Fabric Management: Progress and Plans

PEB

Tim Smith IT/FIO

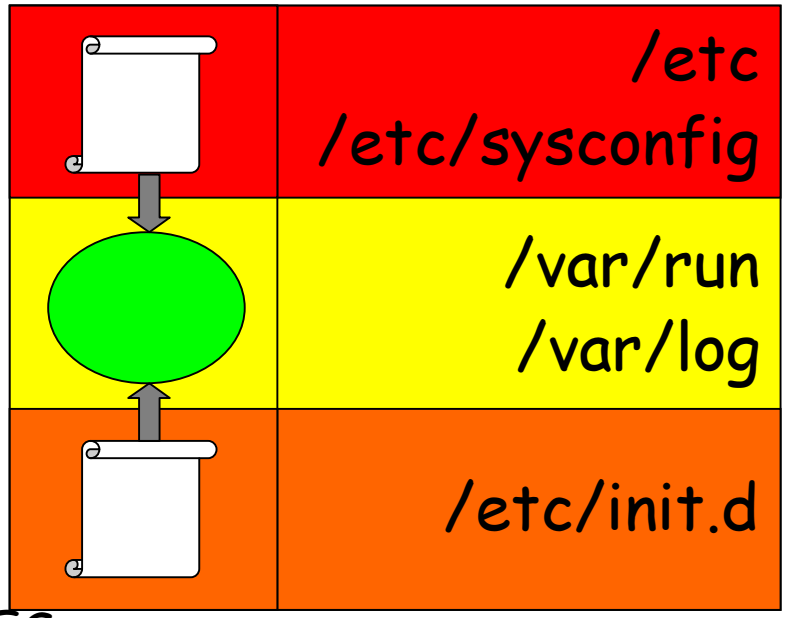
Contents

- Manageable Nodes
- Framework for management
- Deployed at CERN
- Plans

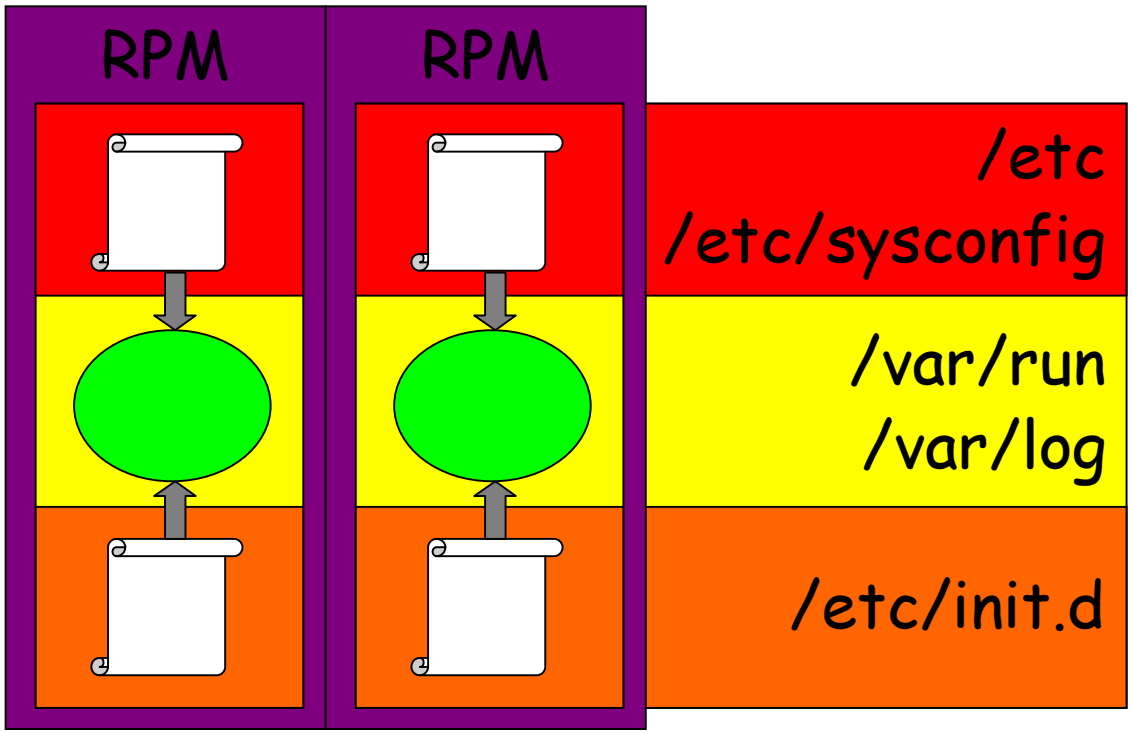
Standard / Autonomous Nodes

- Linux Standards Base
  - http://www.linuxbase.org

- Distributed / Autonomous
  - Decoupling of nodes
    - Local config files
    - Local programs do all work
  - Avoid inherent drift
    - No external crontabs
    - No remote mgmt scripts
    - No remote application updates
    - No parallel cmd engines
    - No global file-systems AFS/NFS

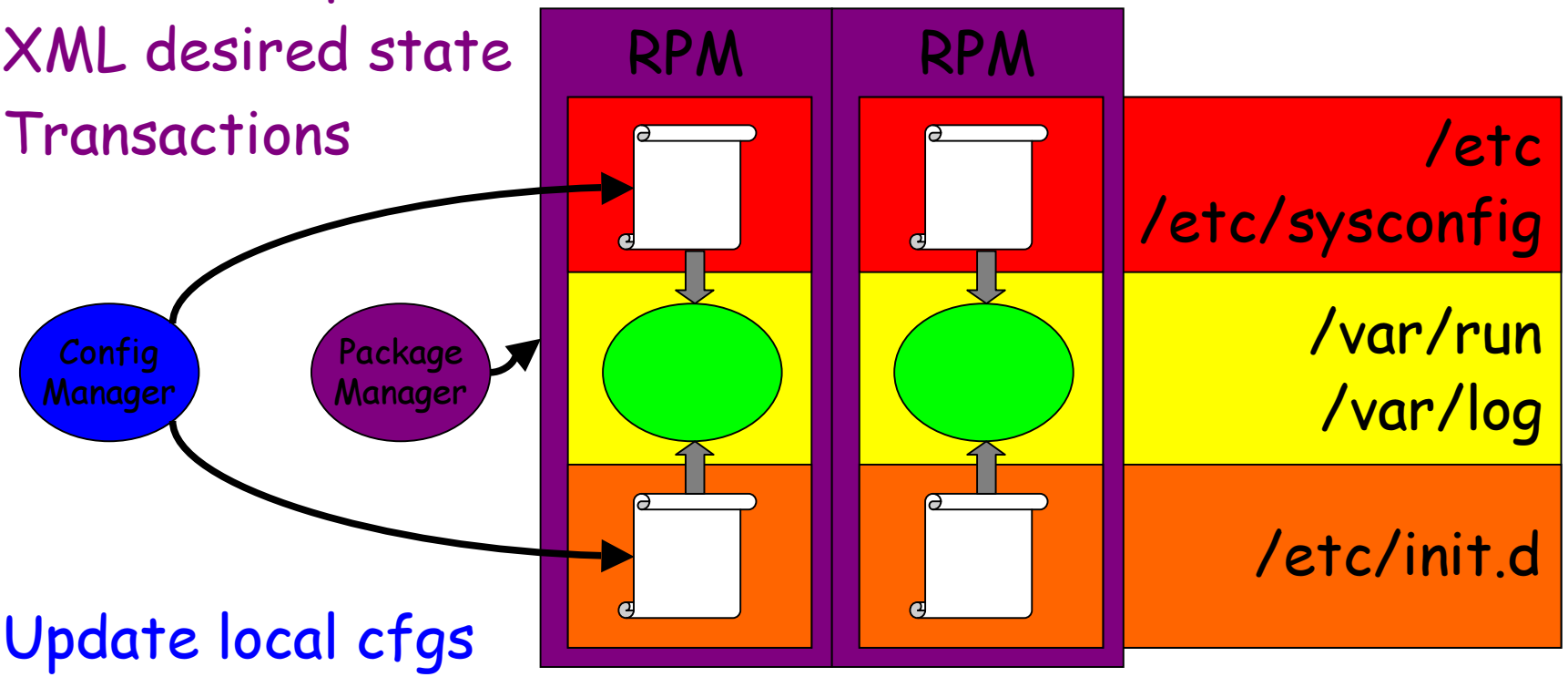


Standard / Autonomous Nodes



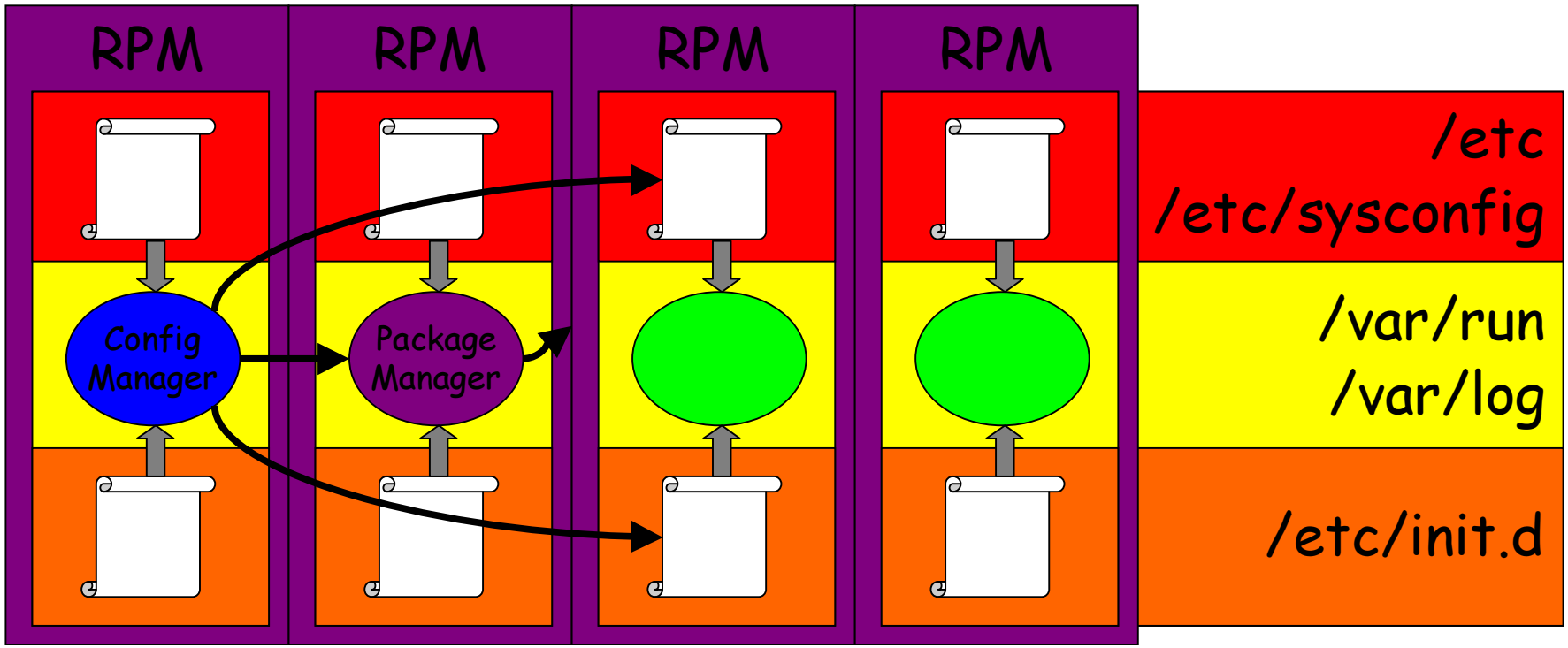
Standard / Autonomous Nodes

- All SW in RPM
- Control complete SW set
- XML desired state
- Transactions



- Update local cfgs
- Manage through SysV scripts

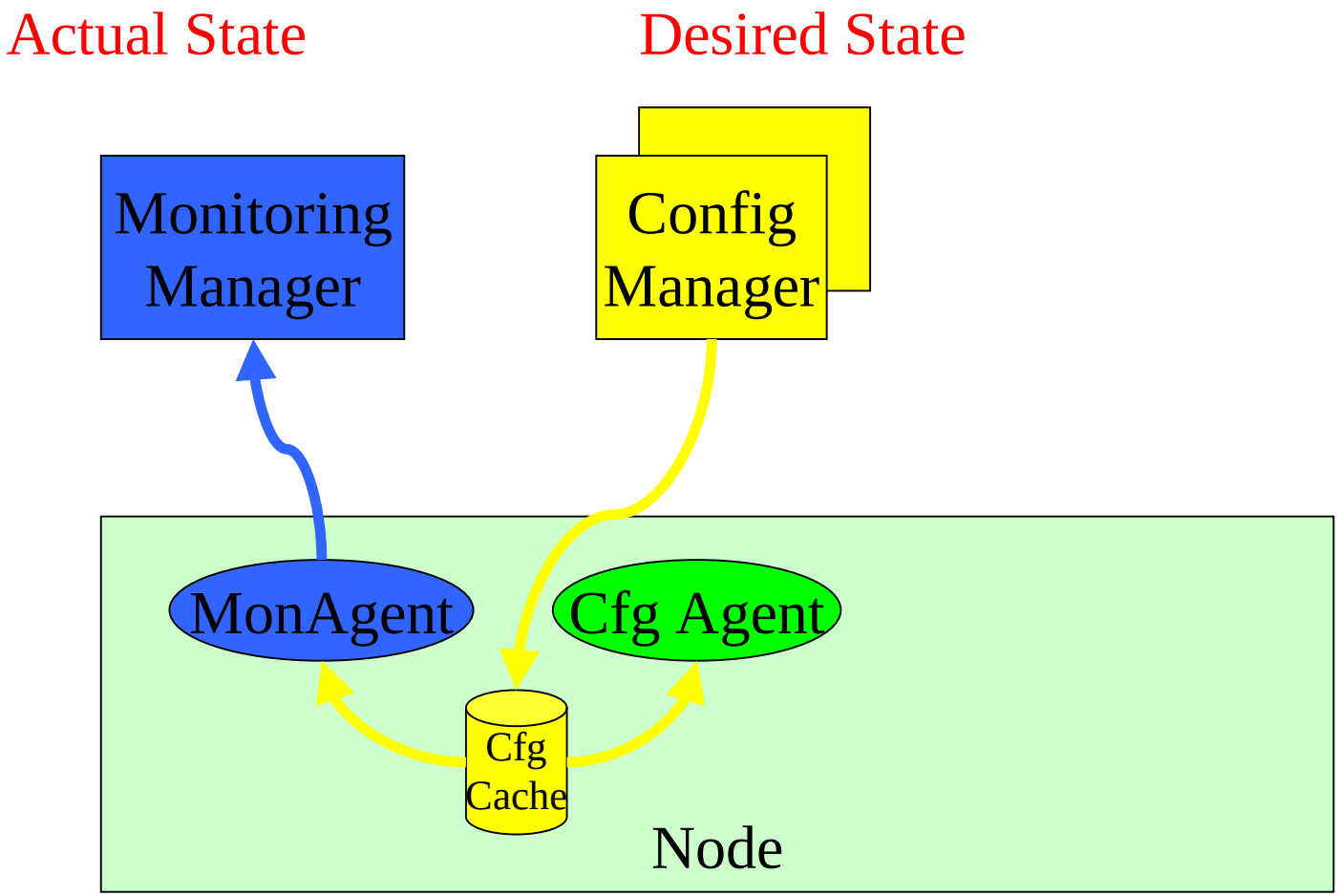
Standard / Autonomous Nodes



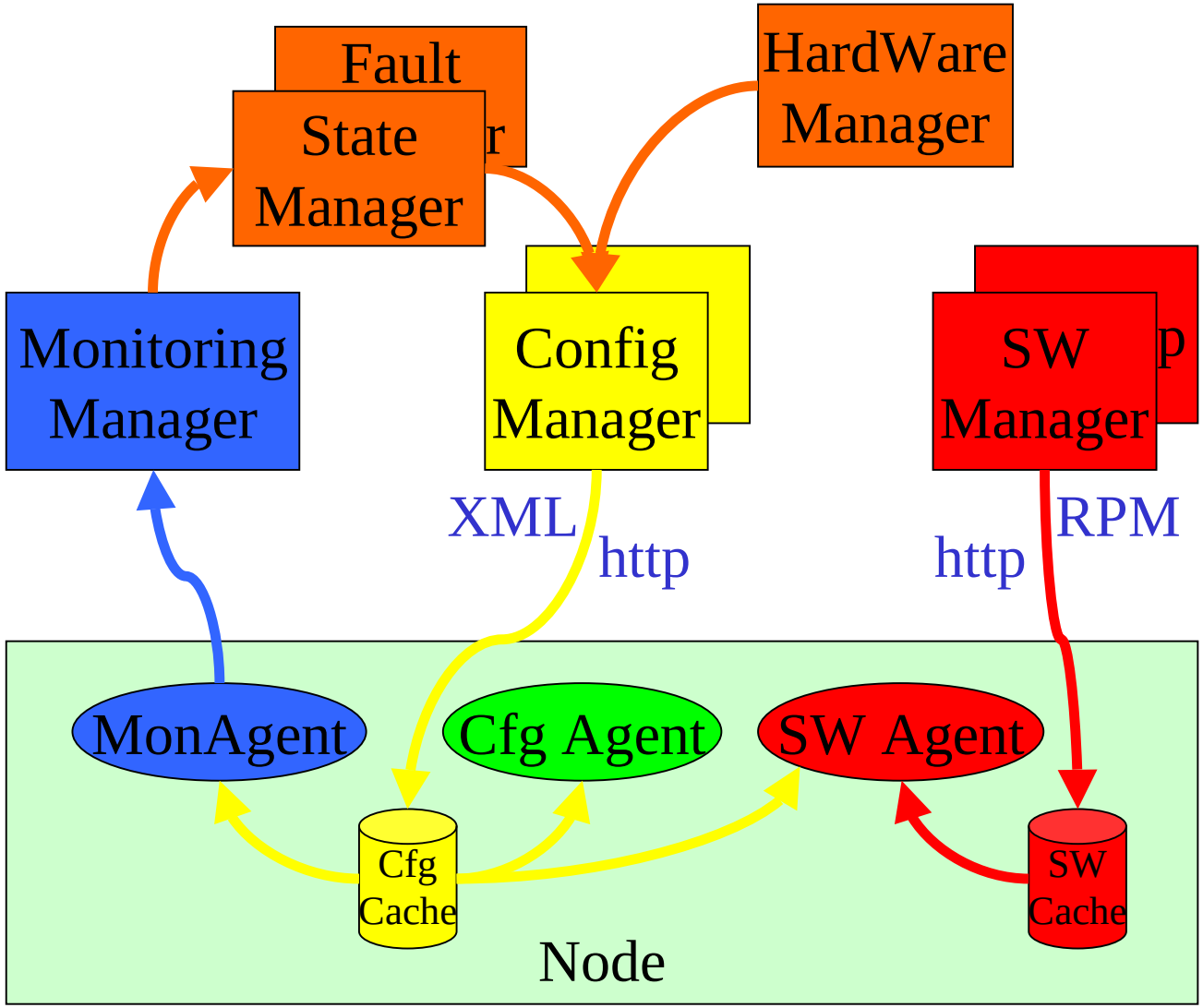
Framework Principles

- Installation
  - Automation of *complete* process
  - Reproducible
  - Clear installation:
    - Short, clean, and reproducible Kickstart base installation
    - *All* Software installation by RPM
    - System configuration by one tool
    - Configuration information through unique interface, from unique store
- Maintenance:
  - The running system must be 'updateable'
  - The updates must go into new installations afterwards
  - System updates have to be automated, but steerable

Framework for Management



Framework for Management

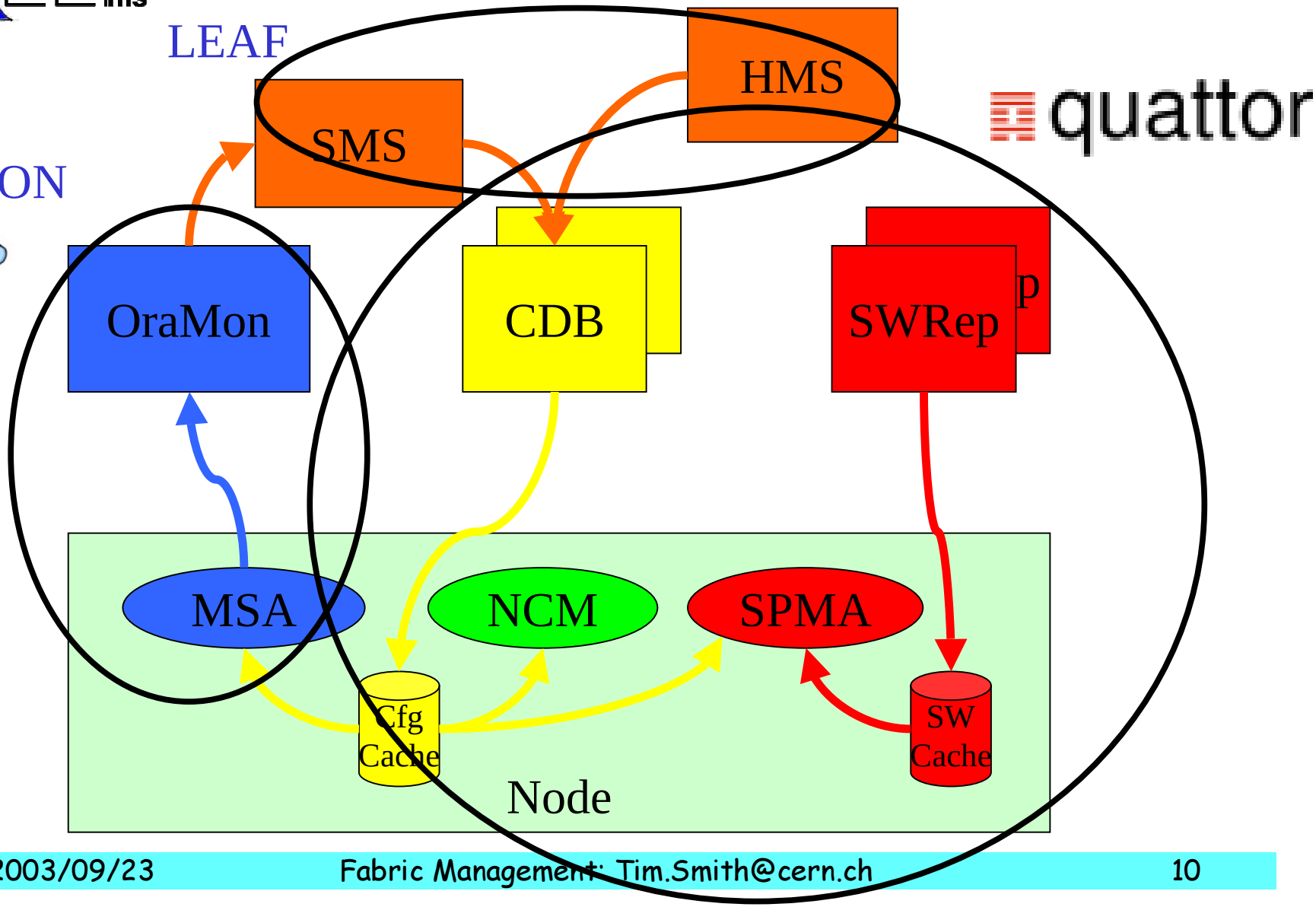


ELFms:

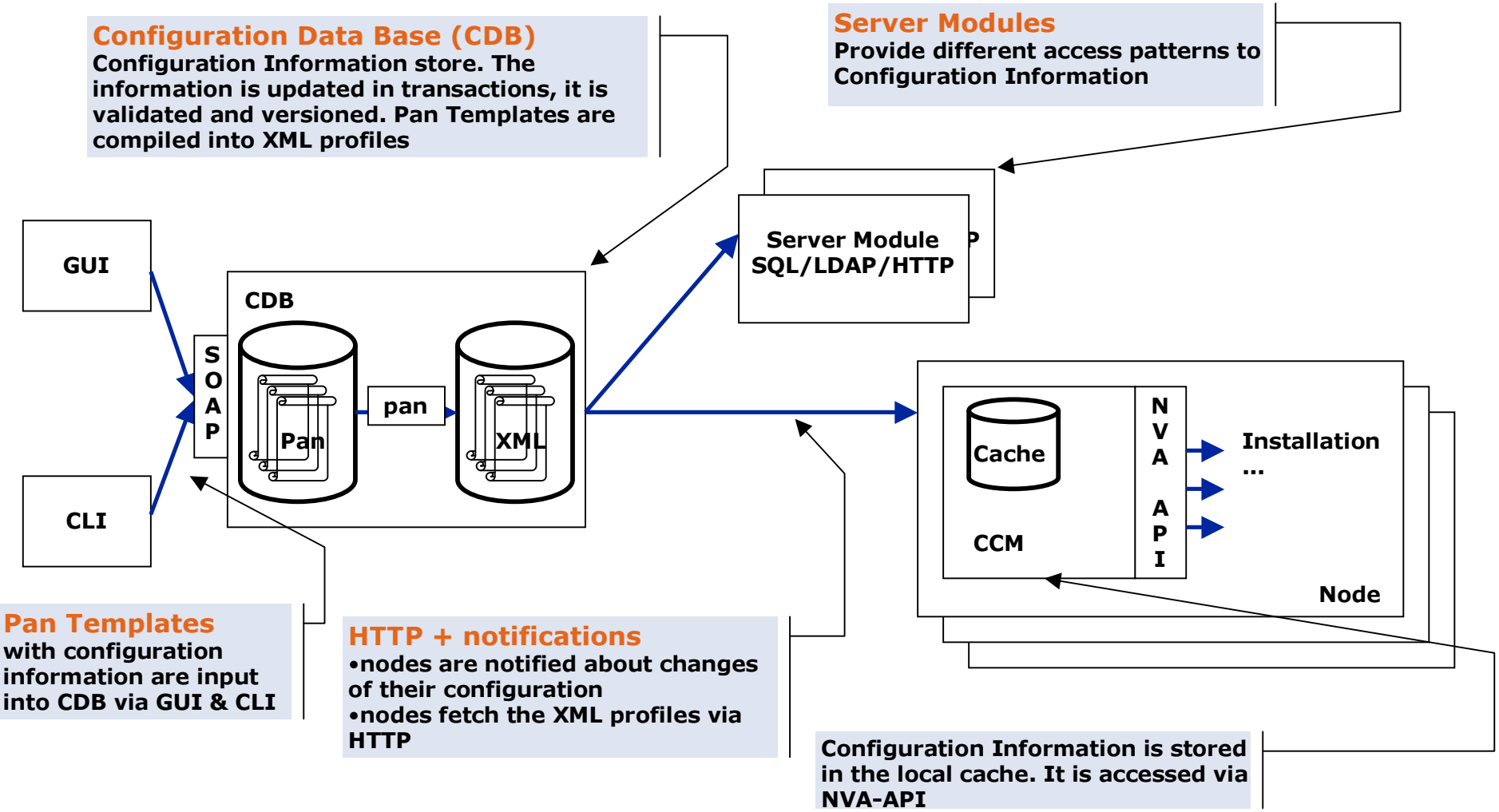
LEAF

LEMON

quattor



quattor configuration



Config/SW Considerations

- Hierarchical configuration specification
  - Graph rather than tree structure
  - Common properties set only once
- Node profiles
  - Complete specification in one XML file
  - Local cache
  - Transactions / Notifications
- Externally specified, Versioned: CVS repos.
- One tool to manage all SW: SPMA
  - System and application
- Update verification nodes + release cycle
- Procedures and Workflows

NCM details

- Modules:
  - "Components"
  - Invocation and notification framework
  - Component support libraries
  - Configuration query tool
- NCM is responsible for ensuring that reality on a node reflects the desired state in CDB.
  - components which make the necessary system changes
    - components **do not** install software, just ensure config is correct
    - components register with the framework to be notified when the node configuration is updated in CDB.
  - Usually, this implies regenerating and/or updating local config files (eg. `/etc/sshd_config`)
  - Use standard system facilities (SysV scripts) for *managing* services
    - Components notify services using SysV scripts when cfg changes.
  - Possible to define configuration dependencies between components
    - Eg. configure *network* before *sendmail*

Hardware variety

```
hardware_diskserv_elonex_1100
hardware_elonex_500
hardware_elonex_600
hardware_elonex_800
hardware_elonex_800_mem1024mb
hardware_elonex_800_mem128mb
hardware_seil_2002
hardware_seil_2002_interactiv
hardware_seil_2003
hardware_siemens_550
hardware_techas_600
hardware_techas_600_2
hardware_techas_600_mem512mb
hardware_techas_800
```

```
structure template
hardware_cpu_GenuineIntel_Pentium_III_1100;

  "vendor" = "GenuineIntel";
  "model"  = "Intel(R) Pentium(R) III CPU family 1133MHz";
  "speed"  = 1100;
```

```
template hardware_diskserver_elonex_1100;

"/hardware/cpus" = list(create("hardware_cpu_GenuineIntel_Pentium_III_1100"),
                        create("hardware_cpu_GenuineIntel_Pentium_III_1100"));
"/hardware/harddisks" = nlist("sda", create("pro_hardware_harddisk_WDC_20"));
"/hardware/ram" = list(create("hardware_ram_1024"));
"/hardware/cards/nic" = list(create("hardware_card_nic_Broadcom_BCM5701"));
```

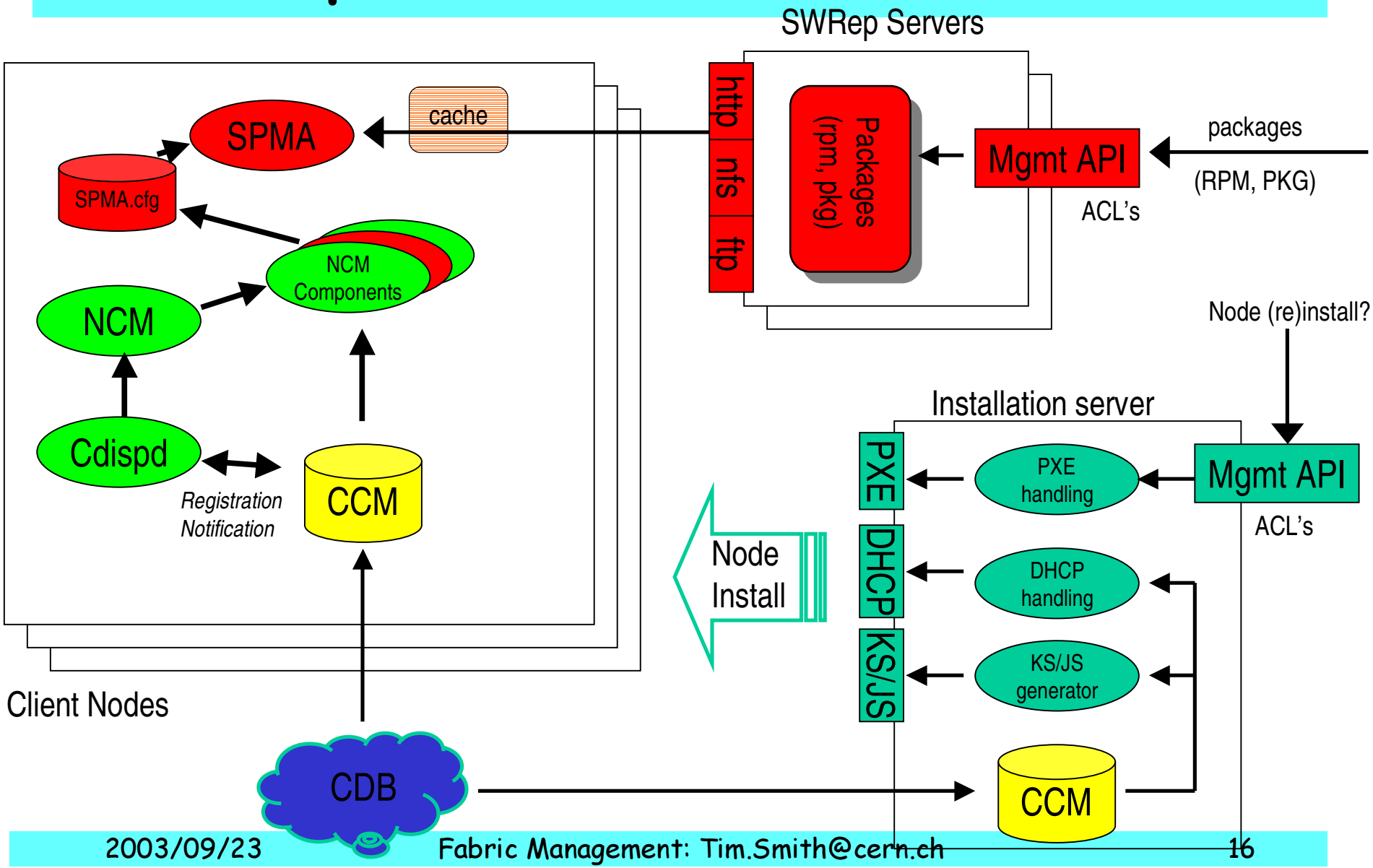
```
structure template hardware_card_nic_Broadcom_BCM5701;

  "manufacturer" = "Broadcom Corporation NetXtreme BCM5701 Gigabit Ethernet";
  "name"         = "3Com Corporation 3C996B-T 1000BaseTX";
  "media"        = "GigaBit Ethernet";
  "bus"          = "pci";
```

What is in CDB ?

- Hardware
  - CPU
  - Hard disk
  - Network card
  - Memory size
  - Location
- Software
  - Repository definitions
  - Service definitions = groups of packages (RPMs)
- System
  - Partition table
  - Cluster name and type
  - CCDB name
  - Site release
  - Load balancing information

quattor installation

quattor



SWRep

- Client-server toolsuite for the **storage** of software packages
- Universal repository:
  - Extendable to multiple platforms and package formats (RHLinux/RPM, Solaris/PKG,… others like Debian dpkg)
  - Multiple package versions/releases
  - Currently, RPM for RH73, RH21ES and soon RH10 on LXSERV cluster
- Management ("product maintainers") interface:
  - ACL based mechanism to grant/deny modification rights (packages associated to "areas")
  - Management done by FIO/FS
- Client access: via standard protocols
  - On LXSERV: HTTP
- Replication: using standard tools (eg. rsync)
  - Availability, load balancing

SPMA

- Runs on every target node
  - All LXPLUS/LXBATCH/LXSHARE/ … nodes; except old RH61 nodes
- Plug-in framework allows for portability
  - FIO/FS uses it for RPM, PS group will use it for Solaris (PKG)
- Can manage either *all* or a *subset* of packages on the nodes
  - Configured by FIO/FS to manage ALL packages.
  - *Which means*: Not configured packages are wiped out.
  - *And*: Packages missing on node but in configuration are (re)installed.
- Addresses scalability
  - Packages can be stored ahead in a local *cache*, avoiding peak loads on software repository servers (simultaneous upgrades of large farms)

SPMA (II)

- SPMA functionality:
  1. Compares the packages currently installed on the local node with the packages listed in the configuration
  2. Computes the necessary install/deinstall/upgrade operations
  3. Invokes the packager RPMT/PKGT with the right operation transaction set
- RPMT (RPM transactions) is a small tool on top of the RPM libraries, which allows for multiple simultaneous package operations resolving dependencies (unlike RPM)
  - Example: 'upgrade X, deinstall Y, downgrade Z, install T' and verify/resolve appropriate dependencies

Currently deployed at CERN

- Scale
  - 1400 node profiles
  - 3M lines of node XML specifications
  - From 28k lines of PAN definitions
- Scalability
  - 3 node load balanced web servers cluster
    - For Configuration profiles
    - For SW Repository
  - Time smeared transactions
  - Pre-deployment caching
- Security
  - GPG key pair per host – during installation
  - Per host encrypted files served

Examples

- Clusters
  - LXPLUS, LXBATCH, LXBUILD, LXSHARE, OracleDBs
  - TapeServers, DiskServers, StageServers
- Supported Platforms
  - RH7.3.3 2400 packages
  - RHES 1300 packages
  - RH10
- KDE/Gnome security patch rollout
  - 0.5 GB onto 700 nodes
  - 15 minute time smearing from 3 lb-servers
- LSF 4-5 transition
  - 10 minutes
  - No queues or jobs stopped
  - C.f. last time 3 weeks, 3 people!

Plans

- Deploying NCM
  - Port SUE features to NCM for RH10
  - Use for Solaris10
- Deploying State Manager (SMS)
- Use case targetted GUIs
- Web Servers
  - Split: FrontEnd / BackEnd Architecture

Conclusions

- Maturity brings…
  - Degradation of initial state definition
    - HW + SW
  - Accumulation of innocuous temporary procedures
- Scale brings…
  - Marginal activities become full time
    - Many hands on the systems
- Combat with strong management automation