

AppFM - Overview

- AppFM stands for Applications Frame Manager
- Main features :
 - 1) Application frame definition (modules)
 - 2) Modules pipeline definition
 - 3) Management of pipeline runs
- Client/Server asynchrone message based Architecture



AppFM WebUI

- hostname:port
- servers : list<hostname:port>




AppFM CLI (cpm)

- server (hostname:port)



AppFM Core

- server specs (nb core, etc.)
-  db (mongo)
- config
- Hostname:port
- API
(- Rest API)





Filesystem

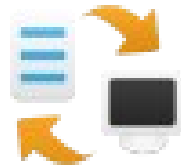
- modules
- corpus
- results/runs



AppFM Module/Service



- module files (binaries, conf, resources, ...)
-  *modulename.module*
-  *Dockerfile (optional)*



Core

- Main Feature (Processus management) :
 - Orchestration (pipelines)
 - Queuing
 - Parallelism
 - Synchrone / Asynchrone modes
- Techs :
 - Restful Message Server (zmq sockets)
 - Scala (jvm 8) + Python Module wrapper



Clients / Interfaces

- Command Line Interface (python module)
 - Basic bash auto completion
 - Allow scripting
- Web UI (one page JS app)
 - Views (corpus files, results files)
 - Module pipelines editor
 - Services iframes
 - Real time process status, notifications



Modules

- Application formal definition (yaml)
 - Lazy input/output type definition
 - Pipeline execution flow definition
 - Built-in operators :
 - Shell command execution
 - Walk/Map parallel execution
 - Condition branching (IF)
 - Docker container integration (better portability)



Services

- Singleton module type
 - Expose global variables usable in modules definition
- Examples
 - Standalone web application (brat)
 - Databases (neo4j, postgres, mongodb, etc.)
 - Application dependency (deepdive)

Roadmap

- Users
- Replica / Distribution
- Services
- Visual modules pipeline edition

Demo / Examples

- Full featured instance
 - Installed on rscpm vm
 - Web ui : rscpm.limsi.fr
 - Docker enabled
- Docker disabled instances :
 - Installed on `/vol/datailes/tools/sys/appfm`
 - Running on labiche and jarry