The ATLAS Software Installation System v2

Alessandro De Salvo
Mayuko Kataoka, Arturo Sanchez Pineda,Yuri Smirnov
CHEP 2015

Overview
Architecture
Performance

LJSFi Overview

- **LJSFi is an acronym of Light Job Submission Framework**
  - Developed in ATLAS since 2003 as a job submission framework for the validation of software releases and other software installation related tasks
  - Evolved with time to cope with the increased load, the use of the WMS and Panda and for HA
  - Using a plugin architecture, in order to be able to plug any other backend in the future

- **Multi-VO enabled**
  - LJSFi can handle multiple VOs, even in the same set of servers

- **Web User Interface**
  - The LJSFi main interface is web-based
  - Users can interact with the system in different ways, depending on their role
  - Anonymous users have limited access, while registered users, identified by their personal certificate, have more deep access

- **Fast job turnaround, Scalability and High-Availability**
  - LJSFi is able to cope with hundreds of resources and thousands of releases, with turnaround of the order of minutes in the submission phase
  - Horizontal scalability is granted by adding classes of components to the system
  - HA is granted by the DB infrastructure and the embedded facilities of the LJSFi components

LJSFi Components

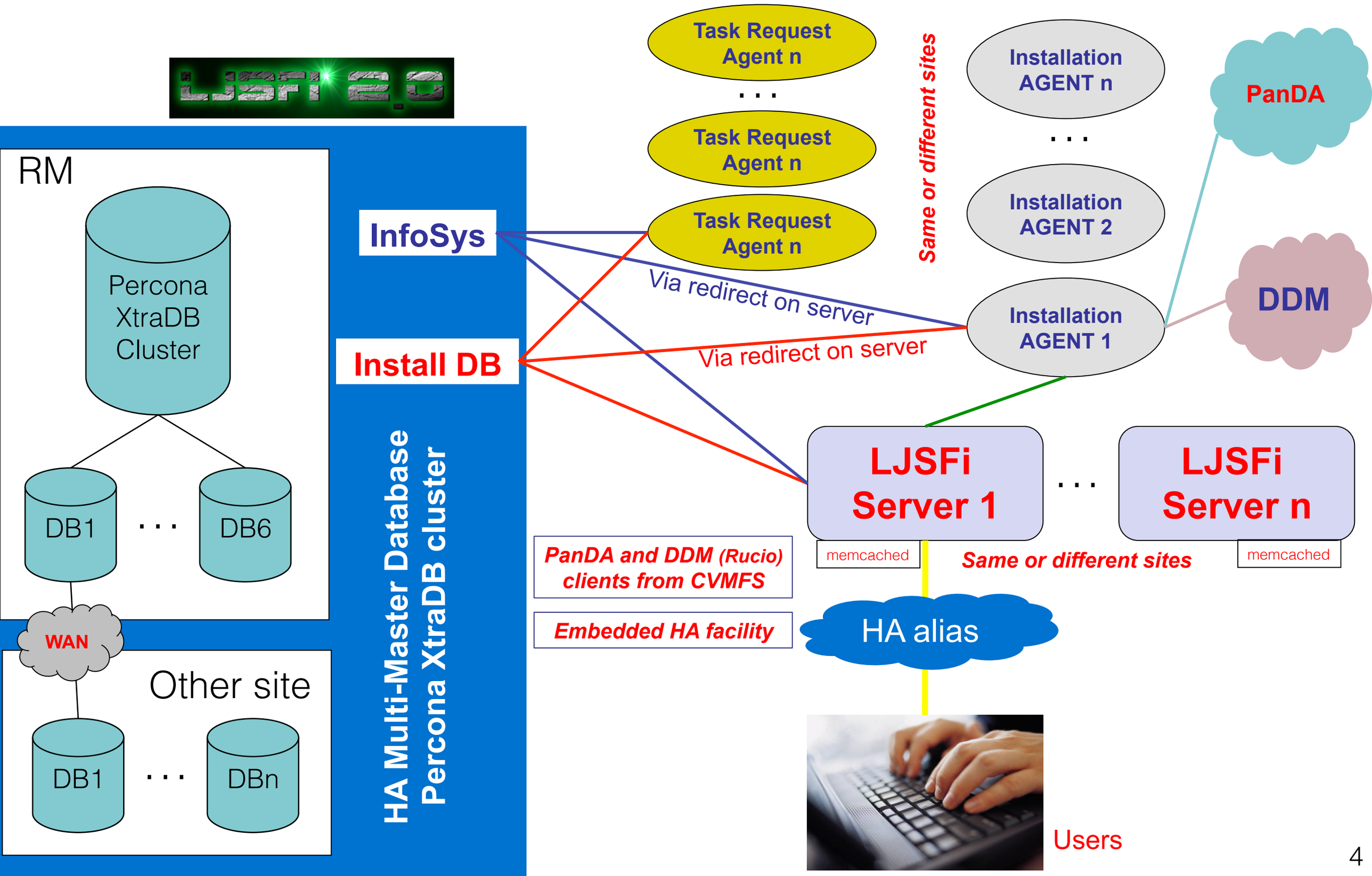
- **The main components of the LJSFi infrastructure are**
  - **The LJSFi Server**
  - **The Request Agents**
  - **The Installation agents**

- **The LJSFi Server is built out of different sub-systems**
  - **The HA Installation DB**
  - **The InfoSys**
  - **The Web Interface**
  - **The monitoring facilities**
  - **The APIs**

- **The Request Agents and Installation Agents are connected to the close Servers**
  - **To request and process the tasks**
  - **To cleanup and notify the admins**

- **In this configuration the failure of a component is not fatal**
  - **Just the logfiles hosted on a failed server won't be accessible**

The LJSFi v2 architecture



LJSFi HA Database

- **Based on Percona XtraDB cluster**
  - Extension of the MySQL engine, with WSREP (Write-Set Replication) patches
  - True multi-master, WAN-enabled engine

- **A cluster of 9 DB machines in Roma + 2 at CERN**
  - The suggested minimum is 3 to have the quorum, and we wanted to be on the safe side!
  - More machines may be added, even in other sites, for better redundancy
  - No powerful machine is needed, but at least 4GB of RAM, 100GB of HD and standard network connectivity (but lower latencies are better)
  - VMs are used at CERN, where we run on the Agile Infrastructure, and no performance issue was seen so far, including the WAN latency

- **Hosting the main DBs used by LJSFi**
  - The Installation DB: the source of release definition for CVMFS and full driver for the ATLAS installations
  - The InfoSys DB: the database used for resource discovery and matchmaking

LJSFi InfoSys

- **Used for resource discovery and matchmaking**
  - **Connected to AGIS (ATLAS Grid Information System) and PanDA**
- **Mirroring the needed AGIS data once every 2 hours (tunable)**
- **Data freshness checks**
  - **No interaction is possible with the InfoSys if the data age is > 4h (tunable)**
- **May use more parameters in the matchmaking than the ones currently present in AGIS**
  - **e.g. OS type/release/version (filled by the installation agents via callback)**
  - **These parameters can be sent to AGIS if needed, as we do for the CVMFS attributes**
- **Sites can be disabled from the internal matchmaking if needed**
  - **For example HPC and opportunitic resources (BOINC), where we should not run automatically the validations as soon as we discover them**

LJSFi APIs

- **LJSFi provides two ways to interact with the servers**
  - The python APIs
  - The REST APIs

- **The python APIs are used by the LJSFi CLI**
  - For the end-users
  - Used by the Installation Agents and Request Agents too

- **The REST APIs are used for a more broad spectrum of activities**
  - Callbacks from running jobs
  - External monitoring
  - CLI commands / Installation Agents
  - Internal Server activities

LJSFi Request Agents

- **The LJSFi Request Agents and responsible of discovering new software releases and insert validation requests into the DB**
  - **Using the infosys and the matchmaker to discover resources not currently offline**
  - **Handling the pre-requirements of the tasks, like**
    - **installation pre-requisites**
    - **OS type, architecture**
    - **maximum number of allowed concurrent jobs in the resources (Multicore resources)**
    - **…**
- **The Request Agents periodically run on all the releases set in auto deployment mode**
  - **Currently the loop is set every 2 hours, but will be shortened as soon as we will bring the request agents to multi-threaded mode**

LJSFi Installation Agents [1]

- **Used to follow the whole job lifecycle**
  - Processing Task Requests from the database
    - Collisions control among multiple agents is performed by central locks on tasks
  - Pre-tagging site as having the given software before sending the jobs
    - Only happening if the sites are using CVMFS
    - Almost all the sites are CVMFS-enabled, with a few exceptions like the HPC resources
  - Job submission
  - Job status check and output retrieval
  - Tag handling (AGIS based)
    - Tags are removed in case of failure of the validation jobs or added/checked in case of success

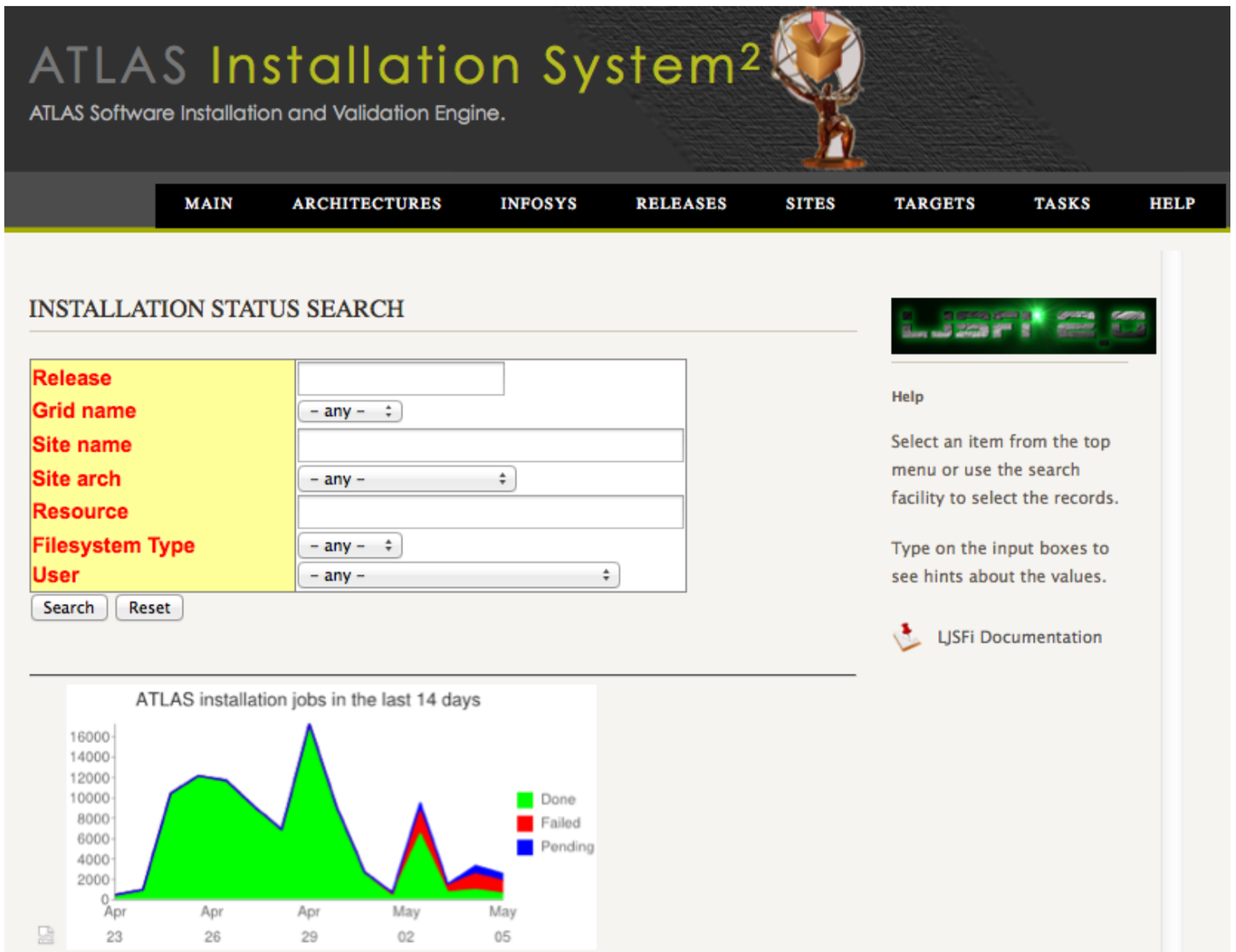
- **The installation agents are fully multi-threaded**
  - Able to send several jobs in parallel and follow the other operations
  - In case of problems, timeouts of the operations are provided either from the embedded commands used or by the generic timeout facility in the agents themselves

LJSFi Installation Agents [2]

- **Several installation agents can run in the same site or in different sites**
  - **Each agent is linked to an LJSFi server, but when using an HA alias it can be delocalized**
  - **Each server redirect the DB calls via haproxy to the close DB machines**
  - **Taking advantage of the WAN Multi-Master HA properties of the DB cluster**
- **Serving all the ATLAS grids (LCG/EGI, NorduGrid, OSG), the Cloud resources, the HPCs and opportunistic facilities via Panda**
- **The logfiles of every job is kept for about a month in the system, for debugging purposes**
  - **Logfiles are sent by the agent to their connected servers**
  - **Each server knows where the logfiles are and can redirects every local logfile request to the appropriate one**

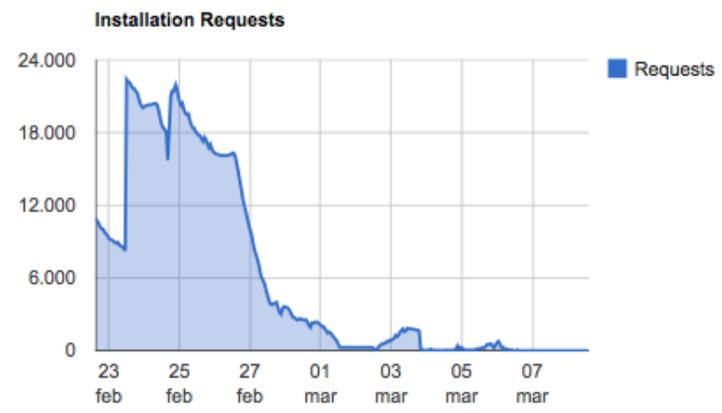
LJSFi Web Interface

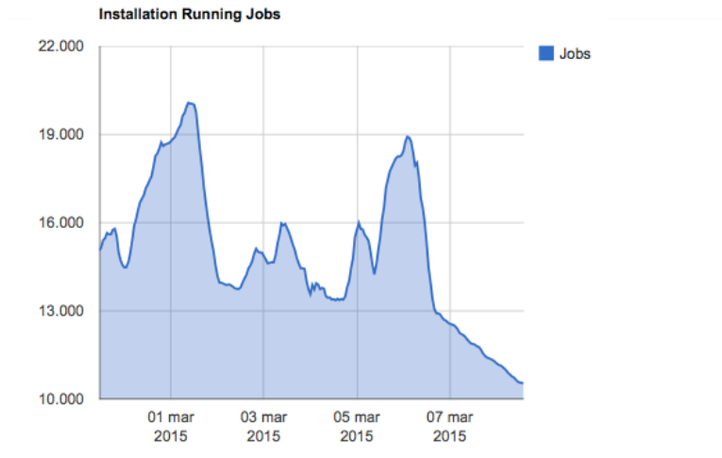
- **The LJSFi Web interface has been designed for simplicity and clearness**
  - https://atlas-install.roma1.infn.it/atlas_install
  - Most of the Input boxes are using hints rather than combo boxes
  - Links to AGIS and Panda for the output resources
  - Friendly page navigation (HTML5)
  - Online Help
- **Each server have a separate Web Interface, but the interaction with the system are consistent, whatever server you are using**

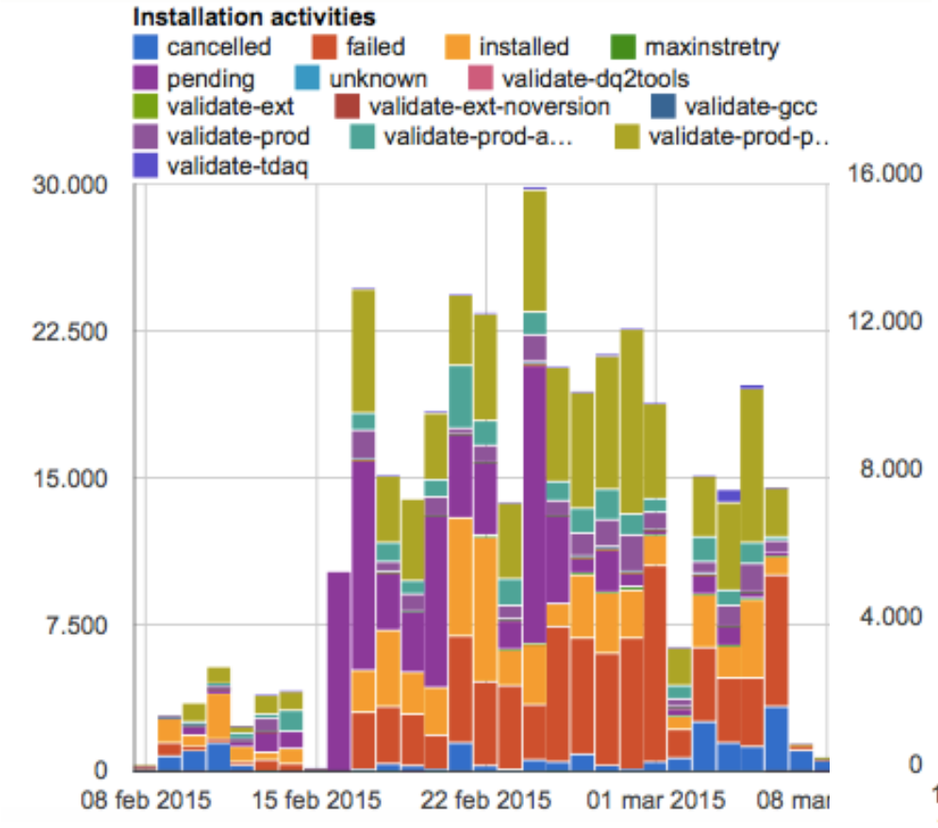


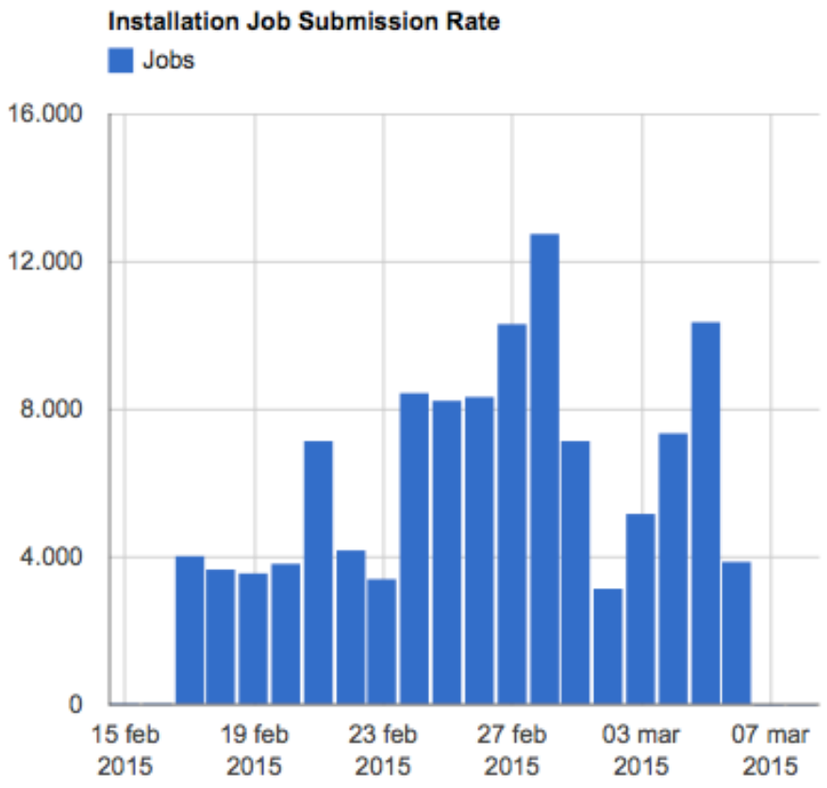
Performance

- **The system can scale up to more than several thousands jobs per day**
  - The horizontal scaling is granted by adding more agents in parallel and increasing the Database cluster nodes
  - To improve performance a limit on the number of jobs handled by the currently running agents has been set to 4000
  - The system processes new requests before the others, to allow a fast turnaround of urgent tasks
  - Generally only a few minutes are needed between the task requests and the actual job submission by the agents
- **The system is able to handle a large number of releases and sites**
  - We currently have > 500 different resources and > 1600 software releases or patches handled by the system









Conclusions

- **LJSFi is in use by ATLAS since 2003**
  - Evolved in time from the WMS to Panda
- **Open System, multi-VO enabled**
  - The infrastructure can be optimized to be used by several VOs, even hosted on the same server
- **Currently handling well all the validation jobs in all the Grid/Cloud/ HPC sites of ATLAS (> 500 resources and > 1600 software releases)**
  - LCG/EGI
  - NorduGrid
  - OSG
  - Cloud sites / HPC sites / Opportunistic resources (Boinc)
- **Fully featured system, able to cope with a big load, scalable and high-available**
  - No single point of failure