quattor

Devolved management of distributed infrastructures with Quattor

13/11/08 – LISA'08 – San Diego

Marco Emilio Poleggi – *INFN-CNAF*

Marco.Poleggi@cnaf.infn.ch

BEgrid The Belgian Grid for Research · CERN · Grid-Ireland · GRIF Grille au service de la Recherche en Ile de France · INFN CNAF · NIKHEF · UAM Universidad Autonoma de Madrid

Outline

quattor

- Background
- Devolved management workflow
- Important features of the Pan language
- QWG: a configuration distro for devolved management
- Experience with distributed deployment
- Integration with other tools
- Conclusion

Background...

quattor

- Grid computing is changing the way resources are used
  - From big data centers to geographically distributed "federated" sites
  - Focus on the tools for managing the "fabric" (HW/SW). Requirements:
    - Scalability to deal with large installations
    - Flexibility to accommodate heterogeneous frameworks
    - Modularity to optimize configuration data usage
  - Balancing flexibility and usability
    - More actors on the scene: different responsibilities
    - Enable sites to customize without having to understand the whole configuration system
    - Share configuration common to multiple sites
    - Allow local policies

...Background

quattor

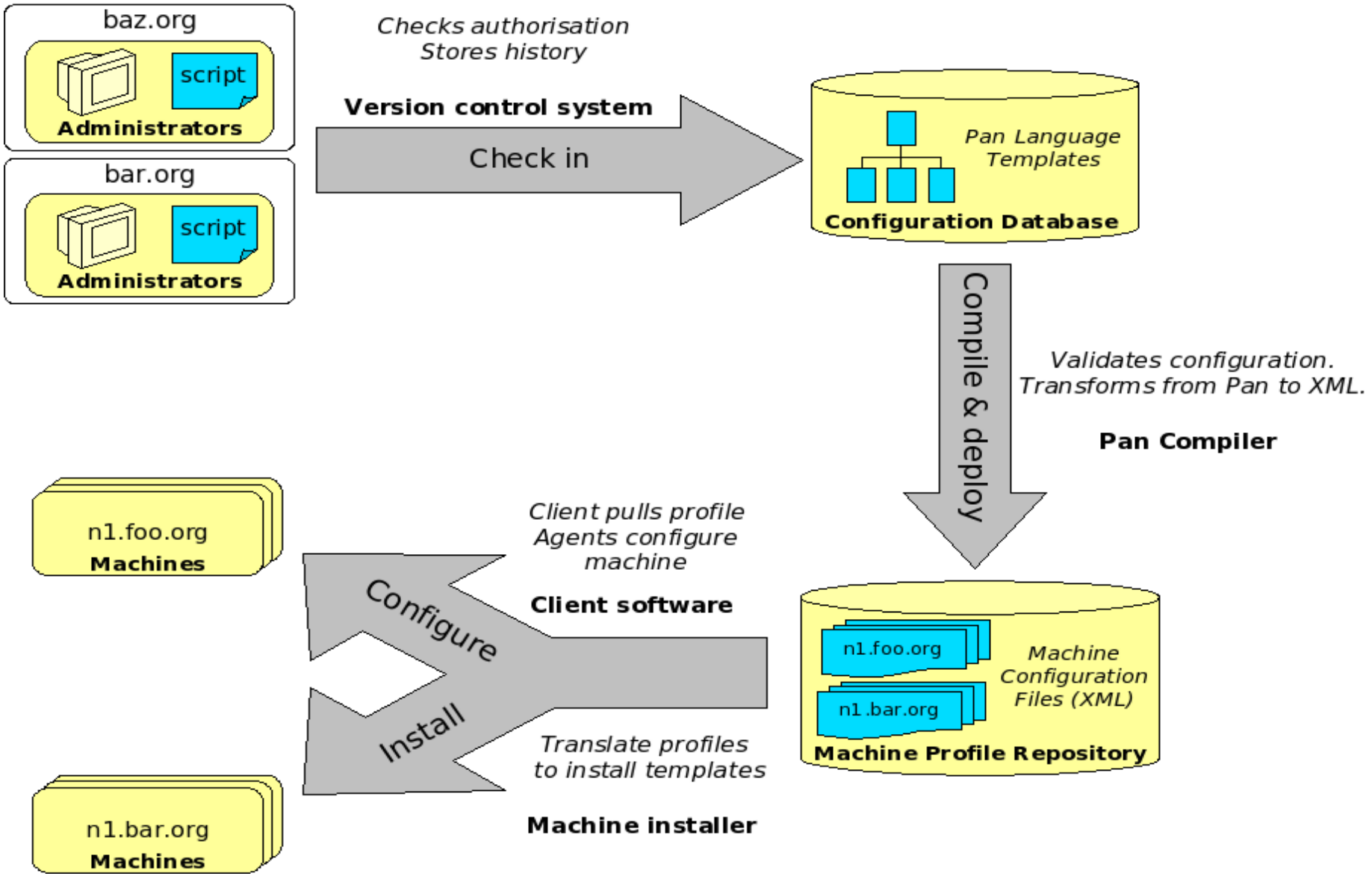
- Quattor was developed to meet the above requirements
  - Aimed to improve on its ancestor LCFG
  - Uses a high-level *declarative* configuration language – *Pan*
    - Hierarchical schema
    - Modularization for data reuse and customization
    - Pre-deployment checks through *validation*
  - Allows different service deployment strategies
  - Provides a full “configuration distribution”
    - Out-of-the box solutions for *gLite* grid services

Table 1: Quattor deployments

	Distributed			Single-site			
Metric	**BEGrid**	**Grid-Ireland**	**GRIF**	**CERN**	**CNAF**	**Nikhef**	**UAM**
Managed machines	260	417	575	8000	800	301	553
Administrators	8	11	25	100	10	4	3
Physical sites	6	18	6	1	1	1	1

Devolved management workflow...

quattor



...Devolved management workflow...

quattor

- Configuration management system
  - Subsystem deployment can be
    - → *Centralized* for strict operation control on the server
      - Sort of broker-based
    - → *Distributed* for more operational flexibility
      - Easier autonomous handling of configuration parts
  - Authentication via X.509/Kerberos5/encrypted passwords
  - Authorization via access control lists (ACLs)
- Automatic installation of managed nodes (all operations can be done remotely)
  - Retrieves information from machine *profiles*
  - Configures DHCP and PXE
  - Generates Kickstart files

...Devolved management workflow

quattor

- Node configuration management
  - Nodes are notified of changes and download fresh profiles
  - Autonomous agents (“components”) triggered by changes in specific parts of the configuration schema
    - Can also deploy manually (automatic dispatching disabled)
    - Pre/post runtime dependencies ensure correct service configuration
  - Idempotent (repeated actions have the same effect)
- Software management
  - Separation of *repository* and *configuration*
    - Different repositories accessed via HTTP
    - Package lists in Pan templates
  - Modes
    - *Strict* -- install only listed packages, remove manual installations
    - *Flexible* -- allow multiple versions, respect manual installation
  - Rollbacks can be easily performed

Important features of the Pan language

quattor

- Validation
  - Types can have attached validation code
  - Allows constraints to be checked before deployment
    - → Maximize the probability of finding bugs at compile time
- Configuration reuse
  - “structure templates” as reusable chunks for invariant configuration parts
- Modularization
  - *Namespacing:* allows similar configuration hierarchies or “modules”
    - → Independent of the configuration schema
  - *Loadpath:* selects one module out of a “namespaced” series
  - *Conditional includes:* depend on the evaluation of an expression

QWG: a configuration distro for devolved management...

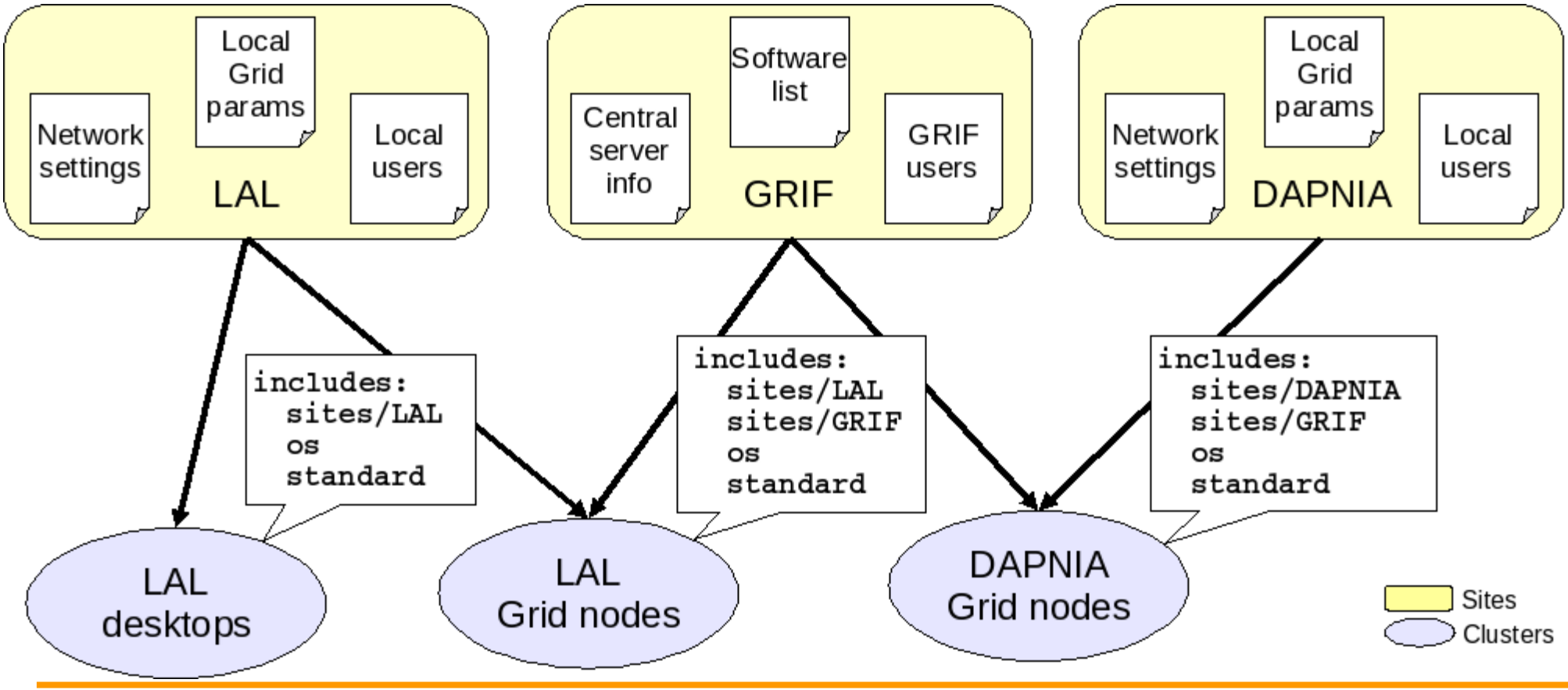
quattor

- *QWG templates* are a full configuration “distribution” for grid services
  - Large code-base of shared configuration templates
  - Local customizations can be applied using a minimal set of parameters
  - “Hook” variables for conditional *includes* allow local customizations
  - The configuration is based on the concepts of *site* and *cluster*
    - A *cluster* is an arbitrary grouping of machines that share configuration information (for example, “compute nodes” or “grid servers”).
    - A *site* is a logical group that defines a set of configuration elements to be shared by different clusters
- A single Quattor instance may be used to manage several physical sites

...QWG: a configuration distro for devolved management...

quattor

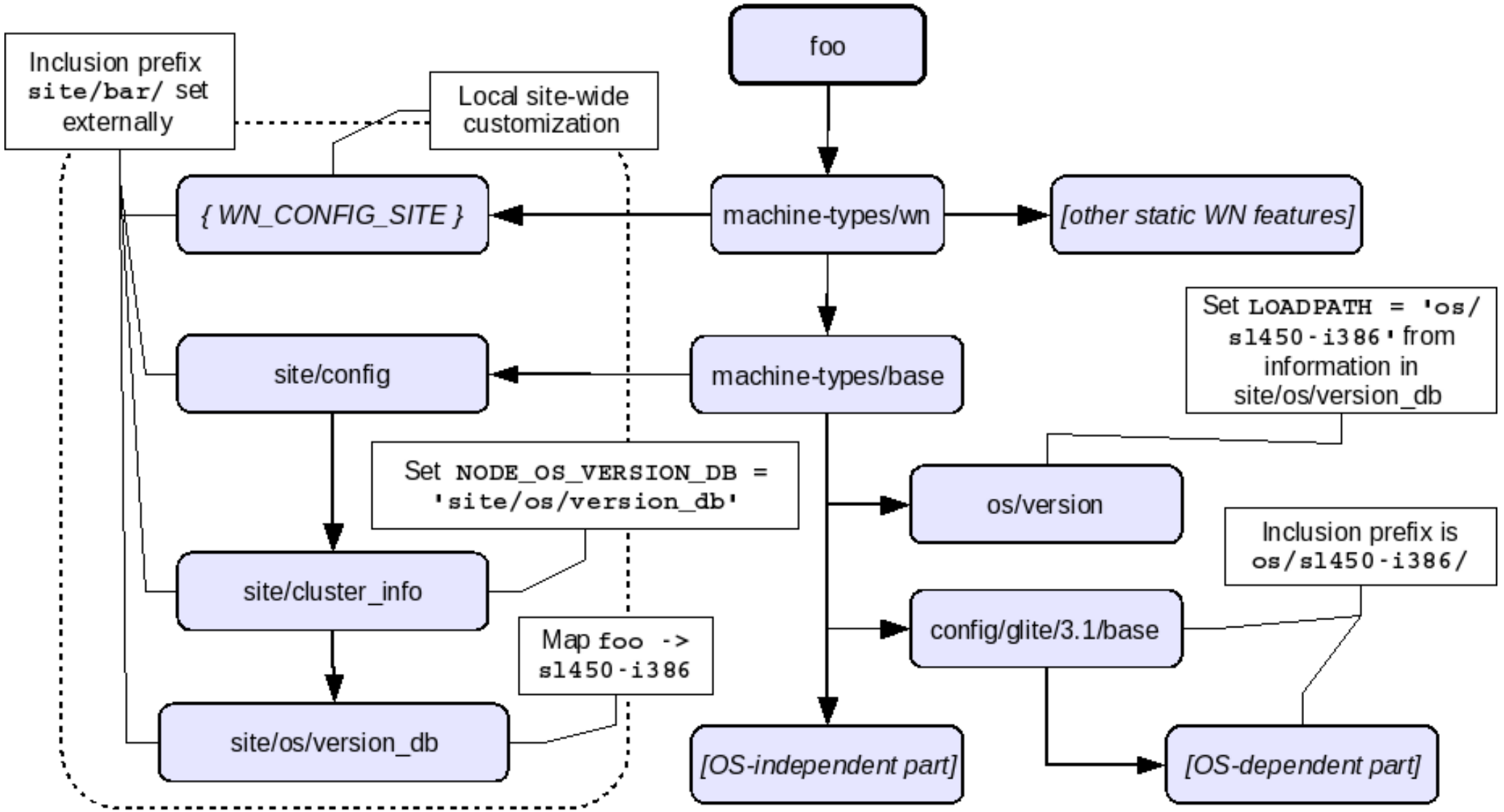
- Sites and clusters configuration at GRIF
  - GRIF is a virtual site containing the base Grid configuration
  - “Includes” is an ordered list defining the precedence in the template search path



...QWG: a configuration distro for devolved management...

quattor

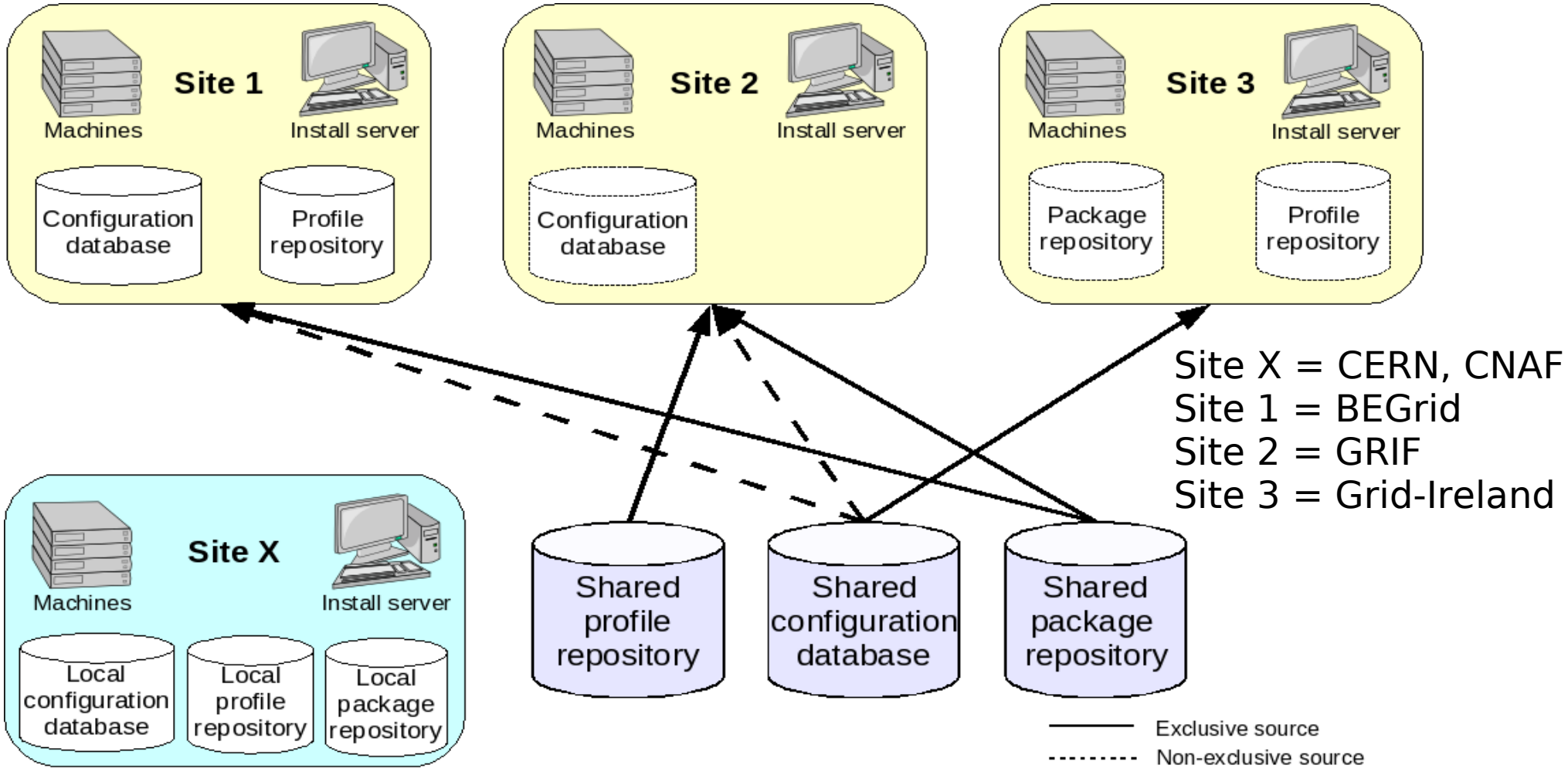
- Local customizations for a worker node *foo* in cluster *bar* at site *LAL*



...QWG: a configuration distro for devolved management

quattor

- Physical deployments of Quattor services
  - Shared configuration can be imported via SVN “externals”



Experience with distributed deployment... quattor

- What worked well
  - Distributed configuration database and disconnected operations
  - Complete representation of desired system state on the nodes
    - Easy detection of misalignments
    - Other nodes' configuration accessible for coordination tasks
  - Namespaces and loadpaths
- Issues
  - The “RPM dependency hell”
    - A tool for pre-deployment checks is under way
  - Administrators need knowledge about many external tools
  - The configuration schema lacks authorization constructs
  - Debugging still a tricky business
    - Improved visualization tools are under development

...Experience with distributed deployment

quattor

- Lessons learned
  - Stability and backward compatibility is paramount
    - Core QWG templates based on wrapper functions
  - Low-effort mechanism for OS/Grid SW updates
    - Single entry point for package updates in QWG templates
  - A community-based synergy improves the project
    - Allows bugs to be quickly located and fixed, despite lack of “core” manpower
    - High probability of finding solutions worked out by some user
    - The community, beside testing, is often the code reviewer

Integration with other tools

quattor

- Quattor is not so invasive as it may seem
  - Only what's expressly defined in the configuration schema is touched on the live system
- Integration means:
  - Preparing a schema and developing Pan templates describing the service
  - Developing “components” to enact the service configuration
- Currently there is support for
  - Monitoring systems (Lemon and Nagios);
  - Virtualization tools (Xen and OpenVZ).
- Moreover, Quattor allows peaceful coexistence with Windows desktops ;-)
  - Respects existing disk partitions and file systems on a node

Conclusion

quattor

- Quattor has demonstrated effectiveness and flexibility in a wide spectrum of site configurations
  - The benefits are clear especially when managing large “farms”
  - The learning curve is rather steep, though the community's support alleviates the pain ;-)
- Pan is the *core* and QWG templates are the *distribution*
- Now deployed in industry as well as academia
- There's still room for improvements:
  - An authorization/entitlement mechanism in the Pan language is under discussion
  - Security is a must. We're working on
    - “safety” wrappers
    - SELinux integration

http://quattor.org/

Management: demanded vs stand-alone

quattor

- Demanded means: a “father” site acts as a service provider to “children” sites
  - Half way between centralized and stand-alone models
- Advantages
  - The configuration base is maintained by developers at the father site
    - children sites get updated automatically
    - Reduced knowledge of configuration's “guts” at children sites ~ reduced manpower
  - Children sites are autonomous for
    - Local customizations (both software and configuration)
    - Also operations, depending on deployment's set-up
- Drawbacks
  - Direct responsibility at the father site for the shared configuration deployment ~ increased manpower

Related work

quattor

- LCFG
  - Lack of overall configuration schema :-(
  - No powerful language constructs :-(
  - validation via PoDIM :-)
- Cfengine
  - Approach is partly “procedural” :-(
  - Difficult to express hierarchical schemes :-(
- Puppet
  - Limited cross-machine validation (“lazy” mechanism) :-|
- PoDIM
  - Powerful built-in conflict resolution :-)
  - Authorization mechanism :-)
  - Performance :-(