Deployment and Operation

Olof Bärring
CERN / IT

Castor Delta Review – December 6-7 2006

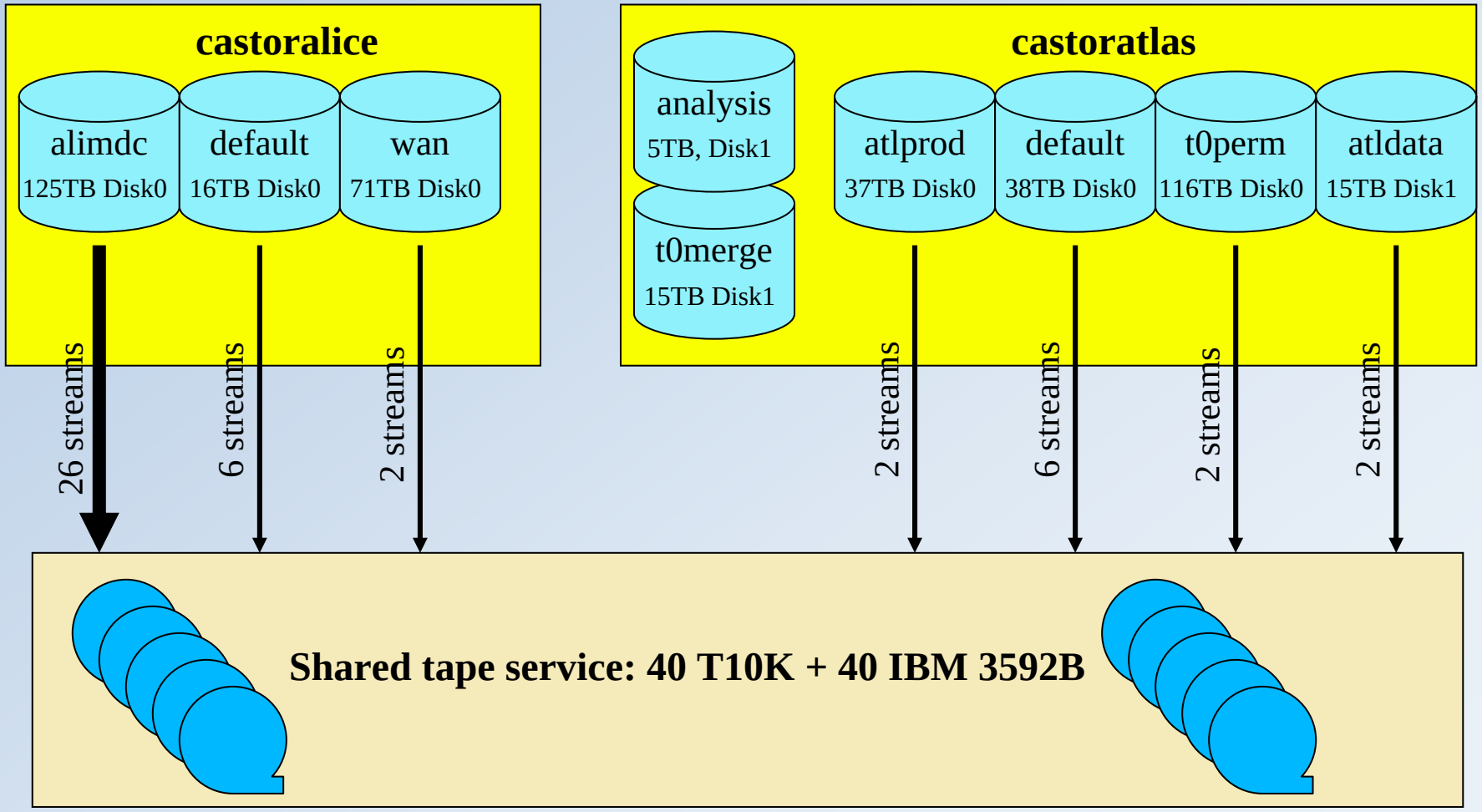
Outline

- Current CASTOR configuration at CERN
- Review of progress since June
- CASTOR and "Disk 1" operation
- Current concerns for future CASTOR operations
- Conclusions

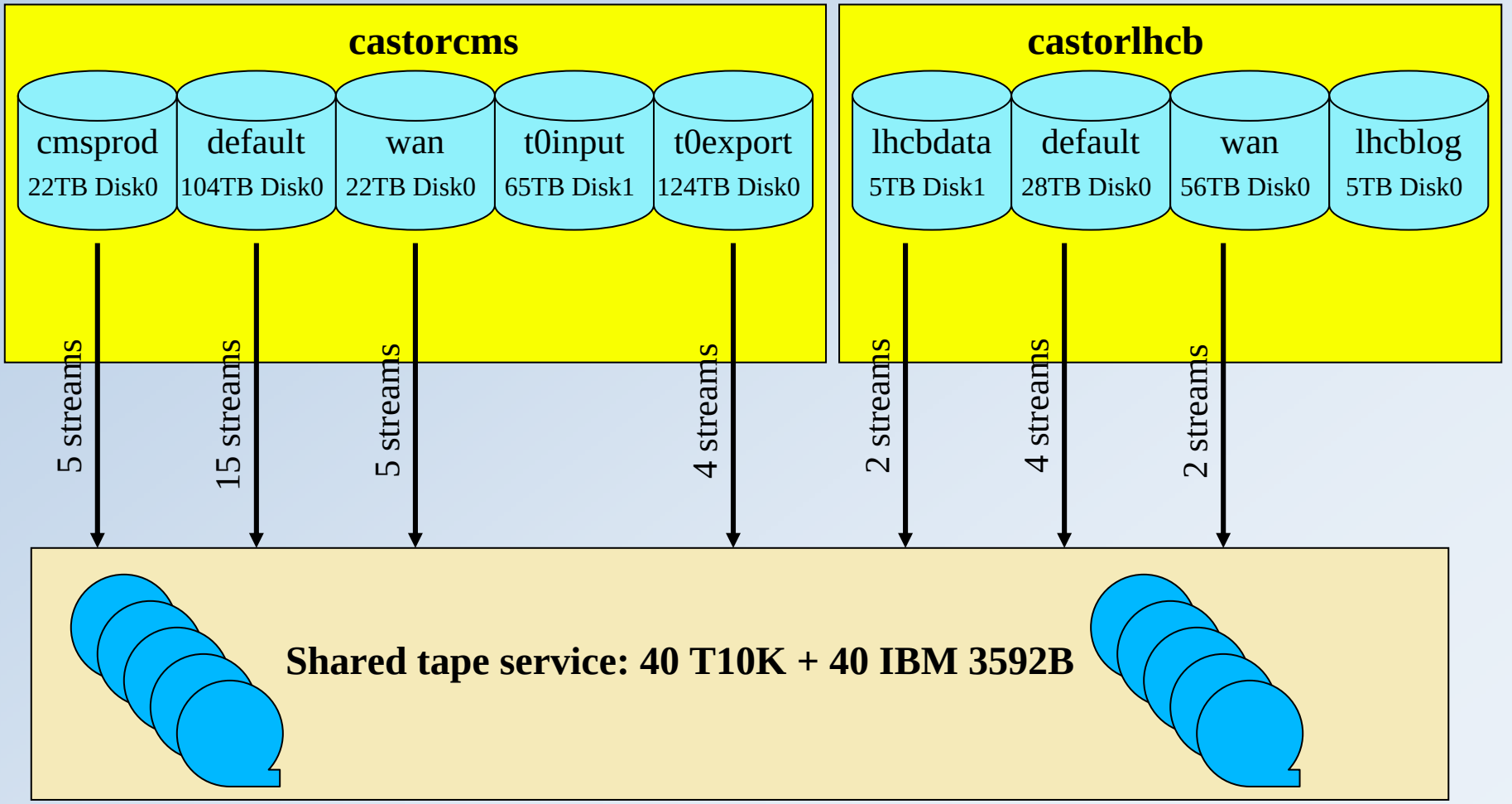
Current CASTOR configuration at CERN

- **CASTOR2 instances**
  - **5 production instances:**
    - **One per LHC experiment: castoralice, castoratlas, castorcms, castorlhcb**
    - **One for all non-LHC (only na48 so far), dteam background transfers and repack: castorpublic**
  - **2 test instances: castortest, castoritdc**
  - **4 development instances**
- **Disk pool sized to meet WLCG MoU**
  - **For some LHC experiments the internal allocations among the different pools matching different activities can be fairly dynamic**

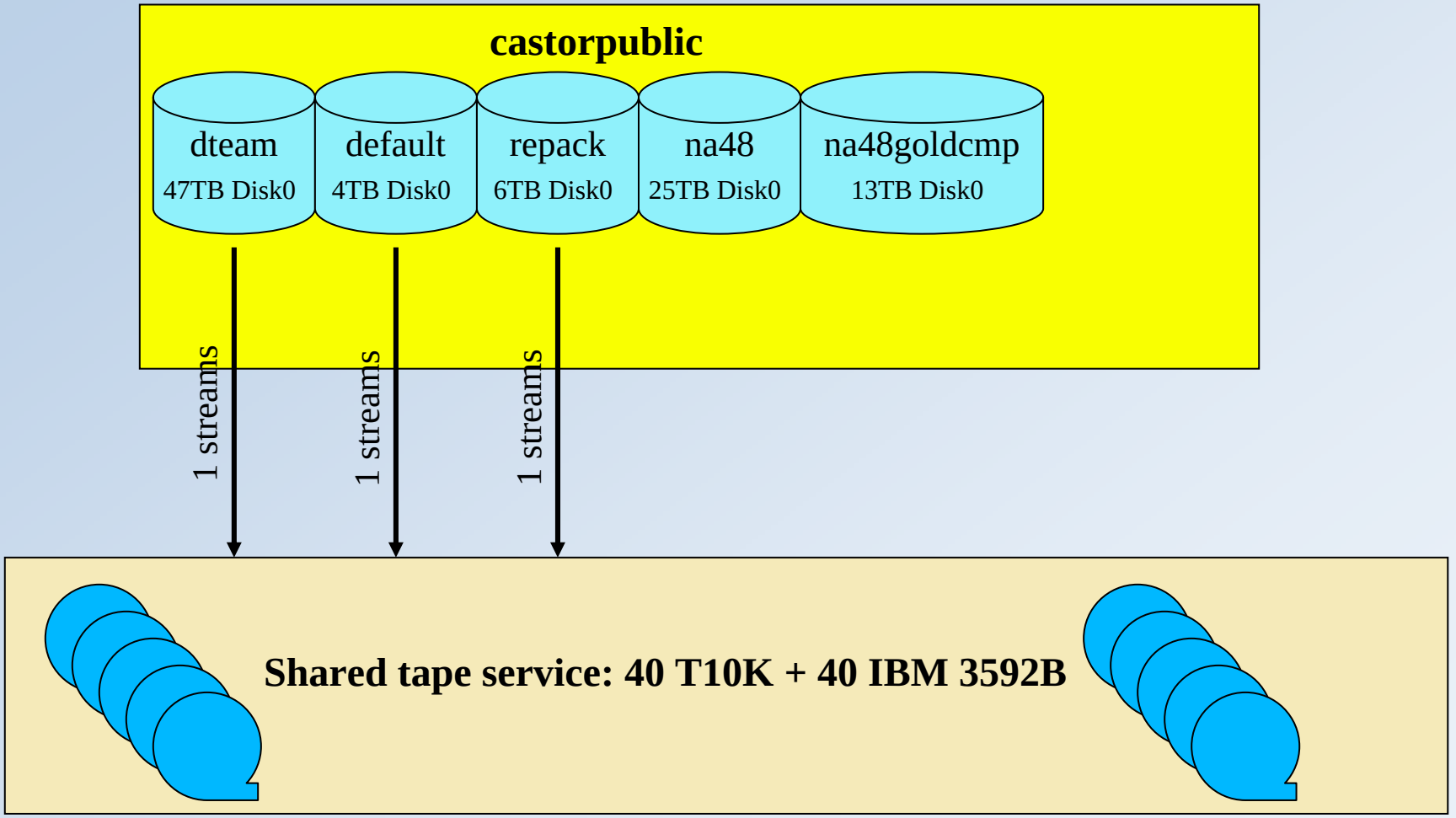
LHC instances, 1/12/06



LHC instances, 1/12/06



Castorpublic: non-LHC instance, 1/12/06



Castorpublic

- A single instance for all non-LHC activities
  - dteam background transfers
  - Repack ~20k 9940B → IBM 3592C / SUN T10K
  - Fixed target experiments
    - NA48 – target date for switching off their castor1 stager: 31/1/2007
    - COMPASS
  - Other non-LHC users (LEP, engineering users, other small collaborations)
- Will only scale to host all concurrent activities with the new LSF plugin + rmmaster
  - Massive repack will only work if LSF job for initiating the tape recalls is removed (work ongoing)
  - COMPASS data-taking for 2007 may require a separate instance

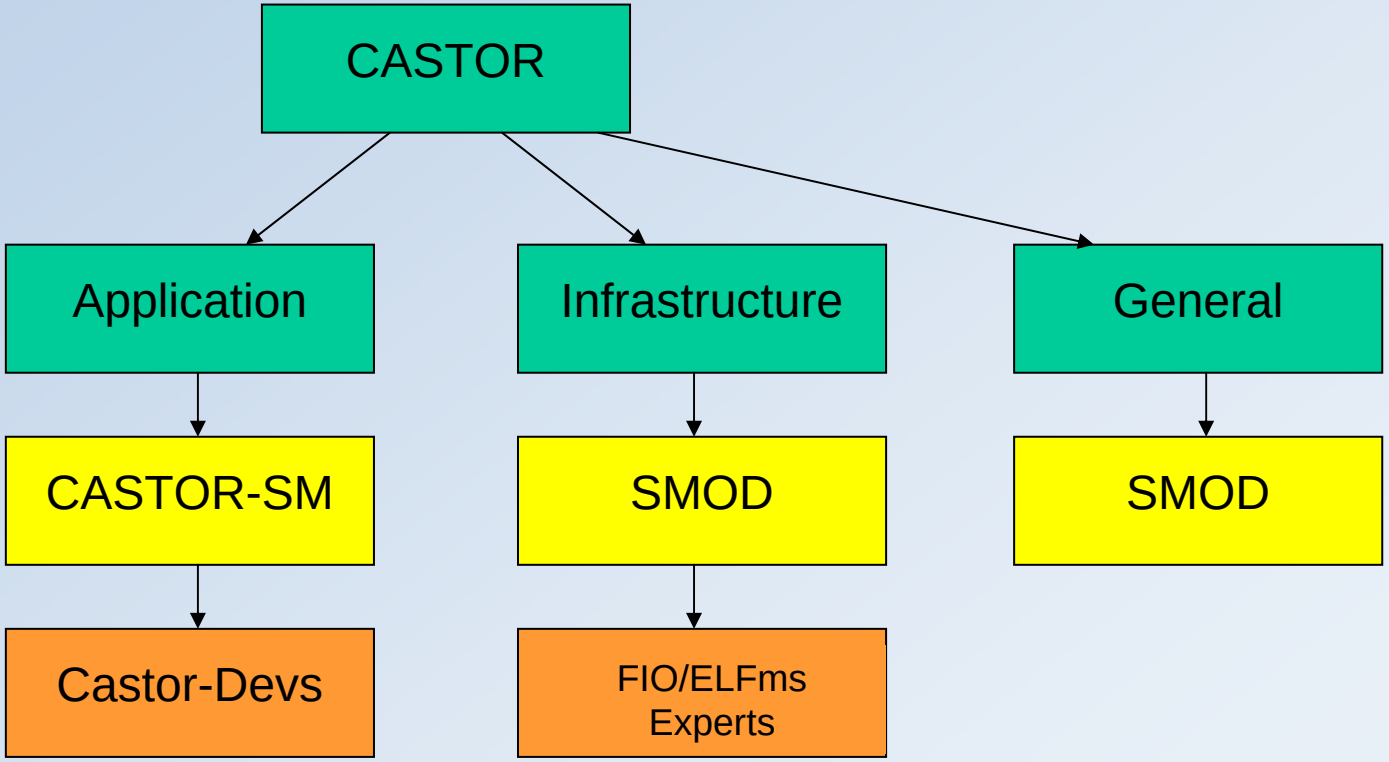
CASTOR2 operation team

- The CASTOR2 operation is part of the Fabric Services (FS) section
  - Groups together CPU and storage services for physics
    - Interactive and batch (lxplus, lxbatch)
    - Specialized clusters for other IT groups and experiments (~125 clusters)
    - CASTOR2 stagers and central services (not tape)
    - Grid services
      - Currently: CE, SE (SRM v11 and v22), WN, LFC
      - To come: FTS, BDII, monbox, gridproxy
  - 7 staff: service management, ELFms developments and supervision
  - 1 fellow and 1 Portuguese trainee: ELFms developments
- Management of a large number of services with a small team of LD staff requires good synergy for how the services are run, maximized automation and well documented procedures

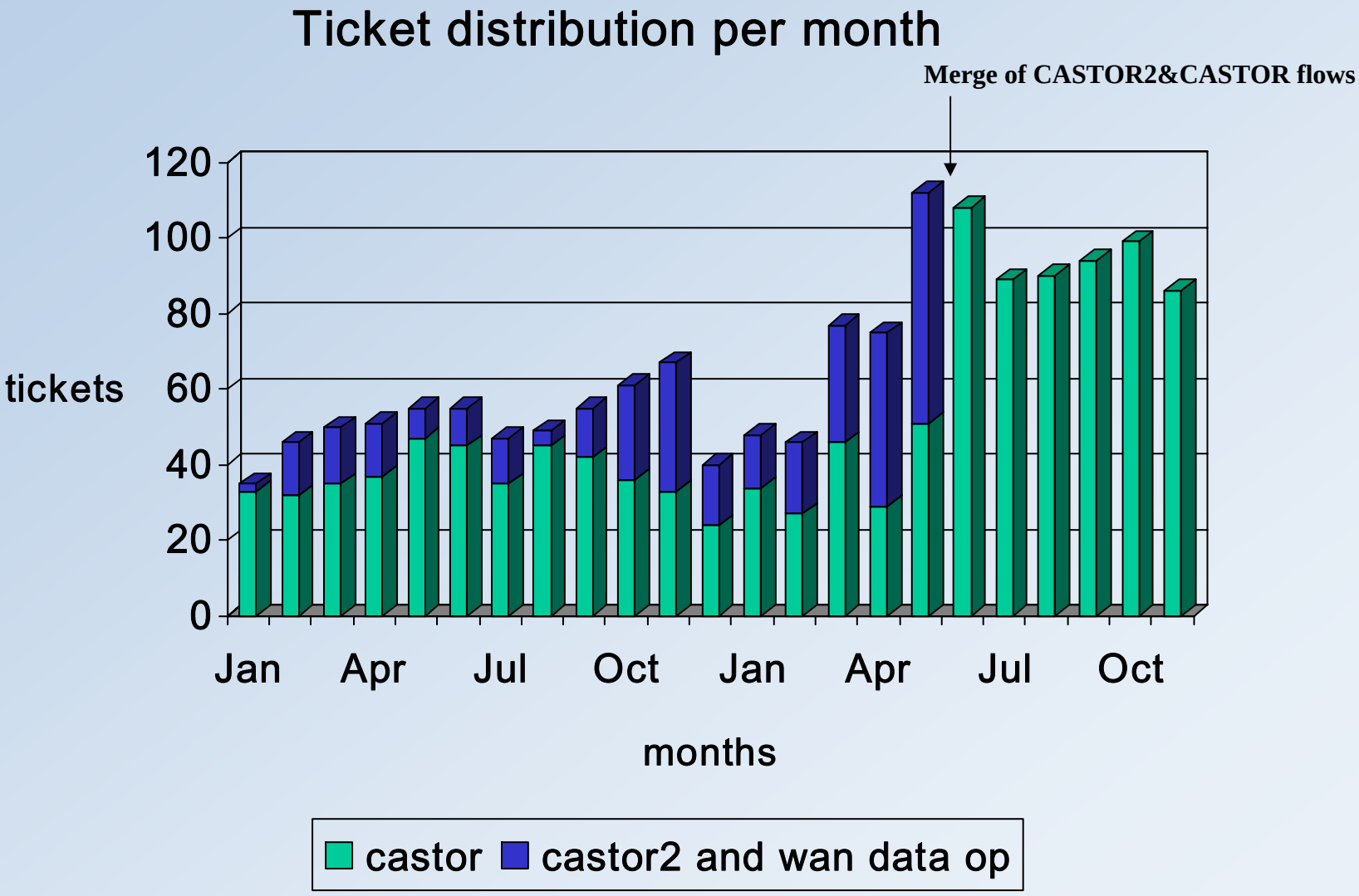
	Interactive, batch	Special clusters	Grid services	CASTOR2	SMOD (service mgr on Duty)
Thorsten	X	X	X		X
Veronique	X	X	X	X	X
Miguel			X	X	X
Jan		X	X	X	X
Ignacio		X	X	X	X
Ulrich	X	X	X		X
Olof (SL)				X	

Reorganization of CASTOR2 support

- Alignment of support flows for all services run by the FS section



Overview of problem handling



Operational improvements since June

- Disk server box management
  - Installation and configuration fully automated
  - Scripted draining procedure for intrusive interventions
    - Still not fully automated due to castor2 bugs (e.g. stager_qry)
    - Medium term goal is to handover the procedure to sysadmins (like it is already the case for CPU servers)
  - State management with standard ELFms tools

Maintenance	No alarms	No CASTOR use at all
Standby	No alarms	Tape migration and replication allowed
Production	Alarmed	Full CASTOR use

  - Host certificate management
    - Monitoring of expiry, automated process for renewal
- Monitoring
  - New DLF allows for implementation of service and accounting metrics (details on following slides)
  - A meeting in November with LHC experiments to review the requirements and agree on metrics

Monitoring - metrics

- New metrics related to the stager & dlf are being implemented at DB level (old metrics are being ported). This makes monitoring with other tools possible. The monitoring agent just needs to sample the DB.
- Various internal metrics were implemented (file age, file access count, file size, request stats, etc)
- Concentrating now on ‘service metrics”. Looking at CASTOR from the user perspective and from the processes the user sees. Example
  - meta-migration has been implemented
    - we try to describe the migration process as seen by the user: file to be migrated or file selected for migration. For each we get avg,min,max size and age in order to produce a list of older files.

Monitoring - Displays

- Currently we run a LEMON dev instance in order to create displays. We expect to migrate to the general LEMON service displays in the beginning of next year.
- Experiments have requested to access metrics both through the LEMON displays and through the LEMON gateway.

Software fixes for operations

- CASTOR op+dev reviewed list of known bugs + workarounds at a dedicated meeting 8/8/06
  - LSF meltdown problem was considered highest priority
  - Looping tape migrators second highest
- Some workarounds are no longer needed or still needed but for different reasons
  - Stager memory leak seems to have been fixed with an updated version of the oracle instant client
    - An attempt to remove the 3hrly restart revealed another problem in the stager: after ~18hrs of running it hit a thread deadlock
  - Stuck disk2disk copy problem may have been fixed in 2.1.0-x but workaround is still in place ‘just in case’
  - New variants of ‘Double LSF jobs for stagerPut’ were recently found
  - GC (garbage collection) problem has been going on since late June. The oracle trigger was for a long while replaced by a workaround oracle ‘cron job’. The trigger was put back in 2.1.1-4

Workaround priority list 8/8/06

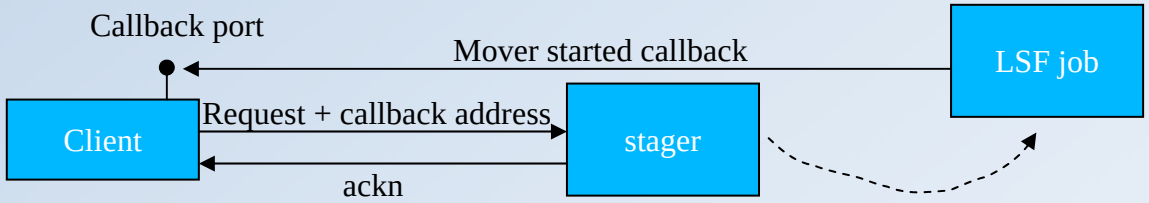
bug	workaround efforts	fix priority (0-10)	Workaround necessity
stager leak	cron-job, minor only during upgrades	2	Memory leak is gone but workaround still needed for different reason (freeze)
stuck disk2disk copy	cron-job	4 (8 for testing if fixed)	Probably not needed
Tapes stuck in FAILED	cron-job	5	Still needed
Thread deadlock in recaller	Manual	3	Still needed
Inconsistency in NBTAPECOPIESINFS	manual, tedious	9	Maybe fixed in 2.1.1-4
Double LSF jobs for stagerPut	manual, tedious, very rare	1	Still needed
rmmaster forks wild	manual, tedious	7	Still needed
rmmaster without persistent state	Disk server state managed in SMS	5	Still needed
GC problems (#17951)	oracle procedure	5	Probably not needed
LSF meltdown	Limit PENDing to 1000	10	Still needed

New problems since June

- Request mixing
- Zero sized files in castor name server
- Stager_qry giving wrong answers
- Problems with putDone
- Looping tape recalls
- LSF jobs without message box 4
- Requests not processed in time-order when resuming from backlog

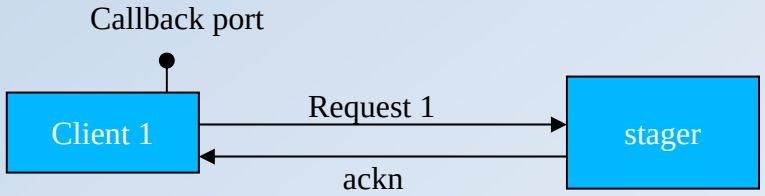
New problems: request mixing

- A potential risk for mixing requests has existed since first releases of CASTOR2 APIs
  - The unique request identifier was not part of the callback → no consistency check
  - If the original client exit (e.g. cntl-C) before the mover callback, a new client risks to re-use the same port (risk ~1/64k)
- The 2.1.0-3 client included a port range, which increased the probability for request mixing in case the original client was killed



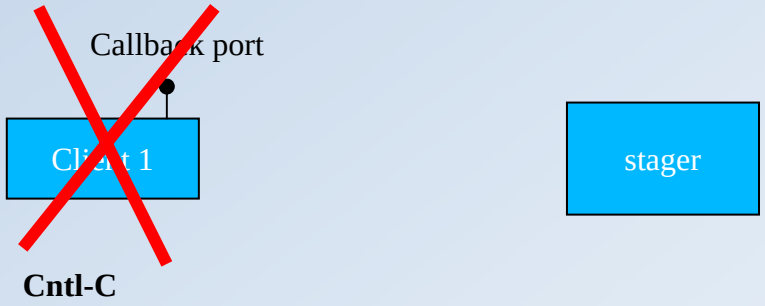
New problems: request mixing

- A potential risk for mixing requests has existed since first releases of CASTOR2 APIs
  - The unique request identifier was not part of the callback → no consistency check
  - If the original client exit (e.g. cntl-C) before the mover callback, a new client risks to re-use the same port (risk ~1/64k)
- The 2.1.0-3 client included a port range, which increased the probability for request mixing in case the original client was killed



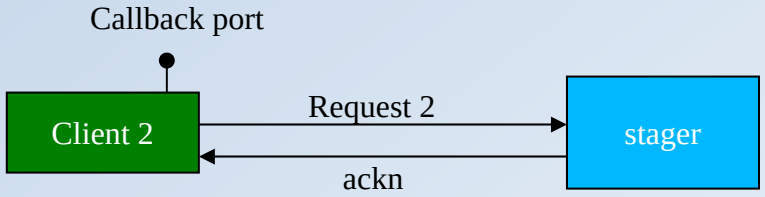
New problems: request mixing

- A potential risk for mixing requests has existed since first releases of CASTOR2 APIs
  - The unique request identifier was not part of the callback → no consistency check
  - If the original client exit (e.g. cntl-C) before the mover callback, a new client risks to re-use the same port (risk ~1/64k)
- The 2.1.0-3 client included a port range, which increased the probability for request mixing in case the original client was killed



New problems: request mixing

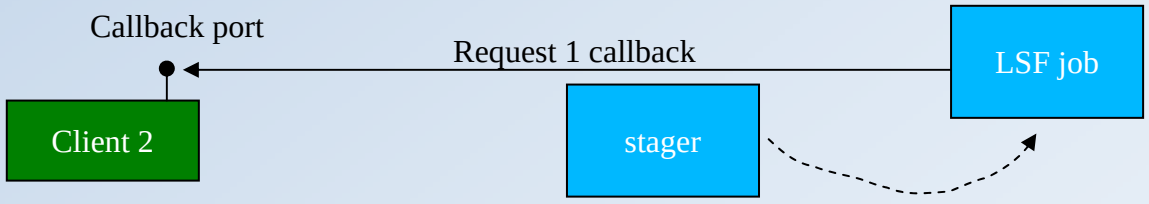
- A potential risk for mixing requests has existed since first releases of CASTOR2 APIs
  - The unique request identifier was not part of the callback → no consistency check
  - If the original client exit (e.g. cntl-C) before the mover callback, a new client risks to re-use the same port (risk ~1/64k)
- The 2.1.0-3 client included a port range, which increased the probability for request mixing in case the original client was killed



In 2.1.0-3 there was a high probability for two clients on the same machine to re-use the same callback port

New problems: request mixing

- A potential risk for mixing requests has existed since first releases of CASTOR2 APIs
  - The unique request identifier was not part of the callback → no consistency check
  - If the original client exit (e.g. cntl-C) before the mover callback, a new client risks to re-use the same port (risk ~1/64k)
- The 2.1.0-3 client included a port range, which increased the probability for request mixing in case the original client was killed



In 2.1.0-3 there was a high probability for two clients on the same machine to re-use the same callback port

Request mixing (2)

- Any CASTOR2 requests could be mixed with different results
  - In the best case, the 2nd request would fail with 'Internal error'
  - In the worst case, when 1st request was a read and the 2nd a write, an existing disk file was corrupted
    - Fortunately an internal check prevented it from being flagged for tape migration…
- Tedious cleanup of corrupted disk files
- Request mixing bug was finally fixed in 2.1.1 release
  - The unique request id (UUID) is passed around between stager and LSF job and checked by the client when it receives the callback

Request 1 \ Request 2	stager_qry	stager_get	stager_put
stager_qry	Wrong answer	Internal error	Internal error
stager_get	Internal error	Read wrong file	**Write to wrong and exiting file**
stager_put	Internal error	Read wrong file	**Write to wrong new file**

New problems: zero size files

- Reason: Oracle deadlock or some other problems when updating the castor name server after a file close()
- Result: file remains zero-size in the CASTOR name server. It will still be correctly migrated to tape.
- Subsequent access may fail if the size is checked
  - rfcp checks that the number of bytes received corresponds to the size. Typically the users see:

    40599847 bytes in 1 seconds through eth0 (in) and local (out) (39648 KB/sec)
    System error : got 40599847 bytes instead of 0 bytes

  - If the file has been garbage collected in the meanwhile, the tape recall will also fail with an Alert for wrong size
- Workaround: manually update the file size in the nameserver.

New problems: stager_qry

- Various bugs in stage_qry causing it to not always telling the truth
  - Repeated entries for same file
  - Files sometimes flagged INVALID even if a valid copy (e.g. CANBEMIGR) existed → causing problems for disk-server draining
  - Files waiting for tape recall (maybe also other blocking operations) not always correctly reported
    - Recently caused tight client stager_get loops and accumulations of requests
  - Query could become very slow when there are large accumulations of (FAILED) diskcopies or requests
    - Fixed in a recent release (2.1.1-x)

New problems: putDone

- The putDone request closes a prepareToPut, Put request cycle
  - Needed when the file is not written to when it is created, e.g. SRM
- The putDone request flags the file for migration and updates the size in the CASTOR name server
- Problems with putDone
  - Runs as a LSF job, which can result in long delays. The associated SRM call (setFileStatus("Done")) is synchronous and risk to timeout
  - If there is an ongoing transfer the putDone will fail with 'Device or resource busy'
  - If the putDone fails for some reason (e.g. Oracle deadlock), the client (srm process) will hang forever → accumulation of srm processes
    - This bug has been fixed in most recent release, not yet deployed
- putDone can probably be further optimized to skip the LSF job
  - LSF job was needed to get a synchronized view of the file size
  - Can probably be avoided, if the Put request updates the file size in the catalogue but some serialization is required for concurrent Puts

New problems: looping tape recalls

- Problem mostly seen for the ATLAS stager
- Reason: stager_rm does incorrect cleanup in case there is an outstanding tape recall
- Result: tape recall proceeds but the catalogue update fails when the file has been copied to disk. The recall is retried, …etc, etc.
  - User cannot access the file
- Workaround
  - Manual and time consuming catalogue cleanup
- The stager_rm bug has been fixed in 2.1.1-4 release

New problems: LSF jobs without msg 4

- Message boxes is a mechanism provided by LSF for adding information to a job before it is dispatched
  - Message box 4 contains the name of the filesystem selected by the CASTOR2 LSF plugin
- If message box is missing the job wrapper is stuck in a sleep(1) loop
  - The loop has been cut to timeout after 10 mins in 2.1.1-x
- Problem rarely seen in 2.1.0-x and appeared to be load related and always intermittent
  - Usually LSF API failing with XDR errors, retried 3 times
- The occurrence seems to have increased for 2.1.1 but with a different failure pattern with systematic failure once it starts
  - LSF API fails with ‘Bad file descriptor’ – new problem with 2.1.1-x and not yet understood
  - Plugin oracle session fails with ‘Invalid cursor’
- Workaround: restart LSF master

New problems: request processing order when recovering from backlog

- When the request rate exceeds the stager processing capacity, e.g. during oracle backups, the backlog is recovered in database order rather than time-order
  - Can cause problem when client interrupts and resubmit the same request for putting a new file into castor
  - A successfully written file may be truncated if the interrupted request happens to start afterwards
    - The name server size remains correct but the original diskcopy is flagged invalid
- Adding a ‘ORDER BY’ is likely to kill the performance under normal operations
  - Use “Advanced queuing” feature in oracle?

Other operational problems

- Tier-2 transfers
  - Routes open for Tier-x (x>1) to all CASTOR2 disk-servers over HTAR (High Throughput Access Route) in May
  - Experiments (LHCb and CMS) reported bad performance July
  - Reasons:
    - One bad HTAR module resulting in single transfer performance drops of a factor 10 – 1000 (sometimes 40k/s instead of 40M/s)
    - Several Tier-2 found with firewall misconfigs and dCache problems (SURL without 'managerv1/srm?SFN=')
  - Tedious and long debugging of non-CASTOR problems
- Loopback interface – iptable interference on SLC3 diskservers
  - CASTOR gridftp V1 implementation uses RFIO over the loopback interface
  - The loopback sometimes block in the middle of the transfer due to bad interference with iptables
    - Reason unknown but only solution seems to be to remove iptables RPM
  - Difficult to debug because of its intermittent nature
  - Hope it's SLC3 related…
  - CASTOR gridftp v2 fixes the problem for good

CASTOR and “Disk1”

- CASTOR2 is designed to work with a tape archive
  - Tape archive is unrelated with “Disk1”.
  - Always do Tape1 (at CERN) because it significantly facilitates disk server management
- CASTOR2 is designed to work with automated garbage collection
  - The after-Mumbai ‘Durable’ concept was renamed ‘Disk1’
  - It is easy to disable garbage collection in CASTOR2 but the operation is difficult because of CASTOR2 design
    - If no space is found, new requests (both read and write) are queued rather than failed → LSF meltdown
    - Read requests for files not in the Disk1 pool results in a tape recall or replication from another diskpool → hidden contribution to used space
    - The listing of disk-resident files is only possible up to a certain limit (30k is a compiled hard limit)
  - The delegation of the space management to the experiments assumes:
    - A serious effort on the experiment side to ‘remember’ what was put in
    - Strict control of how the end-users access the pool

Experience with ‘Disk1’

- ATLAS ‘atldata’ pool
  - Rapidly filled up and caused several LSF meltdowns during the summer
    - Sometimes the only option to unstuck the situation was to add more disk servers
  - ATLAS offline administrators became aware of the problem
    - CASTOR2 operation provided lists of the ~200k files in atldata (since stager_qry is limited)
    - Cleanup and stricter control seems to have been applied
- ATLAS t0merge pool
  - No problems because strict usage
- CMS t0input pool
  - Used during the CSA06
  - Filled up a few times
  - GC trigger was ‘verbally’ triggered: ‘please cleanup all files in /castor/cern.ch/cms/….’
- LHCb lhcbdata and lhcblog pool
  - Never filled up…
  - Do not seem to be used(?)
- Experience at other institutes
  - CNAF has large Disk1 pools, which seems to have been one reason for their problems during summer

Possible improvements that would help the handling of ‘Disk1’ pools

- Always allow for read requests for files which are resident → fixed in 2.1.1-x
- Fail (with ENOSPC) write requests and read requests for non-resident files if ‘Disk1’ pool is full
  - Need to distinguish ‘Disk1’ from ‘Disk0’ before submitting the request to LSF
- Disk pool access control
  - Disallow requests that would result in decreased free space (either new files or replication/tape-recall of existing files) from non-production users
  - Orthogonal to CASTOR file permissions and ACLs

Current concerns for future CASTOR2 operations

- CASTOR2 has proven to meet the requirements for Tier-0 and Tier-1
- The big unknown: does it scale to meet the requirements for chaotic user physics analysis?
  - What are the real requirements for the CERN Analysis Facility (CAF)?
- Current observations
  - Disk mover footprint sets the limit on concurrent transfer slots per disk server
    - Room for some optimizations of rfiod and stagerJob process but it is unlikely to scale beyond 500-1000 slots per GB memory
  - Old model of a forked mover per file access may not scale well for large capacity (30TB) disk-servers

Options for the CAF

- More hardware?
  - Small capacity disk-servers
  - Large memory servers
  - Need to know by ~March - April for 2008 hardware acquisitions
- Other mover protocols optimized for concurrent access
  - CASTOR2/xrootd port may be *the* solution but how would we know?
- Need a changed strategy focus for 2007: move away from Tier-0/1 towards 'CAF/Tier-2 challenges'. Requires resources for
  - Setup and operation of testing infrastructure
    - CASTOR2 instances
    - Disk server hardware
  - Writing and running of test programs?
    - Simulate concurrent, sparse random access patterns
    - Test different protocols, including xrootd
- What are the success criteria?
- Requires involvement and support from LHC experiments

Conclusions

- Despite the relatively long list of new problems/bugs, the CASTOR2 software has definitely undergone a substantial hardening since June
  - The series of successful data and service challenges has proved that CASTOR2 does meet the requirements for CERN Tier-0/1
- CASTOR2 operations has been homogenized with operation of other fabric services, e.g. CPU and grid
  - Exploit synergies using same automation frameworks (ELFms) and identical support flows
  - Limit exposure to LD staff turnover problem
- The CAF (whatever the requirements are) is the biggest future concern