Porting CLEO software to Linux
V. Kuznetsov and C. D. Jones, Cornell University

The Linux operating system has become the platform of choice in the HEP community. The migration process from Tru64 and Solaris systems to Linux was a tremendous effort for developers and system administrators. The ultimate goal of such a transition was to administrate agreement between the final results of identical calculations on the different platforms. Apart from the fine tuning of the existing software the following issues were debated: choice of Linux distribution, development tools (compiler, debugger, profilers etc.), compatibility with third party software, and deployment strategy. As a result the CLEO collaboration ported their code to Linux with the ability to install and run on a variety of Linux distributions. Due to a flexible software model and deployment strategy various components of CLEO software can be installed on user systems, including personal laptops.

Migration from Unix to Linux
Linux strives to make migration as easy as possible by providing tools and capabilities similar to those found in other Unix systems.

- Most Linux distributions include gcc/gdb/glibc as base components of the system.
- Compiler and related tools have been designed as a replacement for vendor ones. As a good example, the gdb debugger can be run in dbx (Solaris) compatible mode.
- Linux is POSIX compliant which simplifies portability issues.

CLEO experience: from Tru64 and Solaris to Linux
- During migration from Tru64 and Solaris most of the problems were found with usage of STL containers.
- Explicit instantiation
- Almost all commercial products have been replaced by their open source counterparts, for example new EventStore, see C. Jones talk, based on open source software components (MySQL, Axis, etc.)
- Some commercial products were not suitable in CLEO environment: We were not able to integrate IBM/Rational PurifyPlus into CLEO environment since it insists on using its own.
- We found valgrind tools to be very good replacements for quantity/verify, which we use under Solaris.

Choice of Linux distribution
- There are ~400 Linux distributions worldwide (http://lwn.net)
- Often choice has been made based on site policies, hardware vendor support, usage of third party software for other projects.
- Make recommendations to your collaborators, as an example see http://www-clus0.fnal.gov/clusters/hardware/clues0_hardware.html.
- Due to self contained packaging system our software can be ported to any Linux distribution.
- We are not restricting our users in their choice, pick your favorite one and make sure that it satisfies minimal CLEO requirements:
  - gcc/g++/glibc++ version 3.2.2 and compatible version of glibc,
  - bash, tcsh, ksh
  - perl, python, am-utils
  - to install CLEO software RPM package should be present on user system
- Variety of Linux distributions helps to maintain portable code.
- To avoid conflicts, version incompatibilities, all third party software we rely on has been shipped as a part of CLEO RPMs.

CLEO Linux zoo:
- critical systems: RedHat Enterprise
- User desktops, farm nodes: RedHat 9
- MC farm (off-site): Fermi Linux LTS
- Debian, Fedora, SUSE were confirmed to work well.

Valgrind toolkit helps to debug applications on a Linux system:
- Memory checker identify memory leaks, stack overflow, mismatched use of malloc/new and free/delete
- Cache-miss profiler:
  - L1 instruction cache reads and misses
  - L2 data cache reads and read misses
  - writes and writes misses
- Heap profiler:

Code development
To help users to get started:
- We have code skeleton generators, helper scripts, wrappers
- We organize software workshops
- How to write programs in C++
- CLEO software structure
- How to generate MC, histograms, etc.
- Storage objects and batch queue usage
- Define user rules
- Provide user Makefile which can be used to build single package or entire release regardless of underlying platform.
- Provide variety of tools to debug your applications: memory checking, profile, report levels, etc.

On development side:
- To support different compilers/platforms, set of “bug flags” included into Experiment/Experiment.h
- Most of them are STL related
- CLEO uses dynamic loading rather than static linked executables
- 780 packages can be easily compiled on 500MHz/256MB node.
- Build is CPU bounded, not RAM
- Link to libraries your module depends on (~20-30 tils/module)
- NFS implementation under Linux is not yet mature:
  - Usage of local file system during linking step can speed up entire process by factor of 6

Code development under new OS requires adjustments. Having different compilers helps us develop portable code, find compiler and limitations.

<table>
<thead>
<tr>
<th>Tool/OS</th>
<th>OSF</th>
<th>Solaris</th>
<th>Linux</th>
</tr>
</thead>
<tbody>
<tr>
<td>compiler</td>
<td>cc</td>
<td>CCL77</td>
<td>g++/g77</td>
</tr>
<tr>
<td>debugger</td>
<td>ledbug</td>
<td>dbx</td>
<td>quantfruitify</td>
</tr>
<tr>
<td>Online bridge</td>
<td>MPP</td>
<td>Multilib</td>
<td>Minilib/Multilib</td>
</tr>
<tr>
<td>C++ compile flags</td>
<td>-O3 -Wall -nopg -nocoreloop</td>
<td>CC-Instances=global KPC-mt-O</td>
<td>g++ -O3 -PC</td>
</tr>
<tr>
<td>Fortran compile flags</td>
<td>-O3 -uKPC -mt</td>
<td>-O3 -uKPC -mmt -oautomatic -fint-local-zero</td>
<td></td>
</tr>
</tbody>
</table>

CLEO-like Linux node:
- /cleo3
- CLEO RPM database
- Event Store database
- other_sources
- Common
- Offline
- pk8, bin16, rel
- cache data rel
- valgrind Tcl Qt
- current dev
- dev1Current 20040823
- other_sources build16 bin include
- bin lib include
- Linux SUSE OS
- g++
Porting CLEO software to Linux
V. Kuznetsov and C. D. Jones, Cornell University

RPM system as a deployment tool for CLEO software
In CLEO we use RPM (RedHat Package Management) system as a deployment tool to port CLEO software off-site. The system is able to install/upgrade/verify/uninstall the necessary software components without particular knowledge of remote system configuration and user privileges.

- RPM is stand-alone open source software that can be installed on any Linux distribution; it is required to be installed on user node
- CLEO RPMs are transparent to Linux distribution of choice, i.e. no additional software is required
- CLEO RPMs are relocatable, i.e. users can install CLEO software anywhere on their system
- We use mixture of binary and source RPMs
- CLEO software is linked with hard coded path to the components to avoid users mistakes (no headaches with LD_LIBRARY_PATH)

- We provide installation script together with RPMs
- We keep separate RPM DB for CLEO packages without touching system one on RPM based systems
- We pack in binary RPM third party packages (Tcl, readline, etc.) w/ CLEO required C/C++ flags (e.g. -fexceptions)
- Formalize CLEO package dependencies
- Install necessary components on your system, for instance only MC generator or graphics packages

Software fine tuning
While porting CLEO software from OSF and Solaris platforms to Linux many problems have been found. They can be classified as follows:
- compiler/preprocessor/linker bugs
- OS specific bugs
- STL related bugs
- user specific bugs

In CLEO a set of C pre-processor bug flags and C pre-processor macros for STL containers and iterators has been used.
We found Linux port as painful as switching to different version of compiler on Solaris or Tru64 platforms. Currently only specific version of GNU compiler is used: version 3.2.2

Solaris vs Linux
A long saga of converting between two platforms helps to clean-up code, use standard (C++) and fix various bugs. Most of the problems found in Fortran code due to round off boundaries and conditions.

- Differences found to be consistent with round off errors
- 15-20% of low momentum p_t tracks differ in p_t by ~ 1%
- Systematic differences between Solaris and Linux are smaller than intrinsic systematic errors in MC.

Run CLEO software on laptop:
12 hours are required to compile and install 780 packages on average Linux/Intel/1GHz/256Mb laptop

Use desktop for development
All CLEO RPMs are produced on a single Linux build node for every stable release. Then they published on the Web for public download. http://www.lns.cornell.edu/~vk/cleo3_rpm/html

Install CLEO RPMs on your farm and convert it into MC production cluster
Minnesota CLEO-c MC farm is able to produce 12M/DD and 14.SM/Continuum MC events per day using 28 dual Xeon CPU 2.4 GHz nodes.
Multiple releases are installed using CLEO RPMs

System requirements to compile and run CLEO software
CPU 500MHz, 256Mb of RAM, 7Gb of disk space

Linux in CLEO:
- Linux replaced Tru64 for MC generation
- Solaris and Linux used for reconstruction
- Linux and Solaris serve on-line calibrations
- Farm nodes are equipped with Linux to serve analysis needs
- MySQL server for EventStore data management system
- File server
- MC farm