

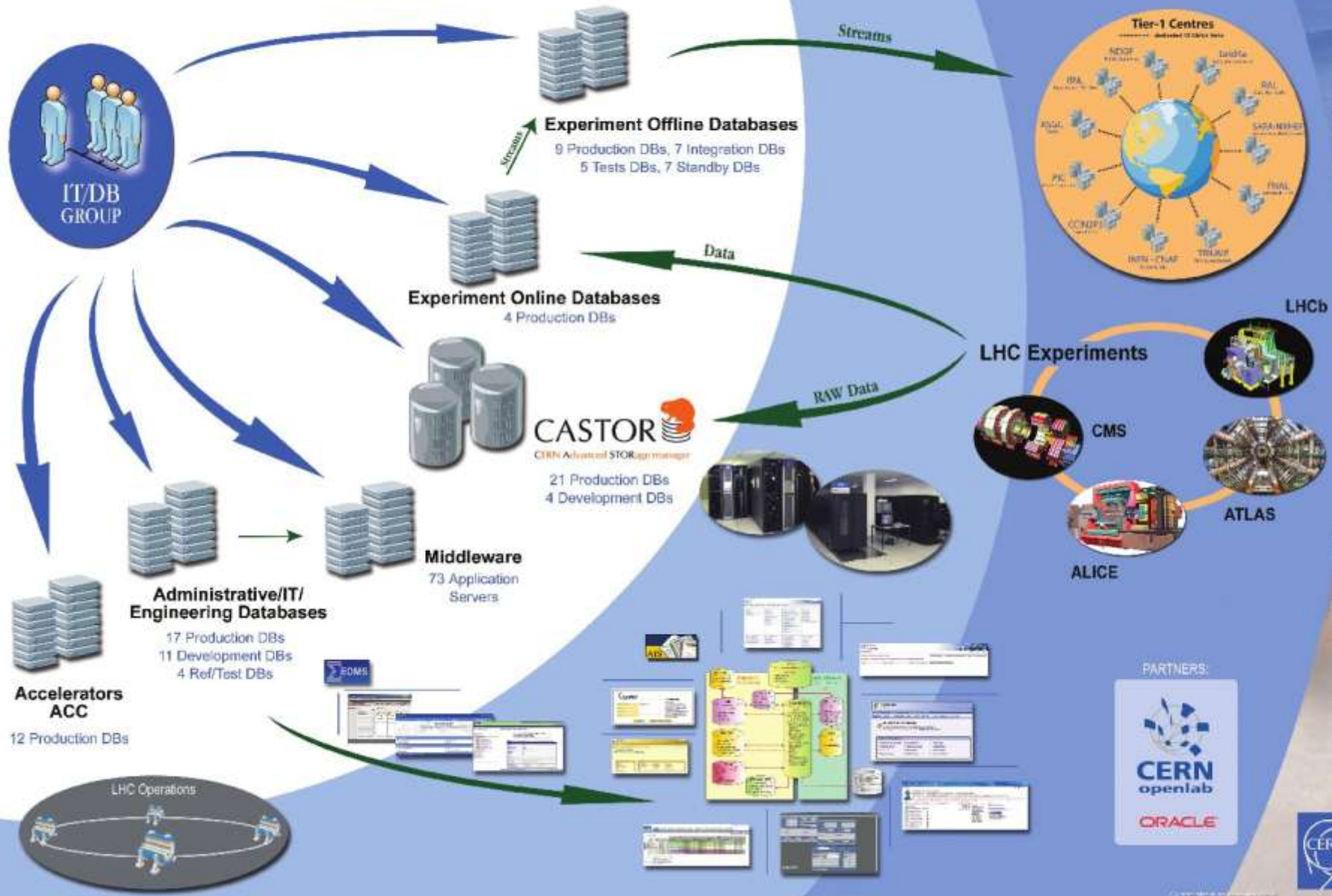
Oracle at CERN

CERN openlab summer students programme 2011



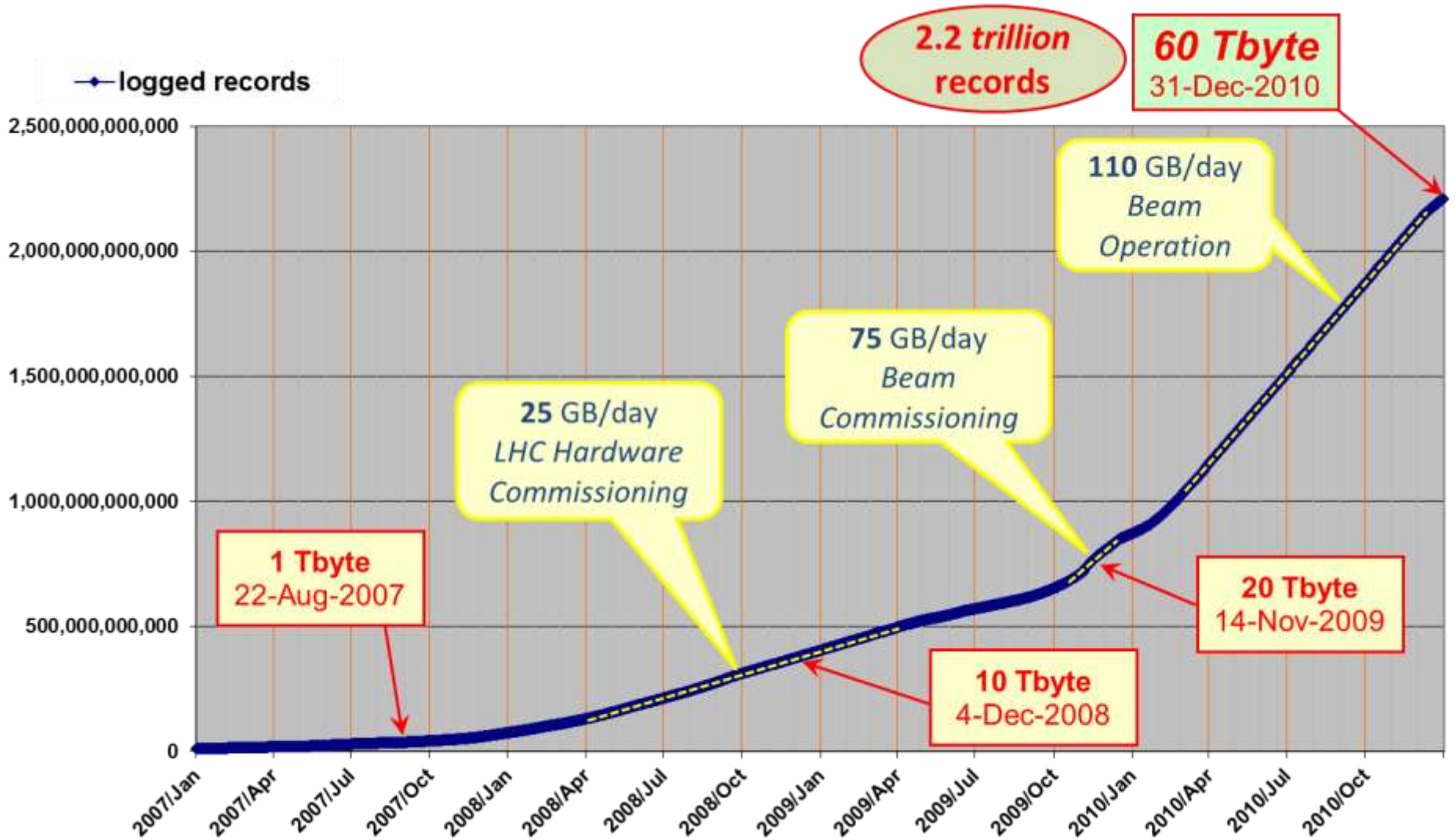
Eric Grancher
eric.grancher@cern.ch
CERN IT department

<http://cern.ch/it-dep/db/>



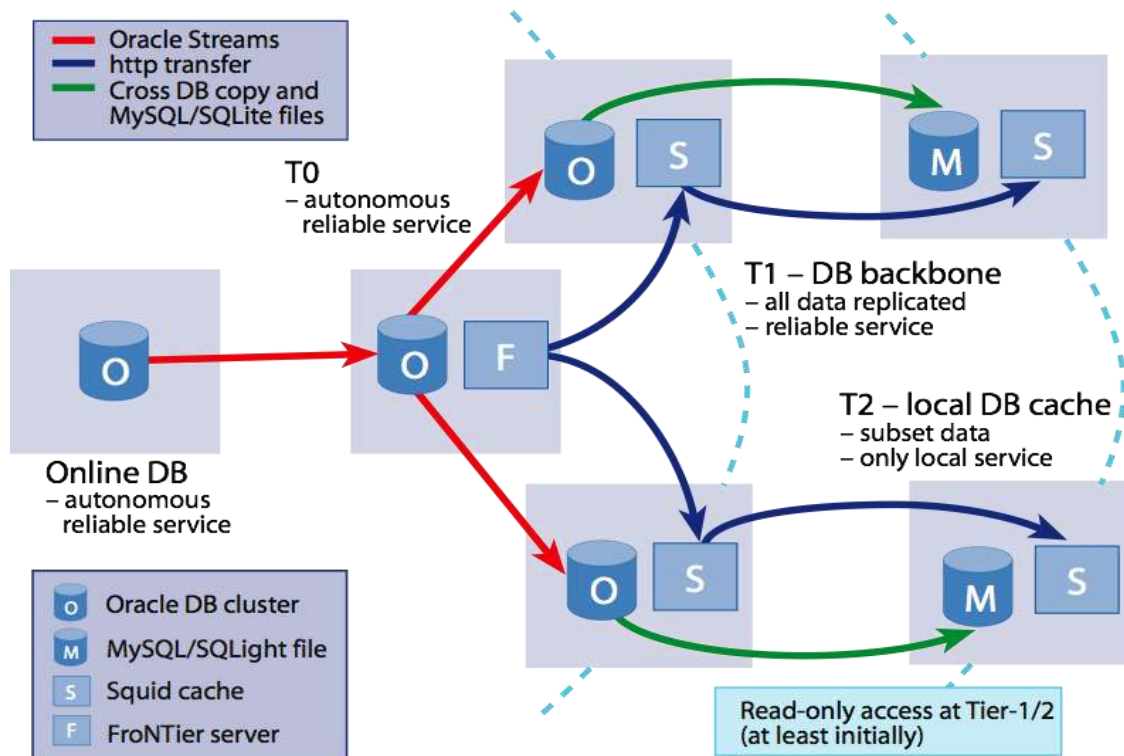
- CERN databases services
 - ~130 databases, most of them database clusters (Oracle RAC technology RAC, 2 – 6 nodes)
 - Currently over 3000 disk spindles providing more than ~3PB raw disk space (NAS and SAN)
 - MySQL service started (for Drupal)
- Some notable databases at CERN
 - Experiments' databases – 14 production databases
 - Currently between 1 and 12 TB in size
 - Expected growth between 1 and 10 TB / year
 - LHC accelerator logging database (ACCLOG) – ~70 TB, $>2 \cdot 10^{12}$ rows, expected growth up to 35(+35) TB / year
 - ... Several more DBs in the 1-2 TB range

LHC logging service, $>2.10^{12}$

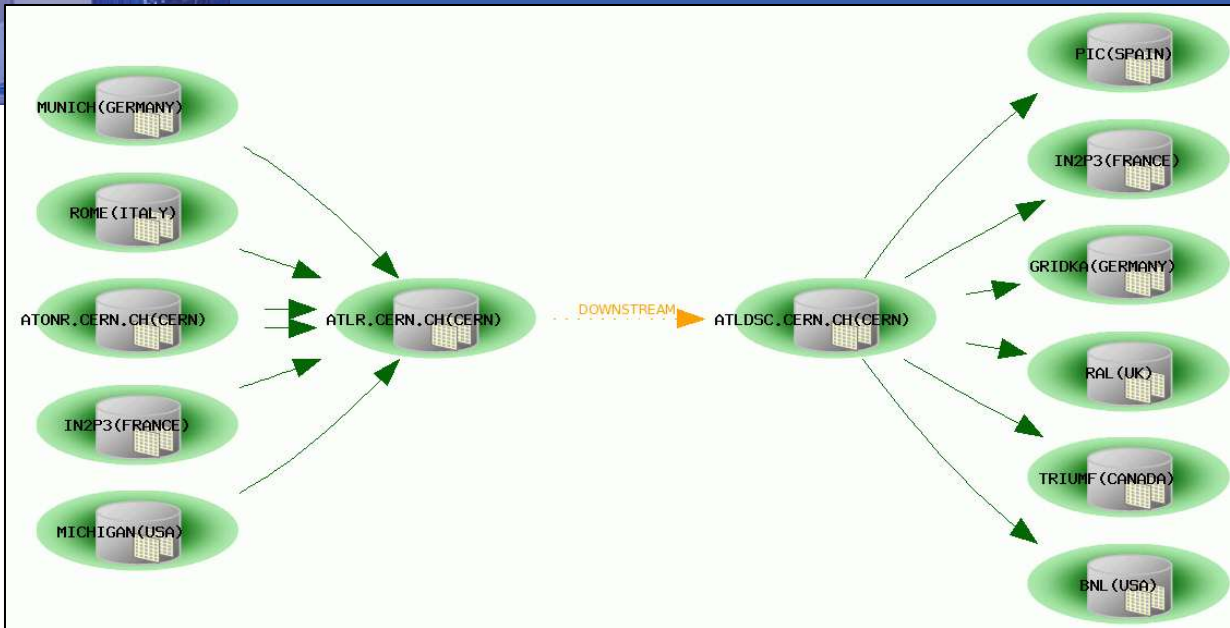


- Online acquisition, offline production, data (re)processing, data distribution, analysis
 - SCADA, conditions, geometry, alignment, calibration, file bookkeeping, file transfers, etc..
- Grid Infrastructure and Operation services
 - Monitoring, Dashboards, User-role management, ..
- Data Management Services
 - File catalogues, file transfers and storage management, ...
- Metadata and transaction processing for custom tape-based storage system of physics data
- Accelerator control and logging systems
- AMS as well: data/mc production bookkeeping and slow control data

- Worldwide distribution of experimental physics data using Oracle Streams



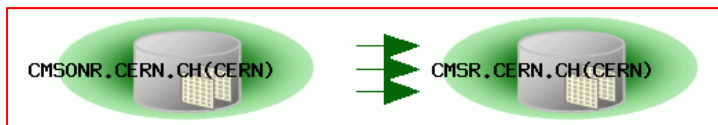
▶ Huge effort, successful outcome



ATLAS

LHCb

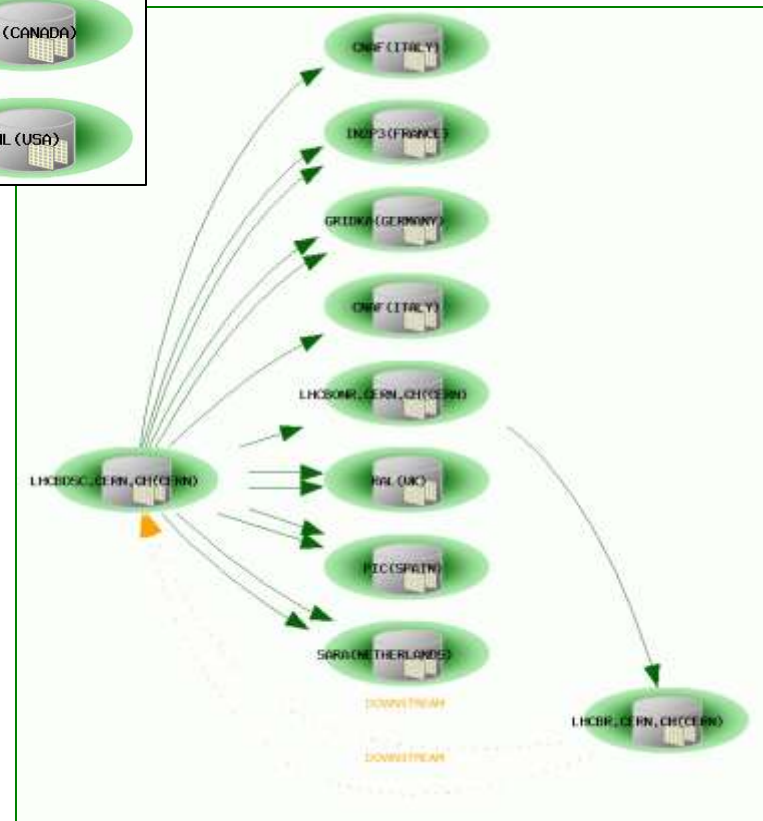
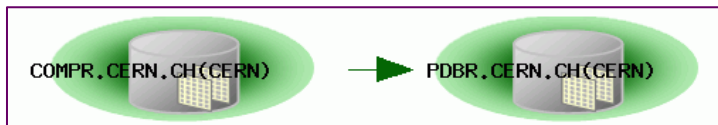
CMS



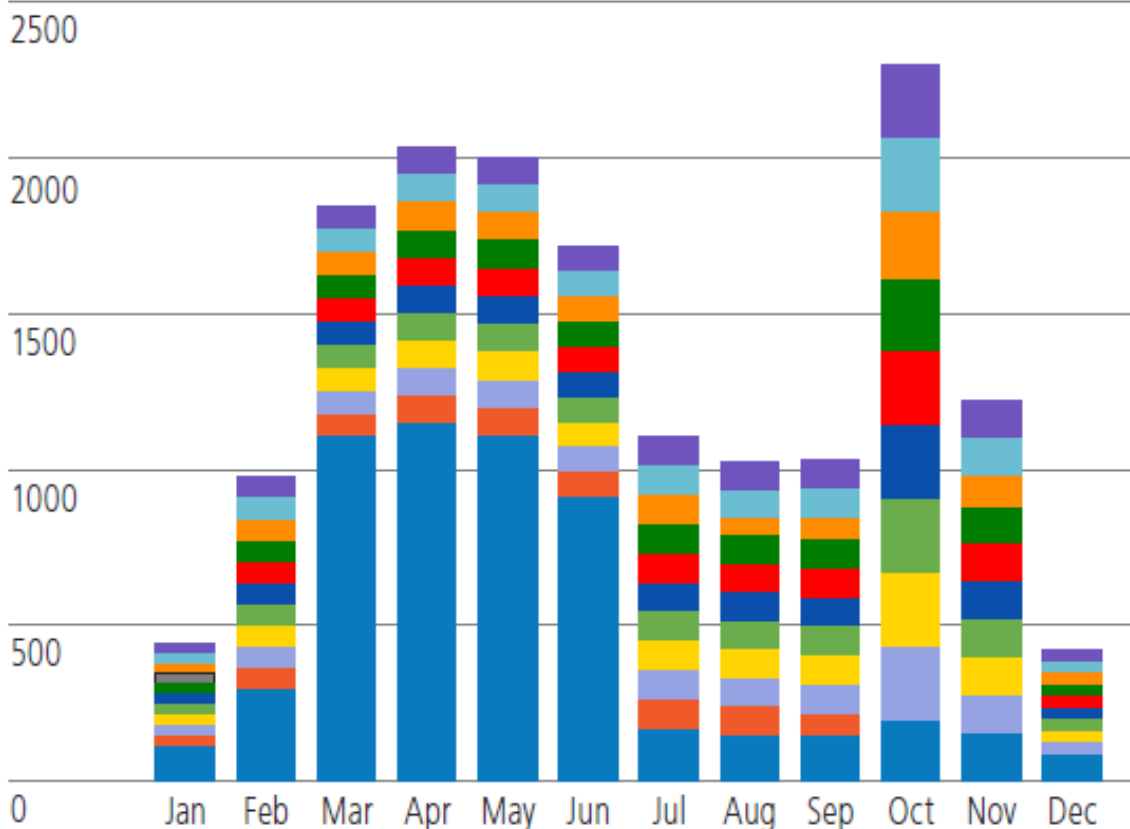
ALICE



COMPASS



Number of Logical Change Records (LCRs) in Millions, per Month, by Tier-1 Site



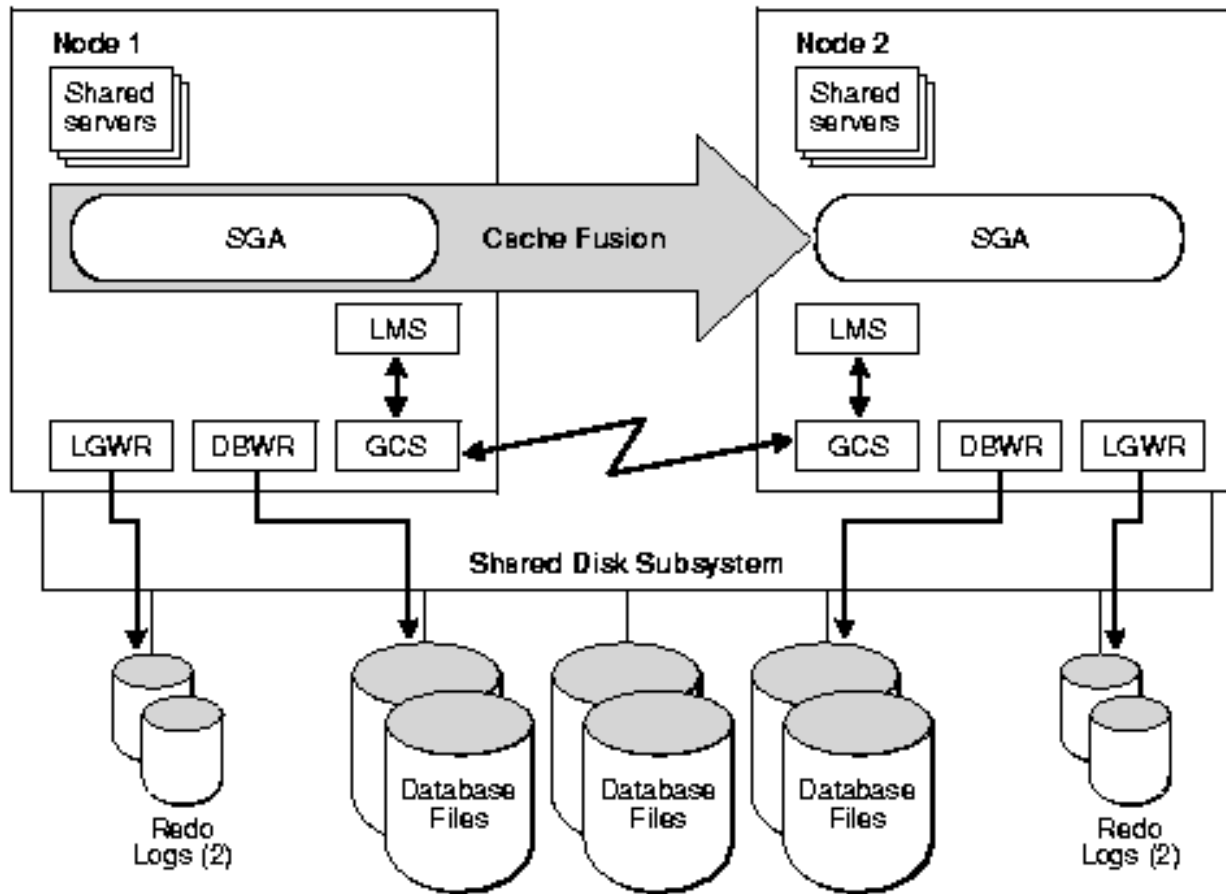
Replication rate for conditions data from the ATLAS experiment to the different WLCG Tier-1 sites in 2010

- All actions
 - Network related
 - IO related
 - Internals (cluster communication, space management, etc.)
 - Application related (transaction locks, etc.)
 - etc.
- Key for “scientific” performance understanding.

- Single.java
 - Tanel Poder's snapper.sql
 - Disable autocommit
 - Tanel Poder's snapper.sql
- Batch.java

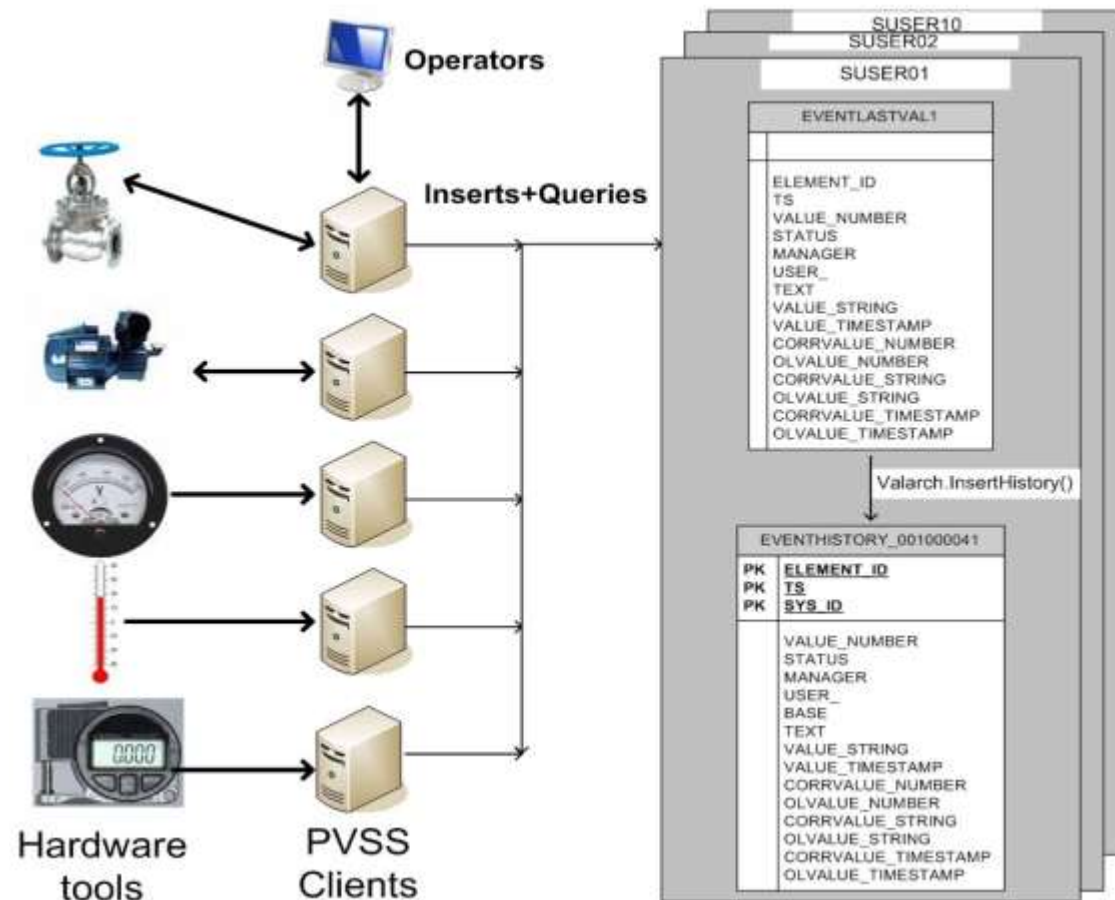
DEMO

Oracle Real Application Cluster



PVSS Oracle scalability

- Target = 150 000 changes per second (tested with 160k)
- 3 000 changes per client
- 5 nodes RAC 10.2.0.4
- 2 NAS 3040, each with one aggregate of 13 disks (10k rpm FC)



The Tuning Process

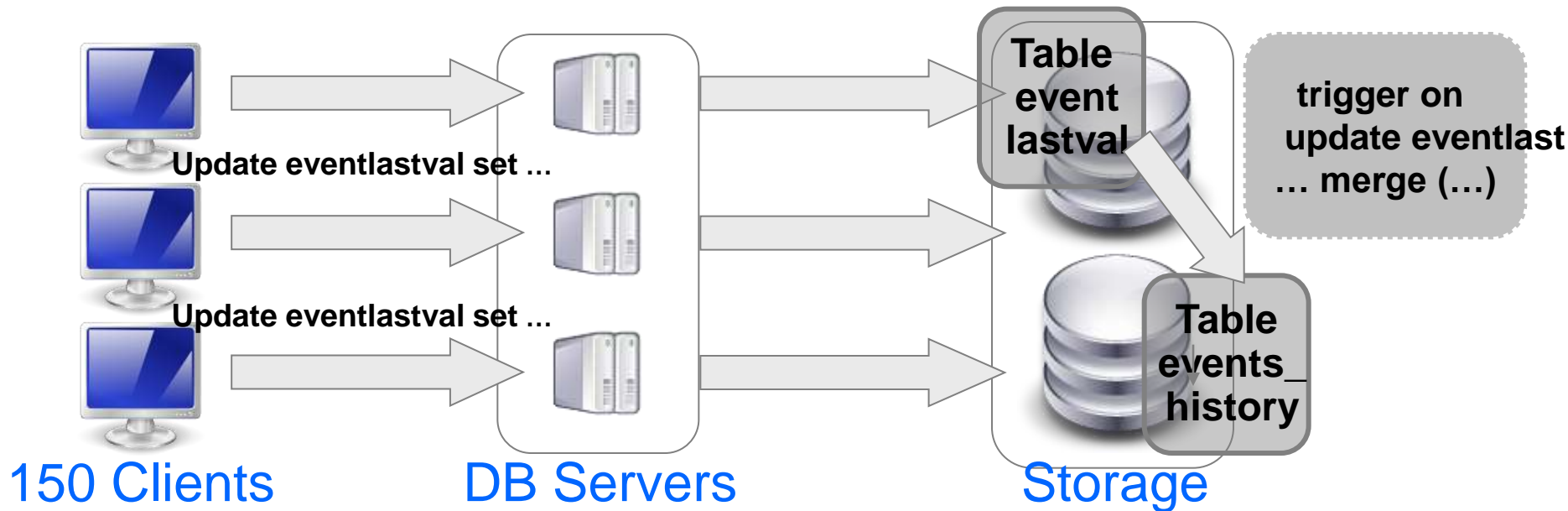
1. run the workload,
gather ASH/AWR
information, 10046...

2. find the top
event that slows
down
the processing



4. modify client
code, database
schema,
database code,
hardware
configuration

3. understand why time
is spent on this event



- **Shared resource:**
EVENTS_HISTORY (ELEMENT_ID, VALUE...)
- Each client “measures” input and registers history with a “merge” operation in the EVENTS_HISTORY table

Performance:

- **100** “changes” per second

Initial state observation:

- database is waiting on the clients
“**SQL*Net message from client**”
- Use of a generic library C++/DB
- Individual insert (one statement per entry)
- Update of a table which keeps “latest state” through a trigger

Changes:

- bulk insert to a temporary table with OCCI, then call PL/SQL to load data into history table

Performance:

- **2000** changes per second

Now top event: “**db file sequential read**”

[awrrpt 1 5489 5490.html](#)

Event	Waits	Time(s)	Percent Total DB Time	Wait Class
db file sequential read	29,242	137	42.56	User I/O
enq: TX - contention	41	120	37.22	Other
CPU time		61	18.88	
log file parallel write	1,133	19	5.81	System I/O
db file parallel write	3,951	12	3.73	System I/O

Changes:

- Index usage analysis and reduction
- Table structure changes. IOT.
- Replacement of merge by insert.
- Use of “direct path load” with ETL

Performance:

- **16 000** “changes” per second
- Now top event: **cluster related wait event**

[test5 rac node1 8709 8710.html](#)

Event	Waits	Time(s)	Avg Wait(ms)	% Total Call Time	Wait Class
gc buffer busy	27,883	728	26	31.6	Cluster
CPU time		369		16.0	
gc current block busy	6,818	255	37	11.1	Cluster
gc current grant busy	24,370	228	9	9.9	Cluster
gc current block 2-way	118,454	198	2	8.6	Cluster

Changes:

- Each “client” receives a unique number.
- Partitioned table.
- Use of “direct path load” to the partition with ETL

Performance:

- **150 000** changes per second
- Now top event : “**freezes**” once upon a while

[rate75000_awrrpt_2_872_873.html](#)

Event	Waits	Time(s)	Avg Wait(ms)	% Total Call Time	Wait Class
row cache lock	813	665	818	27.6	Concurrency
gc current multi block request	7,218	155	22	6.4	Cluster
CPU time		123		5.1	
log file parallel write	1,542	109	71	4.5	System I/O
undo segment extension	785,439	88	0	3.6	Configuration

Problem investigation:

- Link between foreground process and ASM processes
- Difficult to interpret ASH report, 10046 trace

Problem identification:

- ASM space allocation is blocking some operations

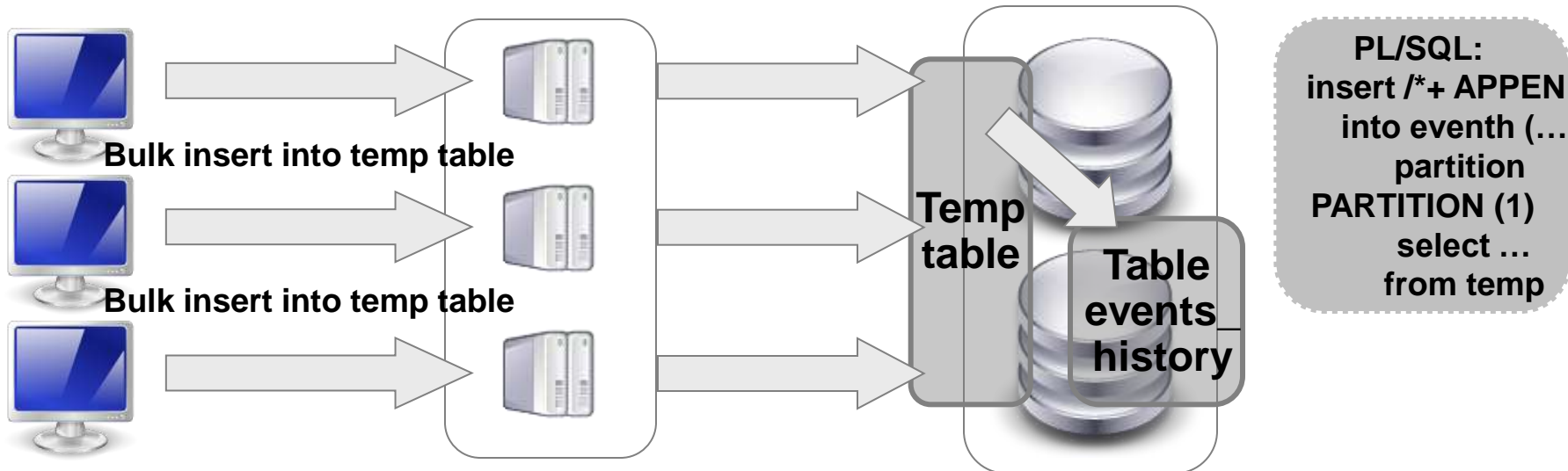
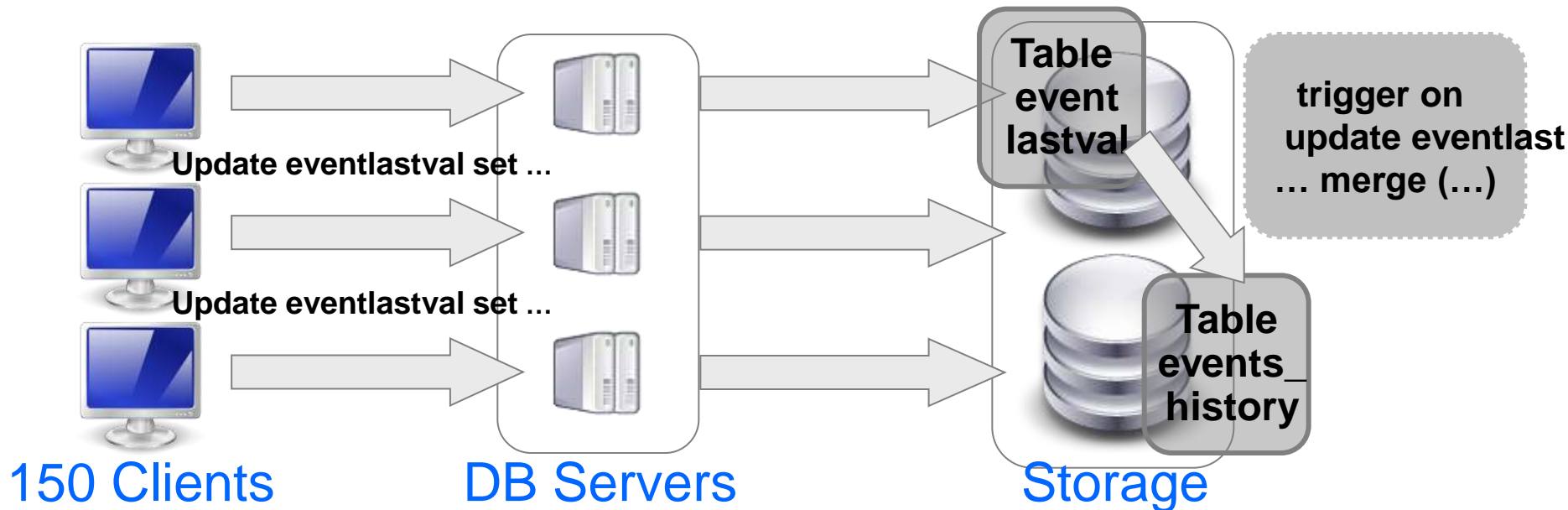
Changes:

- Space pre-allocation, background task.

Result:

- **Stable 150 000** “changes” per second.

PVSS Tuning Schema



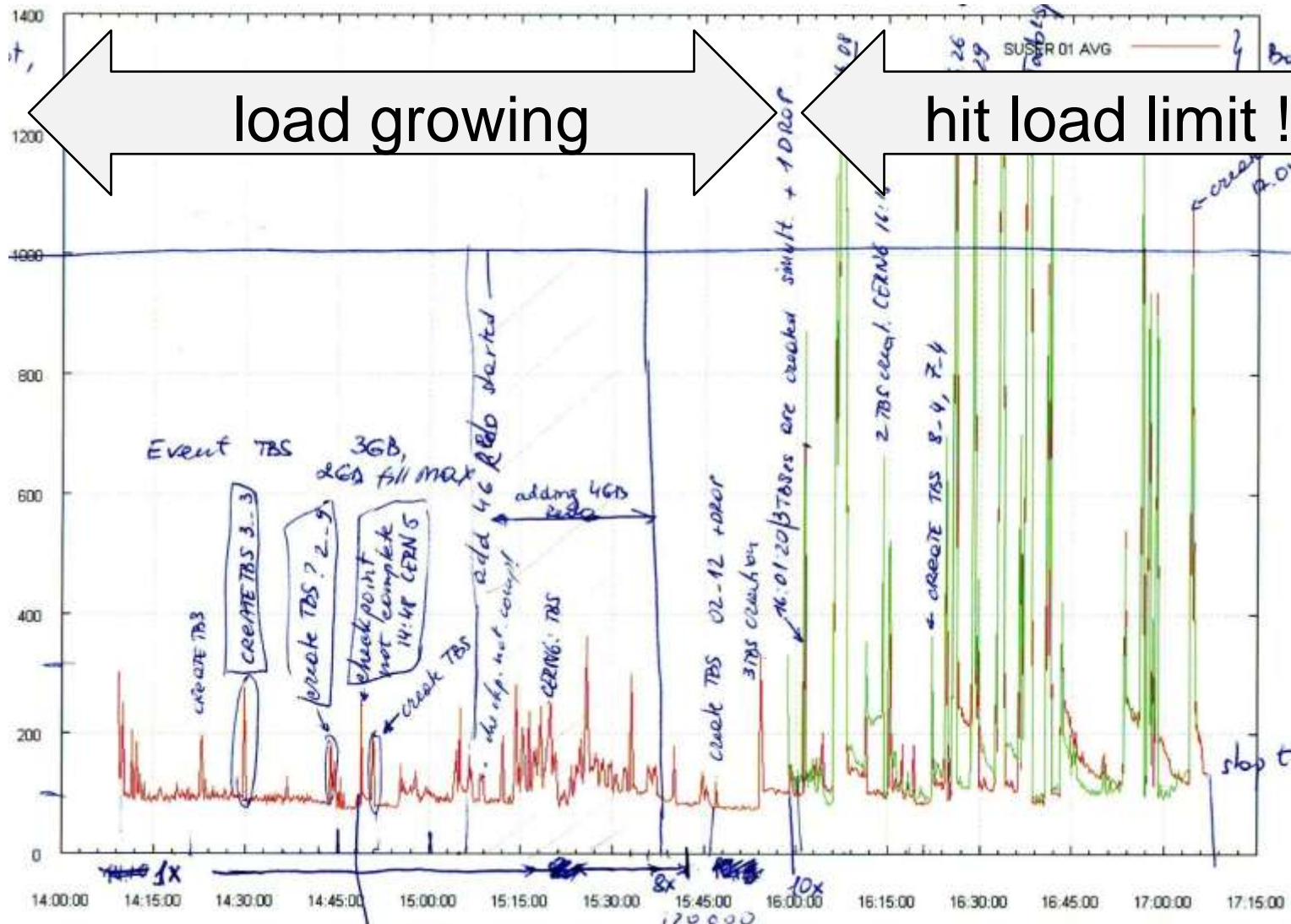
Conclusion:

- from **100** changes per second to **150 000** “changes” per second
- **6 nodes RAC** (dual CPU, 4GB RAM), 32 disks SATA with FCP link to host
- **4 months effort:**
 - Re-writing of part of the application with changes interface (C++ code)
 - Changes of the database code (PL/SQL)
 - Schema change
 - Numerous work sessions, joint work with other CERN IT groups

- Observed many times: “the storage is slow” (and storage administrators/specialists say “storage is fine / not loaded”)
- Typically happens that observed (from Oracle rdbms point of view) IO wait times are long if CPU load is high
- Instrumentation / on-off cpu

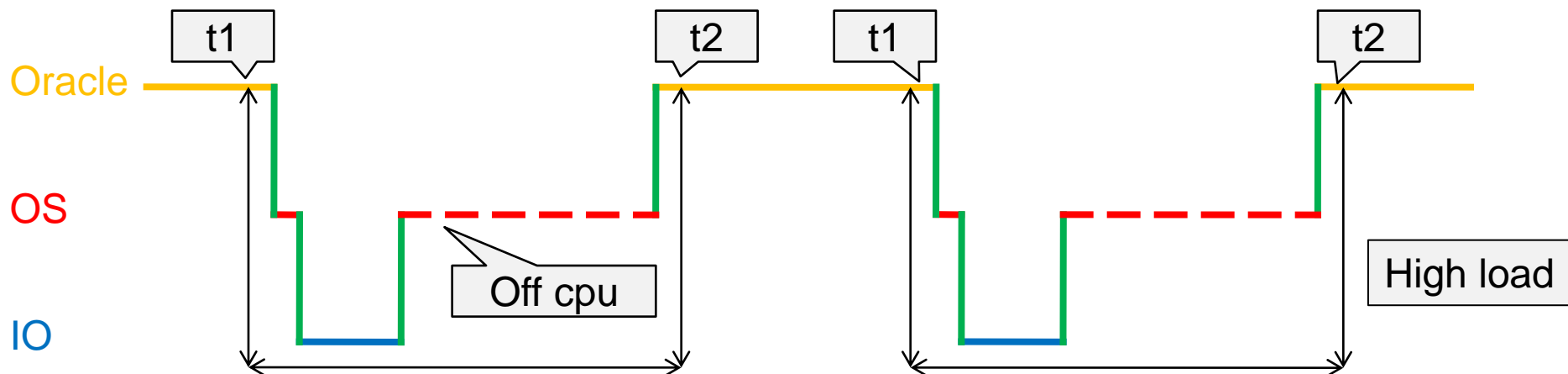
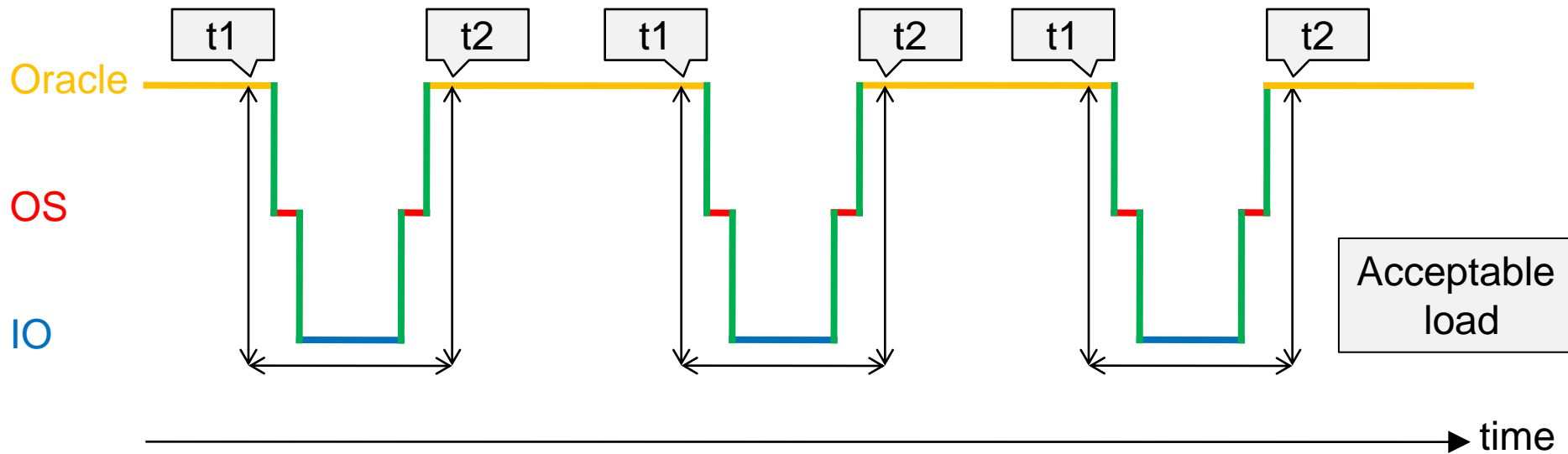
Overload at CPU level (2/) example

Insertion time (ms), has to be less than 1000ms



15k... 30k ... 60k... 90k... 120k ...135k... || 150k (insertions per second)

OS level / high-load



- Dtrace (Solaris) can be used at OS level to get (detailed) information at OS level

```
syscall::pread:entry
/pid == $target && self->traceme == 0 /
{
    self->traceme = 1;
    self->on = timestamp;
    self->off= timestamp;
    self->io_start=timestamp;
}
```

```
syscall::pread:entry
/self->traceme == 1 /
{
    self->io_start=timestamp;
}
```

```
syscall::pread:return
/self->traceme == 1 /
{
    @avgs["avg_io"] = avg(timestamp-self->io_start);
    @[tid,"time_io"] = quantize(timestamp-self->io_start);
    @counts["count_io"] = count();
}
```

```
sched:::on-cpu
/pid == $target && self->traceme == 1 /
{
    self->on = timestamp;
    @[tid,"off-cpu"] = quantize(self->on - self->off);
    @totals["total_cpu_off"] = sum(self->on - self->off);
    @avgs["avg_cpu_off"] = avg (self->on - self->off);
    @counts["count_cpu_on"] = count();
}
sched:::off-cpu
/self->traceme == 1/
{
    self->off= timestamp;
    @totals["total_cpu_on"] = sum(self->off - self->on);
    @avgs["avg_cpu_on"] = avg(self->off - self->on);
    @[tid,"on-cpu"] = quantize(self->off - self->on);
    @counts["count_cpu_off"] = count();
}

tick-1sec
/i++ >= 5/
{
    exit(0);
}
```

Dtrace, “normal load”

```
-bash-3.00$ sudo ./cpu.d4 -p 15854
dtrace: script './cpu.d4' matched 7 probes
CPU      ID                FUNCTION:NAME
  3  52078                :tick-1sec

avg_cpu_on                169114
avg_cpu_off               6768876
avg_io                    6850397

[...]
```

```
1  off-cpu
   value  ----- Distribution ----- count
   524288 |                                0
  1048576 |                                2
  2097152 | @@@@                                86
  4194304 | @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@ 577
  8388608 | @@@@@@@@@@@@                        189
 16777216 |                                2
 33554432 |                                0

[...]
```

```
count_cpu_on                856
count_io                    856
count_cpu_off               857
total_cpu_on                144931300
total_cpu_off               5794158700
```

Dtrace, "high load"

```
-bash-3.00$ sudo ./cpu.d4 -p 15854
dtrace: script './cpu.d4' matched 7 probes
CPU      ID                FUNCTION:NAME
  2  52078                :tick-1sec

avg_cpu_on                210391
avg_cpu_off              10409057
avg_io                    10889597
[...]
```

1	off-cpu				count
	value	-----	Distribution	-----	
	8192				0
	16384				4
	32768	@			11
	65536				2
	131072				0
	262144				0
	524288				0
	1048576				0
	2097152	@			15
	4194304	@@@@@@@@@@@@@@@@			177
	8388608	@@@@@@@@@@@@@@@@			249
	16777216	@@@			41
	33554432				4
	67108864				0

```
[...]
```

count_io	486
count_cpu_on	503
count_cpu_off	504
total_cpu_on	106037500
total_cpu_off	5235756100

- Aiming for high-availability is (often) adding complexity... and complexity is the enemy of availability
- Scalability can be achieved with Oracle Real Application Cluster (150k entries/s for PVSS)
- Database / application instrumentation is key for understanding/improving performance
- NFS/D-NFS/pNFS are solutions to be considered for stability and scalability (very positive experience with NetApp, snapshots, scrubbing, etc.)
- Database independence is very complex if performance is required
- Hiding IO errors from the database leaves the database handle what it is best at (transactions, query optimisation, coherency, etc.)

- 5TB datafile
 - Every 1s update and select
 - Backup using snapshot
- Corrupt datafile
 - Restore from snapshot
 - rman recover
 - SQL*Plus online
- select

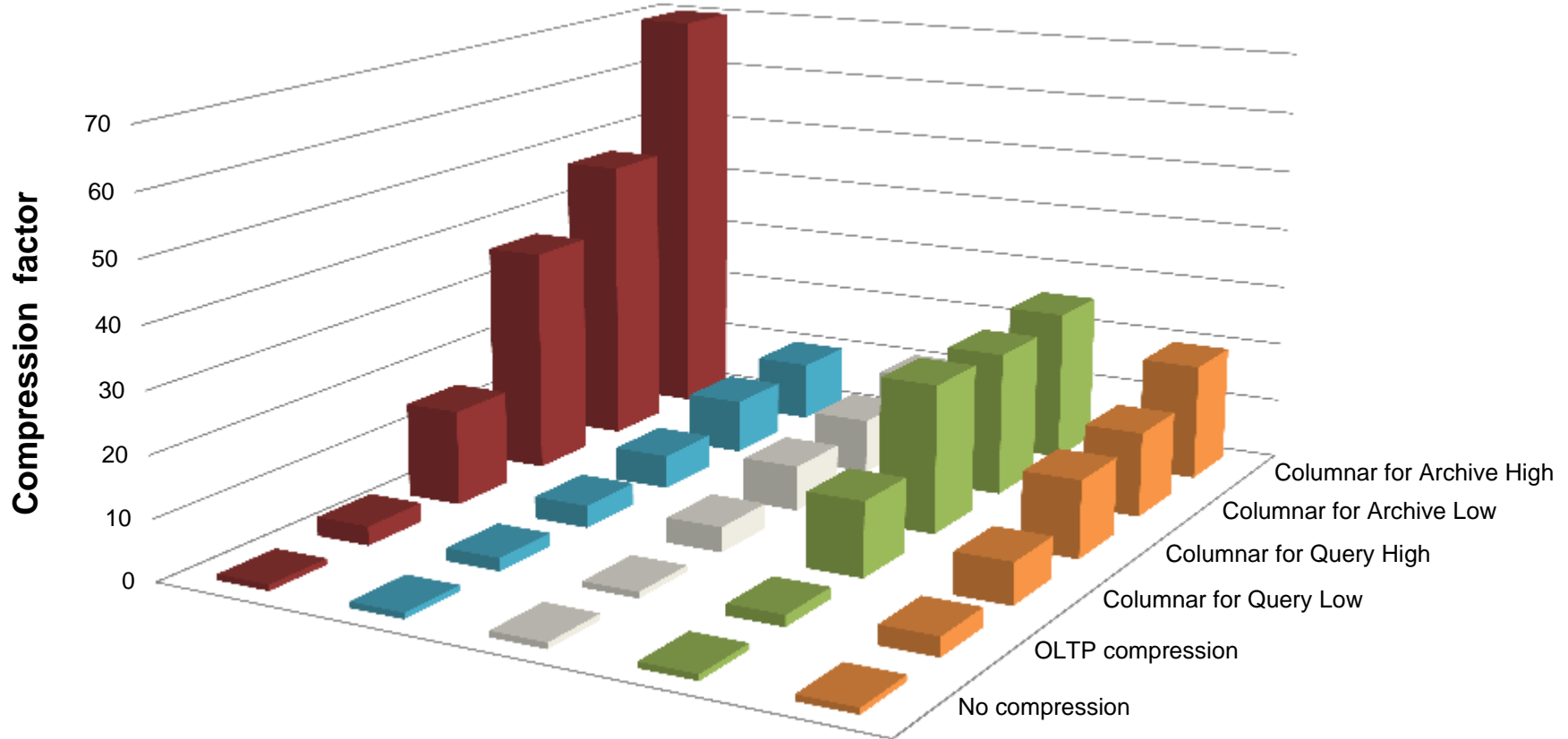
DEMO

- Flash
- Large memory systems
- Compression
- Open source “relational” databases
- NoSQL databases

- Flash changes the picture in the database area IO
 - Sizing for IO Operations Per Second
 - Usage of fast disks for high number of IOPS and latency
- Large amount of memory
 - Enables consolidation and virtualisation (less nodes)
 - Some databases fully in memory
- Compression is gaining momentum for databases
 - For example Oracle's hybrid columnar compression
 - Tiering of storage

Exadata Hybrid Columnar Compression on Oracle 11gR2

Measured Compression factor for selected Physics Apps.



- PVSS (261M rows, 18GB)
- LCG TESTDATA 2007 (103M rows, 75GB)
- ATLAS LOG MESSAGES (323M rows, 66GB)

- LCG GRID Monitoring (275M rows, 7GB)
- ATLAS PANDA FILESTABLE (381M rows, 120GB)

- “Big Data”, analysis of large amount of data in reasonable of time
- Goole MapReduce, Apache Hadoop implementation
- Oracle Exadata
 - Storage cells perform some of the operations locally (Smart Scans, storage index, column filtering, etc.)
- Important direction for *at least* the first level of selection

- Oracle
 - Critical component for LHC accelerator and physics data processing
 - Scalable and stable, including data replication
- CERN central services run on Oracle, for which we have components and experience to build high availability, guaranteed data, scalability
- MySQL is being introduced
 - Nice features and light, lacks some scalability and High-Availability features for solid service
- NoSQL is being considered
 - Ecosystem still in infancy (architecture, designs and interfaces subject to change!)

- NoSQL ecosystem, <http://www.aosabook.org/en/nosql.html>
- Database workshop at CERN <https://indico.cern.ch/conferenceDisplay.py?confId=130874>
and ATLAS Computing Technical Interchange Meeting <https://indico.cern.ch/event/132486>
- Eva Dafonte Pérez, UKOUG 2009 “Worldwide distribution of experimental physics data using Oracle Streams”
- Luca Canali, CERN IT-DB Deployment, Status, Outlook http://canali.web.cern.ch/canali/docs/CERN_IT-DB_deployment_GAIA_Workshop_March2011.pptx
- CERN openlab, <http://cern.ch/openlab/>
- CAP theorem, <http://portal.acm.org/citation.cfm?id=564601>
- ACID, <http://portal.acm.org/citation.cfm?id=291>