DM Data Management

CERN IT Department

The CASTOR SRM2 Interface

Status and plans

Towards databases consolidation

*Shaun De Witt (RAL), **Giuseppe Lo Presti** (CERN/IT)*

CERN IT Department
CH-1211 Genève 23
Switzerland
www.cern.ch/it

HEPiX Conference, ASGC, Taiwan, Oct 20-24, 2008

DM

Outline

CERN IT Department

- The SRM2 interface to CASTOR
  - Architecture overview
  - Status
- Current deployment
  - Tier 0
  - Tier 1s
- Future plans
  - For SRM
  - For CASTOR databases consolidation

A Foreword on SRM

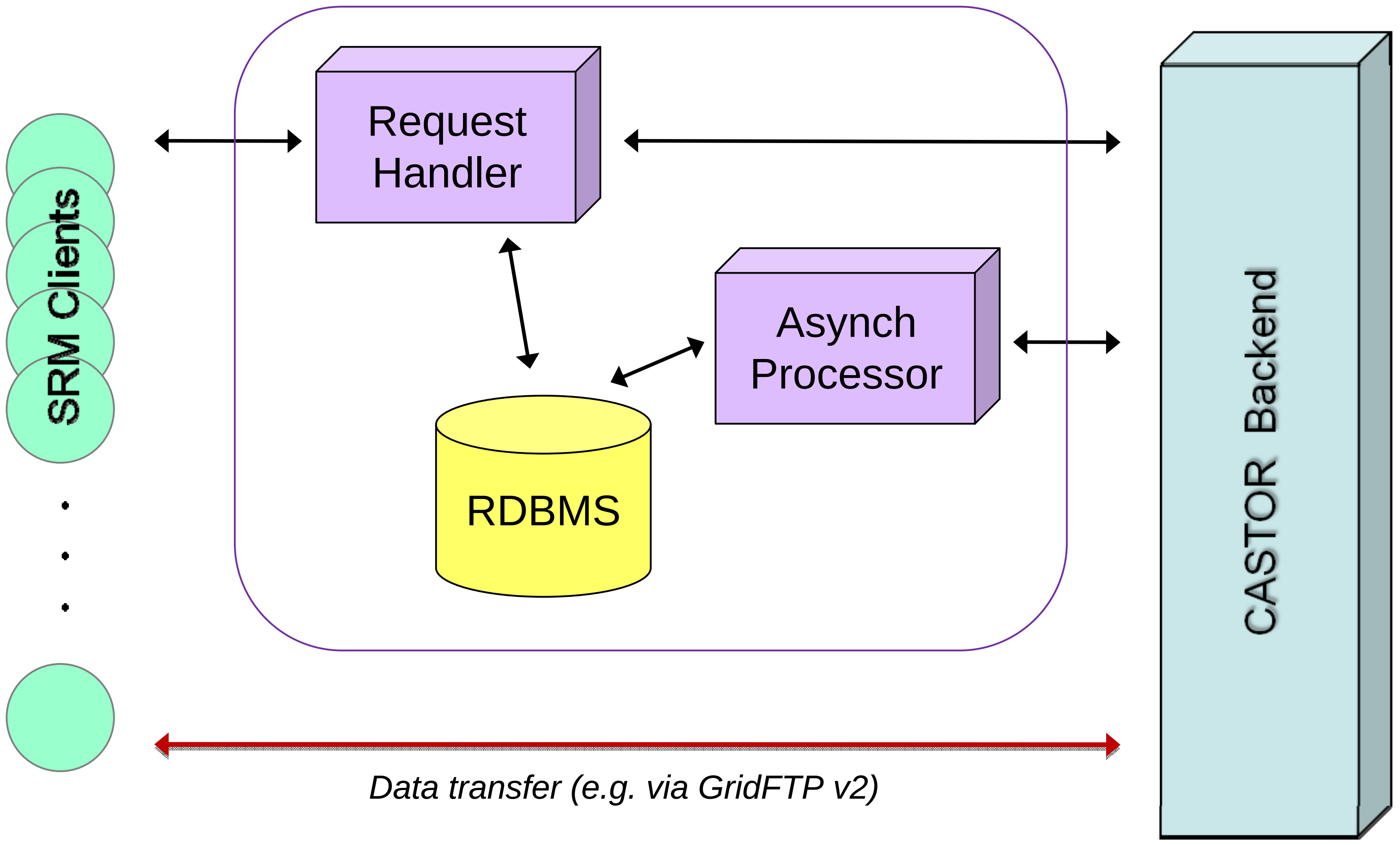
CERN IT Department

- Storage Resource Manager
  - The Storage Element (SE) of the Grid
  - An interface to expose a storage system, hiding the internal details of the different implementations
- Different versions of the specifications exist
  - In particular v1.1 and v2.2
- This talk focuses on the CASTOR implementation of the **v2.2** specifications
  - In particular, including the WLCG MoU document that has been agreed amongst all WLCG SEs implementations

CASTOR SRM2 Architecture

- Database centric
  - Stateless and redundant daemons
    - Ease deployment and scalability
  - Inherits most of the design choices of the CASTOR core system
- A very special CASTOR client
  - The SRM software does not interact directly with the internal components of CASTOR
  - Makes SRM decoupled from the evolution of the internal CASTOR APIs

Architecture Overview



Data transfer (e.g. via GridFTP v2)

SRM Components

- Request Handler (frontend)
  - Performs user authentication
    - GSI-based, can generate high CPU load
  - Serves all synchronous requests
  - Stores asynchronous requests in the database
- Asynch Processor (backend)
  - No user interaction
    - Very low CPU footprint
  - Processes all asynchronous requests
    - e.g. srmBringOnLine
  - Updates db by polling the CASTOR backend
  - Handles srmCopy operations

DM

Current Status

CERN IT Department

- Currently in production: release 1.3 (SLC3)
  – Passed first production tests during CCRC'08
  – Developed and supported by RAL
  – Many known issues
- Being deployed: release 2.7 (SLC4)
  – Major review of the code to address all critical issues faced in the past months
  – Developed by RAL and CERN
  – Towards a release process consistent with the CASTOR core software

Release 2.7 Key Features

- Support for SLC4
- Support for multiple frontends and backends
- Support for the SRM methods agreed in the WLCG MoU addendum (May ‘08)
  - **srmPurgeFromSpace** to allow better handling of Disk1 storage classes
  - **srmCopy** fully functional including support of source space token
  - **srmChangeSpaceForFiles** definitely deprecated
  - improved **srmAbortRequest**
- Improved management of space tokens
  - Admin tools for the service manager
  - Space allocation remains static
- Improved logging
- Improved internal garbage collection
  - Runs as a database job

Current Deployment

CERN IT Department

- At the Tier0
  - One production endpoint per VO, running rel. 1.3
  - Pre-production endpoints per VO being setup, running rel. 2.7
    - Will eventually become the production endpoints
  - Typically **3-4** nodes with DNS load-balanced alias
- At the Tier1s
  - ASGC: one production endpoint for all VOs
    - Upgrade to 2.7 scheduled for this afternoon (!)
  - RAL, CNAF: one production endpoint per VO
    - Running rel. 1.3, upgrade foreseen in the coming weeks
  - Typically **3** nodes with DNS load-balanced alias

Plans and Future Directions

- Short-medium term
  - Deployment and support of release 2.7
  - Stabilizing, addressing remaining less critical bugs
  - Improving logging for better end-to-end monitoring
- Long term vision
  - The choice of an SRM interface loosely coupled with the CASTOR backend proved to be not optimal
    - Processing time often degraded due to the number of layers and interactions between components
    - It is often impossible to achieve the desired behaviour under given borderline conditions (e.g. aborting ongoing transfers)

Plans and Future Directions

- Long term vision (cont'd)
  - A more coupled integration is foreseen, where the SRM frontend daemon directly interacts with the CASTOR main stager database
    - This would enable the stager to seamlessly process CASTOR requests and SRM requests
    - Most of the SRM logic will also be integrated in the CASTOR stager logic
  - SRM will be seen as another gateway to the CASTOR system, similarly to the current CASTOR Request Handler
  - The SRM and Stager database schemas will be merged

Towards DBs Consolidation

CERN IT Department

- The CASTOR system nowadays includes a number of database schemas
  - The CASTOR namespace
  - The CASTOR stager and disk cache management
  - The CASTOR SRM interface
  - The CASTOR VDQM (Volume and Drive Queue Manager) for the tape archive
- It is envisaged to review those schemas in order to improve all internal processing
  - E.g. the namespace, stager and SRM databases share some information and should be merged
  - The VDQM database should evolve to a Tape archive management database

Conclusions

- The CASTOR SRM interface is now a consolidated component of the CASTOR system
  - Used in production (CCRC)
  - New version being deployed
- A development plan has been devised
  - Stabilization of the production release
  - Tighter integration with the stager foreseen
- Comments, questions? www.cern.ch/castor

CASTOR Monitoring

Witek Pokorski, Theodoros Rekatsinas (CERN/IT)

Presented by Giuseppe Lo Presti

HEPiX Conference, ASGC, Taiwan, Oct 20-24, 2008

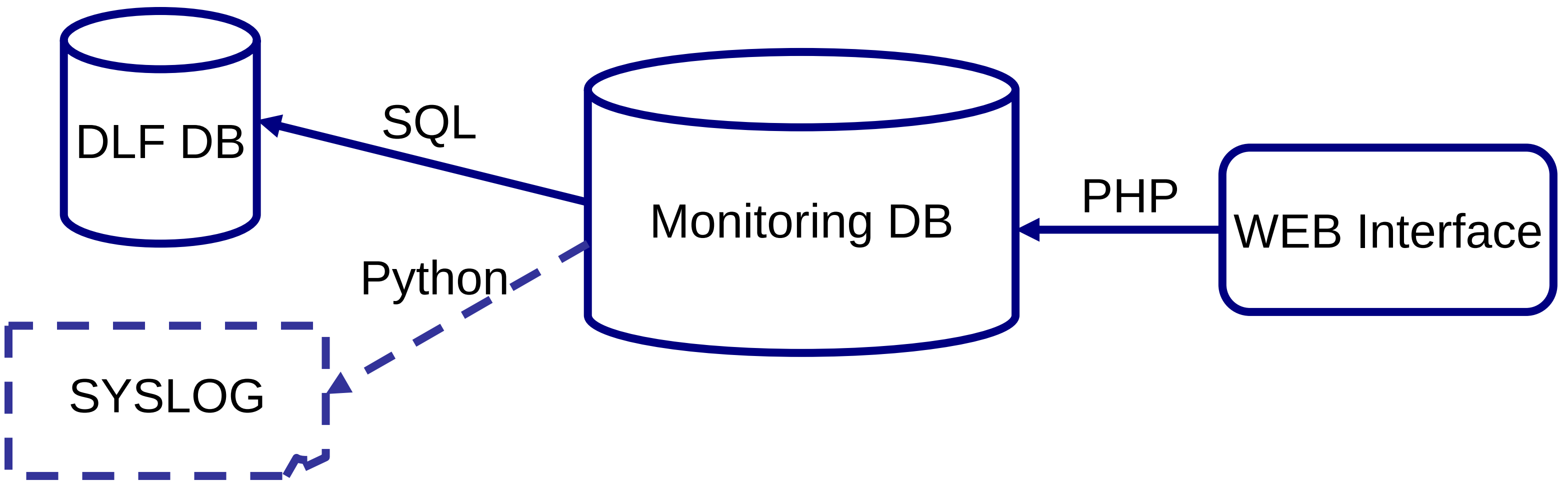
CASTOR end-to-end monitoring

CERN IT Department

- Motivation
  - Provide the users with a *dashboard* presenting the usage of a CASTOR instance
  - Provide the developers with a mean to monitor the running of the system from the point of view of the specific implementation
- Scope
  - Provide end-to-end monitoring covering all the CASTOR sub-components
    - gather the information from SRM, Stager, Nameserver, Garbage collector, diskservers, tapeservers, etc, all together
  - Combine and extend currently existing monitoring infrastructure
    - based on Lemon and SLS at CERN

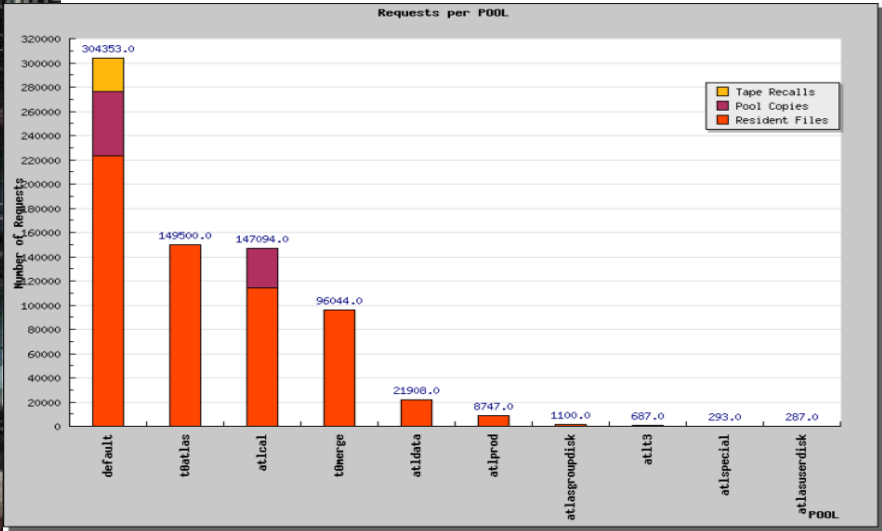
CERN IT Department
CH-1211 Genève 23
Switzerland
www.cern.ch/it

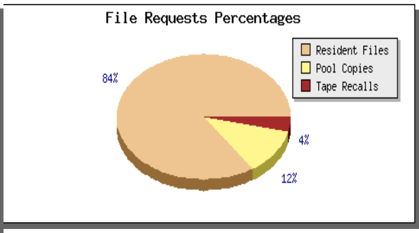
CASTOR monitoring architecture

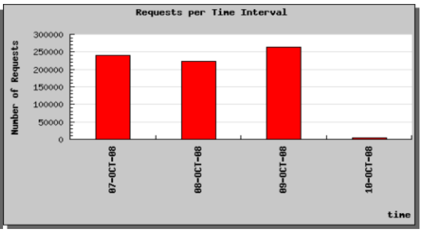


- All the monitoring information comes from log messages sent by different CASTOR components
- Current implementation uses DLF (Distributed Logging Facility)
- Monitoring DB designed in a way to allow quick queries from the web interface
- Future implementation will be based on SYSLOG allowing more efficient real-time parsing of the incoming messages

CASTOR monitoring web interface

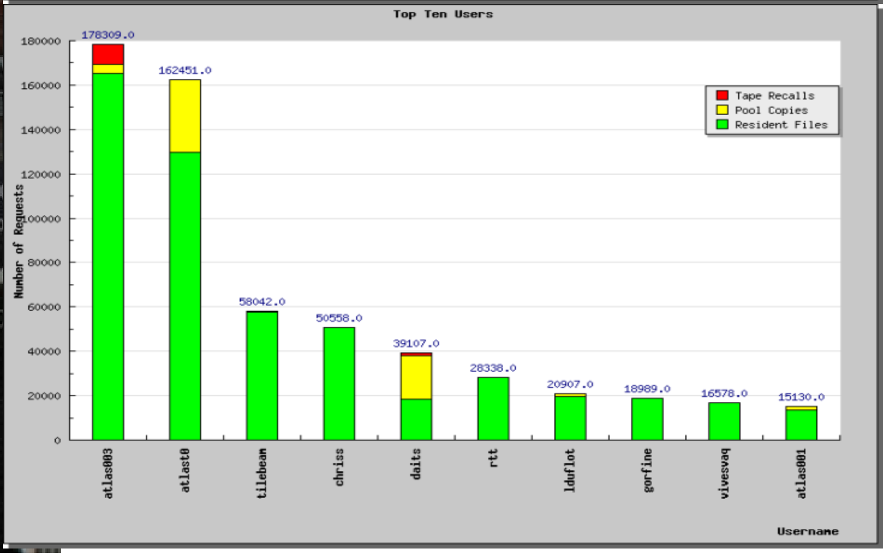


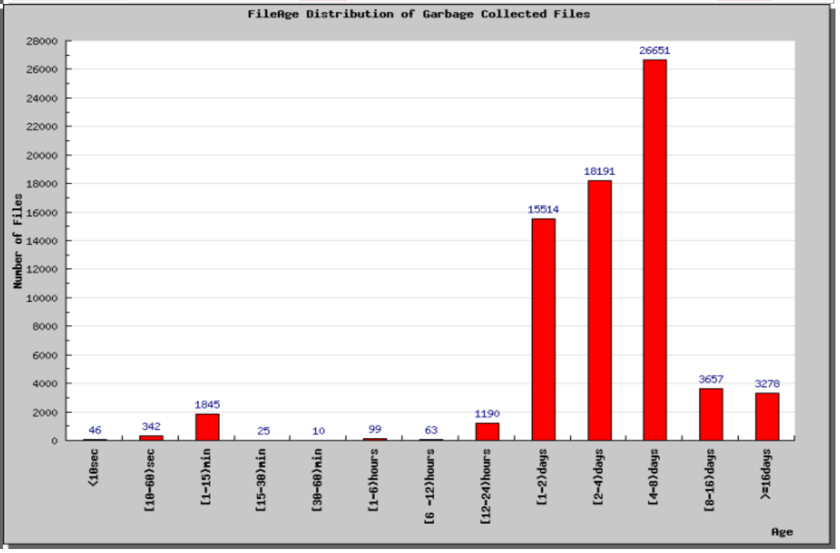




Transaction Between Pools (Number of Files Copied)

FROM - TO	atlasuserdisk	atlprod	atlcal	atldata	atlt3	t0merge	atlasgroupdisk	atlspecial	default	t0atlas
atlasuserdisk	0	0	0	0	0	0	0	0	0	0
atlprod	0	0	0	0	0	0	0	0	8884	0
atlcal	0	0	0	0	0	0	0	0	14141	20
atldata	0	0	0	0	0	0	0	0	460	0
atlt3	0	0	0	0	0	0	0	0	21	0
t0merge	0	0	0	0	0	0	0	0	339	0
atlasgroupdisk	0	0	0	0	0	0	0	0	0	0
atlspecial	0	0	0	0	0	0	0	0	0	0
default	0	0	4	0	1	0	0	0	0	8
t0atlas	0	0	27617	2	0	0	0	0	25817	0





Conclusion and plans

CERN IT Department

- With the massive data-taking approaching, monitoring is an important element of CASTOR
  - need to extend current infrastructure
- A prototype of CASTOR monitoring is being implemented and is under testing
  - only statistic gathering now, dashboard functionalities to come
- Plans for the future developments include:
  - further extensions of the monitoring DB schema
  - move to syslog-based transport
- Comments, questions?

CERN IT Department
CH-1211 Genève 23
Switzerland
www.cern.ch/it