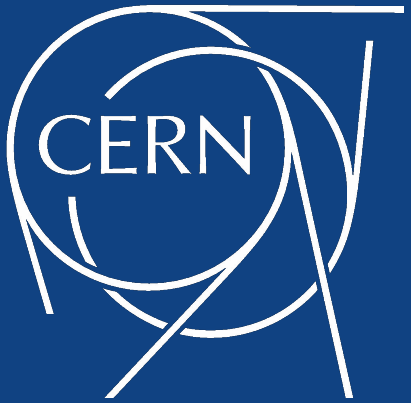
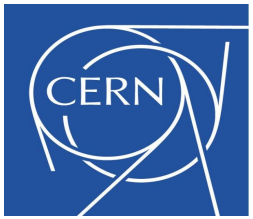

CERN

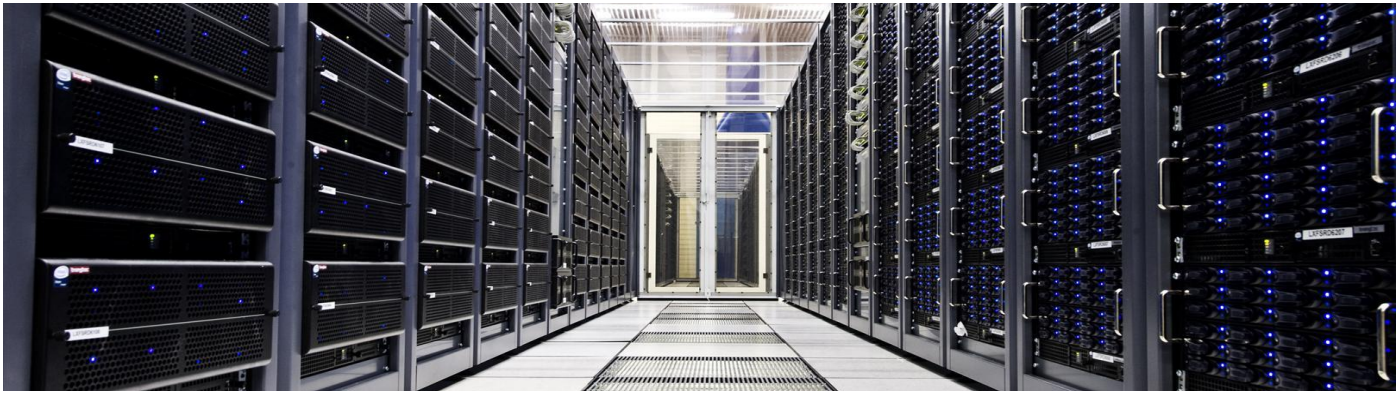
Migrating from Streams to GoldenGate12c



Tech15.UKOUG

Birmingham 7th of December,2015

Zbigniew Baranowski, CERN IT-DB



About Zbigniew

- Joined CERN in 2009
  - Developer
  - **Database Administrator & Service Manager**

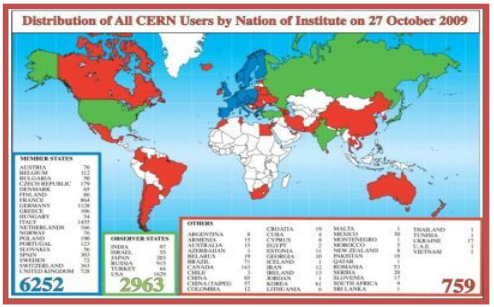
- Responsible for
  - Engineering & LHC control database infrastructure
  - Database replication services in Worldwide LHC Computing Grid
  - Central Hadoop service @CERN

Outline

- Database replication@CERN - overview
- Why GoldenGate?
- Preparation for the migration
- Migration
- Summary

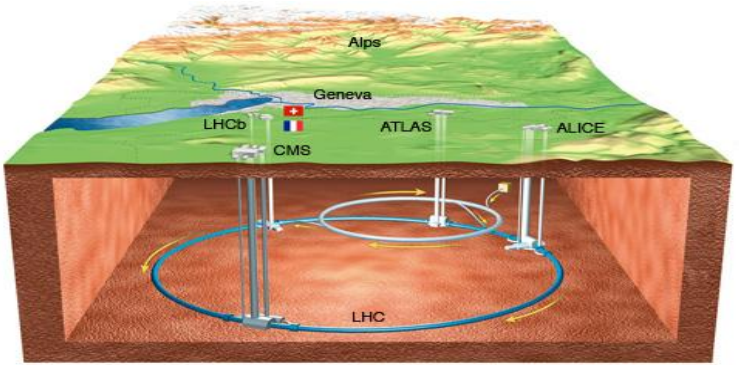
About CERN

- CERN - European Laboratory for Particle Physics
- Founded in 1954 by 12 countries for fundamental physics research
- Today 21 member states + world-wide collaborations
  - 10’000 users from 110 countries



LHC is the world's largest particle accelerator

- LHC = Large Hadron Collider
  - 27km ring of superconducting magnets; 4 big experiments
  - Produces ~30 Petabytes annually
  - Just restarted after an upgrade – x2 collision energy (13 TeV) is expected



Database SERVICES — At the heart of CERN, LHC and Experiment Operations

CERN IT Department

http://cern.ch/it-dep/

Experiment Offline Databases
9 Production DBs, 7 Integration DBs
5 Tests DBs, 8 (Active) Data Guards

Streams

Experiment Online Databases
4 Production DBs, 6 (Active) Data Guards

Data

Streams

CASTOR
CERN Advanced STORage manager

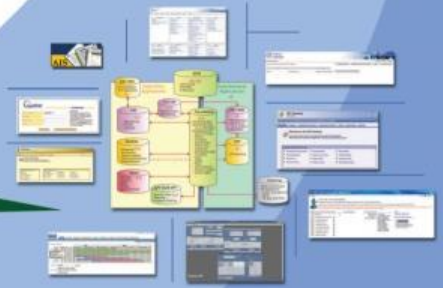
21 Production DBs
4 Development DBs
1 Data Guard

RAW Data

Middleware
73 Application Servers

Administrative/IT/ Engineering Databases
17 Production DBs
11 Development DBs
4 Ref/Test DBs

Accelerators ACC
12 Production DBs



LHC Experiments

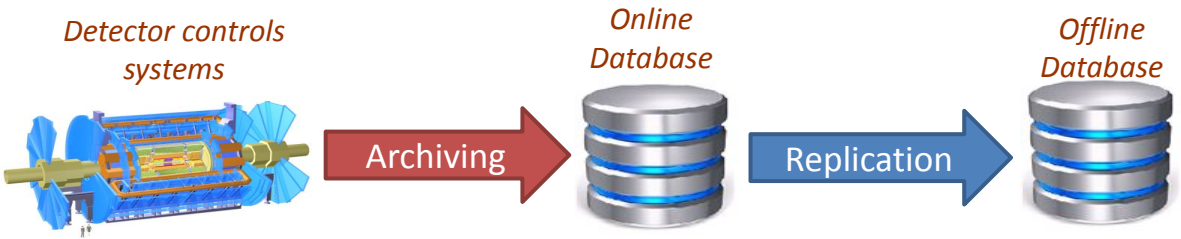
PARTNERS:

CERN openlab
ORACLE

CERN

Data (DML & DDL) replication for online DBs

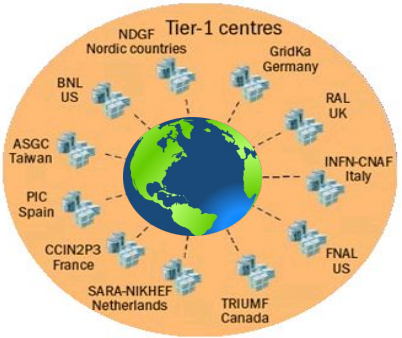
- Key component of online-offline DB model for experiments database services
  - Detector conditions data
  - Detector controls and aquisition system archives (WinCC/PVSS)



Data (DML & DDL) replication for WLCG

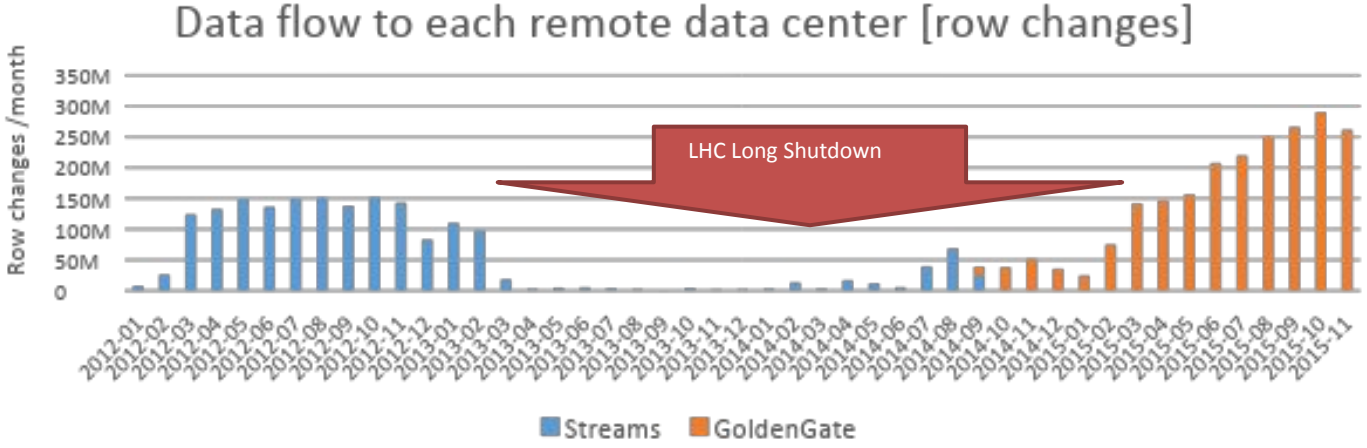
- World wide data distribution for collision reconstruction and analysis
- Consolidation of various data at CERN
  - Calibration data
  - Metadata interfaces

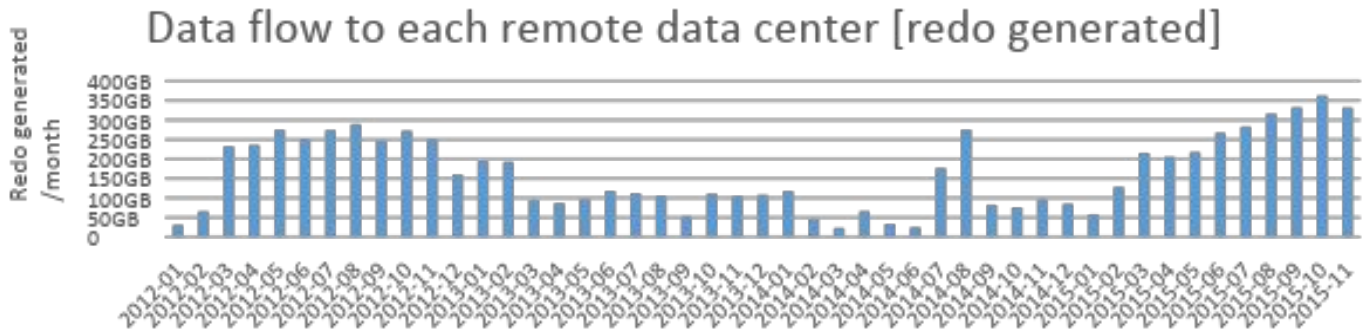
Offline Databases



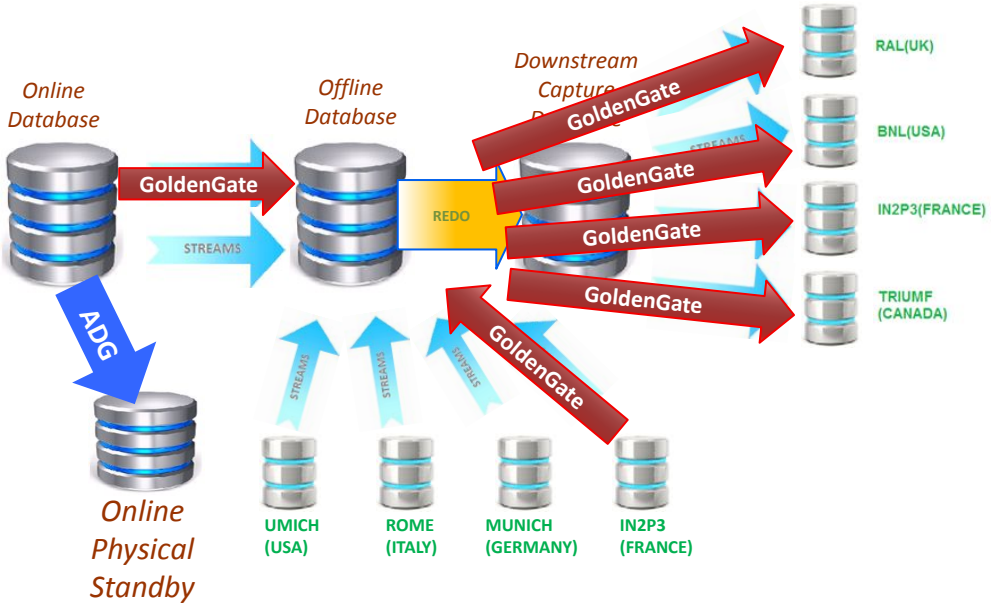
ATLAS conditions are replication to a subset of Tier-1 sites

Data rates


Data flow to each remote data center [row changes]
Row changes /month
350M
300M
250M
200M
150M
100M
50M
0
LHC Long Shutdown
Streams
GoldenGate


Data flow to each remote data center [redo generated]
Redo generated /month
400GB
350GB
300GB
250GB
200GB
150GB
100GB
50GB
0

Replication Setup for ATLAS experiment in 2014

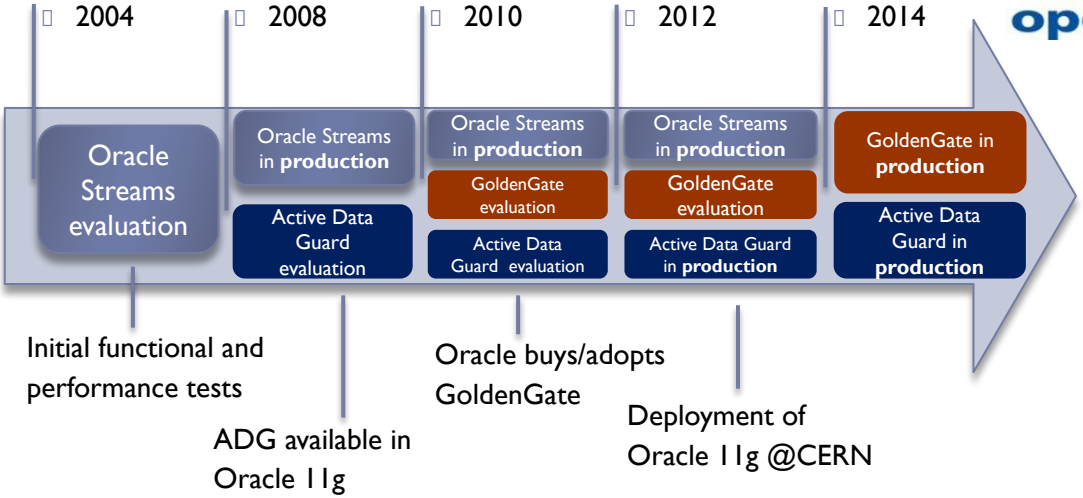


Why GoldenGate?

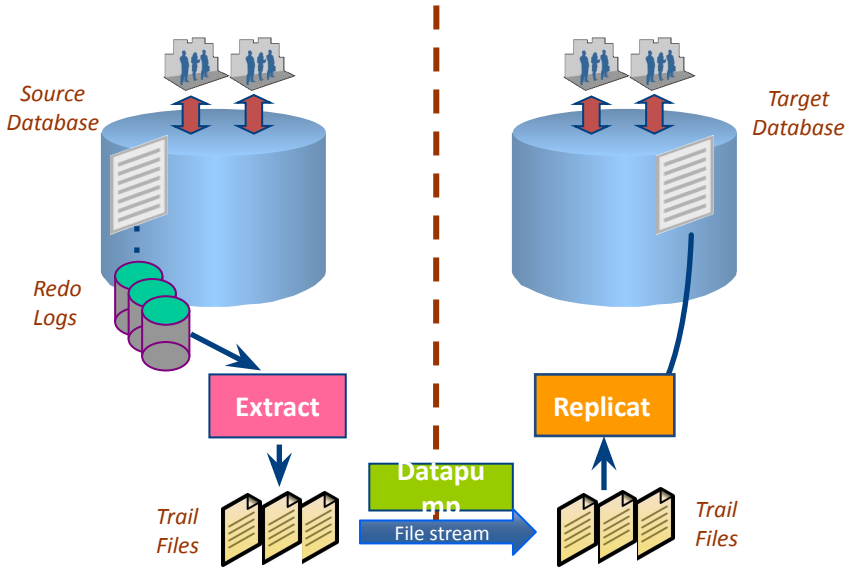
Why logical (SQL based) replication?

- Allows partial database replication
  - Important for hub-and-spoke over WAN
- RDBMS versions decoupled between primary and replica
  - Easier maintenance planning within remote data centres
- Replica in read-write mode
  - Flexibility in building complex replication topologies (cascading…)
  - Improve data access performance from replicas (additional indexes)

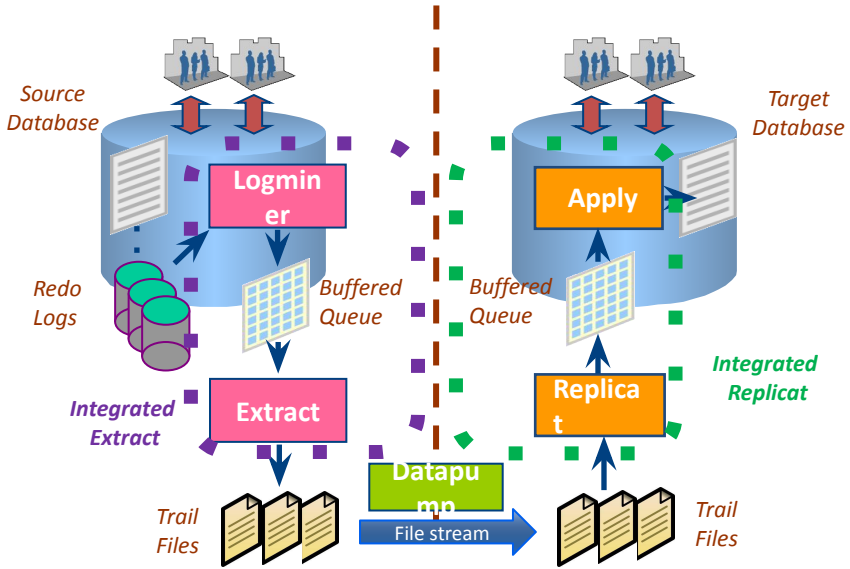
Replication evolution@CERN



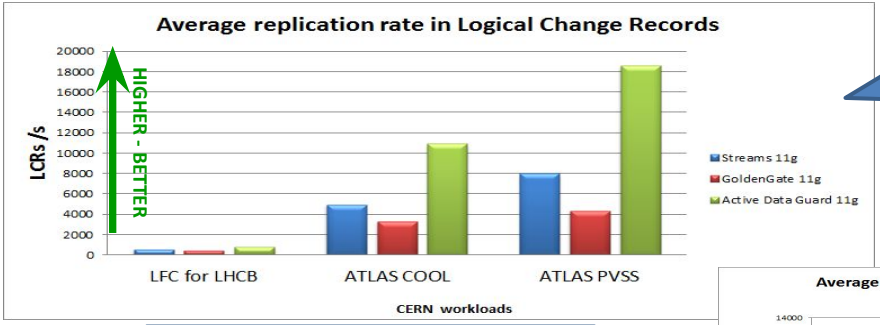
GG architecture (2010)



GG Integrated architecture (2013)

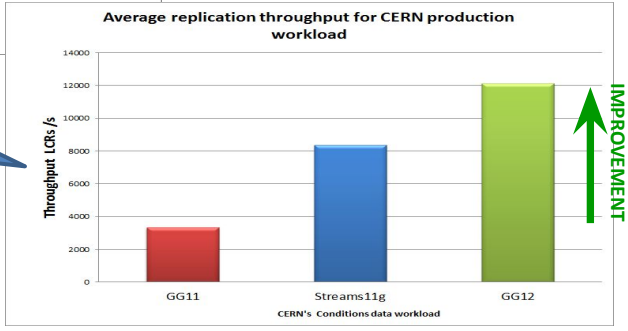


Evaluation - performance



In 2011:
-> Active Data Guard performs the best
-> Golden Gate the worst

In 2013:
-> new version of GoldenGate (12c) beats Streams



Streams vs GoldenGate

- Streams in 11g are mature and reliable
  - but will not be enhanced!
    - Oracle recommended log-based replication technology is now GoldenGate (2010)
  - Streams does not support some data operations

- GoldenGate12c became improved version of Streams!
  - A lot of (good) features taken from Streams
  - Improved scalability - performance better than Streams
  - Availability of in-database monitoring and reporting
  - More functionalities and data types supported

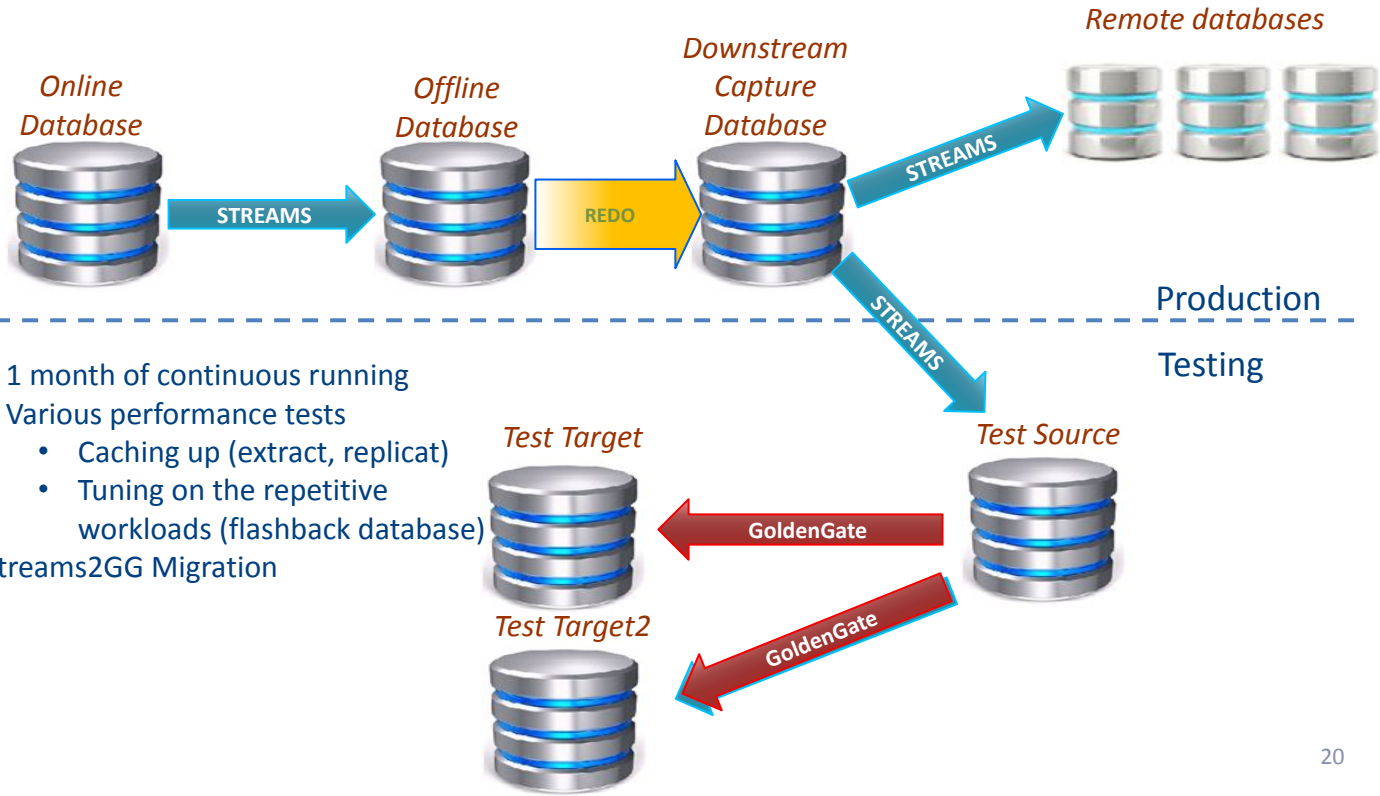
- Experience gained by running Streams will bear fruits when running GoldenGate

Testing and validation

- Performance tests on synthetic data
  - Easy to establish
  - Are good for benchmarking
  - Not necessary reflects real production workloads

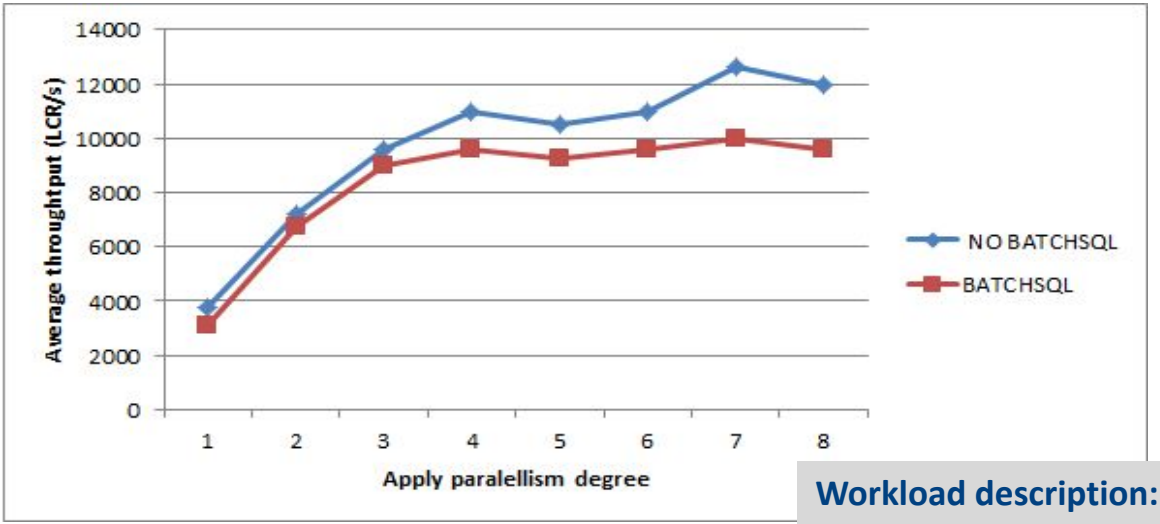
- Validation tests
  - How to be sure that all DML and DDLs will be properly replicated by GG without issues?
  - DML supported tables can be checked in **DBA_GOLDENGATE_SUPPORT_MODE** view
  - Should we only relay on the view?

Testing with production workloads



- 1 month of continuous running
- Various performance tests
  - Caching up (extract, replicat)
  - Tuning on the repetitive workloads (flashback database)
- Streams2GG Migration

Performance measured



Workload description:

- 5 days of ATLAS conditions data
- 675GB of redo volume
- 260k of transaction
- 18.9 M of row changes (LCRs)

Ok, Lets migrate…

Target software configuration

- CRS
  - 12.1.0.1 and 12.1.0.2
- RDBMS
  - 11.2.0.4 and 12.1.0.1
- GoldenGate
  - 12.1.2.1.0
  - Extract and Replicat in integrated mode
- Platform
  - RHEL6

Migration procedure overview

- Steps
  1. Preliminary steps
     - Configure databases
     - Install GG
     - Set up GG process
  2. Online switch between Streams and GoldenGate
     - stop streams
     - start GG
  3. Validate that GG process are up and replicating
  4. (Later) Drop Streams components
     - Capture, Propagation, Apply and AQ queues

The procedure is already well documented (**Doc ID 1383303.1**)

Preparation

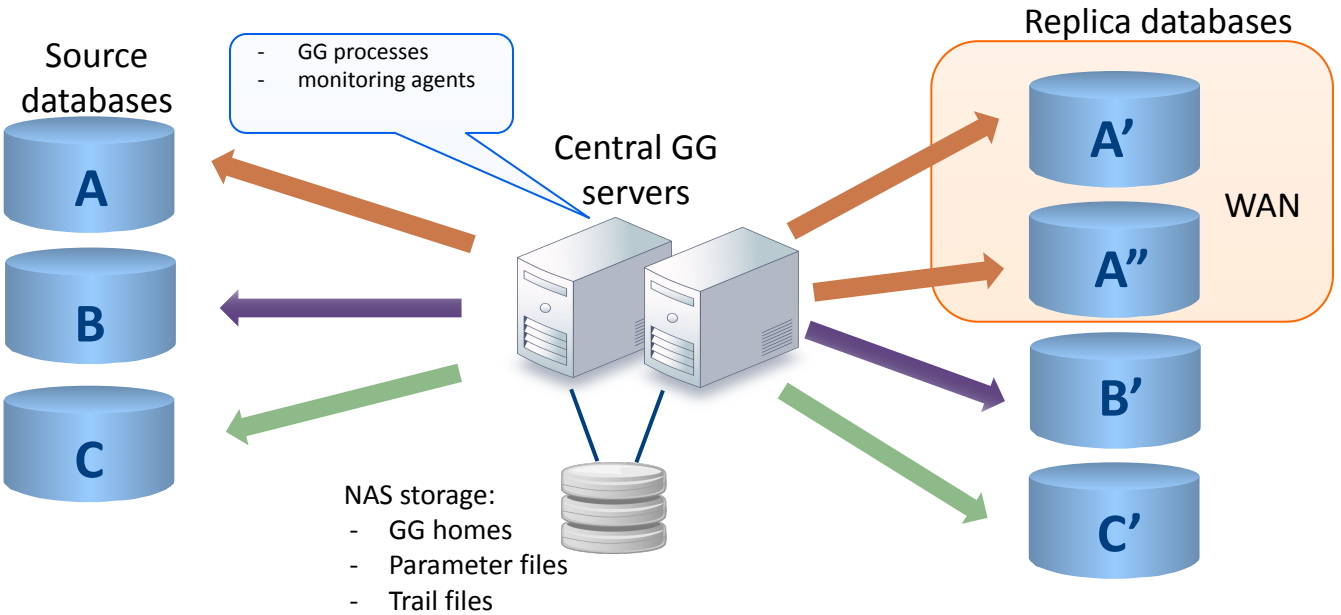
Preliminary steps

- Database preparation
- Install Golden Gate
  - Allocate the storage for GG homes and trails
  - Get GG software and run OUI
  - Open ports for gg *manager* on firewall
- Integration with CRS
- Porting replication configuration
  - Prepare parameter files
  - Create processes

Database preparation for GG

- Most of configuration done when setting up Streams
  - Streams pool (~2GB), supplemental logging, force logging
- set *COMPATIBLE* >= 11.2.0.4 – required by *integrated replicat*
  - Plan it before – database restart required
- set *ENABLE_GOLDENGATE_REPLICATION=TRUE*
- creation of GG administrator schema
  - Grant the right privileges
    - dbms_goldengate_auth.grant_admin_privilege('ggadm')
    - DDL support requires DBA role granted

Central GoldenGate installation @CERN



GoldenGate service design @CERN

- Central GoldenGate installation
  - all GoldenGate configurations run a dedicated two-node cluster
    - *extracts* and *replicats* in the integrated mode => operate on databases remotely
  - binaries & configurations stored on a shared storage
  - monitoring agents installed on the GG cluster
  - cluster in master-slave configuration
    - With automatic failover

Central GoldenGate advantages

- Consolidated deployment and management
  - Installation of GG software on each db server is not needed
  - Everything in one place => easy maintenance and management
  - No need to maintain GG *datapump* process
  - Single trail files in a single place =>less storage needed

- Improved security
  - GG *manager* not exposed
  - No need of opening extra sets of ports on replica RAC machines

- Simplified deployment of GG monitoring

Porting Streams configuration to GG

- Streams2OGG scripts (**Doc ID 1912338.1**)
  - generates GG parameter files, and creation scripts based on Streams config (capture, propagation, apply)
    - Replication rules (DML & DDL)
    - DML and error handlers
    - Supplemental logging
    - Conflict detection and resolution (CDR)
    - ...
  - Best practices are applied in the parameter files
  - Does NOT generate migration scripts
  - Currently v3.0 available (we used 2.5)

Credential store

- Keep your passwords out of parameter files

- Adding credential store

```
GGSCI> ADD CREDENTIALSTORE
```

- Add gg administrator user

```
ALTER CREDENTIALSTORE add user ggadm@dba
Password:

Credential store in ./dircrd/ altered.
```

- Use USERIDALIAS in parameter files and ggsci

```
dblogin useridalias ggadm@dba
Successfully logged into database.
```

Porting Streams configuration to GG

- Streams2OGG scripts usage
  1) download scripts and unzip
  2) grant needed privileges to STRMADMIN
     - CREATE/DROP ANY DIRECTORY
     - SELECT ON DBA_PROCEDURES
  3) load the package to STRMADMIN schema
     - SQLPLUS> *@stream2ogg.sql*
     - *gg_admin* and *staging directory* to be specified
  4) (recommended) use naming and mapping mappings via CSV file

```
sqlplus> streams2ogg.customize
```

  5) Edit the *ogg_name_map.csv* file (key, value)
  6) Run config generator

```
sqlplus> set serveroutput on
sqlplus> streams2ogg.main
```

Porting Streams configuration to GG

- mgr.prm – manager configuration

```
PORT 7809
-- DYNAMICPORTLIST 15000-15040
-- Manager checks for Extract and Replicat lag.
LAGREPORTMINUTES 5
-- Manager reports Extract and Replicat lag.
LAGINFOMINUTES 5
-- threshold that is considered critical-> write warning entry
LAGCRITICALMINUTES 15
```

  - Recommended :

```
AUTORESTART ER *,RETRIES 3, WAITMINUTES 4
AUTOSTART ER *
PURGEOLDEXTRACTS *, USECHECKPOINTS, MINKEEPDAYS 15
```

Porting Streams configuration to GG

- *extract* parameter file

```
extract CAPTCOND

--GENERAL
INCLUDE ./dirprm/dbname_ggadmin.prm
USERIDALIAS ggadm@dbname
exttrail trail_path/oc
REPORTCOUNT EVERY 5 MINUTES
WARNLONGTRANS 1H, CHECKINTERVAL 30m
LOGALLSUPCOLS
UPDATERECORDFORMAT COMPACT
TRANLOGOPTIONS EXCLUDETAG 00
TRANLOGOPTIONS INTEGRATEDPARAMS (_CHECKPOINT_FORCE N, CHECKPOINT FREQUENCY 1000)



--DDL CONFIG
DDL EXCLUDE OBJNAME "*"."DBMS_TABCOMP_TEMP_CMP" EXCLUDE OBJNAME "*"."DBMS_TABCOMP_TEMP_UNCMP"
INCLUDE OBJNAME "SCHEMA1"."*" INCLUDE OBJNAME "SCHEMA2"."*" --...VERY LONG
DDLOPTIONS REPORT

--DML CONFIG
TABLEEXCLUDE "*"."DBMS_TABCOMP_TEMP_CMP" ;
TABLEEXCLUDE "*"."DBMS_TABCOMP_TEMP_UNCMP" ;

TABLE "SCHEMA1"."*";
TABLE "SCHEMA2"."*";
--and so on
```

Enables writing supplemental logging data to trail files

Important for UPDATE DMLs - before and after images stored in a single trail record

We want to have a possibility to exclude tagged sessions

We do not want to replicate Segment Advisor activity

Porting Streams configuration to GG

- *replicat* parameter files

```
replicat CONDREP

#GENRAL
GETENV (NLS_LANG)
INCLUDE ./dirprm/db_name_ggadmin.prm
USERIDALIAS ggadm@dbname
ASSUMETARGETDEFS
discardfile ./dirrpt/CONDREP.dsc, PURGE, MEGA
REPORTCOUNT EVERY 5 MINUTES, RATE
DBOPTIONS DEFERREFCONST
DBOPTIONS SETTAG 01
DBOPTIONS SETTAG null  #DEFAULT IS 00
DBOPTIONS SUPPRESSTRIGGERS
DBOPTIONS INTEGRATEDPARAMS (COMMIT_SERIALIZATION FULL, DISABLE_ON_ERROR Y, PARALLELISM 1)

#DDL
DDL INCLUDE OBJNAME "SCHEMA1".* INCLUDE OBJNAME "SCHEMA2".*
DDLOPTIONS NOTAG   #DEFAULT IS 00
DDLERROR 38307 IGNORE  --ORA-38307: Object not in recycle bin

#DML
MAP "SCHEMA1".* ,TARGET "SCHEMA1".*,
COMPARECOLS (
ON UPDATE ALL,
ON DELETE ALL);
MAP "SCHEMA2".* ,TARGET "SCHEMA2".*,
COMPARECOLS (
ON UPDATE ALL,
ON DELETE ALL);
--and so on
```

Watch out for tagging in a cascading configuration. We do not tag changes applied by GG

Taken from current Stremas Apply parameters -> customize it later

DDLs are tagged by replicat independently from DMLs

Conflict detection for UPDATE and DELETE operations

Porting Streams configuration to GG

- *datapump* parameter file

```
extract DPCOND

#GENERAL
INCLUDE ./dirprm/db_ggadmin.prm
rmthost <host name>, mgrport 7809
rmttrail trail_path/zz
discardfile ./dirrpt/DPCOND.dsc, PURGE, MEGABYTES 500
PASSTHRU
TABLE *.*;
```

Porting Streams configuration to GG

- Scripts generated
  - create_subdirectories.sh – creates dirs for trail
  - ggconfig(2).oby – creation of GG processes

Simplified content

```
dblogin userid GGADMIN, password <password>

#EXTRACT CREATION
register extract CAPTCOND database
add extract CAPTCOND, integrated tranlog, begin now, nodbcheckpoint
add exttrail trail_path/oc, extract CAPTCOND, megabytes 50

#REPLICAT CREATION
register replicat CONDREP database
add replicat CONDREP integrated, exttrail trail_path/zz, nodbcheckpoint

#DATAPUMP CREATION
add extract DPCOND, exttrailsource trail_path/oc
add rmttrail trail_path/zz, extract DPCOND, megabytes 500
```

- Hint: do not run scripts – execute commands manually

Integration with CRS

- Enables high availability of GG service
  - Relocate between RAC nodes GG with all dependencies (vips, shared file systems…)
- Registration of GG manager as cluster managed resource
  - Doc ID 1527310.1
- Requirements
  - Shared storage for
    - binaries (recommended)
    - trail files (needed)
    - parameter file (recommended)

Integration with CRS with bundled agent

- Register service

```
> $CRS_HOME/bin/agctl add goldengate $gg_service_name
--gg_home $gg_software_home
--oracle_home $rdbms_home
--vip_name ora.${ggmgr_host}.vip
```

- (optional) enable GG process monitoring

```
> agctl modify goldengate $gg_service_name --monitor_extracts
[extracts_list] --monitor_replicats [replicats_list]
```

- Start the service (GG MGR has to be turned off brfore)

```
> agctl start goldengate $gg_service_name --node $rac_server
```

Integration with CRS with bundled agent

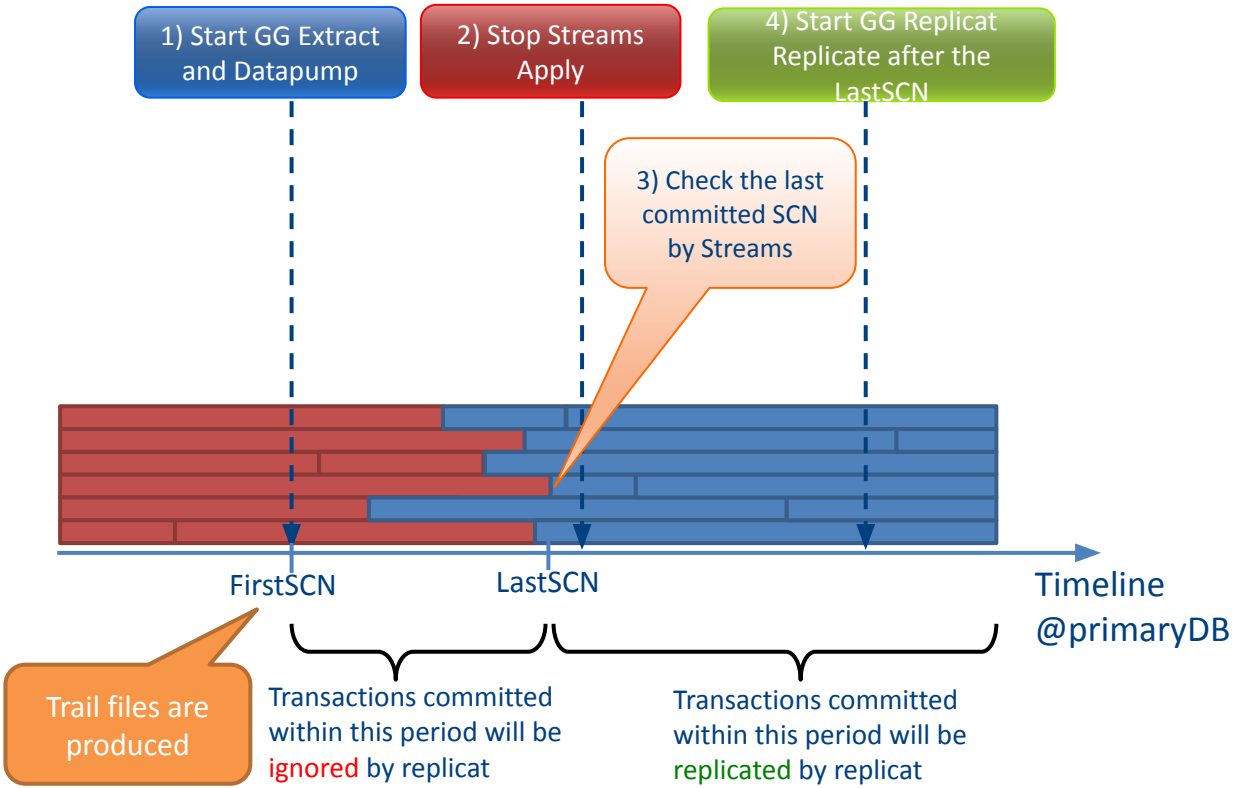
- Checking status

```
> agctl status goldengate my_goldengate
Goldengate  instance 'my_goldengate' is running on serv1
```

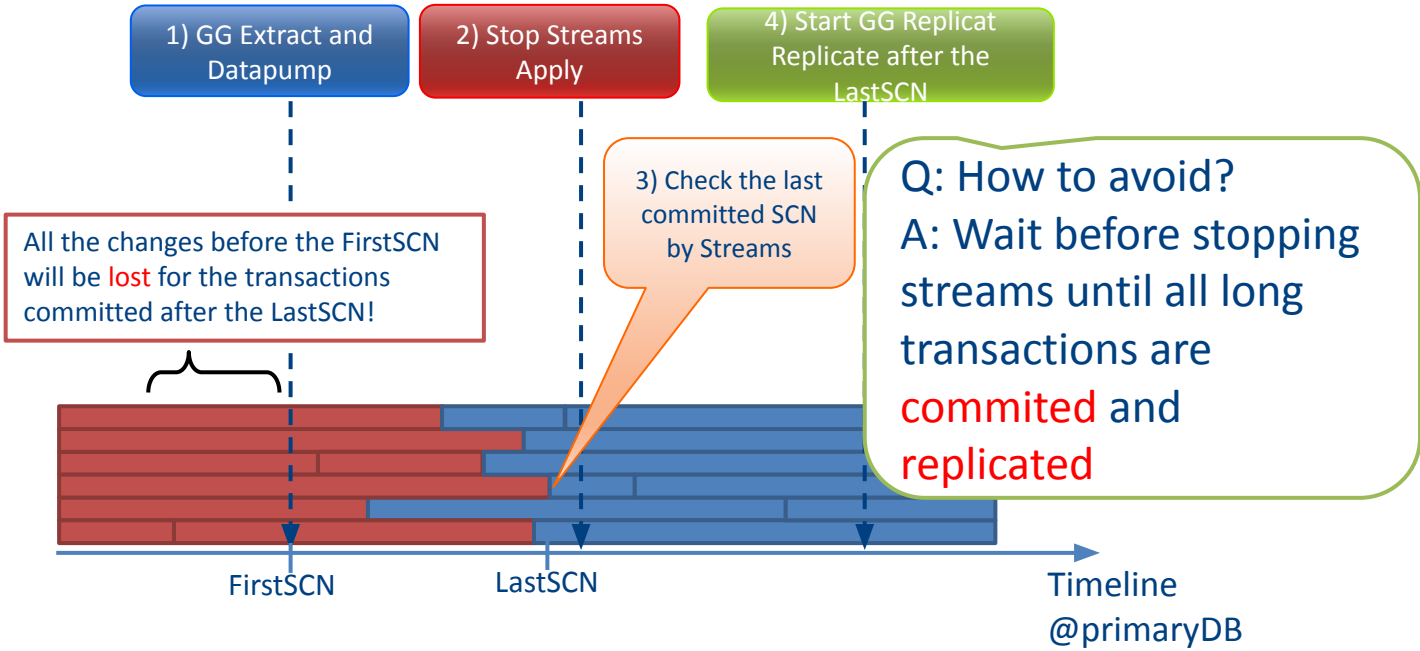
```
>  crsstat.sh

HA Resource                                   Targets                         States
-----------                                   ---------------------------     ------------------------------
ora.LISTENER.lsnr                             ONLINE,ONLINE                   ONLINE on serv1,ONLINE on serv2
ora.LISTENER_SCAN1.lsnr                       ONLINE                          ONLINE on serv2
ora.LISTENER_SCAN2.lsnr                       ONLINE                          ONLINE on serv1
ora.cvu                                       ONLINE                          ONLINE on serv1
ora.serv1.vip                                 ONLINE                          ONLINE on serv1
ora.serv2.vip                                 ONLINE                          ONLINE on serv2
ora.net1.network                              ONLINE,ONLINE                   ONLINE on serv1,ONLINE on serv2
ora.ons                                       ONLINE,ONLINE                   ONLINE on serv1,ONLINE on serv2
ora.scan1.vip                                 ONLINE                          ONLINE on serv2
ora.scan2.vip                                 ONLINE                          ONLINE on serv1
xag.my_goldengate.goldengate                  ONLINE                          ONLINE on serv1
```

Switching from Streams to Goldenagte

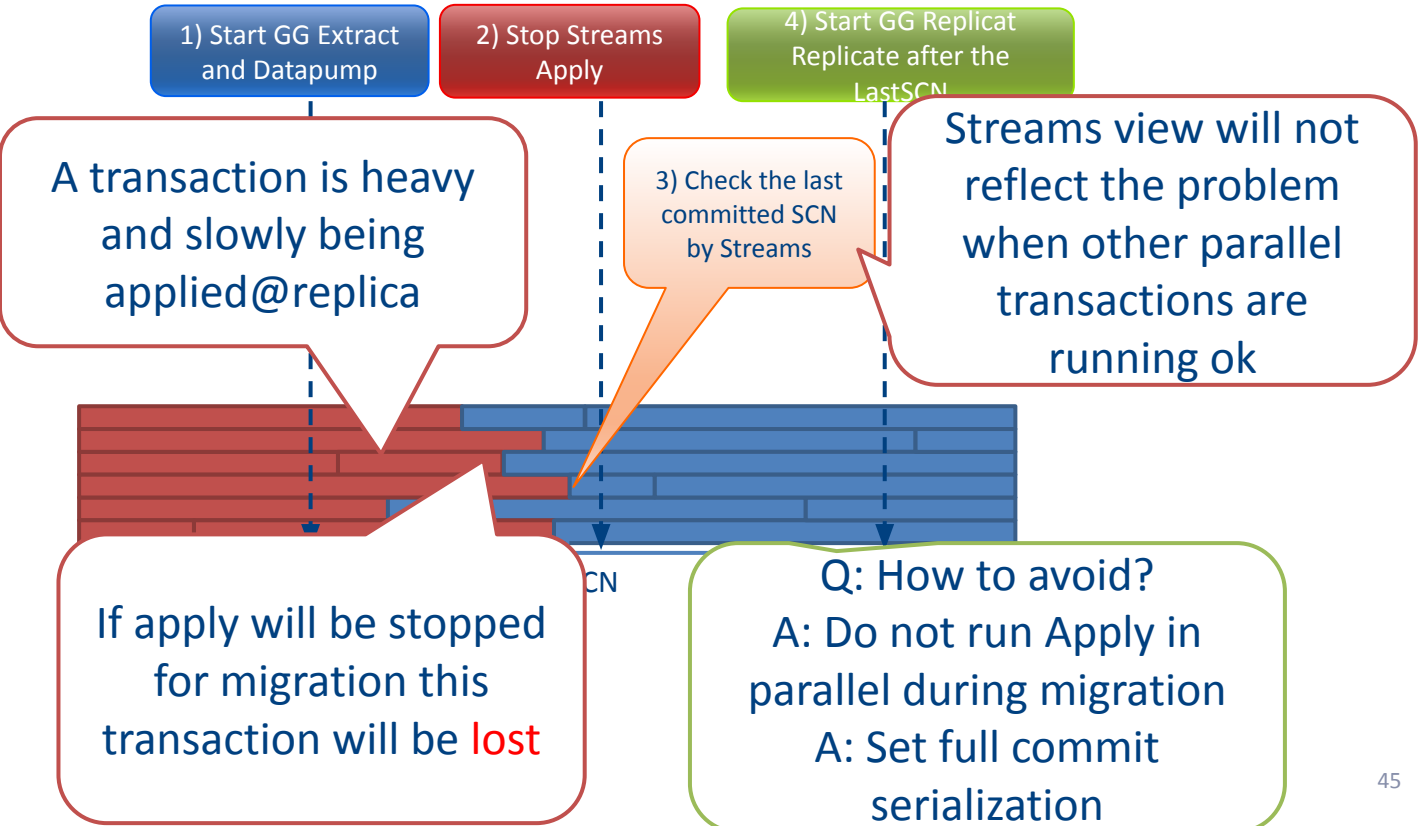
Sequence of actions



What can go wrong (1)



What can go wrong (2)



Replication switching by commands (0)

- create replicat @replica GG home

```
GGSCI> dblogin useridalias ggadm@replica
GGSCI> register replicat CONDREP database
OGG-02528  REPLICAT CONDREP successfully registered with database as
inbound server OGG$CONDREP.
GGSCI> add replicat CONDREP integrated, exttrail trail_path/zz
REPLICAT (Integrated) added.
```

- create datapump @primary GG home

```
GGSCI> add extract DPCOND, exttrailsource trail_path/oc
EXTRACT added.
GGSCI> add rmttrail trail_path/zz, extract DPCOND, megabytes 500
RMTTRAIL added.
```

- create extract @primary GG Home (note the First SCN)

```
GGSCI> dblogin useridalias ggadm@primary
GGSCI> register extract CAPTCOND database
Extract CAPTCOND successfully registered with database at SCN
56532342342.
GGSCI> add extract CAPTCOND, integrated tranlog, scn 56532342342
EXTRACT added.
GGSCI> add exttrail trail_path/oc, extract CAPTCOND, megabytes 500
EXTTRAIL added.
```

Replication switching by commands (1)

- Disable Streams Apply parallelism and enable full commit serialization @replica

```
SQL> exec dbms_apply_adm.set_parameter
('MY_APP',parameter=>'COMMIT_SERIALIZATION',value=>'FULL');
SQL> exec dbms_apply_adm.set_parameter
('MY_APP',parameter=>'PARALLELISM',value=>'1');
```

- Start datapump and extract @primary GG Home

```
GGSCI> start DPCOND
GGSCI> start CAPCOND
```

- Wait until there are no transaction older than extract's 'First SCN' @primary

```
SQL> select count(*) from gv$transaction where START_SCN<56532342342
```

Replication switching by commands (2)

- Wait until Streams applied SCN > 'First SCN' @replica

```
select LWM_MESSAGE_NUMBER from V$STREAMS_APPLY_COORDINATOR where
apply_name= 'MY_APPLY'and LWM_MESSAGE_NUMBER>56532342342
```

- Stop apply @replica

```
exec dbms_apply_adm.stop_apply('MY_APPLY');
```

- Check SCN of last applied transaction by Streams @replica

```
select APPLIED_MESSAGE_NUMBER from DBA_APPLY_PROGRESS where
apply_name= 'MY_APPLY'
```

- Start replicat using SCN from previous step @replica GGH

```
start CONDREP aftercsn [applied_message_number]
```

- That's it!

Replication switching by commands (3)

- Check if extract is running and replicating

```
info all
```

```
info CONFREP
```

```
info all
```

```
info all
```

```
stats CONDREP
```

```
info all
```

```
Info CONFREP
```

```
stats CONDREP
```

Data consistency validation

- What are the options
  - select…minus…select@source…? If one small table
  - Compare and converge…? If less than 10 tables
- Otherwise, Veridata is more convenient
  - Took hours to complete
    - 1.2TB was checked in 14h within CERN network, ~50 hours for remote centers
    - There were some false positives
    - We used default Veridata configuration – something could go suboptimal
  - It built our confidence that everything went ok with the migrations

Streams components removal

- Do not use dbms_streams.remove_streams_configuration

- Drop components step by step with
  - dbms_capture_adm.drop_capture
  - dbms_apply_adm.delete_all_errors(apply_name)
  - dbms_apply_adm.drop_apply
  - dbms_propagation_adm.drop.propagation
  - dbms_streams.remove_queue (x2)

- Do not remove processes or queues with OGG$ in the name

Click to edit Master title style

After the migration

How do we monitor GG

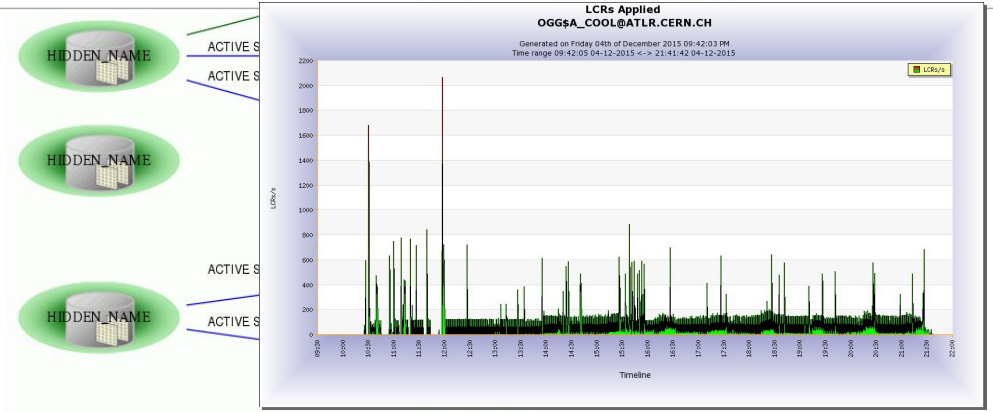
- Director for the central GG cluster
  - Lag and status of *Extracts* and *Replicats* on central cluster
- Custom monitoring for GG integrated back-ends (*Capture* and *Apply*)
  - process status and latencies
  - data flows (LCRs) between databases
  - uses heartbeat tables
  - sends mails/sms notifications

Home-made replication monitoring

ACTIVE STREAMS

#	Stream	LCRs Cap	LCRs Enq	LCRs Deq	LCRs App	Cap Latency	Replication time	Since last shipment	Capture State	Apply State	State
1	HIDDEN_SOURCE=>HIDDEN_TARGET	30.2 /s	29.4 /s	40.33 /s	40.33 /s	1 sec	1 min 21 sec	15 sec	WAITING FOR TRANSACTION	IDLE	
2	HIDDEN_SOURCE=>HIDDEN_TARGET	30.2 /s	29.4 /s	0 /s	0 /s	1 sec	7 sec	22 sec	WAITING FOR TRANSACTION	IDLE	
3	HIDDEN_SOURCE=>HIDDEN_TARGET	30.2 /s	29.4 /s	0 /s	0 /s	1 sec	11 min 21 sec	27 sec	WAITING FOR TRANSACTION	IDLE	
4	HIDDEN_SOURCE=>HIDDEN_TARGET	92.58 /s	81.6 /s	1.05 /s	1.25 /s	1 sec	13 sec	1 sec	WAITING FOR TRANSACTION	IDLE	
5	HIDDEN_SOURCE=>HIDDEN_TARGET	89.71 /s	89.66 /s	48.79 /s	47.3 /s	2 sec	4 sec	1 sec	WAITING FOR TRANSACTION	IDLE	
6	HIDDEN_SOURCE=>HIDDEN_TARGET	214.71 /s	195.09 /s	0 /s	0 /s	2 sec	52 sec	7 min 43 sec	WAITING FOR TRANSACTION	IDLE	
7	HIDDEN_SOURCE=>HIDDEN_TARGET	4.06 /s	3.91 /s	0 /s	0 /s	7 sec	58 min 59 sec	1 min 24 sec	WAITING FOR TRANSACTION	IDLE	

04-12-15 21:39:50 (3s ago) auto refresh



Useful GG db views

- Integrated Extract
  - **DBA_CAPTURE** & **V$GOLDENGATE_CAPTURE**
    - details about log miner session (state, progress, etc)

- Integrated Replicat
  - **DBA_APPLY** – config and process status
  - **V$GG_APPLY_READER** – LCR-level statistics
  - **V$GG_APPLY_COORDINATOR** – transaction-level stats
  - **V$GG_APPLY_SERVER** – status of transactions being applied
  - **V$GOLDENGATE_TABLE_STATS**
    - Changes counters for all tables

Problems with integrated GG (so far)

- No major issues so far!
- Network glitches are not well detected
  - *extract* or *replicat* can hang instead of abort
  - MGR does not detect such frozen processes
  - Manual (hard) restarts are needed
- Logminer crashes quietly while processing big transaction
  - Not detected by *extract*
  - Manual (hard) restarts are needed

Some best practices

- http://www.oracle.com/technetwork/database/availability/maa-gg-performance-1969630.pdf
- Use heartbeat table to validate replication
- Do not use system generated names
- Grantee must exist at replica destinations
- Dumping dictionary to redo every day
- Checking for relevant patches (**Doc Id 1557031.1**)

Future plans

- Move GG monitoring to EM (12.1.0.3.0)
  - Automatic hang detections and handling
- Upgrade to GG 12.2.0.1
- Validate GG for near-real-time
  - migrations, consolidations
- GoldenGate as a real-time data integrator for Hadoop

Summary

- Replication technology evolution at CERN:
  - Oracle Streams (initial solution) was replaced by Golden Gate12c and Active Data Guard
  - improved availability and performance of the data replication services
- The transition was painless
  - The procedures are already well established
  - Still cannot be easily automatize
- We use centralized GG installation
  - Integrated *extract* and *replicat*, without *datapump*
  - Works well so far

Acknowledgments

- CERN IT-DB group
  - Especially: Lorena Lobato Pardavila, Eva Dafonte Perez

- Oracle (via the Openlab partnership)
  - Patricia McElroy, Jagdev Dhillon, Greg Doherty, Monica Marinucci, Kevin Jernigan

Questions?

zbigniew.baranowski@cern.ch