehistorical version of gcc available: **egcs-2.91.66**	**gcc version 4.8.2** (Ubuntu 4.8.2-19ubuntu1)
Makefile	**GNU Make 3.79** Also AIX make exists but outdated, not recommended	**GNU Make 3.81**
ROOT	**5.08**	**5.32**

List of packages

Online

- Common v1 , v1.5

  A set of useful functions for KLOE online (error handling ...).
- Circ v2 , v2.1 , v5

  KLOE circular buffer for passing events beween process.
- Vme v0 , v1

  A Common UNIX VME access interface.
- tclext v1 , v1.5

  Some general purpose extensions to the tcl/tk.
- SQLremote v1

  Remote access to the database.
- rock v0 , v1

  ROCK documentation.
- rockm v0 , v1

  ROCKM documentation.
- calb v0 , v1

  CALB documentation.

- dmap v0

  Detector map documentation.
- TL language 1.5

  A description language for easy implementation of Tcl commands.
- cmdsrv

  The KLOE command server and general purposes SNMP utilities.
- Process Template

  The Template for KLOE DAQ Processes.
- DAQ control (daqctl): v1 , v2

  The KLOE DAQ control libraries and applications.
- Trigger Supervisor v4.2

  Trigger Supervisor documentation.
- Farm software v1 , v2 , v3.6

  Software products to be run in the online farms.
- Spy buffer v1 , v1.5 , v3

  A library implementing spy buffer functions for event monitoring.
- cybos v0

  A library implementing YBOS access.
- kid v0

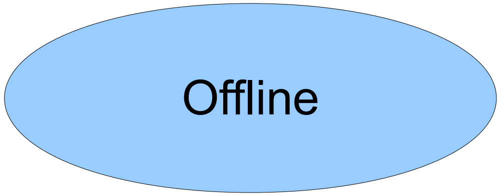
  KID - Kloe Integrated Dataflow

- Monitoring tools
  - Didone
  - Presenter
  - Browsers for data quality

List of packages

- Monte Carlo
  - GEANFI
- Reconstruction libraries emc,trk,ecl,...
- …

(see V. De Leo talk)



Code complexity

Example of code composition for MC

Not all code is in use → discriminating of the folder contents is necessary before porting

```
/kloe/software/off/geanfi
size:1.1 GB
sloccount:
Fortran:    1.100.768 (99.74%)
Csh:            2.674 (0.24%)
perl:             158 (0.01%)
```

Part of the Fortan code is in old-fashioned f77 style, which is hard to compile with f95 compilers e.g. usage of COMMON BLOCK

Compilation on AIX

- Usage of several tools (Unix Product Support (UPS)) to manage the compilation process:
  - **expand** – transforms .kloe files to .f fortran files; fortran program
  - **min_build** – generates makefiles; fortran program
  - several shell scripts
  - **kloe_build (**aka **rebuild)** – shell script that set up the environment, and perform the compilation by calling **expand** and **min_build**

UPS =package for the management of external software products in Unix environment; Fermilab product.

Porting to Linux

- The current version of UPS is outdated (~1998-2000),
- There are many hard_coded links and many not obvious dependencies at the level of shell scripts.
- Strategy options:

    1. Try to port the UPS-like scripts and programs and port packages themselves

    OR

    2. Abandon UPS, and concentrate on creating a new set of scripts under Linux

- Some of the work was done to migrate part of the packages to Linux-flavour in 2001.

Typical problems

- 32 vs 64 libraries
- Numerical types rounding, and size problems.
- Dependencies
- Compilers
- How to deal with compilation errors ?
- Outdated, not-supported libraries
- Different versions of the same code

Suggested strategy

Unless really needed don't change the existing code.

- Assign priorities to packages e.g. which should be ported as first, what backward compatibility is required

  e.g. MC should work “exactly” the same way as previously at the numerical level
- Define some tests that a package must fulfill
- Apply consistent code politics using some repository for package version (svn, git).

Use case I: ECL

- ECL and TRK offline libraries were tested under Linux

- gfortran and fort77 (translator) were used

- UPS tools were left out of the test → only *.f code were tested

- The outcome, together with the problems presented before in this talk, is the need of more time spend in decoupling dependencies

Use case II: Presenter

- Presenter – takes information from logs and monitoring histograms periodically and presents it graphically
- ”Easy” case
- Much easier to port cause written in C++
- Not much dependencies (basically only ROOT)
- Do not use UPS system to compile

Use case II: Presenter

- Presenter – takes information from logs and monitoring histograms periodically and presents it graphically
- ”Easy” case
- Much easier to port cause written in C++
- Not much dependencies (basically only ROOT)
- Do not use UPS system to compile

For an **experienced** person (with knowledge of AIX, and the Presenter itself): ~full two days of work

Use case II: Presenter

- Some problems encountered:
  - ROOT library version problems
  - Makefile incompatibilites (other set of steering flags needed)
  - Modern version of C++ compilers are much strict in flags and inclusion libraries order at the linking stage
  - Modern version of C++ compliers do not allow for undefined types of external variables
  - Modern version of C++ compliers are much strict in interedependencies of the included files

Manpower and time estimate

- A dedicated person with technical skills and experience is needed.
- Ideally:
  - familiar with Linux
  - Fortran, including f77 and differences
  - Shell scripting
  - Some C/C++
  - knowledge of Makefile preparation
  - some AIX experience (that is **unrealistic**, therefore some training time must be included)

Manpower and time estimate

- Assuming that we want to migrate X packages as first and a dedicated full-time person is there:
- the estimated time is at minimum **one year** (including training period)
- also some maintenance time, after the migration process, is necessary.
- No commitment of the collaboration for this project at the present