Virtualization and Multi-core in the Applications Area

Pere Mato (CERN)

LCG-MB Meeting,10 March 2009

Introduction

- Two work-packages on physics data analysis, simulation and computing retained from the R&D proposals in the White Paper under Theme 3
  - WP8 – Parallelization of software frameworks to exploit multi-core processors
  - WP9 – Portable analysis environment using virtualization technology
- Both work-packages started in January 2008 for a duration of 4 years
  - WP9 will be reviewed for continuation after 2 years
- Rather limited newly-funded resources
  - About 14 FTE-year Fellows over 3-4 years

Why these two R&D areas?

- Technologies with potential to be directly applicable to "real" needs of experiments in a short time scale
  - Common to all experiments independently of their software environment
  - Narrowed the scope to make it more effective
- Quite attractive new technologies
  - Attack young people to work in the projects
  - Experience useful in many other domains
- Already some interest in the community
  - Serious possibilities of collaboration

WP8 – Parallelization of Software Frameworks to exploit Multi-core Processors

HEP Software on multi-core

- The aim of the R&D project is to investigate novel software solutions to efficiently exploit the new multi-core architecture of modern computers in our HEP environment
- Activity divided in four “tracks”
  - Technology Tracking & Tools
  - System and core-lib optimization
  - Framework Parallelization
  - Algorithm Parallelization
- Coordination of activities already on-going in experiments, IT, other Labs, etc.

Summary of activity in 2008

- Collaboration established with experiments, OpenLab, Geant4 and ROOT
  - Close interaction with experiments (bi-weekly reports in AF)
  - Workshops each six months (first in April, second in October, next in spring 2009)
- Survey of HW and SW technologies
  - Target multi-core (8-16/box) in the short term, many-core (96+/box) in near future
  - Optimize use of CPU/Memory architecture
  - Exploit modern OS and compiler features (copy-on-write, MPI, OpenMT)
- Prototype solutions
  - In the experiments and common projects (ROOT, Geant4)
  - In the R&D project itself

Technology Trend (Intel & OpenLab)

- Moore's law still apply in foreseeable future
  - Challenge is not surface, is power!
- Multi-core micro-architecture
  - Evolving, do not expect major changes
- New memory layout and connections to cpu
  - New levels in hierarchy
- Hyper-threads
  - They are back!
- SIMD: Intel AVX
  - 1024 bit registers
- Mini-core
  - Not just GPU, full IA
  - 1024 threads in one box

Current measurements

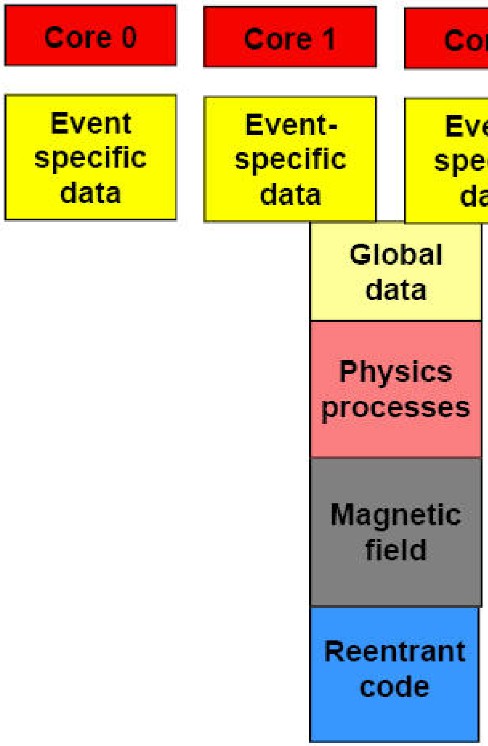
- HEP code does not exploit the power of current processors
  - One instruction per cycle at best, little or no use of SIMD, poor code locality, abuse of the heap
- Running N jobs on N=8 cores still efficient but:
  - Memory (and to less extent cpu cycles) wasted in non sharing
    - "static" conditions and geometry data, I/O buffers, network and disk resources
  - Caches (memory on CPU chip) wasted and trashed
    - Not locality of code and data
- This situation is already bad today, will become only worse in future architectures (either multi-thread or multi-process)

Framework Parallelization

- Objective
  - Investigate solutions to parallelize current LHC physics software at application framework level
  - Identify reusable design patterns and implementation technologies to achieve parallelization
  - produce prototypes
- Current Activities
  - Evaluate pro & cons of various alternatives
    - fork(), C++ objects in shared memory, threads
  - Dedicated workshop in October
    - Report by Atlas, LHCb, CMS, Root, Geant4
    - I/O issues
    - High granularity parallelization of Algorithms

Event parallelism

- Reconstruction Memory-Footprint shows large condition data
- How to share common data between different process?



→ multi-process vs multi-threaded

→ Read-only: Copy-on-write, Shared Libraries

→ Read-write: Shared Memory or files

Exploit Copy-on-Write

- Modern OS share read-only pages among processes dynamically
  - A memory page is copied and made private to a process only when modified
- Prototype in Atlas and LHCb (the latter using WP8 personnel)
  - Encouraging results as memory sharing is concerned (50% shared)
  - Concerns about I/O (need to merge output from multiple processes)

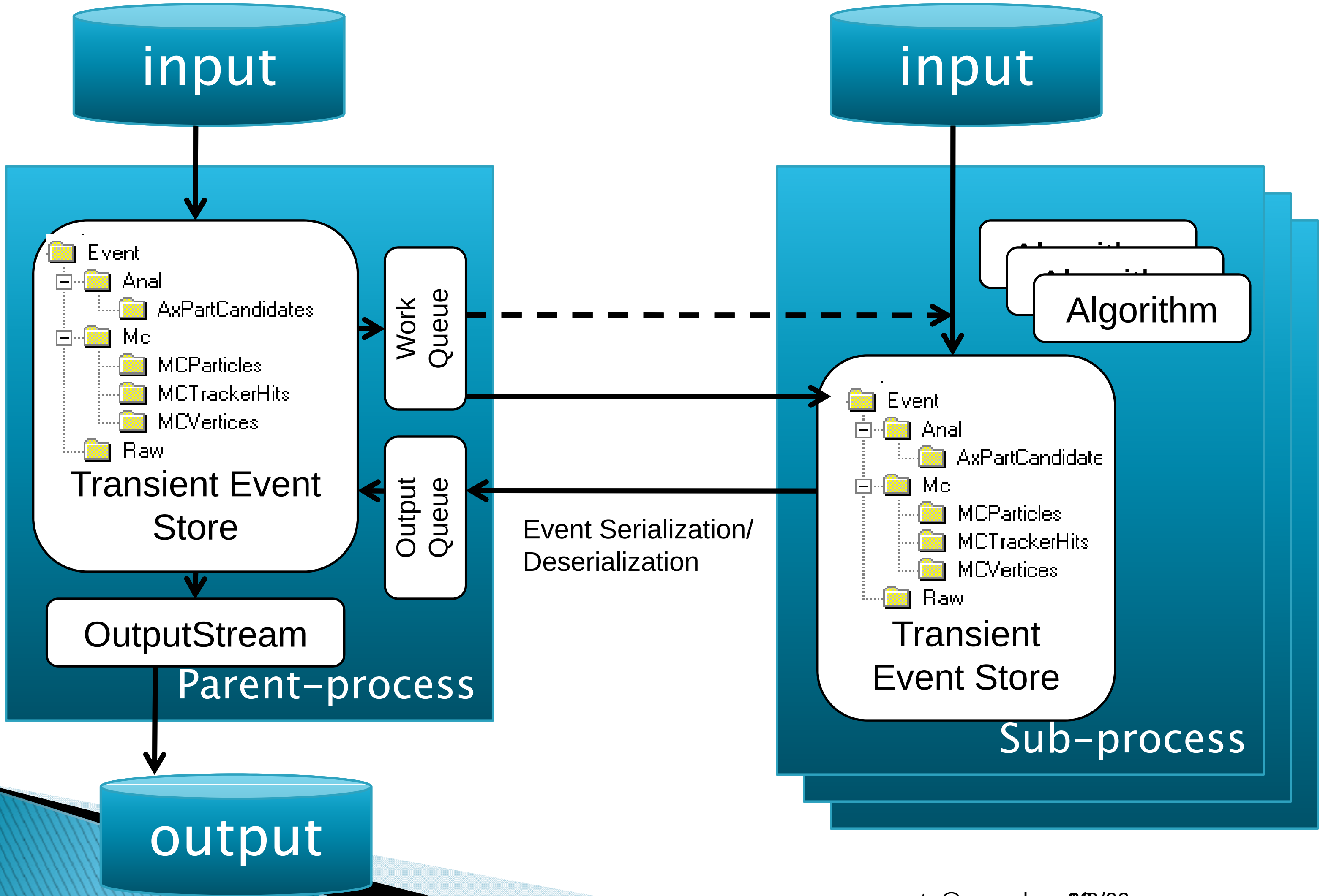
Memory (ATLAS)
One process: 700*MB VMem and 420MB RSS*
(before) evt 0: private: 004 MB | shared: 310 MB
(before) evt 1: private: 235 MB | shared: 265 MB
. . .
(before) evt50: private: 250 MB | shared: 263 MB

Exploit "Kernel Shared Memory"

- KSM is a linux driver that allows dynamically sharing identical memory pages between one or more processes.
  - It has been developed as a backend of KVM to help memory sharing between virtual machines on the same host.
  - KSM scans just memory that was registered with it.
- Test performed "retrofitting" TCMalloc with KSM
  - Just one single line of code added!
- CMS reconstruction of real data (Cosmics)
  - No code change
  - 400MB private data; 250MB shared data; 130MB shared code
- Similar results with ATLAS

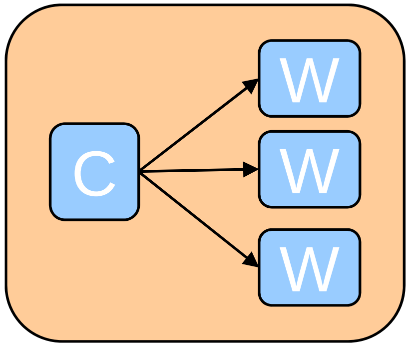
Handling Event Input/Output

Work initiated in the project (Eoin Smith)



PROOF Lite

- PROOF Lite is a realization of PROOF in 2 tiers
  - The client starts and controls directly the workers
  - Communication goes via UNIX sockets
- No need of daemons:
  - workers are started via a call to 'system' and call back the client to establish the connection
- Starts NCPU workers by default



(G. Ganis)

Multi-threaded Geant4

- Event-level parallelism: separate events on different threads
  - Increase sharing of memory between threads
- Phase I : code sharing, but no data sharing Done
- Phase II : sharing of geometry, materials, particles, production cuts Done, undergoing validation
- Phase III : sharing of data for EM processes Todo
- Phase IV : other physics processes Todo
- Phase V : new releases of sequential Geant4 drive corresponding multi-threaded releases Todo
  - Currently 10,000 lines of G4 modified automatically + 100 lines by hand

Gene Cooperman and Xin Dong (NEU Boston)

Algorithm Parallelization

- Objectives
  - Investigate solutions to parallelize algorithms used in current LHC physics application software
  - Identify reusable design patterns and implementation technologies to achieve effective high granularity parallelization
- Example: Parallel Minuit (A. Lazzaro, L. Moneta)
  - MINUIT2 is a C++ OO version of the original MINUIT
    - The most used algorithm in HEP community for Maximum Likelihood and $\chi 2$ fits (www.cern.ch/minuit)
    - Integrated into ROOT
  - Two strategies for the parallelization of the Minimization and NLL calculation:
    - Minimization or NLL calculation on the same multi-cores node (OpenMP, pthreads)
    - Minimization process on different nodes (MPI) and NLL calculation for each multi-cores node (OpenMP, pthreads)

Future Work

- Release production-grade parallel application at event level
  - Exploit copy-on-write (COW) in multi-processing (MP)
  - Develop affordable solution for sharing of the output file
  - Leverage G4 experience to explore multi-thread (MT) solutions
- Continue optimization of memory hierarchy usage
- Expand Minuit experience to other areas of “final” data analysis
- Explore new Frontier of parallel computing
  - Scaling to many-core processors (96-core processors foreseen for next year) will require innovative solutions

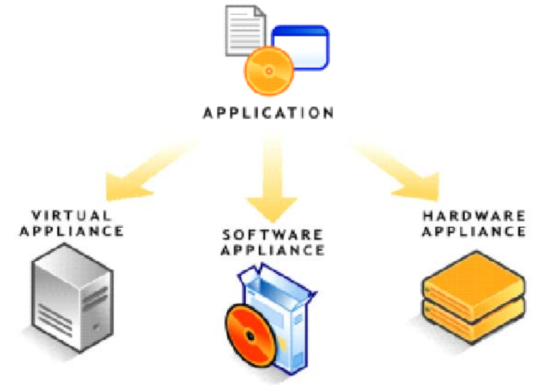
WP9 – Portable Analysis Environment using Virtualization Technology

Project Goal

- Provide a complete, portable and easy to configure user environment for developing and running LHC data analysis locally and on the Grid independent of physical software and hardware platform (Linux, Windows, MacOS)
  - Decouple application lifecycle from evolution of system infrastructure
  - Reduce effort to install, maintain and keep up to date the experiment software
  - Lower the cost of software development by reducing the number of compiler-platform combinations

Key Building Blocks

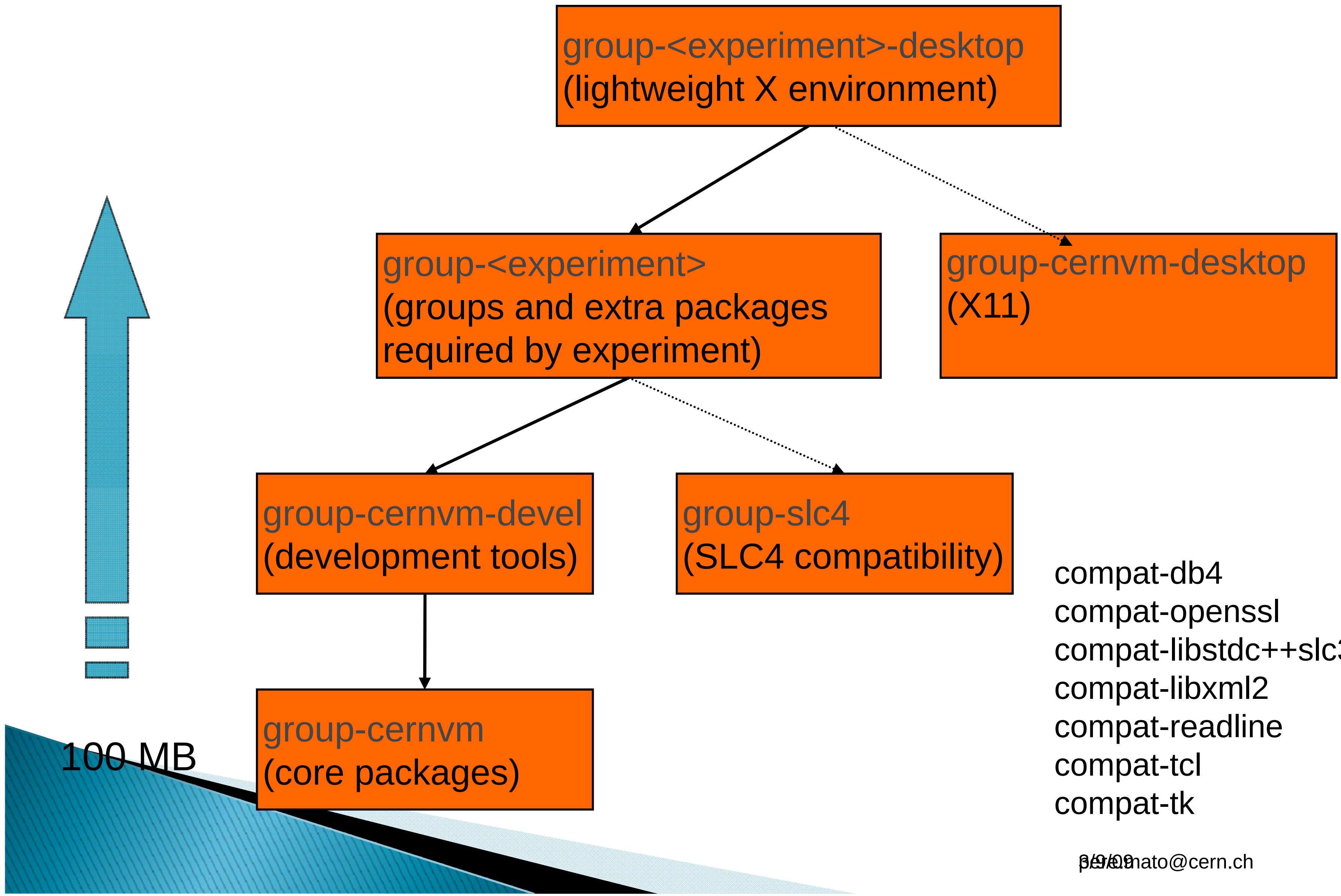
- rPath Linux 1 (www.rpath.org)
  - Slim Linux OS binary compatible with RH/SLC4
- rAA – rPath Linux Appliance Agent
  - Web user interface
  - XMLRPC API
- rBuilder
  - A tool to build VM images for various virtualization platforms
- CVMFS – CernVM file system
  - Read only file system optimized for software distribution
    - Aggressive caching
  - Operational in offline mode
    - For as long as you stay within the cache



Build types

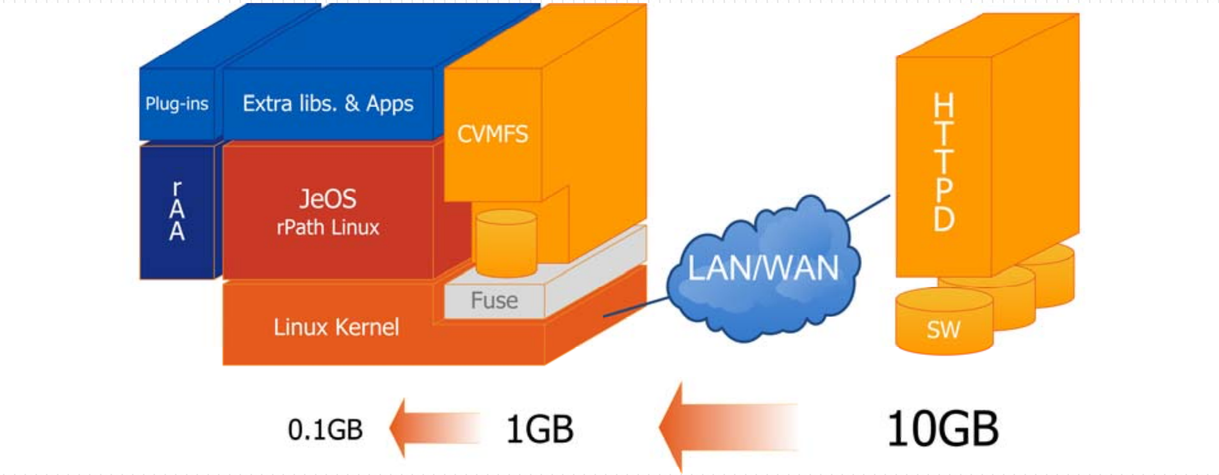
- Installable CD/DVD
- Stub Image
- Raw Filesystem Image
- Netboot Image
- Compressed Tar File
- Demo CD/DVD (Live CD/DVD)
- Raw Hard Disk Image
- Vmware ® Virtual Appliance
- Vmware ® ESX Server Virtual Appliance
- Microsoft ® VHD Virtual Apliance
- Xen Enterprise Virtual Appliance
- Virtual Iron Virtual Appliance
- Parallels Virtual Appliance
- Amazon Machine Image
- Update CD/DVD
- Appliance Installable ISO

CernVM Variations



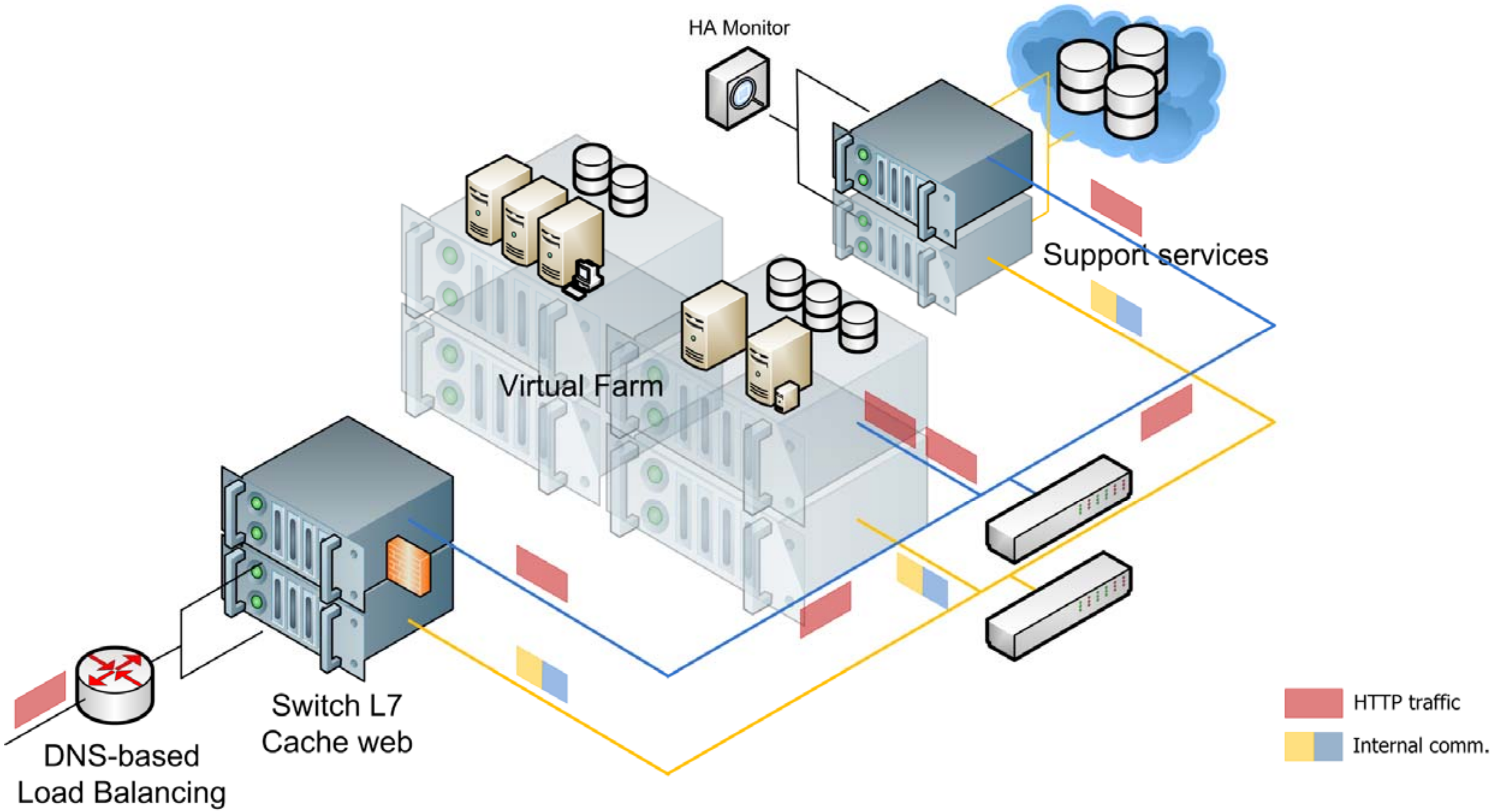
compat-db4
compat-openssl
compat-libstdc++slc3
compat-libxml2
compat-readline
compat-tcl
compat-tk

CVMFS



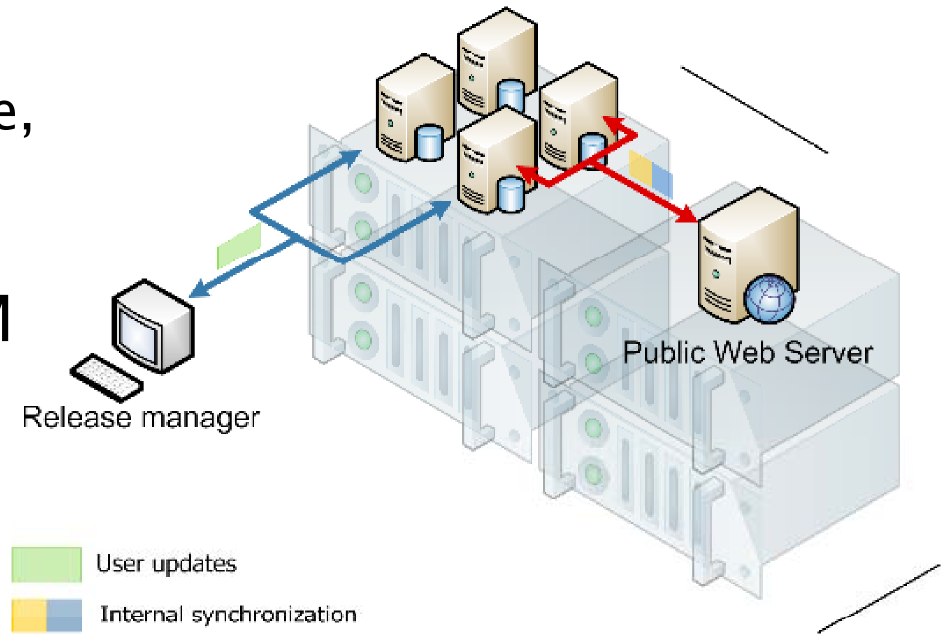
- CernVM File System (CVMFS) is derived from Parrot (http://www.cctools.org) and its GROW-FS code base and adapted to run as a FUSE kernel module adding extra features like:
  - possibility to use multiple file catalogues on the server side
  - transparent file compression under given size threshold
  - dynamical expansion of environment variables embedded in symbolic links

Scalable CVMFS Infrastructure



Publishing Releases

- Experiments publish new releases themselves
  - Installation done in a dedicated Virtual machine, which then synchronizes with Web Server
- Transparent to CernVM end-users
  - New versions appear in the 'local' file system



The CernVM Environment

- A complete Data Analysis environment is available for each Exp.
  - Code check-out, edition, compilation, local small test, debugging, …
  - Castor data files access, Grid submission, …
  - Event displays, interactive data analysis, …
- No user installation required !!
- Suspend/Resume capability !!

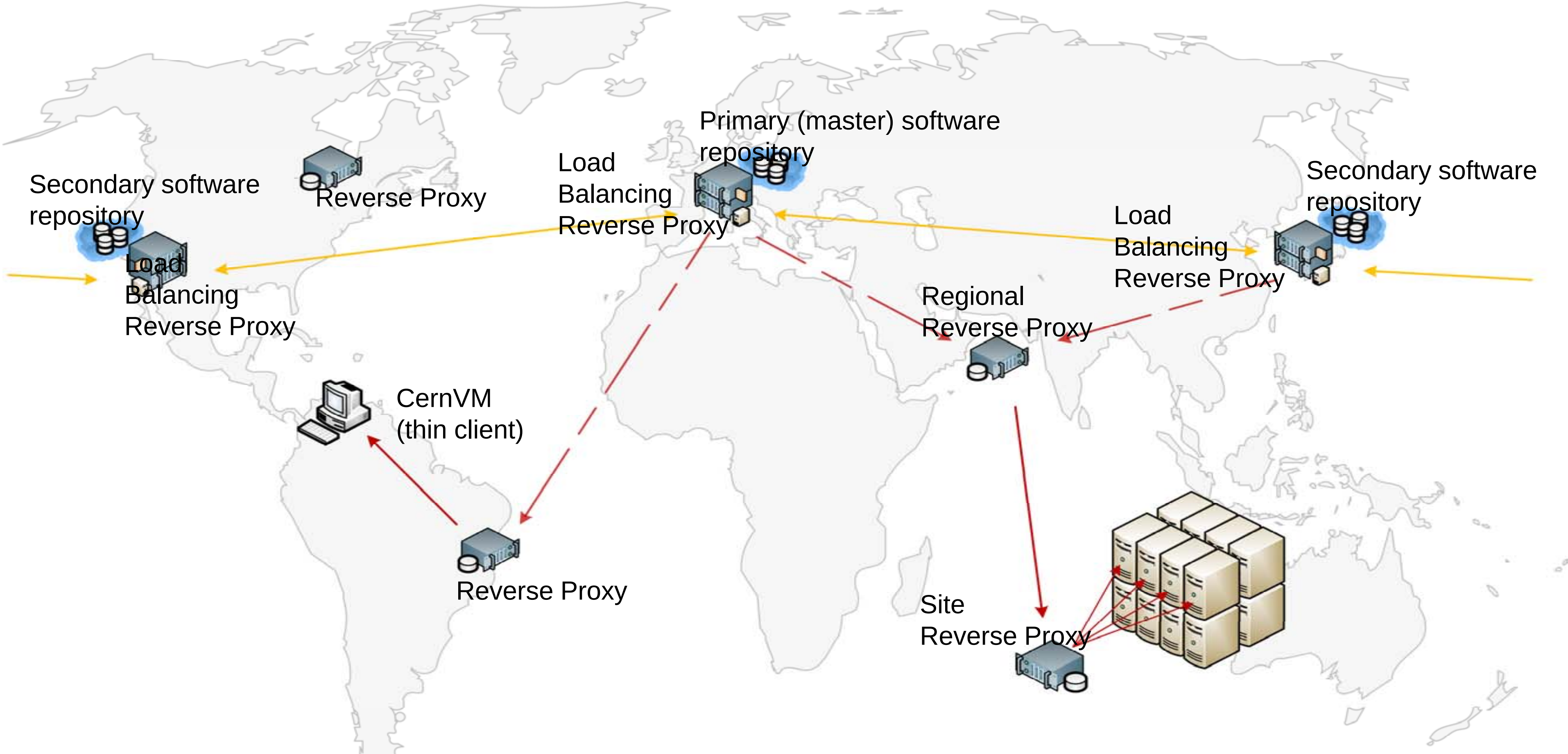
1.01 Release

- Available now for download from
  - http://cern.ch/cernvm/?page=Release1.01
- Can be run on
  - Linux (KVM, Xen,VMware Player, VirtualBox)
  - Windows(WMware Player, VirtualBox)
  - Mac (Fusion, Parallels, VirtualBox)
- Release Notes
  - http://cern.ch/cernvm/?page=ReleaseNotes
- HowTo
  - http://cern.ch/cernvm/?page=HowTo
- Appliance can be configured and used with ALICE, LHCb, ATLAS (and CMS) software frameworks

CernVM Next Steps

- Remove single point of failure, develop and test a Content Delivery Network
- Migrate CernVM to rPath Linux 2 (SLC5 compatible)
- Migration of our pilot services on IT hosted resources
- Investigate CernVM as job hosting environment
  - Voluntary computing such as BOINC
  - Dedicated virtual facilities

Removing Single Point of Failure



Content Delivery Network

Adapting the 'Grid' (1)

- We are confident that running multi-thread or multi-process applications will be efficient
  - Memory footprint, I/O optimization, etc.
  - Scaling-down the total # of jobs to be managed
- Batch systems and 'Grids' need to be adapted for multi-core jobs
  - A user should be able to submit jobs using the 8 cores or more available in a box
  - The scheduling should be strait-forward without having to wait for resources to be available

Adapting the 'Grid' (2)

- The CernVM platform is being used by Physicists to develop/test/debug data analysis
- Ideally the same environment should be used to execute their 'jobs' in the Grid
  - Experiment software validation requires large datasets
  - Software installation 'on-demand'
  - Decoupling application software from system software
- How can the existing 'Grid' be adapted to CernVM?
  - Virtual Machine submission?
  - Building a 'virtual' Grid on top of the 'physical' Grid?

Final Remarks

- Starting to obtain some practical results from the two R&D projects
  - The use of the KSM module is basically ready, adaptation of existing frameworks is progressing, algorithm parallelization promising
  - The CernVM platform is being used already (670 downloads of latest release 1.01)
- Proposed a workshop around June to discuss the two previous issues
  - All stakeholders should participate