Lustre Tools Session

Shine - Administration Tool

Stéphane Thiell, CEA – stephane.thiell@cea.fr

In collaboration with Bull
Jérôme Feyere – jerome.fereyre@bull.net

Contents

- **Introduction to shine**
  - History
  - Project requirements
- **Overview**
  - Code Design
  - Configuration
  - Bull ClusterDB integration
  - Actions
- **Demo**
- **Perspectives**

Introduction to shine, a cluster-wide Lustre administration tool

History

- **2006**
  - Introduction to the **lustre_util** python tool for Lustre 1.4 by Bull at the LUG 2006. Since then, experience highlights:
    - ☛ a set of convenient base commands
    - ☛ the way to an improved tool
- **2007**
  - CEA : shell scripts developed to setup a shared, high availability, Lustre v1.6 petabytes file system
    - ☛ Expertise acquired for large-scale Lustre 1.6 system and storage island
  - Starting Shine project for Lustre 1.6 in collaboration with Bull

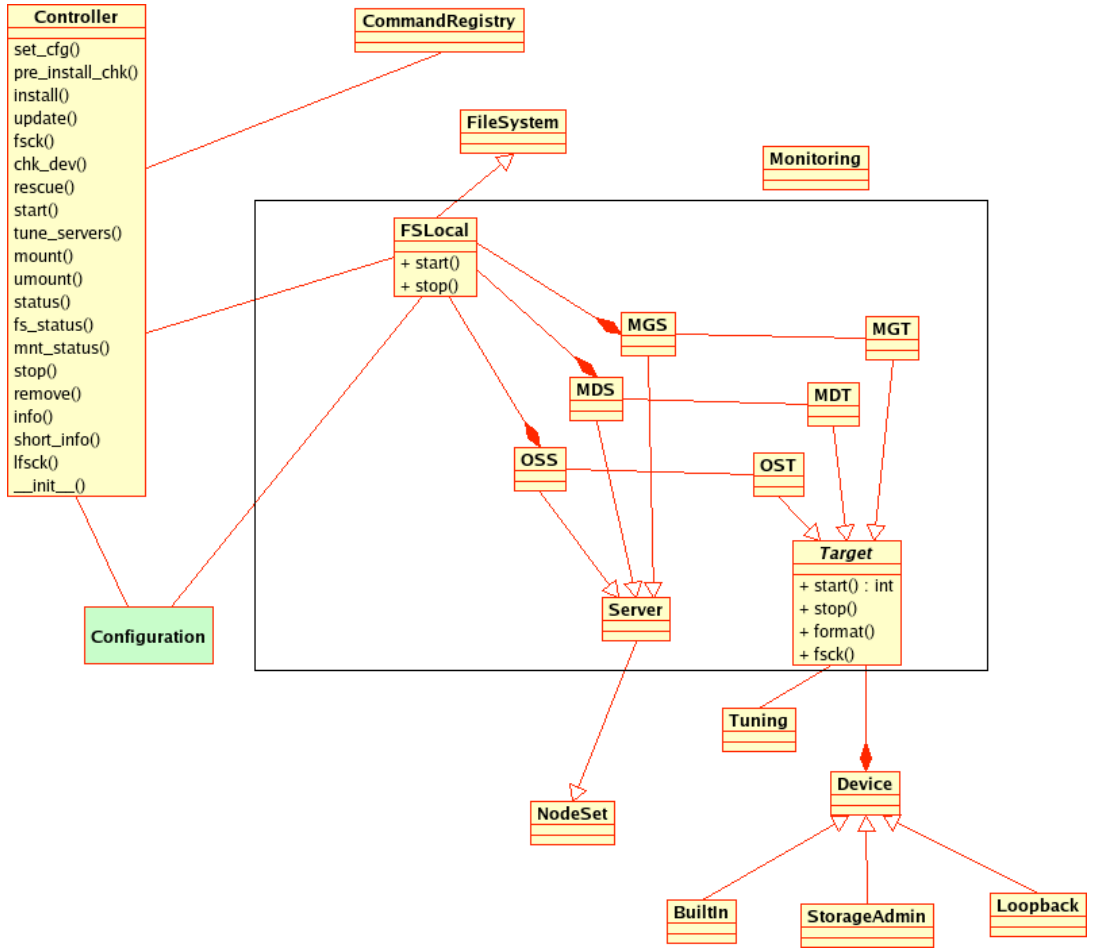
Project requirements

- **Open source**
  - Freely available on Sourceforge.net
  - Under GPL (unchanged since lustre_util was GPL)
- **Readily adaptable**
  - Can quickly conform with Lustre configuration and tuning changes
- **Standalone or Integrated**
  - Use it everywhere in standalone configuration mode
  - Can support vendor cluster configuration like Bull *ClusterDB*
- **Easy to install**
  - Get tarballs or RPMs
  - Simple dependencies on python and pdsh

Overview

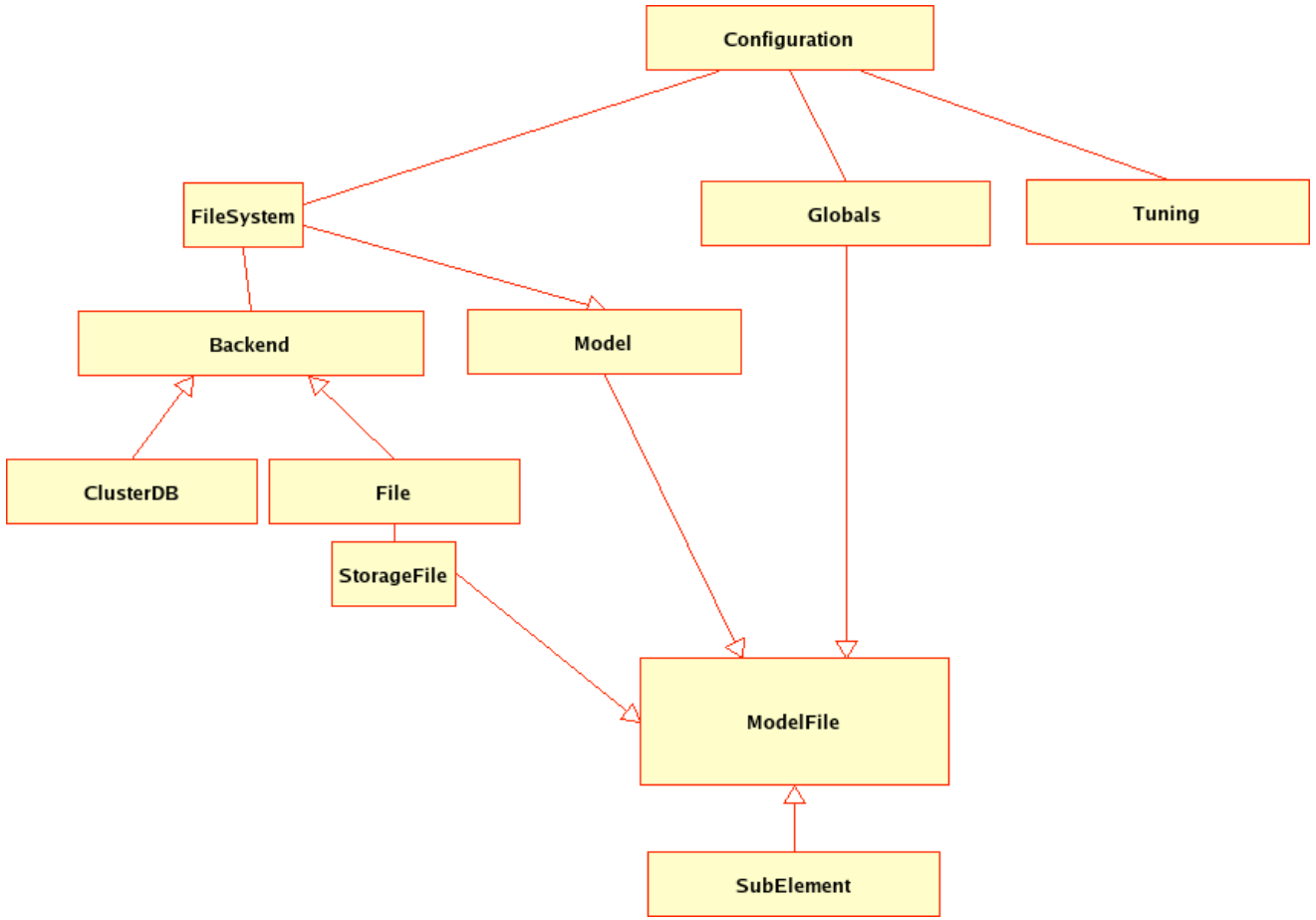
Code Design

- **Object-based design**



Code Design (cont'd)

- **Multiple "backends" in configuration modules**

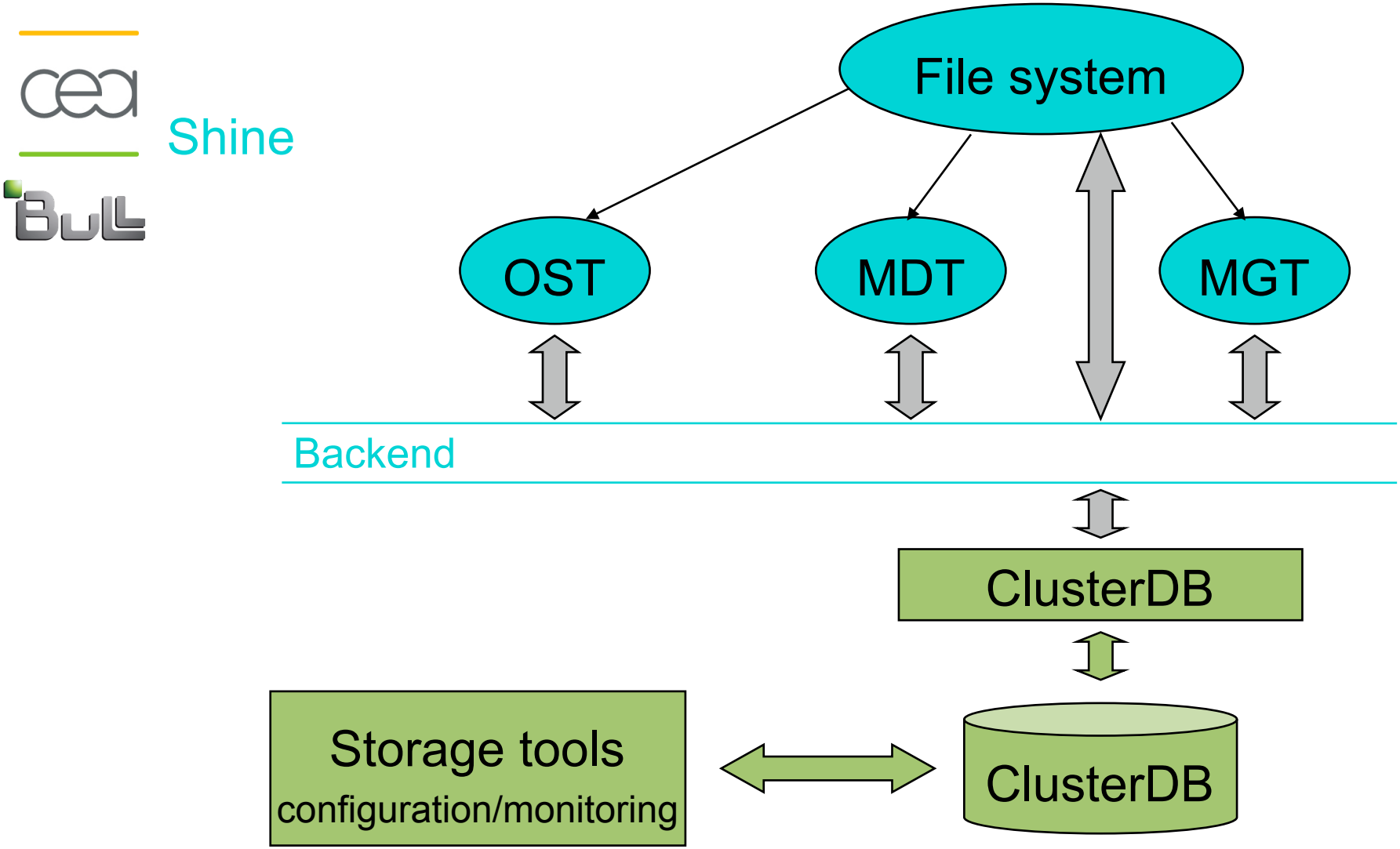


Configuration

- **/etc/shine/shine.conf**
  - General shine configuration information

- **Cluster-wide FS configuration**
  - YAML-like “key: value” configuration files
  - Generated by shine from a FS model file and:
    - a storage configuration file in standalone mode
    - an integrated cluster config info (eg. *ClusterDB*) otherwise
  - Describes the file system with chosen scope:
    - Servers and clients
    - Servers only
    - Clients only

Integrated cluster configuration approach



Integrated cluster configuration approach

- **Database integration**
  - Persistence / integrity of information
  - Database relies on a management station with H.A.
  - Lustre configuration is protected

- **Integrated cluster configuration environment**
  - Storage dedicated tools
    - ☛Provide detailed information on storage component
    - ☛Refine Lustre target state information
  - More precise information available for Shine without complexify the code
  - Use Lustre Targets information to configure HA on IO-nodes

Actions

- **Base shine commands**
  - Install
    - Built file system configuration file from model and storage devices
    - Get file system ready (format, tuning)
  - Start, Stop
    - Start, stop servers (MGS, MDS, OSS)
  - Mount, Umount
    - Mount, unmount file system (clients)
  - Status
    - Get file system live status information
    - Choose specific view of interest (targets, clients…)
  - Show, Info, List

Demo

Perspectives

- **Full lustre_util compatibility**
  - Add lustre_util commands : update, remove, fsck, tune_servers, etc ...
- **File system coherency**
  - Add lfsck support
  - Compare Lustre status in /proc to cached shine status
    - ☛ Get the list of mounted clients from there

Perspectives (cont'd)

- **Event-based Shine API**
  - Build your event-driven scripts with Shine python API
  - Use it to create dynamic web pages to administrate Lustre
    - Adapt existing lustre_util web graphical interface to shine
    - Use latest web technologies that support event-based actions (Web 2.0)

- **High availability**
  - Use Shine as a frontend to retrieve File System information for High Availability solution

Questions ?