Quattor, an overview

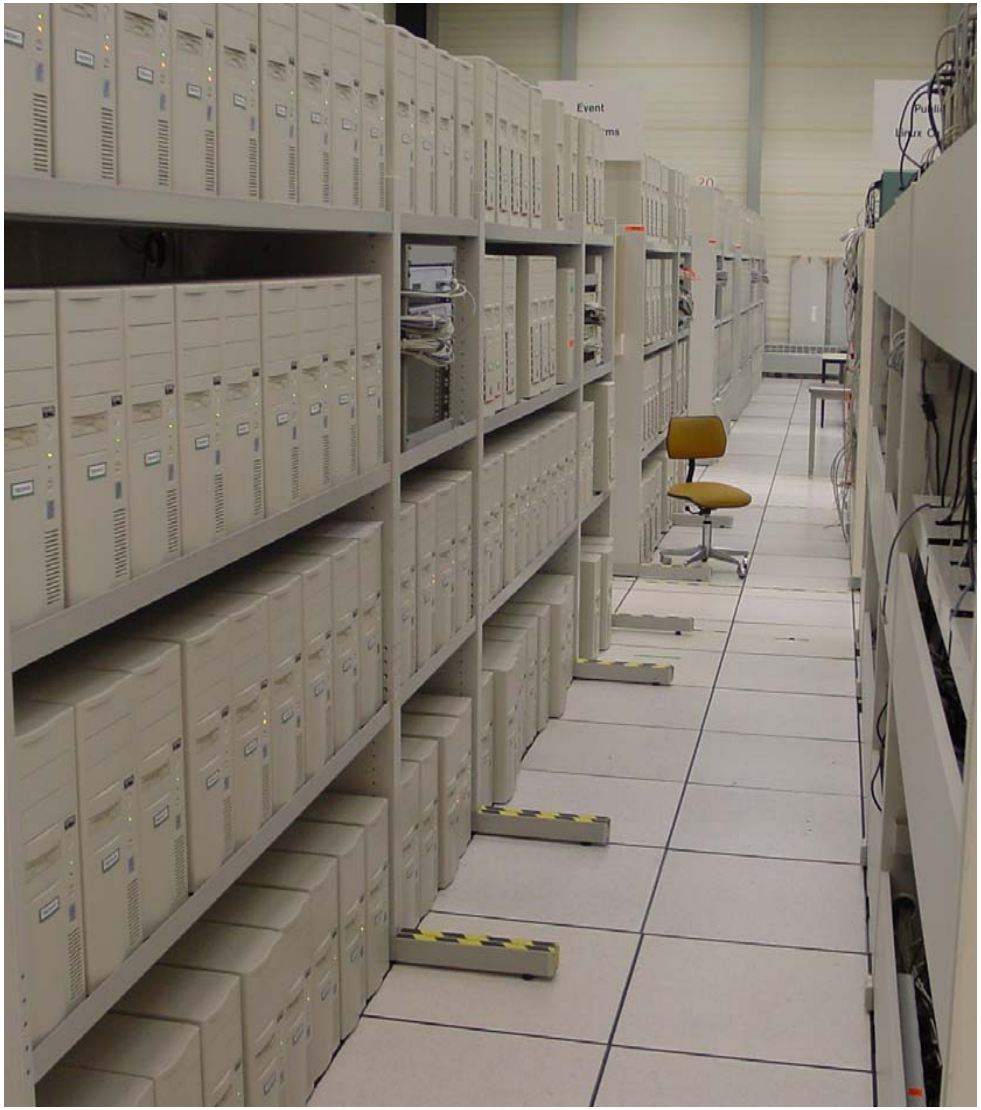
Germán Cancio CERN IT/FIO

LCG workshop, 24/3/04

http://quattor.org

Outline

- Concepts
- Architecture and Functionality
- Deployment status



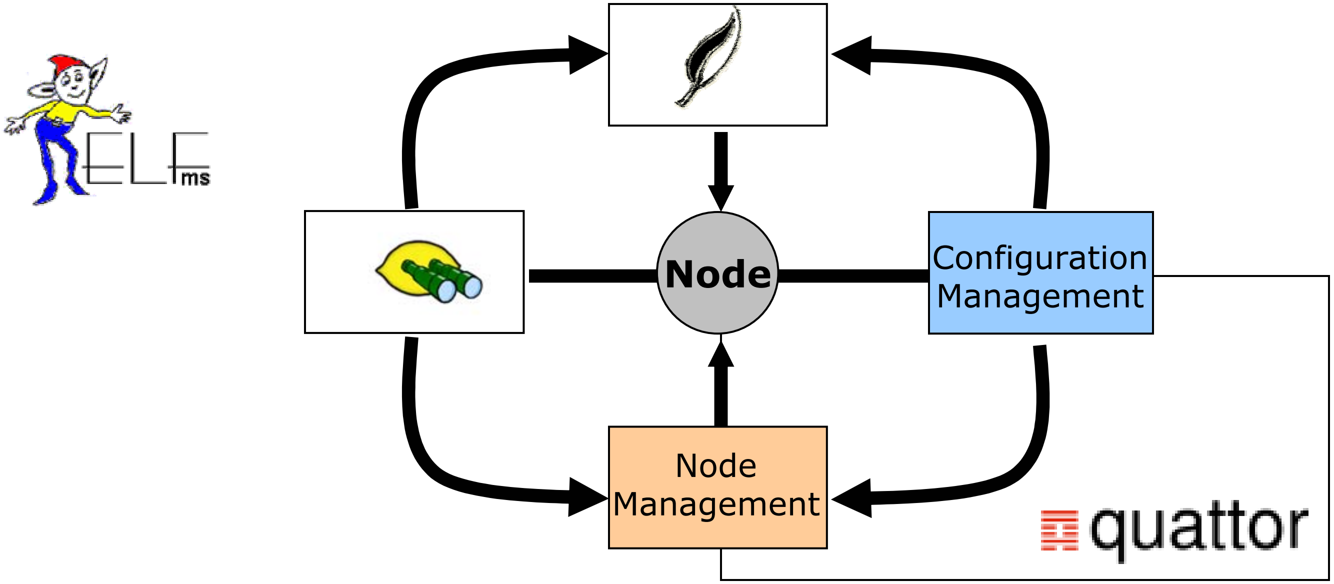
quattor in a nutshell

- quattor: fabric management system
  - Configuration, installation and management of fabric nodes

- Used to manage most of the Linux nodes in the CERN CC
  - >2000 nodes out of ~ 2200
  - Multiple functionality (batch nodes, disk servers, tape servers, DB, web, …)
  - Heterogeneous hardware (memory, HD size,..)

- Started in the scope of EDG WP4
- Part of ELFms , together with
  - LEMON monitoring system
  - LEAF Hardware and State Mgmt system

quattor architecture - overview

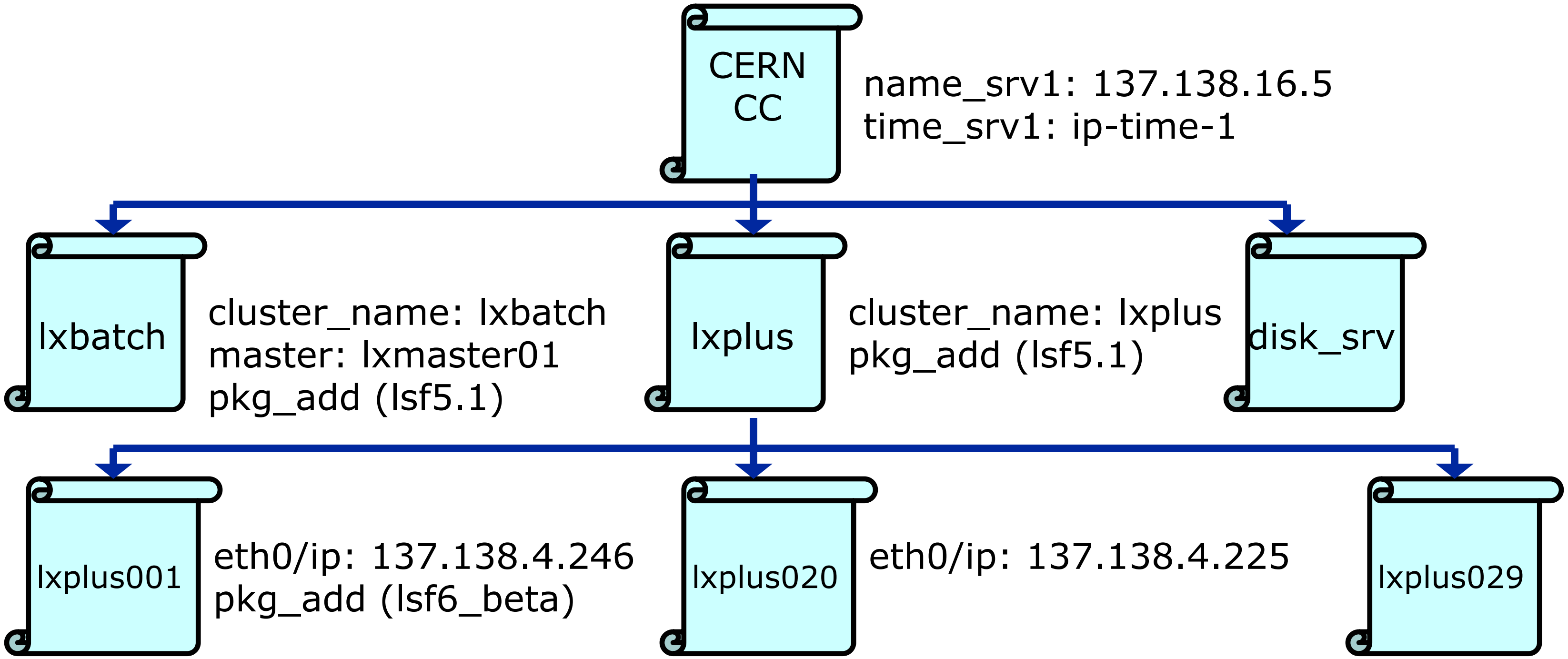


- Configuration Management
  - Configuration Database
  - Configuration access and caching
  - Graphical and Command Line Interfaces
- Node and Cluster Management
  - Automated node installation
  - Node Configuration Management
  - Software distribution and management

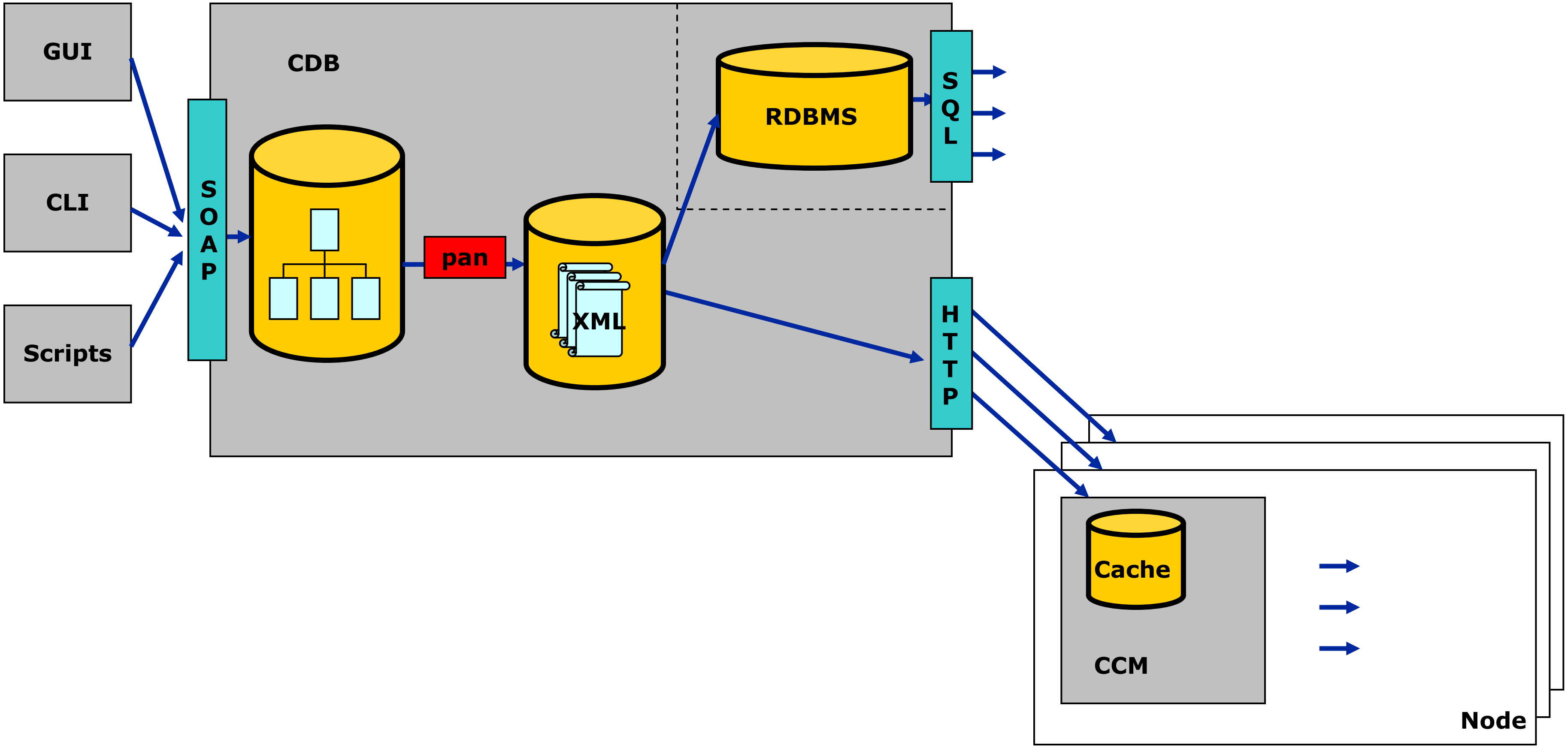
Configuration Management

Configuration Information

- Configuration is expressed using a language called Pan
- Information is arranged into templates
  - Common properties set only once
- Using templates it is possible to create hierarchies to match service structures



Configuration Management Architecture



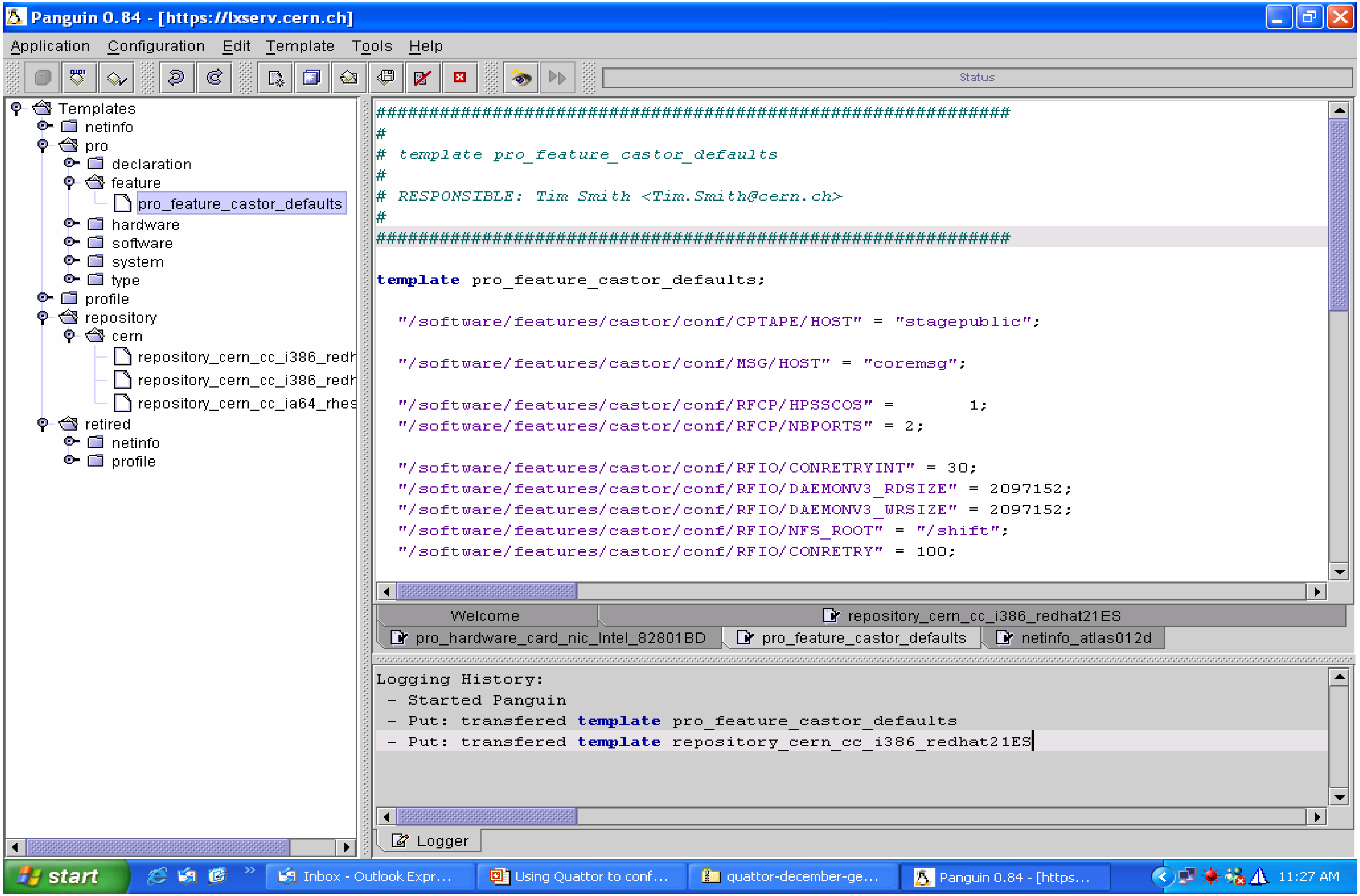
Configuration Database (CDB)

- Keeps complete configuration information
- Configuration describes the *desired* state of the managed machines.
- Data consistency is enforced by a transaction mechanism
  - All changes are done in transactions
- Configuration is *validated* and kept under version control (CVS)
  - Built-in validation (e.g. types), user defined validation
  - Detects concurrent modification conflicts
- SQL query interface for properties spanning across nodes
  - eg. get all machines on LXBATCH with more than 512 Mbytes of memory
- Node-based Configuration Cache Manager (CCM)
  - Fast, network-independent, and local access to configuration
  - Avoid peaks on CDB servers

Examples of information in CDB

- Hardware
  - CPU
  - Hard disk
  - Network card
  - Memory size
  - Physical node location in CC
- Software
  - Installed software packages (RPMs, PKGs)
- System
  - Grid services configuration (currently WN’s)
  - System services configuration (NFS mounts, SSH config..)
  - Partition table
  - Load balancing information
- Cluster information
  - Cluster name and type
  - Batch master
- Audit information
  - Contract type and number
  - Purchase date

Graphical User Interface - PanGUIn

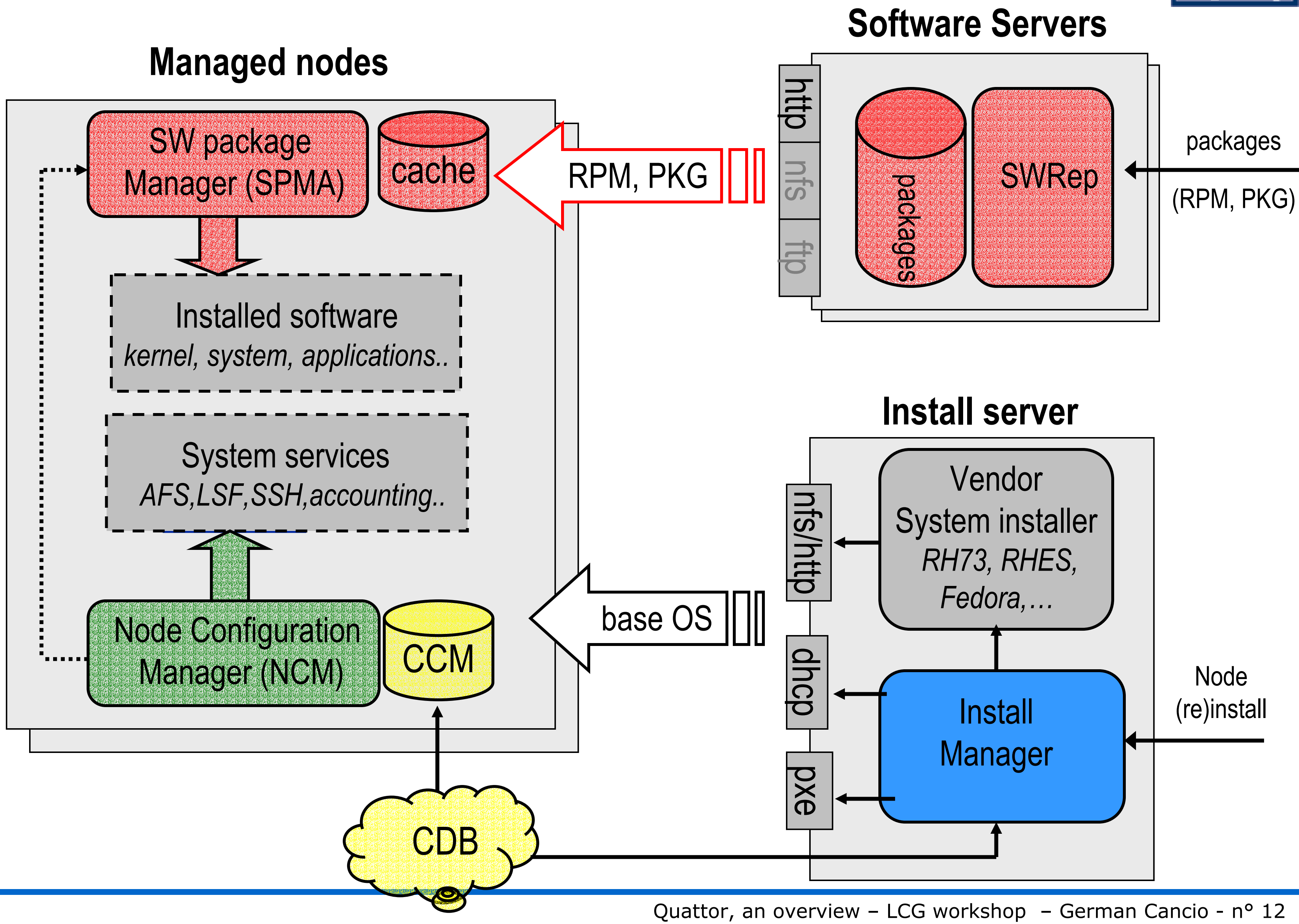


```
##############################################################
#
# template pro_feature_castor_defaults
#
# RESPONSIBLE: Tim Smith <Tim.Smith@cern.ch>
#
##############################################################

template pro_feature_castor_defaults;

  "/software/features/castor/conf/CPTAPE/HOST" = "stagepublic";

  "/software/features/castor/conf/MSG/HOST" = "coremsg";

  "/software/features/castor/conf/RFCP/HPSSCOS" =       1;
  "/software/features/castor/conf/RFCP/NBPORTS" = 2;

  "/software/features/castor/conf/RFIO/CONRETRYINT" = 30;
  "/software/features/castor/conf/RFIO/DAEMONV3_RDSIZE" = 2097152;
  "/software/features/castor/conf/RFIO/DAEMONV3_WRSIZE" = 2097152;
  "/software/features/castor/conf/RFIO/NFS_ROOT" = "/shift";
  "/software/features/castor/conf/RFIO/CONRETRY" = 100;
```

```
Logging History:
 - Started Panguin
 - Put: transfered template pro_feature_castor_defaults
 - Put: transfered template repository_cern_cc_i386_redhat21ES
```

Node (Cluster) Management

Managing (cluster) nodes



Install Manager

- Sits on top of the standard vendor installer, and configures it
  - OS version Which OS version to install
  - Network and partition information
  - What core packages
  - Custom post-installation instructions
- Automated generation of control file (KickStart)
- It also takes care of managing DHCP (and TFTP/PXE) entries
- Can get its configuration information from CDB or via command line

- Available for RedHat Linux (Anaconda installer)
  - Allows for plugins for other distributions (SuSE, Debian) or Solaris

Node Configuration

- NCM (Node Configuration Manager) is responsible for ensuring that reality on a node reflects the *desired* state in CDB.
- Framework system, where service specific plug-ins called *Components* make the necessary system changes
  - Regenerate local config files (eg. /etc/sshd/sshd_config)
  - Restard/reload services (SysV scripts)
  - configuration dependencies (eg. configure *network* before *sendmail*)
- Components invoked on boot, via cron or on CDB config changes
- Component support libraries for ease of component development
- A subset of NCM components already available, full set will be available for the next certified CERN Linux (CEL3) end of April.

Software Management (I - Server)

- *SWRep = Software Repository*
- Universal repository for storing Software:
  - Extendable to multiple platforms and packagers
    (RH Linux RPM, Solaris PKG, others like Debian pkg)
  - Multiple package versions/releases
- Management ("product maintainers") interface:
  - ACL based mechanism to grant/deny modification rights
    (packages associated to "areas")
- Client access: via standard protocols
  - HTTP, AFS/NFS, FTP
- Replication for load balancing/redundancy: using standard tools
  - Apache mod_proxy
  - rsync

Software Management (II - Clients)

- *SPMA = Software Package Management Agent*
- Manage *all* or a *subset* of packages on the nodes
  - On production nodes: *full control* - wipe out unknown packages, (re)install missing ones.
  - On development nodes (or desktops): *non-intrusive*, configurable management of system and security updates.
- Package *manager*, not only *upgrader*
  - Can roll back package versions
  - Transactional verification of operations
- Portability: Generic plug-in framework
  - Plug-ins available for Linux RPM and Solaris PKG, (can be extended)
- Scalability:
  - Supports HTTP (also FTP, AFS/NFS)
  - time smearing
  - Package pre-caching
- Possible to access multiple repositories (division/experiment specific)

Improvements wrt EDG-LCFG

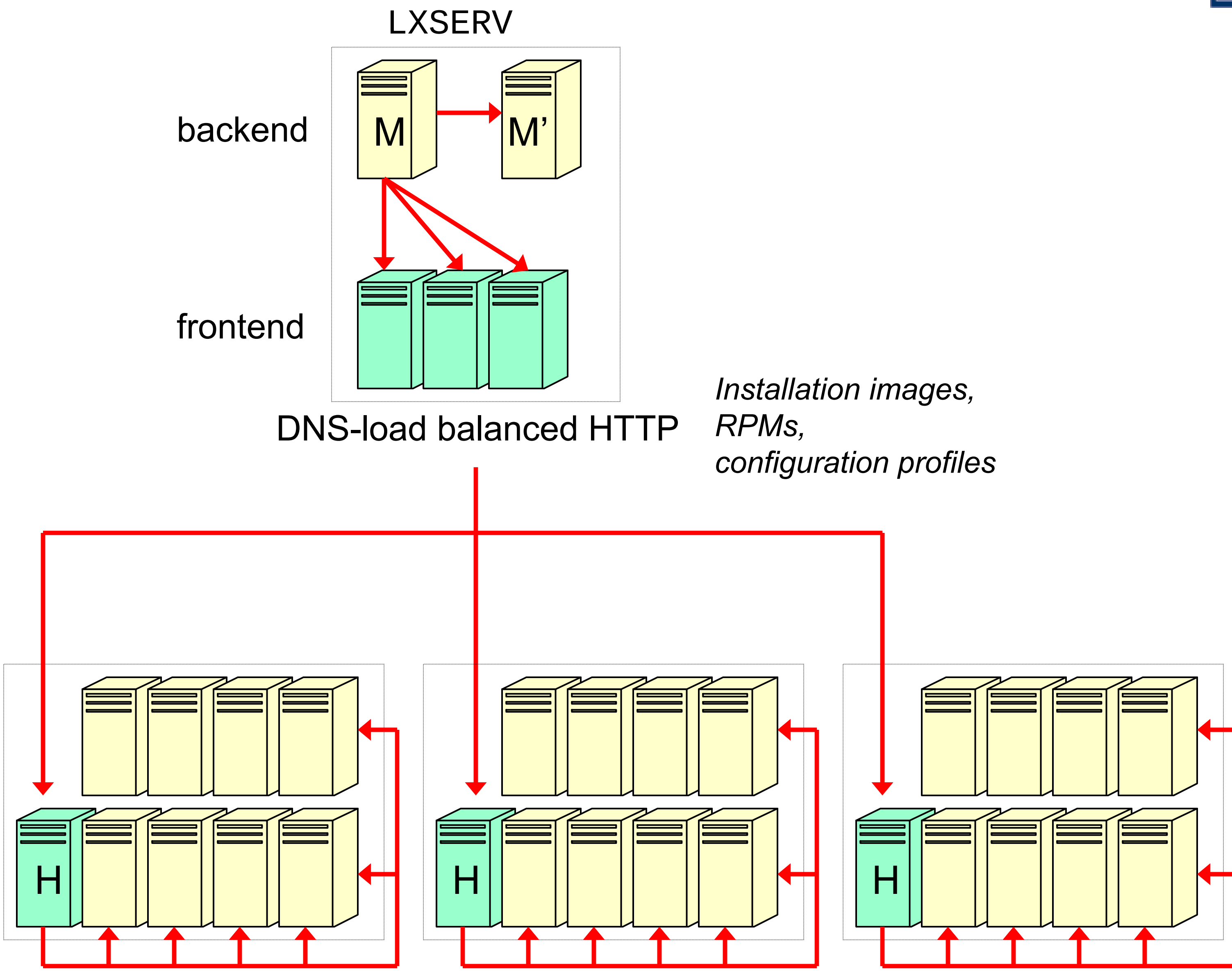
- New and powerful configuration language
  - True hierarchical structures
  - Extendable data manipulation language
  - (user defined) typing and validation
- SQL query backend
- Portability
  - Plug-in architecture -> Linux and Solaris
- Enhanced components
  - Sharing of configuration data between components now possible
  - New component support libraries
  - *Native* configuration access API (NVA-API)
- Stick to the standards where possible
  - Installation subsystem uses system installer
  - Components don't replace SysV init.d subsystem
- Modularity
  - Clearly defined interfaces and protocols
  - Mostly independent modules
  - "light" functionality built in (eg. package management)
- Improved scalability
  - Enabled for proxy technology
  - NFS mounts not necessary any longer
- Enhanced management of software packages
  - ACL's for SWRep
  - Multiple versions installable
  - No need for RPM 'header' files
- Last but not least…: **Support!**
  - EDG-LCFG is frozen and obsoleted (no ports to newer Linux versions)
  - **LCFG -> EDG-LCFGng -> quattor**

Deployment

Quattor deployment @ CERN (I)

- Quattor is used by FIO to manage most CC Linux nodes:
  - >1700 nodes, 15 clusters – to be scaled up to >5000 in 2006-8 (LHC)
    - LXPLUS, LXBATCH, LXBUILD, disk and tape servers, Oracle DB servers
  - RedHat 7.3 and RHES 2.1
  - CEL3 / RHES30 (also on IA64) to come soon (porting now)
  - Server cluster (LXSERV) hosting replicated CDB and SWRep
  - Started now: head nodes using Apache proxy technology for software and configuration distribution (see next slide)
- Quattor will be available on Linux desktops for CEL3
- Solaris clusters, server nodes and desktops to come for Solaris9

Proxy architecture



Installation images,
RPMs,
configuration profiles

Quattor deployment @ CERN (II)

- LCG-2 WN configuration:
  - > 400 nodes configured as LCG-2 Worker Nodes (250 for CMS)
  - Configuration components for RM, EDG/LCG setup, Globus
- Usage examples:
  - Upgrade from LSF 4.2 to LSF 5.1 on >1000 nodes within 15 minutes, *without service interruption*
  - All sw (functional and security) upgrades are done by SPMA
    - openssl/ssh security updates
    - KDE upgrades (~ 400 MB per node) on >700 nodes
    - etc … (~once a week!)
  - Kernel upgrades: SPMA can handle multiple versions of the same package -> Allows to separate in time installation and activation (after reboot) of new kernel

Deployment outside CERN-CC

- EDG: no time for wide deployment
  - Estimated effort for moving from LCFG to quattor exceeded remaining EDG lifetime
  - EDG focus on stability rather than middleware functionality
- Tutorials held at HEPiX and EDG conferences have caused positive feedback and interests:
  - Experiments: CMS, LHCb, Atlas
  - HEP institutes: UAM Madrid, LAL/IN2P3, NIKHEF, Liverpool University
  - Projects: Grille 5K (CNRS France)
- Community driven effort to use quattor for general LCG-2 configuration
  - Workshop this Friday to define initial steps
  - Based on already existing WN config components
- **CERN will help with deployment at other sites**
- **Collaboration** for providing missing pieces, eg. configuration components, GUI’s, beginner’s user guides?

quattor

http://quattor.org

Differences with ROCKS

- Rocks: better documentation, nice GUI, easy to setup
- Design principle: reinstall nodes in case of configuration changes
  - No configuration or software updates on running systems
  - Suited for production? Efficiency on batch nodes, upgrades / reconfigs on 24/24,7/7 servers (eg. gzip security fix, reconfig of CE address on WN's)
- Assumptions on network structure (private,public parts) and node naming
- No indication on how to extend the predefined node types or extend the configured services
- Limited configuration capacities (key/value)
- No multiple package versions (neither on repository, nor simultaneously on a single node)
  - Eg. different kernel versions on specific node types
- Works only for RH Linux (Anaconda installer extensions)

Differences with ASIS/SUE

ASIS: *See post-C5 14/3/2003*

- Scalability
  - HTTP vs. shared file system
- Supports native packaging system (RPM, PKG)
- Manages all software on the node
- ‘real’ Central Configuration database
- (But: no end-user GUI, no package generation tool)

SUE:

- Focus on configuration, not installation
- Powerful configuration language
  - True hierarchical structures
  - Extendable data manipulation language
  - (user defined) typing and validation
  - Sharing of configuration data between components now possible
- Central Configuration Database
- Supports unconfiguring services
- Improved depenency model
  - Pre/post dependencies
- Revamped component support libraries

NCM Component example

```
[...]
sub Configure {
  my ($self,$config) = @_;
  # access configuration information
  my $arch=$config->getValue('/system/architecture’); # CDB API
  $self->Fail (“not supported") unless ($arch eq ‘i386’);
  # (re)generate and/or update local config file(s)
  open (myconfig,’/etc/myconfig’); …
  # notify affected (SysV) services if required
  if ($changed) {
    system(‘/sbin/service myservice reload’); …
  }
}
sub Unconfigure { ... }
```

Key concepts behind quattor

- Autonomous nodes:
  - Local configuration files
  - No remote management scripts
  - No reliance on global file systems AFS/NFS
- Central control:
  - Primary configuration is kept centrally (and replicated on the nodes)
  - A single source for all configuration information
- Reproducibility:
  - Idempotent operations
  - Atomicity of operations
- Scalability:
  - Load balanced servers, scalable protocols
- Use of standards:
  - HTTP, XML, RPM/PKG, SysV init scripts, …
- Portability:
  - Linux, Solaris