Worldwide distribution of experimental physics data using Oracle Streams

Eva Dafonte Pérez

Database Administrator @CERN

Outline

- CERN and LHC Overview
- Oracle Streams Replication
- Replication Performance
- Optimizations: Downstream Capture, Split and Merge, Network, Rules and Flow Control
- Periodic Maintenance
- Lessons Learned
- Tips and Tricks
- Streams Bugs and Patches
- Scalable Resynchronization
- 3D Streams Monitor
- New 11g Streams Features
- Streams Setups Examples
- Summary

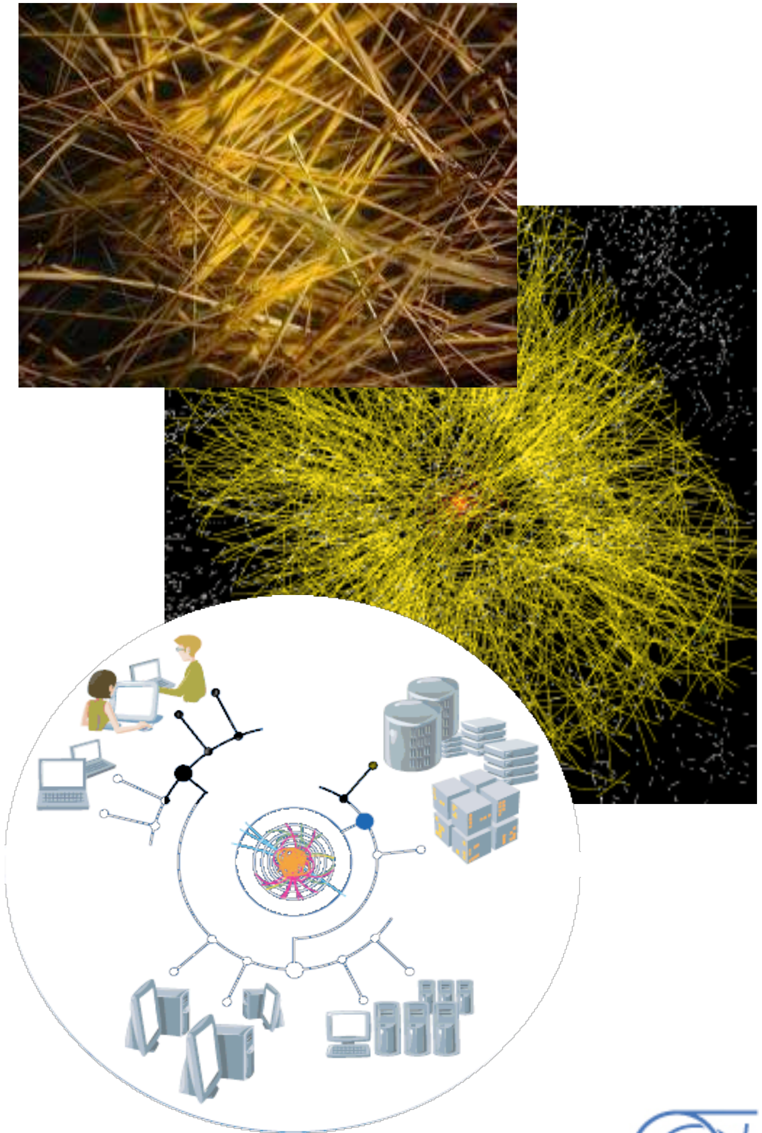
CERN and LHC

CERN IT Department

- European Organization for Nuclear Research
  - world's largest centre for scientific research
  - founded in 1954
  - mission: finding out what the Universe is made of and how it works
- LHC, Large Hadron Collider
  - particle accelerator used to study the smallest known particles
  - 27 km ring, spans the border between Switzerland and France about 100 m underground
  - will recreate the conditions just after the Big Bang

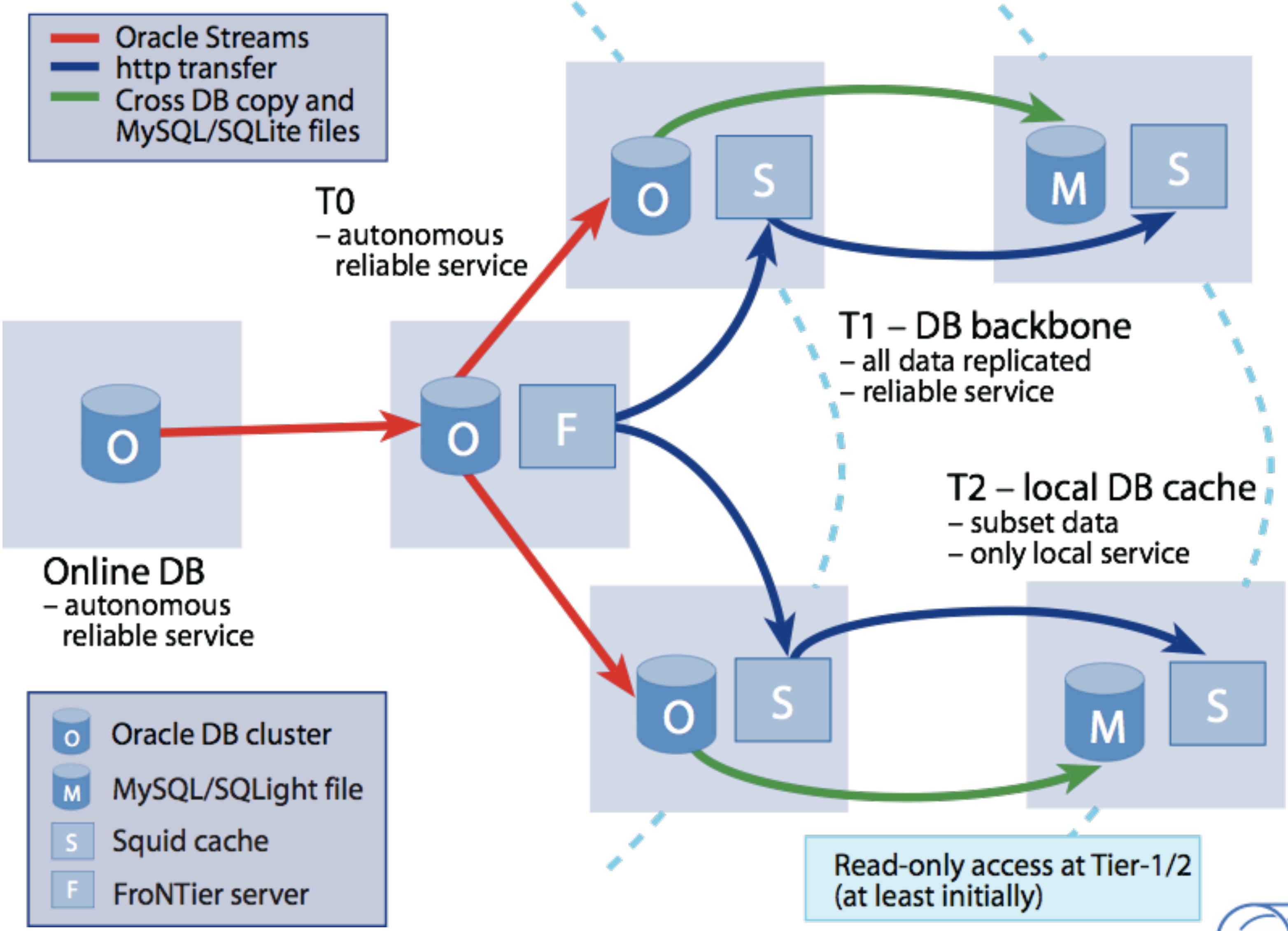
The LHC Computing Challenge

- Data volume
  - high rate x large number of channels x 4 experiments
  - **15 PetaBytes of new data each year stored**
  - **much more data discarded during multi-level filtering before storage**
- Compute power
  - event complexity x Nb. events x thousands users
  - **100 k of today's fastest CPUs**
- Worldwide analysis & funding
  - computing funding locally in major regions & countries
  - efficient analysis everywhere
  - **GRID technology**



Distributed Service Architecture

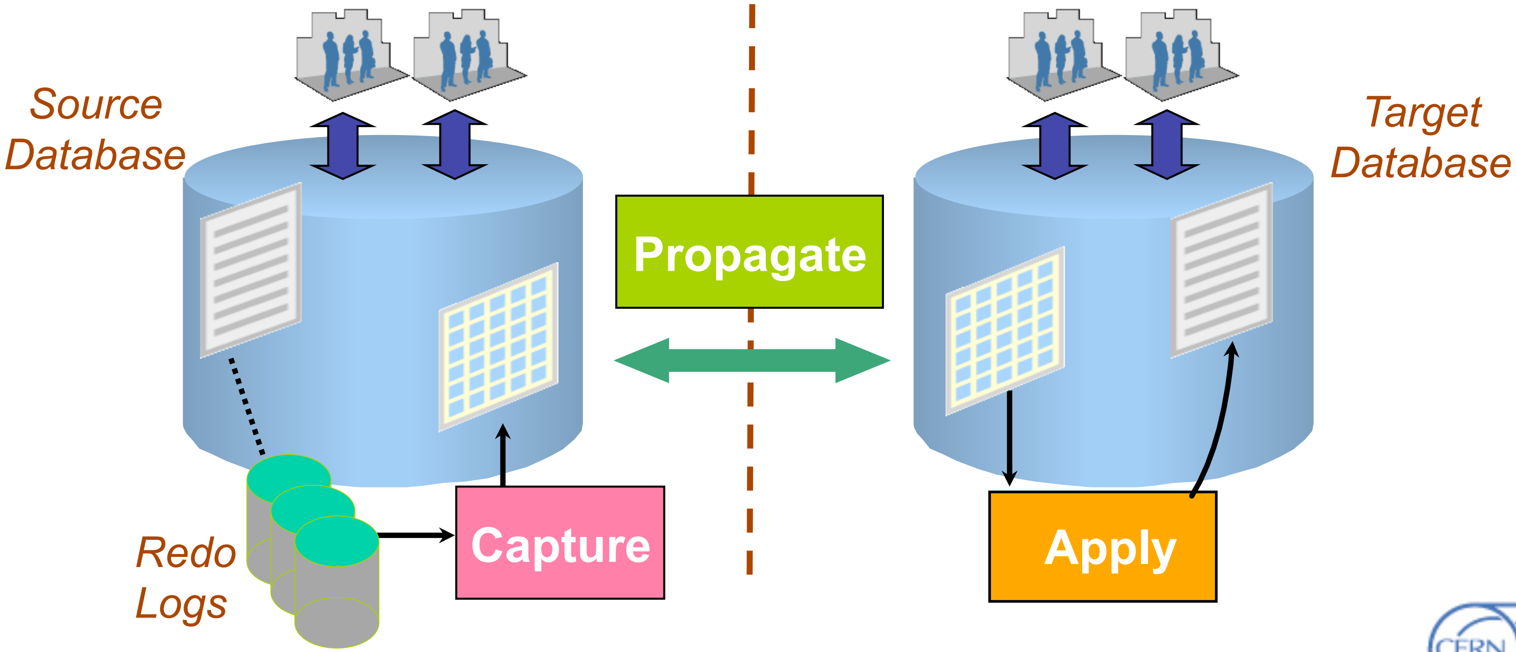
CERN IT Department



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

Oracle Streams Replication

- Technology for sharing information between databases
- Database changes captured from the redo-log and propagated asynchronously as Logical Change Records (LCRs)



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

Replication Performance

CERN IT Department

- The atomic unit is the change record: LCR
- LCRs can vary widely in size

  →Throughput is not a fixed measure

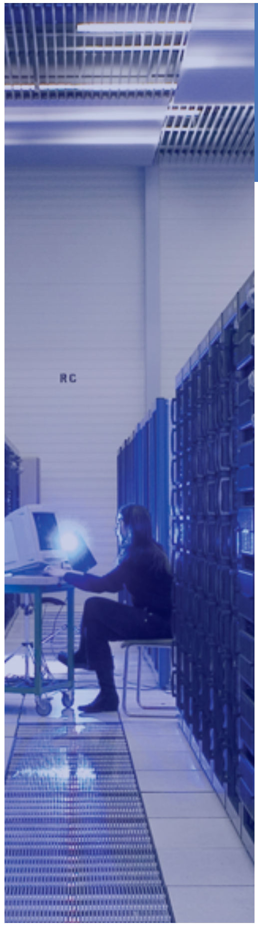
- Capture performance:
  – Read changes from the redo
    - from redo log buffer (memory - much faster)
    - from archive log files (disk)
  – Convert changes into LCRs
    - depends on the LCR size and number of columns
  – Enqueue the LCRs
    - concurrent access to the data structure can be costly

Replication Performance

CERN IT Department

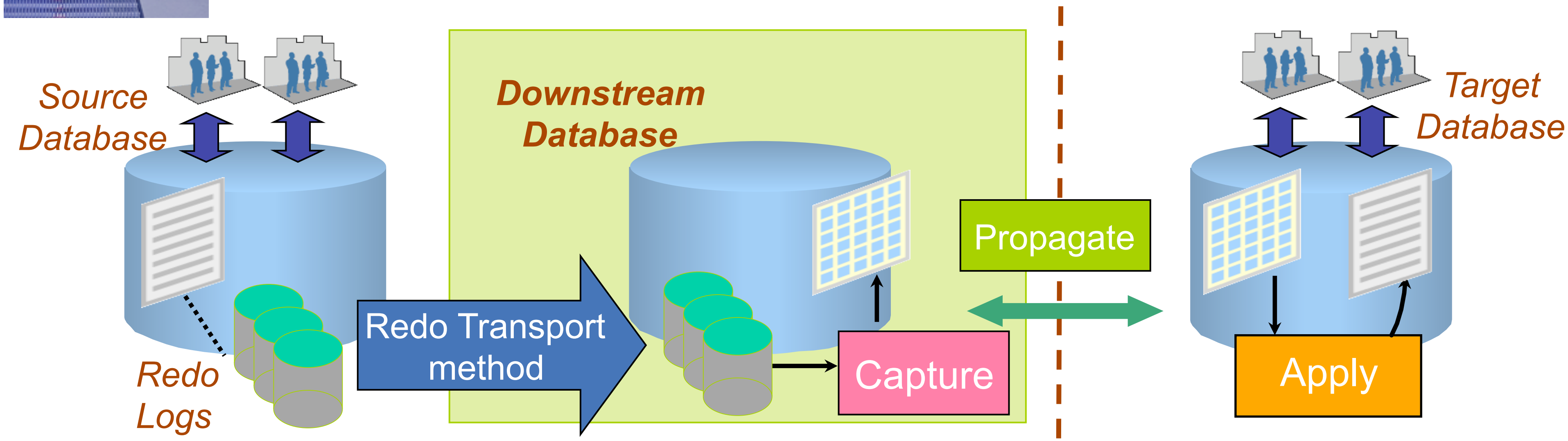
- Propagation performance:
  - Browse LCRs
  - Transmit LCRs over the network
  - Remove LCRs from the queue
    - Done in separate process to avoid any impact

- Apply performance:
  - Browse LCRs
  - Execute LCRs
    - Manipulate the database is slower than the redo generation
    - Execute LCRs serially => apply cannot keep up with the redo generation rate
  - Remove LCRs from the queue



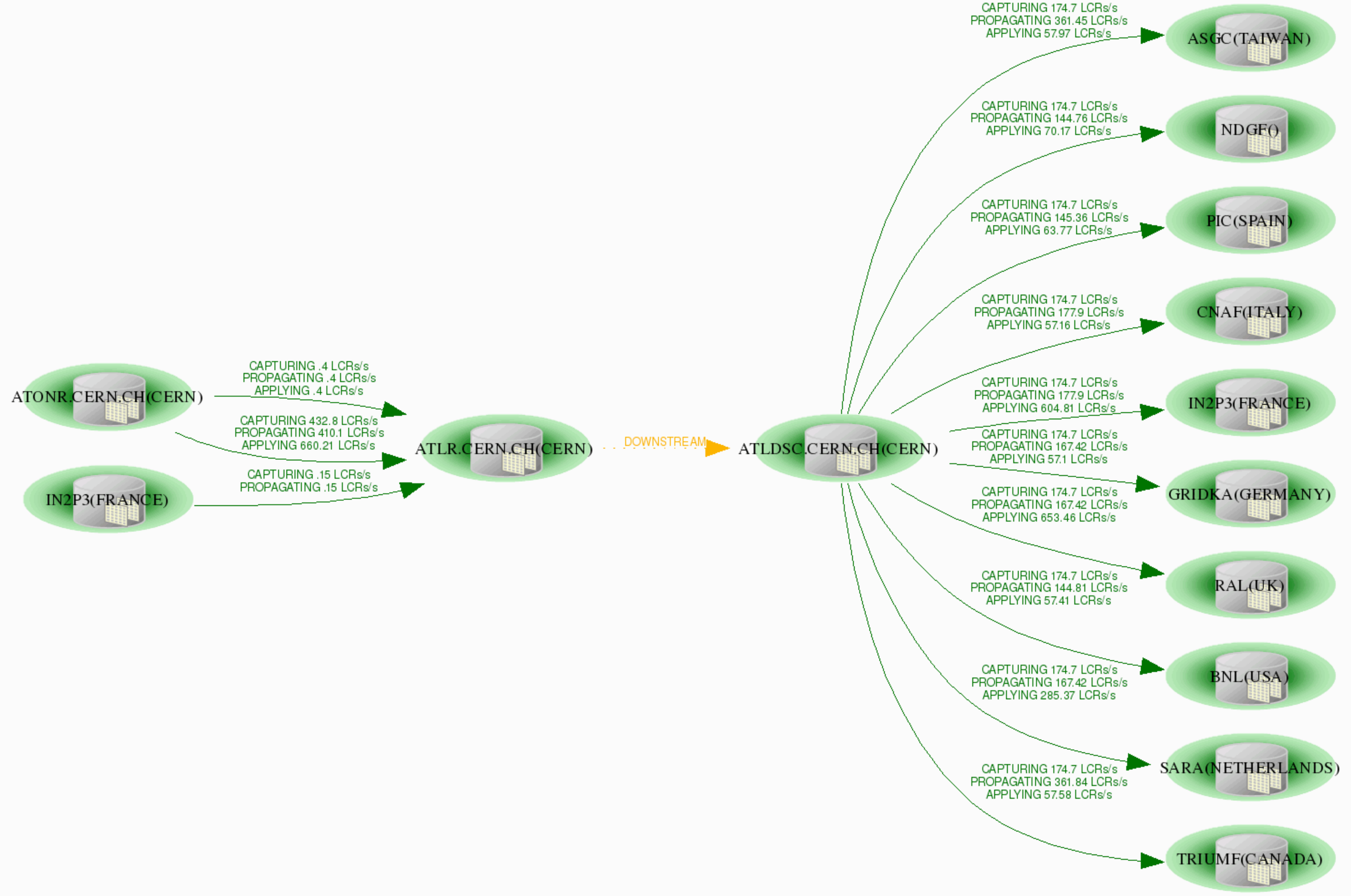
Downstream Capture

- Downstream capture to de-couple Tier 0 production databases from destination or network problems
  - source database availability is highest priority
- Optimizing redo log retention on downstream database to allow for sufficient re-synchronisation window
  - we use 5 days retention to avoid tape access
- Dump fresh copy of dictionary to redo periodically
- 10.2 Streams recommendations (metalink note 418755)

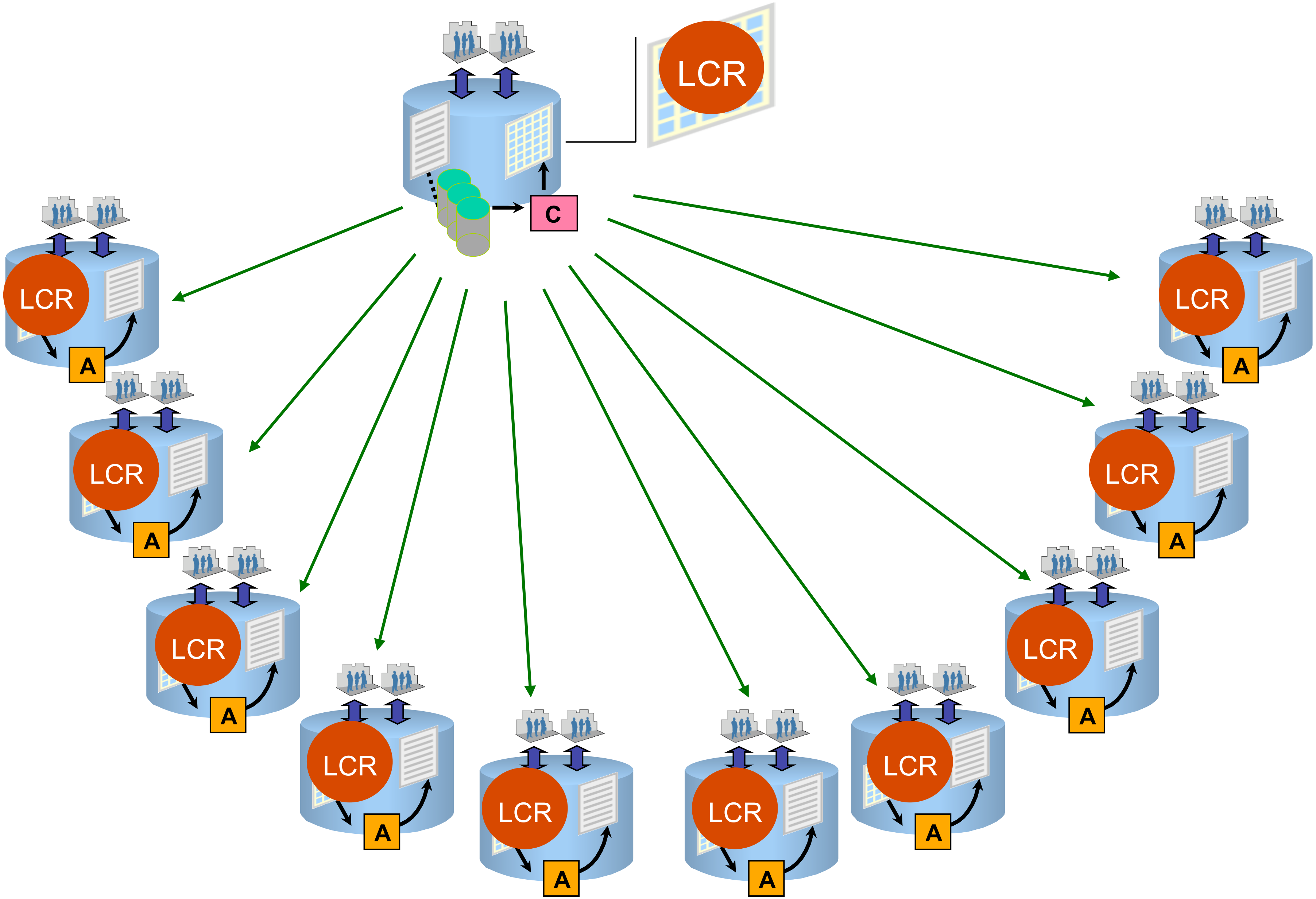


Streams Setup Example: ATLAS

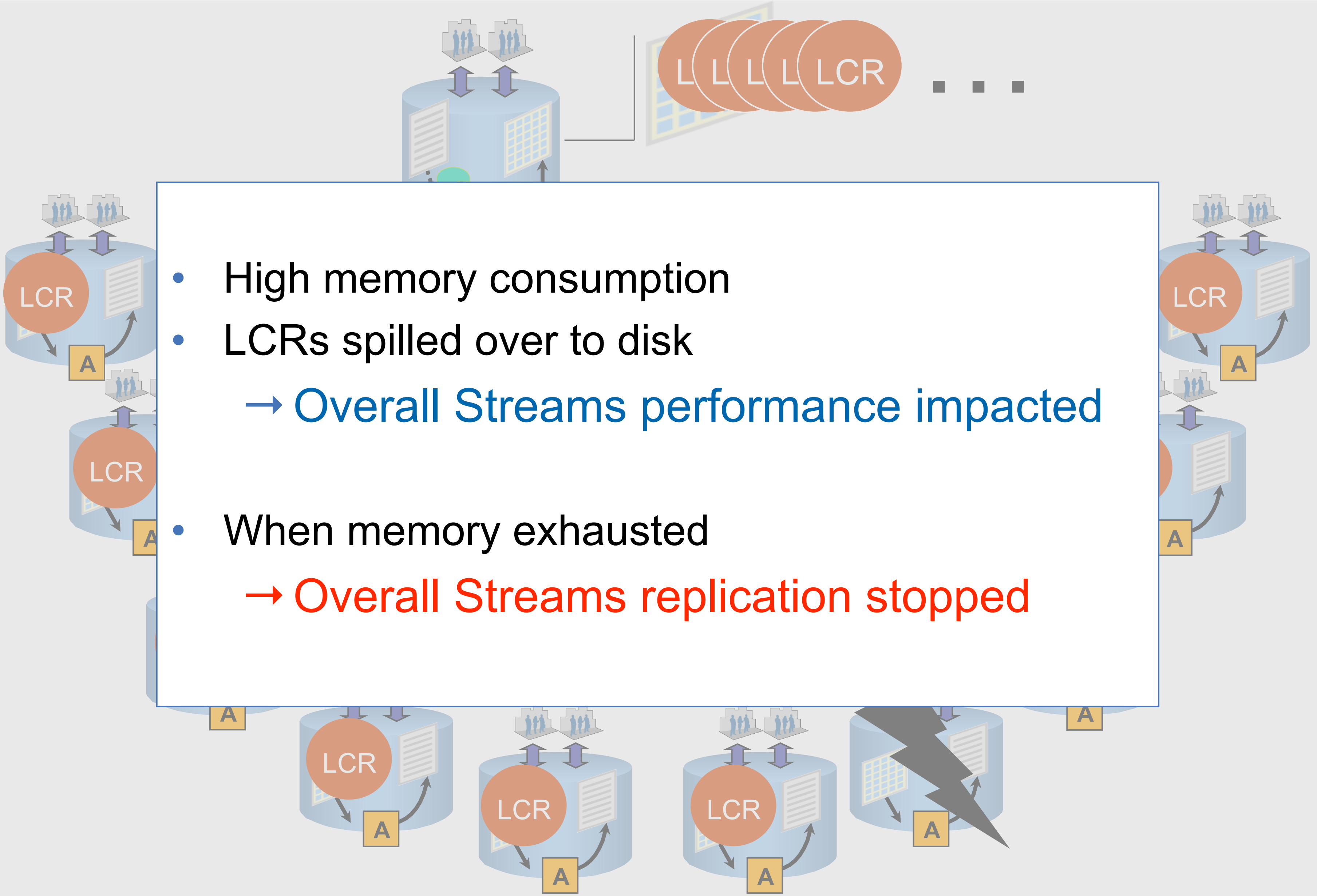
CERN IT Department



Split & Merge: Motivation



Split & Merge: Motivation



- High memory consumption
- LCRs spilled over to disk

  → Overall Streams performance impacted

- When memory exhausted

  → Overall Streams replication stopped

Split & Merge

in collaboration with Patricia McElroy
Principal Product Manager Distributed Systems/Replication - Oracle

- Objective: isolate replicas against each other
  - Split
    - (original) Streams setup for “good” sites
      - drop propagation job/s to “bad” site/s
      - →spilled LCRs are removed from the capture queue
    - (new) Streams setup for “bad” site/s
      - new capture queue
      - clone capture process and propagation job/s
    - does not require any change on the destination site/s
  - Merge
    - move back the propagation job/s to the original setup
    - clean up additional Streams processes and queue
    - does not require any change on the destination site/s

Split & Merge: Details

CERN IT Department

- Split:

```
SQL> exec split ('STRM_PROP_A’,'STRM_CAP_CL’, 'STRMQ_CL', 'STRM_PROP_CL');
        exec resynchronize_site ('STRMTEST.CERN.CH’,'STRM_CAP_CL',
            'STRMQ_CL’,1,2,'STRM_PROP_CL’,'STRMQ_A_AP','RULESET$_18','');
```

  – gather cloning information:
    - **capture**:
      – rule set name
      – start_scn = last applied message scn @target
      – first_scn = previous dictionary build < start_scn
    - **propagation**:
      – rule set name
      – target queue name and db link
- Merge:

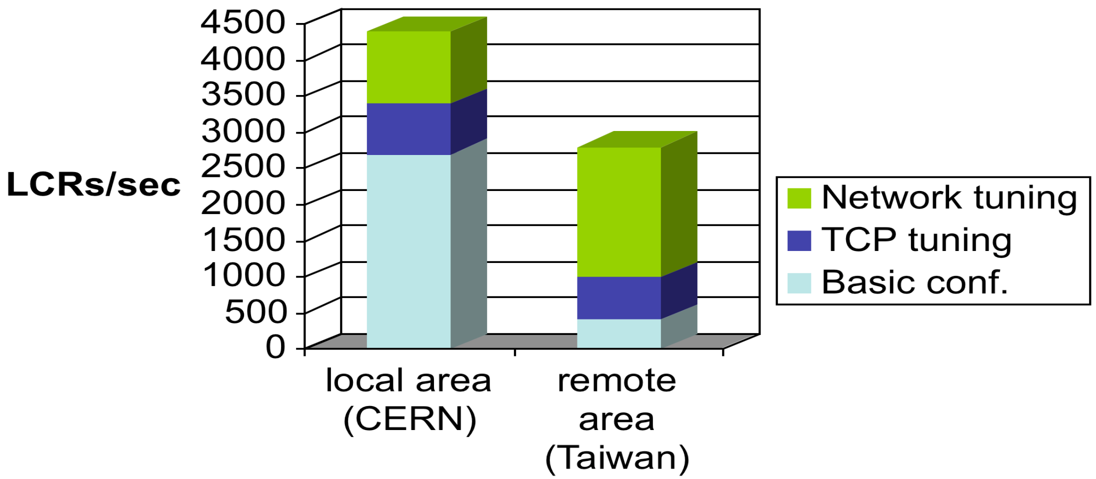
```
SQL> exec merge('STRM_CAP_SA','STRM_CAP_CL’,'STRM_PROP_A','STRM_PROP_CL');
```

  – select the minimum required checkpoint scn between the 2 capture processes
  – recover original propagation

TCP and Network Optimizations

CERN IT Department

- TCP and Network tuning
  - adjust system max TCP buffer (/etc/sysctl.conf)
  - parameters to reinforce the TCP tuning
    - DEFAULT_SDU_SIZE=32767
    - RECV_BUF_SIZE and SEND_BUF_SIZE
      - Optimal: 3 * Bandwidth Delay Product
- Reduce the Oracle Streams acknowledgements
  - alter system set events '26749 trace name context forever, level 2';



Streams Rules

CERN IT Department

- Used to control which information to share
- Rules on the capture side caused more overhead than on the propagation side
- Avoid Oracle Streams complex rules

Complex Rule

condition => '(**SUBSTR(:ddl.get_object_name(),1,7)** IN ("COMP200", "OFLP200", "CMCP200", "TMCP200", "TBDP200", "STRM200")
OR **SUBSTR (:ddl.get_base_table_name(),1,7)** IN ("COMP200", "OFLP200", "CMCP200", "TMCP200", "TBDP200", "STRM200")) '

Avoid complex rules:

- LIKE
- Functions
- NOT

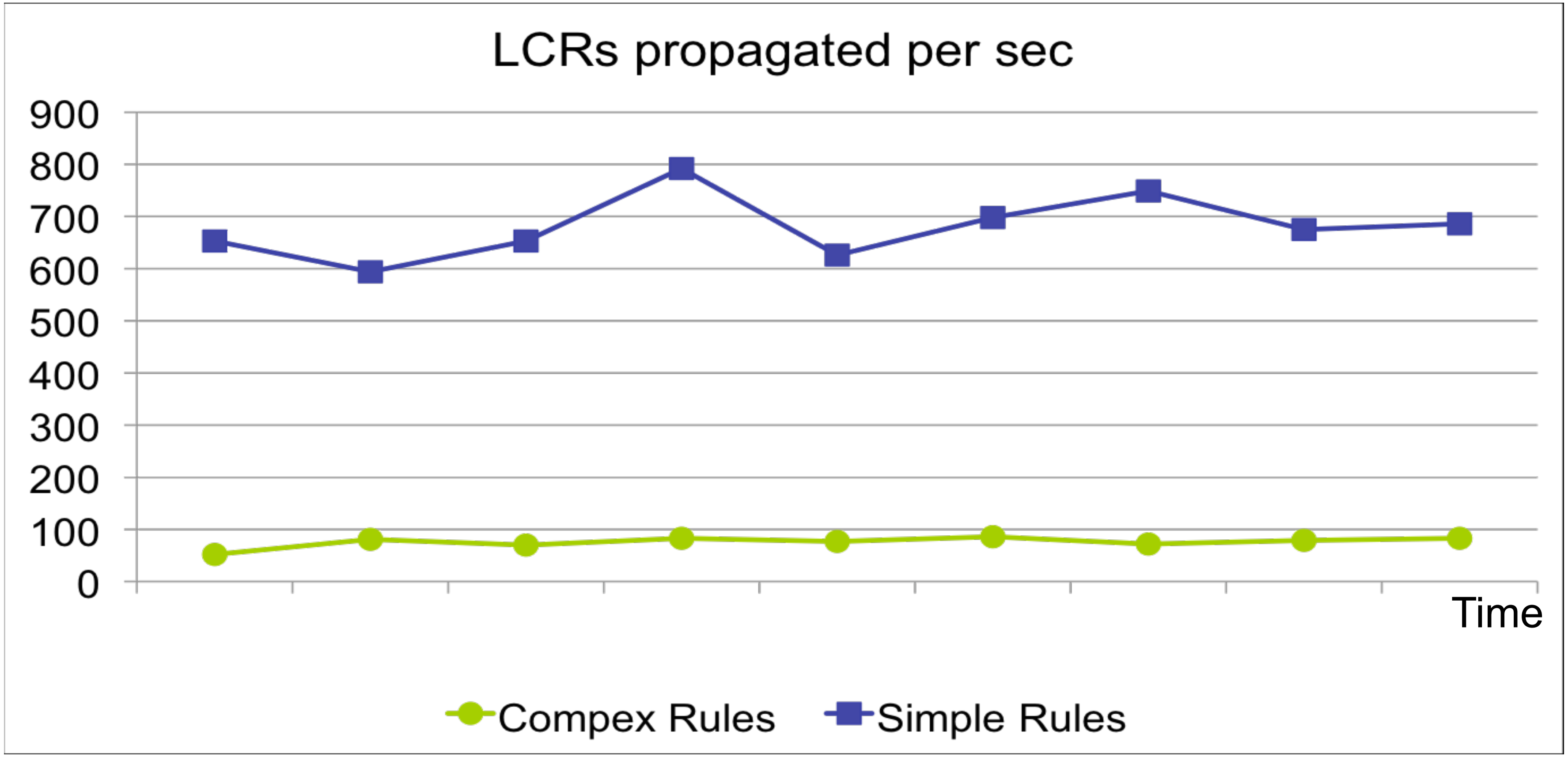
Simple Rule

condition => '(((:ddl.get_object_name() **>=** "STRM200_A" and :ddl.get_object_name() **<=** "STRM200_Z") OR
(:ddl.get_base_table_name() **>=** "STRM200_A" and :ddl.get_base_table_name() **<=** "STRM200_Z"))
OR ((:ddl.get_object_name() **>=** "OFLP200_A" and :ddl.get_object_name() **<=** "OFLP200_Z") OR
(:ddl.get_base_table_name() **>=** "OFLP200_A" and :ddl.get_base_table_name() **<=** "OFLP200_Z"))

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

Streams Rules

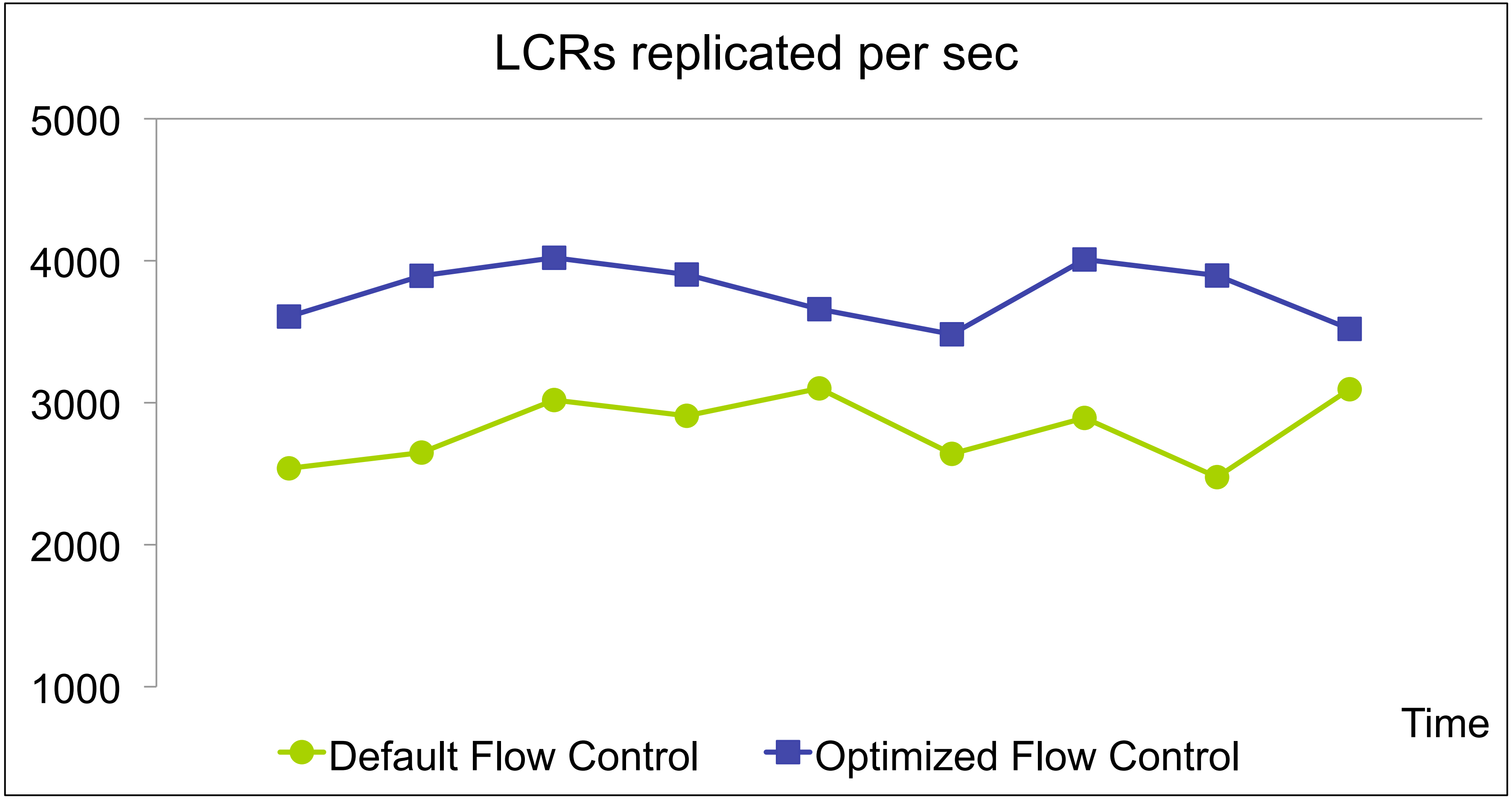
- Example: ATLAS Streams Replication
  - rules defined to filter tables by prefix



Flow Control

- By default, flow control kicks when the number of messages is larger than the threshold
  - Buffered publisher: 5000
  - Capture publisher: 15000
- Manipulate default behavior
- 10.2.0.3 + Patch 5093060 = 2 new events
  - 10867: controls threshold for any buffered message publisher
  - 10868: controls threshold for capture publisher
- 10.2.0.4 = 2 new hidden parameters
  - "_capture_publisher_flow_control_threshold"
  - "_buffered_publisher_flow_control_threshold"

- Example: ATLAS PVSS Streams Replication



Periodic Maintenance

CERN IT Department

- Dump fresh copy of Dictionary redo
  - reduces the amount of logs to be processed in case of additional process creation
- Reduce high watermark of AQ objects
  - maintain enqueue/dequeue performance
  - reduce QMON CPU usage
  - metalink note 267137.1
- Shrink Logminer checkpoint table
  - improves capture performance
  - metalink note 429599.1
- Review the list of specific Streams patches
  - metalink note 437838.1

Lessons Learned

- SQL bulk operations (at the source db)
  - may map to many elementary operations at the destination side
  - need to control source rates to avoid overloading
- Batch processing
  - minimize the performance impact using Streams tags
  - avoid changes being captured, then run same batch load on all destination
- System generated names
  - do not allow system generated names for constraints and indexes
  - modifications will fail at the replicated site
  - storage clauses also may cause some issues if the target sites are not identical

Lessons Learned

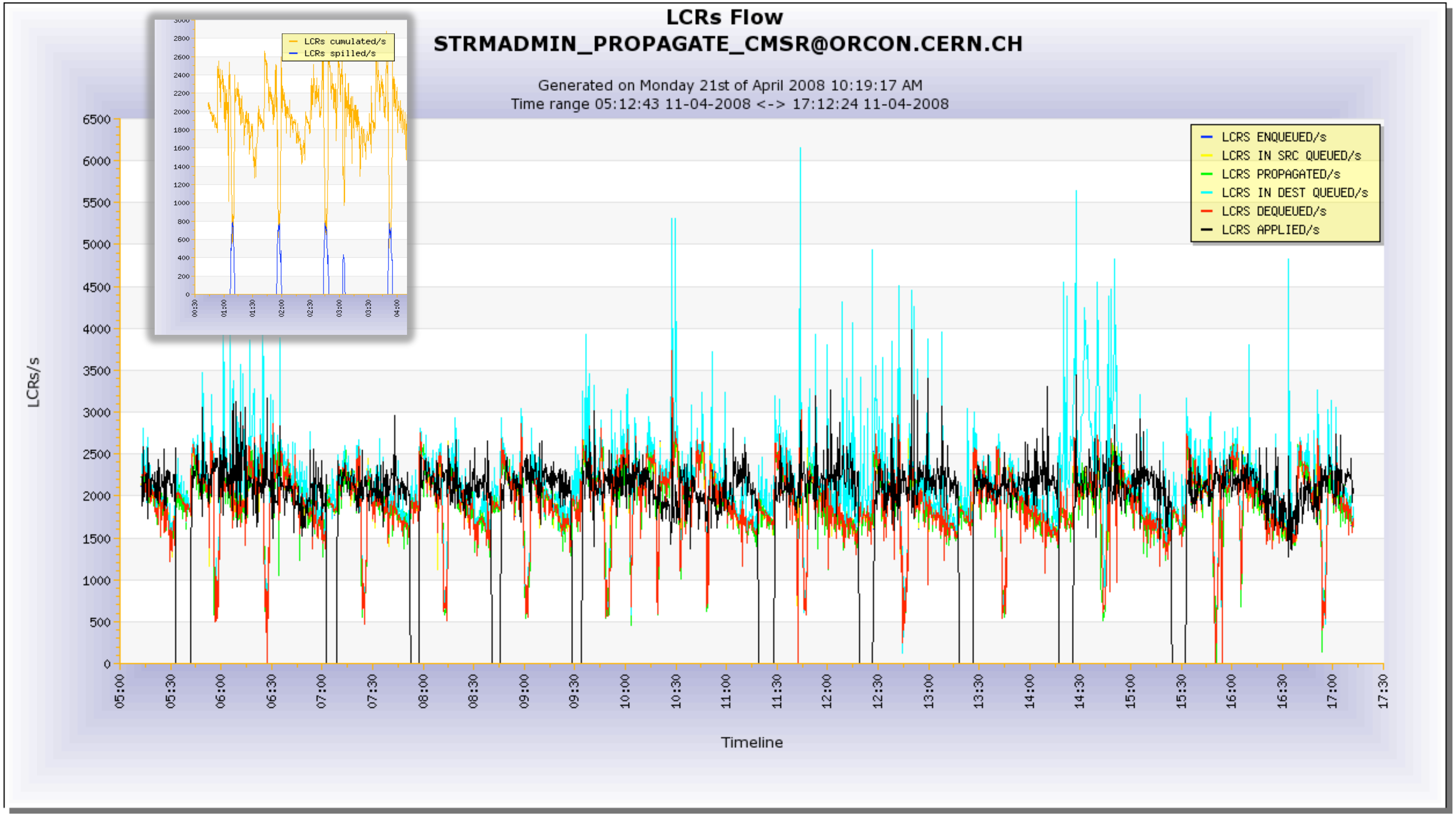
CERN IT Department

- Replication of “grant” operations
  - grants on views, PL/SQL procedures, functions and packages are NOT replicated
  - grantee must exist at all destinations
- Long transactions (non-frequent commits)
  - Total number of outstanding LCRs is too large
  - LCRs are in memory too long
    - → LCRs are spilled over to disk
    - → Apply performance is impacted
  - All LCRs in a single transaction must be applied by one apply server
    - →Parallel servers cannot be used efficiently
  - Too many unbrowsed messages enables flow control
    - →Streams processes are paused

Lessons Learned

CERN IT Department

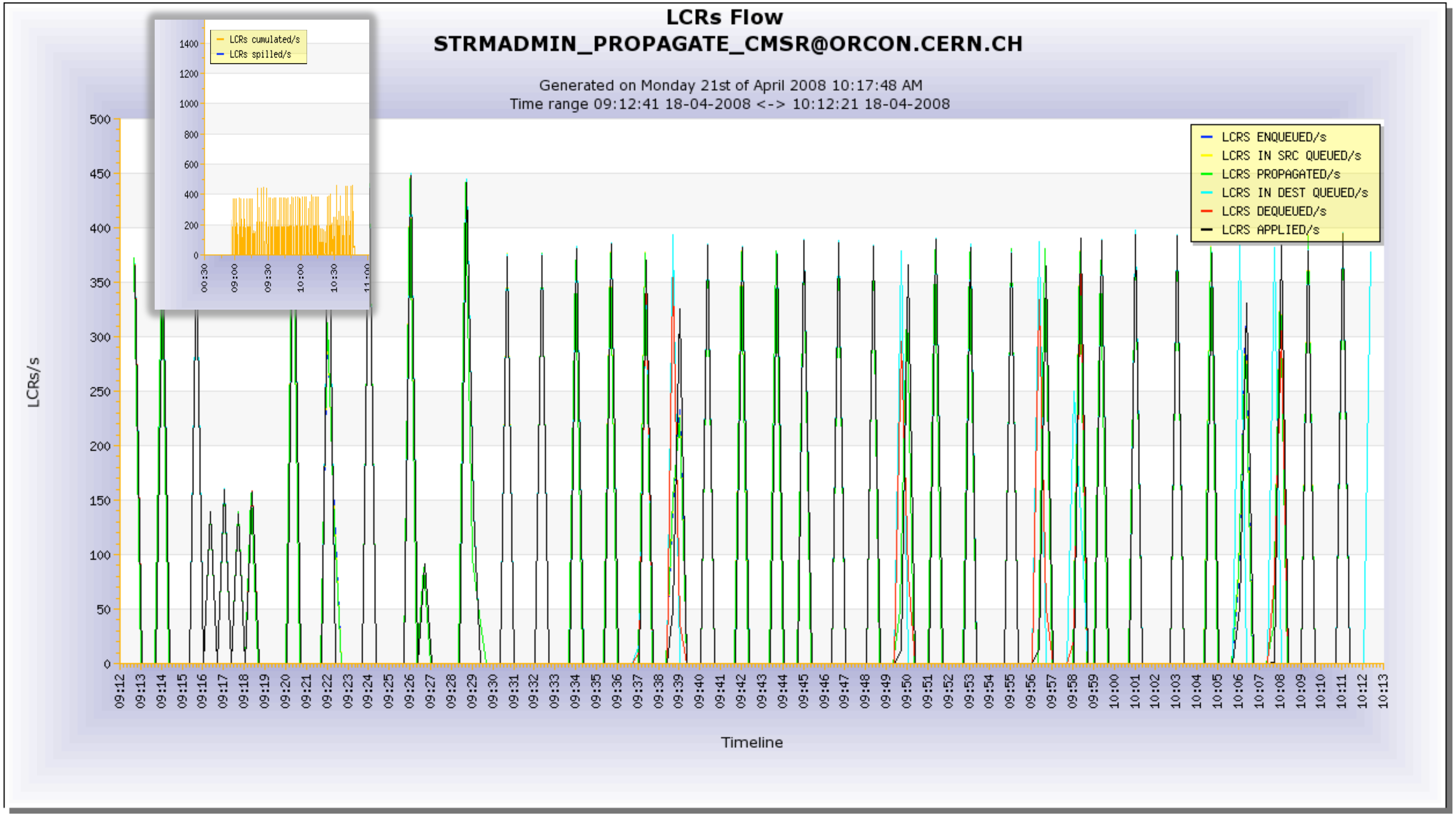
- Example: CMS replication - Online to Offline (CERN)
  - single transaction mapping 428400 LCRs



Lessons Learned

CERN IT Department

- Example: CMS replication - Online to Offline (CERN)
  - use BLOB objects: single transaction mapping 3600 LCRs



Lessons Learned

CERN IT Department

- Apply oldest_message_number is not updated
  - caused by an old transaction not correctly removed from the apply spill table
  - dba_apply_spill_txn view in order to identify the transaction
  - set the apply parameter _IGNORE_TRANSACTION with the transaction id in the apply spill over queue
  - run purge_spill_txn procedure (metalink note 556183.1)

- Apply might degrade performance when applying transactions to tables > 10M rows

Tips and Tricks

- How to **recover** Streams if downstream database crashes
  - use source database as replacement
    - all archive logs are available
  - check the oldest message number applied at each of the destinations
  - select Streams dictionary SCN < min(oldest message numbers)
  - create the Streams queue and all the propagations
  - create capture process where
    - first_scn = dictionary SCN
    - start_scn = oldest_message_number

- Configure **back** the downstream database
  - build a new Streams dictionary
  - stop capture and wait until all LCRs are applied
  - repeat steps above
  - register the archive logs with the capture process

Tips and Tricks

CERN IT Department

- Performing a **switchover** in a Streams environment
  - – database hw migration with minimal downtime
  - – completely transparent for destination databases
  - – source database:
    - before the switchover: move forward first_scn
    - after the switchover: check that the archivelog files are registered with the capture process
      - – otherwise, register them manually (from first_scn)

Streams Bugs and Patches

- Streams specific patches metalink note 437838.1
- Bug 6452375
  - ORA-26687 in Streams from “drop table”
  - when two streams setups between same source and destination databases to replicate different schemas
- Bug 6402302
  - inconsistent capture/propagation/apply of DDLs in Streams
  - for example: “drop synonym” DDL is not captured/ propagated or applied while create synonym is captured/ propagated and applied

Streams Bugs and Patches

CERN IT Department

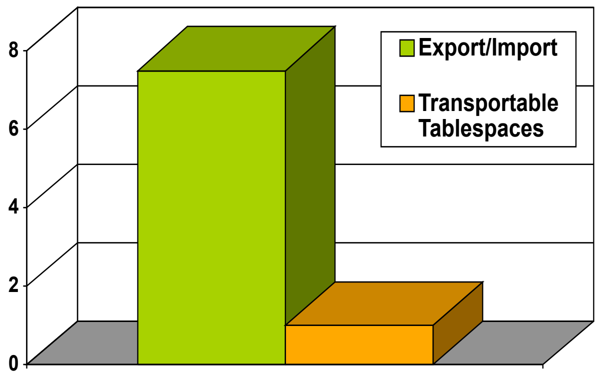
- ORA-00600: [KRVRDCBMDDLSQL1]
  - caused by rebuild index operation using parallel option
  - logminer corruption?
  - capture process could not be restarted at the current SCN
  - workaround proposed by Oracle: recreate capture using new dictionary after the index rebuild operation → data loss!!
  - complete re-instantiation of the Streams environment
- ORA-07445: exception encountered: core dump [kghufree()+485]
  - Oracle Database Change Notification cannot be used in a Streams environment

Scalable Resynchronization

- Target site out of the Streams recovery window
- Complete transfer of data (schemas and tables) using Oracle Data Pump might take too long
  - Example ATLAS Conditions data
- Transportable Tablespaces: move a set of tablespaces from one Oracle database to another
  - Export metadata of tablespace instead of data in tablespace

– But tablespaces must be in read-only while the data is copied

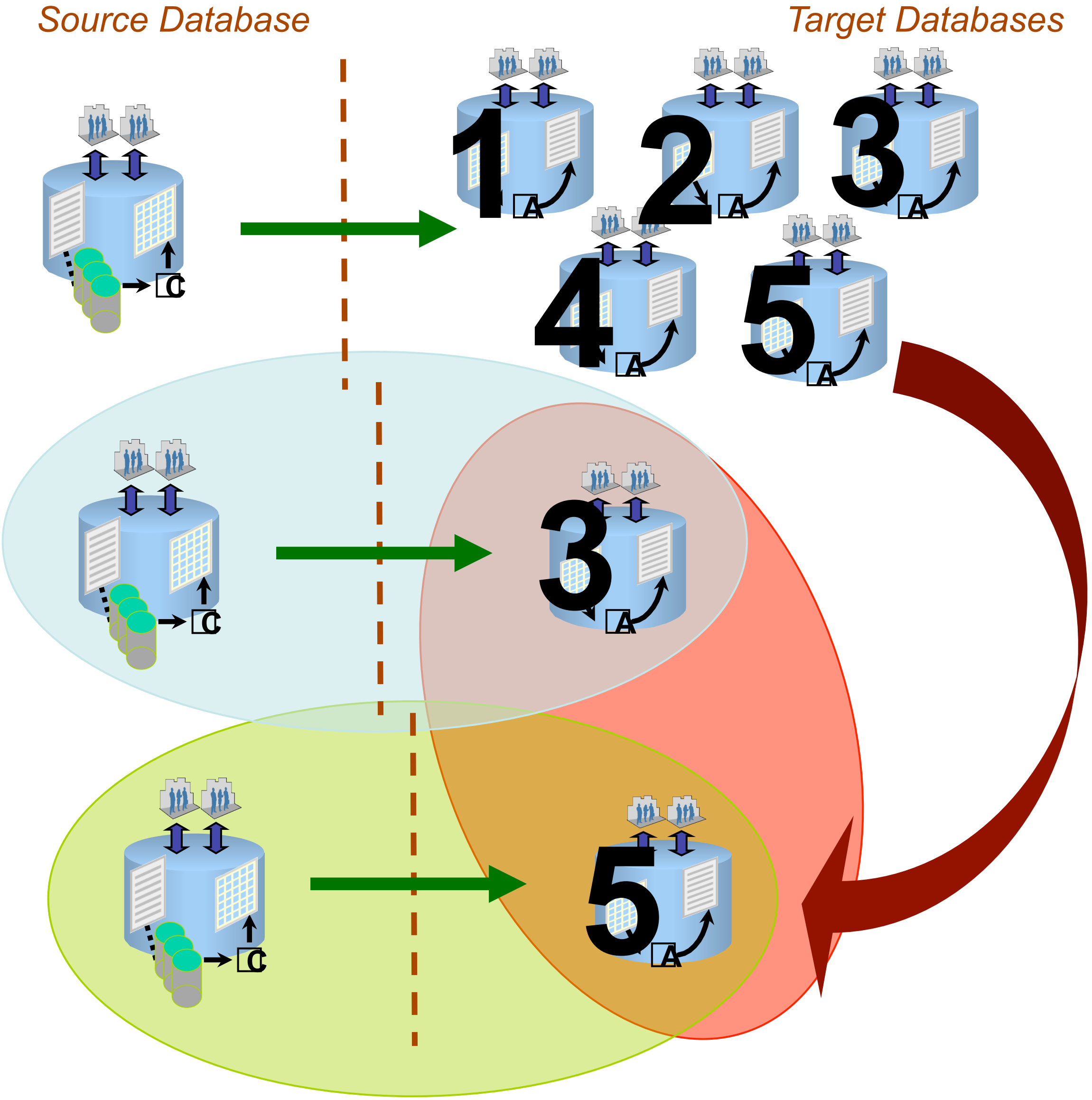
Example: ATLAS COOL test data - 67 GB



Moving data using transportable tablespaces is much faster than Data Pump export/import

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

Scalable Resynchronization



1. Create database links between databases
2. Create directories pointing to datafiles
3. Stop replication to site 5
4. Ensure tablespaces are read-only
5. Transfer the data files of each tablespace to the remote system
6. Import tablespaces metadata in the target
7. Make tablespaces read-write
8. Reconfigure Streams

Streams Monitoring

CERN IT Department

- Oracle Enterprise Manager
  - Streams monitoring enhancements on 10.2.0.5
- Oracle Streams STRMMON monitoring utility
- Streams configuration report and health check script

- Extended tool for Streams monitoring: 3D Streams Monitor tool @CERN

3D Streams Monitor

CERN IT Department

- Features:
  - Streams topology
  - Status of streams connections
  - Error notifications
  - Streams performance (latency, throughput, etc.)
  - Other resources related to the streams performance (streams pool memory, redo generation)
- Architecture:
  - "strmmon" daemon written in Python
  - End-user web application http://oms3d.cern.ch:4889/streams/main
- 3D monitoring and alerting integrated with WLCG procedures and tools

3D Streams Monitor

CERN IT Department

3D Streams Monitor

CERN IT Department

Monitor | Maps | DBs | **Streams** | Graphs | Errors | Availability | History | Reports

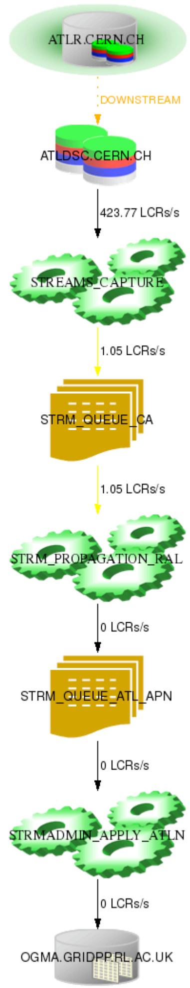
All | ALICE | **ATLAS** | CMS | LHCb | NEW | TEST | TEST11G2 | TEST_ATLAS

ACTIVE STREAMS

Stream	LCRs Cap	LCRs Enq	LCRs Prop	LCRs Deq	LCRs App	Cap Latency	Latency	Capture State	Propagation State	Apply State	State
ATLDSC.CERN.CH=>ASGC3D.GRID.SINICA.EDU.TW	951.34 /s	1.05 /s	1.05 /s	0 /s	0 /s	23 sec	--	CAPTURING CHANGES	ENABLED	IDLE	

SOURCE	ATLDSC.CERN.CH
CAPTURE	
Name	STREAMS_CAPTURE
SCN read	6083362640734
LCRs captured	10491139 (542.77 /s)
SCN captured	6083362640734
LCRs enqueued	613528 (.25 /s)
SCN enqueued	6083362621274
Capture Latency	29 sec
Info	ATLDSC.CERN.CH
State	CAPTURING CHANGES
Error Time	
Error Msg	
QUEUE	
Name	STRM_QUEUE_CA
Id	42947
Outstanding Msg	678227
Cumulative Msg	4538140 (.25 /s)
Cumulative Msg Spilled	711286 (0 /s)
PROPAGATION	
Name	STRM_PROPAGATION_RAL
LCRs Propagated	4532803 (.25 /s)
Bytes Propagated	15453 (0 /s)
State	ENABLED
Error Time	
Error Msg	

DESTINATION	OGMA.GRIDPP.RL.AC.UK
QUEUE	
Name	STRM_QUEUE_ATL_APN
Id	179048
Outstanding Msg	0
Cumulative Msg	94638683 (.05 /s)
Cumulative Msg Spilled	158012 (0 /s)
APPLY	
Name	STRMADMIN_APPLY_ATLN
LCRs Dequeued	49704552 (.05 /s)
Total LCRs Applied	48954298 (0 /s)
SCN Dequeued	6083362647224
Dequeue Latency	0 sec
Transaction Received	528045
Transaction Assigned	528045
Transaction Apllied	528045
State	IDLE
HWM SCN	6083362621274
HWM Apply Latency	29 sec
Error Time	
Error Msg	

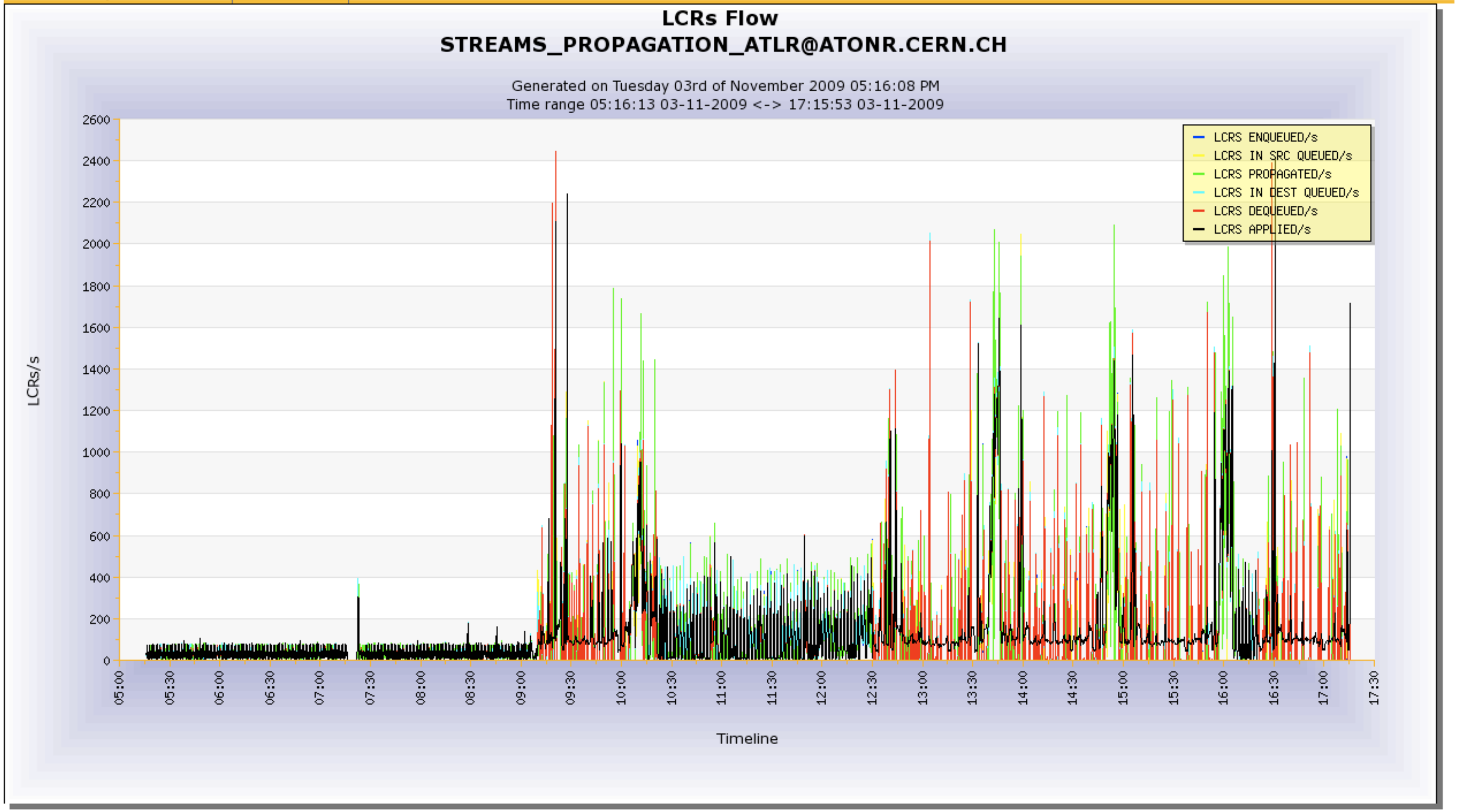


3D Streams Monitor

CERN IT Department

Monitor | Maps | DBs | Streams | **Graphs** | Errors | Availability | History | Reports

Database | Experiments | **Streams**



3D Streams Monitor

Streams report

Options

ALICE streams activities between 27.10.2009 and 03.11.2009

| availability | instances | captures | propagations | applys | queues | undefined primary keys |

Capture processes stats

2009-10-27 <-> 2009-11-02

Legend:

LCRS_CAP - LCRs captured
LCRS_ENQ - LCRs enqueued
CAP_LAT - capture latency - average time taken to capture single LCRs from log
ENQ_LAT - enqueue latency - average time taken to enque single LCRs (capture time included)

max min

DB_NAME	PROCESS_NAME	2009-10-27 LCRS_CAP	2009-10-27 LCRS_ENQ	2009-10-27 CAP_LAT	2009-10-27 ENQ_LAT	2009-10-28 LCRS_CAP	2009-10-28 LCRS_ENQ	2009-10-28 CAP_LAT	2009-10-28 ENQ_LAT	2009-10-29 LCRS_CAP	2009-10-29 LCRS_ENQ	2009-10-29 CAP_LAT	2009-10-29 ENQ_LAT
ALIONR.CERN.CH	STREAMS_PVSS_CAPTURE	8766353	291876	5.5 sec	3.6 sec	7418204	241186	5.4 sec	3.7 sec	6742596	169314	5.2 sec	3.5 sec

DB_NAME	PROCESS_NAME	2009-10-30 LCRS_CAP	2009-10-30 LCRS_ENQ	2009-10-30 CAP_LAT	2009-10-30 ENQ_LAT	2009-10-31 LCRS_CAP	2009-10-31 LCRS_ENQ	2009-10-31 CAP_LAT	2009-10-31 ENQ_LAT
ALIONR.CERN.CH	STREAMS_PVSS_CAPTURE	6795907	186108	5.3 sec	3.7 sec	6317804	193119	5.3 sec	3.7 sec

DB_NAME	PROCESS_NAME	2009-11-01 LCRS_CAP	2009-11-01 LCRS_ENQ	2009-11-01 CAP_LAT	2009-11-01 ENQ_LAT	2009-11-02 LCRS_CAP	2009-11-02 LCRS_ENQ	2009-11-02 CAP_LAT	2009-11-02 ENQ_LAT
ALIONR.CERN.CH	STREAMS_PVSS_CAPTURE	6583790	201615	5.2 sec	3.7 sec	7591331	204735	5.4 sec	3.6 sec

Total

DB_NAME	PROCESS_NAME	TOTAL LCRS_CAP	LCRS_ENQ	CAP_LAT	ENQ_LAT
ALIONR.CERN.CH	STREAMS_PVSS_CAPTURE	50215988	29759103	5.3 sec	2d 11h 2m 45s

Propagation processes stats

2009-10-27 <-> 2009-11-02

Legend:

LCRS_PROP - number of propagated LCRs
BYTES_PROP - number of propagated bytes

max min

DB_NAME	PROCESS_NAME	2009-10-27 LCR_PROP	2009-10-27 BYTES_PROP	2009-10-28 LCR_PROP	2009-10-28 BYTES_PROP	2009-10-29 LCR_PROP	2009-10-29 BYTES_PROP	2009-10-30 LCR_PROP	2009-10-30 BYTES_PROP	2009-10-31 LCR_PROP	2009-10-31 BYTES_PROP	2009-11-01 LCR_PROP	2009-11-01 BYTES_PROP	2009-11-02 LCR_PROP	2009-11-02 BYTES_PROP
ALIONR.CERN.CH	STREAMS_PROPAGATE_PDBR	5827071	130.3 kB	4814418	80.2 kB	3379169	79.7 kB	3714992	130.4 kB	3854643	105.3 kB	4024718	0 B	4086873	92.0 kB

Total

DB_NAME	PROCESS_NAME	TOTAL LCR_PROP	BYTES_PROP
ALIONR.CERN.CH	STREAMS_PROPAGATE_PDBR	29701885	617.9 kB

Queue stats

2009-10-27 <-> 2009-11-02

Legend:

LCRS_IN_Q - number of enqueued messages (LCRs)
LCRS_SPILLED - number of spilled LCRs to disk
SPILL_RATE - spilling percentage
LCRS_AVG - average number of messages in queue

max min

DB_NAME	QUEUE_NAME	2009-10-27 LCRS_IN_Q	2009-10-27 SPILLED	2009-10-27 SPILL_RATE	2009-10-27 LCRS_AVG	2009-10-28 LCRS_IN_Q	2009-10-28 SPILLED	2009-10-28 SPILL_RATE	2009-10-28 LCRS_AVG	2009-10-29 LCRS_IN_Q	2009-10-29 SPILLED	2009-10-29 SPILL_RATE	2009-10-29 LCRS_AVG
ALIONR.CERN.CH	STREAMS_PVSS_QUEUE_CA	5834670	0	0.0 %	660	4821164	0	0.0 %	560	3384564	0	0.0 %	397
PDBR.CERN.CH	STREAMS_PVSS_QUEUE_AP	5866178	0	0.0 %	135	4840116	0	0.0 %	144	3396840	0	0.0 %	96
PDBR.CERN.CH	KUPC$S_1_20091015143048	0	0	0.0 %	0	0	0	0.0 %	0	0	0	0.0 %	0

DB_NAME	QUEUE_NAME	2009-10-30 LCRS_IN_Q	2009-10-30 SPILLED	2009-10-30 SPILL_RATE	2009-10-30 LCRS_AVG	2009-10-31 LCRS_IN_Q	2009-10-31 SPILLED	2009-10-31 SPILL_RATE	2009-10-31 LCRS_AVG
ALIONR.CERN.CH	STREAMS_PVSS_QUEUE_CA	3720078	0	0.0 %	415	3860081	0	0.0 %	405
PDBR.CERN.CH	STREAMS_PVSS_QUEUE_AP	3737837	0	0.0 %	106	3876122	0	0.0 %	81
PDBR.CERN.CH	KUPC$S_1_20091015143048	0	0	0.0 %	0	0	0	0.0 %	0

DB_NAME	QUEUE_NAME	2009-11-01 LCRS_IN_Q	2009-11-01 SPILLED	2009-11-01 SPILL_RATE	2009-11-01 LCRS_AVG	2009-11-02 LCRS_IN_Q	2009-11-02 SPILLED	2009-11-02 SPILL_RATE	2009-11-02 LCRS_AVG
ALIONR.CERN.CH	STREAMS_PVSS_QUEUE_CA	4030247	0	0.0 %	450	4092501	0	0.0 %	471
PDBR.CERN.CH	STREAMS_PVSS_QUEUE_AP	4046299	0	0.0 %	100	4110115	0	0.0 %	111
PDBR.CERN.CH	KUPC$S_1_20091015143048	0	0	0.0 %	0	0	0	0.0 %	0

Total

DB_NAME	QUEUE_NAME	TOTAL LCRS_IN_Q	SPILLED	SPILL_RATE	LCRS_AVG
ALIONR.CERN.CH	STREAMS_PVSS_QUEUE_CA	29743307	0	0.0 %	480.2
PDBR.CERN.CH	STREAMS_PVSS_QUEUE_AP	29873509	0	0.0 %	110.8
PDBR.CERN.CH	KUPC$S_1_20091015143048	0	0	0.0 %	0.0

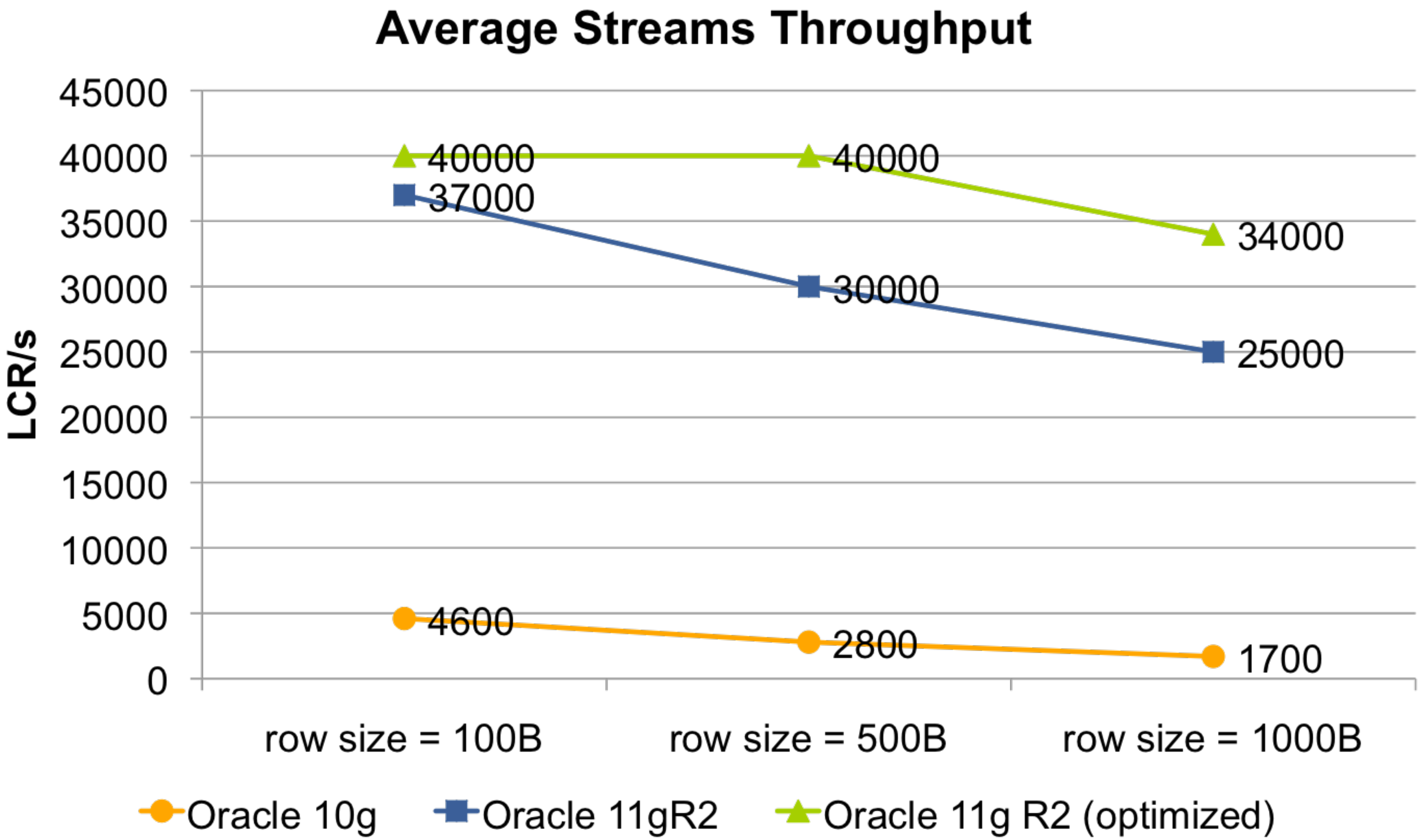
Generated in 39 min 57 sec

New Streams 11g Features

- Performance improvements:
  - reader process mines from the in-memory redo log buffers
    - minimizes disk I/O
    - reduces capture latency
  - direct communication between capture and apply processes: Combined Capture and Apply
    - improves LCR transmission throughput
    - reduces end-to-end replication latency
  - internal mechanism to execute change records and extensive caching
    - reduces CPU consumption
    - minimizes latch contention and other wait events

New Streams 11g Features

CERN IT Department



New 11g Streams Features

CERN IT Department

- Automatic Split and Merge
  - split a stream in cases where a replica is unavailable
  - merge into a single stream when replica catches up
  - procedures and sql script generation
  - automatic replication management based on thresholds
- Compare and Converge
  - compare objects across databases for inconsistency
  - resynchronize objects if required
  - table or column level synchronization
  - additional scripting for schema comparison

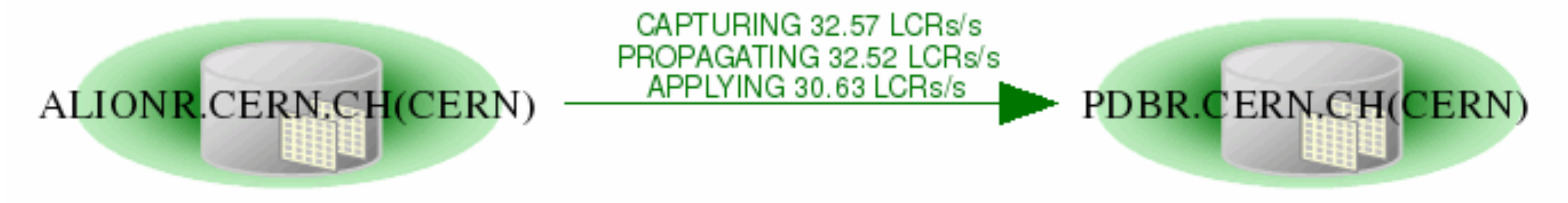
New 11g Streams Features

CERN IT Department

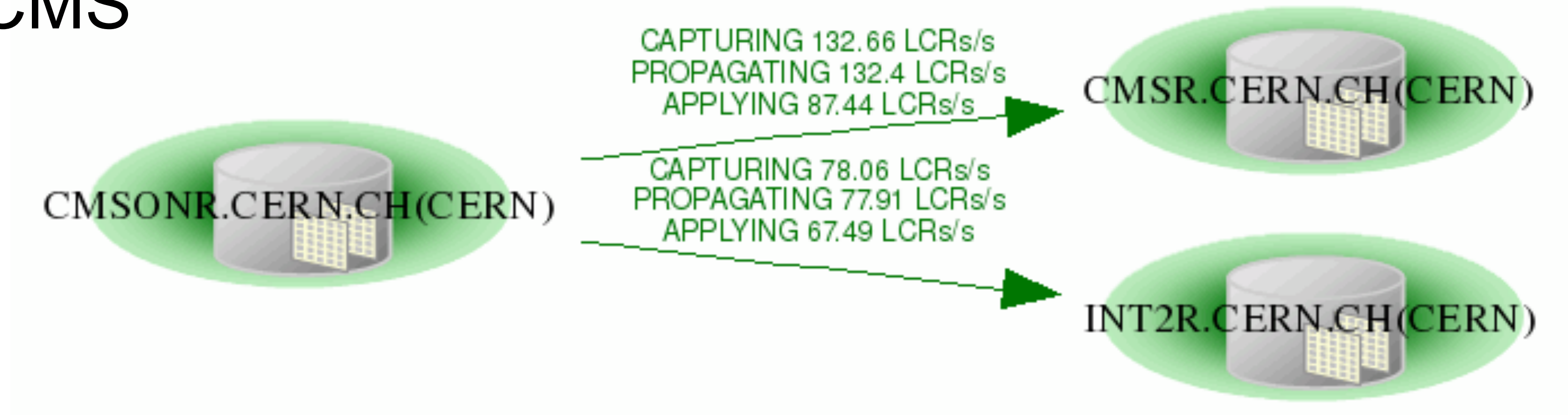
And more…

- Synchronous capture
- LCRs track through a Stream
- Topology and Performance Advisor
- New error messages for error handling

- ALICE

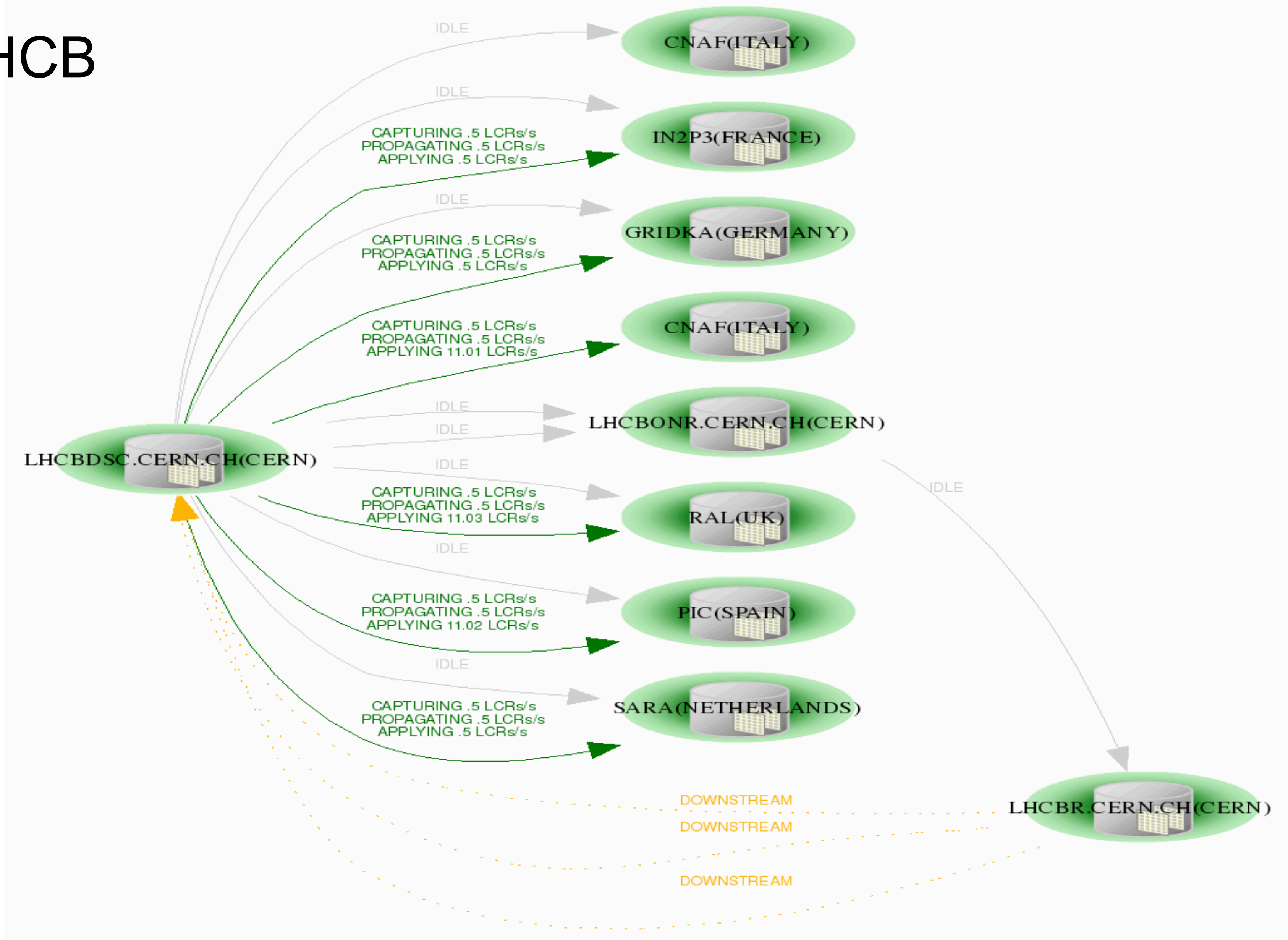

ALIONR.CERN.CH(CERN)
CAPTURING 32.57 LCRs/s
PROPAGATING 32.52 LCRs/s
APPLYING 30.63 LCRs/s
PDBR.CERN.CH(CERN)

- CMS


CMSONR.CERN.CH(CERN)
CAPTURING 132.66 LCRs/s
PROPAGATING 132.4 LCRs/s
APPLYING 87.44 LCRs/s
CMSR.CERN.CH(CERN)
CAPTURING 78.06 LCRs/s
PROPAGATING 77.91 LCRs/s
APPLYING 67.49 LCRs/s
INT2R.CERN.CH(CERN)

More Streams Setup Examples

- LHCB



Summary

CERN IT Department

- The LCG Database Deployment Project (LCG 3D) has set up a world-wide distributed database infrastructure for LHC
  - some 33 RAC clusters = 636 CPU cores at CERN + several tens of nodes at 10 partner sites are in production now
- Large scale tests have validated that the experiment are implemented by the RAC & streams based set-up
  - backup & recovery tests have been performed to validate the operational procedures at all sites
- Monitoring of database & streams performance has been implemented building on grid control and strmmon tools
  - key to maintain and optimize any larger system

Q&A

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