DMLite Webinar

Alejandro Álvarez Ayllón
on behalf of the LCGDM development team

Overview

- Motivation and goals
- Architecture
- Interfaces
- Available plug-ins
- Installation and configuration
- Using the library
- Writing new plug-ins

Motivation

- Existing LCGDM source code is complex and hard to extend
  - Logic separation difficult to follow
  - Adding new functionality (as new pool types) implies modifying the code in a hundred places
    - Easy to break!
  - As a consecuence, it doesn't attract contributors to the code

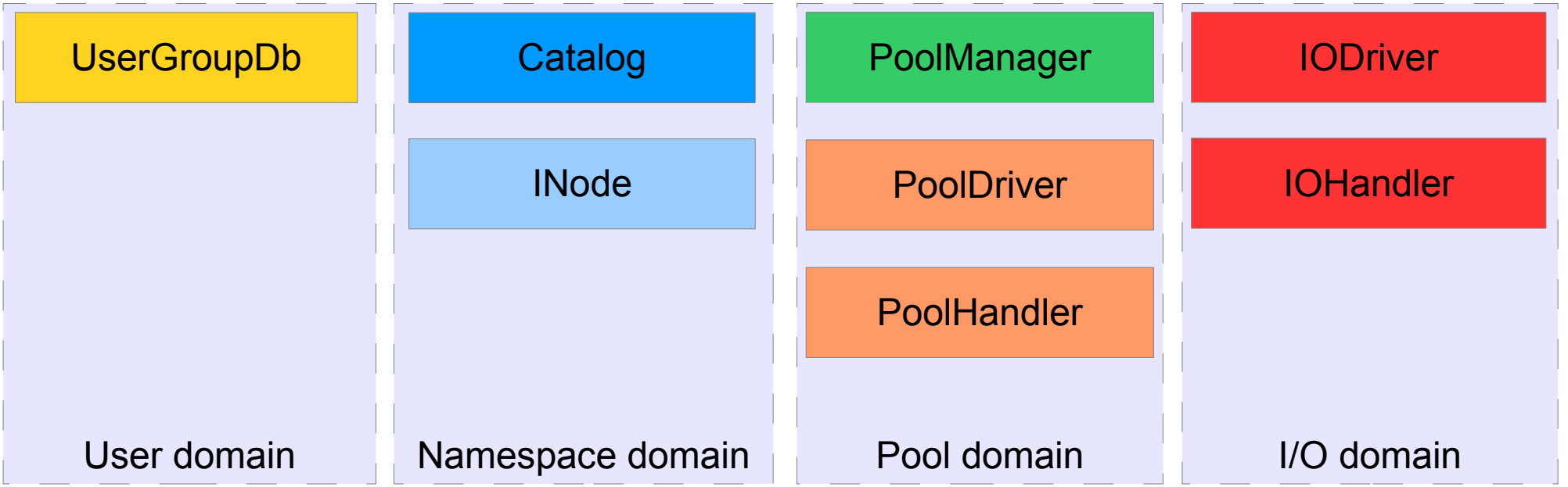
Goals

- Clear code separation
- Reasonably easy to extend
  - But possible to disable new logic if it breaks
- Keep the same behaviour as the existing code as much as possible
- Be compatible!
  - Schema and logic changes must coexist with existing services (DPM, LFC)
- Attract contributors :-)

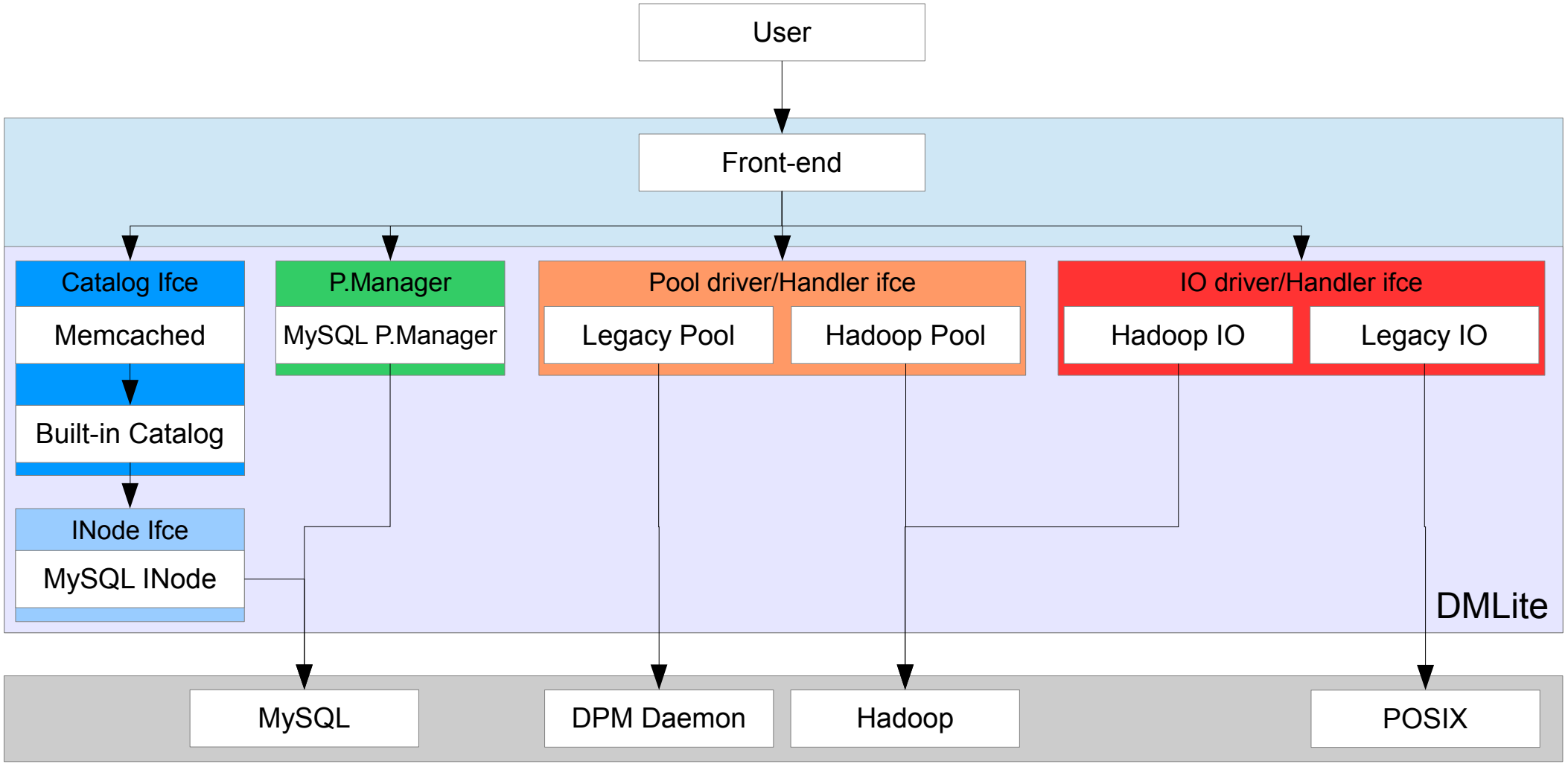
Architecture

- Extensive use of “Abstract Factory” and “Decorator” patterns
  - Same interface, different logic
  - One plug-in can add funcionality (decorate) an underlying plug-in
    - i.e. memcache over MySQL
  - Can be view as a stack collection
- Core and plug-ins written in C++
  - Easier to provide API's in C and Python, eventually in something else (Perl, ...)

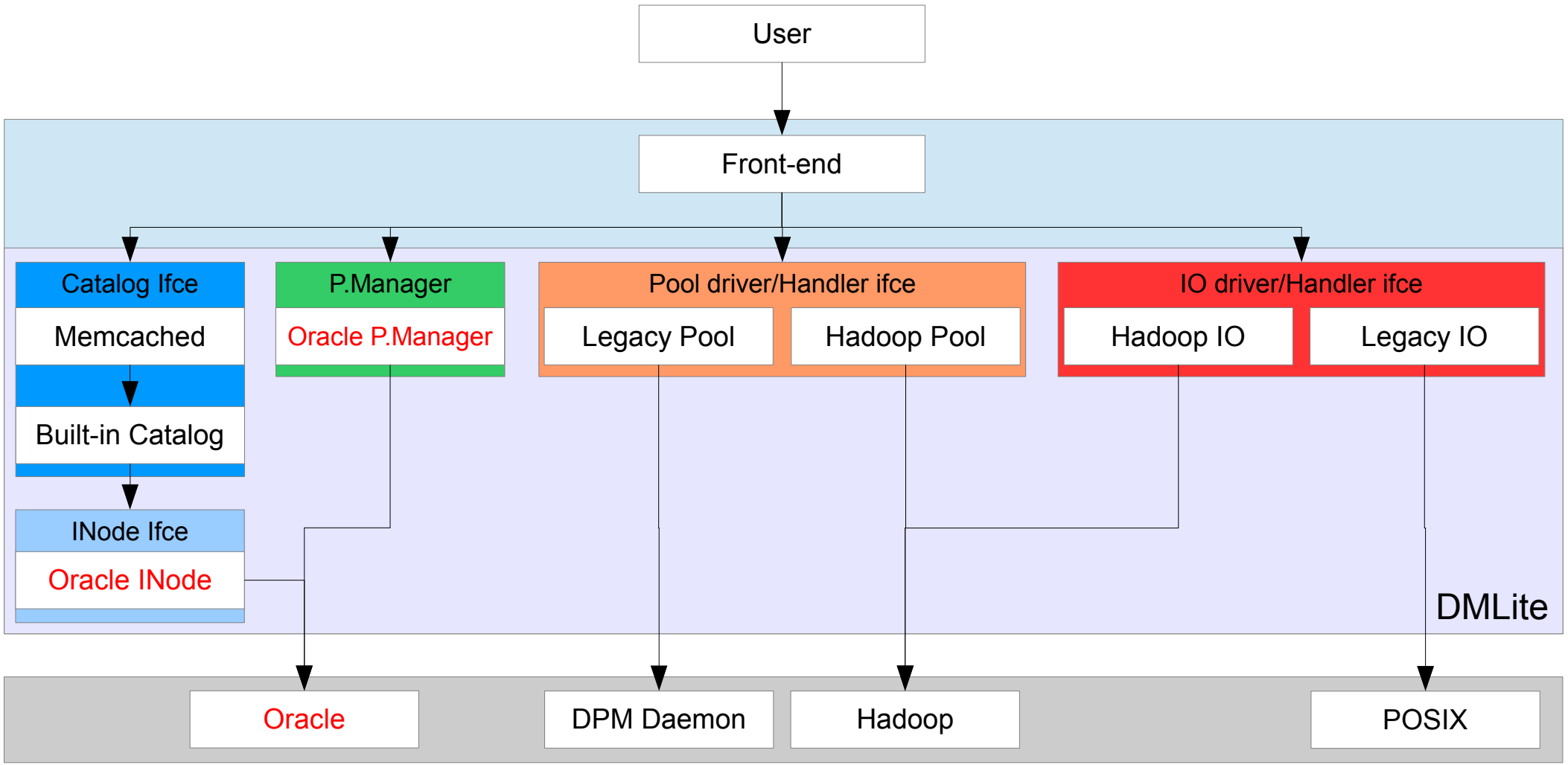
Interfaces


UserGroupDb
User domain
Catalog
INode
Namespace domain
PoolManager
PoolDriver
PoolHandler
Pool domain
IODriver
IOHandler
I/O domain

Stack example



Stack example (II)



Available namespace plug-ins

- Legacy
- MySQL
- Oracle
- Memcached
- Librarian
- Profiler
- Hadoop (coming soon)

Available pool managers

- Legacy
- MySQL

Available pool types*

- Legacy
- Hadoop
- S3 (coming soon)
  - Pool weights need to be added!
- VFS (proof-of-concept)

(*) And their corresponding IO drivers

Other new stuff

- Extended attributes
  - Arbitrary key/value pairs can be associated with files, users, groups or replicas
  - Limited to 1KB

Existing front-ends

- Already in production
  - HTTP/WebDAV
- Coming very soon
  - NFS
- In “proof-of-concept” state
  - SRM 2.2

Installation

- Latest released version: 0.4
  - Available in EPEL and EMI2
- Next release: 0.5
  - It will support the splitting of configuration between multiple files

Installation

- Enable EPEL or EMI2 repository

```
# yum install dmlite-libs dmlite-devel
```

- But that's not enough
  - We need plug-ins that provides some functionality

```
# yum install dmlite-plugins-adapter dmlite-plugins-mysql
```

Configuration

```
LoadPlugin plugin_fs_io       /usr/lib64/dmlite/plugin_adapter.so
LoadPlugin plugin_adapter_dpm /usr/lib64/dmlite/plugin_adapter.so

# This parameter is used by plugin_adapter to know
# to which host it must connect.
DpmHost localhost

# Token generation for Filesystem (and Hadoop)
TokenPassword change-this
TokenId ip
TokenLife 1000
```

Using the C++, C and Python API's

Adding our own plug-in

Help!

- Is more than welcome
  - Developing new plug-ins
  - Developing new front-ends
  - Or extending both
  - Giving a try to the existing front-ends / plug-ins
  - Stress testing
  - Becoming a community project!

Online reference

http://dmlite.web.cern.ch/

Doxygen, Tutorials, Installation and configuration instructions...